Reversible Data Hiding for Color Images Based on Adaptive 3D Prediction-Error Expansion and Double Deep Q-Network

Jie Chang, Guopu Zhu, *Senior Member, IEEE*, Hongli Zhang, *Member, IEEE*, Yicong Zhou, *Senior Member, IEEE*, Xiangyang Luo, and Ligang Wu

***Abstract*— Reversible data hiding (RDH) for color images has attracted increasing attention in recent years. Due to its effective utilization of the correlation between prediction errors, high-dimensional prediction-error expansion (PEE) can achieve much better performance for color image RDH than low-dimensional PEE. However, existing studies only focus on high-dimensional PEE with nonadaptive embedding. To further improve the embedding performance for color images, we propose a novel three-dimensional PEE method that is adaptive to image content. Double deep Q-network (DDQN), introduced to RDH for the first time, is adopted to find the optimal mapping paths for PEE. In addition, an action selection scheme is presented for DDQN to efficiently find the reversible mapping paths. Extensive experiments show that the proposed method outperforms existing color image RDH methods in image quality.**

Manuscript received 22 October 2021; revised 20 December 2021 and 18 January 2022; accepted 22 January 2022. Date of publication 26 January 2022; date of current version 4 August 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1406902; in part by the National Natural Science Foundation of China under Grant 62172402, Grant 62033005, Grant U1804263, Grant 62172435, and Grant 61872350; in part by the Natural Science Foundation of Heilongjiang Province under Grant ZD2021F001; in part by the Zhongyuan Science and Technology Innovation Leading Talent Project of China under Grant 214200510019; in part by the Tip-Top Scientific and Technical Innovative Youth Talents of Guangdong Special Support Program under Grant 2019TQ05X696; in part by the Basic Research Program of Shenzhen under Grant JCYJ20170818163403748; and in part by the Key Science and Technology (S&T) Special Project of the Autonomous Region under Grant 2020A03004-4. This article was recommended by Associate Editor S. Zhu. *(Corresponding author: Guopu Zhu.)*

Jie Chang is with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: jie.chang@siat.ac.cn).

Guopu Zhu is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China, and also with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: guopu.zhu@hit.edu.cn).

Hongli Zhang is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China (e-mail: zhanghongli@hit.edu.cn).

Yicong Zhou is with the Department of Computer and Information Science, University of Macau, Macau, China (e-mail: yicongzhou@um.edu.mo).

Xiangyang Luo is with the State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China, and also with the Key Laboratory of Cyberspace Situation Awareness of Henan Province, Zhengzhou 450001, China (e-mail: luoxy_ieu@sina.com).

Ligang Wu is with the Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, China (e-mail: ligangwu@hit.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2022.3146517.

Digital Object Identifier 10.1109/TCSVT.2022.3146517

***Index Terms*— Reversible data hiding, color images, double deep Q-network, prediction-error expansion.**

I. Introduction

DATA hiding is a promising technique that hides secret information in the original image to protect multimedia content and copyright [1], [2]. Except for the authorized senders and receivers, no one can judge by eye whether the stego image contains secret information. Steganography, digital watermarking and reversible data hiding (RDH) are the three most common types of data hiding techniques. Steganography focuses on the undetectability of secret information [3], and the main purpose of digital watermarking is to protect the intellectual property rights of digital content [4]. However, both of them irreversibly change the original image. Unlike steganography and digital watermarking, RDH can not only accurately extract the secret message from the stego image, but also recover the original image in a lossless way. Due to the lossless recovery of original images, RDH has been widely applied to sensitive images, such as medical images and military images, whose original versions are too important to be damaged [5]. In addition to copyright protection and content authentication, RDH can also be applied to reversible steganography, reversible image processing and reversible adversarial example [6]. In practice, color images have become the most popular media for knowledge dissemination and information sharing because of their intuitive content and rich expressiveness. So, it is important to study RDH for color images.

Extensive studies have been conducted on RDH methods, which mainly include lossless compression (LC)-based RDH [1], [7], histogram shifting (HS)-based RDH [8], [9], and difference expansion (DE)-based RDH [2], [10]. Fridrich *et al.* [1] proposed to embed secret data into free space generated by lossless compression. However, due to limited compression space, the embedding capacity of the LC-based RDH method is small. The HS-based RDH methods, first proposed by Ni *et al.* [8], embed data into an image by expanding or shifting the histogram of pixel values. Their maximum embedding capacities are determined by the height of the peak bin of the histogram. In [2], the secret data is embedded by expanding the difference between two

1051-8215 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

adjacent pixels, and the maximum embedding rate is only 0.5 bpp. Thodi and Rodriguez [10] proposed to expand the difference between the original pixel and predicted values rather than adjacent pixels to increase the embedding rate. The prediction error expansion (PEE)-based RDH method can achieve high image quality. Then, based on PEE, many methods have been presented by improving prediction accuracy [11]–[14] or embedding mechanism [15]–[18]. In [13], Chen *et al.* proposed directionally enclosed prediction (DEP), which can achieve a sharp prediction-error histogram. According to the complexity of a cover image, Li *et al.* [15] generated a sequence of prediction-error histograms (PEHs), and embedded data into the generated multiple histograms with adaptively selected expansion bins. Instead of selecting a single pair of expansion bins in each histogram, Qi *et al.* [16] utilized multiple pairs of expansion bins in each PEH for expansion embedding. In [18], Li *et al.* divided the image to be embedded into nonoverlapped blocks, and then introduced a multiple-layered embedding approach to embed data into each block.

To further exploit image redundancy for RDH, several methods have been presented based on two and three-dimensional histogram modification. By combining every two adjacent prediction errors as a unit, Ou *et al.* [19] proposed a paired PEE method, which modifies the generated two-dimensional (2D) prediction error based on an empirically designed modification mapping. Recently, Zhang *et al.* [20] adaptively selected a 2D mapping according to image content for data hiding, which can achieve better embedding performance compared with Ou *et al.*'s mapping. By dividing the image into nonoverlapped pixel triples, the authors in [21] and [22] obtained three-dimensional (3D) histograms, and then embedded data into each pixel triple with specially designed shifting and embedding schemes.

Color images are much more widely used than gray images due to their powerful visual expression. However, most of the existing RDH methods were designed for gray images. Although these methods can be directly applied to each channel of a color image, their image quality is poor because of ignoring the correlations among the three color channels. Thus, some studies have been conducted to exploit the correlations of the color channels to enhance image quality. Li *et al.* [23] found that adjacent channels have similar edge information, and therefore they extracted edge information from adjacent channels to predict the current channel. This method can obtain a high prediction accuracy, which leads to a significant increase in embedding capacity. To help the current channel select prediction errors, Ou *et al.* [24] utilized the other two adjacent channels as auxiliary information, which can decrease the length of the prediction error sequence to be embedded and thus reduce the embedding distortion. Recently, Hou *et al.* [25] proposed an RDH method by keeping the grayscale values of color images invariant after data hiding. Unlike the above methods, the embedding of this method does not affect the feature extraction on the gray version of color images. However, the requirement of grayscale invariance has a negative impact on image quality. In [26], Zhan *et al.* explored the correlation of prediction errors among the three channels of color images and proposed an RDH method based on 3D histogram shifting. This method is superior to existing RDH methods in color image embedding, but it embeds data based on empirically designed 3D mappings without considering image content.

In a 3D histogram, each triple has eight modification directions, which typically generate multiple mapping paths. As the number of triples increases, the kinds of mapping paths increase exponentially. It is challenging to enumerate all the mapping paths if the number of triples is large. In addition, to ensure the reversibility of 3D mapping paths, the mapping paths of the triples cannot have an intersection. The process of searching reversible mapping paths can be modeled as a Markov decision process, and reinforcement learning [27]–[31] can effectively learn the optimal decision of the Markov decision process. As a representative algorithm of reinforcement learning, double deep Q-network (DDQN) [32] is effective to deal with large state spaces and has been successfully applied in Atari games [32], recommendation systems [33], financial trading [34], autonomous driving [35], etc. In this paper, we propose a novel reversible data hiding method for color images. The main contributions of this paper can be summarized as follows.

1) Considering the trade-off between computational complexity and image quality, an adaptive 3D prediction-error expansion method is proposed. The most frequent triples in the 3D-PEH are employed to embed data with adaptive mapping, and the others are utilized to embed data with nonadaptive mapping.
2) Double deep Q-network, introduced to RDH for the first time, is adopted to find the optimal mapping for the adaptive mapping. An action selection scheme is also proposed to efficiently find reversible mapping paths. Moreover, the reward function is designed by considering both image quality and capacity.
3) Extensive experiments have been conducted to demonstrate the effectiveness of the proposed method. The experimental results show that the proposed method is superior to existing color image RDH methods in terms of image quality.

The remainder of this paper is organized as follows. The pairwise PEE method and the DDQN method are introduced in Section II. Section III describes the proposed RDH method in detail. The performance evaluation of the proposed method is reported in Section IV. Finally, Section V draws a conclusion.

II. Preliminaries

In this section, we briefly introduce the pairwise prediction-error expansion [19] for gray images and double deep Q-network [32].

