2007

Pervasive 2D Barcodes for Camera Phone Applications

Hiroko Kato
Edith Cowan University

Keng T. Tan
Edith Cowan University

10.1109/MPRV.2007.80
This is an Author's Accepted Manuscript of: Kato, H., & Tan, K. (2007). Pervasive 2D Barcodes for Camera Phone Applications. IEEE Pervasive Computing: Mobile and Ubiquitous Systems, 6(4), 76-85. Available here

Pervasive 2D Barcodes for Camera Phone Applications

As 2D barcodes gain popularity, do we need to determine a global standard for camera phone applications? If so, which barcode is best?

Hiroko Kato and Keng T. Tan
Edith Cowan University

Two inventions have contributed to the commercial viability of pervasive 2D barcodes: CyberCode[1] and J-SH09. In 2000, Sony introduced CyberCode—one of the earliest uses of a 2D barcode for developing a visual tagging system. Although CyberCode wasn't implemented on a camera phone—it was for augmented reality systems—it showed the potential for developing pervasive applications using 2D-barcode technology.

Two years later, J-Phone (now Softbank Mobile) and Sharp introduced J-SH09, the first commercial camera phone with a reader for the 1D Japanese Article Number barcode and the 2D barcode, QR (Quick Response) Code. This opened the door for ordinary users to actively interact with 1D- and 2D-barcode technologies in Japan.

However, this new mobile technology is still in its infancy and, despite its potential, hasn't yet been widely adopted outside Japan. Also, although QR Code isn't the only available 2D barcode, it has become the de facto standard for many mobile phone companies and advertising departments in Japan (see the "Mobile Applications Using QR Code" sidebar).[2] We thus wanted to evaluate QR Code against other available options and evaluate the need for a global 2D-barcode standard for camera phone applications. So, in a previous study, we evaluated six 2D barcodes using eight criteria for standardization potential: omnidirectional symbol reading, support for low-resolution cameras, reading robustness under different lighting conditions, barcode reading distance, error correction capability, security, support for multiple character sets, and data capacity.[3] We also considered the fidelity of the camera phone's captured image as a metric for gauging reading reliability. Here, we review the six 2D barcodes and then use an additional metric—a first-read rate—to quantitatively verify our earlier results and better gauge reading reliability.

2D barcodes for camera phones

Although more than 30 2D barcodes are in use, we examined only the six that camera phone applications were using at the time of our study. (Two new codes, mCode and Bee Tagg, are now in use as well.) We can divide these into two categories: *database 2D barcodes* and *index-based 2D barcodes*.

The database 2D barcodes—QR Code, VSCode, and Data Matrix—were initially invented to improve data capacity for industrial applications. However, when integrated into mobile phones with built-in cameras that can scan and decode data, these 2D barcodes can operate as portable databases, letting users access information anytime, anywhere, regardless of network connectivity.

The index-based 2D barcodes—Visual Code,

Published by the IEEE Computer Society ■ 1536-1268/07/$25.00 © 2007 IEEE

Mobile Applications Using QR Code

Japan has witnessed a remarkable increase in the number of camera phone users using Quick Response Code applications. Japan.internet.com and Express Research conducted a joint survey of 330 people (ages 20–69) in August 2006 (see http://japan.internet.com/research/20060830/1.html [in Japanese]). 82.4 percent (187 people) of the respondents who had camera phones with QR Code readers (227 of the 330) used camera phones with QR Code (up from 6 percent in 2004). This trend is especially true for young people. According to Info Plant, a 2005 survey of 7,660 people under age 20 found that nearly 90 percent of the respondents have had experience using QR Code (see www.itmedia.co.jp/survey/articles/0509/21/news091.html [in Japanese]).

Users can create QR Codes using free barcode generators. They can then exchange them over the Internet, save them on mobile phones, or print them. Some users have even developed their own applications. An example is the QR Code Blog (www.qrcodeblog.com), which is written in QR Code symbols.

The media most often used among those 187 respondents were newspaper and magazine articles (62.6 percent), and the data type most decoded by camera phones was URLs (92.0 percent), followed by email addresses (21.9 percent). The most popular QR Code application seems to be linking camera phones to the Internet, where users have "always on" access to online shopping and real-time information about public transport schedules and events. QR Code can also be used without network connectivity. For example, by simply capturing a QR Code symbol printed on a business card, users can transfer the contact information encoded in the symbol to their phone's address books.

QR Code can also act as a portable file such as an e-ticket or e-coupon. For example, Japanese airlines offer self check-in using QR Code e-tickets, and in June 2004, Japan Coca-Cola used QR Code as an e-coupon for its new product (Coca-Cola C2) gift campaign. In the first three days, 200,000 users downloaded the e-coupon, and 20,000 participants got the Coca-Cola C2 from the networked soft-drink vending machine C-mode, which interacts with camera phones via QR Codes (see http://itpro.nikkeibp.co.jp/free/NC/NEWS/20040610/145682 [in Japanese]).

Another QR Code application is the agricultural industry's recent implementation of a food-history information service aimed to improve produce quality and assure consumers of food safety. The Ibaraki Agricultural Produce Net Catalogue (see http://ibrk.jp/default.aspx [in Japanese] or www.cctv.com/english/20070110/102029_1.shtml) now includes a QR Code on its packages. By scanning the code, consumers can obtain the supplier's contact information, the food's harvest and shipment dates, and a list of the fertilizer and agricultural chemicals used.

ShotCode, and ColorCode—were developed for camera phones, so they take into account the reading limitations of these built-in cameras. They have a much lower data capacity than database 2D barcodes, but they offer more robust and reliable barcode reading. Each barcode basically works as an index that links the digital world to the real world, so these barcodes require network connectivity.

QR Code

Denso Wave developed QR Code[4] in 1994 to improve the reading speed of complex-structured 2D barcodes. QR Code's distinctive feature is its position-detection patterns, located at three of symbol's corners (see figure 1a).[2] When a reader scans a symbol, it first detects these patterns. The ratio of the black and white on a line that passes through the center of the pattern (that is, B:W:B:W:B) is 1:1:3:1:1 from any angle. This lets the reader quickly find the detection patterns, which in turn promotes ultra-high-speed barcode reading.

QR Code can encode all types of data including symbols, binary data, control codes, and multimedia data. The maximum data capacities are 7,089 characters for numeric data, 4,296 characters for alphanumeric data, 2,953 bytes for binary data, and 1,817 characters for Japanese Kanji and Kana data. The symbol versions of QR Code range from version 1 (21 × 21 modules) to version 40 (177 × 177 modules). However, mobile applications use only versions 1–10 to take into account camera phone limitations.

