Journal of Theoretical and Applied Information Technology
15th November 2022. Vol.100. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

QR CODE DETECTION AND RECTIFICATION USING PYZBAR AND PERSPECTIVE TRANSFORMATION

FRANCISCO CALVIN ARNEL FERANO[1], JAN KEANE OLAJUWON[2], GEDE PUTRA KUSUMA[3]

Computer Science Department, BINUS Graduate Program – Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia, 11480

E-mail: [1]francisco.ferano@binus.ac.id, [2]jan.olajuwon@binus.ac.id, [3]inegara@binus.edu

ABSTRACT

The increasing and widespread use of QR codes in everyday life has led to an increase in the activity of counterfeiting the QR code itself. Authentication of QR codes is not easy. For this reason, a proper QR code detection method needs to be given to the QR code image. However, the performance of QR code detection is highly dependent on the pre-processing method used. An appropriate pre-processing method is needed to be given to the QR code image so that the QR code becomes easier and faster to detect. This paper proposes the preparation of various pre-processing methods for a QR code dataset containing 1350 QR code photos so that they can be detected properly using Pyzbar. Several pre-processing methods are chosen to be performed in the experiment, which are: images gray scaling, conversion into RGB images, binarization through thresholding, gaussian blur combined with Otsu threshold, Otsu threshold only, a combination of the five methods, and a combination of the three best methods. This set of pre-processing methods improves detection performance using Pyzbar which allows QR code datasets to be rectified properly using a perspective transformation. The experimental results show that the proposed method produces a good percentage of detection results with the best value of 95%. All detected QR codes have been successfully rectified using perspective transformation and produced good alignment accuracy with the best value of 85,661%. The accuracy increases after being evaluated with one-pixel misalignment tolerance with the best value of 92.144%.

Keywords: *QR Code Detection, Image Pre-Processing, Pyzbar Library, Image Rectification, Perspective Transformation.*

1. INTRODUCTION

QR code is a kind of two-dimensional barcode that is used to symbolize facts in square-fashioned styles. QR code stands for Quick Response Code, which may be decoded quickly. The QR code we are currently using was designed by a Japanese company called Denso Wave in September 1994 which was originally used to tune production stock at a car company. QR code is composed of a black module organized in a square sample that represents terrific 2-dimensional facts, which may be studied from each vertical and horizontal direction. QR codes have many features such as high data encoding, stain and damage resistance, fast reading, small print size, 360 degrees reading, and structural flexibility of the application [1].

The advantage that QR Code provides is its ability to have a large record garage. The QR code can store 7,089 records for numbers, 4,296 records for each letter and number, 2,953 binary bytes (eight bits), and 1,817 Japanese Kanji / Kana symbols, good encoding scope, can be published in a small size, readable very fast, capable of correcting errors, resistant to danger and can still be read with a perspective angle of 360 degrees. The QR code is also built outside of car companies due to the ease of use and speed of the learning method and extra garage capabilities compared to traditional UPC barcodes [2].

Journal of Theoretical and Applied Information Technology
15th November 2022. Vol.100. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 1: Example of QR Code

QR codes are objects that have been widely used in everyday life and have become an important thing for various sectors lately. For example, in [3], a system for tracking a person's location by scanning a QR code has been proposed. In our daily life, QR codes are also used to assist in payment methods, checking product authenticity, or monitoring the number of goods and others.

The quick response (QR) code has been used in numerous quarters in many countries, such as Indonesia. Due to the increasing use of QR codes, the number of QR code counterfeiting practices is also increasing. This makes the study topic interesting and needs to be reviewed further to support the effective use and security of every transaction with QR codes.

Authentication of QR codes is not easy, therefore it takes a qualified QR code detection method. The Pyzbar is chosen to do the task. Pyzbar is a Python package developed by L. Hudson [4]. It is a module that is liable for analyzing and interpreting 1-D barcodes or QR codes effortlessly and it calls for PIL module to be able to feature properly. However, the performance of the detection process is highly dependent on the appearance of the QR code images. Variations in illuminations, angles, and scaling, make the appearance of the QR code images vary. Therefore, a proper pre-processing method needs to be applied. The pre-processing method must be able to work effectively and efficiently to speed up and improve the accuracy of the detection process, so we made improvements to the display and focused on objects, namely QR codes.

In this study, we assessed several pre-processing method strategies for QR code image enhancement to address current difficulties. By providing the right pre-processing method, the detection procedure may be executed with good performance and high accuracy. A QR code dataset containing 1350 QR code photos of QR codes is used.

After the QR codes in the dataset are detected using Pyzbar, the rectification process using perspective transformation can be performed to produce the frontal view of the QR code dataset, which is done to support the QR code authentication process

To find out whether the produced rectified QR code images are true to the pattern alignment, a pattern alignment comparison process between the rectified QR code images and the original images of the QR code in the scanned format is performed. Several contributions in this study are: the study proposes the preparation of various pre-processing methods for a QR code dataset and provides performance results of each pre-processing method in supporting the performance of the QR code detection process. These processes are then followed by a rectification process to produce rectified images of the QR codes. A method to measure the alignment accuracy of the rectified QR code's pattern alignment is designed in this study.

