A High Density Colored 2D-Barcode: CQR Code-9

Max E. Vizcarra Melgar and Mylène C. Q. Farias
Dept. of Electrical Engineering
University of Brasilia
Brasilia-DF, Brazil
Email: maxvizcarra@ieee.org and mylene@ieee.org

Flávio de Barros Vidal and Alexandre Zaghetto
Dept. of Computer Science
University of Brasilia
Brasilia-DF, Brazil
Email: fbvidal@unb.br and zaghetto@image.unb.br

***Abstract*—This paper presents an implementation of a colored 2D-barcode which is based on the structure of the CQR Code (Colored Quick Response Code). While the first version of the CQR Code (CQR Code-5) is a 2-D barcode of 5 colors, this new version (CQR Code-9) has 9 colors. In this paper we describe the implementation details of this new CQR Code. This new version of the CQR Code can store up to 2,048 information bits, requiring 4,576 redundancy bits. The Reed-Solomon error correction algorithm provides an error correction rate of 34.54%. Experimental tests were performed by printing CQR Codes in a 1.3 cm $\times$ 1.3 cm area. Results show that the CQR Code-9 is suitable for cryptographic applications that require that a high number of bits be stored in a small printed area.**

***Keywords*-CQR Code; Colored 2D-Barcodes; Color Segmentation.**

I. Introduction

QR Codes are two-dimensional (2-D) monochromatic barcodes that are widely used as optical machine-readable data channel transmission mediums. They were proposed in 1994 by the Japanese company Denso Wave Incorporated and its last standardization was released in 2015 on the document ISO/IEC 18004:2015 [1]. In standard QR Codes, each module represents a single bit, with a black module storing a bit with value ’1’ and a white module storing a bit with value ’0’. The technological concepts used by the QR Codes are similar to what is used on linear or matrix barcodes [1], [2], but QR Codes have a superior data density capacity and a high reading speed. These properties have popularized QR Codes that today can be found in advertisements, business cards, t-shirts, product labels, information security, etc. [3], [4], [5]. When compared to other technologies, like for example radio frequency tags or chips, two-dimensional barcodes are a cheap solution for automated data transmission applications.

Recently, new 2D colored barcodes have been proposed. The use of colors has as main objective of increasing the number of bits that a single module can store [6]. An example of a 2-D colored barcode is the High Capacity Color Barcode (HCCB), which was created by Microsoft Corporation [6]. Another example of a 2-D color barcode is the High Capacity Colored 2-D Code (HCC2D) [7]. HCC2D is based on the QR Code standard, i.e. uses a similar structure, error-correction algorithm, and data masking process. A HCC2D Code can be created using 4, 8, or 16 colors. Fig. 1 (a) and (b) show examples of a 4-colors HCC2D and a 4-colors HCCB, respectively.

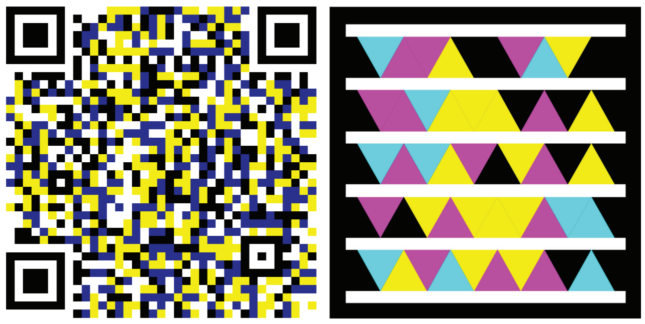
(a) HCC2D [6]. (b) HCCB [7].

Fig. 1. Example of 2D-Color Barcodes of 4 colors

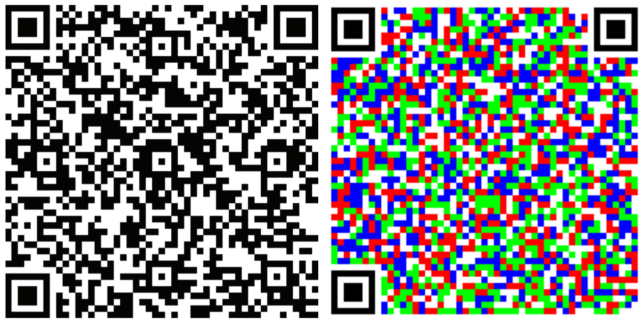
(a) QR Code [1]. (b) CQR Code-5 [8].

