Colored-QRNet: Fast QR Code Color Image Embedding

Karelia Pena-Pena
Electrical and Computer Engineering
University of Delaware
Newark, DE, USA
kareliap@udel.edu

Daniel L. Lau
Electrical and Computer Engineering
University of Kentucky
Lexington, KY 40506, USA
dllau@uky.edu

Gonzalo R. Arce
Electrical and Computer Engineering
University of Delaware
Newark, DE, USA
arce@udel.edu

***Abstract*—Quick Response (QR) codes are widely used to connect offline and online content, and thus many efforts have aimed at improving the visual quality of QR codes to be easily included in publicity designs in billboards and magazines. The most successful approaches, however, are slow since optimization algorithms are required for the generation of each beautified QR code, hindering its online customization. The aim of this paper is the fast generation of visually pleasant and robust QR codes. The proposed framework leverages state-of-the-art deep-learning algorithms to embed a color image into a baseline QR code in seconds while keeping a maximum probability of error during the decoding procedure. Halftoning techniques that exploit the human visual system (HVS) are used to smooth the embedding of the QR code structure in the final QR code image while reinforcing the decoding robustness. Compared to optimization-based methods, our framework provides similar qualitative results but is significantly faster.**

***Index Terms*—QR codes, image embedding, deep learning.**

I. Introduction

Quick Response (QR) codes are extensively used in e-commerce and publicity campaigns, engaging new customers by offering discounts, and measuring the impact of the campaign. QR codes are decodable by any smartphone, and thus, constitute a powerful tool for mobile commerce applications, providing a link between offline and online marketing. The black and white structure of standard QR codes, however, hinders their insertion into artistic designs. Furthermore, traditional QR codes do not provide any visual information about the encoded message. Currently, due to the pandemic crisis, the use of QR codes is on the rise since businesses are looking for ways to offer customers a touch-free experience [1]. For all these reasons, a fast method for the generation of visually enhanced QR codes has emerged as an attractive tool for both big and small businesses.

Although some of the previous methods successfully embed a color image into a QR code while keeping a low probability of error, these methods are either slow due to their reliance on optimization procedures, or fast but without decoding robustness guarantees. In [2], for instance, the embedding of QR code images is formulated as an optimization problem that minimizes the visual distortion of the color image subject to a constraint in the probability of error. Even though QR code images generated by this method are visually pleasant and robust during decoding, its computational complexity prohibits its commercial applications. In this work, this problem is addressed by leveraging the power of deep learning, designing, and training a convolutional neural network (CNN) that embeds the color image into a black-and-white QR code. As shown in Fig. 1, the proposed framework consists of two stages. In the first stage, a binary baseline QR code $\mathbf{Q}_b$ that is more similar to the embedding image is generated, taking advantage of the method proposed by Cox *et al*. [3]. Different from other approaches, the local binarization process performed by popular decoders is taken into consideration, further improving on the decoding robustness as it is shown in our experiments. The second stage corresponding to the image embedding, traditionally based on optimization algorithms, is substituted by a CNN architecture, dubbed as colored-QRNet, as depicted in Fig. 1. This architecture is characterized by a siamese design i.e. composed of two identical branches. To the best of our knowledge, it is the first time that a CNN architecture is used for end-to-end embedding of an image into a QR code, significantly speeding up the process. The contribution of this manuscript is thus three-fold. First, a novel

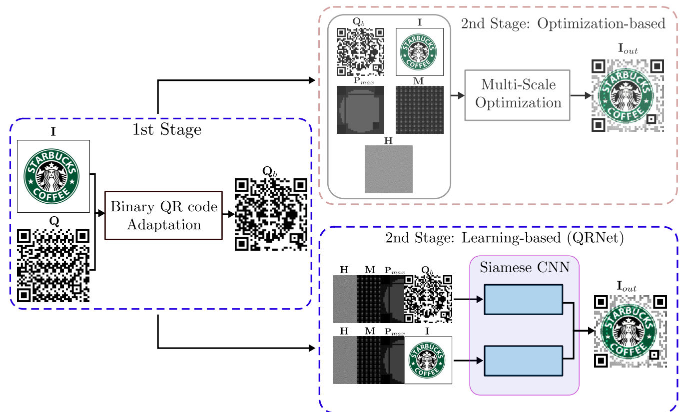

Fig. 1. Two-stage framework for fast QR code image generation (blocks in blue-dashed lines). Stage 1 aims at generating the binary adapted QR code $\mathbf{Q}_b$. Stage 2 embeds the color image $\mathbf{I}$ into $\mathbf{Q}_b$. Traditionally, Stage 2 is performed by optimization-based methods (top red-dashed block), which are slow. Instead, we use the proposed colored-QRNet (bottom blue-dashed block), significantly speeding up the process.

