Dynamic Barcodes for a Better Ubiquitous Understanding

Geert Vanderhulst and Lieven Trappeniers
Alcatel-Lucent Bell Labs
Copernicuslaan 50, 2018 Antwerpen, Belgium
{*geert.vanderhulst, lieven.trappeniers*}*@alcatel-lucent.com*

***Abstract*—Barcodes have survived the shift from paper to screen. Printed boarding passes and paper advertisements are substituted by mobile applications and digital billboards where barcodes remain key to exchange small amounts of information in a quick, uncomplicated way. The move to digital media opens up new opportunities for dynamic usage scenarios involving barcodes. In this paper, we explore how 2D barcodes can increase the awareness of machines in an analog world where information is primarily visualized for humans. Moreover, we exploit displays and cameras as virtual networks to transfer data using 2D barcode slideshows. Hence we address two problems conventional barcodes cope with: a constrained data capacity and a lack of semantics in the data it contains.**

I. Introduction

Mobile devices with an integrated camera and processing unit can act as 'all seeing eyes' trying to understand our surroundings and augment them with additional information. For example, Layar[1] projects information about buildings, nearby metro stations, etc on top of the environment as seen through the lens of a mobile phone. Next to computer vision algorithms that try to recognize real-world objects (e.g. Google Goggles[2] and Layar Vision), pieces of digital information encapsulated in the environment (e.g. 2D barcodes) can help a computing device to make sense of the data produced by an object. With the advent of augmented reality (AR) glasses, we can only imagine the role of meaningful bits and bytes embedded in everyday environments will become increasingly important. Consider for example a digital clock. The clock presents the time in a for humans understandable way, but for a computing device it is not apparent to read the time in the real world and grasp that the object in front of the lens is a clock. While object recognition algorithms can take a guess, there is no uniform way of visualizing information and hence no certainty that pixels are interpreted correctly. This situation can be resolved if we augment the clock with a *dynamic barcode* that encodes the time as a digital string and which is updated every minute in line with the hands or digits as depicted in figure 1. Now, each dimension (physical as well as digital) has its own representation such that the notion of time can be understood both by humans and machines.

[1]http://layar.com/
[2]http://www.google.com/mobile/goggles/

Figure 1. This clock presents the time in two formats – a dynamic barcode also allows a computing device to accurately read the current time.

2D barcodes are widely used, but they still face a number of problems that become more stringent in dynamic usage scenarios such as the clock example:

- **Limited capacity:** most barcodes link to remote data instead of encapsulating it. Although several efforts were made to increase the capacity of barcodes (e.g. color barcodes [5]), higher data loads result in denser image symbols and render it more difficult to scan a barcode.
- **Lack of semantics:** the data referred to or contained in a barcode has typically no associated semantics. It is up to the reader to detect the type of the payload data (e.g. a URL) and find out about the relationship between the digital data and the real-world object to which the barcode is attached.
- **Technology acceptance:** ubiquitous barcodes can only be successful if a majority of people are willing to scan them. Surveys still show increasing adoption numbers [1], but mainstream support for Near Field Communication (NFC) tags may quickly supplant barcodes as optical data vehicles.

II. Related Work

CyberCode [7] is one of the premier visual tagging systems based on 2D barcode technology that can read the ID of a tagged object and determine the object's 3D location in the environment. Other AR frameworks such as ARTag [3] and u-Photo [9] relate recognized patterns with remote object descriptions in the application logic – the physical markers do not carry any data. However, the main issue with 'empty' tags or tags that can only store abstract identifier numbers, is their tight connection to a specific application or lookup service for resolving these IDs. Color barcodes such as HCCB [5] provide a higher data capacity than black and white barcodes (e.g. QR codes [2]) and thus

978-1-4673-5077-8/13/$31.00 ©2013 IEEE

can store additional meta data, albeit still limited amounts. Some works further exploited 2D barcodes for wireless data exchange using LCD-camera pairs. In [4], time-multiplexing of 2D color barcodes is suggested to transmit larger amounts of data that do no fit on a single barcode. The authors address synchronization issues between displaying and recording barcodes and present a solution to improve the robustness of transfers; the data throughput is limited by the capture rate of the utilized mobile devices. PixNet [6] encodes information in the frequency domain using spatial OFDM and can achieve higher data rates and wider view angles at larger distances. In contrast with these works, we do not present a novel encoding mechanism for fast and robust transfers over display-camera links. Instead, we examine the bit rates that can be expected when time-multiplexing conventional QR codes of varying sizes, under the safe assumption that the frame rate and quality of embedded cameras keeps on improving.