A. Pairwise PEE

In [19], Ou *et al.* proposed a pairwise PEE to effectively exploit image redundancy for RDH. Given a cover image, the pairwise PEE method can be described as follows. First, based on a checkerboard-like pattern, the cover image is divided into two disjoined pixel sets, named blank pixel set and shadow pixel set. Since the embedding procedures are

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

similar for the two sets, we take blank pixel set as an example to introduce them. Second, rhombus prediction is adopted to predict each pixel x_i as x_i' according to a raster scan order. The prediction error e_i is calculated as $e_i = x_i - x_i'$. In general, the prediction errors with close distances have a greater correlation. Therefore, the pixel x_i and its bottom-right pixel x_j are joined to construct a prediction-error pair (x_i, x_j). By counting the frequencies of prediction-error pairs, a two-dimensional prediction-error histogram is generated.

For 2D PEE, each quadrant of its 2D embedding space has a similar mapping scheme with others, and here the first quadrant is used to describe the mapping scheme. The prediction error pairs of the first quadrant are divided into four types (A, B, C, and D), and the mapping manners of the four types are listed below.

Type A: $(e_i, e_j) = (0, 0)$. The pair $(0, 0)$ can be extended as $(0, 0)$, $(0, 1)$, $(1, 0)$, and can thus be embedded with $\log_2(3)$ bits.

Type B: $(e_i, e_j) = (1, 1)$. The pair $(1, 1)$ can be extended as $(1, 1)$, $(2, 2)$ by embedding 1 bit of data.

Type C: $(e_i \geq 1, e_j = 0)$ or $(e_i = 0, e_j \geq 1)$. One bit $b \in (0, 1)$ is embedded into (e_i, e_j), and the marked pairs are modified as:

$$(e_i', e_j') = \begin{cases} (e_i + b, e_j + 1), & \text{if } (e_i = 0, e_j \geq 1) \\ (e_i + 1, e_j + b), & \text{if } (e_i \geq 1, e_j = 0) \end{cases}.$$

Type D: $(e_i \geq 1, e_j \geq 2)$ or $(e_i \geq 2, e_j \geq 1)$. The prediction-error pairs are shifted as $(e_i + 1, e_j + 1)$ to vacate room for data hiding.

The pairwise PEE method can attain better embedding performance than the one-dimensional PEE. However, its mapping path is the same for different images without considering image content.

B. Double Deep Q-Network

Reinforcement learning is a semi-supervised learning technique, of which the agent uses the trial and error mechanism to interact with the environment and learns the optimal strategy without a priori knowledge of the environment model. Thus, reinforcement learning has been widely used to solve optimization problems. As a representative reinforcement learning method, deep Q-network (DQN) [32] applies a deep neural network (DNN) to evaluate its action values $Q(S_t, A_c; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the parameters of the deep network, S_t denotes the state of the agent at each time slot t, and A_c denotes the set of all actions. With the output values of the deep network, *i. e.*, $Q(S_t, A_c; \boldsymbol{\theta})$, the agent chooses the best action $a_t \in A_c$ using the epsilon-greedy strategy. Then, the agent receives a reward r_t from the environment and transits to the next state S_{t+1}. To accurately estimate the action value $Q(S_t, a_t; \boldsymbol{\theta})$, a loss function is applied to train current Q-network parameters $\boldsymbol{\theta}$, which is expressed by

$$L = (y_t - Q(S_t, a_t; \boldsymbol{\theta}))^2, \tag{1}$$

where y_t, calculated as $r_t + \gamma \max_a Q(S_{t+1}, a; \boldsymbol{\theta}')$, is the target value of the target Q-network, $\boldsymbol{\theta}'$ is the parameters of the target Q-network, and γ is the distortion factor. Note that the target Q-network has the same structure as the current Q-network, and $\boldsymbol{\theta}'$ is updated by copying the parameters $\boldsymbol{\theta}$ of the current Q-network at a regular interval. Experience replay is introduced to improve neural network efficiency, and is applied to store (S_t, a_t, r_t, S_{t+1}) at each step. Besides, the stored data are randomly sampled to disrupt the correlation between samples.

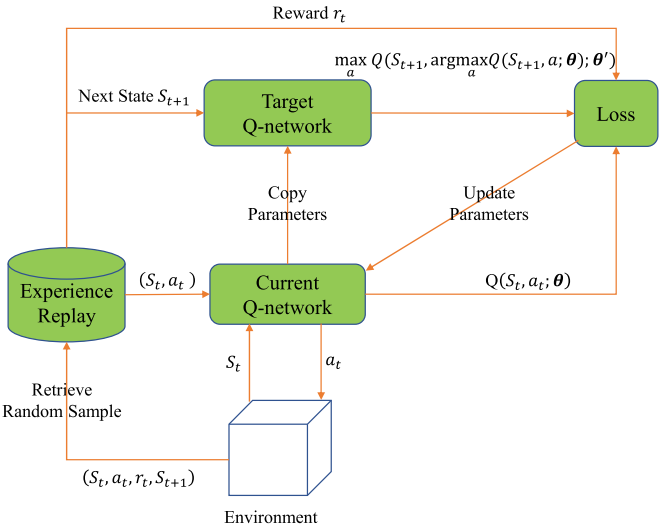


Fig. 1. DDQN framework.

However, DQN adopts the same network to select and evaluate an action, which results in overestimated evaluation. Double deep Q-network can solve the overestimated problem by utilizing the current Q-network to select the best action and evaluating Q values with the target Q-network. Thus, the target value of DDQN is updated as:

$$y_t = r_t + \gamma \max_a Q(S_{t+1}, \arg\max_a Q(S_{t+1}, a; \boldsymbol{\theta}); \boldsymbol{\theta}'). \tag{2}$$

As shown in Fig. 1, at each time slot t, the agent applies the current Q-network to evaluate its action values according to the current state S_t. The best action a_t is selected using the epsilon-greedy strategy. Then, the agent receives a reward r_t from the environment, and its state changes from S_t to the next state S_{t+1}. The transmissions (S_t, a_t, r_t, S_{t+1}) are stored in an experience replay to improve the performance of the current Q-network. The target Q-network, which is used to calculate a target Q value, has the same structure as the current Q-network. The parameters $\boldsymbol{\theta}'$ of the target Q-network are periodically copied from the current Q-network. The current Q-network is trained by minimizing the loss function formulated by Eq. (1), and the parameters $\boldsymbol{\theta}$ are updated based on a gradient descent algorithm. Based on the success of DNNs, DDQN is capable of solving high-correlation and nonconvex problems.

III. Proposed Method

To exploit the correlation of RGB channels, Zhan *et al.* [26] recently proposed an effective 3D PEE reversible data hiding method for color images. Unfortunately, the mapping scheme

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

based on 3D-PEH is empirically designed without considering image content. To further improve the embedding capability, this section proposes a 3D PEE with adaptive mapping for color images. In a 3D-PEH, the frequently occurring triples have a stronger impact on the image quality compared to other ones. Considering the trade-off between computational complexity and image quality, we embed data into the frequently occurring triples with adaptive mapping according to image content, and into the other ones with nonadaptive mapping.

A. 3D Prediction-Error Histogram

In a color image, the pixels of each channel are divided into two nonoverlapped sets, named blank pixel set and shadow pixel set. A pixel x_k^i $(i \in (r, g, b), k = 1, \cdots, N)$ is predicted as $x_k^{i\prime}$ based on directionally enclosed prediction, and the prediction error e_k^i is calculated as $e_k^i = x_k^i - x_k^{i\prime}$. The prediction errors of RGB channels with the same location are taken as a unit to generate a sequence $(\boldsymbol{e}_1, \cdots, \boldsymbol{e}_N)$, where $\boldsymbol{e}_k = (e_k^r, e_k^g, e_k^b)$. The 3D-PEH $g(x, y, z)$ is constructed by counting the frequencies of 3D prediction-error triples, *i. e.*,

$$g(x, y, z) = \#\{1 \le k \le N : e_k^r = x, e_k^g = y, e_k^b = z\},$$

where # denotes the cardinal number of a set.

The 3D-PEH can be divided into 8 quadrants. The central bin of each quadrant is the triple with the highest frequency in the quadrant. In each quadrant, the central triple and its adjacent triples are used to adaptively embed data. The central bins from the first to the eighth quadrant are (0, 0, 0), (−1, 0, 0), (−1, −1, 0), (0, −1, 0), (0, 0, −1), (−1, 0, −1), (−1, −1, −1) and (0, -1, -1), respectively. Fig. 2 shows the frequencies of 3D-PEH triples for Airplane and Tiffany images, and one can observe that the majority of prediction-error triples are around (0, 0, 0), *i. e.*, $(-3 \le e_i^r < 2, -3 \le e_i^g < 2, -3 \le e_i^b < 2)$, and are adopted to embed data with adaptive mapping.

B. 3D-PEH-Based Mapping

All quadrants have a similar embedding way. In the following, the first quadrant $(\boldsymbol{e}_k \ge 0)$ is taken as an example to introduce the proposed embedding process. Fig. 3 shows the mapping of the proposed method. The mapping paths in the red box $(0 \le e_i^r < 2, 0 \le e_i^g < 2, 0 \le e_i^b < 2)$ are adaptively chosen according to image content. Then, the 3D bins in the first quadrant are classified into five types as $A - E$ (refer to Fig. 3). Inspired by [20] and [26], the mapping manners for the five types are described as follows:

Type A: $(0 \le e_i^r < 2, 0 \le e_i^g < 2, 0 \le e_i^b < 2)$. The mapping paths of these triples are adaptively selected using DDQN. The details of this mapping are presented in Section III-C.

