Hindawi
Security and Communication Networks
Volume 2021, Article ID 5546344, 12 pages
https://doi.org/10.1155/2021/5546344

Research Article

Data Hiding Based on Mini Program Code

Jia Chen, Quankai Qi, Yongjie Wang, Xuehu Yan, and Longlong Li

National University of Defense Technology, 230037 Hefei, China

Correspondence should be addressed to Xuehu Yan; publictiger@126.com

Received 3 March 2021; Revised 20 April 2021; Accepted 13 May 2021; Published 24 May 2021

Academic Editor: Zhili Zhou

A mini program code (also known as sunflower code) comes with WeChat mini program APPs. With the popularization of mini program APPs, mini program codes have become more and more widely used. As a new carrier, the study of information hiding technology based on mini program code is of great significance for the expansion of covert communication carriers and can also deal with security problems in advance. At present, to the best of our knowledge, there is no steganographic research based on mini program codes. In this paper, we propose a scheme to embed secret information into mini program codes for the first time. After studying the construction of mini program codes, a coordinate system is constructed to represent its module coordinates. Then, a binary stream of the secret message is embedded into the encoding region or the edge patch. Experiments show that the proposed data hiding scheme is effective and feasible. The embedded secret message could be extracted while keeping the readability of the mini program code. Moreover, the secret payloads of the encoding region and the edge patch for the V-36 mini program code are 72 bits and 29 bits, respectively.

1. Introduction

A mini program code comes with a WeChat mini program app [1]. Mini programs are "miniapplications" built within the WeChat platform. They are instantly loaded and easy to use. WeChat allows third party to develop mini programs and bundle features and capabilities into a single native mobile App. WeChat mini programs provide convenient life services, such as public welfare, online shopping, tourism transportation, and other fields. WeChat mini program relies on WeChat, a super App with about one billion users. Since its launch on 9 January, 2017, it has attracted much attention. By July 2020, there are more than 3.2 million WeChat mini programs, with more than 410 million daily active users and 8.5 daily use per person [2]. Meanwhile, the mini program code is also more and more popular with the widespread mini programs. After the WeChat mini program was used, Alipay and Baidu also developed their mini programs. In this paper, mini program codes refer to WeChat mini program codes unless otherwise stated.

The purpose of covert communication is to transmit secret information between two parties and prevent it from being intercepted by malicious eavesdroppers. As a new carrier, the study of information hiding technology based on mini program codes is of great significance for the expansion of new covert communication. Moreover, the mini program code is a barcode. At present, the security of mini program codes has not been reported, but lots of barcode security problems have been exposed. Especially, with the widespread use of QR codes, its security issues have gradually been paid attention to, such as information leakage and embedded malicious links. Therefore, the study of information hiding technology based on mini program codes can also deal with security problems in advance.

Image steganography is the technique of hiding secret information in an image. The common image steganography methods fall into two categories. The first category is in the spatial domain, and the most common method is based on the modification of least significant bit (LSB) [3]. LSB replaces image pixels with bits of secret message. Luo et al. [4] propose the LSB matching method, where the data is hidden in the edges. The other category is in the frequency domain. The secret message is hidden into the transform coefficients. There are some common techniques to obtain those coefficients, such as discrete cosine transform (DCT), discrete wavelet transform (DWT), and Fresnelet transform (FT).

Chang et al. [5] propose a reversible data hiding scheme where secret messages are embedded into DCT coefficients, and Lin [6] proposes reversible data hiding in the DCT coefficient using a histogram shifting method. Other methods of information hiding, such as reversible data hiding (RDH) in images and reversible image secret sharing (RISS), are also being studied widely. Sun et al. [7] propose a reversible data hiding scheme for encrypted color halftone images. Yan et al. [8] propose a RISS algorithm for a (k, n) -threshold based on the Chinese remainder theorem-based ISS (CRTISS). Yan et al. [9] fuse the advantages of polynomial-based image secret sharing and visual secret sharing to design a common method of share authentication in image secret sharing.

With the further study of QR codes, more and more secret hiding schemes based on QR codes are proposed. These schemes also can be divided into two categories. The first category is to hide the secret message by redesigning the QR module pattern. Teraura and Sakurai [10] propose a scheme in which the QR code modules are segmented into different submodules with a size of 3×3 or 5×5. Then, secret message bits are hidden in the outer of the submodules' pattern. The other category is to hide the secret message in the data module directly by sacrificing the error correction capacity of QR codes. Chiang et al. [11] propose a steganographic method to hide the secret message in the QR code. The Wet Paper Codes' algorithm is applied in this method. They replace the dry modules of the QR code with the secret message bits. Recently, some novel steganography schemes based on QR codes have been proposed. Pan et al. [12] propose a new image steganography algorithm hiding secret messages into an innovative embedding domain by combining compressive sensing with subsampling. Moreover, the combination of information hiding and secret sharing based on QR codes has been proposed. Wang et al. [13] propose a visual secret-sharing scheme (VSSS) with (k, n)-threshold based on QR codes. In the processing of encoding QR code, the bits corresponding to shares generated by VSS from a secret bit are embedded into the same locations of QR codes. Tan et al. [14] propose an XVSS scheme based on the QR code. The bits of shadows are embedded into the pad code words of QR codes. Cheng et al. [15] propose a VSSS to encode a secret QR code into several valid QR codes. These codes can be decoded by a standard QR code reader; meanwhile, the secret messages are recovered by XOR operation.

To expand the carrier in covert communication, there are currently some steganography schemes based on new carriers [16, 17]. As a new carrier of mini program code, it is necessary to study the data-hiding scheme based on it. Steganography based on mini program codes is different from the common image steganography. A mini program code is an image, but the code should be able to be decoded successfully after embedding secret messages. Although steganography based on QR codes also requires to be decoded successfully, the mini program codes are more difficult to implement information hiding than QR codes. The module distribution of mini program codes is more restricted than QR codes. The angles between the rays on mini program codes and the module distribution on each ray are specified. That is to say, there are more stringent restrictions on embedding secret messages. Meanwhile, WeChat mini program apps are only used in WeChat, and the mini program source code is not open. Therefore, we do not know about the coding method, codeword distribution, or encryption method of mini program code. We can only rely on the WeChat scanner to decode instead of designing the decoding program according to our needs. At present, no research has been conducted on data hiding based on mini program codes.

In this paper, to expand the carrier in covert communication and deal with the security problem based on mini program codes in advance, we propose a novel data-hiding scheme based on mini program codes for the first time. First, we study the function and construction of each pattern of mini program codes. Then, we construct a coordinate system to represent the module coordinates. Finally, we replace the modules with secret messages. Moreover, the secret payload of the hiding scheme is tested. Experimental results validate the effectiveness of our scheme.

The contributions of our work are summarized as follows.