Fig. 2. Example Example of QR Code (black and white) and CQR Code-5 (white, black, red, green, and blue)

In a previous publication [8], it was proposed a Colored CQR Codes (CQR Code-5) which uses 5 colors (white, black, red, green, and blue) and can transmit up to 1,024 information bits, what is suitable for cryptosystems that require a large storage capacity. Fig. 2 (a) and (b) show examples of a QR Code and a CQR Code-5, respectively. In this paper, we propose a second CQR Code (CQR Code-9), which uses 9 colors (white, black, red, green, blue, magenta, cyan, yellow, and gray) and, as a consequence, can store twice as much information (in the same printed area) than the previous version, CQR Code-5. CQR Code-9 was designed to carry up to 2,024 information bits in a small printed area.

This paper is divided as follows. In Section II, we briefly introduce the algorithm that generates and decodes the CQR Code-5. In Section III, we describe the algorithm for generating and decoding a CQR Code-9. Experimental results of the print-scan channel are shown in Section IV. Finally, we present our conclusions in Section V.

II. CQR Code-5

The first version of the CQR Code [8] uses 5 different colors (white, black, red, green, and blue). From these 5 colors, 4 are used by the information modules to store data. Therefore, each information module is able to store 2 bits (00 - red, 01 - green, 10 - blue and 11 - white). Black and white modules are used to represent the Finder Patterns, CQR Code-5 does not use black modules in the area where bits are stored in order to make easier the detection of the Finder Patterns without confusing them with clusters of black and white data modules.

Two-dimensional barcodes are capable of storing a certain amount of Bits per Module (BpM) [9], which is determined by the number of colors used. This amount is given in Equation 1:

$$\text{BpM} = \log_2(N_c), \tag{1}$$

where N_c is the number of colors used for information modules.

CQR Code-5 uses modules with a fixed size of 49×49 to encode up to 1,024 information bits. The systematic Reed-Solomon algorithm is used to generate 3,392 redundancy bits [10], making the barcode very robust to error. More specifically, CQR Code-5 has an error correction error of up to 38.41% of error bits. Thus, its decoding process is robust enough to ensure retrieval of information transmitted in a noisy channel as illustrated in Fig. 3.

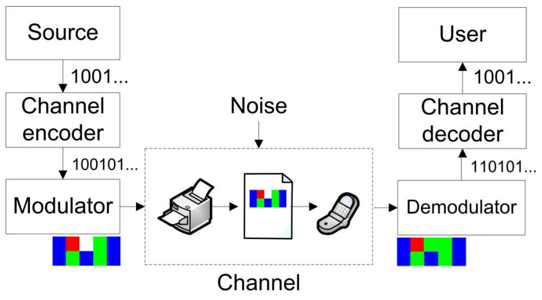

Fig. 3. Block diagram of a communication system that considers print-scan storage and transmission stages [8].

The CQR Code structure is shown in Fig. 4. The modules are distributed over two regions: the Function Patterns and the Encoding Region. Function Patterns have exactly the same structure for QR Codes and CQR Codes, i.e. they are composed by Quiet Zone, Finder Patterns, and Separators. The Quiet Zone is composed by the white modules that surround the barcode on all four sides (top, bottom, left, and right). Finder Patterns are the 3 identical symbols located at the upper-left, upper-right, and lower-left corners of the code, which are used to correct the image position at the decoder. Separators are the 1×8 or 8×1 white modules that separate Encoding Region and Finder Patterns [8]. For CQR Code-5, there are a total of $2,401$ (49×49) modules, in which 192 modules are from Finder Patterns and Separators and $2,209$ are from the Encoding Region. Since each module represents 2 bits, there are $4,418$ available bits, from which $1,024$ are information bits and $3,392$ are redundancy bits. One module is left unused.

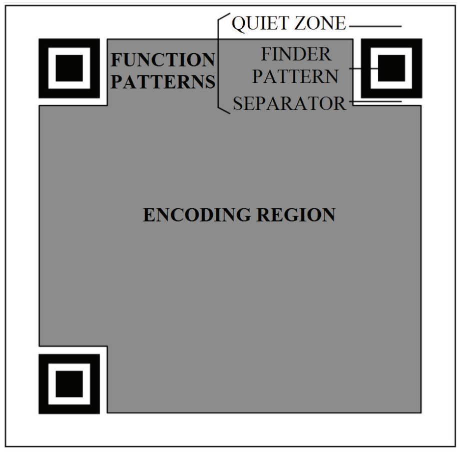

Fig. 4. The CQR Code structure [8].