Type B: $(e_i^r \ge 2, 0, 0)$ or $(0, e_i^g \ge 2, 0)$ or $(0, 0, e_i^b \ge 2)$. The triples are chosen to carry $\log_2(3)$ bit data with a distortion of 5/3. Specifically, $(e_i^r \ge 2, 0, 0)$ is used as an example, and the embedding rule is expressed by:

$$(e_i^{r\prime}, e_i^{g\prime}, e_i^{b\prime}) = \begin{cases} (e_i^r + 1, 0, 0), & \text{if } (b_1 = 0, b_2 = 0) \\ (e_i^r + 1, 0, 1), & \text{if } (b_1 = 0, b_2 = 1) \\ (e_i^r + 1, 1, 0), & \text{if } (b_1 = 1, b_2 = 0) \end{cases}, \quad (3)$$

where b_1 and b_2 denote the embedded bits.

Type C: $(e_i^r \ge 2, 1, 1)$ or $(1, e_i^g \ge 2, 1)$ or $(1, 1, e_i^b \ge 2)$. Only one-bit data is embedded into the triples with a distortion of 2. For example, the embedding rule for $(e_i^r \ge 2, 1, 1)$ is given by:

$$(e_i^{r\prime}, e_i^{g\prime}, e_i^{b\prime}) = \begin{cases} (e_i^r + 1, 1, 1), & \text{if } b = 0 \\ (e_i^r + 1, 2, 2), & \text{if } b = 1 \end{cases}, \quad (4)$$

where $b \in (0, 1)$ is the embedded bit.

Type D: $(e_i^r \ge 2, e_i^g \ge 1, 0)$ or $(e_i^r \ge 1, e_i^g \ge 2, 0)$ or $(e_i^r \ge 1, 0, e_i^b \ge 2)$ or $(0, e_i^g \ge 1, e_i^b \ge 2)$ or $(0, e_i^g \ge 2, e_i^b \ge 1)$ or $(e_i^r \ge 2, 0, e_i^b \ge 1)$. Only one element of type D is applied to embed data $b \in (0, 1)$, and the other two elements are shifted. Specifically, we consider $(e_i^r \ge 2, e_i^g \ge 1, 0)$ as an

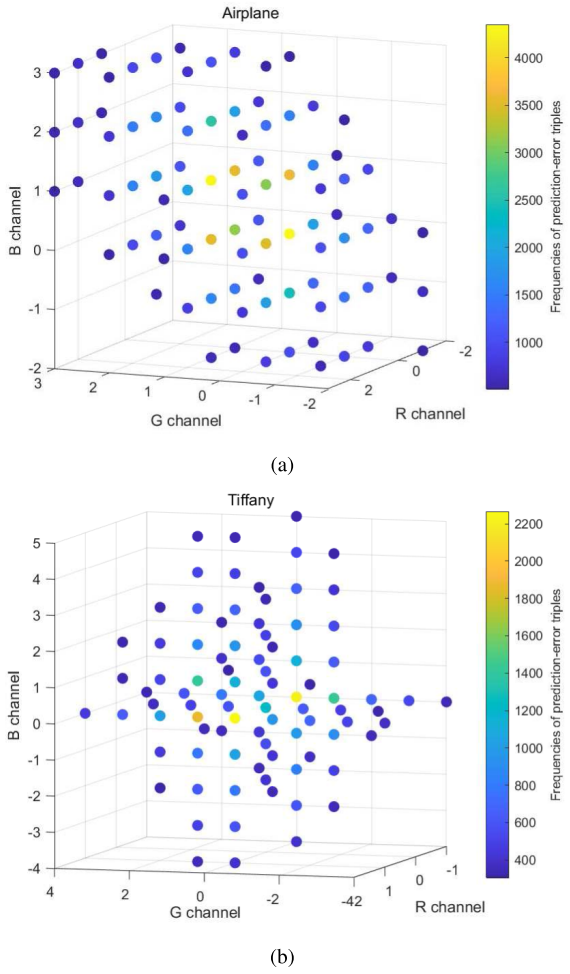


Fig. 2. Frequencies of prediction-error triples for Airplane image (a) and Tiffany image (b), separately.

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

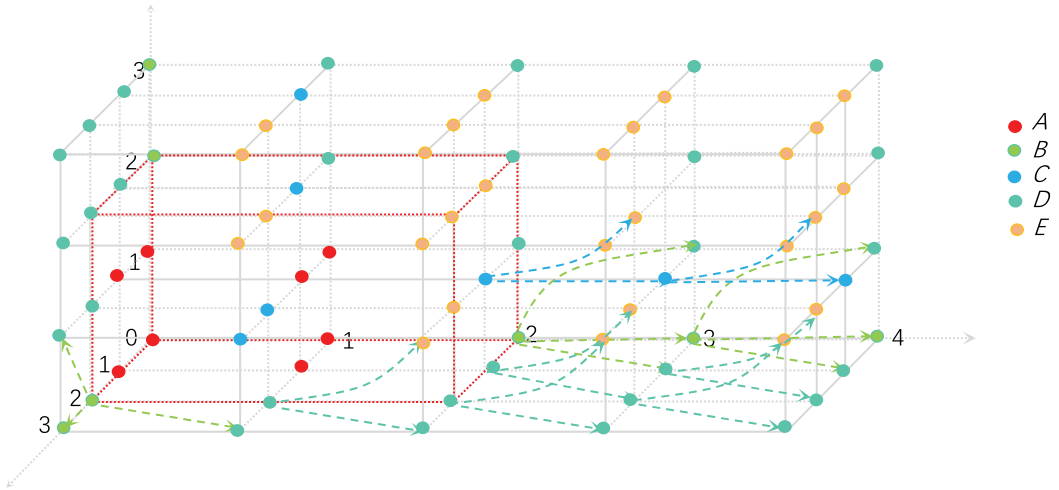


Fig. 3. Mapping of the proposed method, where the mapping paths in the red box are adaptively chosen according to image content and the mapping paths of the other triples are shown by dotted arrows.

example to illustrate the embedding manner, which is described by:

$$(e_i^{r\prime}, e_i^{g\prime}, e_i^{b\prime}) = \begin{cases} (e_i^r + 1, e_i^g + 1, 0), & \text{if} \quad b = 0 \\ (e_i^r + 1, e_i^g + 1, 1), & \text{if} \quad b = 1 \end{cases}. \quad (5)$$

Type E: $(e_i^r \geq 2, e_i^g \geq 2, e_i^b \geq 1)$ or $(e_i^r \geq 2, e_i^g \geq 1, e_i^b \geq 2)$ or $(e_i^r \geq 1, e_i^g \geq 2, e_i^b \geq 2)$. All elements are shifted to release space for other triples to embed data.

According to the mapping manner presented above, the embedding capacity EC and distortion ED of an image can be calculated, respectively, as:

$$EC = c_A \sum_{\boldsymbol{e}\in A} g(\boldsymbol{e}) + \log_2(3) \sum_{\boldsymbol{e}\in B} g(\boldsymbol{e}) + \sum_{\boldsymbol{e}\in C} g(\boldsymbol{e}) + \sum_{\boldsymbol{e}\in D} g(\boldsymbol{e}) \quad (6)$$

and

$$ED = d_A \sum_{\boldsymbol{e}\in A} g(\boldsymbol{e}) + \frac{5}{3} \sum_{\boldsymbol{e}\in B} g(\boldsymbol{e}) + 2 \sum_{\boldsymbol{e}\in C} g(\boldsymbol{e}) + 2.5 \sum_{\boldsymbol{e}\in D} g(\boldsymbol{e}) + 3 \sum_{\boldsymbol{e}\in E} g(\boldsymbol{e}), \quad (7)$$

where c_A and d_A denote the capacity and the distortion of type A, respectively, and $g(\boldsymbol{e})$ denotes the frequency of the prediction-error triple $\boldsymbol{e}$.