Several QR Code features can enhance the code's reading robustness, including error correction, a masking technique,[5] and a structured-append function.[4] By applying Reed-Solomon error correction coding, QR Code can restore the original data even if the symbol is damaged. Four different error-correction levels are available according to the user's requirement: Level L (approximately 7 percent, M (approximately 15 percent), Q (approximately 25 percent), and H (approximately 30 percent). (The percentages are the data restoration rates for total *codewords*—a unit that constructs the data area. One codeword of QR Code is equal to eight bits.)

The masking technique allocates the black and white dots evenly and helps prevent pattern duplication. This helps readers avoid code misreadings so they can quickly process the code. Eight masking patterns exist, and encoders evaluate each pattern to select the best one (see figure 1b). So, after masking, if lines and blocks still exist that are the same color, the points for the mask are reduced. The encoder selects the mask with the highest mark.

The structured-append function en-

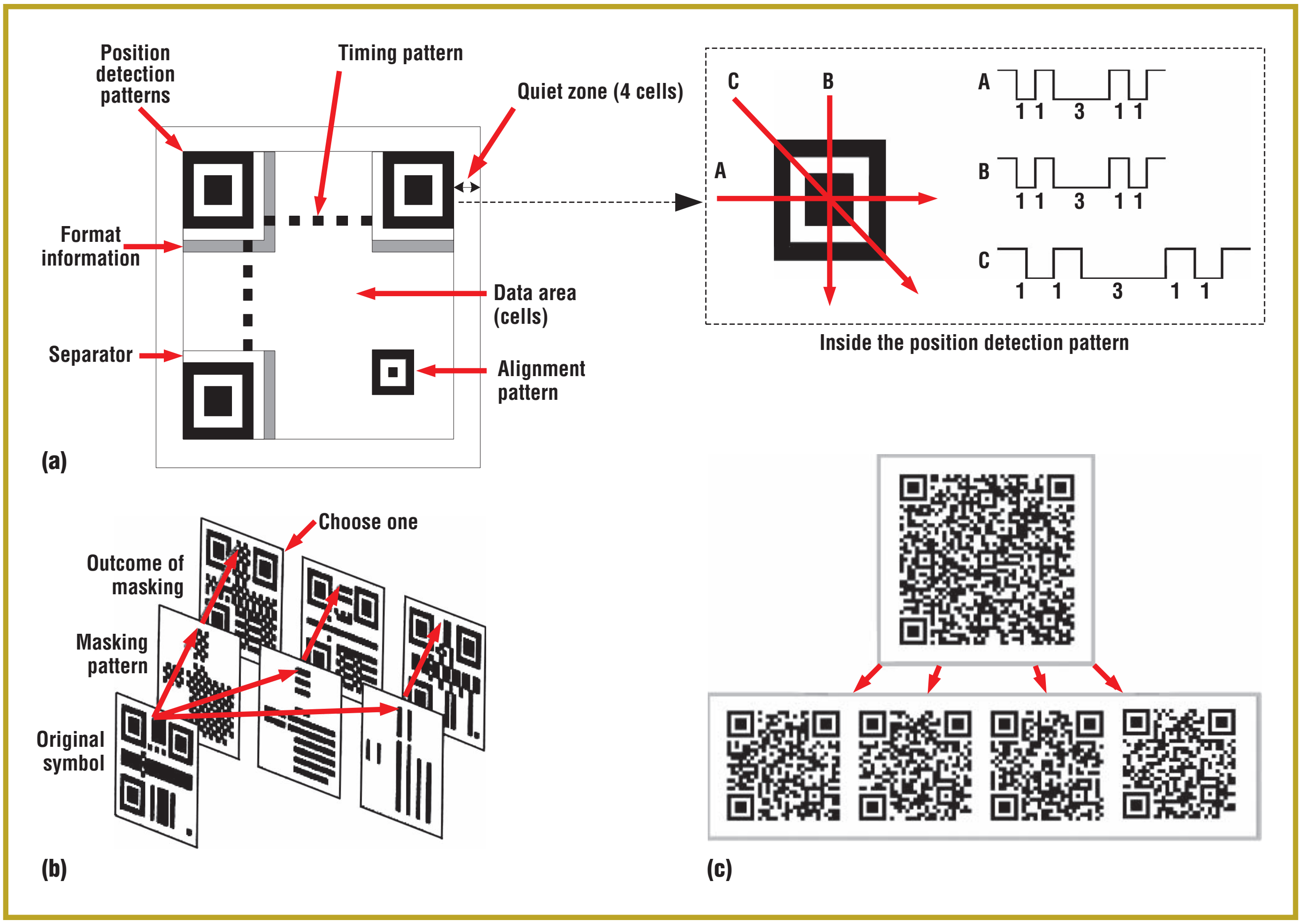


Figure 1. QR Code's (a) symbol structure,[2] (b) masking operation,[5] and (c) structured-append feature, which enhances the code's scalability by dividing it multiple data areas.[4] (images redrawn based on cited works)

hances a QR Code's scalability by dividing the code into up to 16 data areas. This means we can break a large QR Code symbol into smaller ones (see figure 1c), so we can manipulate the code's cell size and data capacity and satisfy the limitations of camera phone applications. A decoder can then reconstruct the information stored in the multiple QR Code symbols.

VSCode

Veritec's VSCode (originally VeriCode) aims to make mobile applications more secure (www.veritecinc.com/vericode.html). A distinctive feature of VSCode is its high data capacity, which lets it store biometric data for access-control applications. Veritec also offers additional security options through custom designs that use unique encryption techniques. So, you can use VSCode with other electronic media, such as an RFID tag or a smart chip, for data access and storage.

As figure 2a shows, VSCode consists of a solid border frame, a data field, and an invisible border called the "quiet zone." The VSCode symbol can store up to 4,151 bytes (the previous VeriCode symbol could only store 500 bytes). VSCode has the largest data capacity of the barcodes presented here, yet it doesn't require a large printing surface because it can adjust its size and shape (square or rectangle) to fit the available space. Furthermore, each cell size is quite small.

Using Reed-Solomon, VSCode has a high percentage (15 to 25 percent) of error detection and correction (EDAC) capability. This lets it restore all the encoded data even if up to 35 percent of the symbol is damaged.

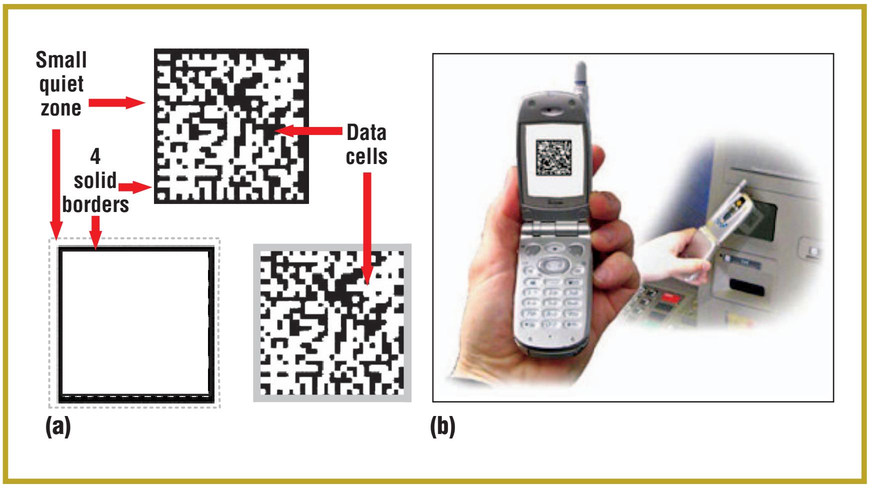
In figure 2b, a mobile phone displaying a VSCode symbol works as an electronic ID card.

Data Matrix

Data Matrix[6] uses two types of error-correction algorithms, depending on the error checking and correcting level employed. ECC levels 000 to 140 offer five different error-correction levels and use convolutional code-error correction. However, ECC-200, which is in common use, uses Reed-Solomon error correction. The symbol size determines ECC-200's correction level. For this article, we focus on the Data Matrix ECC-200.

Data Matrix has a finder pattern comprising two solid lines and two alternat-

Figure 2. VSCode: (a) symbol structure and (b) an electronic ID card example.



ing dark and light lines on the symbol's perimeter, which is surrounded by a quiet-zone border (see figure 3a). The L-shaped solid border defines the physical size, orientation, and symbol distortion, and the broken border on the opposite corner defines the symbol's cell structure.

Data Matrix ECC-200 symbols have an even number of modules on each side. Although the maximum size is 144 rows × 144 columns, each block size is limited to 24 rows × 24 columns to prevent symbol distortions. When the data exceeds the block size, the symbol is divided into four blocks. The maximum data capacity is 3,116 digits, 2,335 alphanumeric characters, or 1,556 bytes. The Data Matrix symbol can be square or rectangle.

Several features enhance Data Matrix's reading robustness. First is its size. In a comparison with QR Code and VeriCode in which eight digits were encoded in a .25 mm cell, it created the smallest symbol (3.3 × 3.3 mm) and maintained the same level (10–15 percent) of error correction (see www.ainix.co.jp/barcode/c_04_02.shtml [in Japanese]).

Second, a structured-append function can divide a Data Matrix symbol into 16 multiple symbols. The original data can be restored from these, regardless of scanning order, by adding three bytes to each of the smaller symbols. The first additional byte identifies the position for the particular symbol and the total number of symbols in the sequence. The remaining two bytes serve as a file (a "file" is the linked Data Matrices) identifier with a possible value (or ID number) between 1 and 254 (thus allowing up to 254 × 254 = 64,516 identifiers). (See http://idautomation.com/datamatrixfaq.html for more information.)

Third, data compaction can compress data before encoding it in a Data Matrix symbol. This increases the barcode's data capacity.

Semacode (http://semacode.org/about/technical) has adopted the Data Matrix format to encode plain-text URLs and implement them as physical hyperlinks. For example, in figure 3b, a phone scans a business card to display the company's URL.

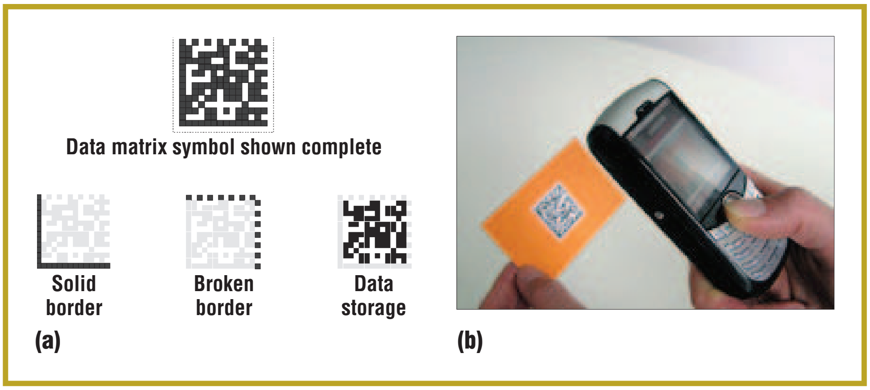


Figure 3. Data Matrix: (a) the symbol structure and (b) a symbol displayed on a business card.

Visual Code

Visual Code[7] was developed to enable HCI using camera phones. It has a code coordinate system that maps arbitrary points in an image plane to corresponding points in the code plane. The movement-detection algorithm lets the mobile phone function as an optical mouse. Also, camera phones, because they're mobile, sometimes randomly rotate or tilt symbols when scanning them. Visual Code can use the amount of rotation and tilt in a captured image as additional input parameters. So, users can obtain different information by rotating the phone, and the users can continuously update the information by moving the phone. This enables applications that support real-time interactions between the camera phone and nearby active displays (see figure 4).

Although the data capacity is limited to 83 bits, Visual Code can function both as a portable database and as an index to a remote database. For EDAC, Visual Code employs a (83, 76, 3) linear block (that is, it generates an 83-bit code word from a 76-bit value and has a Hamming distance of 3).

ShotCode

ShotCode's distinguishing feature is its aesthetic round shape. High Energy Magic developed the symbol, originally called SpotCode, to enhance HCI.[8] SpotCode is also a derivative of another cir-

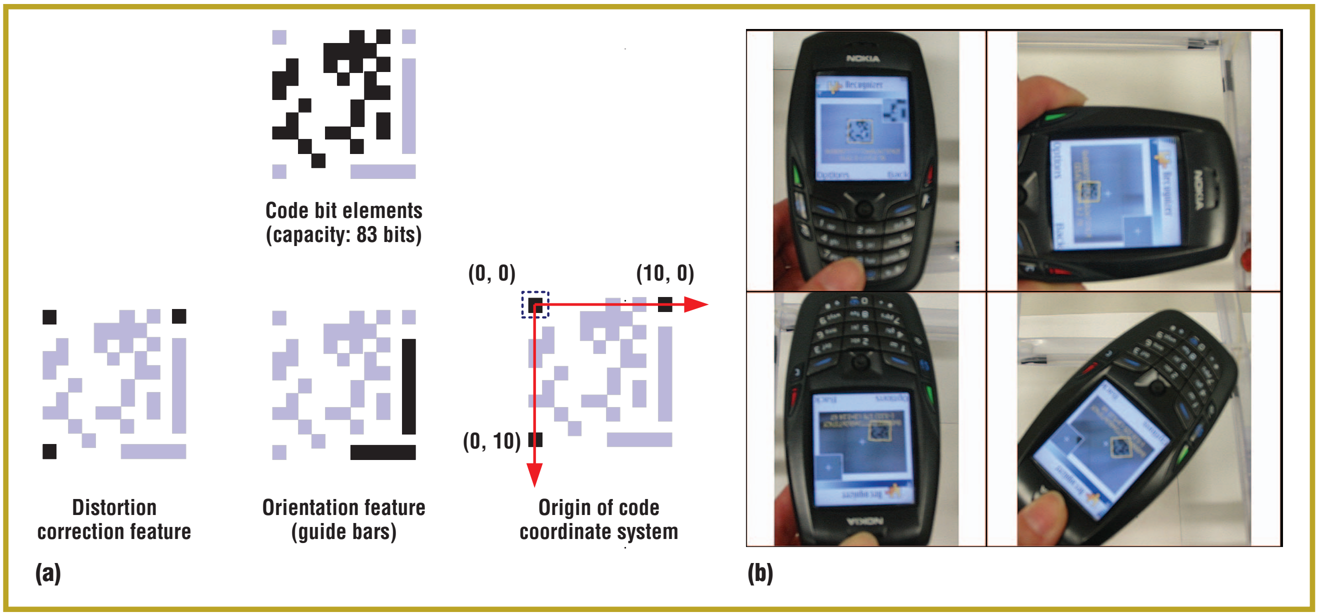


Figure 4. Visual Code: (a) symbol structure and (b) tracking.

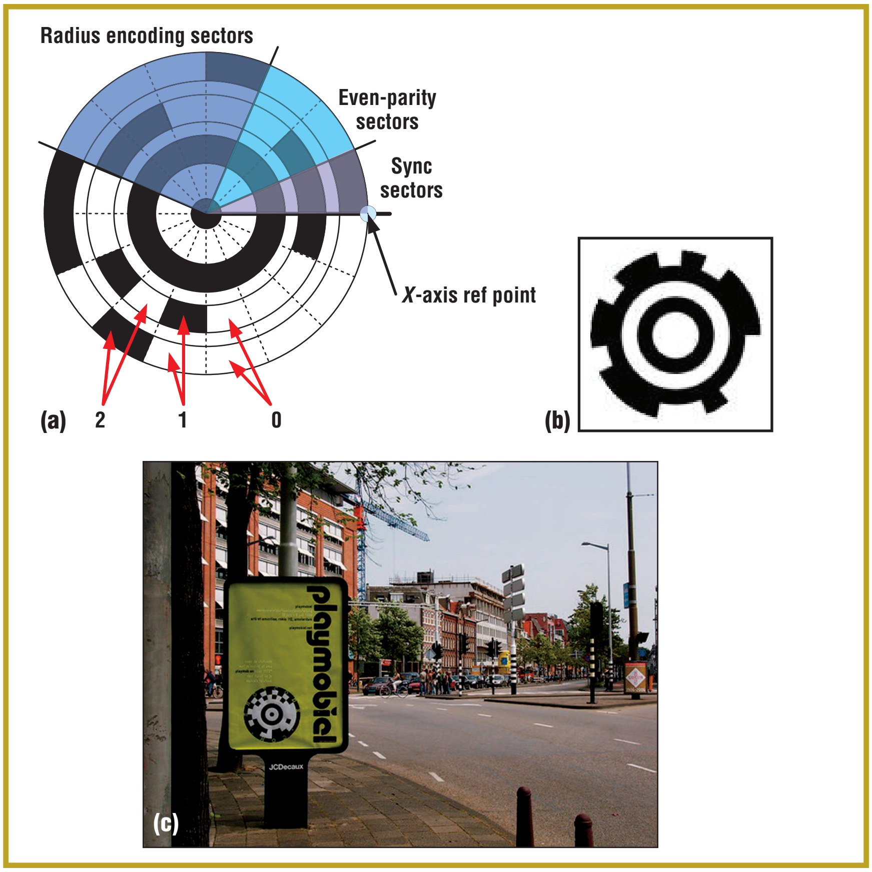


Figure 5. ShotCode: (a) the TRIP code symbol structure (image redrawn),[9] (b) SpotCode (the original ShotCode symbol), and (c) ShotCode as a physical hyperlink used for advertisement.

cular 2D-barcode tag or "ringcode" known as a TRIP (Target Recognition using Image Process) tag or TRIP code. Inexpensive CCD cameras such as Web cams used TRIP to dynamically track a moving target and obtain its real-time location information.[9]

TRIP tags encode a ternary number in the range of 1 to 19,683 (that is, 3^9) using two concentric rings surrounding a bull's-eye target. As figure 5a illustrates, the two rings are divided into 16 sectors. The first sector (the synchronization sector) indicates where the TRIP code begins. The subsequent two sectors store an even parity check on the encoded identifier (TRIP code), which detects possible decoding errors. The following four sectors encode the radius of its central bull's-eye in millimeters. The remaining nine sectors encode a ternary identifier. For example, the value of the tag in figure 5a is 102011221210001.

High Energy Magic originally used SpotCode (see figure 5b) and camera phones as sophisticated pointing devices that enable interactions between humans and active displays. After OP3 acquired SpotCode, it changed the name to ShotCode and redefined it as an index tag that links the real world to the digital

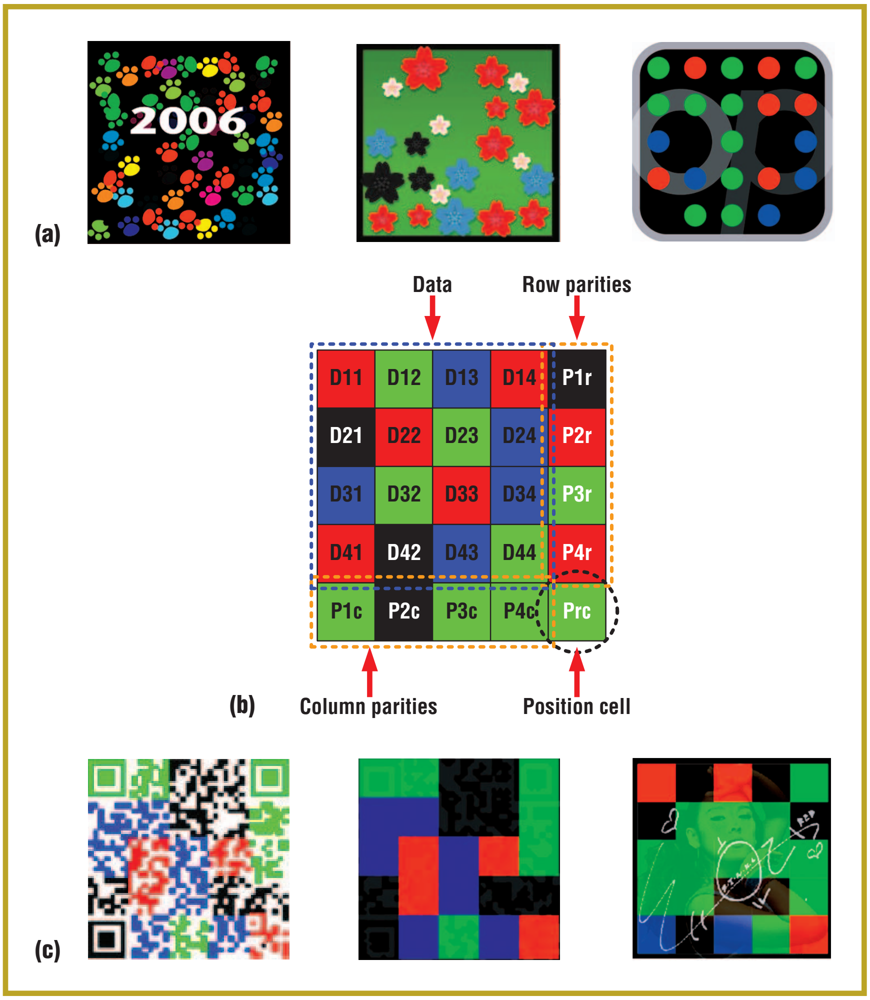


Figure 6. ColorCode: (a) design variations, (b) symbol structure, and (c) compound code.

world by accessing remote databases (see figure 5c).

ColorCode

In 2000, Tack-Don Han and his team at Yonsei University developed ColorCode.[10] This might be the most eye-catching visual tag to date that's also flexible in design and form (see figure 6a). A standard ColorCode tag encodes 10 digits and comprises a matrix of 5 × 5 cells rendered in a combination of four different colors—black, blue, green, and red. Cells can be circles, ovals, or polygons.

Different image-capturing, image-processing, and printing devices, as well as paper quality, can affect a color image's reproduction. Lighting conditions can also affect a captured image's color value. To overcome this problem, ColorCode uses reference cells that provide the standard color and hue for correctly distinguishing each reproduced cell. The value of each cell color and hue in the data area is determined relative to the value of the standard color in the reference cells. Because the relative difference between the cell color and hue and the standard color and hue is consistent, a reader can correctly retrieve the data.

ColorCode includes an *error parity check* that detects any incorrect color recognition and corrects it. The exclusive operation of code values in each column and row becomes the code value of the parity cell for the respective column and row, which the encoder converts to its corresponding color value in the symbol's parity area (see figure 6b). So if the encoder detects a different color in the parity area, it has misread the color in the corresponding row or column.

ColorCode decoding only requires 40 percent visibility of an individual cell. So, you can easily incorporate a barcode into a company logo or some other graphic design, making the most of the remaining 60 percent of the symbol's space. Recently, ColorZip Japan introduced a new type of ColorCode 2D barcode—mixed code—which comprises more than one type of barcode or symbol (see figure 6c).[11]

First-read rates

To verify our initial study's results,[3] we analyzed the first-read rate (FRR) of each sampled 2D barcode. We calculated the FRRs by dividing the numbers of successful first reads by the number of attempts made (50) to read each sample.

We created four samples for each of the six 2D barcodes (see figure 7a). To see how data quantity and symbol size affect each barcode's FRR, we encoded an identical set of data in the first two samples in two different symbol sizes (2.5 cm^2 and 5.0 cm^2) and repeated this operation using another identical set of data. (For the QR Code and VSCode samples, we used the most frequently selected error correction level—15 percent. The Data Matrix error correction levels are automatically set, so they were 28–39 percent for the first data sample and 22–34 percent for the second data sample.)

We used two different URLs as our test data: www.k.st/ and http://www.scis.ecu.edu.au/current/. The short URL let us encode the same information (that is, information to invoke an identical action) into all sampled 2D barcodes regardless of their data capacity so that we could better compare database and index-based barcodes.

Because there's no commercial version of the Visual Code system, we couldn't directly implement a database server to map a Visual Code to the URL. So, we converted the short URL

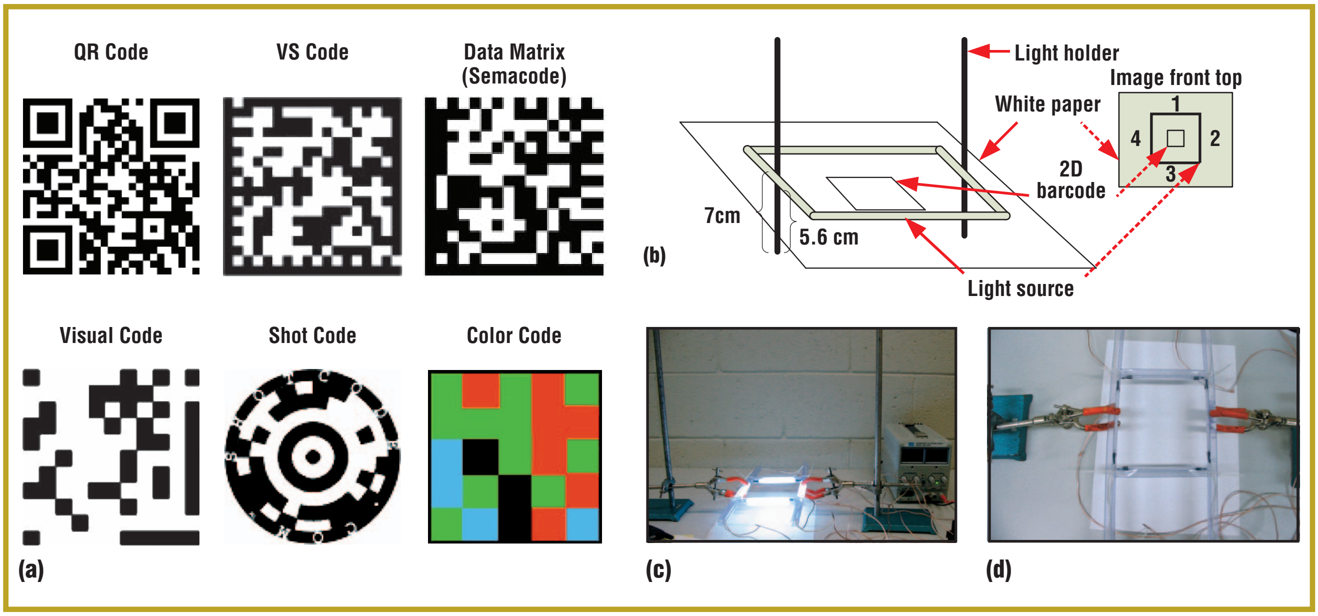


Figure 7. Analyzing first-read rates for each sampled 2D barcode: (a) examples of the 2.5 cm2 sample symbols with an identical short URL encoded within, (b) a diagram showing the various light sources, (c) an experiment using full-power lighting, and (d) an experiment using no additional lighting.

to its equivalent hexadecimal number (7777772E6B2E73742F) and encoded it as a Visual Code. However, we couldn't do this for the longer URL owing to Visual Code's low data capacity. To overcome this and yet maintain a fair comparison, we encoded the same quantity of numeric data as the longer URL test data set. This is consistent with the designed functionality of Visual Code as an index-based code, which usually carries a fixed quantity of numeric data.

Two camera phones captured each barcode sample: a Nokia 6600 with a VGA camera and a Nokia 6630 with a 1.3-megapixel camera. This let us observe the relationship between FRR and camera resolution. We also observed the performance of two different barcode readers installed on a Japanese camera phone (a Sanyo W33SA II). The readers were a built-in QR code and 1D-barcode reader and a Hybrid code reader that included a ColorCode reader, QR Code reader, and a Japanese Article Number Code (1D-barcode) reader. We used a PC-based VSCode decoder to decode images the Nokia phones captured because VSCode reader and decoder software for camera phones wasn't available.

We captured each sample image from the most appropriate distance (between 5 and 25 cm away from the target) using Cold Cathode Fluorescent lights under three lighting conditions:

- half power, first using the lights horizontally placed against the sample (light sources 1 and 3 in figure 7b) and then using lights vertically placed (light sources 2 and 4 in figure 7b);
- full power (see figure 7c); and
- no additional lighting (see figure 7d).

Ordinary fluorescent ceiling lights lit the room used for the experiments, so ambient light existed even when we used no additional lighting.

One nonskilled user performed the entire symbol-capture process. The user first performed several trials to get used to reading each sampled 2D barcode and to measure the appropriate distance from each target. So, we expected a certain amount of tilt and rotation of the captured images.

Results and observations

Overall, index-based 2D barcodes achieved better FRR results (99.3 percent) than the database 2D barcodes (88.1 percent). With the exception of ColorCode FRRs (97.9 percent), the FRRs of the index-based 2D barcodes were 100 percent regardless of lighting condition (see figure 8), symbol size, camera resolution, or data quantity (see table 1). However, the difference in the FRRs between the index and database 2D barcodes was insignificant. FRRs of the Quick Mark QR Code reader (www.quickmark.com.tw/English/download.html) and the Kaywa reader (http://reader.kaywa.com) used for decoding Data Matrix were also 100 percent.

Unlike the index 2D barcodes, the database 2D barcodes' FRRs were greatly impacted by factors such as symbol size and data quantity. Under the same conditions, the FRR of larger symbols was always higher than that of smaller ones. Likewise, generally, the FRR of denser symbols was lower than that of sparser symbols (see table 1). This is because data density, together with symbol size, determines each symbol's cell size.

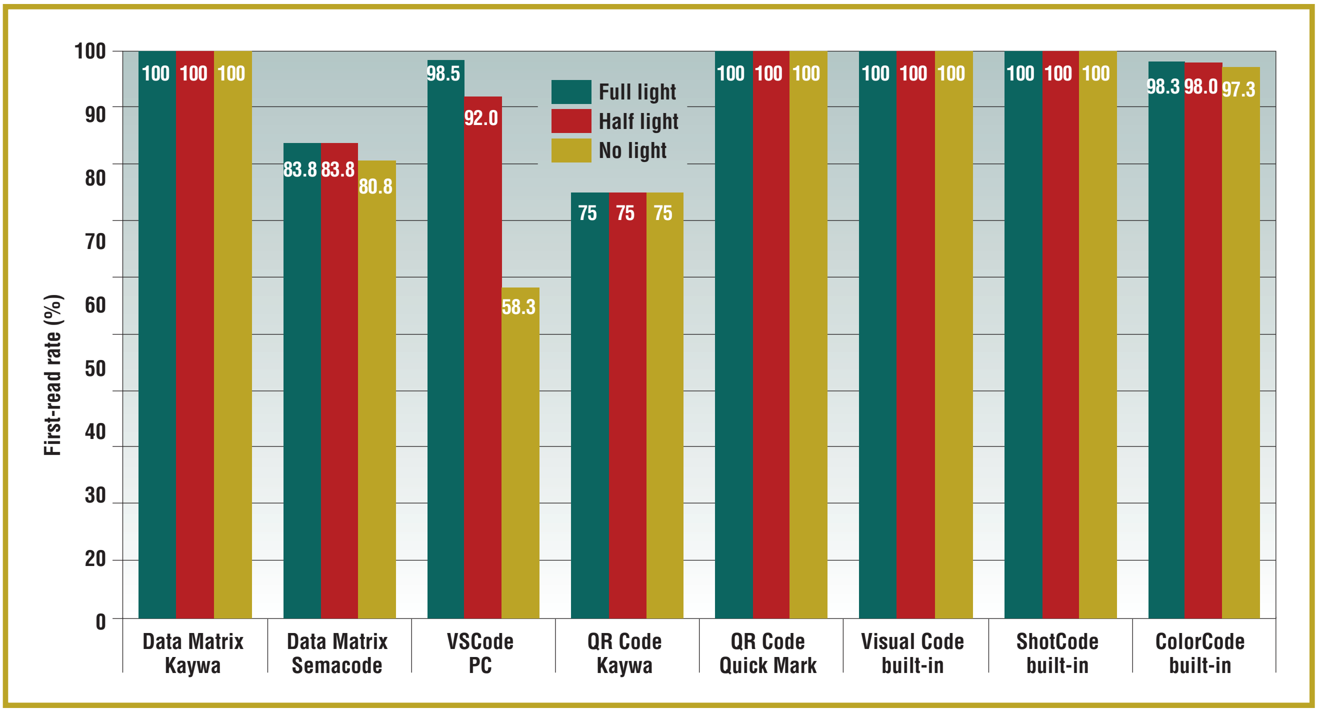


Figure 8. The mean FRR of each 2D barcode in different lighting conditions.

TABLE 1
Mean FRR (%) for each 2D barcode in different data quantities, symbol sizes, and camera resolution.

		Data Matrix		VSCode	QR Code		Visual Code	ShotCode	ColorCode
Data (URL)	Symbol size (sq.cm2)	Kaywa reader VGA/1.3M	Semacode reader VGA/1.3M	PC reader VGA/1.3M	Kaywa reader VGA/1.3M	Quick Mark reader VGA/1.3M	Built-in reader VGA/1.3M	Built-in reader VGA/1.3M	Built-in reader VGA/1.3M
Short	2.5	100/100	97.3/97.3	88.7/77.3	100/100	100/100	100/100	100/100	98.0/98.0
	5.0	100/100	99.3/99.3	100/87.3	100/100	100/100	100/100	100/100	98.0/96.7
Long	2.5	100/100	69.3/14.0	60.0/54.6	0/0	100/100	100/100	100/100	98.0/98.6
	5.0	100/100	93.3/92.0	95.3/100	100/100	100/100	100/100	100/100	98.0/97.3

Higher camera resolution wasn't important in reading the ColorCode or black-and-white 2D barcodes. This clearly debunked the myth that you need an expensive, high-resolution camera to read color symbols. In fact, the VGA camera produced higher FRRs than the 1.3-megapixel camera in most cases. Notably, the Semacode reader (http://semacode.org/about/hardware) for the Nokia 6600 performed five times better than the Nokia 6630 when reading a small, dense Data Matrix symbol. However, the Semacode and ColorCode readers used different versions of the software for each phone (www.colorcode.com/sg/Download.html). Moreover, differences in other camera features, such as autofocus and sensitivity to lighting, might affect their reading accuracy. So, while our results don't favor high-resolution cameras, it's still too early to conclude that lower camera resolution is always better, because other factors, such as hardware capability, reader software, and programming languages, can affect the outcome.

In terms of the maximum legible distance, the overall performance of database 2D barcodes (average measured distance: 21.7 cm) was greater than that of index-based 2D-barcodes (average measured distance: 16.9 cm). (The observations don't include VSCode with PC decoding and QuickMark with zooming function, unless stated otherwise.) Some highlights in our observations are the remarkable performance of the Semacode reader with a VGA camera and the Kaywa reader with a 1.3-megapixel camera in reading Data Matrix symbols. The average reading distances of these readers, when decoding the short URL, were 28 cm for the small symbol and 56 cm for the larger symbol. When decoding the

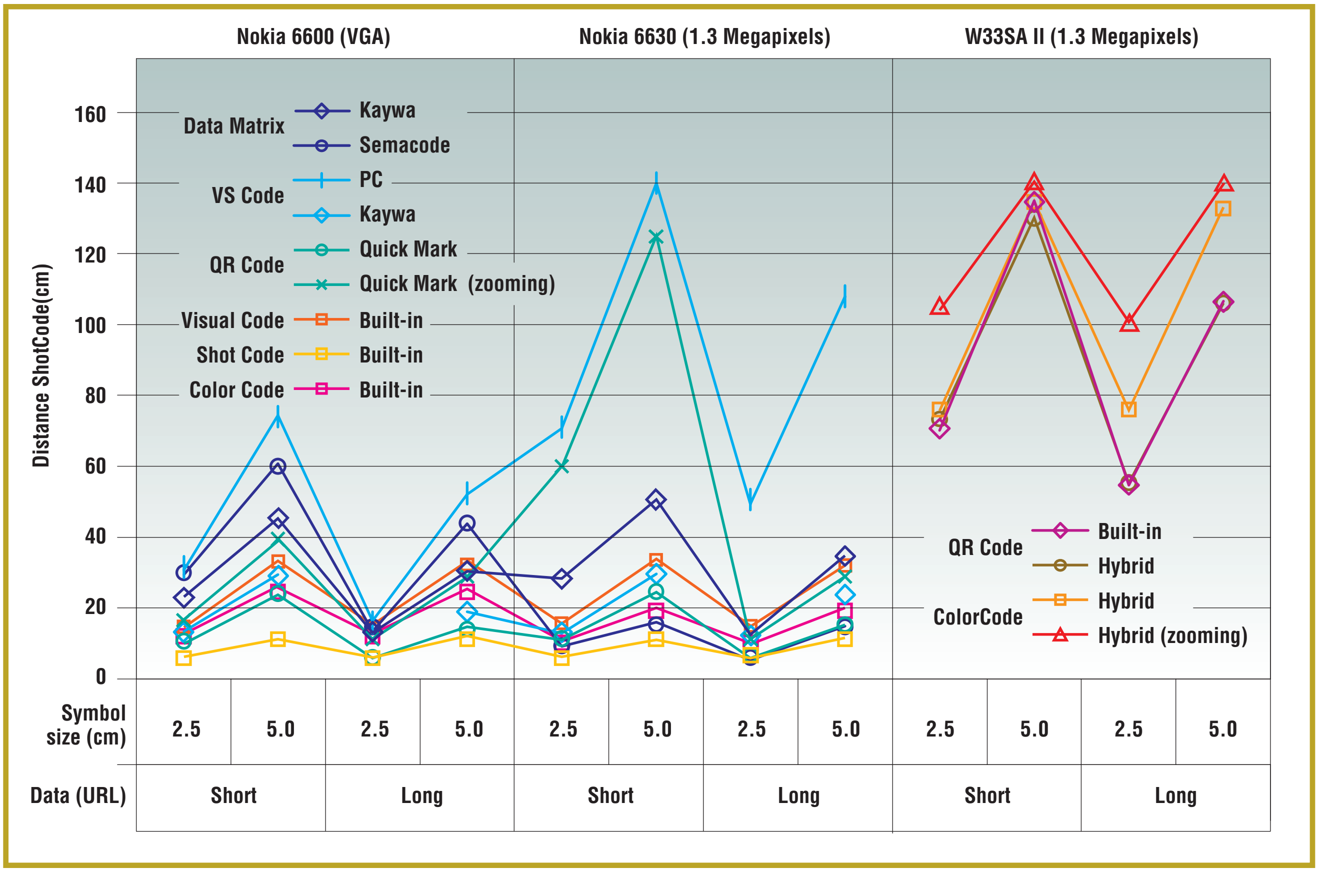


Figure 9. Maximum legible distance.

long URL, the average reading distances were 14 cm and 39 cm, respectively. In contrast, the legible distances of ShotCode readers (www.shotcode.com/dlownload) were rather poor (see figure 9).

Generally, the reading distance doubled when the symbol size doubled, regardless of the code type. However, the reading distance of the double-sized dense database symbols nearly tripled. This might be due to the use of error correction coding. When the codes are too small to be decoded properly, the whole decoding fails. However, once readability is no longer an issue, error correction capability dramatically improves the reading distance.

Camera resolution had only a negligible effect on the legible distance of all the sampled 2D barcodes except for the VSCode symbols that were decoded by the PC decoder and QuickMark (with a zooming function). (The VS Code reader, provided by Veritec Iconix Ventures [www.vi-vi.com/index.asp], was the only reader not publicly available.) The VSCode's distinctive performance in this instance might be due to the PC's superior processing power compared to the camera phones. Another noteworthy observation is the remarkable performance of the QR Code and ColorCode readers installed on the W33SA II with a 1.3-megapixel camera. The FRRs for both readers were 100 percent. Furthermore, the reading distance of both was comparable to that of the VSCode with PC decoding (see figure 9).

Three key factors to improving the robustness of 2D-barcode reading are the cell size of symbols, the software's decoding algorithm, and the reader's hardware capability. For example, for the Kaywa reader, the FRRs of the $2.5cm^2$ QR Code with dense data were 0 percent regardless of camera resolution, whereas those of Data Matrix were 100 percent. The difference between them was cell size. When a 2D barcode's cells become smaller than the recognizable size for a particular reader, then the reader can't successfully read the code.

The Quick Mark reader was superior to the Kaywa reader in terms of FRR. However, the reverse was true for legible reading distance. Such differences in reading capability should result from variations in the reading software algorithm. Visual Code's strong and continuous tracking ability also shows that reader software makes a difference (www.inf.ethz.ch/personal/rohs/visualcodes).

Better hardware capability could improve the decoding algorithms' performance. For example, as figure 9 shows, the Quick Mark reader can improve its reading distance by six times using the Nokia 6630's zooming function (that is, 6x digital zoom) but only up to two times using the Nokia 6600 (that is, 2x digital zoom). The W33SA II's autofocus significantly contributed to decoder robustness.

No 2D barcode completely satisfied the criteria of a global 2D-barcode standard for mobile applications. However, 2D-barcode technologies are rapidly improving. For example, newly released database 2D-barcode readers such as Kaywa and Quick Mark performed nearly as well as the index-based 2D-barcode decoders. The quality of camera phone hardware has also continuously improved. Furthermore, our study identified three key factors where researchers can improve the robustness of 2D-barcode reading and the reading distance. Applications that let camera phones interact with 2D barcodes on a plasma display will no longer be a utopian dream. Considering the rapid improvements in decoding reliability, availability, and functionalities, we conclude that database 2D barcodes are more suitable as candidates for a global standard.

However, the recent trend in Japan is that more than one barcode system such as QR Code, ColorCode, and the JAN code coexist in the marketplace. (The JAN code reader's FRR is 100 percent but its reading distance is limited to approximately 20 cm.) If the proper infrastructures and systems are in place, widespread use of 1D barcodes for mobile applications is possible, because they're already printed on billions of product packages worldwide. The introduction of mixed code, which comprises more than one type of barcode or symbols, also follows this trend. So, instead of choosing one standard 2D-barcode system, users might prefer different types of barcode systems according to the mobile applications they need.

the AUTHORS

Hiroko Kato is a PhD candidate with the School of Computer and Information Science at Edith Cowan University. Her research interests are 2D barcodes and pervasive mobile computing technologies. She received her BS in computer science from Edith Cowan University. She's a student member of the IEEE and the Institute of Electronics, Information, and Communication Engineers (IEICE) Japan. Contact her at the School of Computer and Information Science, Edith Cowan Univ., 2 Bradford St., Mount Lawley 6050, Western Australia, Australia; hkato@student.ecu.edu.au.

Keng T. Tan is a senior lecturer and coordinator of the Mobile Computing Program at Edith Cowan University's School of Computer and Information Science. His research interests include mobile networks, pervasive mobile computing, and signal processing for mobile communications. He received his PhD in electrical and electronic engineering from the University of Western Australia. He's a member of the IEEE and the Institute of Engineers, Australia. Contact him at the School of Computer and Information Science, Edith Cowan Univ., 2 Bradford St., Mount Lawley 6050, Western Australia, Australia; a.tan@ecu.edu.au.

REFERENCES

1. J. Rekimoto and Y. Ayatsuka, "CyberCode: Designing Augmented Reality Environments with Visual Tags," *Proc. Designing Augmented Reality Environments* (DARE 00), ACM Press, 2000, pp.1–10.
2. *2 Dimensional Symbologies Handbook*, Keyence Corp., May 2005; www.keyence.co.jp/ req/info/2jigenpdf/show.jsp [in Japanese].
3. H. Kato and K.T. Tan, "2D Barcodes for Mobile Phones," *Proc. 2nd Int'l Conf. Mobile Technology, Applications and Systems* (MTAS 05), IEE, 2005, pp. P1A–4.
4. *Information Technology—Automatic Identification and Data Capture Techniques—QR Code 2005 Bar Code Symbology*, ISO/IEC 18004, Int'l Organization for Standardization, 2006.
5. *Understanding 2D Symbologies: Detailed Information on Barcodes*, Japan Automatic Identification Systems Assoc., 2004.
6. *Automatic Identification and Data Capture Techniques—Data Matrix Bar Code Symbology Specification*, ISO/IEC 16022, Int'l Organization for Standardization, 2006.
7. M. Rohs, "Real-World Interaction with Camera-Phones," *Proc. 2nd Int'l Symp. Ubiquitous Computing Systems* (UCS 04), LNCS 3598, Springer, 2005, pp. 74–89.
8. A. Madhavapeddy et al., "Using Camera Phones to Enhance Human-Computer Interaction," *Adjunct Proc. 6th Int'l Conf. Ubiquitous Computing* (UbiComp 04), 2004; http://ubicomp.org/ubicomp2004/adjunct/demos/madhavapeddy.pdf.
9. D. López de Ipiña, P.R.S. Mendonça, and A. Hopper, "TRIP: A Low-Cost Vision-Based Location System for Ubiquitous Computing," *Personal and Ubiquitous Computing J.*, vol. 6, no. 3, 2002, pp. 206–219.
10. T.-D. Han et al., *Machine Readable Code Image and Method of Encoding and Decoding the Same*, US Patent 7,020,327, US Patent and Trademark Office, 2006.
11. C.-H. Cheong et al., *Mixed Code and Method and Apparatus for Generating the Same*, Japan Patent 134,336, Japan's Patent and Trademark Office, [US Patent 20060097062, US Patent and Trademark Office], 2006.