International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:6, 2007

2D Bar Codes Reading: Solutions for Camera Phones

Hao Wang, and Yanming Zou

***Abstract*—**Two-dimensional (2D) bar codes were designed to carry significantly more data with higher information density and robustness than its 1D counterpart. Thanks to the popular combination of cameras and mobile phones, it will naturally bring great commercial value to use the camera phone for 2D bar code reading. This paper addresses the problem of specific 2D bar code design for mobile phones and introduces a low-level encoding method of matrix codes. At the same time, we propose an efficient scheme for 2D bar codes decoding, of which the effort is put on solutions of the difficulties introduced by low image quality that is very common in bar code images taken by a phone camera.

***Keywords*—**2D bar code reading, camera phone, low-level encoding, mixed model

I. Introduction

BAR codes have been widely used for dozens of years, which performs the role of an index key for accessing a database. However, the traditional one-dimensional bar codes have an apparent shortness in terms of information density [1][13]. The vertical dimension does not carry any information but only provides a redundancy that is especially convenient for decoding by handset laser scanner when the user is not careful about the orientation and registration bounds. Nowadays more and more applications require a much longer bar code to encoding larger amount of information tips such as the price, product name, manufacturer, functionality, and expiration date of a product. Therefore the 2D bar codes were designed to carry significantly more data than its 1D counterpart.

An area scanner, such as a charge coupled device (CCD) scanner, is generally used in industries to scan a 2D bar code. The emergence of camera phones may change the current status: it will bring mobility to the traditional bar code readers so as to build a ubiquitous information and computing platform based on which many compelling applications will be driven.

More than thirty 2D bar codes have been invented or redesigned since the first true 2D bar code, Code 49, was introduced nearly twenty years ago[1] and a certain amount of research on 2D codes has been conducted. Combining with camera phones, however, questions concerning the performance of decoding issues come out: are the existing 2D bar codes suitable for scanning by the mobile phones? If yes, which one is the best choice? If no, is it necessary to design a new 2D bar code tailored for the mobile phones or even establish a standard of that?

Research has been put on the criteria of 2D bar codes specific for mobile phones [2]. Although standardization will involve a lot of aspects ranging from technologies to business modes and can not be established in one night, a basic fact is that some special considerations, at least for decoding process, should be focused on because of the restrictions of mobile phone optics. Resolution limit, distortion, out of focus blurring, and noise with illumination variety induced by the phone camera are the killers of direct use of most existing 2D bar code for mobile phones.

This paper tries to propose a new design of 2D bar code suitable for mobile phones, as well as the discussion of the principle of encoding and decoding scheme. The rest of the paper is arranged as following. Section II introduces the requirements and possible choices of 2D bar codes for mobile phones, and our design concept and solution. Although many researchers have proposed approaches dealing with image processing and recognition for both of 1D and 2D bar codes decoding, efforts on special focus of phone cameras are still important. Section III describes our decoding method specific for the mobile phones with low quality cameras, especially from the image processing and pattern recognition perspective. Experimental results are given in section IV, and section V draws the conclusion for future considerations.

II. 2D bar codes Design

Comparing with traditional usage of laser or CCD scanners, the images of 2D codes taken by a phone camera are relatively poor, say, the resolution of the images is limited, the quality of the images is low since the 2D code in the images is blurred and deformed caused by the too short distance photo-capture and perspective. Thus the design of code finder, the border, the size indication as well as the data encoding should be robust for low quality images.

A. Requirements and possible choices

H. Kato and her colleagues have put their efforts on the

Manuscript received April 29, 2006. This work was supported by SW & Application Technologies laboratory and Multimedia Technologies laboratory, Nokia Research Center.

Hao Wang is with the Multimedia Technologies laboratory of Nokia Research Center, No. 11 He Ping Li Dong Jie, Beijing, PRC (phone: +86-10-65392828; fax: +86-10-84210595; e-mail: hao.ui.wang@nokia.com).

Yanming Zou is with the Multimedia Technologies laboratory of Nokia Research Center, No. 11 He Ping Li Dong Jie, Beijing, PRC (e-mail: yanming.zou@nokia.com).

criteria for a 2D bar code used for a camera equipped mobile phone and identified factors that require certain attributes of the 2D bar code [2]: 1. Use of mobile phone for image processing, 2. Nature of objects, 3. Applications that are commonly offered for end users with a camera phone, 4. Multi-language encoding capability. From the technical point of view, however, the requirements should be reviewed in a hierarchical perspective: low-level encoding deals with fundamental features such as bar code shape, orientation, and module characteristics that can directly affect image processing and low-level decoding, i.e., to identify each module with black and white (or even with grayscale or color if such information has been used); whereas those features related to applications, error correction, language, and additional aspects should be carried out by the high-level encoding. Therefore the design or selection of the 2D code for camera phones should focus on image processing requirements that are the main issues and special problems introduced by the phone camera whose imaging quality might not always be satisfactory for decoding of a high density 2D bar code.

By examination of mobile phone cameras and the normal conditions of using a camera phone to capture a bar code image by the end users, the minimum requirements of the 2D bar codes for camera phones are identified:

- Stacked codes such as Code 49 and PDF417 are not suitable for image processing with cameras, so matrix codes are preferred.
- Visual code size grows proportionally to the data. This means that there should be no sudden “jumps” in code size; instead the size should grow linearly with the data accretion.
- Code should be as easy to detect and read independently of the size and data amount in the code.
- Since the mobility inherent in mobile phones should be considered, thus 360 degrees read flexibility is desired, which means the code should be easily read under any angle.

In addition to the preliminary requirements, further consideration is deserved to be drawn before the design of 2D bar code for camera phones [4]. Although a visual code can be of any shape – square, rectangle, circle, triangle, and whatever, it is easy to be recognized that pixel efficiency will restrict the choices of code shapes. Example of non-efficient use is ”Data Glyphs”, which needs terrible amounts of resolution to represent a single bit, although data glyphs can be hidden into pictures quite well, and be scanned with a high resolution scanner [3]. The problem is that data glyphs use diagonal stripes to represent bits (as shown in Fig. 1a) – and we need to read these through bitmap data. Thus square and rectangle are preferable rather than circle and triangle or any other non-regular shapes. There are also different alternatives related to using colors in visual codes: colored codes (as shown in Fig. 1b), monochrome code, and grayscale codes. After considering different media types that are used for keeping the code, colors and grayscale are not suggested to be used so that it can give support for wider media range that can bear the visual code and allows faxing and photocopying visual codes without losing readability. In addition, image capture by the phone camera in a variety of lighting conditions may not retain enough discriminate information of color and grayscale. In order to deal with camera distortion, a visual code might include some elements as compensation that would help to overcome known weaknesses for camera modules built-in mobile phones (e.g. sharp center and blurred edges). However, it would require different compensation for different camera modules, which seems not feasible for real use cases. Finally, scalability is an optional requirement that could support the future use of more powerful cameras by applying multi-layer data encoding and multi-resolution module alignment (as shown in Fig. 1c).

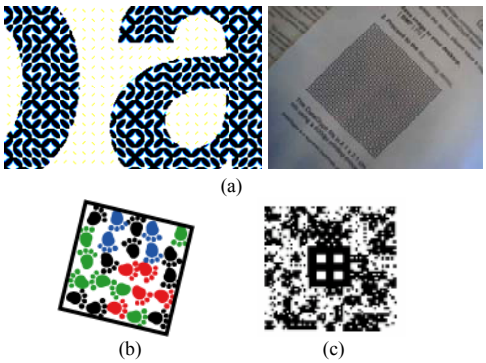


Fig. 1 example pictures of (a) Data Glyphs, (b) colored code, (c) layering data technique (using bigger cells and sub cells)

B. Code Design: Size Indication and Module Placement

In this paper, a low-level encoding scheme for camera readable visual code is proposed that is proved to be robust for application environment using existing camera phone models (e.g., NOKIA S60 camera phones, such as N6600). Here we make our efforts on the simple new size indication method, which can omit code finder patterns and support linear growth of code size.

The basic shape of a code is a square and each module is also a square, for pixel efficiency and simplicity, as mentioned above. The whole code is made of an array of modules (small squares). The number of modules in one column of the square is defined as the size of the code, noted as n in following specification.

The suggested size of a module is not less than 0.75mm, which is taken granted as the trade-off point between information density and decoding robustness. It is obvious that with the emergence of auto-focus phone cameras, the information density could be dramatically increased by the decrease of module size, however, the worst case should be considered for generality. The most outside around modules with black color are used to indicate the position of a code. It is ensured that the three of the four corners and more than 1 third of the modules in the borders are black. It can provide accurate enough position of a code with our decoding method.

We count the columns in a code from left to right, starting from 0 and ending with n-1. The rows are also counted from

top to bottom, from 0 to *n*-1. So for a module located at the *x* column and *y* row, it can be referenced as module (*x*,*y*).

For a code with orientation indication, the placement is as follows:

1. The 3 most right cells of the three rows 0, 1, 2 are used to indicate the orientation of the code. They are set as:

0	1	1
0	1	1
0	0	0

And the other three corners of the square, is 0 (where 0 stands for 'black') constantly.

2. The most left column 0 is used to record the size of the square. The bit stream '010010…' is written to the boarder from cell (0,1) to cell (0, n-2). The positions of black to white and white to black edges can be got from the image, thus the size of one module can be estimated and then finally the size of the code can be calculated.

*3. The first (n-2)×2-3 bits are record in the other 3 boarders of the square. One '0' bit is inserted before each 2 continuous bits, thus totally n-3 bits (0) are inserted and a stream of (n-2)×3-4 bits is gotten like '0**0**….'. The bit stream is written to the 3 boarders in the order as:*

The top boarder from the left cell (1,0) to the right (n-4,0);

The right boarder from the top cell (3,n-1) to bottom (n-2,n-1);

The bottom boarder from the left cell (1,n-1) to the right (n-2,n-1).

4. Other bit streams are written to the modules in the columns from 1 to n-4 from left to right. In each column it is written from top cell (,1) to bottom cell (*,n-2).*

5. For the column (n-3) and (n-2) the writing is from row 3 to row (n-2).

The situation can be divided into three kinds, depending on the remainder of division between the square size *n* and 3. An Example is given out in Fig. 2.

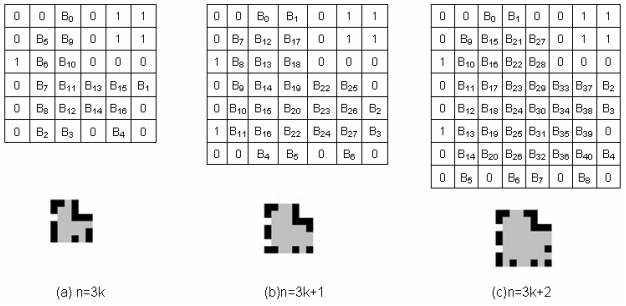


Fig. 2 module placement with orientation indication. For all example *k*=2. Gray color means that module can take either 1 or 0 (white or black). The symbol B_j represents the *j*th bit in the original bit stream

C. High-level Encoding

Any higher-level encoding scheme such as codeword encryption can be performed based on the basic module placement indication (the current placement is just bit by bit). Error detection and correction can also be included. Pattern randomizing will be helpful on the top of the proposed module placement method: it can avoid the occurrence that modules with same value (black or white) cluster together, which is very harmful to the decoding stage.

III. Decoding

In this section we present our image processing and analysis scheme that is tailored for decoding of the proposed low-level encoding method. Although the decoding scheme is specific to handle our own code with low quality images taken by phone cameras, it is not difficult to extend the scope for dealing with other rectangle-shape 2D codes with any reasonable levels of image quality.

The framework of proposed approach is illustrated in Fig. 3. At first a 2D code is located by corner detection and then segmented from the background. Code size is detected based on the periodic edge information of the first column. Then the code image is scanned using Bi-directional centripetal run-length (BCRL) method, where a mixed model is applied to perform decoding bit by bit. In order to enhance the tolerance of high order distortion, grid vibration method is introduced to compensate the effect of non-linear distortion. An efficient deblur approach is also introduced before decoding, whose purpose is to eliminate the blurring caused by short-distance imaging with phone cameras as much as possible.

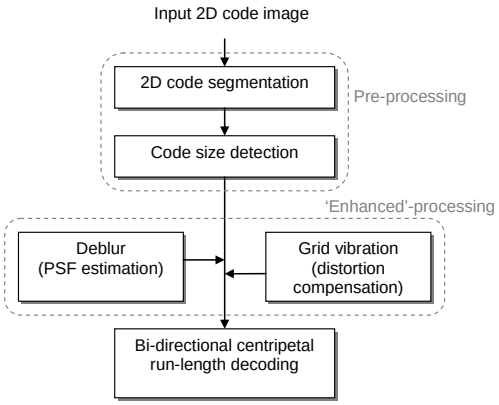


Fig. 3 framework of decoding method

A. Segmentation and Size Detection

Normally the bar code region contains special texture information other than its background, there had been many approaches using texture analysis to locate the bar code area from an image [5][6][12][14][16][17]. Considering the use case that the bar code region always occupies a large area in the bar code image taken by a camera phone and the background is not complex, in this paper, we simply use Hough transform to locate the four corners of the bar code region. Because our size indication method is simple but efficient, it is easy to compute the size of the bar code based on the periodic edge information (the first column of the code contains the size indication) after it has been located.

B. Bi-directional CentripetalRun-length (BCRL)

Previous work on bar code reading tried a series of approaches such as 2nd derivatives [6], peak locations [7][8], selective sampling [9] and EM algorithm [10][11]. However, these approaches are mainly applied to decode 1D bar codes or stacked 2D bar codes that do not really use 2D layout of the

code. Most of these approaches require high resolution of the scanned images, or sharp images with small point spread function (less than 5 pixels), neither of which is easy to be satisfied if the bar codes are taken by a normal phone camera. E. Ouaviani et al. directly compute the bit pattern of a matrix 2D bar code after binarizing the image [5], but a common binarization method is fragile if the image is blurred or distorted.

In this paper, we propose a heuristic method, Bi-directional Centripetal Run-Length (BCRL), whose basic idea is to use both horizontal and vertical waveforms as complementary cues in order to obtain credible bit patterns from a matrix code, i.e., the proposed code in the above section.

After code segmentation and size detection, we have obtained the coordinates of the four corners of the code and the side length n (supposing the code is square). Due to uncontrolled factors of photo capture, the square code in the image may deform as an irregular quadrilateral in an arbitrary orientation rather than an upright square or rectangle. Then affine-transform could be used to compensate the deformation including scaling and rotation, however, it is time-consuming especially for mobile devices and may not be benefit if there are high-order nonlinear deformations with the code area (may caused by the lens). In this paper, we just keep the circumscribed rectangle of the code quadrilateral and scale it to a normalized size (so that each module of the code will consist of about 10×10 pixels) with bi-linear interpolation (supposing the slant of the code is not more than 5º thus rotation is not needed), as shown in Fig. 4. The BCRL approach can be directly applied on the scaled rectangle region. In order to reduce the effects of global intensity changes, we use the 2nd derivative image instead of the original gray-level image.

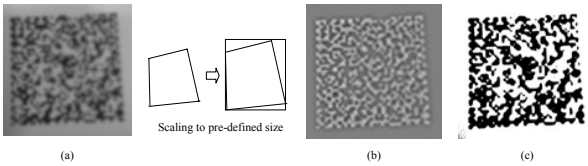


Fig. 4 pre-processing of the code region with only scaling: (a) original code image taken by a camera phone, (b) 2nd derivative image (c) typical binarization that is not capable for decoding.

BCRL: It is found that the modules placed close to the sides of the code are comparatively easier to be identified than their central peers, by human sight observing the 2D code image. The reason may be that there is less contamination in the area of the sides because of the quiet zone around the code. This observation is helpful for our decoding process: we can assign higher confidence to the straps close to the sides and use the decoding result of those rows and columns for predicting the decoding parameters of the central peers. Fig. 5 depicts our scan method. The code region is divided into four quarters. In each quarter, scans are performed from outer to inner along columns (Quarter I and III) or rows (Quarter II and IV) and previous scans are used to predict the parameters of the successive ones.

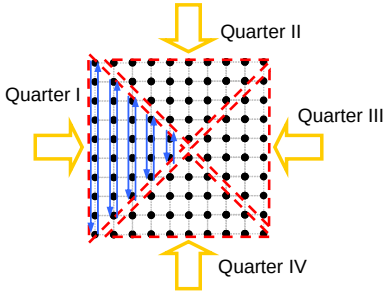


Fig. 5 BCRL scan method

For the j^{th} column of Quarter I (implementation of other quarters are similar):

1. *Set j to 0.*
2. *Sample the 1D waveform along the j^{th} column from the 2nd derivative image by a weighted average method.*
3. *Compute the run-length based on a mixed model that use both of the peak location and zero crossings of the 2nd derivative (ZC2) from the top to the bottom, marking the result as $\{RL_i^{down}, 0 \le i < m^{down}\}$, where m^{down} denotes the number of runs obtained by this scan, and the run-length RL_i^{down} is the number of successive white or black modules (not the number of pixels) in the i^{th} run.*
4. *Compute the run-length from the opposite direction (bottom to top), marking the result as $\{RL_i^{up} 0 \le i < m^{up}\}$.*
5. *Since the scans from two different directions are not symmetrical due to the mixed model, the run-length arrays may not be same. If $m^{dwon} = m^{up} \equiv m$ and $RL_i^{down} = RL_{n-i}^{up}$ for all $0 \le i < m$, or $N_{\max}$ iterations have been applied at the scan of the current column, go to 6. Otherwise tune the parameters of the mixed model and return to 3.*
6. *Predict the mixed model parameters of the next column (j+1) based on the current scanning, set j=j+1, return to 2.*

After scans of all the four quarters are finished, the bit-by-bit decoding result is naturally obtained, noting that if the bi-directional run-lengths happen to be unequal for a column or row when the iterations complete in step 5 (which indicates that error accumulations from both directions are not symmetric), arbitrary selection of the alternatives is made.

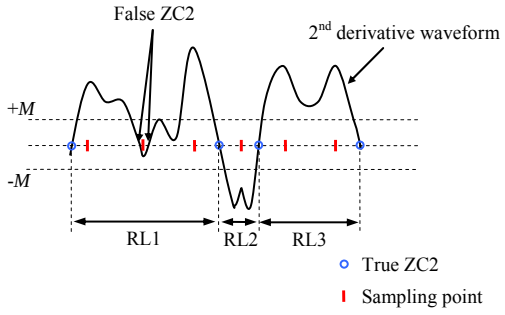


Fig. 6 the mixed model used in BCRL

Fig. 6 depicts the mixed model used in BCRL. The model consists of following parameters:

- A threshold of $\pm M$ is used to define a band across the waveform of 2nd derivative that contains all the noise peaks. The value of M is determined by the average local amplitude of the waveform multiplied by a coefficient *k*, where *k* is set as 0.1-0.2 in our experiments. Thus M can have different values for different waveforms or even for different segments of waveforms if there are obvious gradient along one scan line.

- ZC2 points that imply the alternatives of black and white segments. The ZC2s of which the waveform segments are contained within the $\pm M$ band are discarded.

- Peak locations that give hints of proper sampling points.

- Sampling points that are located based on the peak locations and the estimated local sampling intervals.

In each scan along one direction (e.g., from left to right in Fig. 6), sampling interval between two neighboring sampling points is first initialized as the average module size, and the first sampling point is located at the center of the first module (or predicted by the previous scan). Then the next sampling position is calculated by adding the sampling interval to the position of the current point. If the sampling position falls into a small adjacency of a salient peak location, replace the sampling position by the peak location, and adjust the previous sampling positions within the same segment between two ZC2s, then update the local sampling interval. The restriction of the local sampling intervals is that any two adjacent intervals should be close and any intervals should not be too larger or too smaller than the average module size. After all the sampling points are located, run lengths are easily calculated with how many sampling points fall into one segment between two ZC2s. Because the order of the peak locations affects the scanning result, run lengths calculated from opposite directions might be different. Thus the parameters of the mixed model, esp., the judgment of the peak locations will be adjusted and iterations perform as described above.

C. Grid Vibration

The BCRL approach achieves satisfactory decoding result under geometric distortion and blurring, however, it is not good enough to deal with the highly non-linear distortion such as barrel or pincushion distortion caused by the lens of the phone camera. Since the form of the distortion is not easy to predict beforehand due to the uncontrolled behavior of the users while taking pictures of the 2D codes, it is not feasible to address a specific correction of the distortion.

Therefore we propose a model whose aim is to adjust the sampling positions of the modules by observing the circumstance around each sampling point. The model consists of a grid of sampling points that each point can vibrate around its reference position (the vibration parameter is estimated by a Bayesian decision method where in the training stage we have obtained the prior distribution of the local 2nd derivative vectors under the conditions of all the possible local vibrations), as if the grid is made of an elastic material under a global constraint, as shown in Fig. 7 (quasi-square shape).

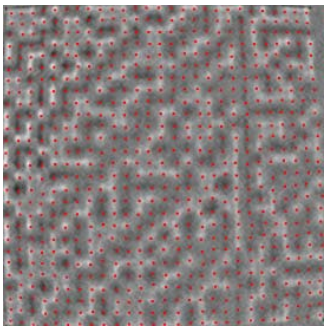


Fig. 7 vibration of the sampling grid (in the image of 2nd derivative)

D. Deformable Model

Blurring is a serious problem that is harmful for the decoding of the 2D codes. The mixed model used in BCRL approach can reduce the effect of blurring in some senses, but it is not a systematic solution. In theory, we want to estimate the blurring parameters and use de-blur methods to eliminate blurring. N. Normand et al., used a cosine shape function as the blurring model to calculate a new set of blurred characters that is used as new reference for character matching [6]. But their model parameter, i.e., the base of the blurring function, is not explicitly estimated by the input signals, which means the model is somewhat fixed but not adaptive to new conditions. E. Joseph et al. gave a practical appoach to estimate the parameter of blurring in [7][8], where they assumed that the point spread function (PSF) has a symmetric bell-shape such as a Gaussian kernel. One question comes with the PSF, does it always be a Gaussian? Actually, we noticed that the form of the PSF is often related with the specific lens and sensors.

In this paper, we applied a PSF estimation method that aims to quantify the blur distortion due to camera being at close distance to small target images [18]. This situation is quite common in bar code images, where the camera is placed at a distance of less than 15 cm in order to maximize the resolution, whereas focus range of the lens is over 30cm. The idea is to obtain "realistic" ground-truth about the blur due to misfocus for later use in restoration.

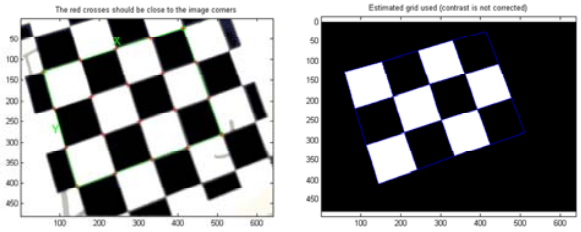


Fig. 8 corner detection on checkerboard pattern and reconstruction of the blur-free image

The PSF estimation method consists of three steps. In the first step, we apply corner detection on the blurred checkerboard images (captured under 15cm). Then we

calculate the boundary lines between the detected corner locations, and reconstruct the blur-free image, as shown in Fig. 8. Now that we have the blurred and 'original image', we apply our PSF estimation technique so called Pseudo-Inverse filtering (Fourrier domain) and Gaussian blur fitting. This process is carried out for all the checkerboard images, the resulting averaged PSF is used as the estimated blurring model, shown in Fig. 9.

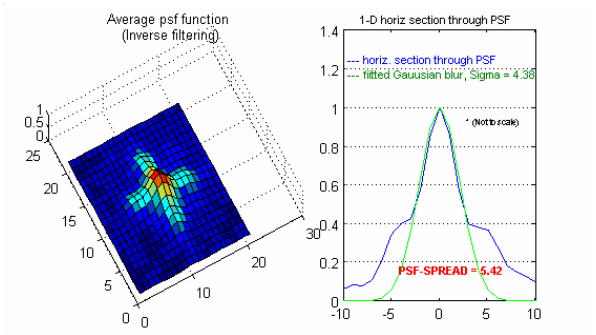


Fig. 9 average estimated PSF

It was found that the PSF spread is much larger than that of in-focus conditions, and the function is not an ideal Guassian shape. It is also worth noting that the PSF is not circular, the blurring is more accentuated on the horizontal and vertical axis. This can be explained with the geometry of the sensors which includes some non-sensitive areas in between pixels.

Thus the estimated PSF is applied for de-blur of the 2D code images before decoding is performed. For computation efficiency, we apply the Maximum Likelihood (ML) technique to restore the image, although not optimal because of not taking into account a best prior term, experiments have shown the effectiveness of the deformable model.

IV. Experimental Result

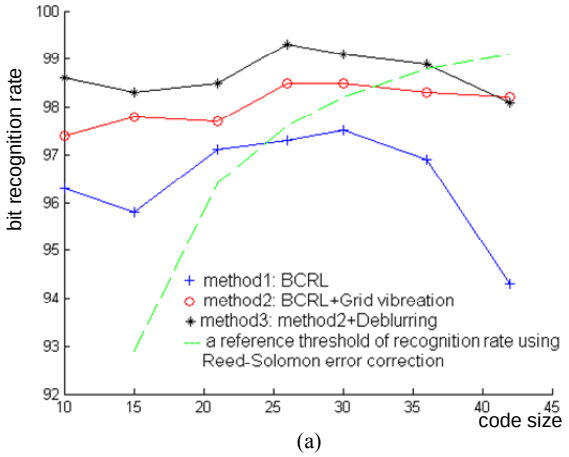
For experimental purpose, we simply apply an intuitive encoding method of which ASCII code is used directly. A whole byte is the smallest unit to measure the information length. A code with size n can totally record $n\times(n\text{-}2)/8$ bytes. No information about the length of the original data is recorded in the codes, instead, the encoding method uses the smallest code to record those bytes automatically. Reed-Solomon code with 8-bit symbol is used for error correction. 16 parity symbols will be added to the original data. Thus 8 errors can be corrected. The length of the original information which can be contained by a code is then $n\times(n\text{-}2)/8$ -16 bytes.

There are 146 sample images of the proposed 2D codes with a series of code size (ranging from 10 to 42) and module size (ranging from 0.75mm to 1.5mm) used in our experiment. Fig. 10 depicts the bit recognition rate of different code size and module size respectively, which is proved to be robust for existing camera phone models (e.g., NOKIA N6600). The bit recognition rate r is calculated as

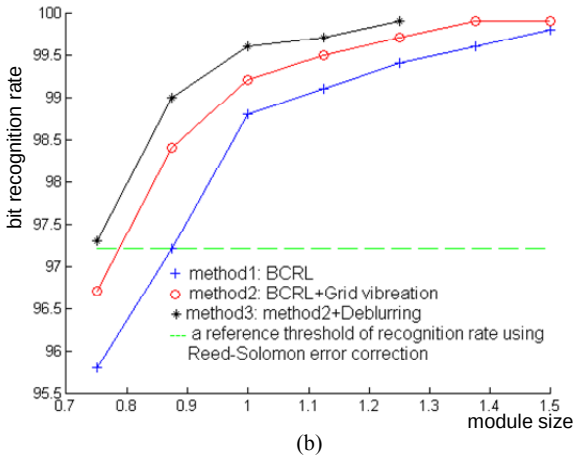
$$r = (1 - n_{err} / n_{total}) \times 100\% \quad (1)$$

Where n_{err} stands for the number of error bits accumulated for the whole test set, and n_{total} represents the total number of bits contained in the test set.

The dashed line in the figure gives a reference threshold that the bit recognition rate should surpass for correct decoding when the above error correction is used. From the figure we can see that the selection of the code size for current phone camera models should be within a suitable range: large size code contains more information but has more difficulties to be decoded because of the higher distrotion; small size code does not always mean easy decoding since the user wants to take more bigger picture of the code so that makes the image capture more closer, thus the image will be more blurred. Selection of module size could be a tradeoff between information density and decoding robustness. Currently the suggested module size is beyond 0.75mm for the cameras with more than 30cm focus distance. Using micro-lens or optical focus camera, it is obvious that the limitation of the module size can be dramatically enhanced.



(a)



(b)

Fig. 10 bit recognition rate: (a) code size ranging from 10 to 42 with fixed module size 0.875mm. It is noticed that the optimal code size range is under 35. (b) module size ranging from 0.75mm to 1.5mm with fixed code size 24. considering the trade-off between decoding robustness and information density, the module size is suggested to be set between 0.75mm to 1.0mm

V. Conclusion

In this paper we present a low-level encoding method of 2D bar codes that is tailored for camera phones. With the proposed decoding approach, we are able to build a practical framework of camera readable 2D bar codes codec running on mobile phones and some applications have been developed based on it. Future work will concern about improving the performance of the decoding algorithm that should be capable to deal with more complex distortions in a variety of use cases. One possible solution is to use statistical methods to carry out image patch (bit patterns) recognition as we have already implemented for the 1D bar code recognition [12]. Thus the efforts should be focused on the training of local image patches in a statistical sense and combination of the local bit patterns together to get the final result. Another consideration is to use color information in the encoding stage that each cell would contain more than one bit, which might be more beneficial than just black and white patterns, and then develop corresponding decoding scheme. The stress could start with analysis of the color resolution of the typical phone camera modules.

Acknowledgment

We would like to thank Mr. Yrjanainen Jukka and Dr. Kangas A. Jari, who initialized the project and provided helpful materials and supports to us. And we also would like to thank Dr. Burian Adrian, Dr. Wang Kongqiao, Dr. Oleg Beletski, and Mr. Mejdi Trimeche, who shared with us useful ideas and information.

References

[1] Theo Pavlidis, *et al*., "Information encoding with two-dimensional bar codes", *IEEE COMPUTER*, v25, n6, pp. 18-28, 1992.
[2] H. Kato, *et al*., "2D barcodes for mobile phones", *IEE Mobility Conference*, 2005.
[3] http://www.bmva.ac.uk/bmvc/2002/papers/60/full_60.pdf
[4] Oleg Beletski, *et al*., "Requirements of visual code", *technical report, Nokia Research Center*, 2004.
[5] E. Ouaviani, *et al*., "A common image processing framework for 2D barcode reading", *IEE conference on IPIA*, pp. 652-655, 1999.
[6] N. Normand, *et al*., "A two-dimensional bar code reader", *ICPR*, v3, pp. 201-203, 1994.
[7] E. Joseph and T. Pavlidis, "Bar code waveform recognition using peak locations", *IEEE Trans. on PAMI*, v16, n6, pp. 630-640, 1994.
[8] E. Joseph and T. Pavlidis, "Deblurring of Bilevel Waveforms", *IEEE Trans. on Image Processing*, v2, n2, pp. 223-235, 1993.
[9] Stephen J. Shellhammer, *et al*., "Novel signal-processing techniques in barcode scanning", *IEEE Robotics & Automation Magazine*, v6, n1, pp. 57-65, 1999.
[10] W. Turin and R. A. Boie, "Minimum discrimination information bar code decoding" , *19th convention of Electrical and Electronics Engineers in Israel*, pp. 255-258, 1996.
[11] W. Turin and R. A. Boie, "Bar code recovery via EM algorithm", *IEEE Trans. on PAMI*, v46, n2, pp. 354-363, 1998.
[12] K.Q. Wang, *et al*., "Barcode reading from images captured by camera phones", *IEE Mobility Conference*, 2005.
[13] Theo Pavlidis, *et al*., "Fundamentals of bar code information theory", *IEEE COMPUTER*, v23, n4, pp. 74-86, 1990.
[14] R. Muniz, *et al*., "A robust software barcode reader using the Hough transform", *ICIIS*, pp.313-319, 1999.
[15] E. Joseph, *et al*., "Waveform recognition with application to bar codes", *ICSMC*, v1, pp. 129-134, 1991.
[16] A.K. Jain, "Bar code localization using texture analysis", *ICDAR*, pp. 41-44, 1993.
[17] S. ANDO, *et al*., "Automatic visual searching and reading of barcodes in 3-D scene", *IVEC*, pp. 49-54, 2001.
[18] Mejdi Trimeche, "Out of focus analysis & correction for document images", *technical report of Nokia Research Center*, 2003.

Hao Wang is a research engineer in Nokia Research Center, Beijing. He received his B.S. degree and M.S. degree in Electronics Engineering from Tsinghua University, Beijing, in 1998 and 2000 respectively. He is currently pursuing his PhD in Electrical and Communications Engineering Department of Helsinki University of Technology, Finland, major in Computational Engineering. His research interests include image processing and analysis, pattern recognition, and computer vision.

Zou Yanming received his B.S. degree and a PhD degree in Electronic and Communication System from Beijing Institute of Technology in 1994 and 1999 respectively. His research interests include genetic algorithms, statistical pattern recognition and image processing. Now he works in Beijing site of Nokia Research Center as a senior research engineer.