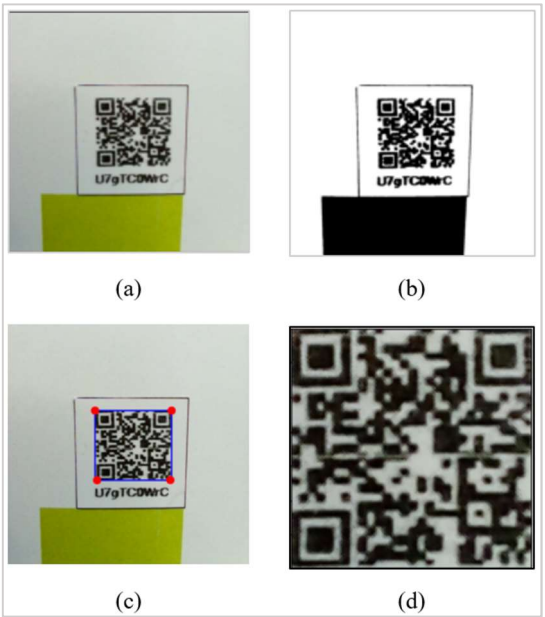

Figure 2: QR code photo in the dataset (a), pre-processed image (b), detected QR code (c), rectified QR code (d)

2. RELATED WORKS

Many studies try to analyze and develop QR Code detection algorithms, but all these detection methods are built on one flat principle, namely the structure of the QR code itself. In [5], analysis and research have been carried out using a QR code with high pixels. The QR code used in the paper is a QR code with a size >= 45 x 45 pixels. The study

explains that a QR code has a structure as shown in Figure 3.

As was done in [6], a QR code with a smaller size has been chosen in this study. Such QR code has a simpler structure as shown in Figure 4.

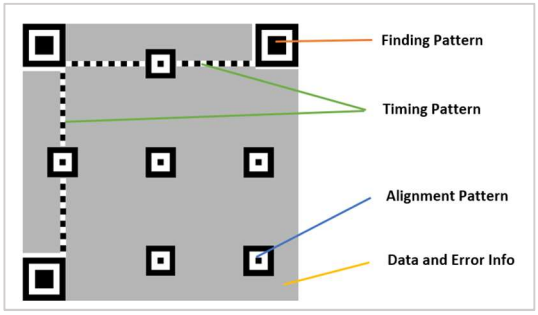

Figure 3: Structure of QR Code

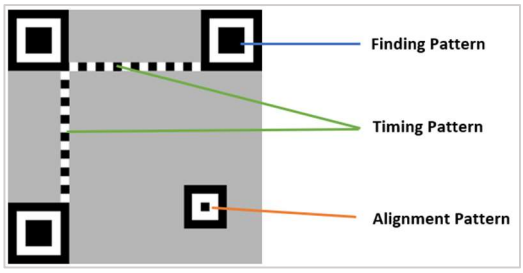

Figure 4: Structure of Lower Size of QR Code

A QR code stores more information than the information already mentioned, however, QR code parts shown in Figure 3 and Figure 4 are the important parts of a QR code that provide important information and features for a QR code detection algorithm.

However, sensing conditions, the environment, and the surface of the board all have an impact on QR code detection processes, resulting in a variety of flaws such as noise pollution, local enhancement, and geometric distortion. According to [7], some parameters become key points to find the best QR detection method, those are time, accuracy, angular perspective, motion and blur, and illumination variance. Since these drawbacks lead to a decrease in recognition rate, image pre-processing of QR code images was required to solve the problems. One of the common problems with QR code datasets is uneven brightness and low illumination. This problem has been tried to be solved by several studies, such as in [8] and [9].

Traditional QR code recognition algorithms have only been used on barcodes printed on flat surfaces. The authors proposed a cylinder 2D barcode recognition algorithm in [10], which was implemented and tested on Android. The emphasis is on how to change and localize the QR code image. As a result, the bilinear transform was first used to correct the QR code image's geometry distortion. The same problem of cylindrical geometric distortion was also studied in [11].

The effects of QR code interference were experimentally investigated by placing paper or any solid object on top of the QR code label. In addition, the QR code tag has been coated with reflective transparent plastic. As a result, we discovered that if the camera captured the QR code completely without obscuring it, the system could read it and display the data in the book. Furthermore, because of the reflective transparent plastic, the system was unable to read the QR code. [12].