Concise alternatives for well-established formal knowledge representation formats such as RDF include N3 and N-Triples, both having shorthand syntaxes. Furthermore, dedicated formats like Entity Notation [10] were suggested to efficiently encode semantically annotated messages on resource-constrained sensor nodes. Although the aforementioned data formats have a relatively small footprint, they do not scale well to barcodes for a number of reasons. First, barcodes do not specify a content type or file extension indicating that the payload message is e.g. encoded as N3 triples. To decode the message, an application must know the content type in advance or try to figure it out. Second, by encapsulating a link to a resource, the size constraints of a barcode are bypassed and the content type is typically indicated in an HTTP header. However, in this case both the receiver and the sender need a permanent network connection, to acquire data and publish live state updates. We propose a data notation that is optimized for (animated) barcodes and inspired on 'pretty AJAX URLs', a de facto standard introduced by Google[3] for making AJAX applications crawlable.

III. Split and Play

Although we witness a boost in the data capacity of barcode formats, high-density 2D barcodes are still hard – if not impossible – to scan in practice with contemporary mobile devices. Several factors have a negative impact on the scanning effectiveness, such as poor lighting conditions, perspective distorts and rotations of the geometry of a symbol as a result of the user's angle and distance w.r.t. the barcode, blurred images or images of poor quality, and the color balancing approach of the used camera in case of color barcodes [8], [5]. Barcodes with a lower data density are easier to scan in general and more invariant to poor scanning conditions. Therefore, two lower-capacity barcodes might be preferred over one high-density barcode in favor of readability, yet to convey the same message.

A. Concatenated Barcodes

A straight-forward approach to overcome size constraints of a medium is to split up a message across multiple instances. We designed a custom barcode-independent header (inspired on the 'structured append' feature supported by QR codes) which is prepended to the payload of a QR code and supports up to 256 parts. The header defines three 8-bit fields – index, amount, parity – and is used in combination with the 8-bit byte mode of QR codes which encodes data at a density of 8 bits per character. *Index* refers to the position of a symbol within a sequence of QR codes, *amount* indicates the total number of symbols in the sequence and the *parity* value enables a reader to verify that all symbols read are part of the same message. The parity data is obtained by XORing byte by byte the ASCII values of the original input data. Figure 2 depicts a QR code of version 6 whose data can be equally represented by multiple smaller symbols that have a header prepended to the split data.

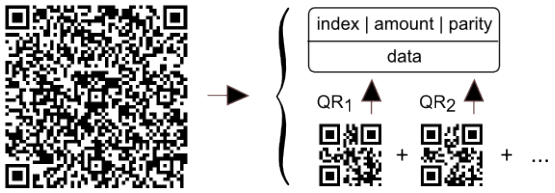


Figure 2. A long message is split over multiple barcodes and a custom header is added to each barcode to overcome the limitations of a QR code's structured append mode.

B. Animated Barcodes

With the term 'animated barcodes' we refer to a series of concatenated barcodes which are displayed one after one, like a video. Animated barcodes can be used to transfer data from anywhere on a compatible display – i.e. LCD/LED/-plasma panels [4] – to a device with a commodity camera, without setting up a network connection. This is particularly useful for downloading information from large surfaces such as an interactive shopping window: a large semi-transparent screen where users can interact with and retrieve information about promotions. Local deals and personalized items of interest can be represented in files of a few kilobytes containing user preferences like size and color, a promotion code, a thumbnail and an URL that points to generic information about the item. This data is then encoded using multi-part barcodes which are displayed as a slideshow after tapping a 'play' button. Figure 3 illustrates this concept. It shows a place holder for an animated barcode which is indicated via a special marker. To download the data, a mobile device

[3]https://developers.google.com/webmasters/ajax-crawling/docs/specification

should focus its camera on the marker and capture QR codes that appear in the area when the transfer is initiated. Obviously, the lens should remain pointed and focused at the barcode area on the screen during the transfer and the frame rate of the receiver should top the frame rate of the sender.