(1) Relevant theories on mini program codes are studied to provide experience for scheme design and follow-up research

(2) We propose the data-hiding scheme based on mini program codes for the first time

(3) We test the secret payload of the data-hiding scheme based on mini program codes

The rest of this paper is organized as follows. In Section 2, we introduce the function and construction of mini program codes. The proposed scheme is described in Section 3. Section 4 gives experimental results for validating the proposed scheme. Finally, conclusions are drawn in Section 5.

2. Preliminaries

A module is the smallest unit of data storage in the mini program code, representing 1 bit, and one code word has 8 bits. There are three different versions of mini program codes: 36 rays, 54 rays, and 72 rays, named V-36, V-54, and V-72, respectively. As the code of V-36 is used widely, the research of this paper is mainly on the V-36 mini program codes. All the modules are evenly distributed over these rays. Figure 1 shows the three versions.

In Figure 2, there are four functional patterns, two logo areas, and an encoding region in a mini program code. The isosceles right triangle distribution of the three finder patterns is conducive to detection. Finder patterns contain numerical information about the rays. The version information pattern consists of error correction level and mask. And, modules of the version information pattern can correct themselves. The contour patch highlights the logo shape and separates the user logo area from the encoding region. The official logo area not only indicates what platform the mini

(a) (b) (c)

Figure 1: Three versions of mini program codes.

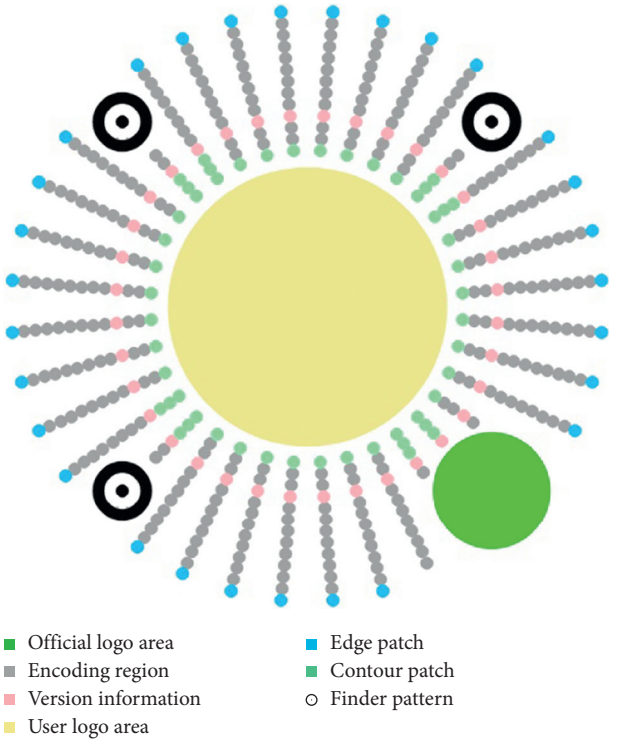


Figure 2: The structure of a V-36 mini program code.

program is based on but also assists in detection. To be detected from all directions and decoded successfully, these patterns must be placed in the corresponding area without errors. Of course, some patterns can be modified within a certain range without affecting the normal decoding. The user logo area improves the accessibility of the mini program code. It is easy to see what the app is at a glance. The edge patch is the outermost module of the ray, which does not carry any information and is only used to outline the pattern. The encoding region consists of data and error correction code words.

When the mini program code is used, it may have some interference, such as dirt, cropping, and fouling. So, the Reed–Solomon (RS) code is applied to correct errors in mini program codes. These error correction code words correct errors and recover data. Table 1 lists error correction capabilities of standard mini program codes in four levels.

Table 1: Error correction levels and their durability.

Error correction level	Data recovery capability (%)
L	10
M	15
Q	25
H	35

Corresponding to different versions, error correction levels, and different character sets, the data and error correction code words are different. A mini program code version with a lower error correction level has more data capacity. Table 2 shows the maximum encoding capacity under different conditions [18]. As the number of rays increases, so does the maximum encoding capacity, regardless of error correction levels.

When the input messages are encoded, the data are first converted into a bit stream. Then, error correction codes are added to get the final bitstream. Next, the bitstream are padded in the encoding region according to the rule that the white and black modules represent "0" and "1," respectively. To avoid the sparsity of black modules, one of 32 data mask patterns is selected to mask the encoding area after encoding all the code words.

Like QR codes, mini program codes are encoded using error correction mechanisms that allow errors to occur during decoding. Meanwhile, mini program codes are decoded by reading the pixel value of each module in turn. Each module is judged to be a binary digit of either "1" or "0" by a certain threshold. Based on the analysis of QR code information hiding technology, we can apply relevant principles to mini program codes and then design a data-hiding scheme based on mini program codes.

3. The Proposed Scheme

The proposed data-hiding scheme for mini program codes can be split into a secret embedding procedure and an extraction procedure.

3.1. Embedding Procedure. Given a mini program code, secret message S, and random seed k, the proposed scheme embeds the secret message S into the encoding region or the edge patch of the mini program code and retains the

Table 2: Maximum encoding capacity.

Version level		Data modes		
		Numeric	Alphanumeric	8 bit byte
V-36	L	63	32	25
	M	55	29	22
	Q	43	22	17
	H	29	15	11
V-54	L	87	44	35
	M	79	41	32
	Q	58	29	23
	H	43	22	17
V-72	L	106	54	43
	M	96	49	39
	Q	72	37	29
	H	53	27	21

remaining patterns unmodified. The embedding process is given in Algorithm 1.

By modifying mini program code modules, the secret messages can be embedded. Meanwhile, the marked code could be decoded successfully when the embedding capacity is within the error correction capability.

3.2. Extraction Procedure. In the extraction procedure in Algorithm 2, the receiver can recover the secret message S from the marked code according to the random seed k by our extract program.

3.3. Specific Implementation

3.3.1. Determine Center and Module Segmentation. Looking at mini program code visually, the code blooms from inside to outside like a sunflower, and all modules are evenly distributed over each ray. The first thing is to determine the center (c_x, c_y) of the mini program code and the length l_{cw} of the module. We adopt the Hough transform [19] to detect the circle. For a given cover code, by setting the radius range of the detection circle, the radii and centers of three finder patterns and the official logo area can be obtained and represented by $((c_{1x}, c_{1y}), r_{f1})$, $((c_{2x}, c_{2y}), r_{f2})$, $((c_{3x}, c_{3y}), r_{f3})$, and $((c_{4x}, c_{4y}), r_{ol})$. According to the design in [18], (c_x, c_y) and l_{cw} can be computed by equations (1) and (2), respectively:

$$\left(c_x, c_y\right) = \left(\frac{c_{1x} + c_{2x}}{2}, \frac{c_{1y} + c_{3y}}{2}\right), \tag{1}$$

$$l_{cw} = \frac{r_{ol}}{10}. \tag{2}$$

After obtaining l_{cw}, we can draw concentric circles which start with the circle in the area of the user's logo and take the center of the circle as the center and the length of the code word as the increasing radius. Thus, all the modules are segmented by the concentric circles, as shown in Figure 3.

3.3.2. Establishing Coordinate System. To conveniently represent the position of each module, we need to establish a coordinate system and keep it consistent with the actual pixel coordinates. The positions of the modules can be determined by the number of rays and the order of modules on the rays. We start from the ray between the center of the user logo area and the center of the finder pattern in the upper right corner and then number of the rays counterclockwise.

The established coordinate system has undergone four transformations. Figure 4 shows the three transformations of the established coordinate system.

First, we use $P(i, j)$ to denote the module coordinates which represent the i^{th} ray and the j^{th} module on the ray.

Second, we establish a coordinate system according to the location relationship among the three finder patterns to form an isosceles right triangle. Take the line between the center of the user logo area and the center of the finder pattern in the upper right corner as x' axis and the line between the center of the user logo area and the center of the finder pattern in the upper left corner as y' axis. Then, the coordinate of P in the coordinate system $x'o'y'$ is $P(x', y')$ computed by

$$\begin{cases} x' = (r + j * l_{cw}) * \cos(i * \theta), \\ y' = (r + j * l_{cw}) * \sin(i * \theta). \end{cases} \tag{3}$$

Then, the coordinate system $x'o'y'$ is rotated by α radian clockwise to the coordinate system $x''o'y''$, where $\alpha = (\pi/4)$. $P(x'', y'')$ can be obtained by

$$\begin{cases} x'' = x' \cos(\alpha) - y' \sin(\alpha), \\ y'' = x' \sin(\alpha) + y' \cos(\alpha). \end{cases} \tag{4}$$

Finally, we translate the coordinate system $x''o'y''$ to the upper left corner of the image xoy to keep xoy consistent with the actual pixel coordinates. Therefore, the final coordinate $P(x, y)$ can be computed by equation.

$$\begin{cases} x = x'' + c_x, \\ y = c_y - y''. \end{cases} \tag{5}$$

3.3.3. Modified Strategy. To better embed the secret message, we formulate a modified strategy to choose the embedded region and decide how to embed it. First, the embedded area is determined by the length n of the secret message and the number n_e of modules in the edge patch of each version of the mini program code. As n_e is fixed, if $n > n_e$, the secret message should be embedded in the encoding region or the combination of the encoding region and the edge patch. When $n \le n_e$, we can choose the encoding region or the edge patch to embed.

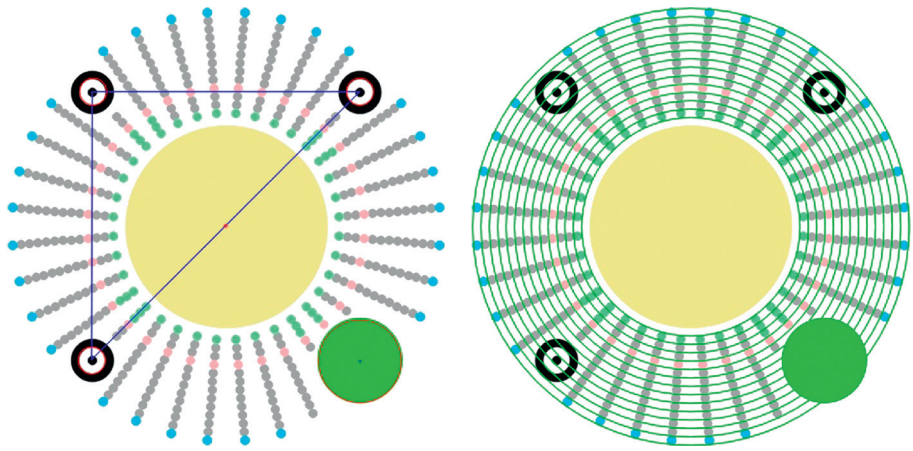
Then, we choose direct substitution to embed the secret message. This is to replace the modules with $\text{Bin}(S)$. The black and white modules represent binary digits "1" and "0," respectively. As we can see, the mini program code is composed of several circular modules. Some continuous circular modules are joined into a thick line. When we choose a module to modify, we need to consider the situation of the two modules before and after this one on the ray-

Input: a cover code, secret message S, and random seed k
Output: a marked code with the secret message and the length n of the secret message
Step 1: determine the center (c_x, c_y) of the mini program code and the length l_{cw} of each module
Step 2: establish a coordinate system and store module coordinates $P(x, y)$ of encoding region in an array
Step 3: convert the secret message S into a binary stream and compute its length n: $n = \text{len}(\text{Bin}(S))$.
Step 4: choose the embedded region based on the secret length n, and determine the module to start modification according to the random number generated by random seed k
Step 5: replace modules with $\text{Bin}(S)$ from the start modification module continuously
Step 6: repeat Step 5 until all the bits are hidden in the cover code

Algorithm 1: The embedding procedure.

Input: a marked code with the secret message S, the length n of the secret message S, and random seed k
Output: the secret messages S
Step 1: determine the center (c_x, c_y) of the mini program code and the length l_{cw} of each module
Step 2: establish a coordinate system and store module coordinates $P(x, y)$ of encoding region in an array
Step 3: determine the embedded region and the start modification module based on n and the random number generated by random seed k
Step 4: read the module pixel of the marked code from the start modification module; if the module is white, the secret message is 0; otherwise, it is 1
Step 5: repeat step 4 until all the secret bits have been extracted

Algorithm 2: The extraction procedure.



(a) (b)

Figure 3: The diagram of (a) determine center and (b) module segmentation.

whether they are independent modification circles or whether they need to be connected into a thick line. Specifically, if the black module needs to be modified to white, there are four cases shown in Table 3, and so for the case where the white module is modified to a black module.

3.4. Capacity Analyses. The embedding capacity of a secret message depends on many factors. When we use a module or a pixel value in a module to represent a bit of the message, the embedding capacity is different. Moreover, the size of the image and secret message encoding method also affects the embedding capacity. Essentially, the embedding capacity is related to the number of modules available to be modified in the mini program code. In the structure of the mini program code, the RS error correction algorithm is employed. The encoding region consists of data code words and error correction code words by using the RS code. From Chen [20], we could know that there may be 20 code words in the data area and 18 code words in the error correction area in the V-36 mini program code. According to the error correction principle of RS error correction code, two RS error correction code words can correct one code word error [21]. Therefore, in the V-36 mini program code, 18 error correction code words can correct 9 code words. That is to say, we could embed up to 9 code words of secret messages. In

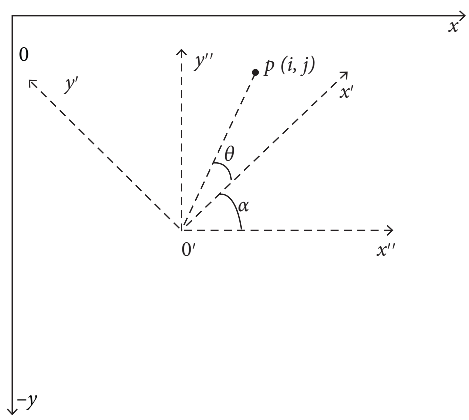


Figure 4: Four transformations of the established coordinate system.

Table 3: Four cases of the module for “1” needs to be “0.”

Case	Cover code		Marked code	
	Bits	Modules	Bits	Modules
1	(1, 1, 1)		(1, 0, 1)	
2	(0, 1, 1)		(0, 0, 1)	
3	(1, 1, 0)		(1, 0, 0)	
4	(0, 1, 0)		(0, 0, 0)	—

Note: ● represents the modified module.

this scheme, modules in the region area are directly replaced to embed information without data encoding. This is the simplest embedding method which can only deal with general noise interference. The secret message could not be recovered for interference, such as dirt, cropping, and fouling, in the scheme.

4. Experiment and Evaluation

In this section, some experiments are conducted to verify the feasibility and effectiveness of the proposed scheme. Two examples are given in Section 4.1 to evaluate the practicability of the proposed scheme. To determine the type of modification, noise experiments are performed in Section 4.2. The secret payload of the proposed scheme is tested in Section 4.3. In Section 4.4, a subjective evaluation method is used to test the effectiveness of our scheme.

4.1. Examples of the Proposed Scheme. Through the above analysis, a secret message could be embedded in the edge patch and the encoding region by the proposed scheme. Two examples are given in Python to evaluate the practicability of the proposed scheme. The cover image is a V-36 mini program code of WeChat open class. The image size is 512 × 512 pixels. When the length of the secret message is less than 29 bits, both edge patch and encoding region can be selected to embed. The secret message is the string “NUDT,” and its ASCII value is converted into a binary stream. The binary stream of the cover image is obtained by reading the code words in the embedded area sequentially.

The binary streams of the secret message and the edge patch are shown as follows:

(10**01**11**0**10101**01**100**01**00101**0**1**0**0)

(10**10**11**1**10101**10**100**10**00101**1**1**1**0)

The binary streams of the secret message and the encoding region are shown as follows:

(**10**0111**0**10**1**0**101**100**01**0**0**1**0**1**0**10**0**)

(**01**0111**1**10**0**0**010**100**10**0**1**1**1**1**1**10**1**)

Figure 5 shows two examples of the proposed scheme, where Figures 5(a)–5(d) show the process of embedding the secret message in edge patch and Figures 5(e)–5(h) show the process in the encoding region. Figures 5(a) and 5(e) are the cover images. The green circles in Figures 5(b) and 5(f) are the positions to embed the secret message. The embedded results are shown in Figures 5(c) and 5(g). From the binary stream, we can see how many bits are different between the secret message and the embedded area marked in bold. Also, the red circles in Figures 5(d) and 5(h) represent the modified positions.

Both of the marked images in the example could be decoded by the WeChat scanner, and then, navigate into the mini program. The user will not notice its difference because it can be decoded and used normally. However, we can extract the secret message by reading the modules in the embedded area sequentially with our program. Through these two examples, we verify that both the edge patch and the encoding region could embed the secret message.

4.2. Modified-Type Test. In this section, noise experiments are performed to determine the type of modification. To perform a series of experiments based on the mini program code, we randomly collect 51 mini program codes from the Internet as a database. The decoding exception of a mini program code is defined as (i) the mini program code cannot be decoded correctly, (ii) the speed of decoding the code is slower than normal, and (iii) the scanner needs to be tilted to decode successfully.

According to the analysis of the function patterns of the mini program codes, we find that the secret message could be embedded in the edge patch and the encoding region. Although we suspect that mini program codes may use the RS code to encode data, we do not know in what order the code words are arranged. Therefore, according to the effect of noise on decoding, we divide the noise into scattered point-like noise and continuous line-like noise. Furthermore, two modified types correspond to these two types of noise, namely, random average embedding (RAE) and continuous area embedding (CAE). RAE means that the modules on each ray have an equal chance to be modified. CAE is to modify the code words on the ray continuously.

For RAE, we test three modifications of 10, 15, and 20 points. As for CAE, we also test three modifications of 3, 5, and 7 lines. Each line contains 4 ~ 6 consecutive points. Taking the mini program code of WeChat open class as an

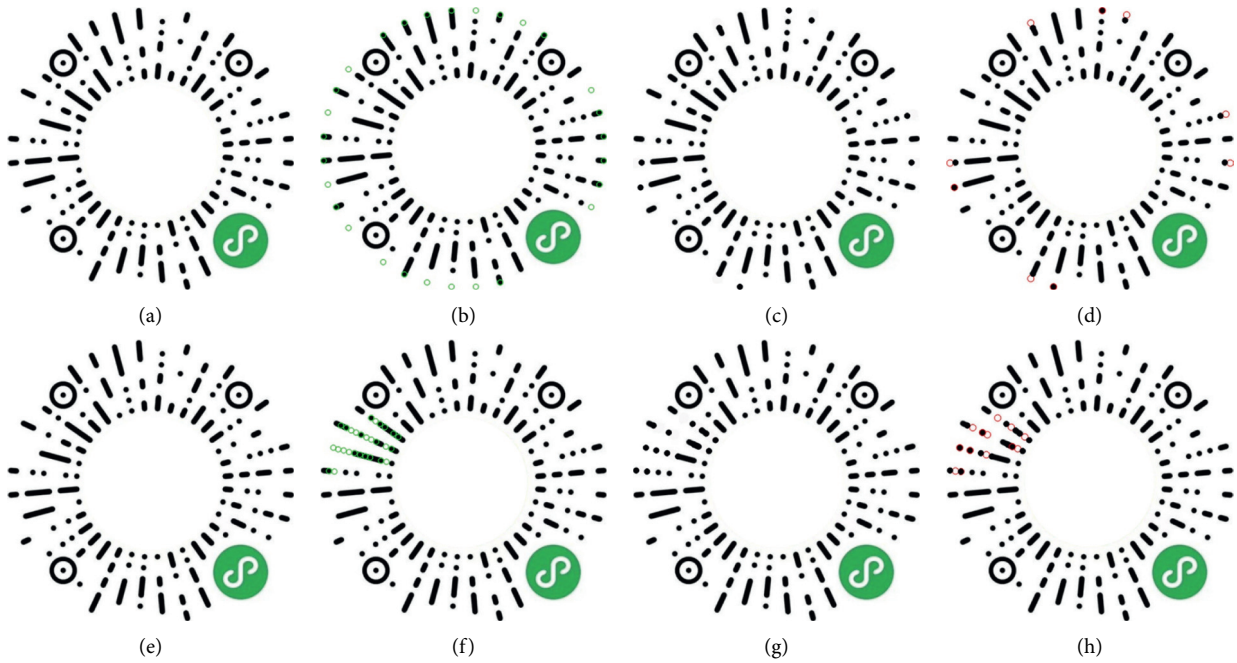


Figure 5: Two examples of the proposed scheme. (a) Cover image. (b) Embedding position. (c) Marked image. (d) Modified position. (e) Cover image. (f) Embedding position. (g) Marked image. (h) Modified position.

example, six modification versions are shown in Figure 6. Fifty-one mini program codes are tested separately, and the normal decoding rates (NDR) of six modification versions are calculated.

As we can see from the results in Table 4, when we choose CAE, even if we modify the modules with 5 lines, which is approximately 20 ~ 30 bits, the normal decoding rate is still 100%. With the same embedding capacity, when we modify 20 points, the normal decoding rate can only reach 44.83% by using RAE. Therefore, to improve the embedding capacity and ensure a high normal decoding rate, we choose CAE as the modified type.

4.3. The Secret Payload of the Proposed Scheme. In this section, we test the secret payload of the proposed scheme. First, according to the analysis of the function of patterns, the edge patch is used to make the appearance of the mini program code more harmonious. The modules in the edge patch have no concrete meaning. Therefore, we flip all bits in the edge patch to test the capacity. If the mini program code could be decoded normally, all bits in the edge patch can be modified. The edge patches of 51 mini program codes of the database are tested, and all codes are decoded correctly. There are 29 bits in the edge patch of the V-36 mini program code. Therefore, the embedding capacity of the edge patch is 29 bits.

Secondly, we test the embedding capacity in the encoding region. At present, to the best of our knowledge, there is no official documentation on how to encode data into a mini program code. In Section 4.2, we conclude that CAE should be adopted to embed secret messages. Observing the existing V-36 mini program code, we can find that there are 304 modules in the encoding region. And, according to the introduction of the mini program code in Chen [20], the encoding region is divided into two parts. There may be 160 bits in the data area and 144 bits in the error correction area. The data area is located in the top half of the mini program code, where the rays are sequenced from left to right, and the bits on each ray are encoded from inside to outside. The error correction area is located in the lower half of the mini program code followed by the data area and the error-correcting codes are encoded in the same order. All information about the data encoding method of mini program code is not officially confirmed, so we have to test our conjectures experimentally to design a secret message hiding scheme and obtain the secret payload.

To obtain the maximum capacity of the encoding region, 51 mini program codes in the database are tested. Several experiments are designed on each code. We start with the first bit of the data area, and then, we continuously flip 56, 64, 72, and 80 bits. From the results in Table 5, we can guess that the maximum capacity should be between 72 and 80 bits. Then, we flip the 73 or 74 continuous bits. The normal decoding rate drops significantly when the number of bits flipped switches from 72 to 73. Therefore, we can tentatively conclude that the maximum capacity is 72 bits. However, the normal decoding rate is not satisfactory when we flip 72 continuous bits from the second bit to the 73rd bit. Based on the above situation, we guess that the 8 bits of each code word are distributed continuously. According to the RS code, they have the same effect when we modify a bit and a byte. Through experiments with serial nos. 3 ~ 19, we can determine that the embedding capacity is 9 code words and verify that every 8 continuous bits compose a code word.

To further test the effect of different distributions of 9 code words on normal decoding rate, 51 mini program codes

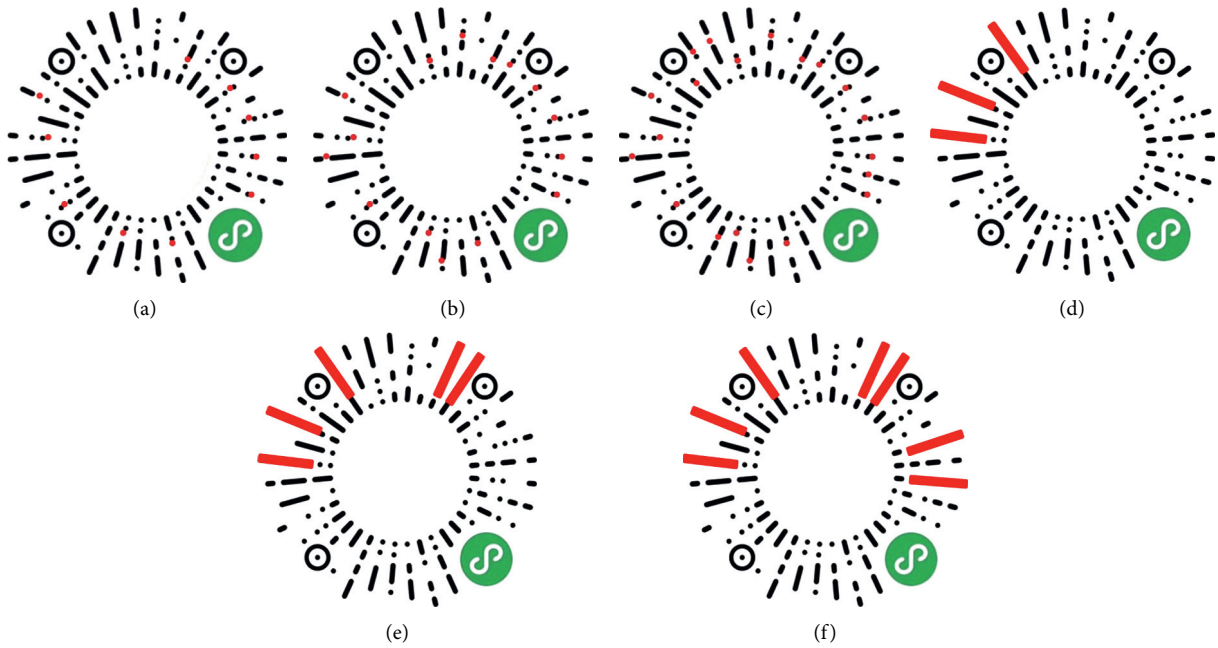


Figure 6: Six modification versions of embedding. (a) 10 points. (b) 15 points. (c) 20 points. (d) 3 lines. (e) 5 lines. (f) 7 lines.

in the database are tested. There are 20 code words in the data area of the V-36 mini program code. We number the 20 code words from 1 to 20. Then, we design four distributions of 9 code words as follows: (i) every modified code word is followed by one code word (1M + 1I), (ii) every two modified code words are followed by one code word (2M + 1I), (iii) every three modified code words are followed by one code word (3M + 1I), and (iv) all nine code words are modified without any intervals (9M + 0I). As can be seen from the experimental results in Table 6, the more intervals between 9 code words, the lower the normal decoding rate.

Based on the above experiments, for the V-36 mini program code, we can draw the following conclusions:

(1) The secret payload in the edge patch is 29 bits.

(2) The secret payload in the encoding region could reach 9 code words, that is, we could embed up to 72 bits in the encoding region.

(3) When 72 bits of the secret message are embedded in the encoding region, modification needs to start from a specific module which is the first bit of each code word. When the length of the secret message is less than 56 bits, the starting position of modification could be any modules.

(4) Different embedding methods have different embedding capacities. We can apply steganographic code in data hiding based on mini program codes to improve the embedding capacity.

4.4. Subjective Evaluation. There are many statistical indicators for evaluating image quality [22], such as MSE, PSNR, and SSIM. When we use the mini program codes, the users do not know the specific design or coding of them. They usually only focus on whether the code can be decoded and used successfully. Thus, we use a subjective evaluation method to test the effectiveness of our scheme. In statistics, $D\xi$ and $E\xi$ are usually utilized to analyze the subjective evaluation scores. $D\xi$ is the variance of the scores, and the mean of the scores is $E\xi$. The larger the $E\xi$ is, the better the effect of our scheme is, and the smaller the $D\xi$ is, the better the validity of this sample is. To test the effectiveness of our scheme, we conduct a subjective evaluation as follows.

Table 4: The normal decoding rates of six modification versions.

RAE	10 points	15 points	20 points
NDR	100%	72.41%	44.83%
CAE	3 lines	5 lines	7 lines
NDR	100%	100%	62.07%

In the experiments, we test a total of 50 different mini program codes with a size of 512×512 and different embedding capacities n through a questionnaire online. The 50 different mini program codes contain 10 types of modification, among which 8 are modified in the encoding region area and 2 are modified in the edge patch. Each type consists of 5 codes. We use "10e" and "20e" to represent that 10 bit and 20 bit modules are modified in the edge patch, respectively. We send questionnaires to a specific group of students and teachers with ages between 18 and 45, normal visual abilities, and no knowledge of the mini program code. According to several cases of decoding exceptions, we design criteria to evaluate the user experience of the marked code as follows: whether it can be correctly decoded and the speed of recognition, whether the modification trace can be seen, etc. One mini program code of each modification type is displayed, and part of mini program codes from the questionnaire is shown in Figure 7. After viewing each modified code, each participant of the questionnaire marks a score

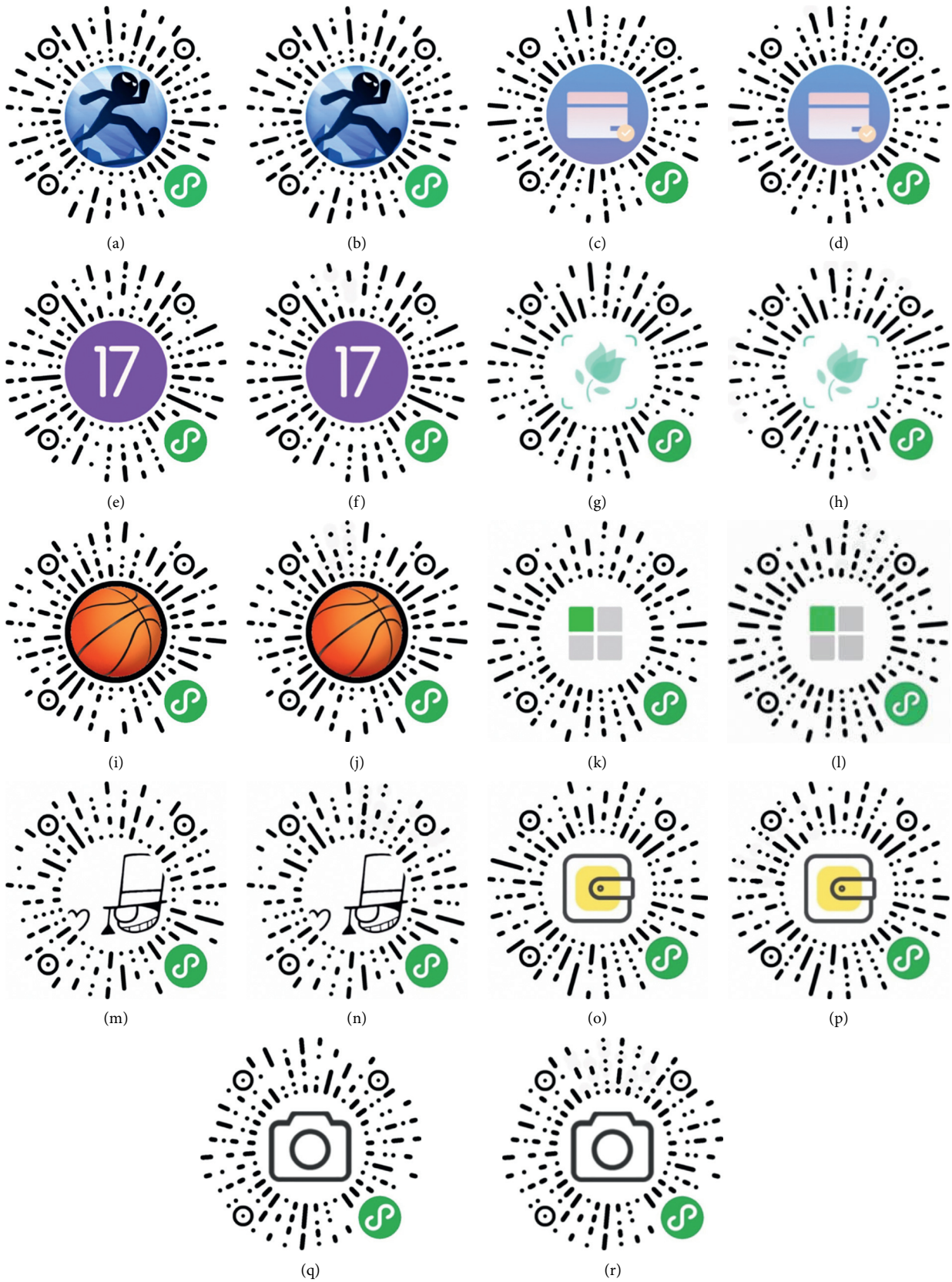


Figure 7: Part of mini program codes from the questionnaire. (a) Cover (10). (b) Modified (10). (c) Cover (10*e*). (d) Modified (10*e*). (e) Cover (20). (f) Modified (20). (g) Cover (20*e*). (h) Modified (20*e*). (i) Cover (30). (j) Modified (20). (k) Cover (40). (l) Modified (40). (m) Cover (50). (n) Modified (50). (o) Cover (60). (p) Modified (60). (q) Cover (70). (r) Modified (70).

Table 5: The normal decoding rates of several experiments.

No.	The range of bits	The case of bytes	NDR (%)
1	[1, 56]	7 bytes	**100**
2	[1, 64]	8 bytes	**100**
3	[1, 72]	9 bytes	**100**
4	[1, 80]	10 bytes	0.00
5	[1, 73]	9 bytes + 1 bit	27.45
6	[1, 74]	9 bytes + 2 bits	3.92
7	[2, 73]	7 bits + 8 bytes + 1 bit	19.61
8	[8, 73]	1 bit + 8 bytes + 1 bit	70.59
9	[9, 73]	8 bytes + 1 bit	**100**
10	[9, 80]	9 bytes	**100**
11	[9, 81]	9 bytes + 1 bit	15.69
12	[16, 88]	1 bit + 9 bytes	66.67
13	[17, 81]	8 bytes + 1 bit	**100**
14	[17, 88]	9 bytes	**100**
15	[17, 89]	9 bytes + 1 bit	68.63
16	[24, 96]	1 bit + 9 bytes	56.86
17	[25, 89]	8 bytes + 1 bit	**100**
18	[25, 96]	9 bytes	**100**
19	[25, 97]	9 bytes + 1 bit	60.78

Table 6: The success rates of decoding under four distributions of 9 code words.

Types	Groups	Code word distribution	NDR (%)
1M + 1I	4	1 – 3 – 5 – 7 – 9 – 11 – 13 – 15 – 17 2 – 4 – 6 – 8 – 10 – 12 – 14 – 16 – 18 ..... 4 – 6 – 8 – 10 – 12 – 14 – 16 – 18 – 20	90.19
2M + 1I	8	1 – 2 – 4 – 5 – 7 – 8 – 10 – 11 – 13 2 – 3 – 5 – 6 – 8 – 9 – 11 – 12 – 14 ..... 8 – 9 – 11 – 12 – 14 – 15 – 17 – 18 – 20	91.91
3M + 1I	10	1 – 2 – 3 – 5 – 6 – 7 – 9 – 10 – 114 2 – 3 – 4 – 6 – 7 – 8 – 10 – 11 – 1 ..... 10 – 11 – 12 – 14 – 15 – 16 – 18 – 19 – 20	97.06
9M + 0I	12	1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 ..... 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 20	100.00

Table 7: Scoring system in the subjective evaluation.

Scores	Descriptions
4	Comfortable
3	Moderate
2	Poor
1	Disastrous

following Table 7 to evaluate the user experience of mini program codes. There are four levels in the scoring system including disastrous, poor, moderate, and comfortable corresponding to the scores from 1 to 4.

Finally, 33 questionnaires were collected and all of them are valid. The statistics of the subjective evaluation are shown in Table 8. The frequency of samples is too high and $D\xi$ of all samples is very small, so the actual reference significance of $D\xi$ is not significant. Therefore, the evaluation of the proposed scheme considers the average scores $E\xi$. The statistical analysis results are shown in Figure 8. We can draw the following conclusions from the histogram:

(1) The average scores of normal codes and modified codes are 3.61 and 3.40, respectively, with a decline rate of 5.81%. It indicates that the modified codes have little impact on the use of mini program codes. Furthermore, it shows that our scheme is effective.

(2) With the same embedding capacity, the embedding effect in the encoding region is better than that in the edge patch.

(3) Excluding some human factors, the embedding effect decreases with the increment of embedding capacity.

Table 8: The subjective evaluation result.

Capacity	Scores				$E\xi$	$D\xi$
	4	3	2	1		
0	120	30	11	4	3.61	1.1295
10	116	34	11	4	3.59	1.1023
10*e*	92	53	13	7	3.39	0.9933
20	104	46	11	4	3.52	1.0087
20*e*	82	66	10	7	3.35	0.8821
30	81	58	19	7	3.29	0.9245
40	102	44	12	7	3.46	1.0803
50	98	42	19	6	3.41	1.0697
60	87	51	20	7	3.32	0.9892
70	83	55	23	4	3.32	0.9012

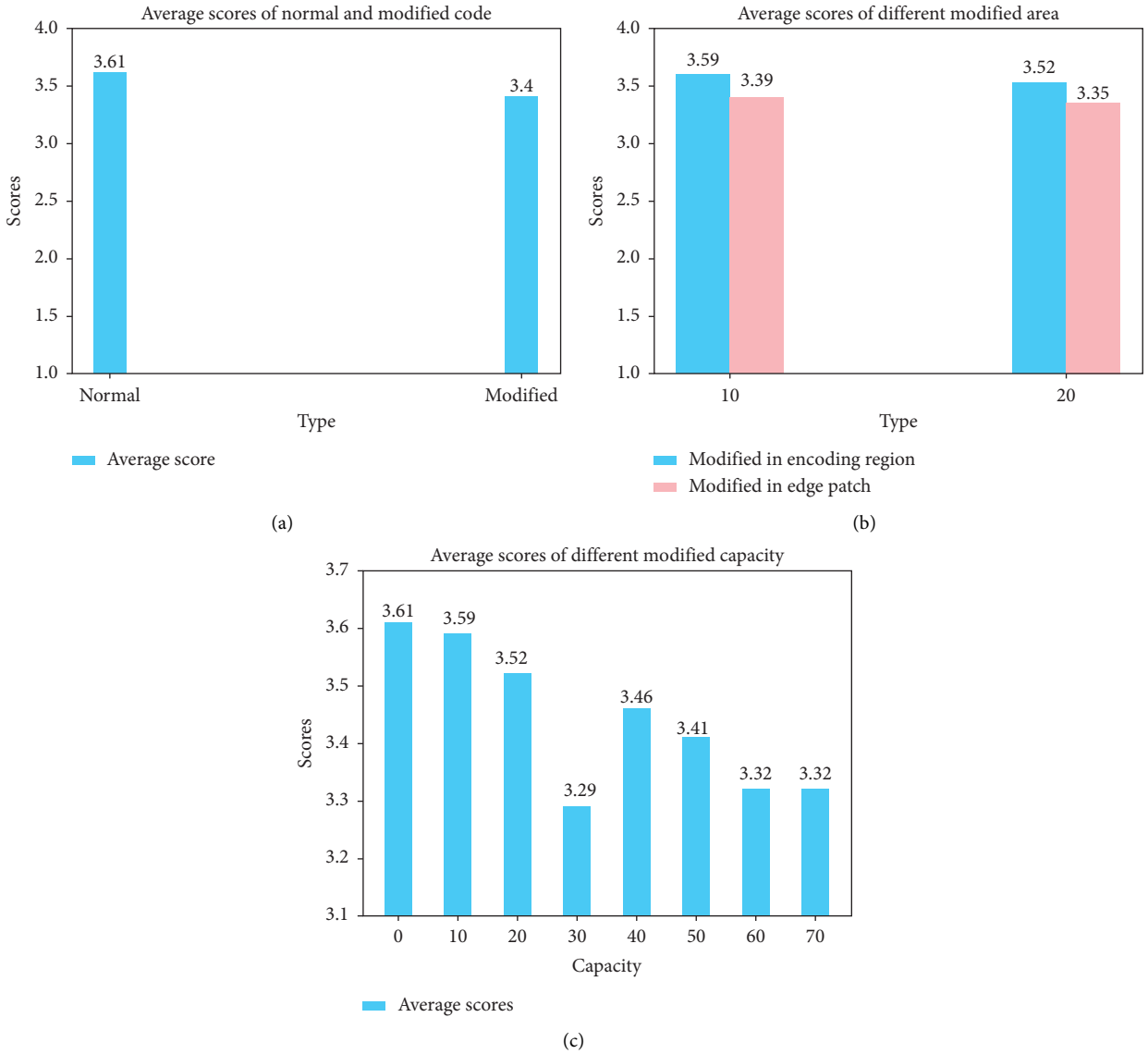


Figure 8: The statistical analysis results. (a) Different codes. (b) Different area. (c) Different capacity.

5. Conclusions and Future Work

In this paper, based on the research of mini program codes, a data-hiding scheme for mini program codes is proposed, and the embedding capacity of the proposed scheme is tested. The proposed scheme is the first to realize information hiding based on mini program codes. This research could deal with the security issues of mini program code in advance. At present, we adopt a bit replacement technique to achieve data hiding based on mini program codes. When the embedding capacity is less than 9 code words, the secret messages could be recovered. Meanwhile, mini program codes still can be decoded correctly. This is the simplest embedding method. Other steganography coding methods could also be applied, such as syndrome-based data-hiding techniques. In practical application, the secret message could be protected by channel coding to deal with that the mini program code goes through a noisy channel. In future work, we will study different versions and error correction levels of the mini program code. More modification techniques need to be studied to improve embedding capacity, such as syndrome-based data-hiding techniques. Furthermore, we will study the way to resist steganography analysis to make the data-hiding schemes based on mini program codes more secure.

Data Availability

Some or all data, models, or code generated or used during the study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was funded by the program of the National University of Defense Technology and the National Natural Science Foundation of China (no. 61602491). The authors would like to thank TopEdit (https://www.topeditsci.com) for its linguistic assistance during the preparation of this manuscript.

References

[1] Miniapp Standardization, "White paper w3c first public working draft," 2019, https://www.w3.org/TR/mini-app-white-paper/.

[2] Aladdin Institute, "White paper on the internet development of the mini program in the first half of 2020," 2020, http://www.aldzs.com/viewpointarticle?id=11527.

[3] N. F. Johnson, Z. Duric, S. Jajodia, and N. Memon, "Information hiding: steganography and watermarking-attacks and countermeasures," *Journal of Electronic Imaging*, vol. 10, no. 3, p. 825, 2001.

[4] W. Luo and J. H. Fangjun 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.

[5] C.-C. Chang, C.-C. Lin, C.-S. Tseng, and W.-L. Tai, "Reversible hiding in DCT-based compressed images," *Information Sciences*, vol. 177, no. 13, pp. 2768–2786, 2007.

[6] Y. K. Lin, "High capacity reversible data hiding scheme based upon discrete cosine transformation," *Journal of Systems and Software*, vol. 85, no. 13, pp. 2395–2404, 2012.

[7] Y.-X. Sun, B. Yan, J.-S. Pan, H.-M. Yang, and N. Chen, "Reversible data hiding in encrypted color halftone images with high capacity," *Applied Sciences*, vol. 9, no. 24, p. 5311, 2019.

[8] X. Yan, Y. Lu, L. Liu, and X. Song, "Reversible image secret sharing," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3848–3858, 2020.

[9] X. Yan, Y. Lu, C. Nung Yang, X. Zhang, and S. Wang, "A common method of share authentication in image secret sharing," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, p. 1, 2020.

[10] N. Teraura and K. Sakurai, "Information hiding in subcells of a two-dimensional code," in *Proceedings of the 1st IEEE Global Conference on Consumer Electronics (GCCE 2012)*, Tokyo, Japan, October 2012.

[11] Y. J. Chiang, P. Y. Lin, R. Z. Wang, and Y. H. Chen, "Blind QR code steganographic approach based upon error correction capability," *KSII Transactions on Internet and Information Systems*, vol. 7, no. 10, pp. 2527–2543, 2013.

[12] J.-S. Pan, W. Li, C.-S. Yang, and L.-J. Yan, "Image steganography based on subsampling and compressive sensing," *Multimedia Tools and Applications*, vol. 74, no. 21, pp. 9191–9205, 2015.

[13] S. Wan, Y. Lu, X. Yan, and L. Liu, "Visual secret sharing scheme with (*k*, *n*) tthreshold based on QR codes," in *Proceedings of the 2016 12th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, vol. 1, pp. 374–379, Hefei, China, October 2017.

[14] L. Tan, Y. Lu, X. Yan, L. Liu, and X. Zhou, "Xor-ed visual secret sharing scheme with robust and meaningful shadows based on QR codes," *Multimedia Tools and Applications*, vol. 79, no. 9, pp. 5719–5741, 2019.

[15] Y. Cheng, Z. Fu, and B. Yu, "Improved visual secret sharing scheme for qr code applications," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2393–2403, 2018.

[16] X. Yan, S. Wang, A. A. Abd El-Latif, and X. Niu, "New approaches for efficient information hiding-based secret image sharing schemes," *Signal, Image and Video Processing*, vol. 9, no. 3, pp. 499–510, 2015.

[17] A. A. A. El-Latif, B. Abd-El-Atty, S. E. Venegas-Andraca, and W. Mazurczyk, "Efficient quantum-based security protocols for information sharing and data protection in 5G networks," *Future Generation Computer Systems*, vol. 100, pp. 893–906, 2019.

[18] Lincolnlin, Endyxu, and Changoran, *Mini program Design: Bloom like Chrysanthemums*, 2017, https://cloud.tencent.com/developer/article/1005027.

[19] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[20] S. Chen, *Wechat Mini program Code private Coding protocol Analysis*, 2019, https://zhuanlan.zhihu.com/p/85164898.

[21] P. C. Huang, C. C. Chang, Y. H. Li, and Y. Liu, "High-payload secret hiding mechanism for QR codes," *Multimedia Tools and Applications*, vol. 78, no. 10, pp. 22331–22350, 2019.

[22] X. Yan, S. Wang, A. El-Latif, X. Niu, and Z. Wei, "A new assessment measure of shadow image quality based on error diffusion techniques," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 4, no. 2, pp. 118–126, 2013.