In [6], the study proposed an approach to build a QR code recognition algorithm. First, a search space-reducing algorithm is used to locate the QR code's regions of interest. The second step is to localize the QR code pattern to find the location or coordinate of the QR code finding pattern and alignment pattern, with their unique attribute or characteristics. QR code finding pattern is composed of pixels in a ratio of 1:1:3:1:1 and as for the alignment pattern, it was composed of pixels in a ratio of 1:1:1:1:1.

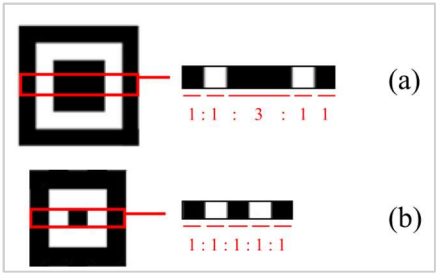

Figure 5: Pixel ratio of finding pattern and alignment pattern. (a) finding pattern. (b) alignment pattern

The algorithm is then followed with a method of exclusion of false-positive data using Principal Components Analysis (PCA), QR code localization, and rectification. The same problem of removing false-positive data was also studied in [13] by the same author. The rectification process in the first study is performed by computing the homograph matrix and applying the geometric correction. The PCA performance is compared with several pattern

classification methods, such as random forest (RF), support vector machine (SVM), and correlation method (CR). PCA achieved the best value in both accuracy and processing time.

In [14], the authors proposed a novel approach to locating 2D QR codes in arbitrary images. The method of the QR code localization was also done by finding three finding patterns. The localization method starts with image conversion to grayscale and is followed by image binarization using adaptive thresholding. Afterward, the search algorithm for the finding patterns was performed. The search algorithm results in group triplets of finding a pattern that forms a right-angled triangle. Using this information, perspective transformation can be done. Lastly, the rectified QR code image was decoded. This study provides a novelty in completing the perspective transformation process, which is done in two alternative ways, by evaluating edge projections of the QR code and by evaluating the overlap of the boundary line and the QR code respectively.

In [15] a novel QR code image binarization method was proposed. The proposed method was built inspired by Bernsen's approach and based on J. Sauvola's adaptive binarization approach, which aimed to retrieve a QR code image that is successfully separated from the background in an image that has uneven illumination. The first step taken was to convert the image into a grayscale image, hence, the brightness of the image can be determined. The step is then followed by mean and standard deviation calculation for each pixel within two windows. Local contrast value within the external windows is also calculated and normalized. Finally, the binarization image is generated using Sauvola-based threshold calculation. The experiments done by the authors show that the approach has advantages over other methods in terms of image clarity and noise reduction.

The study proposed a fast general object detection algorithm, BING, for QR tag positioning in [16]. To compensate for the shortcomings of the BING algorithm, an Adaboost-SVM-based accurate positioning approach was developed. Adaboost-SVM is used for training and prediction after the image has been enhanced with CLAHE, which can significantly reduce training time and improve the prediction precision of Adaboost-SVM. This proposed algorithm has a higher recall and precision rate, with obvious advantages in prediction speed.

Several methods have been proposed to solve the problem of QR code detection and rectification, and several studies, such as study [17], have attempted to apply deep learning methods to complete this task. In essence, QR code detection necessitates the use of an appropriate pre-processing method so that the detection model can recognize and extract the features or information contained in the QR code object. After detecting the QR code, the rectification process can begin by paying close attention to the uniqueness of the corresponding QR code pattern.

Although several studies have examined pre-processing methods to support QR code detection, studies on this matter are still lacking and need to be investigated further. studies on QR codes are mostly done by building a good detection model, without emphasizing the pre-processing stage that is needed before carrying out the detection stage. In contrast to other studies, this study conducted research on QR codes which focused on the pre-processing stage of QR code images. Several pre-processing method strategies for QR code images have been determined to be conducted in this study.

3. PROPOSED METHOD

3.1 Overview

The QR code image identification flow was primarily concerned with image pre-processing, QR code region extraction, correction processing, and QR code decoding. The QR code images are pre-processed to improve image quality so that the QR code detection process can run with good performance. Several pre-processing techniques have been discovered through experiments, and the effectiveness of each method is measured by the number of QR codes detected from a given dataset of QR code images.

Each pre-processed QR code image will then be given to Pyzbar as an input image. Pyzbar produces an output as the coordinates of the QR code located on the image. Using this information, perspective transformation can be performed on every QR code image, resulting in a rectified image of the QR code. The process is then followed by a method to geometrically correct the rotation of the QR code image. As a result, all detected and rectified QR code images are geometrically correct. These geometrically correct QR code images will then be compared to their original generated pattern of the QR code image.

