# Unsynchronized 4D Barcodes

## Coding and Decoding Time-Multiplexed 2D Colorcodes

Tobias Langlotz and Oliver Bimber*

Bauhaus-University Weimar

**Abstract.** We present a novel technique for optical data transfer between public displays and mobile devices based on unsynchronized 4D barcodes. We assume that no direct (electromagnetic or other) connection between the devices can exist. Time-multiplexed, 2D color barcodes are displayed on screens and recorded with camera equipped mobile phones. This allows to transmit information optically between both devices. Our approach maximizes the data throughput and the robustness of the barcode recognition, while no immediate synchronization exists. Although the transfer rate is much smaller than it can be achieved with electromagnetic techniques (e.g., Bluetooth or WiFi), we envision to apply such a technique wherever no direct connection is available. 4D barcodes can, for instance, be integrated into public web-pages, movie sequences, advertisement presentations or information displays, and they encode and transmit more information than possible with single 2D or 3D barcodes.

## 1 Introduction and Motivation

Encoding and decoding digital information into printed two dimensional barcodes becomes more and more popular. They are used in advertisements, on business cards or e-tickets, or for referencing to web-pages as in Semapedia (www.semapedia.org). The amount of information that can be decoded robustly from a 2D barcode with ordinary mobile devices, such as mobile phones, is usually restricted to several characters only. Thus, usually IDs, URLs or simple addresses are encoded. Yet, professional industrial scanners are able to decode a much larger amount of characters (several thousands) with an acceptable reliability. In this paper we present a new kind of barcode that we refer to as *4D barcode*. It encodes data in four dimensions: width, height, color and time. Consequently, it cannot be printed on paper but is displayed on screens of mobile or spatial devices. Time-multiplexing colored 2D barcodes allows to transmit a larger amount of information robustly to off-the-shelf mobile phones without requiring an explicit synchronization (cf. figure 1(a)).

* e-mail:{Tobias.Langlotz, Oliver.Bimber}@medien.uni-weimar.de

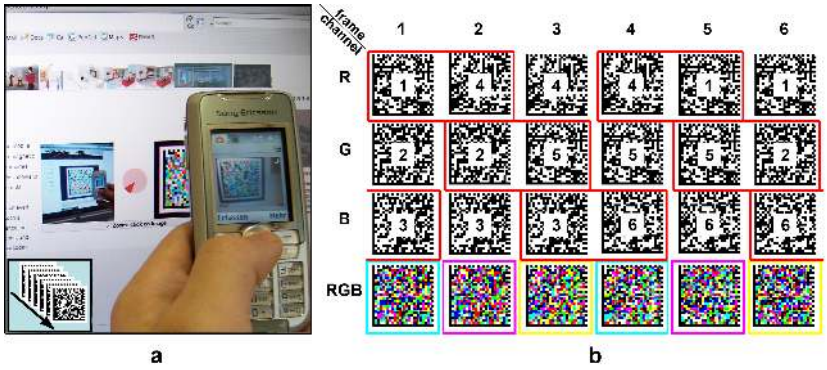


**Fig. 1.** 4D barcodes: (a) displaying and capturing, (b) encoding scheme for embedding 2D barcodes into a sequence of time-multiplexed 3D barcodes (barcode transitions are framed in red).