C. Adaptive Mapping Constructed by DDQN

Generally, a prediction error e_i is expanded to carry one-bit data with modification directions e_i and $e_i + 1$, and its average distortion is 0.5 bit. Thus, a prediction-error triple (e_k^r, e_k^g, e_k^b) can be applied to carry three bits of data, and its mapping path is $\boldsymbol{P} = (\boldsymbol{p}_{\sigma(1)}, \boldsymbol{p}_{\sigma(2)}, \boldsymbol{p}_{\sigma(3)}, \cdots, \boldsymbol{p}_{\sigma(7)}, \boldsymbol{p}_{\sigma(8)})$, which contains eight modification directions. Nevertheless, the expandable prediction-error triple introduces a large distortion of 1.5 bits. Each prediction-error triple can select one or more modification directions to construct the mapping path by considering the trade-off between embedding capacity and distortion. By the principle of permutation and combination, there are 255 $(C_8^1 + C_8^2 + \cdots + C_8^8 = 255)$ kinds of mapping paths for a triple to select. To ensure embedding reversibility, the mapping paths $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ of any two prediction error triples $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$ should be disjoint, namely $\boldsymbol{p}_i \cap \boldsymbol{p}_j = \emptyset$. For convenience, the eight triples in a quarter are sorted in ascending order as $(\boldsymbol{e}_{\sigma(1)}, \boldsymbol{e}_{\sigma(2)}, \boldsymbol{e}_{\sigma(3)}, \cdots, \boldsymbol{e}_{\sigma(7)}, \boldsymbol{e}_{\sigma(8)})$. Specifically, the first elements of the triples are sorted in ascending order firstly; if any two triples have the same first or first two elements, then their second or the third elements are sorted in the same way. For each quarter, the mapping paths of the eight triples in type A construct a group of reversible mapping path $\boldsymbol{P} = (\boldsymbol{p}_{\sigma(1)}, \boldsymbol{p}_{\sigma(2)}, \boldsymbol{p}_{\sigma(3)}, \cdots, \boldsymbol{p}_{\sigma(7)}, \boldsymbol{p}_{\sigma(8)})$. The process of searching $\boldsymbol{P}$ can be modeled as a Markov decision process. Dynamic programming is one of the most effective methods to solve the Markov decision process, however, it requires the knowledge of the transition matrix among triples. Without considering reversibility, $\boldsymbol{P}$ has $255^8 \approx 1.788 \times 10^{19}$ kinds of mapping paths.

Even if all kinds of the mapping paths of $\boldsymbol{P}$ are reduced by half to ensure reversibility, it remains difficult to enumerate them to compute the transition probability. In the following, we apply DDQN to find optimal mapping paths.

1) DDQN Structure for RDH: For reinforcement learning, its agent observes the state of environment, and then chooses an action to obtain the maximum cumulative reward value. For example, a chess agent chooses a move to win according to the current state of the chess game. Reinforcement learning can effectively learn the optimal decision of the Markov decision process. When the number of states and actions is relatively small, the classical Q-learning method can be applied to enumerate all Q-values for all possible action-state pairs. However, for large action-state spaces, neural networks are needed to approximate the values of Q function. DDQN is a representative algorithm using neural networks to estimate Q function, and it has been successfully applied to Atari games and recommendation systems. Thus, this paper employs DDQN to find the optimal path with large action-state

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

spaces. The general structure of the DDQN model is presented below.

a) Environment state: Each triple has eight modification directions, which can be selected to form a mapping path. For convenience, the selected modification directions are marked as 1, and the others are marked as 0. Thus, the mapping path of $\boldsymbol{e}_{\sigma(1)}$ is initialized as $\boldsymbol{p}_{\sigma(1)} = (0, 0, 0, 0, 0, 0, 0, 0)$, and it is regarded as the state $(s_1, s_2, s_3, \cdots, s_8)$ of the triple $\boldsymbol{e}_{\sigma(1)}$.

In each quadrant, we need to find reversible embedding paths for eight triples. The state $\boldsymbol{S} = (s_1, s_2, s_3, \cdots, s_{64})$ of each quadrant represents the reversible mapping path $\boldsymbol{P}$, and $\boldsymbol{S}$ is initialized as $(0, 0, \cdots, 0)$.

b) Agent's action: Based on the analysis of Section III-C, each triple has 255 kinds of mapping paths. Each path represents an action, and then the set of all actions is denoted as $A_c = (a_1, a_2, \cdots, a_{255})$.

c) Reward function: The goal of DDQN is to find an optimal mapping path according to the obtained maximum reward value. The calculation of the reward value is very important for DDQN. Most of the existing RDH methods focus on embedding more data into an image with lower distortion. Thus, the reward $\hat{r}$ should be related to the embedding capacity cap and distortion dis for every triple. The mapping path $\boldsymbol{p}_{\sigma(i)}$ of the triple $\boldsymbol{e}_{\sigma(i)}$ contains n modification directions, and the embedding capacity and distortion are calculated as $cap = \log_2(n)$ and $dis = (\text{sum}(\boldsymbol{p}_{\sigma(i)}) - (e^r_{\sigma(i)}, e^g_{\sigma(i)}, e^b_{\sigma(i)}) \times n)/n$, respectively.

According to whether cap and dis are equal to 0 or not, the calculation of the reward $\hat{r}$ can be divided into three cases.

Case 1: Both cap and dis are nonzero. The optimal embedding for RDH is generally determined with the minimum distortion rate $rate = dis/cap$. Whereas, the goal of DDQN is to achieve the maximum reward value. Thus, the reciprocal of the distortion rate, namely $\hat{r} = cap/dis$, is taken as the optimization goal of DDQN. It is easy to obtain that the value range of $\hat{r}$ is [0.4, 2.67].

Case 2: Both cap and dis are 0. The current triple is not utilized for expanding or shifting. In this case, the value of $\hat{r}$ is meaningless. The reward value should be set below 0.4 to avoid the waste of pixel resources. Here we set the reward value to 0.1.

Case 3: cap is zero, while dis is nonzero. This case only increases the embedding distortion, but does not increase the embedding capacity. This process results in poor image quality of the stego image. The reward of this case should be a small value. Thus, we set it to 0.01.

Since the number of prediction errors in each quadrant is different from each other, different quadrants apply their own optimal embedding paths to embed data for high image quality. While, all the quadrants select their optimal mapping paths in the same way. Here, the first quadrant is taken as an example to show the objective function, which is used to find the optimal mapping path, and is given by

$$\max\left(\sum_{i=1}^{8} g(\boldsymbol{e}_{\sigma(i)}) \times \hat{r_i}\right), \tag{8}$$

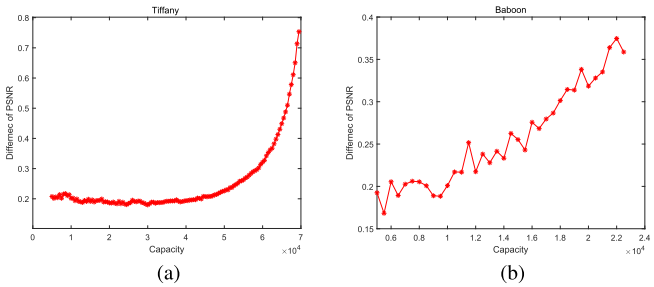


Fig. 4. PSNR differences between the method with the modified reward r and that with the reward $\hat{r}$ for Tiffany image (a) and Baboon image (b), separately.

where $g(\boldsymbol{e}_{\sigma(i)})$ is the frequency of the i-th prediction-error triple of type A, and $\hat{r_i}$ is the corresponding reward.

The value of $\hat{r}$ for low capacity is known to be higher than that for a high capacity. The optimal embedding path selected only according to the maximum reward is suitable for a low embedding capacity but not for a high embedding capacity. However, we cannot know the embedding capacity of each quadrant in advance. Considering the optimization goal of DDQN and the embedding capacity, the reward of Case 1 is modified as $r = cap/dis + cap$, which can make the selected mapping path $\boldsymbol{P}$ suitable for both low and high embedding capacity. To verify that the modification of the reward is necessary, we select the optimal mapping paths based on the rewards before and after modification, separately. Data is embedded into an image with the selected optimal mapping paths. Then, the peak-signal-to-noise ratio (PSNR) values of the stego images are compared. Fig. 4 shows the difference of PSNRs between the method with the modified reward and that with the original reward. The embedding capacity increases from 5,000 bits to their maximum with a step size of 500. It is shown from Fig. 4 that the difference increases with embedding capacity. This means that the modified reward can improve image quality, especially for high embedding capacity.

In summary, the reward function described above is designed as follows:

$$r = \begin{cases} 0.1, & \text{if } \ cap = 0, dis = 0 \\ 0.01, & \text{if } \ cap = 0, dis \neq 0 \\ \frac{cap}{dis} + cap, & \text{if } \ cap \neq 0, dis \neq 0. \end{cases} \tag{9}$$

Note that the case of $cap \neq 0$ and $dis = 0$ does not exist.

2) Optimal Path Search Based on DDQN: To ensure the reversibility of the mapping paths of all triples, the modification directions of any two triples cannot have an intersection. Specifically, a triple has eight modification directions, which construct 255 kinds of mapping paths. If three modification directions of the current triple have been selected another triple, the kinds of reversible mapping paths for the current triple decrease to 31 ($C_5^1 + C_5^2 + \cdots + C_5^5 = 31$). That is, available mapping paths are reduced by 87.84%. In a general way, DDQN chooses an action according to the maximum Q value, and then judges whether the mapping path corresponding to the selected action is reversible or not. If the mapping path is irreversible, the agent terminates the search of this

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

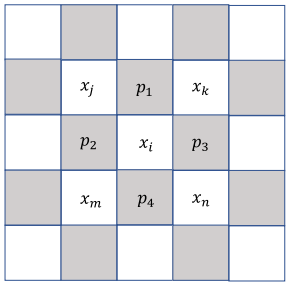


Fig. 5. Shadow and blank pixel partition.

mapping path, and researches the optimal mapping path from the beginning. This process will waste a lot of time to find a reversible mapping path $\boldsymbol{P}$.