Journal of Theoretical and Applied Information Technology
15th November 2022. Vol.100. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

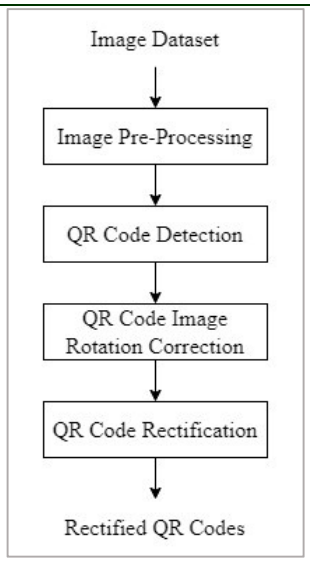

Figure 6: Proposed Method's Flowchart

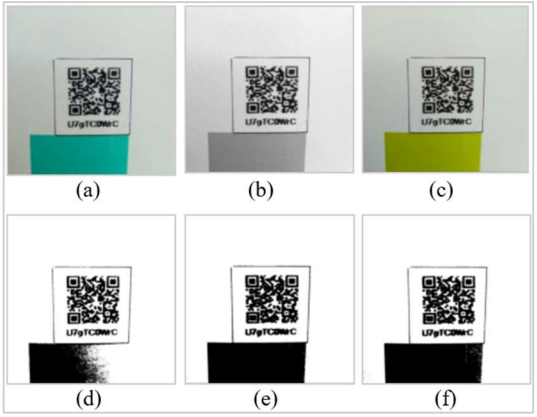

Figure 7: Input Image (BGR) (a), grayscale image (b), RGB image (c), binary image with threshold value = 128 (d), gaussian blur and Otsu threshold image (e), Otsu threshold image (f)

3.2 Image Pre-processing

The QR code image identification flow was primarily concerned with image pre-processing, QR code region extraction, correction processing, and QR code decoding. The QR code images are pre-processed to improve image quality so that the QR code detection process can run with good performance. Several pre-processing techniques have been discovered through experiments, and the effectiveness of each method is measured by the number of QR codes detected from a given dataset of QR code images.

Several pre-processing methods have been chosen to conduct the experiment, which are: converting images into grayscale images, converting images into RGB images, binarization using threshold value = 128, gaussian blur combined with Otsu threshold, and lastly, Otsu threshold without gaussian blur. The additional pre-processing methods that are also conducted in the experiment are the combination of all pre-processing methods mentioned above, and the top three of the pre-processing method assessed by each method's detection percentage.

The use of Grayscale in the dataset is to reduce the coloring dimension to a single dimension. Therefore, QR Code detection can be made easier by changing colors that tend to darken more sharply.

Blurring is the process of making anything less distinct or apparent. In the context of image analysis, this could be taken to mean anything that diminishes or distorts the detail of an image. A blurring effect is created by using a low pass filter to remove detail occurring at high spatial frequencies. A Gaussian blur is a filter that uses a Gaussian kernel as its basis.

After the filter is applied, we consider rectangular groups of pixels surrounding each pixel in the image, in turn. The kernel is a separate group of pixels (a separate matrix / small image) that moves along with the pixel being worked on by the filter. The width and height of the kernel must be odd numbers so that the pixel being worked on is always in the middle. A weighted average of the color values of the pixels surrounding the pixel must be calculated before applying the kernel to it. In a Gaussian blur, the pixels closest to the kernel's center are given more weight than those further away. In a Gaussian blur, the pixels closest to the kernel's center are given more weight than those further away. This averaging is done channel by channel, with the average channel values becoming the filtered image's new pixel value. Larger kernels blur the image more than smaller kernels because they have more values on their average.

3.3 QR Code Detection

In this study, the QR code detection is done using the Pyzbar library. To successfully decode a QR code, it is necessary to determine the exact location of all four corner points that form the bounding rectangle.

Journal of Theoretical and Applied Information Technology
15th November 2022. Vol.100. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Pyzbar detects and at the same time decodes the QR code on the given image. Pyzbar works using Zbar which is a software used to read barcodes. It can be concluded that Pyzbar is an integration of Zbar in the python programming language. Through Zbar, information about the QR code contained in the input image can be extracted, such as the size, data, and content of the QR code content. The QR code information will then be added with the location information for the QR code rectangle coordinates and the coordinates of the QR code's four corners.

Figure 8: Detected QR Code

3.4 QR Code Image Rotation Correction

The detection process conducted by Pyzbar is prone to errors. The first detected QR code vertices are treated as the first quadrangle vertices, which will be placed on the top-left vertices by the perspective transformation algorithm. This can cause an error, where Pyzbar may detect the QR code corner located at the bottom-left first and treat it as the top-left corner of the QR code, resulting false rectification image of the QR code. This error can occur due to the skew of the QR code image in the dataset which is not geometrically correct.

The proposed QR code image rotation correction method is performed by sorting the stored value of location coordinates of the four corners of the quadrangle from the detected QR code images.

Figure 9 shows the correct order of the coordinates of the four quadrangle vertices of a QR code that should be generated from the detection process. the detection process carried out by Pyzbar is prone to errors in detecting the order of the four corners of the QR code. Pyzbar often detects P2 as P1, P3 as P2, P3 as P4, and P1 as P4. This will cause an error in the rectification process, where the resulting QR code image will be rotationally incorrect. Therefore, a method of correcting the order of the points needs to be done.

Figure 9: Visualization Example of Detected Four Quadrangle Points

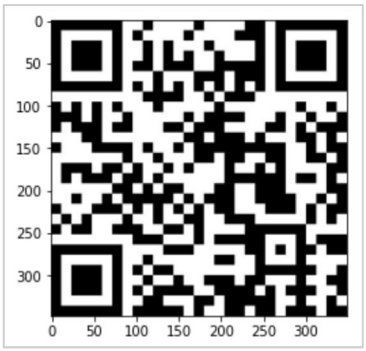

Figure 10: Example of QR code Plotting

Using the rules for the order of the corner points shown in Figure 9, it can be concluded that in the QR code presented in Figure 10, the point located in the top-left position is P1, the point located in the top-right position is P2, the point located in the bottom-left position is P3, and the point located in the bottom-right position is P4. The QR code image plotting as in Figure 10 shows that each corner point has its characteristics on the value of the x and y coordinates. P1 has the smallest sum of x and y values. P3 has the biggest sum of x and y values. P2 has a smaller x value than P4 (has a slight difference from the x value of P1), while P4 has a smaller y value than P2 (has a slight difference from the y value of P1). Using the characteristics or criteria possessed by each of these corner points, the method of sorting the order of the QR code quadrangles vertices can be done by comparing the value of the x and y coordinate owned by each corner point.

3.5 QR Code Rectification

After the four points have been sorted successfully into the correct order, the rectification process can then be conducted. The rectification process is conducted using the perspective transformation method. The perspective

transformation method is used to remove perspective effects and generate a frontal view image. When an image has geometric distortion, the QR code has a gradient value by an unsuitable angle, then the image is corrected to a square which we call rectified image. The effect of rectification based on the perspective transformation method is shown in Figure 11. Remarkably, the rectified QR code image doesn't have any distortions, which that concludes the rectification has been successful.

Figure 11: Rectified QR code

4. EXPERIMENTS

4.1 Dataset

As an initial study of a counterfeit QR code detection study, in this paper, a total of 1080 photos of printed counterfeit QR codes and a total of 270 photos of printed genuine QR codes are given. Thus, the total dataset is 1350 QR code photos. The dataset consists of a total of 30 QR code patterns that were taken with various camera angles and illuminations. The images of the QR code are also taken in a correct rotation, hence the images do not need to be rotated to get the correct pattern of the QR Code. All these photos were taken using a smartphone camera.

Each QR code pattern is taken photo multiple times with several variations in terms of illumination and camera angle. The illumination variations are images with sufficient light, images with normal light, and images with more light. The camera angle variations are 0 angles, +20 angles, and -20 angles. Each illumination variation is combined with each angle variation, resulting in a total of 9 variations of QR code images, each consisting of images of 30 QR Code patterns. Hence, a total of 270 images of QR code is obtained.

The counterfeit QR code dataset is created by printing all the QR codes using 4 different printers. The photo of each QR code that has been printed is then taken. This method of producing counterfeit dataset results in creating four categories of the counterfeit QR code dataset. Each category consists of 9 variations that have been described above. Hence, a total of 1080 images of counterfeit QR code is obtained.

Since this study does not aim to discern genuine and counterfeit QR codes, therefore these datasets are treated the same (does not differentiate between genuine and counterfeit QR codes). These datasets are considered regular image photos that contain a QR code in each of them.

Other than the 1350 photos of the QR codes dataset, we also have the ground truth of the QR code images which are the thirty images of the QR code pattern in the dataset. These QR code images are the original images of the generated QR codes and are not photos of the printed QR codes. These ground-truth images are used to compare the alignment between the patterns in the rectified QR code and the original generated QR code image.

(a) (b)

Figure 12: Photo of QR Code in the Dataset (a), Ground-Truth QR code Image (b)

4.2 Experimental Design

To achieve a high QR code detection success rate, proper image-pre-processing should be done. Hence, in this study, several pre-processing experiments are done to determine which pre-processing method is the best to be given to the dataset used.

Pre-processing methods that are performed in the experiment are: converting images into grayscale images, converting images into RGB images, image binarization using threshold value = 128, gaussian blur combined with Otsu threshold, and Otsu threshold only. The combination of all pre-processing methods mentioned above is also performed in the experiment. The detection success percentages of the five pre-processing methods above are calculated and the combination of the best three pre-processing methods is then selected as an additional pre-processing method.

In the combined methods, the entire dataset photos are detected sequentially. The entire dataset photos are pre-processed using the first method in the combined methods. Afterward, Pyzbar runs the QR code detection process to detect the QR codes on the pre-processed photos. If there are still exist QR codes that are failed to be detected, those photos are then pre-processed using the second method of the combined method and given back to Pyzbar to detect the QR codes once again. This process will continue until all QR codes in the dataset photos have been detected or all pre-processing methods in the combination have been carried out.

After the QR code detection process is done, each detected QR code is then subjected to the rectification process, which starts with the four-point QR code corner correction method. All rectified images of QR codes are stored and evaluated using the proposed alignment accuracy measurement method.

Through the detection process by Pyzbar, the QR code decoding process is also automatically carried out, so that the data or links embedded in each QR code can be saved. That way, we also perform detection on the ground-truth QR code dataset and store the data or links obtained from the detection results. These stored links are the key to finding the correct ground-truth QR code pair for each rectified QR code being evaluated. If the links resulting from the detection of a rectified QR code and a ground-truth QR code are the same, it can be ascertained that the pattern of the QR code is the same. Thus, the ground-truth QR code was chosen to be compared with the corresponding rectified QR code.

Each stored rectified QR code image and the correct ground-truth QR code pair will be converted into a binary image, resized to 348 × 348 pixels, and overlaid with an opacity of 50% each. As a result, overlaid images will only consist of three-pixel intensities, namely black (0), white (255), and gray (128).

The calculation process is carried out by counting the number of pixels that have the same intensity value between the rectified image and the ground-truth image from the QR code pattern. If both images have the same intensity at the same pixel coordinate, then the corresponding pixel intensity value at the overlaid image will be 0 (black) or 255 (white). The difference in both images' pixel intensity on the same coordinates will result in a pixel intensity value of 128 (grey) in the overlaid images.

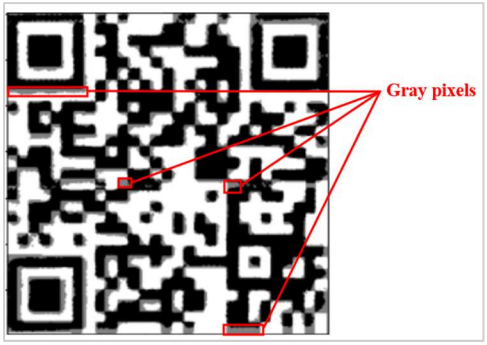

Figure 13: Overlaid QR code image

$$\text{AlignmentAccuracy} = \frac{n_{true}}{n_{true} + n_{false}} \times 100\% \quad (1)$$

The alignment accuracy calculation process is carried out using formula (1), where n_{true} is the number of black and white pixels of the overlaid image, while n_{false} is the number of gray pixels.

In the experiments, this calculation process is carried out with two treatments or methods, namely an alignment accuracy calculation without misalignment tolerance and with a misalignment tolerance of one pixel.

In the first method (without misalignment tolerance), the calculation process is carried out according to the formula, without any modification, but in the second method (with misalignment tolerance of one pixel), the alignment of the rectified QR code will be considered true if there is at least one pixel of the ground-truth QR code image that has the same pixel intensity value around the pixel of the rectified QR code being evaluated and the difference in pixel coordinates is x-1, x+1, y-1 or y+1. In short, the second alignment accuracy calculation is done using a 3x3 pixel kernel.

The alignment accuracy calculation process is carried out by iterating all stored rectified QR codes and calculating the alignment accuracy of the QR codes one by one. Thus, the calculation process produced *n* percentages of alignment accuracy values, where *n* is the number of QR codes that have been successfully detected and rectified. The average of these percentage values is the result reported in this study and is used as the alignment accuracy value of each pre-processing method.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.3 Experiment Result on Detection Accuracy

Through the carried-out experiments, it has been found that of the five pre-processing methods, the first (grayscale image), the third (binary image with threshold value = 128), and the second (RGB image) pre-processing methods are the best three pre-processing methods with detection success percentage values of 81%, 76%, and 75%, respectively. Therefore, these methods are selected as pre-processing methods to be combined into a new pre-processing method containing the best three pre-processing methods as an additional pre-processing method.

Table 1: Pre-processing Method's Detection Success Percentage

Pre-processing	Detection Success Percentage
Grayscale Image	81%
RGB Image	75%
Binary Image (Threshold = 128)	76%
Gaussian Blur + Otsu Threshold	60%
Otsu Threshold	69%
Combined Method	**95%**
Grayscale Image + RGB Image + Binary Image (Threshold = 128)	94%

After the entire set of pre-processing methods are determined and run until the Pyzbar detection process is complete, the results of the detection performance for each pre-processing method can be obtained and analyzed.

The experimental results on the performance of Pyzbar detection on pre-processed data showed quite good results in some methods and very good results in the other methods. The best result obtained from the experiment is the Pyzbar detection process of the dataset which is pre-processed by the combination of the five methods. The percentage of successful detection using the pre-processing method is 95% with an average success rate of 79%.

4.4 Experiment Result on Rectification

All rectified images of QR codes that have been stored are given to the two proposed alignment accuracy evaluation methods. The calculation process on the two alignment accuracy methods shows very good accuracy results.

Table 2: Alignment Accuracy Results: Without Misalignment Tolerance

Pre-processing	Alignment Accuracy Result (Avg)
Combined Method	84,946%
Grayscale Image	84,909%
RGB Image	85,004%
Binary Image (Threshold = 128)	85,271%
Gaussian Blur + Otsu Threshold	85,656%
Otsu Threshold	**85,661%**
Grayscale Image + RGB Image + Binary Image (Threshold = 128)	84,924%

Table 3: Alignment Accuracy Results: With Misalignment Tolerance of One Pixel

Pre-processing	Alignment Accuracy Result (Avg)
Combined Method	91,483%
Grayscale Image	91,453%
RGB Image	91,547%
Binary Image (Threshold = 128)	91,483%
Gaussian Blur + Otsu Threshold	92,024%
Otsu Threshold	**92,144%**
Grayscale Image + RGB Image + Binary Image (Threshold = 128)	91,464%

The experimental results show that rectified QR code images have good alignment accuracy with the best value being 85,661% with an average value of 85,196%. The results of the alignment accuracy also increased after being evaluated with a misalignment tolerance of one pixel. The best value of the alignment accuracy with misalignment tolerance of one pixel is 92,144% with an average value of 91,657%.

4.5 Experiment Analysis

All results in the detection and rectification process have been successfully obtained. To conclude which pre-processing method is the best, a multiplication calculation between the detection success percentage and the average value of each pre-processing method's alignment accuracy percentages has been carried out.

Through the multiplication of these data, it can be concluded that the combined method of the five

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

selected pre-processing methods succeeded in achieving the highest results in both methods of calculating alignment accuracy.

In the calculation of alignment accuracy without misalignment tolerance, the best pre-processing method achieves an accuracy value of 80,699% with an average value of 64.61%. In the calculation of alignment accuracy with misalignment tolerance of one pixel, the pre-processing method achieves an accuracy value of 86.908% with an average value of 71.99%.

Table 4: Results: Without Misalignment Tolerance

Pre-processing	Alignment Accuracy Result (Avg)	Detection Success Percentage	Accuracy Result (Avg) * Detection Success %
Combined Method	**84,946%**	**95%**	**80,699%**
Grayscale Image	84,909%	81%	68,777%
RGB Image	85,004%	75%	63,753%
Binary Image (Threshold = 128)	85,271%	76%	64,806%
Gaussian Blur + Otsu Threshold	85,656%	60%	51,393%
Otsu Threshold	85,661%	69%	59,106%
Grayscale Image + RGB Image + Binary Image (Threshold = 128)	84,924%	94%	79,829%

Table 5: Results: With Misalignment Tolerance of One Pixel

Pre-processing	Alignment Accuracy Result (Avg)	Detection Success Percentage	Accuracy Result (Avg) * Detection Success %
Combined Method	**91,483%**	**95%**	**86,908%**
Grayscale Image	91,453%	81%	74,077%
RGB Image	91,547%	75%	68,660%
Binary Image (Threshold = 128)	91,483%	76%	69,527%
Gaussian Blur + Otsu Threshold	92,024%	60%	55,215%
Otsu Threshold	92,144%	69%	63,580%
Grayscale Image + RGB Image + Binary Image (Threshold = 128)	91,464%	94%	85,976%

5. CONCLUSION AND FUTURE WORKS

The experiments were carried out using a dataset containing 1350 printed QR code photos. The photos in this dataset are pre-processed using the proposed method. The selected pre-processing methods have succeeded in improving the performance of Pyzbar QR code detection. Several pre-processing methods have shown a good detection success percentage and the other several methods have shown a very good percentage. The best pre-processing method was achieved by the combined of the five pre-processing methods. The pre-processing method obtained a detection success percentage of 95% with an average value of 79%. All detected QR codes have been successfully rectified using perspective transformation and evaluated using the proposed method of alignment accuracy calculation. The experimental results show that the rectified QR code images that are pre-processed using the combined five pre-processing methods have good alignment accuracy with a value of 85,661% with an average value of 85,196%. The alignment accuracy also increased after being evaluated with a misalignment tolerance of one pixel and managed to achieve an alignment accuracy of 92.144% with an average value of 91.657%.

Thus, this study has made several contributions to the study of QR code, namely it proposes the preparation of various pre-processing methods for a QR code dataset, provides performance results of several methods of pre-processing QR code images in supporting the performance of the detection process, and produces a method to measure the alignment accuracy of the rectified QR code's pattern alignment. The study also succeeds in achieving its main purpose to produce rectified images of the QR codes, which is proved by the good alignment accuracy produced in the experiment.

The results obtained in the detection process are strongly influenced by the quality of the image provided. One of the things that cause the failure of QR code detection is the poor quality of the photo from the QR code, which makes crucial information or features regarding the pattern of the QR code cannot be recognized and extracted.