## 2 Related Work

A large number of applications for mobile phones exist that read and decode printed *QR-codes* [1] as a standardized black-and-white 2D barcode. *Datamatrix* [2] is a similar example (yet not as common as QR-codes) and is used by applications like Semacode (www.semacode.com). As mentioned earlier, only a small amount of information can be decoded robustly with consumer camera phones - limiting QR-code or Datamatrix barcodes to encode a few characters only. Han et al. [3] propose *Colorcode*, a 3D barcode which -in addition to a 2D matrix layout- uses colored bits as third dimension. But due to its small resolution of 5x5 cells it encodes only IDs that are resolved through a central lookup service (www.colorzip.co.jp). Besides applications in advertisement, the Colorcode is used for context aware systems [4]. 2D barcodes have also been applied to realize interaction techniques with mobile phones. Rohs [5] describes a barcode named *Visual Code* that stores up to 83 bits of data. Displaying it on a screen, it is used for tracking the movement and rotation of the phone relative to the screen [6, 7]. Another novel approach is presented by Scott et al. [8], who use *Spotcode* (a circular 2D barcode) for out-of-band device discovery and service selection - bypassing the standard Bluetooth in-band device discovery. This is applied by Madhavapeddy et al. [9] to also implement interaction techniques with Spotcodes that are displayed on a screen - using an online Bluetooth connection for data exchange. Similar techniques that apply displayed 2D barcodes for supporting mobile phone based interaction methods in combination with screens can be found in [10–12].

Besides barcodes, other possibilities for optical data transfer exist. Shen et al. [13], for example, explain how to read 7-segment digits from LCD/LED displays with camera equipped mobile phones. This system was mainly developed to support people with vision disorders by using their phones to recognize and read the digits on simple displays, such as on clocks. The system requires approximately

two seconds for capturing and reading the digits on a Nokia 6630 - which is comparable to other OCR Software for mobile phones. Yet another interesting new approach for transmitting data optically is to use light sources instead of displays. In *visible light communication*, ordinary light sources are modulated with a digital signal. Approaches presented by Tanaka et al. [14] or Komine and Nakagawa [15] use white-light LEDs for illuminating a room and for transmitting time-multiplexed signals. The modulation frequency is high enough so that the transmitted signal remains invisible to the human eye. The embedded signal is received by photo diodes and is finally decoded. Using such a system, it is possible to transmit up to 100 Mbit/s and more.

## 3 Time-Multiplexed Colored 2D Barcodes

One possibility to enlarge the data volume that can be embedded into a 2D barcode is to increase the code matrix resolution. However, the optics used for consumer cameras set clear limitations. Consequently, this is not an option when off-the-shelf mobile phones are used. The main idea of 4D barcodes is to split

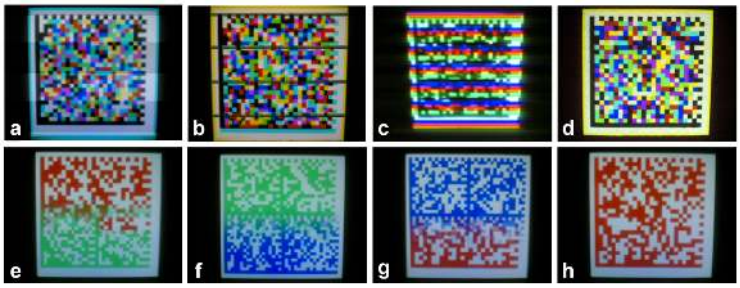


**Fig. 2.** Captured unsynchronized 3D barcodes from (a) 120Hz and (b) 60Hz CRT monitor, (c) DLP projector with white color wheel segment, and (d) LCD projector. Time-multiplexed R,G,B 2D barcode sequence captured form LCD monitor, (e-g) with and (h) without frame transitions.

the data into smaller chunks that are embedded into a series of decodable 3D barcodes. Thereby, the color dimension is used for increasing the robustness of the transmission. The animated 3D barcodes are displayed in an endless-loop on screens, and can be recorded by mobile camera phones. The looping duration and state is visually indicated on the display to give a feedback on how long the code sequence needs to be captured. After recording, individual barcodes are extracted, assembled and decoded on the phone to reconstruct the entire data content. Thereby, the challenge is the missing synchronization between displaying and recording. Our system is able to support LCD panels (or projectors) and Plasma screens. CRT monitors (or projectors) can only be applied if the decay rate of the utilized phosphor and the display's refresh rate

ensure no full blank regions during the integration time of the camera chip. We found that fast 120Hz CRT monitors (cf. figure 2a) are sufficient, while most slow (e.g. slower than 85Hz) CRT monitors (cf. figure 2b) are not. Due to an image generation via time-multiplexing (color and gray levels), DLP-based displays (i.e., projectors or back-projected screens) are not supported (cf. figure 2c). We use Datamatrix barcodes in our prototype for encoding and decoding since decoders are freely available. Yet, it is extended to carry nested color bits. Animated GIFs are used to display the sequences of color codes. They can be easily embedded into web-pages. The following sections describe the encoding and decoding process in more detail.