CNN architecture is proposed and successfully used to perform the embedding of an image into a QR code. Second, the selection of the QR code modules to be modified considers not only the contrast but also the saliency map and the edges; simultaneously reducing the visual contrast and giving priority to the region of interest in the image. Third, each element within the framework is carefully designed to consider the local binarization process performed by the decoder, generating codes that are more robust when compared to the prior art.

The remainder of this paper is organized as follows. Section II presents the multi-scale optimization-based QR code embedding procedure, adapted from [2], and used for the generation of the training data; together with the description of the proposed CNN architecture corresponding to the second stage of the framework. Section III introduces the first stage of the proposed framework, devoted to generating a binary adapted QR code. Section IV presents the evaluation and comparison of the results. Section V introduce the conclusions.

II. QR Code Image Embedding

In this section, we first introduce the optimization-based method for QR code image embedding used for the generation of a baseline dataset, followed by the description of the proposed network architecture for the fast learning-based and optimization-free QR code image embedding.

A. Multi-Scale Optimized QR Code Image Embedding

For the generation of the dataset, we used a modified version of the method proposed by Garateguy *et al.* [2] in which the luminance of the input color image $\mathbf{I}$ is modified to embed the QR code information, taking advantage of the decoding procedure of standard QR codes to achieve a trade-off between the visual quality and the decoding robustness of the QR code. This section focuses on the proposed modifications to the embedding algorithm, a mode detailed explanation can be found in [2].

We consider the fact that during decoding only the central pixel of each module within the QR code is sampled, thus central pixels in each module are selected for luminance modification by the mask $\mathbf{M}$, and their luminance parameters $\boldsymbol{\alpha_c}, \boldsymbol{\beta_c}$ are then optimized to encode the QR code information. In the same way, a halftone [4] mask $\mathbf{H}$ with concentration parameter $\mathbf{p_c}$ is used to select non-central pixels for modification with luminance parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$. Non-central pixels could be sampled due to errors during the sampling grid estimation. Thus, the use of the halftone mask to select non-central pixels improves the decoding robustness as well as reduces the coarse square structure of the QR code. Pixels that are not selected for modification preserve the luminance of the input image $\mathbf{I}$ [2]. As shown in Fig. 1(red-dashed), the inputs to the multi-scale optimization are the original color image $\mathbf{I}$, the QR code $\mathbf{Q}_b$, the mask for the maximum probability of error $\mathbf{P}_{max}$, and the masks $\mathbf{M}$ and $\mathbf{H}$ for the selection of central and non-central pixels, respectively. As in [2], the concentration of modified pixels $\mathbf{p_c}$ and the luminance parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\alpha_c}, \boldsymbol{\beta_c}$ are locally optimized to achieve a trade-off between visual quality and decoding robustness. To estimate the probability of decoding error, we use a normalized version of the probability of error model proposed in [2] in which two independent probability models are developed and combined based on the sampling accuracy of the central pixels of the QR code modules. In this paper, for the local optimization, we propose the use of a module-based overlapping window, motivated by the thresholding method utilized by the popular ZXing library [5]. In this way, the result is more robust against the difference between the size of the image at the generation stage and the size of the image at the scanning stage of the QR code. To this end, in the embedding optimization of a QR code with $q \times q$ modules, the $(m, n)-$th module of size $w \times w$ is optimized independently using an overlapping window of 5×5 modules centered at the $(m, n)-$th module in the QR code. A total of q^2 independent local optimizations are then required. Finally, the formulation of the local optimization problem for each overlapping window consists of minimizing a visual quality metric subject to a constraint over the corresponding probability of error as

$$\begin{aligned} \underset{\alpha,\beta,\alpha_c,\beta_c,p_c}{\arg\min} \quad & J_{m,n}(\alpha, \beta, \alpha_c, \beta_c, p_c) \\ \text{subject to} \quad & P_{err_{m,n}} < P_{max_{m,n}}, \end{aligned} \tag{1}$$