The open issue left unattended in this problem is the design of a pre-processing method that is capable to be applied to QR code images with a variety of more extreme conditions in illuminations, viewing angles, and various other conditions that damage the pattern in the QR code. This issue is left unattended due to the limited variations of image conditions available in the dataset. Extreme damage to the quality of QR code images is a major challenge and threat in conducting the validity of the QR code image pre-processing process and requires further study of the pre-processing method of QR code images.

In future works, this task also can be improved by designing a pre-processing method that can remove pixels that damage the QR code pattern in the dataset and improve the quality of the dataset by providing better contrast to the black and white square pixels in the QR code. The alignment accuracy can be improved by removing untidy pixels and noises so that the resulting QR code pattern will have a similar pattern to the scanned format of the QR code generated by a computer.

REFERENCES:

[1] Mehta A, Solanki DK, "Design and Development of QR Code Recognition from Digital Image", *International Journal of Engineering Research & Technology (IJERT)*, Vol. 9, No. 5, 2021, pp. 183–186.

[2] Liu Y, Liu M, "Automatic Recognition Algorithm of Quick Response Code Based on Embedded System", In: *Sixth International Conference on Intelligent Systems Design and Applications*, IEEE, 2006.

[3] Nicolas C, "Quick Response (QR) Code Image Pattern Recognition Time Utilization Tracking Application". *Southeast Asian Journal of Science and Technology*, Vol. 4, No. 1, 2019, pp. 42–49.

[4] L. Hudson, "Pyzbar: Read one-dimensional barcodes and QR codes from Python 2 and 3" [Internet], PyPI, [cited 2022 Feb 2], Available from: https://pypi.org/project/Pyzbar.

[5] Zhang X, Luo H, Peng J, Fan J, Chen L, "Fast QR Code Detection", *In: 2017 International Conference on the Frontiers and Advances in Data Science (FADS)*, IEEE, 2017, pp. 151-154.

[6] Tribak H, Zaz Y, "QR Code Recognition based on Principal Components Analysis Method", *International Journal of Advanced Computer Science and Applications*, Vol. 8, No. 4, pp. 241–248.

[7] Jain V, Jain Y, Dhingra H, Saini D, Taplamacioglu MC, Saka M, "A Systematic Literature Review on QR Code Detection and Pre-Processing", *International Journal on "Technical and Physical Problems of Engineering" (IJTPE)*, Vol. 13, No. 46, 2021, pp. 111–119.

[8] Chen R, Yu Y, Xu X, Wang L, Zhao H, Tan H-Z, "Adaptive Binarization of QR code Images for Fast Automatic Sorting in Warehouse Systems", *Sensors*, Vol. 19, No. 24, 2019.

[9] Li M, Cao P, Feng L, Yu L, Chen J, Wang J, "The research of QR Code Image Correction Based on Image Gray Feature", In: *2017 First International Conference on Electronics Instrumentation & Information Systems (EIIS)*, IEEE, 2017, pp. 1-5.

[10] Jin J, Wang K, Wang W, (2021). "Research on Correction and Recognition of QR Code on Cylinder", In: *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, IEEE, 2021, pp. 1485–1489.

[11] Xuan W, Peng C, Liuping F, Jianle Z, Peijun H, "Research on Correcting Algorithm of QR Code Image's Distortion", In: *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, IEEE, 2017, pp. 1752–1756.

[12] Supriyono H, Fitriyan MR, Muamaroh, "Developing A QR Code-Based Library Management System with Case Study of Private School In Surakarta City Indonesia", In: *2018 Third International Conference on Informatics and Computing (ICIC)*, IEEE, 2018, pp. 1–6.

[13] Tribak H, Zaz Y, "QR Code Patterns Localization based on Hu Invariant Moments", *International Journal of Advanced Computer Science and Applications*, Vol. 8, Issue 9, 2017.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[14] Karrach L, Pivarčiová E, Božek P, "Identification of QR Code Perspective Distortion Based on Edge Directions and Edge Projections Analysis", *Journal of Imaging*, Vol. 6, No. 7, 2020.

[15] He Y, Yang Y, "An Improved Sauvola Approach on QR CodeImage Binarization", In: *IEEE 11th International Conference on Advanced Infocomm Technology (ICAIT)*, IEEE, 2019, pp. 6–10.

[16] Yuan B, Li Y, Jiang F, Xu X, Zhao J, Zhang D, et al, "Fast QR code detection based on BING and AdaBoost-SVM", In: *IEEE 20th International Conference on High Performance Switching and Routing (HPSR)*, IEEE, 2019, pp.1-6.

[17] Blanger L, Hirata NST, "An Evaluation of Deep Learning Techniques for QR Code Detection." In: *IEEE International Conference on Image Processing (ICIP)*, IEEE, 2019, pp/ 1625–1629.