### 3.1 Encoding

As mentioned above, the whole data set is split into smaller portions. They are encoded into a series of 2D Datamatrix barcodes having a size and resolution that can be decoded robustly by consumer phones. Binary data is preconverted into a sequence of 6-bit characters that is supported by Datamatrix. Therefore, we apply a similar technique as proposed by Josefsson [16]. After decoding, the reconstructed 6-bit character sequence is converted back to its original format. Furthermore, the data can be compressed before encoding and is uncompressed after decoding to achieve a possibly higher throughput. The sequence of 2D barcodes are then converted into an animated GIF for presentation and recording. Due to the missing synchronization between camera phone and display, however,

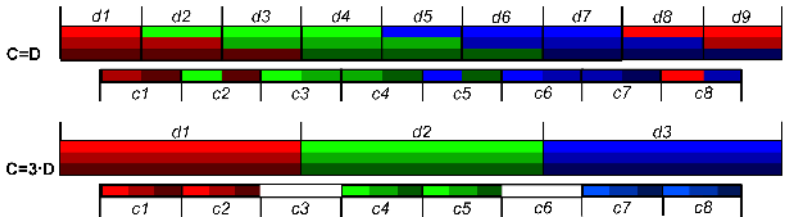


**Fig. 3.** Encoding: sifted (top) and non-shiftet (bottom) encoding scheme ($c_i$ and $d_i$ are captured and displayed frames respectively, the individual frame-embedded and captured/transmitted barcodes are color-coded).

such a simple approach would be very vulnerable to failures. The reason for this is that during the integration time of the camera, the screen content can change. This effect is illustrated in figures 2e-f for an LCD display. Here, full red, green, and blue 2D barcodes are displayed sequentially. Recording the sequence might show two different frame portions (and consequently two different barcode portions) in the same captured image. We solve this synchronization problem with a new encoding scheme. Instead of encoding one 2D barcode, we encode three different 2D barcodes simultaneously into each frame of the displayed sequence.

Each of them will be embedded into the red, green and blue color channels - making it a 3D barcode. This, however, is not being done to triple the transfer throughput, but to increase the robustness of the system by adding redundancy. Every 2D barcode of the original sequence is embedded exactly three times - ones in each of three subsequent 3D barcodes, and it is always encoded into the same color channel. This is illustrated in figure 1(b). Only one 2D barcode is replaced between two subsequent frames. Combining the three color channels in each frame leads to the displayed colored 3D barcodes in lower row of figure 1(b). In addition, we surround each 3D barcode by a colored border. This is necessary for detecting if a captured frame was recorded while the barcode was replaced. Therefore the border color is alternating between yellow, magenta and cyan - colors that are complementary to the RGB code colors. Furthermore, the border color allows detecting which barcodes are encoded and which one will be replaced in the next frame. We use the complementary border color for indicating an upcoming barcode transition within a particular color channel. For example, if a barcode will be replaced in the next frame's blue channel, the border color for the current frame is chosen to be yellow.

Our encoding scheme (figure 3-top) applies equal capture ($C$) and display ($D$) rates and adds a two-fold redundancy. It shifts each barcode to three subsequent display frames and ensures that it can be captured completely in at least two frames. The same result (i.e., redundancy and transmission rate) could be achieved with an unshifted encoding scheme and with $C = 3 \cdot D$ (figure 3-bottom), for example. Shifting, however, increases the recognition probability during code resolution transitions and for non-constant capturing times (caused by online JPEG compression in our case). Note that both cases satisfy the Nyquist-Shannon theorem.