where $P_{err_{m,n}}$ is the probability of error of the (m, n)-th module computed with a normalized version of the probability of error model. $P_{max_{m,n}}$ is a parameter, selected by the user, which determines the maximum probability of error of each module. $J_{m,n}(\alpha, \beta, \alpha_c, \beta_c, p_c)$ is the distortion quality metric defined in [2] computed on the $(m, n)-$th overlapping window. Since the optimization process is performed independently for each window, it can be parallelized and a different bound in the probability of error $P_{max_{m,n}}$ can be given to each module. For instance, a higher bound in the probability of error can be given to those modules that are modified during the first stage. Finally, since color differences are measured in the HSL color space which is a perceptually uniform color space [2], to obtain the optimal RGB values of a modified pixel given the target luminance Y, the input image $\mathbf{I}$ is first transformed to the HSL color space $(H, S, L) = T(R, G, B)$. Then, the H and S components are kept fixed and the L component is changed depending on the luminance Y. An optimization problem is formulated in [2] to find the value of L, which is computationally expensive due to the required number of evaluations of the objective function. We, on the other hand, determine L from the piecewise, linear, and monotone function $Y = f(L)$ as follows

$$L^* = \begin{cases} \dfrac{Y}{2\,Y_{0.5}} & \text{if } L \leq 0.5, \\[2ex] \dfrac{Y - 2\,Y_{0.5} + 1}{2 - 2\,Y_{0.5}} & \text{si } L > 0.5, \end{cases} \tag{2}$$

where $Y_{0.5}$ is the luminance value when $L = 0.5$, i.e. for $(R, G, B) = T^{-1}(H, S, 0.5)$, and $Y = 0.2989R + 0.5870G + 0.114B$.

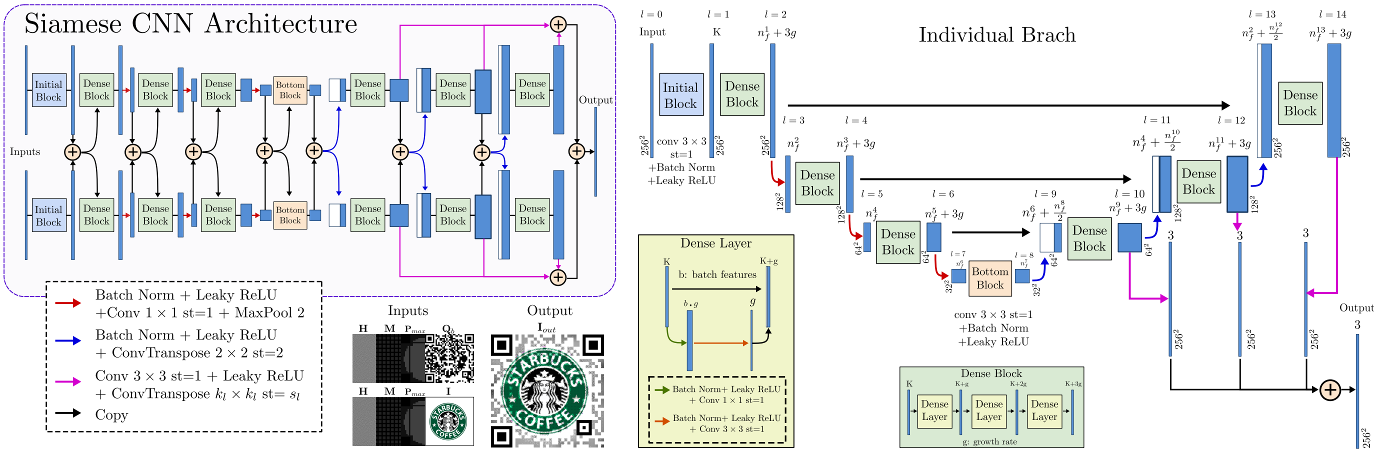

Fig. 2. Siamese CNN Architecture for QR code image embedding conformed of two identical branches (left). The architecture of each branch (right) is based on U-Net [6] and dense blocks for feature reuse [7]. The inputs for the first branch are the QR code $\mathbf{Q}$ and the masks $\mathbf{P}_{max}$, $\mathbf{M}$, and $\mathbf{H}$, and for the second branch are the color image $\mathbf{I}$ and the masks $\mathbf{P}_{max}$, $\mathbf{M}$, and $\mathbf{H}$. n_f^l is the number of features maps in layer l.