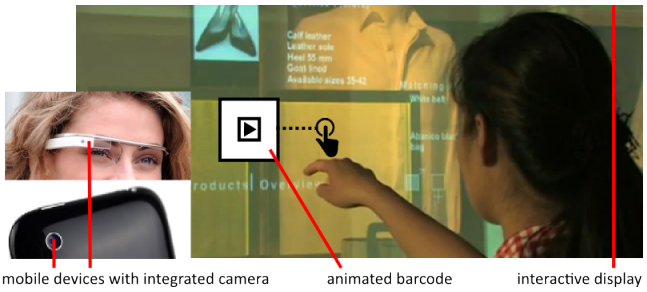


Figure 3. An interactive shopping window can play a slideshow of linked 2D barcodes anywhere on its surface, enabling a mobile device to download lively composed data through its camera.

IV. Adding Semantics to Barcodes

Next to size constraints, barcodes suffer from limited or no data semantics. Barcode headers differ amongst barcode formats and contain only minimal information to decode their payload which can be any kind of raw data. It is up to the reader to detect if the payload corresponds to a number, an address, an URL, etc. Reconsider the clock shown in figure 1 that displays the time in two formats simultaneously. The user immediately understands that 9:24 refers to the current time as she knows that the object in front of her is a clock, but a computer will need to analyze the time string and assume it is probably a time value based on the ':' and 'am' characters. Another use case of dynamic barcodes in a city context is to augment public transport with digital trajectories. If a bus or train can visually explain the line it serves to a computing device, a navigation service can pick up this information and notify a user waiting at a bus stop which bus to embark and when to get off.

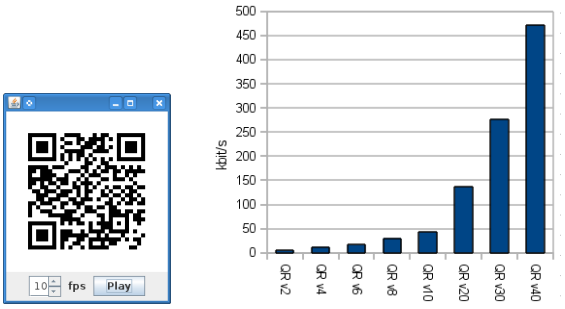
To support barcodes with varying data capacities, we present a compact notation that can be used to (i) identify objects in a uniform way and (ii) retrieve data directly from an object. We opted for an URL-based format to ensure backward compatibility with existing applications for decoding barcodes that are unaware of the extra semantics. As introduced in section II, our approach is based on 'pretty AJAX URLs'. A pretty AJAX URL such as http://www.example.com/ajax.html#!key=value consists of an hash fragment (that is, everything after the # sign in the URL) and has to begin with an exclamation mark. Hash fragments are never sent to the server as part of an HTTP request (by specification), unless the hash sign is escaped as follows: http://www.example.com/ajax.html_escaped_fragment_key=value. The client can then request the regular page using the pretty URL (which is typically populated client-side using JavaScript code) or a server-side generated HTML snapshot of the page using the ugly URL. We exploit this technique to create *literal URLs* which combine type information and instance data in a single URL. For instance, consider an ontological concept (bc:Bus) with a number of properties (bc:line, bc:destination, bc:schedule) describing a generic bus that is operated by a particular company. Each bus owned by the company, is an anonymous instance of the bc:Bus concept whose trajectory and schedule are identified by a line number and a destination property. This information is summarized as follows in a literal URL: http://buscompany.com/Bus#!line=45&destination=Boston. The URL identifies a specific bus serving line 45 that is headed for Boston.

