Detection and Iterative Decoding of a 2D Alphabetic Barcode

Wadih Sawaya, Stéphane Derrode, Mustava Ould-Barikalla and Jacques Rivaillier

***Abstract*— An algorithm dedicated to the detection and decoding of an alphabetic 2D barcode, namely the Alphacode, from a captured image is developed. When the camera is freely handled, images are degraded by several artifacts (motion, acquisition, low resolution...), making recovery of the code a more difficult task. We propose a two-stage strategy. The code structure is first recovered using image processing steps including localization, geometric correction and code grid structure retrieval. The code is then recognized using an iterative structure based on a factor graph representation of the displaying/reading process and making use of the specific structure of the code. The proposed algorithm is evaluated in case of strongly blurred images and shows high improvements in performances compared to the basic processing which is unable to decode reliably.**

I. Introduction

With the improvement of the quality and the resolution of their equipped camera, mobile phone are no more restricted to radio communication. Used as a capture device in conjunction with barcodes, mobile phones are addressed to interact with the real-world [3]. Emerging applications, for large public as well as professional usages, require drastic increasing in coding capacity and robustness with respect to the usual 1D barcode. Almost all solutions converge now to 2D barcodes meeting consequently the markets of identification (e.g. electronic stamps), authentication (e.g. counterfeit medicines) and interacting (e.g. e-ticketing).

Many 2D barcodes are now available such as Datamatrix, Maxicode, QR-code and PDF417 (see [1] for an example of the different symbols). Beside all of these codes, the Alphacode, patented by the eponymous society [2], is built on a very different principle. Whereas, the aforementioned 2D barcodes represent a monolithic block of symbols generated by dedicated software, Alphacode is based on a predetermined alphabet defined by a font a characters, the Dotem font, and a given code consists with a chain of characters. The number of characters is not limited allowing to code as much information as needed for a given application. An example of an Alphacode word with five characters (two columns each) is shown in Fig. 1.a. This font based implementation grants Alphacode with numerous advantages especially in flexibility and interoperability. Standard keyboards, word processors and printers are the only materials required to generate a code on a physical support (screen, paper …). Remarkably enough, the Dotem alphabet is full compatible with Unicode norm (8, 16 and 32).

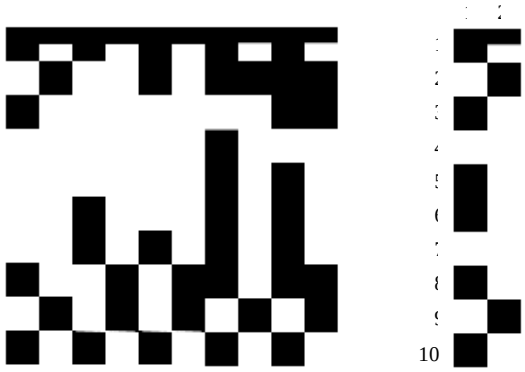


Fig. 1. A 2D Alphacode illustration. (a) a word of five characters. (b) an isolated character with two columns and ten rows.

In order to confirm this novel technology for the identification, authentication and mobile markets we propose robust solutions for detecting and decoding the Alphacode. Most of the barcode readers are designed with limited signal processing circuitry [1]. The response time satisfying the human operator wins out over quality, or false decoding and rejections. Besides, classical artifacts inherent to low-quality cameras (e.g. sensor noise, sensor resolution) or to motion caused by camera manipulation by end-user or to unfocused image and to geometric distortion…impact drastically reading performances [4]. Classical processing techniques have a rate of good detection in the context described above at best of about 90% before any error correction blocs. This is definitely not sufficient when aiming at developing robust, reliable and sustainable receivers. In digital transmission, error probability is a quality measure used to evaluate different receiver designs. A value of 10^{-6} is a commonly acceptable error rate for data. We propose then cutting edge solutions that enable us to attain this low error probability when decoding the Alphacode with standard cameras.

We propose a two-steps strategy which consists first in localizing the word and recovering the grid structure, second, in decoding using iterative structure based on statistical processing and making use of the Dotem structure.

Manuscript received June 29, 2009. This work was supported by the *Institut TELECOM* (IT) and *Fondation TELECOM*, France.

W. Sawaya is with IT / Télécom Lille1/LAGIS, Cité Scientifique, Rue Guglielmo Marconi, 59653 Villeneuve d'Ascq, France (wadih.sawaya@telecom-lille1.eu).

M. Ould-Barikalla is with TELECOM Lille1.

S. Derrode is with Institut Fresnel (CNRS UMR 6133), and Ecole Centrale Marseille, France (e-mail: stephane.derrode@centrale-marseille.fr).

J. Rivaillier is with Alphacode®, Paris, France (e-mail: jacquesrivaillier@orange.fr).

Fig. 2. A typical image illustrating the various artifacts encountered in barcode reading from a camera-equipped mobile phone.

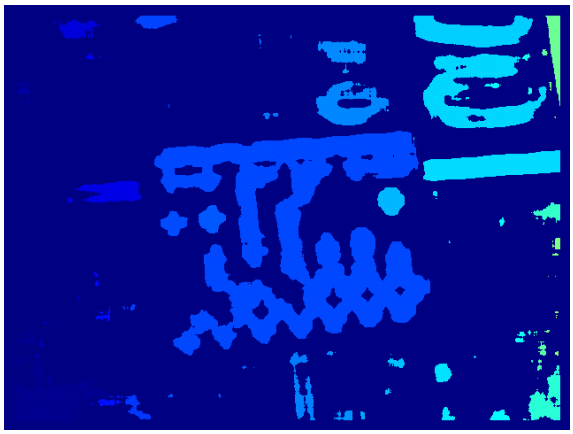
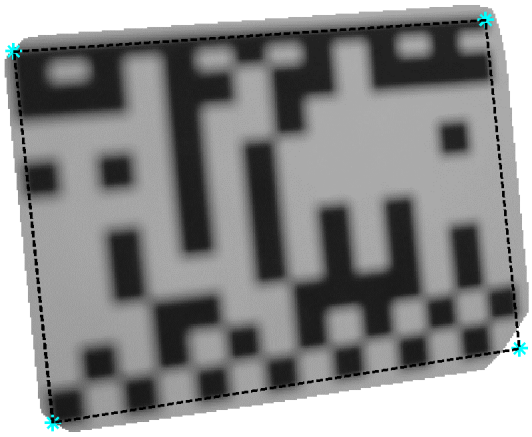


Fig. 3. Connected component based on image gradient intensity.



(a)

(b)

Fig. 4. Code rectification from estimated projective transformation.

The remaining of the paper is as follows. Section 2 presents the structure of a Dotem character. Section 3 exposes the image processing steps to localize the code and to correct geometrical distortions due to acquisition. Section 4 studies some optimal decoding strategies, taking into account image blur. Finally, Section 5 concludes on the proposed algorithm and discusses possible improvements.

II. Dotem Code Structure

In the basic Dotem alphabet presented in Fig. 1.b. (namely truetype font *DoteM02*), a character is made of at least two columns and ten rows of dots. A dot is a basic black or white square element designing a binary symbol. Dots from rows 2 to 7 are called "data dots" and allow to code up to 2^{12}=4096 characters in a 2-column structure, 2^{18}=242164 characters for a 3-column structure… Rows 1 and 10 have a fixed pattern used to make reading easier, especially the half dot in row 1, column 2. The black dot in row 9, last column (see Section 2.2) is an indicator for the end of a character. Dots from row 7 equip the character with a parity check binary element used to enhance the decoding process (Section 4).