B. Fast Deep Learning-Based QR Code Image Embedding

Even though the optimization-based method presented in Sec. II-A successfully embeds the QR code into the color image with a constraint in the maximum probability of error, this process is very slow and computationally expensive. Inspired by the developments in learning-based image operators [8], we propose a learning-based QR code image embedding operation by means of a convolutional neural network architecture (CNN). Different architectures have been explored to learn an image operator f that takes a color image $\mathbf{I}$ as input and transforms its content without changing its dimensions [8] e.g. image processing operators like multiple variational models, non-local dehazing, photographic style transfer, and non-photo-realistic stylization. In our case, we aim at learning an operator that transforms multiple input images to the desired output image $\mathbf{I}_{out}$; all with the same size. Additionally, the network should adjust to different image resolutions, generating as output an RGB image of the same size as the inputs.

A naive approach would be to concatenate all input images and use a fully-convolutional network proposed either for fast image processing [8] or for pixel-wise classification [6] with regression loss to produce continuous colors. After experimenting with many already proposed network architectures, we designed the proposed architecture colored-QRNet, shown in Fig. 2, achieving the best performance at embedding an image into a QR code. Colored-QRNet has a siamese architecture in which the baseline QR code $\mathbf{Q}_b$ and the color image $\mathbf{I}$, each concatenated with the masks $\mathbf{P}_{max}$, $\mathbf{M}$, and $\mathbf{H}$ are fed to identical branches and features from the two branches are combined at different scales as depicted in Fig. 2, blending the features from the different images in each scale. As in [9], each network branch is inspired on U-net for multi-scale analysis [6] and dense blocks for feature reuse [7]. The siamese structure is based on architectures previously used in stenography and image recognition settings [10], [11]. Different from the approach in [9], the embedding happens in the color image instead of its luminance.

Let the neural network architecture colored-QRNet be denoted by $E()$ and $\mathcal{K}$ be the set of parameters across the whole network. Then $E_{\mathcal{K}}()$ is trained on a set of inputs-output pairs $\mathcal{D} = \{\mathcal{I}_i, f(\mathcal{I}_i)\}$ where $(\mathbf{I}_{out})_i = f(\mathcal{I}_i)$ and $\mathcal{I}_i = \{\mathbf{I}_i, \mathbf{M}_i, \mathbf{H}_i, \mathbf{Q}_i, \mathbf{P}_{max_i}\}$ correspond to the i-th element in the dataset. To map the inputs $\mathcal{I}$ to the desired output $\mathbf{I}^{out}$, the network parameters $\mathcal{K}$ are optimized, by minimizing the following regression loss across all the images in the dataset:

$$E^*_{\mathcal{K}} = \arg\min_{E_{\mathcal{K}}} \frac{1}{N} \|\mathbf{I}_{out} - E_{\mathcal{K}}(\mathbf{I}, \mathbf{Q}, \mathbf{M}, \mathbf{H}, \mathbf{P}_{max})\|_2^2 \quad (3)$$

where N is the number of pixels in each of the input images. Even though this loss minimizes the mean-squared error (MSE) in the RGB color space and the MSE is known to have limited correlation with perceptual image fidelity, experiments in [8] demonstrated that image operators trained to minimize the MSE yield high accuracy in terms of other measures like the peak signal to noise ratio (PSNR) and the structural similarity index (SSIM).

The optimization-based method for image embedding introduced in Sec. II-A was used for the generation of the inputs-output dataset $\mathcal{D}$. The input images $\mathbf{I}$ were selected at random from the DIV2K dataset [12] that contains 2K resolution high-quality images. These input images $\mathbf{I}$ were randomly resized to train the network at different resolutions; hence the resolution of the images in our data set range from 256×256 to 2000×2000 pixels. For the standard QR codes $\mathbf{Q}$, versions 3, 4, and 5 with random alphanumeric messages were considered. The maximum probability of error mask $\mathbf{P}_{max}$ was designed to be constant with value P_{max} within the QR code and zero on the finder patterns. The constant value P_{max} was randomly varied from 0.01 to 0.8. With respect to the mask $\mathbf{M}$ selecting the central pixels, the ratio $R = d_a/w$ was randomly varied from 0 to 1. The final data set contains $|\mathcal{D}| = 830$ samples for training and 300 for validation. Additionally, data