To address the above issue caused by the requirement of reversibility, here we present an action selection scheme for DDQN to quickly find a reversible mapping path. Before the description of the selection scheme, we introduce two variables $h(a_t, \boldsymbol{e}_{\sigma(i)})$ and sd, where $h(a_t, \boldsymbol{e}_{\sigma(i)})$ denotes the modification directions of the triple $\boldsymbol{e}_{\sigma(i)}$ after selecting action a_t, and sd is used to store the already selected modification directions. Instead of selecting action first and then judging the reversibility, we first find a reversible mapping path, and then select the corresponding action. The detailed procedure is as follows. Firstly, for the current triple $\boldsymbol{e}_{\sigma(i)}$, we select action a_i such that $Q(S_t, a_i; \boldsymbol{\theta})$ is the maximum Q value, and hence the modification directions are $h(a_i, \boldsymbol{e}_{\sigma(i)})$. Secondly, we judge whether $h(a_i, \boldsymbol{e}_{\sigma(i)})$ and sd have an intersection. If there is no intersection, the action a_i will be formally selected, and the modification directions $h(a_i, \boldsymbol{e}_{\sigma(i)})$ will be added to sd. Whereas, if there is an intersection, we select action a_j that has the second maximum Q value, and judge whether $h(a_j, \boldsymbol{e}_{\sigma(i)})$ and sd have an intersection. This process is repeated until finding the action whose modification direction does not intersect with sd. The detailed method of finding the optimal mapping path with DDQN is illustrated in Algorithm 1. It mainly consists of four phases: Firstly, the parameters of DDQN are initialized. Secondly, based on the ξ greedy policy, the action with a reversible mapping path is selected. Specifically, at time t, the agent interacts with the environment according to the current state S_t and obtains the Q values of all actions. To quickly find a reversible mapping path, the Q values are firstly sorted in descending order or random order. Then, from the first to the last Q value, the action a_t with the first-appearing reversible mapping path is selected. Thirdly, the current state S_t, the next state S_{t+1}, the executed action a_t, and the obtained reward r_t are stored in experience replay D_e. Finally, the current Q-network parameters are updated by randomly sampling a minibatch replay memory.

Algorithm 1 Searching the Optimal Path With DDQN

Input : Episode T, action set A_c, learning rate α, exploration rate ξ, minibatch size k, discount factor γ, network update period F, experience replay D_e with capacity N_c

Output: Optimal paths for all triples

Initialization: Empty replay memory D_e, initialize current Q-network (Q) with random weights $\boldsymbol{\theta}$, initialize target Q-network (Q') with weight $\boldsymbol{\theta'}$, initialize already selected direction $sd = [\ ]$
for *episode* $\leftarrow 1$ **to** T **do**
  for $t \leftarrow 1$ **to** 8 **do**
    Initialize state S
    `// Select action` a_t `based on` ξ `greedy policy`
    if random(1) $< \xi$ **then**
      Sort $Q(S, A_c; \boldsymbol{\theta})$ as $\hat{Q}(S, A_c; \boldsymbol{\theta})$ in random order
    else
      ;
      Sort $Q(S, A_c; \boldsymbol{\theta})$ as $\hat{Q}(S, A_c; \boldsymbol{\theta})$ in descending order
    $num = 1$
    while True **do**
      Select a_i with $\hat{Q}(num)$
      $md = h(a_i, \boldsymbol{e}_{\sigma(t)})$
      if $md \cap sd = \emptyset$ **then**
        $a_t = a_i$
        break
      else
        ;
        $num = num + 1$
    $sd = sd \cup md$
    `// Store transition in the experience replay` D_e
    Execute action a_t, and observe the reward r_t and the next state S_{t+1}
    Set $S_t = S_{t+1}$, and store transition (S_t, a_t, r_t, S_{t+1}) in D_e
    `// Update the current Q-network parameters`
    if D_e is full **then**
      for $t \leftarrow 1$ **to** k **do**
        Randomly sample a minibatch of transitions (S_t, a_t, r_t, S_{t+1}) from D_e, and compute target Q-network y_t
        if $t == 8$ **then**
          $y_t = r_t$
          else
          ;
          $a = \arg\max_a Q(S_{t+1}, a; \boldsymbol{\theta})$
          $y_j = r_t + \gamma \max_a Q(S_{t+1}, a; \boldsymbol{\theta'})$
        Perform a gradient descent step on $(y_t - Q(S_t, a_t; \boldsymbol{\theta}))^2$ with respect to the network parameters $\boldsymbol{\theta}$
    Reset $Q' = Q$ for every F steps

D. Implementation Details

The cover image is divided into two sets, as shown in Fig. 5. Since the embedding procedures of the two sets are similar, the blank set is taken as an example to show the procedures of data embedding and extraction. To embed more data into smooth regions, we apply the noise level to measure the complexity of a color pixel (x_i^r, x_i^g, x_i^b). For pixel x_i, the local noise level is given by

$$F_x^i = |p_1 - p_2| + |p_2 - p_4| + |p_3 - p_4| + |p_1 - p_3|, \quad (10)$$

where p_1, p_2, p_3 and p_4 are the nearest four neighbors of pixel x_i. It may not be accurate to measure the complexity of

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

a pixel only based on its nearest four neighbors. Therefore, here the noise levels of adjacent pixels are also applied to calculate the complexity of pixel x_i, which is formulated by

$$LC(x_i) = F_x^i + \left\lfloor \frac{F_x^j + F_x^k + F_x^m + F_x^n}{4} \right\rfloor, \tag{11}$$

where F_x^j, F_x^k, F_x^m and F_x^n denote the noise levels of x_j, x_k, x_m and x_n, respectively, and $\lfloor \cdot \rfloor$ represents the floor function. Then, the complexity of a color pixel $\boldsymbol{x}_i = (x_i^r, x_i^g, x_i^b)$ is calculated as the sum of the complexity values of x_i^r, x_i^g and x_i^b. In general, it is expected that more smooth regions are utilized to embed data while minimizing the embedding distortion. The lower the complexity value is, the smoother the pixel is. Thus, under the premise of meeting embedding capacity, the smallest complexity value is adopted as a threshold T_C. The pixels whose complexity values are less than T_C are chosen to embed data, while the others are skipped. Note that the threshold T_C is selected according to image content, therefore different images may have different threshold values.

The directionally enclosed prediction has high prediction accuracy compared with other prediction methods, such as median-edge detector (MED), gradient-adjusted prediction (GAP) and rhombus prediction [13]. For DEP, the predicted value of a pixel x_i is obtained by comparing the horizontal and vertical predicted values of pixel x_i. The procedure of DEP can be described below. Firstly, the horizontal and vertical predicted values of pixel x_i are calculated as $x'_{i_h} = \lfloor \frac{p_2+p_3}{2} \rfloor$ and $x'_{i_v} = \lfloor \frac{p_1+p_4}{2} \rfloor$, respectively. Secondly, the horizontal predicted value x'_{i_h} and the vertical predicted value x'_{i_v} are compared with the pixel x_i, and the horizontal and vertical prediction errors are obtained by $e_{i_h} = x_i - x'_{i_h}$ and $e_{i_v} = x_i - x'_{i_v}$, respectively. Finally, the smallest absolute value of e_{i_h} and e_{i_v} is selected as the prediction error e_i of pixel x_i. Hence, the predicted value of pixel x_i is x'_{i_h} or x'_{i_v}. To accurately extract the embedded information, only the prediction errors that satisfy $e_{i_h} \times e_{i_v} \geq 0$ are used for prediction. In summary, the prediction error and predicted value x'_i of pixel x_i can be expressed, respectively, by

$$e_i = \begin{cases} e_{i_h}, & \text{if} \quad e_{i_h} \times e_{i_v} \geq 0 \quad \text{and} \quad |e_{i_h}| \leq |e_{i_v}| \\ e_{i_v}, & \text{if} \quad e_{i_h} \times e_{i_v} \geq 0 \quad \text{and} \quad |e_{i_h}| > |e_{i_v}| \\ \phi, & \text{otherwise} \end{cases} \tag{12}$$

and

$$x'_i = \begin{cases} x'_{i_h}, & \text{if} \quad e_i = e_{i_h} \\ x'_{i_v}, & \text{if} \quad e_i = e_{i_v} \\ \phi, & \text{otherwise,} \end{cases} \tag{13}$$

where ϕ represents that the pixel is skipped without prediction.

Some auxiliary information needs to be stored in the cover image to extract secret data exactly and recover the original image without distortion. The auxiliary information of the proposed method includes the complexity threshold T_C (12 bits), the optimal mapping path (512 bits), the size of the compressed location map (18 bits), and the end embedding position (16 bits). A location map is constructed to avoid overflow/underflow of the cover image after data embedding. Specifically, the pixels with a value of 255 are changed to 254, and those with a value of 0 are changed to 1. Simultaneously, the location map is generated to mark these changed pixels as 1 and the others as 0. Then, the location map is losslessly compressed.

Finally, the procedures of data embedding and extraction for blank pixel set are presented as follows.

1) Embedding Procedure:

Step 1: In the image to be embedded, the pixels with value 0 or 255 are modified to 1 or 254, respectively. Meanwhile, the modified pixels are marked to form a location map, which is then losslessly compressed. The size of the compressed location map is denoted as L_{lp}.

Step 2: The first three rows of the modified image are reserved for storing the auxiliary information. Specifically, the least significant bits (LSBs) of the first 558 pixels are emptied to store auxiliary information. Then, the emptied LSBs and the compressed location map are embedded into the image with half of the secret message.

Step 3: Except for the first three rows of the image, the blank pixels are predicted by directionally enclosed prediction, and then a 3D-PEH is constructed based on the generated prediction errors.

Step 4: The complexity of each triple is calculated, and the pixels with complexity below T_C are chosen to embed data with the adaptive or nonadaptive mapping. After the embedding of the payload, the end position of the embedding process as well as other auxiliary information are embedded into the LSBs of the first 558 pixels.

2) Extraction Procedure:

Step 1: The auxiliary information is extracted by collecting the LSBs of the first 558 pixels of the stego-image.

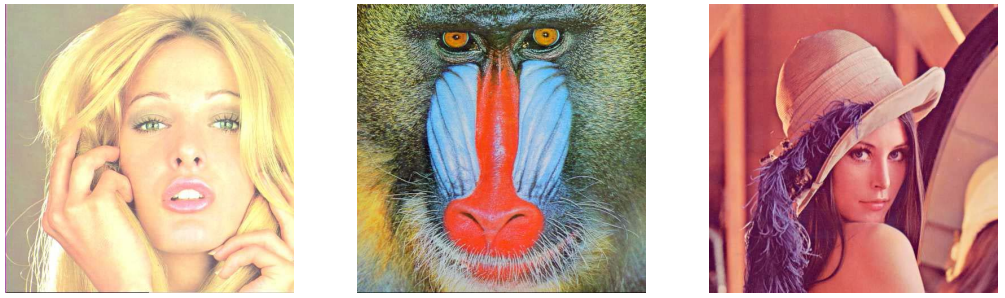
Step 2: Except for the first three rows of the stego-image, the blank pixels in the stego-image are predicted by directionally enclosed prediction. Then, a 3D-PEH is constructed according to Section III-A.

Step 3: For each triple $\boldsymbol{x}'_i$, the complexity is first calculated. Then, we choose the triples that satisfy $LC(\boldsymbol{x}'_i) \leq T_C$ to extract data and to recover pixel values in the reverse order of the data embedding.

Step 4: The extracted data contains half of the secret message, the compressed location map, and the replaced LSBs. The first 558 pixels of the stego-image are recovered with the replaced LSBs. Then, the compressed location map is decompressed to retrieve the pixels marked with 1.

Note that once the data embedding is completed for the blank pixel set, the shadow pixel set embeds the remaining half of the secret message in a similar embedding manner. Besides, since data extraction is the reverse process of data embedding, the extraction of the shadow pixel set must be performed before that of the blank pixel set. The extraction manners of the blank and shadow pixel sets are similar. After the data extraction, the original image is recovered at the end.

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.



(a) Tiffany (b) Baboon (c) Lena

(d) Airplane (e) Lake (f) Peppers

Fig. 6. Six standard images.

IV. Experimental Results

In this section, we conduct several experiments to evaluate the embedding performance of the proposed method. As shown in Fig. 6, the standard color images used for the test are Tiffany, Lena, Baboon, Airplane, Lake and Peppers, each of which has a size of 512 × 512 pixels, and are downloaded from the USC-SIPI database. The proposed method is compared with Li *et al.*'s [23], Ou *et al.*'s [24], Hou *et al.*'s [25] and Zhan *et al.*'s [26] methods.

The first experiment is performed to demonstrate that the approximate optimal embedding path can be found with DDQN. The setting of the DDQN parameters is shown in Table I. In the first quadrant of Lena image, the occurring frequencies of the triples are 1823, 475, 475, 500, 235, 213, 273 and 119 for (0, 0, 0), (0, 0, 1), (0, 1, 0), (1, 0, 0), (1, 1, 0), (1, 0, 1), (0, 1, 1) and (1, 1, 1), respectively, and the reward values of DDQN are shown in Fig. 7. For the embedding capacity of 30,000 bits, the optimal embedding paths found with DDQN and the greedy search method separately are shown in Fig. 8. It is seen that the reward values of the two optimal embedding paths are 15,036 and 15,125 respectively, which are very close to each other. Moreover, for the embedding capacity of 30,000 bits, the PSNR values of the embedding methods with DDQN and the greedy search method are 60.22 dB and 60.29 dB, respectively. The difference between these two PSNRs is only 0.07.

Second, to verify the effectiveness of the action selection scheme, we carry out an experiment to compare the average running time of finding a reversible embedding path $\boldsymbol{P}$ before and after applying the proposed action selection scheme for different test images. The experiment is implemented on Intel Silver-4210 CPU with 4.40 GHz and 128 GB RAM. The comparison results are shown in Table II, where ART_1 and ART_2 represent the average running time of 500 epochs before and after applying the proposed action selection scheme,

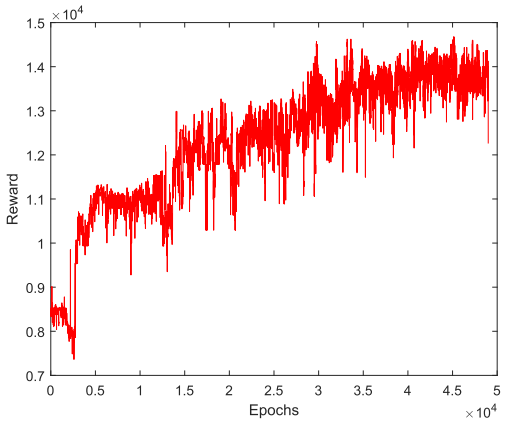
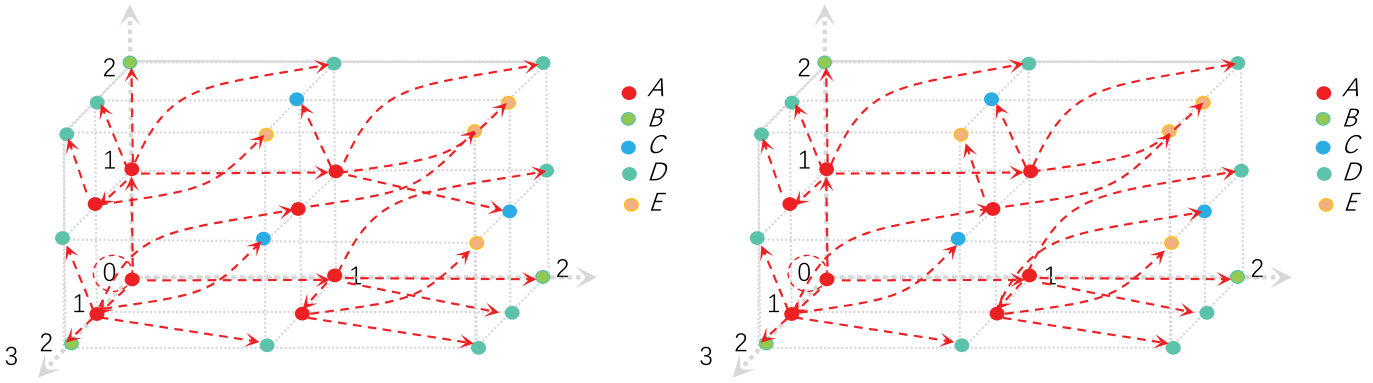


Fig. 7. Reward curve of the training process.

TABLE I
Parameter Setting of DDQN

Parameter	Value
Number of hidden layers	2
Size of fully connected neural network	516 × 516
Activation function	tanh
Optimizer	Adam
Learning rate α	0.001
Discount factor γ	0.75
Exploration rate ξ	0.2
Minibatch size	256
Size of experience replay memory	5,000
Update frequency of target Q-network	200

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.



(a) Constructed by DDQN. (b) Constructed by greedy search method.

Fig. 8. Optimal mapping paths constructed by DDQN and greedy search method, separately. Here, the embedding capacity is set as 30,000 bits.

TABLE II

COMPARISON OF THE AVERAGE RUNNING TIME ($\times 10^{-4}$ sec) ACHIEVED WITH AND WITHOUT THE PROPOSED ACTION SELECTION SCHEME

	Tiffany	Baboon	Lena	Airplane	Lake	Peppers	Averaged
ART_1	51.52	44.69	48.97	50.32	43.51	46.75	47.63
ART_2	9.96	10.00	9.98	9.99	10.00	10.41	10.06

TABLE III

COMPARISON OF THE PSNRS (DB) AMONG THE PROPOSED METHOD, LI *et al.*'S [23], OU *et al.*'S [24], HOU *et al.*'S [25] AND ZHAN *et al.*'S [26] METHODS UNDER EMBEDDING CAPACITY OF 30,000 bits

Images	Li *et al.*	Ou *et al.*	Hou *et al.*	Zhan *et al.*	Proposed
Tiffany	54.60	58.75	48.05	55.43	**59.19**
Baboon	54.40	54.49	42.59	54.80	**56.23**
Lena	57.10	58.78	46.65	58.02	**60.22**
Airplane	61.87	62.76	53.94	61.92	**64.06**
Lake	56.84	58.07	47.04	57.75	**59.83**
Peppers	56.32	55.51	44.11	55.42	**57.49**
Average	56.86	58.06	47.06	57.22	**59.50**

respectively. It can be seen from Table II that for the six test images, the average running time is greatly reduced by 78.88% on the whole after applying the proposed scheme.