III. Image Rectification

When acquiring a 2D barcode with a mobile camera such as a webcam or a camera in a mobile phone, the so-obtained image is degraded by several artifacts [4] (see figure 2 for a typical example):

- **Complex background:** the image may contain other objects lying in the scene which gives arise to some difficulties in detecting and localizing the code in the image;
- **Distortions:** geometric distortions can be expected from the acquisition angle between the sensor and the code resulting in a projective deformation. A step is required to rectify the image in order to recover the rectangular-shaped code.
- **Noise and blur:** Apart from the inherent sensor noise, images are blurred by (i) motion during acquisition, and (ii) necessary interpolation for rectification. Hence dots intensity runs on neighboring ones, making the decision on the state of a dot more difficult to take.

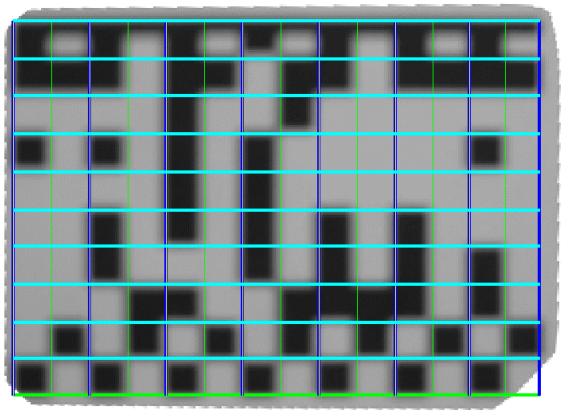


Fig. 5. Result of grid structure recovery. The blue vertical lines denote character separation.

In this section we develop an algorithm for the recovery of the underlying grid from an image as in Fig. 2. The latter will be used to illustrate the three algorithm steps.

A. *Code localization (Fig. 3)*

To localize the region of interest inside a complex scene, we opt for the image gradient which is able to exhibit code edges. Then we label all connected components based on local gradient density. Using a priori geometrical property of the desired code, we select the region of interest among possible candidates.

B. *Image rectification (Fig. 4)*

As explained before, images are distorted by some projective transformation. In this step, we intend to estimate the geometric parameters and apply the projection backward to get the code rectified. The first step consists in retrieving the four corners of the parallelepiped including the code, see Fig. 4. a. We use the points of the convex hull to set the left, right, up and down segments, which give by intersection the searched corners.

Then, the parallelepiped corners are registered to a rectangle one to estimate the projection matrix. The projection matrix we get for image in Fig. 2 (see [3]) is

$$P = \begin{pmatrix} 0.94 & 0.07 & 0 \\ -0.10 & 1.07 & 0 \\ 3.14 & -21.24 & 1 \end{pmatrix}$$

which gives the image in Fig. 4. b. We may expect from this transformation that dots are stretched according to one direction, giving rectangular-shaped dots.

C. *Grid recovery (Fig. 5)*

To recover the number of characters and the number of columns by character we make use of the different tags present in each character: (i) the semi-dot in the first row and second column, (ii) the black dot in the ninth row and last column. Basic image processing based on column and line profiles allow us to recover the code structure. For the studied example, the mean width of a dot is 18 pixels, and the mean height is about 18.7 pixels.

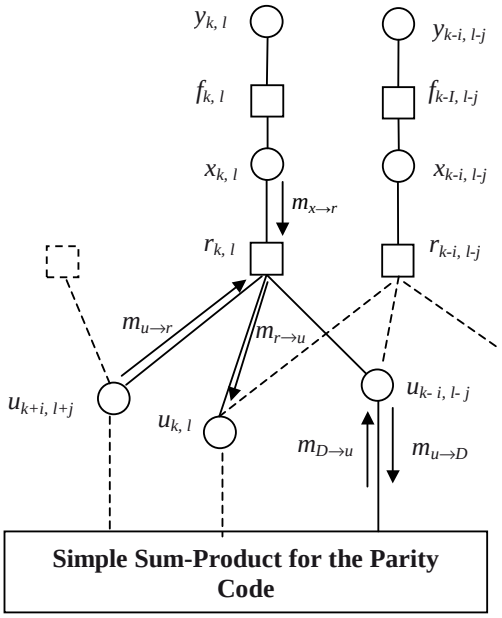


Fig. 6. Factor graph representation of the 2D ISI channel combined to the iterative equalizer making use of the parity code.

The processing implemented to recover the grid structure takes about 1.5 seconds for a PC with an Intel Core 2, 2.0 GHz processor, using Matlab.

IV. Code Reading

According to the grid the observed image is now a matrix **Y** of observed dots with elements:

$$y_{k,l} = \sum_{i=-n}^{n} \sum_{j=-p}^{p} u_{k-i,l-j} h_{i,j} + w_{k,l} \,. \qquad (1)$$

Elements $u_{k,l}$ represent black and white modules (dots) of the original code, $w_{k,l}$ are zero-mean additive white Gaussian noise samples and the $[h_{i,j}]$ describe the blur affecting the image. From a communication point of view, expression (1) models the reading/displaying process as a linear 2D ISI channel. A factor graph representation of this model is derived. We propose then a robust decoder based on a message-passing algorithm running on this graph and involving interactions between adjacent modules or dots. We make use of the parity dot to enhance performances with an iterative structure. We aim then at proposing a simplified version of the studied message passing algorithm without attempting much on performances, motivated by the design of receivers with a humanly acceptable complexity. We finally show some simulation results and conclude on the necessity of using such structure when simple binarization is not able to satisfy a given quality of service.

The factor graph consists of variable nodes (circles) and factor nodes (squares) connected by edges (Fig. 6.). Variable nodes represent the hidden data dots $\{u_{k,l}\}$, the resulting gray-levels from ISI $\{x_{k,l}\}$ and the observed variables $\{y_{k,l}\}$. Factor nodes $r_{k,l}$ represent local functions connecting a

subset of these variables while $f_{k,l}$ is the likelihood function $f_{Y_{k,l}}(y_{k,l} / x_{k,l})$. The sum-product algorithm (SPA) is a message-passing algorithm that computes the marginal *a posteriori* probabilities (APP) of any data dot $P(u_{k,l}/\mathbf{Y})$, by passing "messages" through the factor graph [5].

We apply SPA to detect data dots (detection step). The outputs of this detector constitute soft extrinsic information about each dot and enter the parity decoder (decoding step). We realize an iterative detection and decoding procedure as the detector and the decoder exchange extrinsic information between each other for several iterations. In the SPA, messages are resumed by the following rules [5] for $-n \le i \le n$ et $-p \le j \le p$ (Fig. 6.):

1) Messages from variable u nodes to factor nodes r:

$$m_{u_{k,l}\to r_{k-i,l-j}} \propto \left(\prod_{(i',j')\neq(0,0)} m_{r_{k-i',l-j'}\to u_{k,l}} \right) m_{D\to u_{k,l}} \quad (2)$$

In the first formula we distinguish the message of the form $m_{D\to u}$ coming from the decoder and calculated according to the extrinsic information that the latter deliver:

$$\begin{aligned} P(u_{k,l}=1) &= m_{D\to u_{k,l}}(u_{k,l}=1) \\ P(u_{k,l}=0) &= m_{D\to u_{k,l}}(u_{k,l}=0) \end{aligned} \quad (3)$$

Practically, all messages $m_{r\to u}$ in (2) are initialized to 1 in order to avoid short cycles. We thus have $m_{u\to r} = m_{D\to u}$.

2) Messages from factor nodes r to variable nodes u :

$$\begin{aligned} m_{r_{k,l}\to u_{k-i,l-j}} \propto & \sum_{U'} \sum_{x_{k,l}} r_{k,l}(x_{k,l}, U) \; m_{x_{k,l}\to r_{k,l}} \\ & \cdot \prod_{(i',j')\neq(i,j)} m_{u_{k-i,l-j}\to r_{k,l}} \end{aligned} \quad (4)$$

In expression (4) we define $U = \{u_{k-n,l-p}, \ldots, u_{k+n,l+p}\}$, $U' = U - \{u_{k-i,l-j}\}$, and $r_{k,l}$ is given by the following logical proposition:

$$r_{k,l}(x_{k,l}, U) = \left[x_{k,l} == \sum_{i=-n}^{n} \sum_{j=-p}^{p} h_{i,j} u_{k-i,l-j} \right] \quad (5)$$

such that $r(x, U) = [x = = \gamma(U)]$ is defined by the following, $r(x, U) = 1$ if $x = \gamma(U)$ and 0 if not. Finally, we have $m_{x_{k,l}\to r_{k,l}} = f_{Y_{k,l}}(y_{k,l} / x_{k,l})$. Assuming a white and Gaussian additive noise we rewrite (4) as:

$$\begin{aligned} m_{r_{k,l}\to u_{k-i,l-j}} \propto & \sum_{x_{k,l}} \exp\left(-\left(y_{k,l} - x_{k,l} \right)^2 / N_0 \right) \\ & \cdot \sum_{U'/r_{k,l}=1} \left(\prod_{(i',j')\neq(i,j)} m_{D\to u_{k-i',l-j'}} \right) \end{aligned} \quad (6)$$

The second sum in (6) is the probability $P(v)$ that the intersymbol interference affecting the dot $u_{k-i,l-j}$ and observed from $y_{k,l}$ is equal to $v = x_{k,l} - h_{i,j} u_{k-i,l-j}$. The extrinsic information transmitted to decoder $L_{k,l}$ is then:

$$L_{k,l} = \sum_{i=-n}^{n} \sum_{j=-p}^{p} \log \left(\frac{m_{r_{k-i,l-j}\to u_{k,l}}(u_{k,l}=1)}{m_{r_{k-i,l-j}\to u_{k,l}}(u_{k,l}-1)} \right) \quad (7)$$

as there are $(2n+1)$x$(2p+1)$ observed variables related to one hidden variable $u_{k,l.}$

Complexity of the sum product algorithm grows exponentially with the memory size, which make its application practically unfeasible for some channels such for long memory channels. The objective of this work is primarily to propose a reduced complexity version of the sum product algorithm, aiming at proposing an algorithm with a linear complexity according to the memory size. In estimation theory, linearity is obtained when the joint density function of the hidden and the observed variables is Gaussian. Let us assume this hypothesis which could fairly be justified in case of long memory channel and yielding a Normal distribution for $P(v)$ with mean and variance

$$\mu_{i,j}^{k,l} = \sum_{(i',j')\neq(i,j)} h_{i',j'} \mu_{k-i',l-j'} \quad (8)$$

$$\sigma_{i,j}^{k,l\,2} = \sum_{(i',j')\neq(i,j)} (h_{i',j'} \sigma_{k-i',l-j'})^2 \quad (9)$$

$\mu_{k-i',l-j'}$ and $\sigma^2_{k-i',l-j'}$ are the mean and the variance of the variable node $u_{k-i',l-j'}$ calculated according to the *a priori* information provided by the parity check decoder. This approximation enables us to conclude a simple expression for the extrinsic information in (7):

$$L_{k,l} = \sum_{i=-n}^{n} \sum_{j=-p}^{p} \frac{2h_{-i,-j}\left(y_{k-i,l-j} - \mu_{k,l}^{k-i,l-j} \right) - h_{-i,-j}^2}{2\left(\sigma_{k,l}^{k-i,l-j}\right)^2 + N_0} \quad (10)$$

After several iterations this iterative structure converges and one makes the following decision:

$$m_{u_{k,l}\to D}(u_{k,l}=1) \times m_{D\to u_{k,l}}(u_{k,l}=1) \underset{u_{k,l}=0}{\overset{u_{k,l}=1}{>}} m_{u_{k,l}\to D}(u_{k,l}=0) \times m_{D\to u_{k,l}}(u_{k,l}=0)$$

which may be used by an error control decoder. We assume as in classical 2D barcodes, Reed Solomon (RS) encoder [6] protecting information from errors. The redundant symbols may be added at the end of the Alphacode chain of character. We consider an RS encoder constructed over Galois Field GF(256), and compare the data error probability obtained with the novel iterative equalizer and the classical binarisation based detector which equips almost all available readers in the market. We intentionally simulate poor quality images with high amount of blur. In these simulations the matrix modeling the blur is larger than a dot.

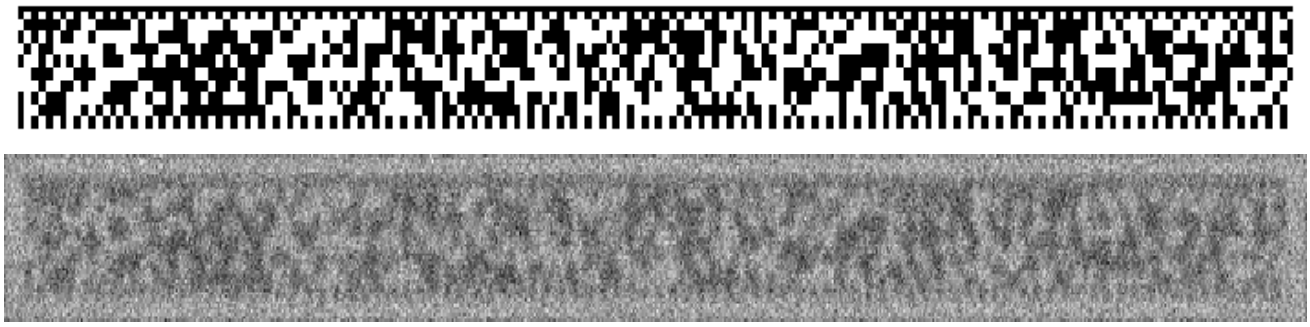


Fig. 7. A 2D Alphacode chain of character (60 data characters and 30 redundant characters) with a dot of 4x4 pixels and a blur matrix of 5x5. This image is corrupted with a noise with $E_p/N_0 = 0$ dB, where *Ep* is the mean energy received by one dot and N_0 the spectral density of the noise.

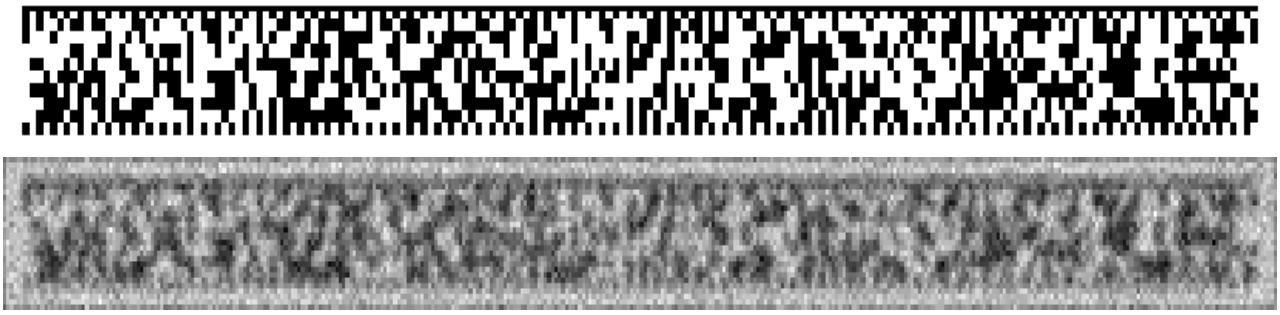


Fig. 8. A 2D Alphacode chain of character (60 data characters and 30 redundant characters) with a dot of 2x2 pixels and a blur matrix of 3x3. This image is corrupted with a noise with $E_p/N_0 = 9$ dB, where E_p is the mean energy received by one dot and N_0 the spectral density of the noise.

Fig. 7 shows the bad quality of the received image with a dot of 4x4 pixels and a blur matrix size of 5x5, and Figure 8 represents an image with a dot of 2x2 pixels and a blur of 3x3. Fig. 9 and 10 show the performances obtained for the two latter cases. It is obvious that the proposed algorithm outperforms standard readers. These figures represent the data error probability with respect to the received signal to noise ratio expressed in terms of E_p/N_0 in dB, where E_p is the mean energy received by one dot and N_0 the spectral density of the noise.

V. Conclusion

In this work, we present a study of the 2D alphabetic barcode based on the Dotem alphabet in real situation, *i.e.* under difficult acquisition conditions (noise, blur, complex background and geometrical deformation). We have implemented a two-steps solution to decode a word from an image acquired using a hand-held camera. First, we defined some image processing tasks, including code localization in a complex scene and projective-based image rectification to recover the word structure. Then, we studied the decoding process using a modified sum-product algorithm running on a factor graph modeling the displaying/reading process. We make use of the parity dot in each column to derive an iterative structure of this equalizer. This algorithm has a linear complexity which makes it a realistic and attractive reading solution for camera-phones. Reed-Solomon encoders are then used to control errors. The resulting data error rate is greatly improved compared to basic processing techniques equipping most of the barcode readers for the identification and interacting markets.

References

[1] E. Ouaviani, A.Pavan, M.Bottazzi, E.Brunelli, F.Caselli and M.Guerrero, "A common image processing framework for 2D barcode reading", *Int. Conf. on Image Proc. and App*., Manchester (U.K.), 13-15 July 1999.

[2] Principal patents for the Dotem code, held by Alphacode: (1) Dote Patent – PCT Sept. 18, 2001 N°01-02889 (2) DoteM French Patent Feb. 15, 2005 N°05-01552 (3) DoteM-R French patent, July 09, 2007 N°07-04964.

[3] M. Rohs, "Real-World Interaction with Camera-Phones", *2nd Int. Symp. on Ubiquitous Computing Systems* (UCS), pp. 39-48, Tokyo (Japan), Nov. 2004.

[4] K. Houni, W. Sawaya and Y. Delignon, "1D barcode reading: an information theoretic approach", *Applied Optics*, vol. 47(8), pp. 1025-1036, March 2008.

[5] F. R. Kschischang., B. J. Frey and H. A. Loeliger, "Factor Graphs and the Sum-product Algorithm", *IEEE Trans. Inf. Theory*, vol. 47(2), pp. 498-519, Feb. 2001.

[6] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, Wiley., England, 2005.