### 3.2 Decoding

After the sequence of 3D barcodes have been recorded on the mobile phone, each captured frame is analyzed for extracting the individual 2D barcodes and finally the encoded information. This task can be split into two preprocessing steps and one final decoding step, as illustrated in figure 4.

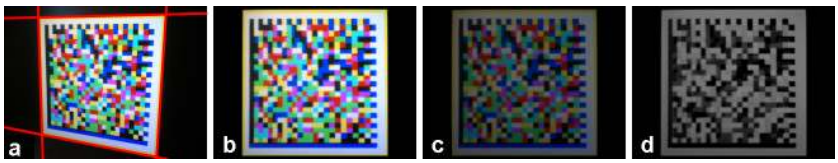


**Fig. 4.** Preprocessing steps for decoding: (a) captured frame and detected corners, (b) rectified image, (c) contrast and brightness adjusted image, (d) extracted 2D barcode (red channel) in gray scale.

**Preprocessing:** During the preprocessing steps, the 3D barcode in each captured frame is rectified to compensate for perspective distortions and is then contrast and brightness enhanced to compensate for noise. For rectification, the edges of the colored borders are detected through a conventional scan line algorithm. Having found multiple points on each edge, the corresponding line equations can be determined by solving a linear equation system. The intersections of the four edge lines lead to the corner points of the border that can be used to estimate a homography for rectification [17]. In our current prototype, the rectification works well for small barcode resolutions, but our Datamatrix decoder (we applied the Symbian Semacode library) fails often for rectified high resolution codes. However, this has not been critical in our case, since we have to limit the barcode resolution for mobile phone decoding anyway.

$$col_{new} = a \cdot col_{old} + b, a = 255/(255 - 2 \cdot \Delta), b = a \cdot (l - \Delta), \Delta = 127 \cdot c/100 \quad (1)$$

Following this step, the contrast and the brightness of the rectified images are adjusted using equation 1 to reduce image noise. Experimentally we found that a constant brightness reduction of $l$=20% and a constant increase in contrast of $c$=50% was optimal in combination with the (unknown) build-in white-balancing function of our mobile phones. This pushes the recognition rate up by a maximum of 20% (compared to no adjustments).

**Handling Code Transitions:** After optimizing the captured frames, the embedded 2D barcodes can be extracted and decoded. The first step is to detect whether or not a barcode transition happened within a frame. This can be detected by analyzing the border color (which has already been found during rectification, as explained above). If the colors of the upper and the lower border edges are the same, the barcode recorded in the frame is consistent. In this case, all three 2D barcodes that are encoded into the RGB color channels are completely captured. They can be separated, converted into gray scales, and decoded by the Datamatrix decoder. If the colors of the upper and lower edge are unequal, an inconsistency is detected. However, due to our encoding scheme it is possible to guarantee that always two barcodes are consistent (and completely recorded) in one frame. The reason for this is that only one of the three 2D barcodes is replaced between two subsequent 3D barcodes images. By analyzing the color of the upper border edge, we can determine in which color channel a 2D barcode is replaced in the following 3D barcode image (and is consequently recorded inconsistently in the current frame), and which ones are completely recorded. In correspondence to the coding example from section 3.1, a yellow upper border indicates a code transition in the blue color channel. Thus, two different barcodes are captured in the upper and in the lower portions of the current frame's blue channel, while the barcodes in the red and green channels are consistent and complete, in this example. The same applies for the other two possible variations. The complete 2D barcodes can be decoded after converting them into gray scales (cf. figure 5). Note, that the intensity variations of the code bits are not critical for decoding.The inconsistent 2D barcode is discarded,

but our encoding scheme guarantees that it will be complete in at least two of the three frames in which it was encoded (i.e., in the best case the same barcode is consistent in all three 3D barcodes; in the worst case it is only consistent in two 3D barcodes). After decoding the individual 2D barcodes, the encoded data