Third, PSNR is adopted to test the effectiveness of the proposed method. The embedding capacity of the five test methods varies from 10,000 bits to their capacity limits with a step size of 2,000 bits. Fig. 9 shows the PSNRs of the test methods. As a representative RDH method for color images, Li *et al.* [23] achieved high prediction accuracy by utilizing the correlation between color channels. From Fig. 9, the proposed method outperforms Li *et al.*'s method on every test image. This may be because that the proposed method applies a higher-dimensional PEE method. Ou *et al.* [24] first proposed to divide the payload according to the dependence between color channels. More data is embedded in the smooth channel than in the rough one, which leads to the reduction of embedding distortion. Whereas, it is seen from Fig. 9 that the proposed method achieves better performance than Ou *et al.*'s method for all test images, which is mainly due to the full utilization of the correlations among RGB channels. Hou *et al.* [25] proposed an RDH method for color images,

TABLE IV

COMPARISON OF THE PSNRS (DB) AMONG THE PROPOSED METHOD, LI *et al.*'S [23], OU *et al.*'S [24], HOU *et al.*'S [25] AND ZHAN *et al.*'S [26] METHODS UNDER EMBEDDING CAPACITY OF 50,000 bits

Image	Li *et al.*	Ou *et al.*	Hou *et al.*	Zhan *et al.*	Proposed
kodim01	59.94	60.32	52.57	57.22	**61.74**
kodim02	60.49	60.51	53.06	57.93	**62.51**
kodim03	61.95	62.08	54.97	59.48	**63.05**
kodim04	60.47	61.03	53.98	57.82	**62.22**
kodim05	59.67	59.78	52.05	57.07	**61.08**
kodim06	59.25	60.90	53.21	57.51	**62.92**
kodim07	61.66	61.81	54.81	59.12	**62.87**
kodim08	56.54	58.12	50.28	53.80	**59.05**
kodim09	59.06	60.68	53.63	56.73	**60.79**
kodim10	59.08	60.51	53.66	56.77	**61.00**
kodim11	62.15	62.25	55.02	59.80	**63.23**
kodim12	60.41	61.05	53.67	57.81	**62.69**
kodim13	55.51	58.33	47.84	52.77	**58.91**
kodim14	58.85	60.35	52.20	56.57	**60.78**
kodim15	59.77	60.79	54.49	57.88	**63.16**
kodim16	61.72	61.56	54.91	59.02	**63.85**
kodim17	60.16	60.86	54.15	57.61	**62.21**
kodim18	56.90	59.03	51.36	55.07	**59.24**
kodim19	58.94	60.20	53.08	56.84	**60.82**
kodim20	54.52	60.35	-	58.20	**64.99**
kodim21	58.81	60.08	53.23	56.94	**60.63**
kodim22	58.67	59.76	52.29	56.47	**60.42**
kodim23	61.17	60.96	54.35	58.29	**62.96**
kodim24	60.72	62.14	53.80	59.67	**64.82**
Average	59.43	60.56	53.16	57.35	**61.91**

which can keep the grayscale values of image pixels invariant. Moreover, a 3rd-degree polynomial predictor was proposed to improve prediction accuracy. But, the constraint of grayscale invariance has a negative impact on image quality. We can see that the proposed method is significantly superior to Hou *et al.*'s method. To use the correlations between the three color channels more effectively, Zhan *et al.* embedded data based on 3D prediction-error histogram, and the image quality was improved by discarding mappings with large distortion. While, as shown in Fig. 9, the proposed method also outperforms Zhan *et al.*'s method. The advantage of the proposed method is that it employs a high accuracy predictor and embeds data with the adaptively selected mapping path to reduce distortion. For more details, the PSNRs of all the test methods under an embedding capacity of 30,000 bits are

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

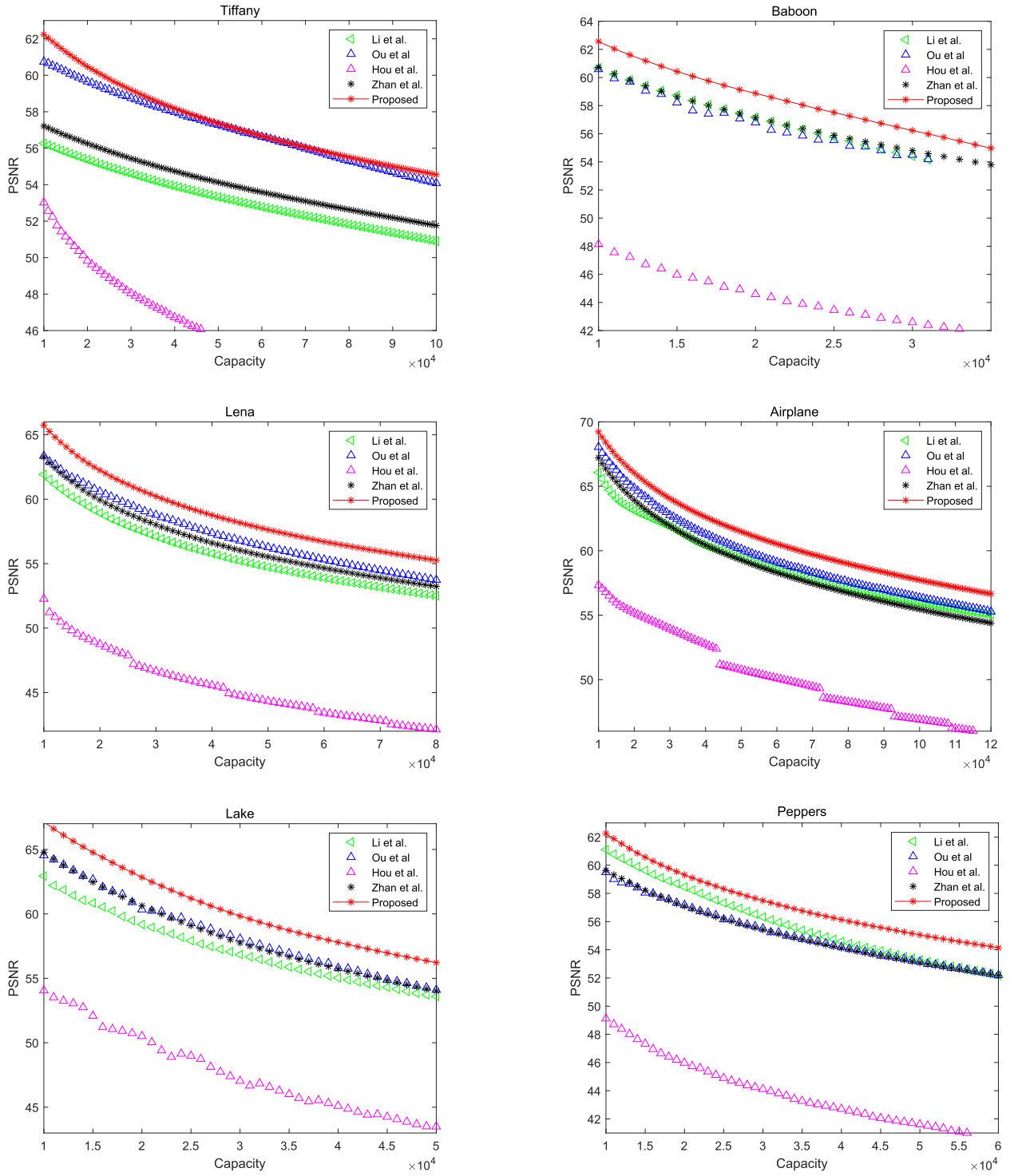


Fig. 9. Comparison of the PSNR (dB) among the proposed method, Li *et al.*'s [23], Ou *et al.*'s [24], Hou *et al.*'s [25] and Zhan *et al.*'s methods [26].

reported in Table III. The best PSNR values are highlighted in bold. Table III shows that, compared with Li *et al.*'s [23], Ou *et al.*'s [24], Hou *et al.*'s [25] and Zhan *et al.*'s [26] methods, the proposed method improves the PSNRs on average by 2.65 dB, 1.44 dB, 12.44 dB and 2.28 dB, respectively.

Last, 24 color images from Kodak image dataset are also tested to validate the effectiveness of the proposed method. The size of these color images is 512×768 or 768×512. Table IV shows the PSNRs of the five test methods under an embedding capacity of 50,000 bits. The symbol '—' is used to indicate that the embedding capacity of Hou *et al.*'s method cannot reach 50,000 bits for kodim20 image. From Table IV, the proposed method achieves better image quality than Li *et al.*'s [23], Ou *et al.*'s [24], Hou *et al.*'s [25] and Zhan *et al.*'s [26] methods with an increase of 2.48 dB, 1.35 dB, 8.75 dB, 4.56 dB on average, respectively. Note that

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

the average PSNR of Hou *et al.*'s method is calculated over all the test color images except Kodim20 image. Moreover, it is also shown that the proposed method outperforms all of the compared methods on all the 24 test images.

V. Conclusion

This paper has presented an adaptive 3D prediction-error expansion method for color image RDH. Double deep Q-network has been introduced to select the optimal mapping paths for 3D-PEH. An action selection scheme is proposed to quickly find reversible mapping paths. Extensive experimental results have demonstrated the superior performance of the proposed method over existing color image RDH methods. Nevertheless, considering the efficiency of data embedding, the optimal embedding mappings are applied to only a few, but not all, triples. In the future, a more efficient method will be investigated to optimize the embedding mappings for more triples.