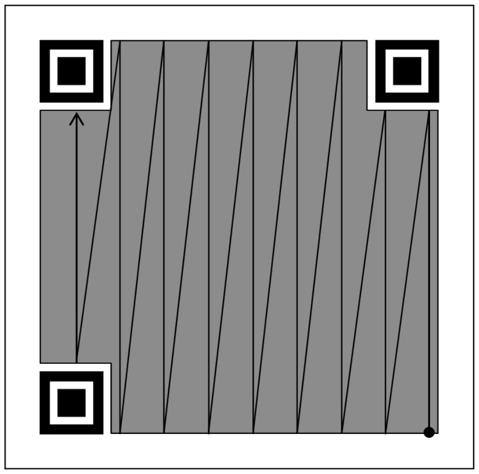

Fig. 5. The CQR Code positioning of the modules in the Encoding Region [8].

CQR Code-5 uses the Reed-Solomon error-correction algorithm to protect the transmitted information [10]. This error-correction algorithm is very popular, being frequently used in satellite communications and optical recording systems, like CD, DVD and Bluy-Ray [11]. The Reed-Solomon algorithm takes as input k symbols of s bits and adds $n-k$ redundancy symbols, forming a code-word of n symbols. For the CQR Code, the Reed-Somolom parameters [10] are $s = 16$ (amount of bits per symbol), $k = 64$ (amount of information symbols), and $n = 276$ (total amount of symbols). The resulting string of symbols $RS(n, k)$ is given by the following equation:

$$RS(276,64) = [D_1 \cdots D_{64}\ RS_1 \cdots RS_{212}], \tag{2}$$

where D_i $(i = 1 \cdots 64)$ represents the k information input symbols and RS_j $(j = 1 \cdots 212)$ represents the $n-k$ parity symbols. The primitive polynomial used in the generation of redundancy bits is the following:

$$p(D) = D^{16} + D^{12} + D^3 + 1. \tag{3}$$

A Reed-Solomon decoder can correct up to t symbols that contain errors in n symbols, t is determined on Equation 4.

$$t = \frac{n-k}{2}. \tag{4}$$

For the CQR Code-5, $t = 106$, this means that it is possible to completely recover the 64 information symbols, even if up to 38.41% of the n symbols are lost or degraded.

The stages for decoding of the CQR Code are illustrated in Fig. 6. The modules are placed in the Encoding Regions in a bottom-up order, i.e. from the most right to the most left column. Fig. 5 (b) shows the order in which Encoding Region is filled with the data modules. Other details of the decoding process, such as image segmentation and experimental results, can be found in previous publication [8]. The impact of compression algorithms on the decoding of the CQR Code-5 was showed in a previous publication [12], here it was checked the maximum possible JPEG, JPEG2000 and H.264/AVC compression bitrates that can be applied on the CQR Code-5.

III. CQR Code-9

CQR Code-9 and CQR Code-5 have the same structure and decoding process, as illustrated in Fig. 4, Fig. 5 and Fig. 6. Both have 49×49 modules and omnidirectional image capture rotation features that use as input the Finder Patterns centers, which are Right Finder Pattern (RFp), Down Finder Pattern (DFp) and Central Finder Pattern (CFp), and their mutual distances in pixels, this process is shown in Fig. 7. On the other hand, this new version of the CQR Code, has 9 colors, using 8 colors to encode the information. Therefore, the 8 colors used for storing information allows using an alphabet of 3 bits (000 - red, 001 - green, 010 - blue, 011 - cyan, 100 - magenta, 101 - yellow, 110 - white and 111 - gray). As in the previous version, Function Patterns use only 2 colors (white and black), requiring less computational cost. CQR Code-9 does not use the black color in the Encoding Region, reserving this color for the Finder Patterns [8].

Also, CQR Code-9 uses the same decoding process as CQR Code-5 (see Fig. 6). CQR Code-9 is illustrated in Fig. 8.