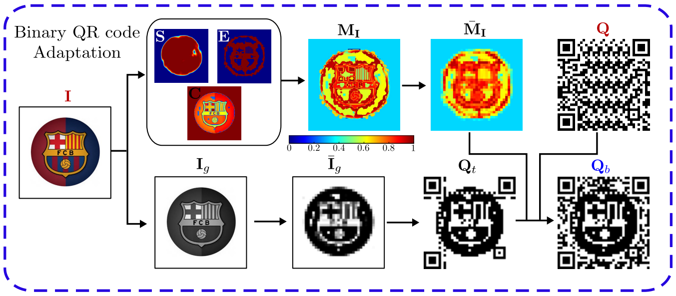

Fig. 3. First Stage corresponding to Binary QR Code Adaptation. The inputs are the original color image $\mathbf{I}$ and the traditional black and white QR code $\mathbf{Q}$ and the output is the binary adapted QR code $\mathbf{Q}_b$.

augmentation was applied to the dataset, further increasing the diversity of the samples. To this end, each image in the dataset was randomly cropped to 256×256 and randomly flipped horizontally [13]. The proposed architecture $E_{\mathcal{K}}()$ was then trained by minimizing Eq. (3) with the Adam optimizer [14]. A learning rate of 0.001 and betas equal to (0,0.999) were used. The stopping criterion was based on the stabilization of the value of the loss function on the training set and the validation set. The total training time was 2 days. The batch size was set to 4. The network was trained using TensorFlow 2.1 and a GeForce GTX 1080 Ti graphic card.

III. Binary QR Code Adaptation

Cox et al. in [3] showed that a standard QR code $\mathbf{Q}$ can be modified without affecting its decoding rate. Given that the padding bits of the original bitstream $\mathbf{s_o}$ in the standard QR code $\mathbf{Q}$ do not carry relevant information, they can be modified without altering the message. However, if any padding bit is modified, the error correction bits should be updated accordingly. Otherwise, the error correction algorithm would fail and, consequently, the decoding would be unsuccessful. The padding bits can be modified and the error correction bits updated through the exclusive-or operation (XOR). Recently, Xu *et al*. [15] proposed a method to schedule the changeable modules based on the minimization of the contrast between the blended image $\mathbf{I}$ and the QR code black-and-white modules. Although this strategy improves the visual performance, it can generate noise patterns on important parts of the image. For this reason, in this paper, changeable modules are scheduled by considering not only the minimization of the contrast, but also prioritizing the edges and the saliency region within the image. Hence modules in the QR code $\mathbf{Q}$ that correspond to pixels in $\mathbf{I}$ with highest contrast (darkest/lightest) and that belong to either an edge or the saliency region are schedule for modification. The image priority mask $\mathbf{M_I}$ is then computed as $\mathbf{M_I} = \lambda_S \mathbf{S} + \lambda_E \mathbf{E} + \lambda_C \mathbf{C}$ where $\mathbf{S}$ is a saliency map computed using a pyramid feature attention network [16], $\mathbf{E}$ identifies with '1' the edges of $\mathbf{I}$ using a Canny edge detector, and $\mathbf{C}$ is the contrast attention mask [15]. The values of λ_S, λ_E, and λ_C are equal in our experiments. A general scheme of this process is depicted in Fig. 3. The per-module weight $\bar{\mathbf{M}}_\mathbf{I}$ is computed from the priority mask $\mathbf{M_I}$ as $(\bar{\mathbf{M}}_\mathbf{I})_{m,n} = \frac{1}{w \times w} \sum_{i,j \in \mathbf{B}_{m,n}} (\mathbf{M_I})_{i,j}$where $\mathbf{B}_{m,n}$ represents the (m,n)-th QR code module of size $w \times w$. Modules with higher weight, which are considered to be more important within the image are then schedule for modification.

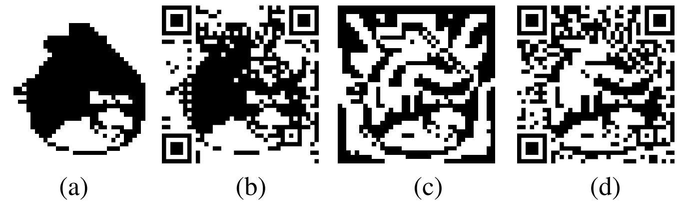

Fig. 4. (a),(b) Binary QR code proposed in [17] and (c),(d) proposed adapted binary QR code (bottom). (a),(c) $\bar{\mathbf{I}}_g$ obtained after binarization. (b),(d) Binary baseline QR code $\mathbf{Q}_b$.

Traditionally, the target QR code $\mathbf{Q}_t$ is obtained from the color image $\mathbf{I}$ through a module-based binarization method as proposed in [15], [17]. This binarization method, however, does not account for the local thresholding procedure performed by popularly used standard QR code decoders as it is in the case of the ZXing [5] library, causing the decoding to fail under unexpected circumstances. As it can be observed in Fig. 4, the module-based binarization and the local binarization outcome can be quite different on even regions of the image. While the decoding robustness is not notoriously affected on black and white QR codes, this effect can detriment the decoding robustness when an image is embedded into a QR code. In this paper, a per-module local binarization method that is more robust against this problem is proposed. To this end, we first compute a local threshold for each module $\mathbf{B}_{m,n}$ as the mean value of the image intensity $\mathbf{I}_g$ in an overlapping window $\mathbf{W}_{m,n}$ of 5×5 modules centered at the (m,n)-th module in the QR code: $t_{m,n} = \frac{1}{5w \times 5w} \sum_{(i,j) \in \mathbf{W}_{m,n}} (\mathbf{I}_g)_{i,j}$, where $(\mathbf{I}_g)_{i,j}$ is the luminance value of the (i,j)-th pixel. Each pixel in the image $\mathbf{I}_g$ is then binarized based on the corresponding local threshold $t_{m,n}$ to generate $\mathbf{I}_b$. Next, a per-module grayscale value is obtained as: $(\bar{\mathbf{I}}_g)_{m,n} = \sum_{i,j \in \mathbf{B}_{m,n}} (\mathbf{I}_b)_{i,j} \, (\mathbf{G_W})_{i,j}$, where $\mathbf{G_W}$ is a scaled Gaussian Distribution. The target QR code image $\mathbf{Q}_t$ is determined from the per-module grayscale image $\bar{\mathbf{I}}_g$ as:

$$(\bar{\mathbf{I}}_b)_{m,n} = \begin{cases} 1 & \text{if } (\bar{\mathbf{I}}_g)_{m,n} \geq 0.5, \\ 0 & \text{if } (\bar{\mathbf{I}}_g)_{m,n} < 0.5. \end{cases} \tag{4}$$

The proposed binarization method increases the robustness as shown in [9].

IV. Experiments and Results

A. Speed

In Table I, the run-time of the proposed method is compared to the run-time of the optimization-based method proposed in [2]. The first stage, which can be incorporated into both methods, takes less than 2 seconds in our experiments. Regarding the second stage, the optimization-based method was originally implemented in MATLAB and later optimized to be faster using C++. As observed in Table I, the proposed

TABLE I

SPEED FOR PROPOSED FAST DEEP LEARNING-BASED QR CODE IMAGE EMBEDDING VS THE OPTIMIZATION-BASED QR CODE IMAGE EMBEDDING FOR VARYING IMAGE RESOLUTIONS. OUR METHOD PROVIDES RESULTS WITH SIMILAR ROBUSTNESS AND VISUAL QUALITY BUT IT IS MUCH FASTER.

Image Size	Optimization-based		Colored-QRNet	
	MATLAB	C++	GPU	CPU
256×256	4.03×10^3 s	41 s	1.48 s	0.95 s
512×512	1.13×10^4 s	132 s	1.68 s	2.15 s
1024×1024	1.18×10^4 s	515 s	2.38 s	7.41 s

TABLE II

AVERAGE VISUAL QUALITY COMPARISON OF QR IMAGES GENERATED BY COLORED-QRNET AND THE OPTIMIZATION-BASED ALGORITHM.

Method	PSNR	MSSIM	J
Colored-QRNet	11.6914	0.5685	0.3156
Optimization-based	10.8111	0.4918	0.3507

learning-based framework running either on the CPU or the GPU is still much faster.

B. Visual Quality

The peak signal to noise ratio (PSNR), the multiscale structural similarity index (MSSIM), and the visual metric introduced in [2] are used here for image quality assessment on a testing set of 100 QR images with varying robustness parameter P_{max}. In Table II, the proposed fast learning-based embedding method is compared to the optimization-based method explained in Sec. II-A. It is observed that visual quality metrics are very similar for both methods, being slightly better for QR images generated by the proposed colored-QRNet.

C. Decoding Robustness

The robustness of the generated QR images by the proposed framework was evaluated experimentally. The experiment consisted of manually scanning QR Images with varying robustness parameter P_{max}. Each printed QR image of size 3.15in$\times$3.15in was scan 5 times and the scan was considered successful if the decoder returned the message within 5 seconds. A total of 50 QR images from the testing set were used in this experiment. As depicted in Fig. 5, the decodability rate decreases as the robustness parameter P_{max} decreases.

V. CONCLUSION

In this paper, novel deep learning techniques are leveraged to significantly reduce the time in the generation of visually pleasant QR codes with robustness guarantees. The first stage modifies the binary QR code without affecting the decoding rate and the second stage instantly embeds a QR code into a color image, being decodable by QR code standard applications. The proposed local binarization for the generation of the adapted binary QR code further improves the robustness of the QR images. The method proposed was tested and compared with the baseline method introduced in [2]. The decoding robustness and the visual quality of the generated embedded images were verified. The proposed adaptive window for

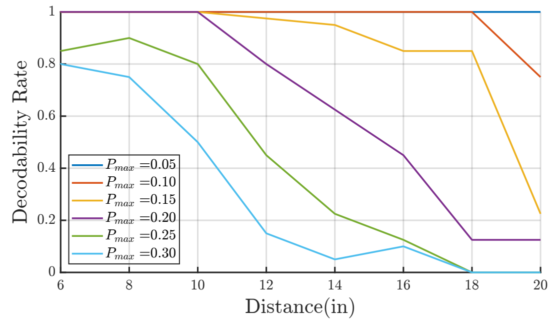

Fig. 5. Decodability rate vs scanning distance for QR images generated by colored-QRNet using different robustness parameters $P_{max} = \{0.05, 0.10, 0.15, 0.20, 0.25, 0.30\}$.

per-module QR code optimization improves considerably the decoding robustness of the QR code, taking into account more real scanning conditions.

REFERENCES

[1] C. Mullen, *The pandemic has given QR codes a shot in the arm. Will it last post-Covid?*, 2020 (accessed November 15, 2020), bizjournals.com/bizwomen/news/latest-news/2020.

[2] G. Garateguy, G. R. Arce, D. Lau, and O. Villarreal, "QR images: Optimized image embedding in QR codes," *IEEE Trans. Image Process.*, vol. 23, pp. 2842–2853, Jul. 2014.

[3] R. Cox, "Qart codes (2012, apr.)," https://research.swtch.com/qart, accessed: 2018-06-30.

[4] J. B. Rodriguez, G. R. Arce, and D. L. Lau, "Blue-noise multitone dithering," *IEEE Trans. Image Process.*, vol. 17, no. 8, pp. 1368–1382, 2008.

[5] S. Owen, "Multi-format 1d/2d barcode image processing library with clients for Android, Java and C++ (ZXing)," accessed: 2018-05-30.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Int. Conf. on Med Image Comput Comput Assist Interv*. Springer, 2015, pp. 234–241.

[7] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit*, 2017, pp. 4700–4708.

[8] O. Chen, J. Xu, and V. Koltun, "Fast image processing with fully-convolutional networks," in *Proceedings of the IEEE Int. Conf. on Computer Vision*, 2017, pp. 2497–2506.

[9] K. Pena-Pena, D. L. Lau, A. J. Arce, and G. R. Arce, "Qrnet: fast learning-based qr code image embedding," *Multimed. Tools Appl.*, pp. 1–20, 2022.

[10] E. Wengrowski and K. Dana, "Light field messaging with deep photographic steganography," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit*, 2019, pp. 1515–1524.

[11] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015.

[12] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *IEEE Conf Comput Vis Pattern Recognit (CVPR) Workshops*, July 2017.

[13] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit*, 2017, pp. 1125–1134.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] M. Xu, H. Su, Y. Li, X. Li, J. Liao, J. Niu, P. Lv, and B. Zhou, "Stylized aesthetic QR code," *IEEE Trans. on Multimedia*, vol. 21, no. 8, pp. 1960–1970, 2019.

[16] T. Zhao and X. Wu, "Pyramid feature attention network for saliency detection," in *IEEE Conf Comput Vis Pattern Recognit (CVPR)*, 2019.

[17] M. Xu, Q. Li, J. Niu, H. Su, X. Liu, W. Xu, P. Lv, B. Zhou, and Y. Yang, "Art-up: A novel method for generating scanning-robust aesthetic qr codes," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 17, no. 1, pp. 1–23, 2021.