References

[1] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding—New paradigm in digital watermarking," *EURASIP J. Adv. Signal Process.*, vol. 2002, no. 2, pp. 1–12, Dec. 2002.

[2] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.

[3] A. Cheddad, J. Condell, K. Curran, and P. M. Kevitt, "Digital image steganography: Survey and analysis of current methods," *Signal Process.*, vol. 90, no. 3, pp. 727–752, Mar. 2010.

[4] C. I. Podilchuk and E. J. Delp, "Digital watermarking: Algorithms and applications," *IEEE Signal Process. Mag.*, vol. 18, no. 4, pp. 33–46, Jul. 2001.

[5] G. Coatrieux, C. Le Guillou, J.-M. Cauvin, and C. Roux, "Reversible watermarking for knowledge digest embedding and reliability control in medical images," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 2, pp. 158–165, Mar. 2009.

[6] D. Hou, W. Zhang, J. Liu, S. Zhou, D. Chen, and N. Yu, "Emerging applications of reversible data hiding," in *Proc. 2nd Int. Conf. Image Graph. Process. (ICIGP)*, 2019, pp. 105–109.

[7] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Trans. Image Process.*, vol. 14, no. 2, pp. 253–266, Feb. 2005.

[8] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.

[9] J. Wang, J. Ni, X. Zhang, and Y.-Q. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 315–326, Feb. 2017.

[10] D. M. Thodi and J. J. Rodríguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.

[11] M. Fallahpour, "Reversible image data hiding based on gradient adjusted prediction," *IEICE Electron. Exp.*, vol. 5, no. 20, pp. 870–876, 2008.

[12] I. C. Dragoi and D. Coltuc, "On local prediction based reversible watermarking," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1244–1246, Apr. 2015.

[13] H. Chen, J. Ni, W. Hong, and T.-S. Chen, "High-fidelity reversible data hiding using directionally enclosed prediction," *IEEE Signal Process. Lett.*, vol. 24, no. 5, pp. 574–578, May 2017.

[14] Y. Jia, Z. Yin, X. Zhang, and Y. Luo, "Reversible data hiding based on reducing invalid shifting of pixels in histogram shifting," *Signal Process.*, vol. 163, pp. 238–246, Oct. 2019.

[15] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 2016–2027, Sep. 2015.

[16] W. Qi, X. Li, T. Zhang, and Z. Guo, "Optimal reversible data hiding scheme based on multiple histograms modification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2300–2312, Aug. 2020.

[17] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Process.*, vol. 93, no. 1, pp. 198–205, 2013.

[18] X. Li, B. Li, B. Yang, and T. Zeng, "General framework to histogram-shifting-based reversible data hiding," *IEEE Trans. Image Process.*, vol. 22, no. 6, pp. 2181–2191, Jun. 2013.

[19] B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 5010–5021, Dec. 2013.

[20] T. Zhang, X. Li, W. Qi, and Z. Guo, "Location-based PVO and adaptive pairwise modification for efficient reversible data hiding," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2306–2319, 2020.

[21] X. Gui, X. Li, and B. Yang, "High-dimensional histogram utilization for reversible data hiding," in *Proc. Int. Workshop Digit. Watermarking*. Cham, Switzerland: Springer, 2014, pp. 243–253.

[22] S. Cai, X. Li, B. Xue, and Z. Guo, "A new reversible data hiding scheme based on high-dimensional pixel-intensity-histogram modification," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2016, pp. 1–6.

[23] J. Li, X. Li, and B. Yang, "Reversible data hiding scheme for color image based on prediction-error expansion and cross-channel correlation," *Signal Process.*, vol. 93, no. 9, pp. 2748–2758, 2013.

[24] B. Ou, X. Li, Y. Zhao, and R. Ni, "Efficient color image reversible data hiding based on channel-dependent payload partition and adaptive embedding," *Signal Process.*, vol. 108, pp. 642–657, Mar. 2015.

[25] D. Hou, W. Zhang, K. Chen, S.-J. Lin, and N. Yu, "Reversible data hiding in color image with grayscale invariance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 2, pp. 363–374, Feb. 2018.

[26] Y. Zhan, Y. Su, X. Wang, and Q. Pei, "Three-dimensional prediction-error histograms based reversible data hiding algorithm for color images," *Multimedia Tools Appl.*, vol. 78, no. 24, pp. 35289–35311, Dec. 2019.

[27] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[28] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[29] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[30] Y. Chen, Z. Wang, Z. J. Wang, and X. Kang, "Automated design of neural network architectures with reinforcement learning for detection of global manipulations," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 5, pp. 997–1011, Aug. 2020.

[31] W. Tang, B. Li, M. Barni, J. Li, and J. Huang, "An automatic cost learning framework for image steganography using deep reinforcement learning," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 952–967, 2021.

[32] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, vol. 30, no. 1, pp. 1–7.

[33] S.-Y. Chen, Y. Yu, Q. Da, J. Tan, H.-K. Huang, and H.-H. Tang, "Stabilizing reinforcement learning in dynamic environment with application to online recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1187–1196.

[34] Z. Gao, Y. Gao, Y. Hu, Z. Jiang, and J. Su, "Application of deep Q-network in portfolio management," in *Proc. 5th IEEE Int. Conf. Big Data Anal. (ICBDA)*, May 2020, pp. 268–275.

[35] W. Zhang, J. Gai, Z. Zhang, L. Tang, Q. Liao, and Y. Ding, "Double-DQN based path smoothing and tracking control method for robotic vehicle navigation," *Comput. Electron. Agricult.*, vol. 166, Nov. 2019, Art. no. 104985.

Jie Chang received the M.S. degree in communication engineering from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2016. She is currently an Engineer with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. Her research interests focus on multimedia security and image processing.

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.

Guopu Zhu (Senior Member, IEEE) received the B.S. degree in transportation from Jilin University, China, in 2002, and the M.S. and Ph.D. degrees in control science and engineering from the Harbin Institute of Technology, China, in 2004 and 2007, respectively. He is currently a Professor with the Harbin Institute of Technology. He has authored or coauthored more than 40 articles in peer-reviewed international journals. His main research areas are multimedia security, image processing, and control theory.

Hongli Zhang (Member, IEEE) received the B.Sc. degree in computer science from Sichuan University, Chengdu, China, in 1994, and the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1999. She is currently a Professor with the School of Cyberspace Science, Harbin Institute of Technology. Her research interests include network and computer security, network modeling, and parallel processing.

Yicong Zhou (Senior Member, IEEE) received the B.S. degree in electrical engineering from Hunan University, Changsha, China, and the M.S. and Ph.D. degrees in electrical engineering from Tufts University, Medford, MA, USA.

He is currently a Professor with the Department of Computer and Information Science, University of Macau, Macau, China. His research interests include image processing, computer vision, machine learning, and multimedia security.

Dr. Zhou is a fellow of the Society of Photo-Optical Instrumentation Engineers (SPIE). He was recognized as one of the highly cited researchers in 2020 and 2021. He received the Third Price of Macao Natural Science Award as a Sole Winner in 2020 and a co-recipient in 2014. He has been the Leading Co-Chair of Technical Committee on Cognitive Computing in the IEEE Systems, Man, and Cybernetics Society since 2015. He was an Associate Editor for four other journals and received the Best Editor Award in the *Journal of Visual Communication and Image Representation* in 2020 and 2021. He serves as an Associate Editor for IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING.

Xiangyang Luo received the B.S., M.S., and Ph.D. degrees from the State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, China, in 2001, 2004, and 2010, respectively. He is currently a Professor with the State Key Laboratory of Mathematical Engineering and Advanced Computing and also with the Key Laboratory of Cyberspace Situation Awareness of Henan Province. He is the author or the coauthor of more than 150 refereed international journals and conference papers. His research interests are multimedia security and network security.

Ligang Wu received the B.S. degree in automation from the Harbin University of Science and Technology, China, in 2001, and the M.E. degree in navigation guidance and control and the Ph.D. degree in control theory and control engineering from the Harbin Institute of Technology, China, in 2003 and 2006, respectively.

He was a Research Associate/Senior Research Associate with The University of Hong Kong, the City University of Hong Kong, and Imperial College London. He is currently a Professor with the Harbin Institute of Technology. He has published seven research monographs and more than 200 research articles in internationally refereed journals. His current research interests include analysis and design for cyber-physical systems, robotic and autonomous systems, intelligent systems, and power electronic systems. His awards and recognitions include the National Science Fund for Distinguished Young Scholar, the China Young Five-Four Medal, and the Distinguished Professor of Chang Jiang Scholar. He has been the Highly Cited Researcher since 2015. He serves as an Associate Editor for a number of journals, including IEEE TRANSACTIONS ON AUTOMATIC CONTROL, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE/ASME TRANSACTIONS ON MECHATRONICS, and *IET Control Theory and Applications*.

Authorized licensed use limited to: Universidade de Macau. Downloaded on August 05,2022 at 00:40:49 UTC from IEEE Xplore. Restrictions apply.