Since each module represents 3 bits, CQR Code-9 has $6,987$ bits available in the Encoding Region. CQR Code-9 allows transmitting up to $2,048$ information bits, requiring $4,576$ redundancy bits. It also uses the Reed-Solomon error correction algorithm [10], but in this case the Reed-Solomon parameters are $s = 16$, $k = 128$ and $n = 414$. The resulting code-word is:

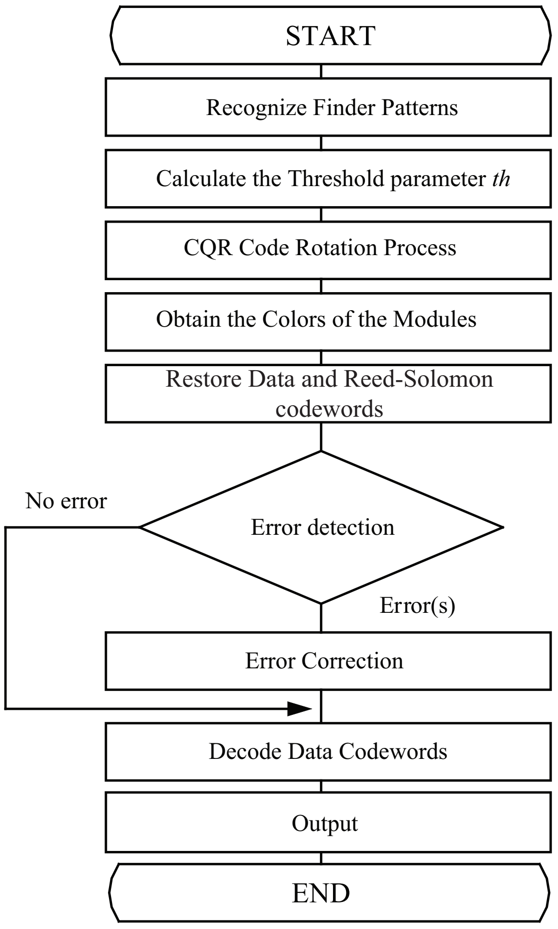
START
Recognize Finder Patterns
Calculate the Threshold parameter *th*
CQR Code Rotation Process
Obtain the Colors of the Modules
Restore Data and Reed-Solomon codewords
No error
Error detection
Error(s)
Error Correction
Decode Data Codewords
Output
END

Fig. 6. Fluxogram of CQR Codes decoding stage [8].

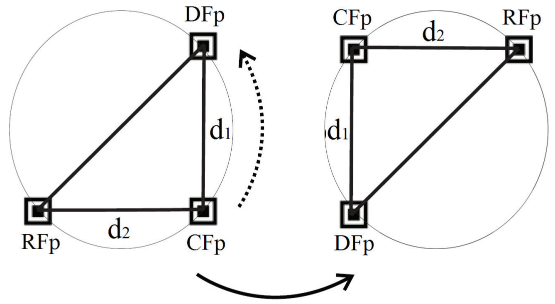

Fig. 7. CQR Code rotation scheme.

$$RS(414,128) = [D_1 \cdots D_{128}\ RS_1 \cdots RS_{286}], \tag{5}$$

where D_i $(i = 1...128)$ represents the 128 information symbols and RS_j $(j = 1...286)$ represents the 286 redundancy symbols. Therefore, $t = (414 - 128)/2 = 143$ symbols can be corrected, i.e. an error correction rate of 34.54%.

Since the CQR Code-9 has more colors than the CQR Code-5, its segmentation decision algorithm is different. The new segmentation algorithm aims to find the predominant

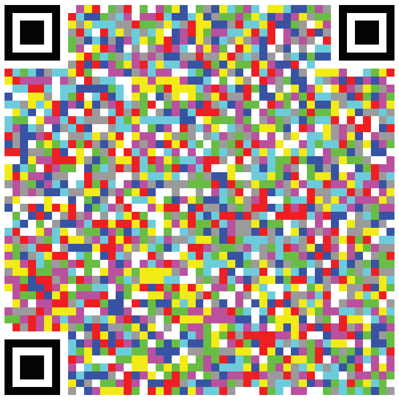

Fig. 8. Example of CQR Code-9 (white, black, red, green, blue, cyan, magenta, yellow and gray).