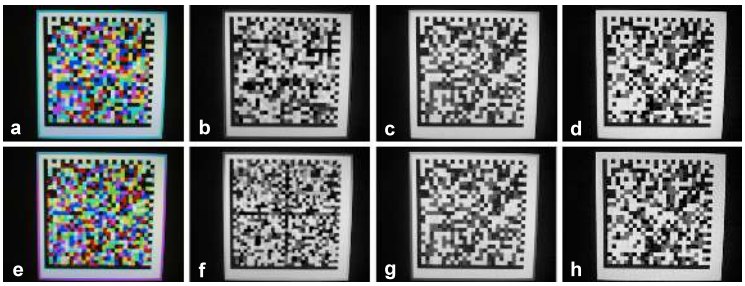


**Fig. 5.** Decoding: (a) captured 3D barcode with all three embedded 2D barcodes (b-d) completely recovered, (e) example with inconsistent 2D barcode in red channel (f) while green and blue channels can be recovered (g+h).

packages from each one have to be rebuilt in the primal order of coding. Since it is possible that entire 2D barcodes cannot be decoded at all, and it is likely that recording the 3D barcode sequence does not start with the first frame (users might start recording at an arbitrary point during the looped image sequence), a correct order of reconstructed data packers is not given by the order of decoding. To overcome this problem, we add a unique frame ID into a header section of each 2D barcode. This allows rebuilding the entire data set in the correct order. The data might have to be uncompressed and transformed back into the original representation, if necessary (in analogy to the encoding, as explained in section 3.1).

## 4 Results

We tested and optimized our system in two steps: First, it was evaluated with respect to adjustable parameters, such as animation speed, barcode size, capturing resolution, and an optional data compression to maximize its data throughput and robustness. Second, we have carried out a user study to find the final transfer and failure rates under realistic conditions, as well as to get feedback on the acceptance of our approach.

### 4.1 Optimizing Parameters

Several parameters can freely be chosen in our system: the size of the 2D barcodes (i.e., the number of encoded characters per barcode), the capturing resolution (the capturing speed depends on the adjusted resolution), the animation speed

(i.e., the frame rate of the displayed 3D barcode sequence), and whether or not the data should be compressed and decompressed.

All of these parameters interplay with each other and influence the final result. While, for instance, choosing a small capturing resolution and a high animation speed might allow to recorded many 3D barcodes during a particular time period, recognition can fail often since capturing might become too unreliable. As another example, encoding many compressed characters into a single barcode might maximize the transmission of data per 3D barcode, but more time is required for decoding and uncompressing the data. If the recognition rate drops, as yet another example, barcodes might have to be decoded again - which also costs additional time and consequently reduces the overall transfer rate. To achieve the highest possible transfer rate and robustness, the optimal configuration of parameters have to be found. For this, we designed several experiments that recorded recognition and transfer rates under varying parameter settings. All experiments were carried out with a Nokia 6630 mobile phone, using version 1.5 of the Semacode Symbian C++ library for decoding Datamatrix barcodes. First, the recognition rate for 2D barcodes with respect to different capturing resolutions, barcode sizes, and an optionally applied compression was evaluated. As it can be seen in figure 6, the highest recognition

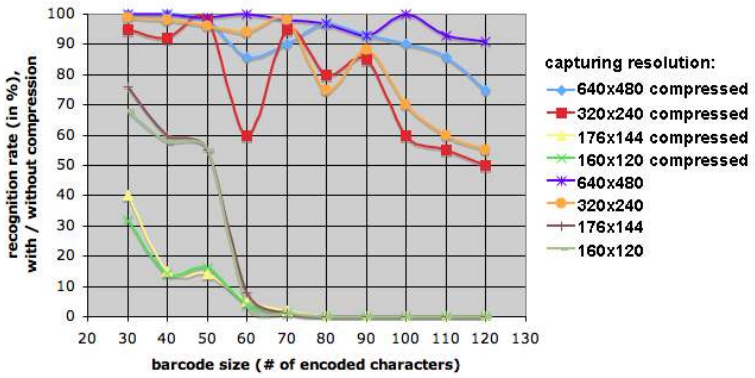


**Fig. 6.** Recognition rate: successful decoded 2D barcodes under varying capturing resolutions, barcode sizes, and an optionally applied compression.