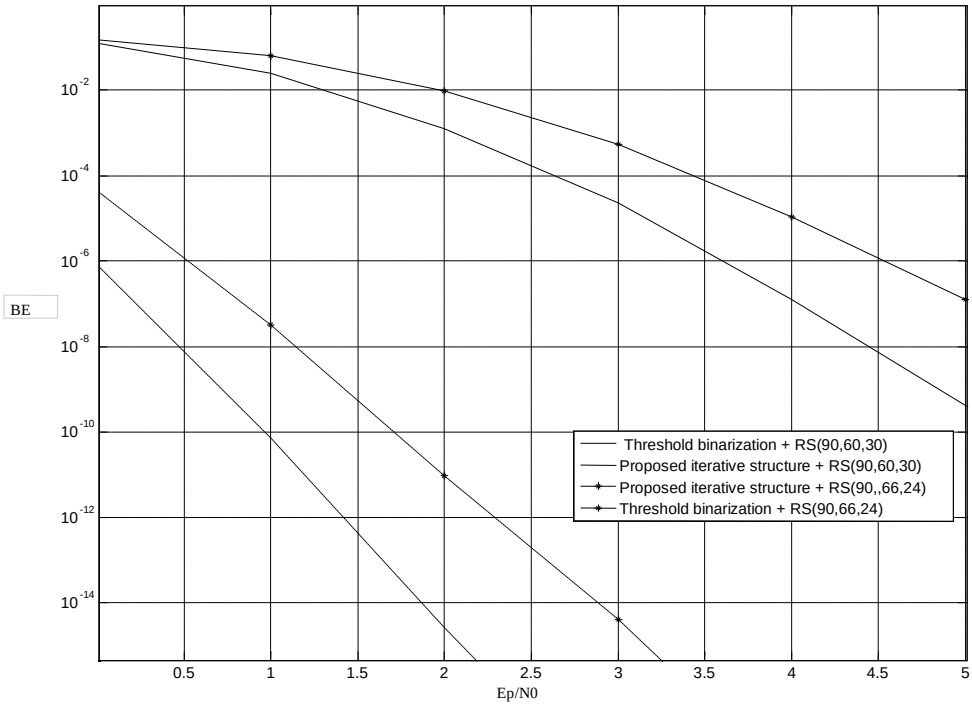


Fig. 9. Performance in bit error rate of an Alphacode with the proposed algorithm (dashed lines) compared to a classical reader (solid line). The received image have a dot size of 4x4 pixels and a blur of size 5x5. Different RS encoders are used.

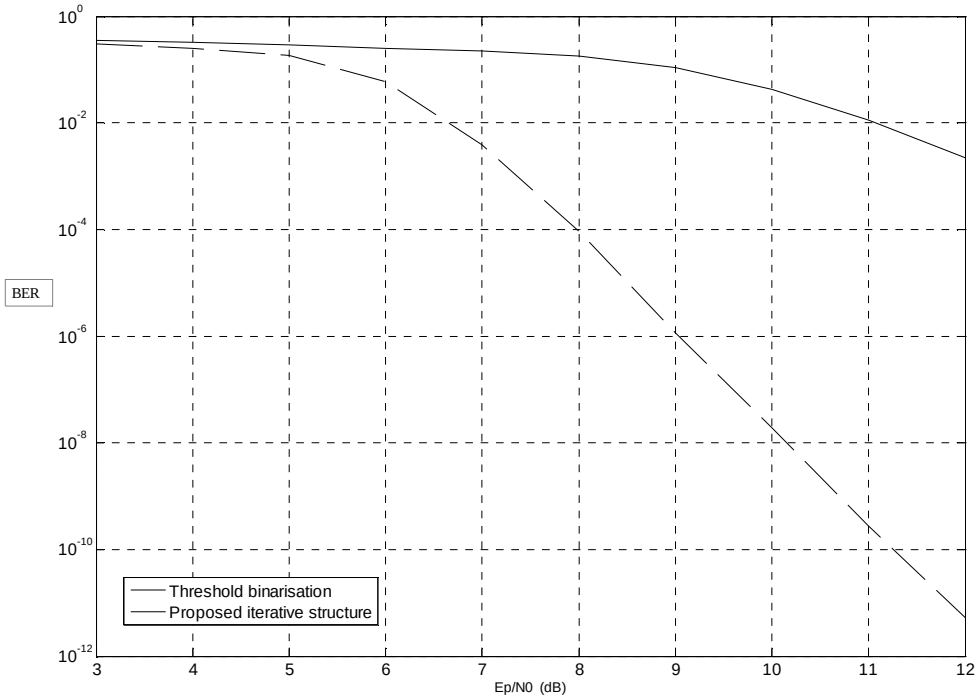


Fig. 10. Performance in bit error rate of an Alphacode with the proposed algorithm (dashed lines) compared to a classical reader (solid line). The received image have a dot size of 2x2 pixels and a blur of size 3x3. The Alphacode is encoded with a RS(90,60,31) constructed over GF(256).