TABLE I
DENSITY OF TWO-DIMENSIONAL BARCODES, CONSIDERING IMAGES PRINTED AT 600 DPI. CQR CODE WAS PRINTED IN AN AREA OF 0.26195 SQUARE INCHES [7].

2D barcode	Data Density [KBytes per in^2]
QR Code	0.627
HCCB	2.000
HCC2D	1.881
CQR Code-5	2.057
CQR Code-9	3.086

color in each captured pixel. The algorithm takes as input the image C and its previous calculated threshold th [8], and estimates the maximum, middle and minimum RGB values in order to find the distance between the 9 possible RGB values (white, gray, red, green, blue, yellow, magenta, and cyan). The detailed algorithm is shown in Algorithm 1. In our tests, this segmentation algorithm provided better results than Otsu's segmentation algorithm [13] (applied on the R, G and B planes).

CQR Code-9 can store and transmit $6,624$ bits in the printed area of 1.3 cm $\times$ 1.3 cm, this gives a rate of 3.086 KBytes per square inch that can be achieved for the smallest module size. Table I summarizes the data density for the following two-dimensional barcodes: QR Code, HCCB, HCC2D, and CQR Codes.

Algorithm 1 CQR Code-9 Segmentation - (C, th)

```
for $i = 1$ to $C.height$ do
    for $j = 1$ to $C.width$ do
        $MaxC = max(C(i,j,:))$
        $MidC = mean(C(i,j,:))$
        $MinC = min(C(i,j,:))$
        $D = MaxC - MinC$
        if $D < th/2$ then
            if $MinC > 1.5 * th$ then
                $C(i,j,RGB) = \{255, 255, 255\}$          ▷ $white$
            else
                $C(i,j,RGB) = \{127, 127, 127\}$          ▷ $gray$
            end if
        else
            if $MaxC == C(i,j,R)$ then
                if $MaxC - MidC < th/2$ and $MidC == C(i,j,G)$ then
                    $C(i,j,RGB) = \{255, 255, 0\}$        ▷ $yellow$
                else if $MaxC - MidC < th/2$ and $MidC == C(i,j,B)$ then
                    $C(i,j,RGB) = \{255, 0, 255\}$        ▷ $magenta$
                else
                    $C(i,j,RGB) = \{255, 0, 0\}$          ▷ $red$
                end if
            else if $MaxC == C(i,j,G)$ then
                if $MaxC - MidC < th/2$ and $MidC == C(i,j,R)$ then
                    $C(i,j,RGB) = \{255, 255, 0\}$        ▷ $yellow$
                else if $MaxC - MidC < th/2$ and $MidC == C(i,j,B)$ then
                    $C(i,j,RGB) = \{0, 255, 255\}$        ▷ $cyan$
                else
                    $C(i,j,RGB) = \{0, 255, 0\}$          ▷ $green$
                end if
            else
                if $MaxC - MidC < th/2$ and $MidC == C(i,j,R)$ then
                    $C(i,j,RGB) = \{255, 0, 255\}$        ▷ $magenta$
                else if $MaxC - MidC < th/2$ and $MidC == C(i,j,G)$ then
                    $C(i,j,RGB) = \{0, 255, 255\}$        ▷ $cyan$
                else
                    $C(i,j,RGB) = \{0, 0, 255\}$          ▷ $blue$
                end if
            end if
        end if
    end for
end for
```

IV. RESULTS: CQR CODE-9

Our test set is composed by 170 snapshots of CQR Codes-9, acquired with the built-in camera of the Samsung Galaxy S5 smartphone under indirect daylight illuminance. All images were captured with the best possible resolution (16 Megapixels). The evaluation of performance was done considering the corrupted symbols versus the distance of the captured image. The printed versions of the codes have a size equal to 1.3 cm $\times$ 1.3 cm. A printer Ricoh MP C2051 was used to print these images on a regular office A4 paper with $75 g/m^2$ density. Fig. 9 and Fig. 10 show examples of captured CQR Codes-9 at different distances, this distance variation influences on the decoding result. The proposed CQR Code-9 was not successfully decoded at any distance using the camera of the Samsung Galaxy S4 Mini, which is a camera of inferior resolution.

The average decoding results that were obtained for 170 snapshots (ten images for each different distance) are shown in Fig. 11, for which the image capture distances were varied in intervals of 1 cm. The brown line in Figure 11 represents the error correction threshold (error correction rate is equal to 34.54%), i.e. images captured with errors above this line are not completely decoded. The data stored in the CQR Code-9 can be perfectly decoded for capture distances between 7 cm and 13 cm, corresponding captures with an error percentage below 34.54%.