rates are achieved with capturing resolutions of 320x240 pixels (QVGA) and 640x480 pixels (VGA), while smaller resolutions are mainly not suitable for decoding barcodes sufficiently robust as their sizes increase. In general, barcodes containing uncompressed data decode slightly better because smaller barcodes are required in this case. The reason for this is that most 2D barcodes, such as Datamatrix, are normally used for encoding text and optimize their matrix sizes depending on the probability of character appearance in defined alphabets (e.g. capital letters are less likely and require more coding bits within the barcode

matrix than lower-case letters). Compressed data (we applied a deflate compression [18]) is transformed to random characters that require larger matrix sizes in general. In a second experiment, we evaluated the resulting transfer rate against

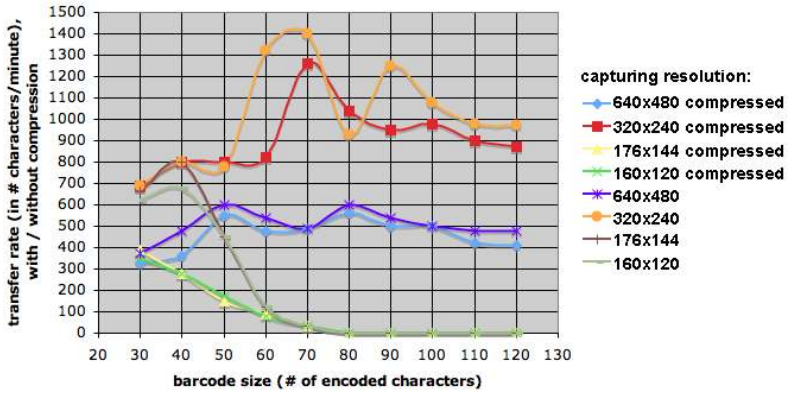


**Fig. 7.** Transfer rates: number of transmitted characters per minuted under varying capturing resolutions, barcode sizes, and an optionally applied compression.

the same parameters as for the recognition rate. It can be seen in figure 7 that the transfer rate is maximal when encoding 70 characters in a single 2D barcode and capturing with a QVGA resolution. With respect to figure 6, a VGA capturing resolution is more accurate, but requires significantly more time, and consequently leads to a lower overall transfer rate. Compressed data performs worst than uncompressed data in this case for the same reason as explained above. Note, that the steep drop-off of recognition rate at around 60 characters is due to possible resolution transitions of the code matrix that require the integration of helper lines (cf. figure 5(f)). If 2D barcodes with different resolutions (depending on the encoded content) are encoded into the same 3D barcode, the recognition rate drops significantly if helper lines are inserted.

Based on our experiments, we apply a QVGA capturing resolution, no compression, and encode 70 characters per 2D barcode. This leads to a maximum transfer rate of 1400 characters per minute (23 characters per second) and to a maximal recognition rate of 95% for an experienced user. The fastest capturing rate that was supported by our mobile phone at QVGA resolution (with view finding enabled and direct recording into the integrated flash memory) was 2.5 frames per second.

### 4.2 User Study

To verify our system using optimal settings under realistic conditions, we carried out a study with a total of 48 unexperienced users during a public event (university's open house). For an experiment we encoded 700 characters of text

into a 13x13 cm large 4D barcode that was played with 2.5 fps in a looping animated GIF sequence embedded into a HTML page. The page was displayed in a web browser on a 17 inch LCD screen. Nokia 6630 mobile phones were used for capturing and decoding. The users were able to see the live recording of the camera on the LCD panel of the phone. We asked them to fill as much as possible of the recorded image with the displayed barcode. By pressing a button on the phone, they triggered the beginning and the end of the recording. After recording, the barcode was decoded on the phone. With this study, we were mainly interested in finding the realistic recognition behavior of our system and on getting concrete user feedback. Unexperienced subjects might not always use the system in an optimal way (e.g. they might sometimes not capture the entire barcode image due to arm jitter, or they might record the barcode from a too large distance). On average, we found a recognition rate of 73% for individually extracted 2D barcodes, but due to the encoded redundancy the overall recognition rate was 82% under realistic conditions. Consequently, in 18% of all cases, some parts of the 700 character text were missing while in all remaining cases the whole text was recovered. The averaged time for decoding was about 35 seconds. This is mainly due to the performance of the Semacode library and could not be influenced by our system. However, since decoding was carried out after recording, the users did not have to aim the phone at the screen for this duration. Only for recording (on average 5 seconds), this was necessary. Each subject was finally asked to fill out a questionnaire to provide feedback on the usability of the system - rating various questions between 7 (very good) and 1 (very poor). The averaged results from the questionnaires are shown in figure 8. In general, we can say that the user feedback was overall positive. The long

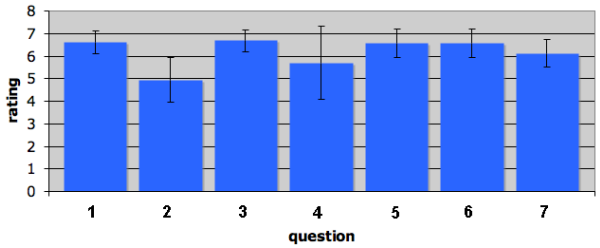


**Fig. 8.** Results of the user feedback: 1) How easy was it to aim at the barcode? 2) How do you rate the decoding time? 3) How easy was it for you to learn how to use the system? 4) How do you judge the recognition rate? 5) How easy was the handling of the software? 6) How good was the graphical user interface? 7) How much do you like the general concept?

decoding time was criticized most. As mentioned above, this was mainly due to long decoding requirements of the Semacode library which our system could not influence.

## 5 Summary and Future Work

In this paper we presented the concept and an implementation of unsynchronized 4D barcodes. With our technique, we are currently able to transmit 1400 characters per minute (23 characters per second) with a success rate of 82% (95% for experienced users) from LCD, Plasma, and fast CRT displays to unsynchronized mobile phones. A user study has shown that such a technique would be accepted, if the decoding speed can be improved. Our technique has a much smaller transmission rate than established electromagnetic techniques, such as Bluetooth or WiFi. But it can be used in cases where such connections are are not established per se. Furthermore, it transmits significantly more data than corresponding 2D or 3D barcodes. Besides transmitting data from location- and device-independed public web-pages, we envision applications for recorded and broadcasted video content, for advertisement with billboard displays (as being done already with 2D barcodes) or in movie theaters, for information displays (e.g., transmitting updated schedules at airports or in trains), or for electronically displayed e-tickets (2D barcodes are already accepted to be displayed on mobile phones instead on printed paper). In future, the decoding time has to

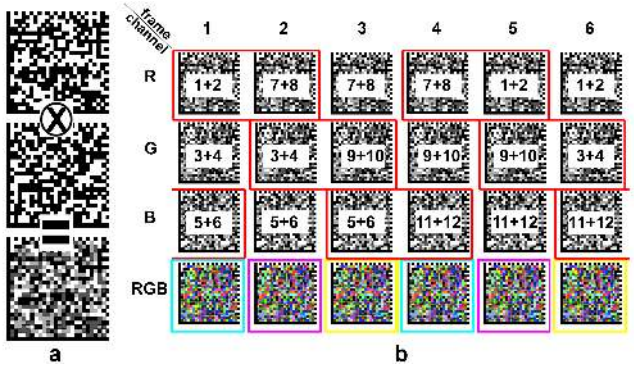


**Fig. 9.** 5D barcodes: (a) intensity coding of two 2D barcodes, (b) encoding scheme for embedding intensity coded 2D barcodes into a sequence of time-multiplexed 3D barcodes (barcode transitions are framed in red).

be decreased. The Kaywa-Reader (reader.kaywa.com) for example, offers a more robust decoding and a speed-up by a factor of two compared to Semacode. Porting our system to newer Symbian versions (e.g., Symbian Series 60 Version 9.x) allows benefitting from improved camera control functions. This may increase the quality of the captured frames and might open the door to embedding six or more 2D barcodes instead of only three by using discrete intensity variations in addition (cf. figure 9). We have implemented and tested these 5D (height, width, color, time and intensity) barcodes, but deferred them due the too low image quality provided by the utilized mobile phones.

## References


1. ISO/IEC: International Organization for Standardization: QR Code.ISO/IEC 18004. (2000)
2. ISO/IEC: International Organization for Standardization: DataMatrix. ISO/IEC 16022. (2000)
3. Han, T.D., et. al.: Machine readable code and method and device of encoding and decoding the same, japan patent 3336311 (2002)
4. Han, T.D., et. al.: Implementation of personalized situation-aware service. In: Proceedings of the First Internaltional Workshop on Personalized Context Modeling and Management for UbiComp Applications (ubiPCMM 2005). (2005)
5. Rohs, M.: Real-world interaction with camera-phones. In: 2nd International Symposium on Ubiquitous Computing Systems (UCS 2004). Number 3598 in Lecture Notes in Computer Science (LNCS), Tokyo, Japan, Springer-Verlag (2004) 74–89
6. Ballagas, R., Rohs, M., Sheridan, J.G.: Sweep and point and shoot: phonecam-based interactions for large public displays. In: CHI '05: CHI '05 extended abstracts on Human factors in computing systems, New York, NY, USA, ACM Press (2005) 1200–1203
7. Rohs, M.: Visual code widgets for marker-based interaction. In: IWSAWC'05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems – Workshops (ICDCS 2005 Workshops), Columbus, Ohio, USA (2005)
8. Scott, D., S.R.M.A., Upton, E.: Using visual tags to bypass bluetooth device discovery. ACM Mobile Computer Communications Review **9** (2005) 41–53
9. Madhavapeddy, A., S.D.S.R., Upton, E.: Using camera-phones to enhance human-computer interaction. In: Adjuct Proc. of Ubicomp 04, Springer-Verlag (2004)
10. Toye, E., Sharp, R., Madhavapeddy, A., Scott, D., Upton, E., Blackwell, A.: Interacting with mobile services: an evaluation of camera-phones and visual tags. Personal Ubiquitous Comput. **11** (2007) 97–106
11. Vartiainen, P., Chande, S., Rämö, K.: Mobile visual interaction: enhancing local communication and collaboration with visual interactions. In: MUM '06: Proceedings of the 5th international conference on Mobile and ubiquitous multimedia, New York, NY, USA, ACM Press (2006) 4
12. Ballagas, R., Borchers, J., Rohs, M., Sheridan, J.G.: The smart phone: A ubiquitous input device. IEEE Pervasive Computing **5** (2006) 70
13. Shen, H., Coughlan, J.: Reading lcd/led displays with a camera cell phone. In: CVPRW '06: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop, Washington, DC, USA, IEEE Computer Society (2006) 119
14. Tanaka, Y., et. al.: Indoor visible light transmission system utilizing white led lights. In: IEICE Transactions on Communications, vol.E86-B, no.8. (2003) 24402454
15. Komine, T., Nakagawa, M.: Performance evaluation of visible-light wireless communication system using white led lightings. In: ISCC '04: Proceedings of the Ninth International Symposium on Computers and Communications 2004 Volume 2 (ISCC"04), Washington, DC, USA, IEEE Computer Society (2004) 258–263
16. Josefsson, S.: The Base16, Base32, and Base64 Data Encodings, RFC 3548, United States. (2003)
17. Heckbert, P.S.: Fundamentals of texture mapping and image warping. Technical report, University of California at Berkeley, Berkeley, CA, USA (1989)
18. Deutsch, P.: DEFLATE Compressed Data Format Specification version 1.3, RFC1951, United States. (1996)