PAPER WIDGETS: DISTRIBUTED "SMARTS" FOR PAPER DOCUMENTS WITH HUMAN-READABILITY

Serene Banerjee, Yogesh Sankarasubramaniam, Krusheel Munnangi, Anjaneyulu Kuchibhotla

HP Labs India, Bangalore, India

ABSTRACT

Traditional 1D and 2D barcodes provide high data density, but they are visually jarring and require isolated white margins for placement. In this work, we introduce new machine-readable "smarts" for paper documents, called paper widgets. Unlike barcodes, paper widgets have very small footprint (fraction of a sq. cm.) and a human-readable component that provides visual meaning. They can also be placed and extracted from any position on the paper document, for instance, right beside contents of interest. This paper describes in detail, the design and implementation of the encoding, extraction, and decoding of paper widgets. Constrained coding techniques are used to combat the print-scan intersymbol interference (ISI), while Gabor filtering and adaptive thresholding help extract widget regions from a scanned document. Experimental evaluations have revealed that paper widgets can be decoded from printing/scanning operations with near-100% accuracy.

Index Terms— Machine-readability, Paper documents, Gabor filter, Constrained coding, Print-scan ISI

1. INTRODUCTION

There has been growing interest in the use of barcodes with printed forms, invoices and documents. Typically, the barcode acts as a container for machine-readable data, which is then extracted and processed when the barcode is scanned. Some well-known applications include digital postage for online postal services [1], multimedia security [2], image reconstruction using side-information [3], document and image security [4, 5, 6, 7], and automated forms processing [8]. Most of these applications involve printing the barcode in isolated white margins and then recovering the data upon scanning. However, the look and feel of a barcode is considered to be visually jarring. Moreover, the barcode means little to the human eye, and there is no way of spatially relating the data on a barcode to specific portions/fields of a document. This might be important if, for instance, only certain fields desired by the user are to be extracted from a paper document.

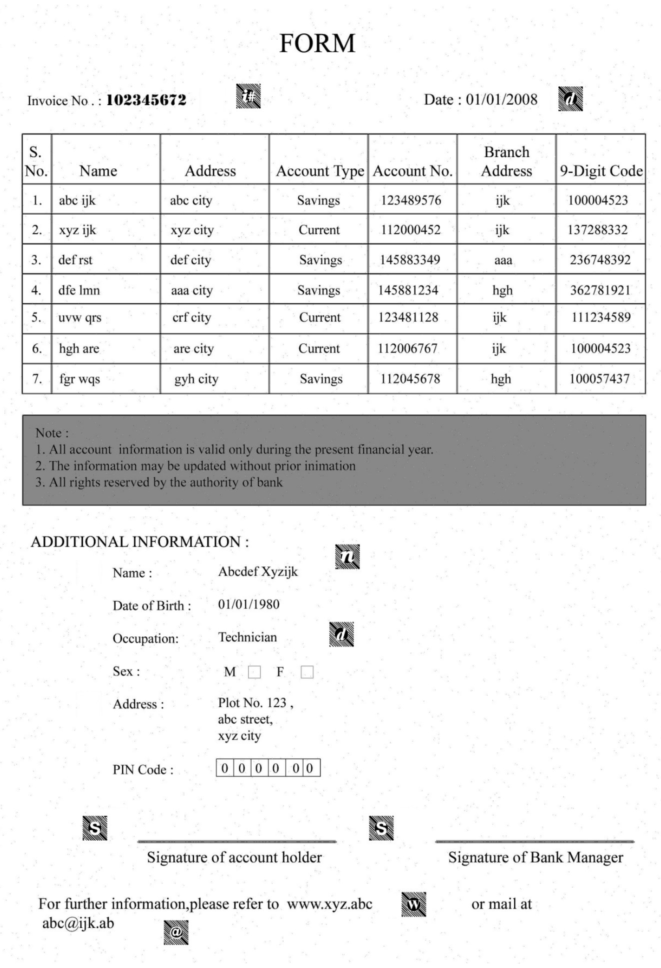
In this paper, we introduce new machine-readable "smarts" for paper documents, called paper widgets. Unlike 1D and 2D barcodes, paper widgets have very small footprint (around 0.5 sq. cm.) and a human-readable component that provides visual meaning. They can also be placed and extracted in a distributed fashion. For instance, paper widgets can be placed right beside contents of interest as shown in Fig. 1. This must be contrasted with the *centralized* placement required for 2D barcodes, which typically lump all the machine-readable data pertaining to the entire document on an isolated white margin at the bottom of the page. Furthermore, the human-readable component of paper widgets gives it an "icon"-like appeal, which can provide meaning and context to the human eye. For example, the email widget in the title area displays the icon @, which indicates that the widget contains email-addresses. Such an email widget can facilitate automatic extraction of email addresses upon scanning, as opposed to the tedious exercise of manually searching and then re-entering them (especially with the long Indian surnames!).

In the rest of this paper, we describe in detail, how to design and implement paper widgets. Clearly, the printing/scanning distortions pose a considerable challenge for our dual goal of small size and human-readability. Moreover, our vision is to make the encoding, extraction and decoding of widgets extremely light-weight so that they can be embedded into printers/scanners/AiOs or even run on handhelds, cameras, and smart phones.

****organization of paper to be filled; Simple, light-weight operations such as Toeplitz encoding, Gabor filtering, Sliding-block decoding, are used for widget encoding, extraction and decoding. This makes them amendable to embedding in the device or in the printing/scanning software****

2. RELATED WORK

There exist several machine-readable symbologies today, such as PDF417 [9], DataMatrix [10], Maxicode [11], dataglyphs [12] and QR code [13], but none of them directly address human-readability on small footprint. One variant of the QR



FORM

Invoice No . : **102345672**

Date : 01/01/2008

S. No.	Name	Address	Account Type	Account No.	Branch Address	9-Digit Code
1.	abc ijk	abc city	Savings	123489576	ijk	100004523
2.	xyz ijk	xyz city	Current	112000452	ijk	137288332
3.	def rst	def city	Savings	145883349	aaa	236748392
4.	dfe lmn	aaa city	Savings	145881234	hgh	362781921
5.	uvw qrs	crf city	Current	123481128	ijk	111234589
6.	hgh are	are city	Current	112006767	ijk	100004523
7.	fgr wqs	gyh city	Savings	112045678	hgh	100057437

Note :
1. All account information is valid only during the present financial year.
2. The information may be updated without prior inimation
3. All rights reserved by the authority of bank

ADDITIONAL INFORMATION :

Name : Abcdef Xyzijk

Date of Birth : 01/01/1980

Occupation: Technician

Sex : M □ F □

Address : Plot No. 123 ,
abc street,
xyz city

PIN Code : 0 0 0 0 0 0

Signature of account holder

Signature of Bank Manager

For further information,please refer to www.xyz.abc or mail at abc@ijk.ab

Fig. 1. **Example document with various types of paper widgets placed in a distributed fashion.**

code called Micro QR code [13] is targeted at small footprints, but it does not provide any visual significance for the human eye. There are also a few efforts which are aimed at embedding a human-readable icon or logo on top of a QR code or any other 2D barcode. For instance, [14] uses evolutionary computation algorithms to find positions for a logo to be embedded in a QR code such that the decoding is not affected. Other examples include [15] and virtual reality applications as in [16], [17]. However, most of these efforts support only certain specific icons. On the other hand, techniques like [18], [19] support human input icons, but have low data capacity. ****other efforts like VSB, EVSB, Simske using base logos, tiles, bars etc****

The above-mentioned prior art can be cast into one of two categories: either they attempt to embed human-readable icons on already existing machine-readable symbologies, or they attempt to convey information through the human-readable icons themselves. Obviously, the latter approach limits the amount of machine-readable content. The former case again compromises on data density, but in an indirect way. Embedding a logo/icon onto a 2D barcode, for instance, is equivalent to simply overwriting the corresponding barcode pixels. Thus, to maintain reliable decoding of the machine-readable content, one has to accordingly reduce data on the barcode. We believe that both these approaches are rather round-about ways of providing joint human and machine readability. In contrast, the design for paper widgets takes a bottom-up approach, factoring-in both human-readability and machine-readability while encoding the widget itself. We provide a novel Toeplitz encoding with runelngth constraints to combat the print-scan intersymbol interference (ISI), and propose Gabor filtering with adaptive thresholding to identify widget areas from the rest of the scanned document.

3. PAPER WIDGETS: DESIGN AND IMPLEMENTATION

The paper widgets system consists of three modules: 1) the encoder module (on the printing end), which encodes data onto the widget, 2) the widget extractor module (on the scanning end), which performs widget detection and extraction from the scanned document image, and 3) the decoder module (on the scannign end), which decodes the widget data. We now describe each of the modules.

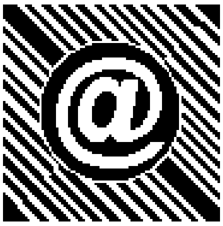
3.1. Encoder

The encoder design poses a combined challenge of human-readability on small footprint, while at the same time ensuring robustness against printing/scanning (PS) distortions. It is well-known that PS operations cause inter-symbol interference (ISI) due to their inherent low-pass nature [20]. This has the effect of "smearing" neighboring pixels, thus posing further challenges during decoding. The usual approach [21, 22] is to guard against such ISI using a naive technique called *repetition coding*. In a repetition code, each barcode pixel is printed using several identical black/white pixels, so that the entire group of repeated pixels is protected against ISI. For example, each black (white) barcode pixel could be printed as a 3 pixel $\times$ 3 pixel black (white) square. Equivalently, the printer and scanner could be set at a higher resolution as compared to the barcode image resolution, thus creating a repetition effect. Another implementation that is typically used, is to scan the barcode at sufficiently higher resolution as compared to the barcode image resolution.

However, it is well-known from the theory of digital communications [23] that the repetitive code is a most primitive and naive solution to combat ISI. In particular, the use of constrained coding can improve robustness against ISI, or equivalently, increase the information density for any given ISI level. Our novelty here is to model the ISI in the print-scan channel for a particular Toeplitz structure that is chosen for the encoding. A Toeplitz structure is a square matrix in which each descending-diagonal from left to right is a constant, as shown in the example of Fig. 2(b). However, we impose an additional minimum runlength constraint of 3 to combat the PS ISI. This means that a "strip" of black or white is at least 3 pixels wide, as seen again from the example of Fig. 2(b).

The encoder now has to map binary sequences to such an ISI-resistant runelngth sequence. The encoder mapping for a rate $1/3$, $(3, 5)$ runlength-constrained code is given in Table **??**. It can be verified that the output sequence from Table **??** is constrained to only have '0'/'1' (black/white) runs of length 3 bits, 4 bits, or 5 bits. A minimum runlength of 3 provides robustness against print-scan ISI, while a maximum runlength constraint of 5 enables widget extraction using a bank of four Gabor filters. This runlength limited sequence is then written onto the BCDT widget structure. An example of the runlength-constrained BCDT encoding is seen in Fig. 2 on the left. Note that marker sequences of 6 bits are used, which help in locating the four corners during decoding. Using a constrained code instead of the repetition code offers several advantages including robust widget extraction using Gabor-filtering, and better system margins - which can be used to reduce the widget size for a given data density, or to increase the widget data density, or to reduce the decoding bit error rates, or to suitably trade-off between all these factors.

A double-triangular structure is further used on top of this constrained Toeplitz structure to double the storage capacity. Thus, our design follows a Binary Constrained Double-triangular Toeplitz (BCDT) structure, which can currently accommodate around 10 to 15 data bytes (plus human-readable icon) within an area of 0.5 sq. cm. An example of the BCDT-based email paper widget is shown in Fig. 2(a). This is a magnified version of the email paper widget shown in the title area.



```
0 0 0 0 1 1 1 1 0 0 0
0 0 0 0 0 1 1 1 1 0 0
0 0 0 0 0 0 1 1 1 1 0
0 0 0 0 0 0 0 1 1 1 1
0 0 0 0 0 0 0 0 1 1 1
1 0 0 0 0 0 0 0 0 1 1
1 1 0 0 0 0 0 0 0 0 1
1 1 1 0 0 0 0 0 0 0 0
0 1 1 1 0 0 0 0 0 0 0
0 0 1 1 1 0 0 0 0 0 0
0 0 0 1 1 1 0 0 0 0 0
```

(a) Sample paper widget (b) A toeplitz matrix

Fig. 2. **Sample paper widget and a toeplitz matrix**

Such a BCDT structure has obvious advantages. In addition to offering PS resilience, the redundancy in the Toeplitz structure can help simplify the extraction and decoding. For example, it allows a simple Gabor-filtering approach for the quick extraction of widget areas from other tables, logos, figures, etc. which may be found in the scanned document image. It also allows better estimation of pixel values during decoding. However, the most beneficial outcome of the BCDT structure is the human-readability. Any human-readable icon/logo/character can simply be overwritten on the central one-third of the widget area without losing any of the encoded data by virtue of the Toeplitz structure. In the example of Fig. 2(a), the icon **'@'** conveys that it contains email addresses. Thus, in our design, the central one-third of the widget contains the human-readable icon and the remaining two-thirds of the widget area carries the encoded data.

The rate $1/3$ code described above helps achieve encoding density of around 10 data bytes within the widget area of 0.5 sq. cm. We have also designed a greater rate $1/2$ code, which can increase the encoding density by 50% over the rate $1/3$ code. While this rate-improvement can be used directly to increase the data density, it could also be utilized in other appropriate ways like reducing the widget size for given data density, or to reduce the decoding error rates by adding more error correction parities, or to suitably trade-off between all these factors based on further application requirements. On the other hand, one disadvantage of the rate $1/2$ code is that it has a greater maximum runlength of 8, which necessitates the use of seven Gabor filters during widget extraction. Also, the marker sequences are now 9 bits long.

3.2. Widget extractor and decoder

The widget extractor module works on the consumer side, and provides fast and easy detection and extraction of paper widgets from scanned documents. The Toeplitz structure provides spatially localized spectral features, which can be extracted using a bank of Gabor filters. The traditional Gabor filter is a sinusoidal signal of particular frequency and orientation that is modulated by a Gaussian envelope. For accurate widget extraction, sinusoids of frequencies $1/3$, $1/4$, $1/5$, and $1/6$, were chosen. The absolute values of the response of the four filters are summed and normalized to get the final response.

As the widgets contain black and white stripes, the Gabor filter response at the widgets is subsequently higher than any response coming from the document textures or print-scan variations. We adaptively threshold the Gabor filter response in order to robustly detect the widgets. At the correct threshold, the number of squares detected attains a plateau, and the difference between the number of connected components extracted and the number of squares detected, attains a minimum. For up to 20 thresholds between the mode and the maximum of the Gabor filter response, the absolute difference of the number of components extracted and the number of squares extracted is plotted, and the correct threshold is where the above graph attains its first minima. The approach is illustrated in Fig. 3 and Fig. 4. The proposed method guarantees that we always find the right threshold to extract all widgets in the document, irrespective of background textures or print-scan variations.

After the proposed optimal thresholding, morphological closing operation is performed to fill the gaps with a circular structural element of 6 pixels. Here, we note that the one-third region around the base icon is circular, where there is no Gabor filter response. As the circular structure allows maximum surface area for the regions having Gabor filter response, the structuring element of 6 pixels is sufficient to close the gaps and the widgets are extracted as square connected components. This parameter could also be tuned to the document image resolution. To reduce the computational complexity of Gabor filtering, the filtering is done on a downsized image. The location of the widgets in the original resolution image is further tuned utilizing the 3-pixel wide quite zone in the image. For each of the connected components, the horizontal and the vertical projection profile is found to locate the quite zone around the columns and rows, respectively. If the quite zone is not found in either the top, right, bottom or left boundaries, the widget boundary is appropriately adjusted to include the quite zone. The proposed block diagram for the widget extractor is shown in Fig. 5.

After extraction, the widget data now has to be decoded. This involves corner detection, auto-calibration of B/W thresholds (for printer/scanner independence), followed by soft-detection and runlength checks and finally Viterbi decoding as shown in Fig. 6. For cases where the Viterbi decoder might be prohibitively complex, we propose a heuristic sliding-block decoding algorithm that trades-off BER with complexity. We skip the associated details here due to lack of space.

4. EXPERIMENTAL EVALUATION

We have carried out extensive testing of paper widgets over a testset comprising of business forms, invoices, and purchase orders. As a test of robustness, three different document background shades were used - white, light, and dark backgrounds. On the whole, we encoded, extracted, and decoded around 200 widgets on 35 different base documents with different background shades. Table VI captures the results of our experiments using HP Laserjet 9040 for printing at the producer end, and HP Scanjet 2400 for scanning at the consumer end. By virtue of the decoder auto-calibration (preprocessing Step 2), the performance should not significantly vary if other printers and scanners are used.

In Table VI, widget extractor accuracy is defined as the percentage of the number of widgets that were correctly extracted. We did not encounter a single extraction failure (false negatives or false positives), which suggests that our widget extractor module is robust against all base documents and backgrounds.

Decoder accuracy is defined as percentage of the number of widgets decoded correctly from amongst the extracted widgets. Recall that each widget can encode upto 10 or 15 bytes of data (depending on the constrained coding used). In our computations, we declare a decoding failure for the entire widget even if one bit of data is incorrectly decoded. While this is a useful practical measure of performance, the decoding error rates are better illustrated by the byte error rate (BER), which is defined as the percentage of data bytes that are incorrectly decoded. The data BER is also given alongside for comparison.

Table 1. Experimental results of testing paper widgets

Background / Module Widget	extractor accuracy(%)	Decoder accuracy(%)	Data BER (%)
White background	100	97.8	1.74
Light background	100	97.8	0.65
Dark background	100	95.0	1.25

The accuracies listed in Table VI are obtained, in each case, by averaging over several paper widgets, each of size 0.5 sq. cm and carrying 10 bytes of data. As a comparison, the required size of a PDF417 barcode is 1.5 sq. cm and that of the QRcode is around 1 sq. cm, to store 10 bytes of data. During our experiments, we also noted that while paper widgets could easily be placed anywhere on the document, the larger size of the PDF417 and QRcode meant that they could be placed only along isolated spaces or document margins. Such a *distributed* placement adjacent to document content is an important differentiator for paper widgets. Furthermore, unlike the PDF417 and QRcode, the human-readable base icon provides meaning and context to the data carried by the widget.

5. REFERENCES

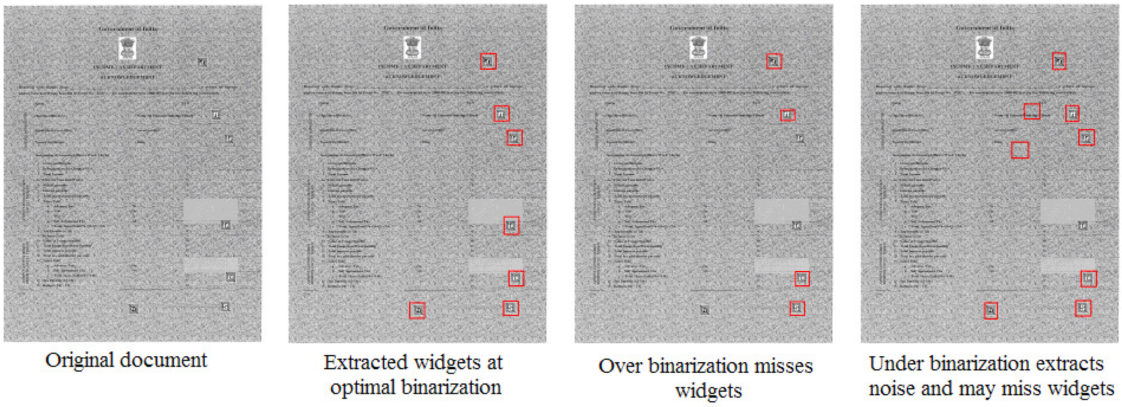
List and number all bibliographical references at the end of the paper. The references can be numbered in alphabetic order or in order of appearance in the document. When referring to them in the text, type the corresponding reference number in square brackets as shown at the end of this sentence.

6. REFERENCES

[1] France Telecom R&D, "M-commerce, an emerging sector," Website, `http://www.francetelecom.com/sirius/rd/en/ddm/en/technologies/ddm20.0405/printindex1.htm`.

[2] D. Kirovski, N. Jojic, and G. Jancke, "Tamper-Resistant Biometric IDs," in *Issue 2004-Securing Electronic Business Processes: Highlights Of The Information Security Solutions Europe 2004 Conference*. Friedrick Vieweg & Son, 2004, p. 160.

[3] D. Mukherjee and R. Samadani, "Distributed image coding for imformation recovery from the print-scan


Original document

Extracted widgets at optimal binarization

Over binarization misses widgets

Under binarization extracts noise and may miss widgets

Fig. 3. **Proposed adaptive binarization for robust widget extraction irrespective of background textures or print/scan variations.**

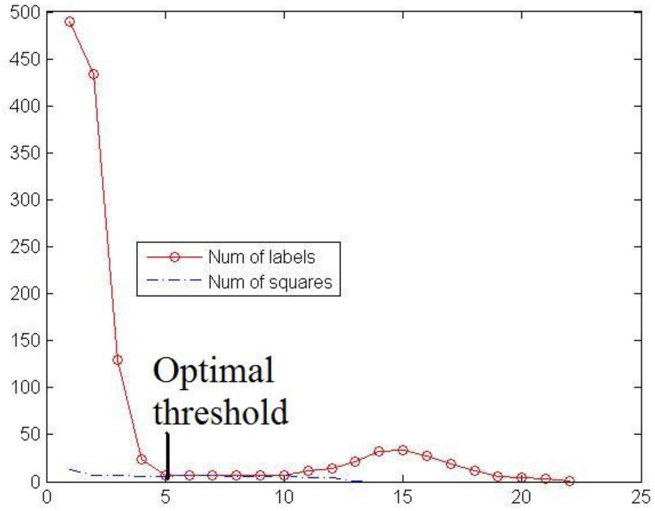


Fig. 4. **Our approach of finding optimal threshold**

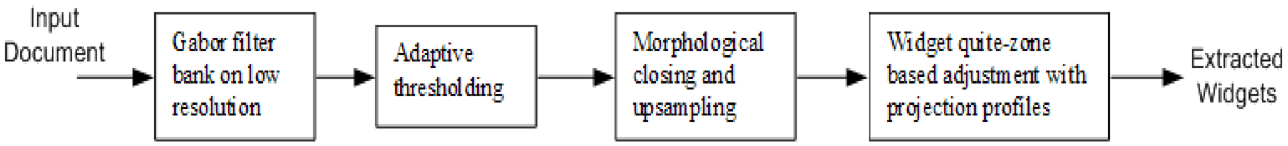

Input Document
Gabor filter bank on low resolution
Adaptive thresholding
Morphological closing and upsampling
Widget quite-zone based adjustment with projection profiles
Extracted Widgets

Fig. 5. **Proposed block diagram for widget extractor**

channel," in *15th IEEE International Conference on Image Processing, 2008. ICIP 2008*, 2008, pp. 2212–2215.

[4] Srinivasan Ramani, Anil Kumar Pai, Darpan Goel, Srinivasu Godavari, and Kuchibhotla Anjaneyulu, "Document Security for Enterprises and Government," in *HP Techcon*, 2007.

[5] Yogesh Sankarasubramaniam and Badri R. Narayanan, "Application of Error Correcting Codes for Modification Identification in Paper Documents," *HP Labs Tech. Report*, HPL-2008-98, 2008.

[6] S.J. Simske, M. Sturgill, and J.S. Aronoff, "Comparison of image-based functional monitoring through resampling and compression," *HP Labs Tech. Report*, HPL-2009-145, 2009.

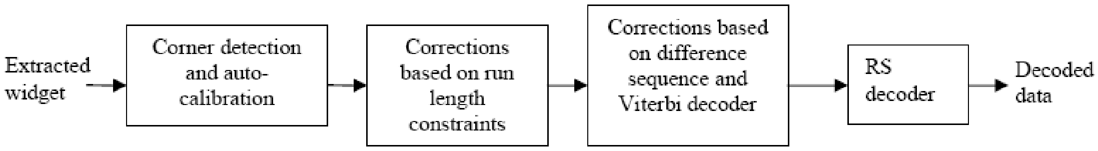

Extracted widget
Corner detection and auto-calibration
Corrections based on run length constraints
Corrections based on difference sequence and Viterbi decoder
RS decoder
Decoded data

Fig. 6. **Block diagram of widget decoder.**

[7] SJ Simske, JR Hattersley, G. Golodetz, and J. Stasiak, "Ink-Specific Handheld Readers and Security Variable Data Printing (SVDP)," in *Proc. International Conference on Digital Printing Technologies*, 2006, vol. 22, p. 441.

[8] Adobe Inc, ," `http://www.adobe.com/government/forms.html`.

[9] ," ISO/IEC 15438:2001, Information technology – Automatic identification and data capture techniques – Bar code symbology specifications – PDF417.

[10] ," ISO/IEC 16022:2006, Information technology – Automatic identification and data capture techniques – Data Matrix bar code symbology specification.

[11] ," ISO/IEC 16023:2000, Information technology – International symbology specification – MaxiCode.

[12] ," `http://www.xerox.com/Static_HTML/xsis/dataglph.htm`.

[13] ," ISO/IEC 18004:2006, Information technology – Automatic identification and data capture techniques – QR Code 2005 bar code symbology specification.

[14] S. Ono, K. Morinaga, and S. Nakayama, "Barcode design by evolutionary computation," in *Artificial Life and Robotics*, 2009, vol. 13, pp. 238–241.

[15] ," `http://2d-code.co.uk/bbc-logo-in-qr-code/`.

[16] H. Kato and M. Billinghurst, "Marker tracking and HMD calibration for a video-based augmentedreality conferencing system," in *2nd IEEE and ACM International Workshop on Augmented Reality, 1999.(IWAR'99) Proceedings*, 1999, pp. 85–94.

[17] M. Fiala, "ARTag, a fiducial marker system using digital techniques," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, 2005, vol. 2.

[18] E. Costanza and J. Robinson, "A region adjacency tree approach to the detection and design of fiducials," *Vision, Video and Graphics (VVG*, pp. 63–70, 2003.

[19] Enrico Costanza and Jeffrey Huang, "Designable visual markers," in *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, 2009, pp. 1879–1888.

[20] N. Degara-Quintela and F. Perez-Gonzalez, "Visible encryption: Using paper as a secure channel," in *Proceedings of SPIE*, 2003, vol. 5020, p. 413.

[21] R. Villan, S. Voloshynovskiy, O. Koval, and T. Pun, "Multilevel 2 D bar codes: toward high-capacity storage modules for multimedia security and management," in *Proc. SPIE*. Citeseer, 2005, vol. 5681, pp. 453–464.

[22] A. Malvido, F. Perez-Gonzalez, and A. Cousino, "A Novel Model for the Print-and-Capture Channel in 2D Bar Codes," *Lecture Notes in Computer Science*, vol. 4105, pp. 627, 2006.

[23] J.G. Proakis and M. Salehi, *Digital communications*, McGraw-Hill New York, 1995.