Table II shows an analysis of the results of the decoding process. In this table, the first column shows the capture

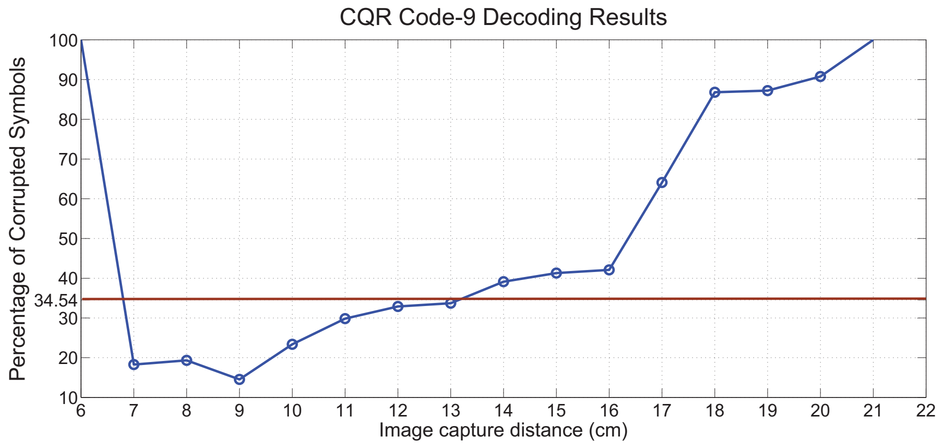

Fig. 11. CQR Code-9 decoding results due capture distance on Samsung Galaxy S5.

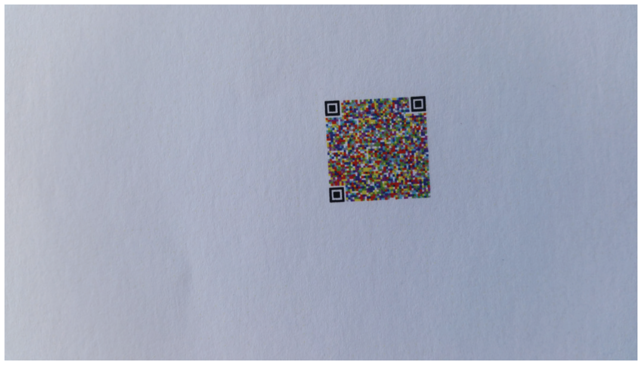

Fig. 9. CQR Code-9 captured and correctly decoded at a distance of 7 cm.

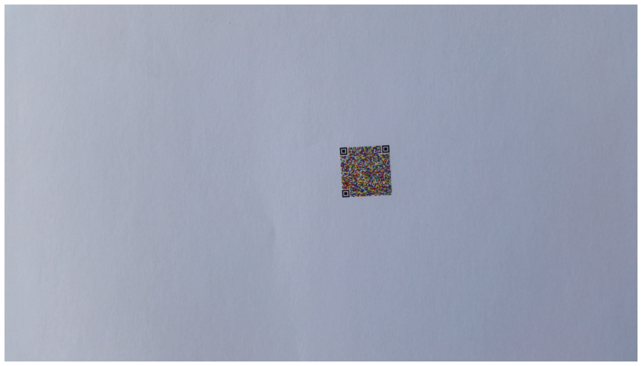

Fig. 10. CQR Code-9 captured and incorrectly decoded at a distance of 14 cm.

distance and the second column shows the corresponding number of correctly decoded snapshots. The third column shows the average error percentage of distorted symbol (for a total of 414 symbols). Finally, the last 2 columns show the minimum and maximum amount of corrupted symbols for the 10 snapshots in each line.

TABLE II
RESULTS: CAMERA DISTANCE, AMOUNT OF CORRECTLY DECODED CQR CODES-9, PERCENTAGE OF AVERAGE SYMBOL ERROR AND MINIMUM AND MAXIMUM AMOUNT OF CORRUPTED SYMBOLS.