When a request for the (pretty) URL is made, the hash fragment is omitted by specification and thus the bc:Bus resource is retrieved. Unless the client explicitly asks for a semantic content representation, the server returns a dynamic HTML page that represents a generic bus and which executes a client-side JavaScript function that injects instance data derived from the hash fragment in the HTML code. That is, the line and destination in this example which can be retrieved from the window.location.hash variable in JavaScript. A smart client can also use HTTP content negotiation to fetch the bc:Bus in N3 notation and learn about its properties. From then on, the acquired knowledge can be transferred to similar resources (i.e. other buses) such that a computing device will be able to understand that http://buscompany.com/Bus/#line=22&destination=Pittsburgh is another bus heading for Pittsburgh, just by interpreting the URL (i.e. without dereferencing it). By requesting the ugly version of the URL, the server is asked to provide details about a specific bus, i.e. the bus operating line 45. The server transforms the data contained in the escaped hash fragment into the requested semantic format (e.g. N3 or RDF/XML) and aggregates it with any other information it can associate with this instance. This approach is very similar to the way humans reason about their environment: the type of object shapes the context for the data it produces. Since an instance can have multiple types, an object can be described using several literal URLs which summarize a different granularity of detail. For instance, on a more general level a bus can be described as a vehicle eligable to transport 54 passengers.

V. Implementation and Evaluation

In section III-B we introduced animated barcodes as a means to transfer data between two devices: a sender displays a stream of barcodes over time and a receiver captures this stream using a camera. Since the receiver must remain steady to keep its lens focused during the data transfer, transfer times should be minimized. The data rate is dictated by the size of individual frames and the frequency

by which they are displayed and captured. We conducted two preliminary experiments to identify suitable frame sizes and rates. First, we measured the cost of encoding and decoding QR codes of varying sizes, ranging from a QR code of version 1 with a maximum binary data capacity of 17 bytes to a QR code of version 40 with a maximum binary data capacity of 2953 bytes with error correction level L. To this end, we used the popular Zebra Crossing (ZXing) barcode image processing library for Java[4] and produced QR codes of 200 by 200 pixels in JPEG format. Second, we addressed the issue of flow control. The sender is unaware at which frequency it should display frames in order that the receiver can capture them in time. Data will get lost if the sender (e.g. broadcasting at 10 fps) outruns the receiver (e.g. capturing at 5 fps). To overcome this, we programmed the sender to loop through the sequence of QR codes, each time at half of the frame rate until a minimal frame rate is reached. We experimented with random data transfers using different simulated frame rates (from 1 to 20) and frame sizes corresponding to the capacity of different QR versions minus 3 bytes for the multi-part headers as discussed in section III-A). We were able to handle transfers of version 1 QR codes at 3 fps (336 bit/s) up to transfers of version 40 QR codes at 20 fps (472 kbit/s). Figure 4 (b) shows the achieved bit rates at 20 fps. When using QR codes of version 20 and up at 20 fps, benchmarks approximate the bit rates of NFC which range from 106 kbit/s to 424 kbit/s. If barcodes can be captured in decent quality at frame rates of 20 fps and more, we believe optical data transfers might become a viable alternative for low-speed radio communications and give rise to a new interaction experience.



(a) (b)

Figure 4. An animated barcode player (a) loops over the message parts and gradually decreases its frame rate to accommodate both fast and slow receivers. The graph (b) shows the bit rate for various QR code versions when sending (displaying) at 20 fps.

VI. Conclusions

Starting from the perception that a computing device can hardly make sense of information that is intended for humans and embedded in the real world, we explored the use of dynamic 2D barcodes to exchange meaningful data between devices. On the one hand, we addressed the size constraints of QR codes by splitting up data across multiple symbols and (re)playing the data as a video of barcode frames. We concluded that optical transfers can be achieved at bit rates comparable to those of NFC. However, real-world experiments with a variety of mobile devices are needed to verify results in practice. On the other hand, we presented a concise semantic data format based on URLs that is tailored to compact data carriers such as barcodes. Literal URLs aggregate pre-published and live semantic data and hence support AR applications that act upon data rather than being driven by ad-hoc use cases.

When the combination of these technologies is further refined for practical use in the real world, digital information can be seamlessly embedded on displays and understood by a computing device by simply looking at it through a camera.

[4]http://code.google.com/p/zxing/

References

[1] Jonathan Chyou, Hao-Chun Kang, and Bill Cheng. Acceptance of QR code in Taiwan: an Extension of the Technology Acceptance Model. In *16th Pacific Asia Conference on Information Systems*, 2012.

[2] Denso-Wave. QR Code Standardization (ISO/IEC18004). http://www.qrcode.com/en/qrstandard.html.

[3] