

# Quick Response Code Binary Research Based on Basic Image Processing

Yun Wu,<sup>1</sup> Shaoyong Yu,<sup>2\*</sup> and Mei Yang<sup>2</sup>

<sup>1</sup>College of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China

<sup>2</sup>School of Mathematics and Information Engineering, Longyan University, Longyan City 364012, China

(Received October 15, 2018; accepted February 8, 2019)

**Keywords:** QR code, binarization, block truncation coding

The goal of this paper is to binarize quick response (QR) codes with basic image processing methods, so that they can be well recognized in extreme cases. In this paper, we avoid the complex computation that comes from sophisticated algorithms, and employ block truncation coding (BTC) to simplify the computation to make processing fast. The final results show that complex algorithms are not required in QR code binarization, and the basic image processing algorithm can achieve faster and better processing results.

## 1. Introduction

At present, the applications of quick response (QR) codes are booming. Phone numbers, web addresses or postal addresses can be converted into QR codes for mobile devices to read. However, changes in light and shadow in the external environment during image capture often result in undeciphered or uninterpreted QR codes, such as insufficient or uneven light. Therefore, various methods have been proposed to solve this problem, and these methods are nothing more than the methods put together by more complex algorithms such as the gray estimation formula or the Otsu method. Their computation is large and the program code is complex, which cannot meet modern people's requirement of high processing speed. Thus, we propose to replace the complicated algorithms with the basic image processing technique. The following is a paper selected for comparison.<sup>(1)</sup>

The global threshold method is a common and relatively simple binarization method, such as the Otsu and histogram bimodal methods. However, these algorithms can only produce a global threshold, and when the whole image is binarized, it may produce an all-black or all-white image, which cannot be adjusted. The above problems can be effectively solved by the binarization method based on the background grayscale.

Background grayscale binarization is effective in the image processing of nonuniform brightness. However, the expansion of the background grayscale matrix requires a great deal of computation and reduces the efficiency of binarization. In Fig. 1, the steps after modification are as follows:

---

\*Corresponding author: e-mail: 82018007@lyun.edu.cn  
<https://doi.org/10.18494/SAM.2019.2165>

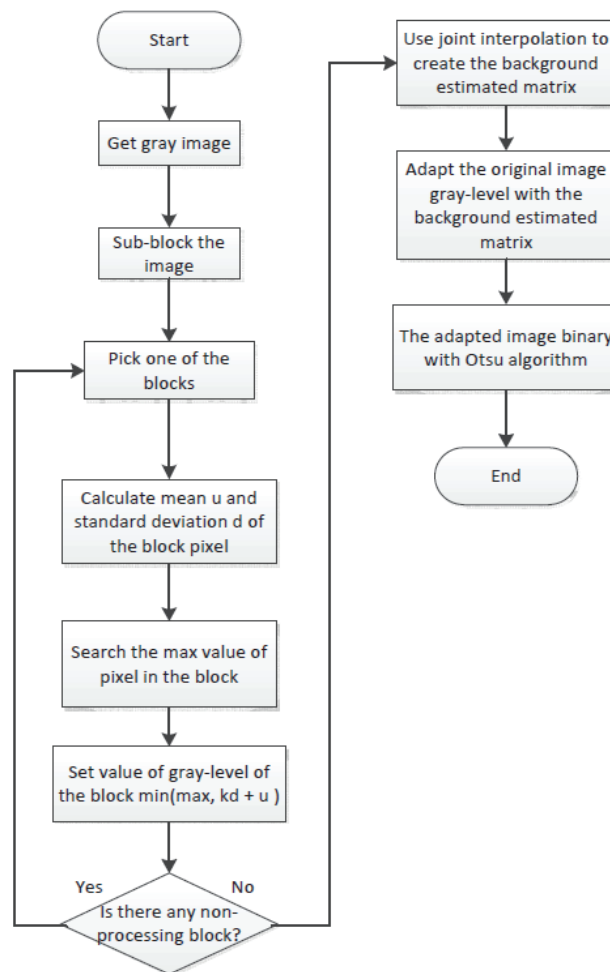


Fig. 1. Process flow diagram.

- (1) Divide the block according to the original image.
- (2) Calculate the value of each block through grayscale estimation.
- (3) Follow the combined image interpolation method to generate adjustable background grayscale images and make them as large as the original image.
- (4) By replacing the original image with an adjustable background grayscale image, the noise can be removed, and the image can be enhanced. Overcome the shortcomings of threshold selection and image brightness inequality.
- (5) Binarize using the Otsu method.

The grayscale matrix of the image is divided into the matrix set, and matrix  $B$  is one of the submatrices, as shown in Eq. (1). The size of the submatrix should preserve details and suppress noise. According to the experiment, select  $l = 1/16$ , where  $M$  and  $N$  are the width and height of the original image, respectively, and  $l$  is the number of submatrices.

$$B_{m*n} = \begin{bmatrix} B_{11} & \dots & B_{1n} \\ \dots & \dots & \dots \\ B_{m1} & \dots & B_{mn} \end{bmatrix} \quad (1)$$

$$i = \left\lfloor \frac{M}{m} \right\rfloor, j = \left\lfloor \frac{N}{n} \right\rfloor, l = i * j$$

In Eq. (2), the background gray level of each block is calculated, where  $u$  is the average grayscale value of the original image,  $u'$  is the average grayscale value of each block of the image, and  $k$  is the standard deviation of each block of the image.

$$t = u' - u + k\sigma \quad (2)$$

Finally, the size of the grayscale matrix is selected using the gray order value of each block, and  $k$  is the adjustment coefficient, from 0 to 1. In the experimental analysis,  $k = 0.3$  is the most appropriate.

$$R_{i*j} = \begin{bmatrix} t_{11} & \dots & t_{1j} \\ \dots & \dots & \dots \\ t_{i1} & \dots & t_{ij} \end{bmatrix} \quad (3)$$

If the block is low in brightness or in background, the values of  $u$  and  $c$  will be small, while the background grayscale  $t$  will be smaller. In contrast, if the block is high in brightness and the background is excessive, the values of  $u$  and  $c$  will be large, and the background grayscale  $t$  will be larger.

### 1.1 Increase in the combined interpolation of background grayscale

The original image is divided into different regions, which use different interpolation algorithms to achieve the image quality maintained after magnification, while reducing the purpose of the calculation. The general method of region segmentation requires image segmentation. Considering the complexity of image segmentation, there is no uniform algorithm for all images. In this paper, the first segmentation image is avoided, but in the original image, the variance of the pixel value adjacent to the interpolation point is calculated for the first time. If the variance is less than the threshold value  $T$ , the average of the four-point pixel values,  $E$ , is directly used as the pixel value for interpolation calculation. In contrast, the calculation is based on bicubic interpolation. The variance  $Var$  is calculated as

$$Var = (E - f_{11})^2 + (E - f_{12})^2 + (E - f_{21})^2 + (E - f_{22})^2, \quad (4)$$

where  $f_{11}, f_{12}, f_{21}$ , and  $f_{22}$  are the pixel values of the neighboring point of the current interpolation point in the original image.  $E$  denotes the average of the four-point pixel values. To calculate

$E$ , multiplication and division are required five and three times, respectively. To calculate the variance, multiplication and division are required four and seven times, respectively. In total, multiplication and division are required nine and ten times, respectively; this calculation method is relatively simpler than the bicubic interpolation that requires multiplication and division 70 and 45 times, respectively. Although every point of the image should be calculated once in this calculation method, compared with the calculation using the bicubic interpolation method, the calculation complexity is much less. For more image smoothing areas, the reduction in computation complexity is clearer. The following experimental results verify the effectiveness of the algorithm. Selecting the threshold  $T$  is the first step of the algorithm. If the threshold value is small, fewer pixels avoid the bicubic interpolation, and the image quality is closer to the result of bicubic interpolation, whereas the amount of calculation is still large. In contrast, the larger the threshold, the larger the number of pixels will avoid the bicubic interpolation, which reduces the amount of computation but makes the image blurred.  $Var$  reflects the smoothness of neighboring pixels. It is difficult to define a uniform threshold for different images. In different environments, the threshold value should be adjusted frequently to meet the actual application requirements. The experimental experience suggests that the reductions in computational complexity and image quality are reasonable at the threshold  $T = 20$ . In practice, the threshold should be adjusted to about 20 primarily to obtain the most reasonable image. In this paper, the threshold  $T = 20$  is selected in the experiment. The algorithm flow is illustrated in Fig. 2.

## 1.2 Adjustment of grayscale and binarization of the original image

The adjusted image grayscale matrix  $I_r$  is obtained by subtracting the background grayscale matrix from the previous step by using the original image grayscale matrix  $I_g$ . Then, the part of the image that is very dark owing to subtraction is adjusted. Finally, the Otsu method is used to binarize the adjusted image grayscale matrix.

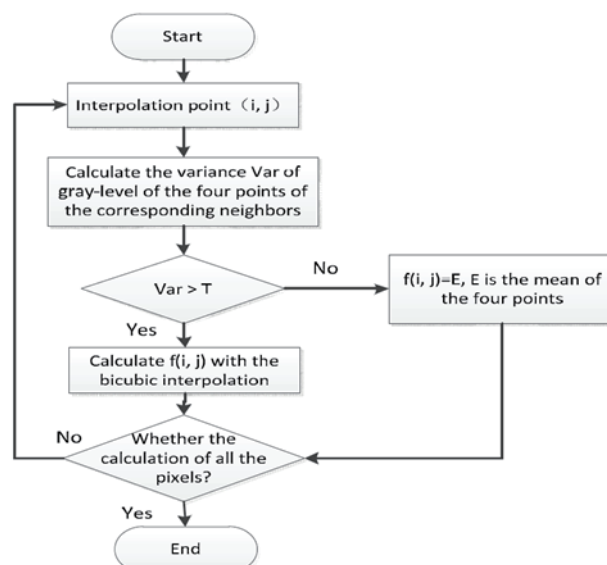


Fig. 2. Joint interpolation process.

The Otsu method<sup>(2)</sup> is the automatic threshold value determination principle that is commonly employed to select the threshold value of image binarization. Undoubtedly, the use of the Otsu method is not limited to the image binarization. It can be used for all the selected thresholds. Since Otsu is a threshold selection criterion, before using Otsu, we must obtain the distribution of the state space that we want to segment with a threshold.

In the case of image binarization using Otsu, since pixel values are to be segmented, the distribution of pixel values (i.e., histogram of pixels) should be obtained. The process of image binarization using Otsu is as follows:

Step 1. Obtain the histogram of the video (Fig. 3).

Step 2. Find the optimal threshold value from the histogram using the Otsu method.

Step 3. Binarize the image with the threshold found (Fig. 4).

Otsu's task is to find the optimal threshold from the state distribution, i.e., Step 2 (Fig. 5).

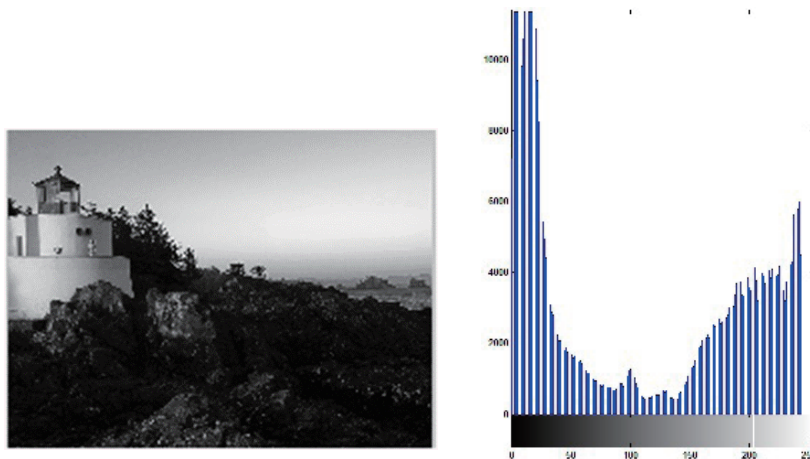


Fig. 3. (Color online) Grayscale image and histogram.

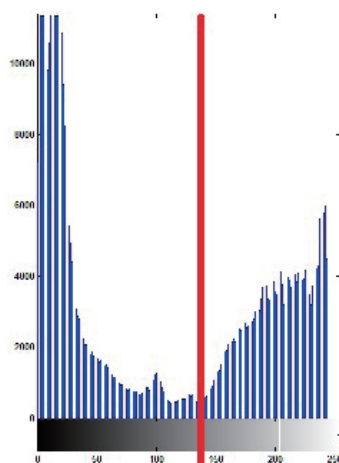


Fig. 4. (Color online) Schematic diagram not representing the Otsu threshold position of the histogram.



Fig. 5. After Otsu-based binarization.

## 2. Improved Method

In this paper, the basic image processing is used to make the QR code interpreted promptly and correctly and to increase the interpretation rate. First, histogram equalization is used to enhance the image contrast to improve the later binarization. Subsequently, block truncation coding (BTC) was used to divide the image into several small blocks for binarization. Finally, noise is removed to achieve a good QR code binarization image. The flow is shown in Fig. 6.

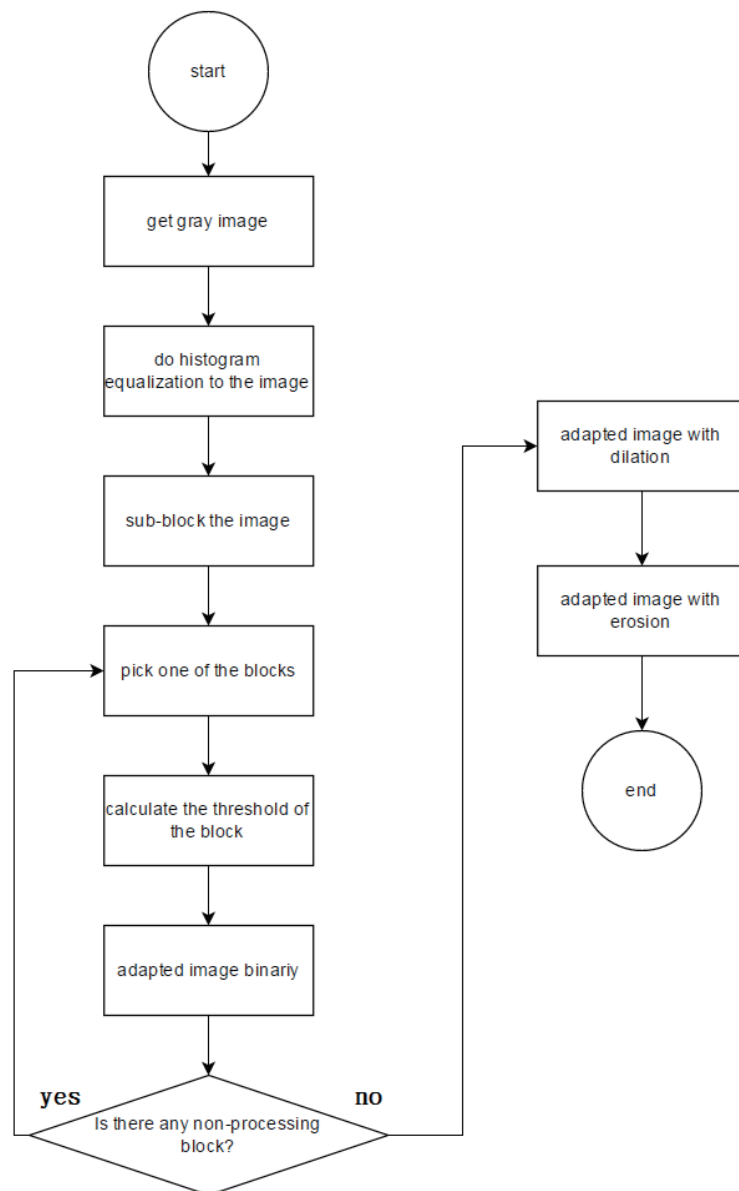


Fig. 6. Process flow diagram.

### 2.1 Histogram equalization<sup>(3)</sup>

Histogram equalization is an image processing program designed using statistical methods. It helps to spread the color distribution of statistical histograms evenly in the histogram, i.e., to make the histogram distribution of a graph even. Likewise, a look-up table (LUT) is also employed.

- (a) is the original gray-level concentration histogram;
- (b) is a method of partially reinforcing contrast;
- (c) is a method of equalizing the entire image, as shown in Fig. 7.

Here, the (c) method is employed. After all, the whole picture is to be processed rather than the small areas. The method is expressed as shown in Fig. 8.

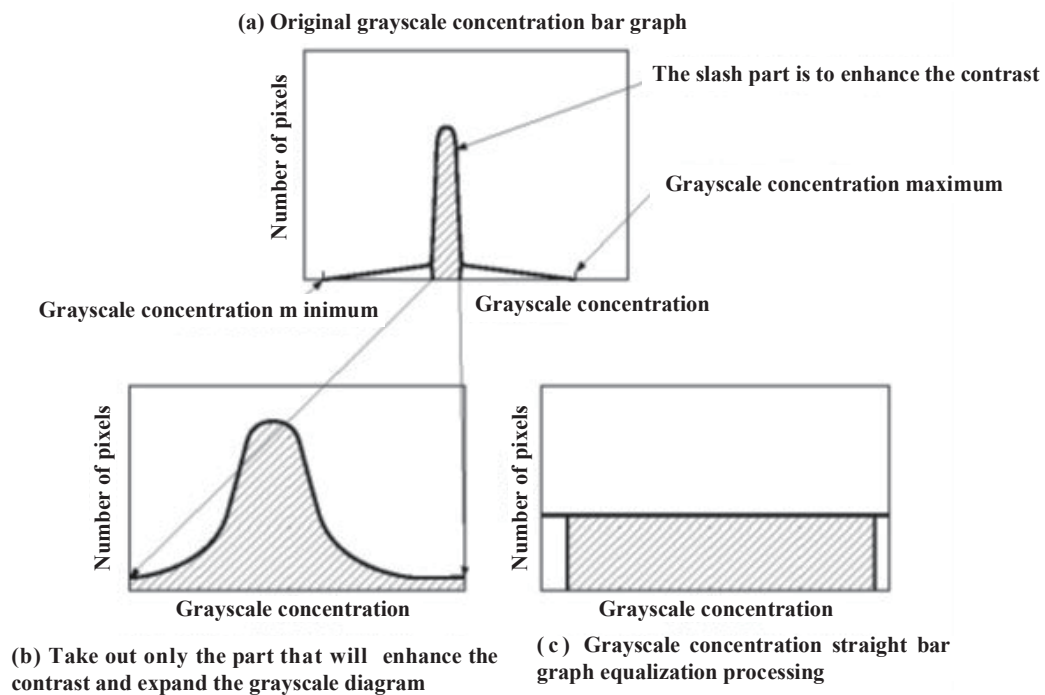


Fig. 7. Method of equalization.

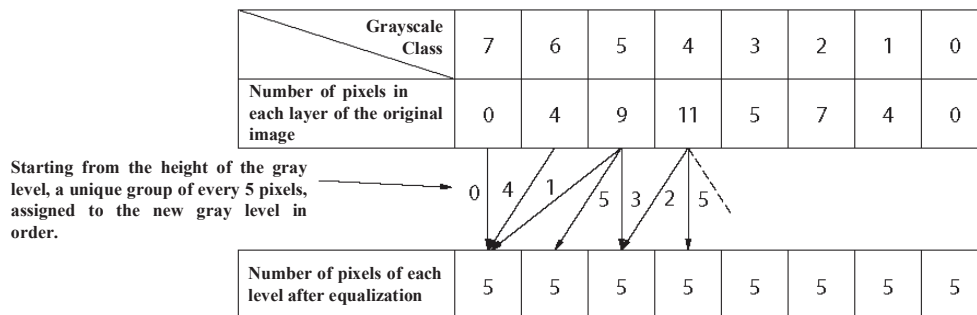


Fig. 8. Method of image processing.

### 2.2 Dynamic critical binarization<sup>(3)</sup>

According to the results of the original image equalization, if all the blocks are binarized with a single critical value, the data that we want may be eliminated and the amount of data that we do not need will be increased instead. Therefore, we use different thresholds for each block.

The procedure of the dynamic critical value method is to change the critical value of each pixel. As shown in Fig. 9, the image is first divided into blocks, and the critical value is selected at each grid point. Second, as shown in Fig. 10, the critical value of each grid point is used to perform quadratic linear interpolation calculation to obtain the critical value of each pixel. The critical value  $t(x,y)$  of each pixel is equal to four grid points around the pixel  $(x,y)$  to be calculated, which is calculated using Eq. (5) according to the corresponding distance ratio.

$$t(x,y) = (1-q)\{(1-p)*t(u,v) + p*t(u+1,v)\} + q\{(1-p)*t(u,v+1) + p*t(u+1,v+1)\} \quad (5)$$

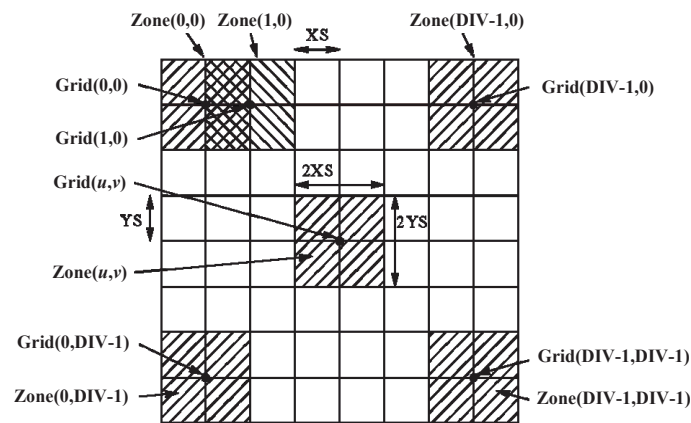


Fig. 9. Dynamic critical value method.

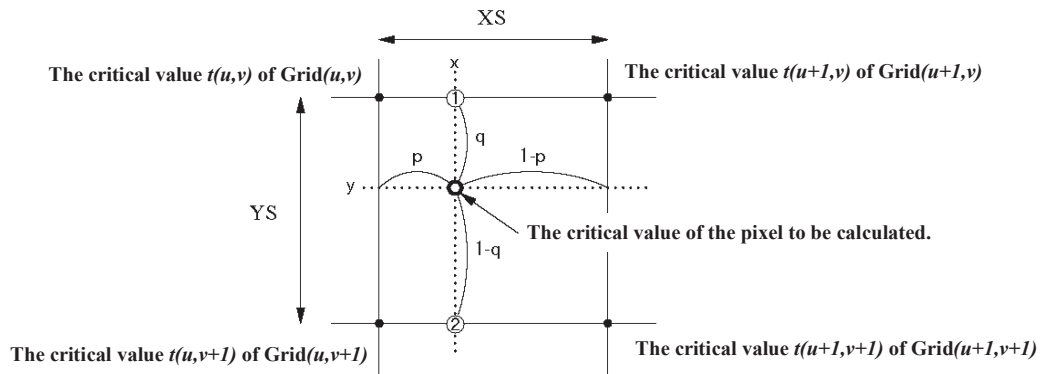


Fig. 10. Quadratic linear interpolation.



### 2.3 Erosion and expansion<sup>(3)</sup>

From the image after equalization binarization, we can find a lot of noise around the image. Most of these noises are small noises with only a few pixels together. Therefore, we use expansion and erosion to remove the small noises and make the whole image cleaner. Expansion refers to the point at which one of the 8 neighboring pixels around a pixel is 1, and the pixel is set to 1 and the others to 0. Erosion refers to the point at which one of the 8 neighboring pixels around a pixel is 0, and the pixel is set to 0 and the others to 1. As shown in Fig. 11, when the image undertakes the processing from expansion to erosion, black noise is eliminated in the expansion. As shown in Fig. 12, white noise is eliminated during erosion when the image undertakes the processing from erosion to expansion.

### 3. Experimental Results and Analysis

To verify the necessity of our proposed method, we use the 8-megapixel lens commonly used on ordinary phones. We take 4 images of QR codes in a low light source and an uneven light source for the comparison of the results. The size of the original video is  $512 \times 512$  overall. During BTC processing, since we do not know which block size is the most appropriate result, we conduct a processing for different block sizes to find the most suitable size.<sup>(4-7)</sup>

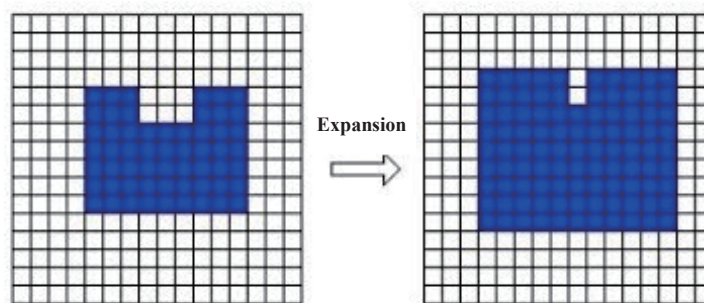


Fig. 11. (Color online) Diagram of expansion.

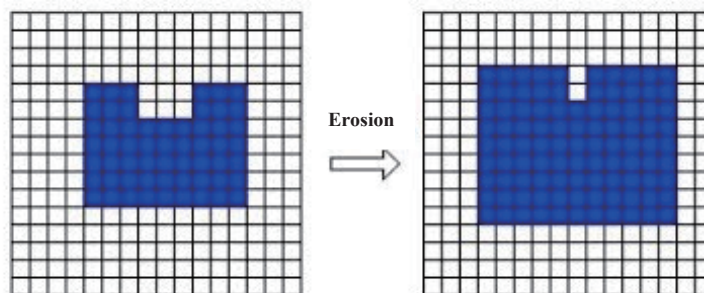


Fig. 12. (Color online) Diagram of erosion.

### 3.1 Full dark image

We take 4 images of QR codes in the low light source for the comparison of the results. Figures 13–16 show four images of test processes.

### 3.2 Semidark image

We take 4 images of QR codes in the uneven light source for the comparison of the results. Figures 17–20 show four images of test processes. In Table 1, we compare the time spent.

According to the full shadow image, the processing range is too large or too small, which makes the noise even louder. The smaller the scale, the less the data will be lost. The Otsu method has no way of accurately finding the foreground and background in processing semidark images. However, the quality of images processed by the Otsu method is much higher. Although the image sizes of  $6 \times 16$  and  $32 \times 32$  can be fully interpreted, the image information of  $32 \times 32$  will be more damaged. Although the APP can be successfully interpreted faster than  $16 \times 16$ , we will choose  $16 \times 16$  as the final result.



Fig. 13. (Color online) Full dark QR code image test 1.



Fig. 14. (Color online) Full dark QR code image test 2.



Fig. 15. (Color online) Full dark QR code image test 3.



Fig. 16. (Color online) Full dark QR code image test 4.



Fig. 17. (Color online) Semidark QR code image test 1.



Fig. 18. (Color online) Semidark QR code image test 2.

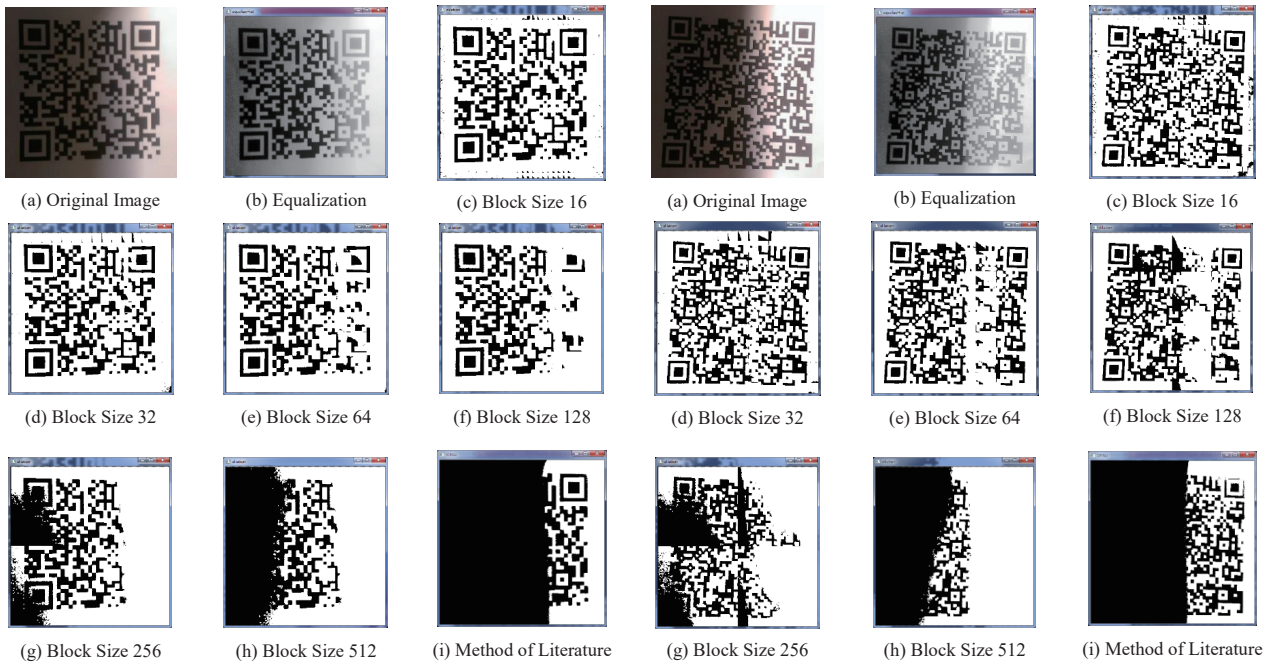


Fig. 19. (Color online) Semidark QR code image test 3.

Fig. 20. (Color online) Semidark QR code image test 4.

Table 1  
4 QR code images in full dark and semidark with processing time.

	(1)	Otsu <sup>(2)</sup>	16	32	64	128	256	512
Full dark image (ms)								
Fig. 13	208	74	93	88	82	83	81	83
Fig. 14	207	68	94	88	83	81	83	84
Fig. 15	208	69	94	88	82	84	83	83
Fig. 16	210	69	94	88	83	82	82	82
Correct rate	75%	75%	100%	100%	100%	100%	100%	0%
Semidark image (ms)								
Fig. 17	208	66	90	84	84	83	84	83
Fig. 18	207	74	91	82	80	78	81	82
Fig. 19	212	71	92	82	81	80	75	82
Fig. 20	212	72	91	80	76	77	75	75
Correct rate	0%	0%	100%	100%	0%	0%	0%	0%

#### 4. Conclusion

The speed and image quality of QR code binarization based on the basic image processing algorithm are not inferior to those of the complex algorithm. If you want to improve the speed further, you may need to do some processing on the hardware side. There should not be much room for improvement on the algorithmic instruction cycle. Therefore, in the next stage, research on hardware should be of greater significance.

## Acknowledgments

This work was supported by the National Science Foundation of Fujian Province of China (No. 2018J01575) and the Project of Scientific Climbing Plan by Xiamen University of Technology No. XPDKQ18015.

## References

- 1 Y. Zhang, T. Gao, D. Li, and H. Lin, 2012 2nd Int. Conf. Consumer Electronics, Communications and Networks (CECNet) (IEEE, 2012) 21–23.
- 2 Otsu Function Introduction: <http://www.dotblogs.com.tw/dragon229/archive/2012/12/28/86092.aspx>
- 3 S. Inoue and N. Yagi: C Language Digital Image Processing (2010) 27.
- 4 Z. Xiong, H. Cuiqun, L. Guodong, and L. Zhijun: 2nd Int. Conf. Signal Processing Systems (ICSPS) 3 (IEEE, 2010) 317–320.
- 5 J. Zhou, Y. Liu, and P. Li: Int. Conf. Multimedia Technology (ICMT) (IEEE, 2010) 1–4.
- 6 J. Bernsen: Proc. 8th Int. Conf. Pattern Recognition (IEEE, 1986) 1251–1255.
- 7 Bicubic interpolation and matlab design: <http://wenku.baidu.com/view/34dea625192e45361166f504.html>

## About the Authors



**Yun Wu** received her Ph.D. degree from Xiamen University in 2007. Her research interests are in the areas of artificial intelligence and big data. Her scientific contribution to AI has more to do with soft computing and clustering algorithms. (yw@xmut.edu.cn)



**Shaoyong Yu** received his Ph.D. degree from Xiamen University in 2017. He is also an associate professor in the School of Mathematics and Information Engineering of Longyan University. His research interests are in the areas of computer vision and deep learning. (82018007@lyun.edu.cn)



**Mei Yang** is a lecturer and teacher in the School of Mathematics and Information Engineering of Longyan University. Her research interests are in the areas of software development and artificial intelligence. (wulv07@163.com)