Distance of CQR Code-9 (cm)	Correctly Decoded CQR Code-9	Average Symbol Error (%)	Min. amount of Corrupted Symbols	Max. amount of Corrupted Symbols
6	0	100.00	414	414
7	10	18.28	69	81
8	10	19.34	76	102
9	10	14.54	54	65
10	10	23.38	88	101
11	10	29.83	115	130
12	10	32.89	130	140
13	8	33.71	130	147
14	0	39.13	158	169
15	0	41.30	165	178
16	0	42.10	169	178
17	0	61.10	249	285
18	0	86.81	330	388
19	0	87.22	338	378
20	0	90.77	358	401
21	0	100.00	414	414
22	0	100.00	414	414

V. CONCLUSIONS

CQR Codes are suitable for storing and transmitting symmetric and asymmetric cryptography data. When compared to other color barcodes, CQR Codes have a higher data density per area. The currently available CQR Code-5 allows storing and transmitting up to $1,024$ bits. This paper describes the CQR Code-9, a new CQR Code that allows using 9 colors to store information. CQR Code-9 allows storing and transmitting up to $2,048$ bits of information.

Results show that, on average, a smaller number of corrupted symbols is obtained when images are captured at a 9 cm distance. For very small (< 7 cm) and very big (> 13 cm) distances, the snapshots could not be decoded. The quality of the acquisition camera was an important factor. The

Samsung Galaxy S5 camera provides the necessary quality to successfully decode the CQR Code-9 images, for distances between 7 cm and 13 cm. Better results can be obtained using cameras with superior resolution and quality.

Acknowledgment

This work was supported in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and in part by the University of Brasilia.

References

[1] "ISO/IEC 18004:2015 - Information technology - Automatic identification and data capture techniques - QR Code bar code symbology specification," February 2015.

[2] I.C. Dita, M. Otesteanu and Q. Franz., "Data Matrix Code - A reliable optical identification of microelectronic components."

[3] Keng T Tan, Douglas Chai, Hiroko Kato and Siong Khai Ong, "Designing a Color Barcode for Mobile Applications," in *IEEE Pervasive Computing*, June 2012, pp. 50–55.

[4] Hiroko Kato and Keng T Tan, "Pervasive 2D barcodes for camera phone applications," in *IEEE Pervasive Computing*, vol. 6, Oct-Dec 2007, pp. 76–85.

[5] Max E. Vizcarra Melgar and Santander, L.A., "An alternative proposal of tracking products using digital signatures and QR codes," in *Proc. of the 2014 IEEE Colombian Conference on Communications and Computing*, June 2014.

[6] H. Bagherinia and R. Manduchi, "A Theory of Color Barcodes," in *Proceedings of the IEEE Color and Photometry in Computer Vision Workshop*, 2011.

[7] A. Grillo, A. Lentini, M. Querini and G. Italiano, "High Capacity Colored Two Dimensional Codes," in *Proceedings of the International Multiconference on Computer Science and Information Technology*, 2010, pp. 709–716.

[8] Max E. Vizcarra Melgar; Zaghetto A.; Macchiavello B. and Nascimento A. C. A, "CQR Codes: Colored Quick-Response Codes," in *Proceedings of the2ND IEEE International Conference on Consumer Electronics - Berlin (IEEE 2012 ICCE-Berlin)*, September 2012.

[9] Antonio Grillo; Alessandro Lentini; Marco Querini and Giuseppe F. Italiano, "High Capacity Colored Two Dimensional Codes," in *In: Proceedings of International Multiconference on Computer Science and Information Technologyr*, vol. 5, October 2010, pp. 709–716.

[10] Berlekamp E., "Nonbinary BCH decoding," in *Proc. of the Int. Symp. Inf. Theory (ISIT)*, vol. 14 n.2, 1967.

[11] J. Lee and K. A. S. Immink, "An efficient decoding strategy of 2D-ECC for optical recording systems," in *IEEE Trans. Consumer Electronics*, vol. 55, 2009, pp. 1360–1363.

[12] Max E. Vizcarra Melgar; Farias C. Q. M.; Zaghetto A., "An evaluation of the effect of JPEG, JPEG2000 and H.264/AVC on CQR Codes decoding process," in *Proceedings of SPIE-IS&T Electronic Imaging*, vol. 9404, February 2015.

[13] N. Otsu, "A threshold selection method from gray-level histograms," in *In: Proceedings of IEEE Trans. Sys., Man., Cyber*, vol. 9, January 1979, pp. 225–236.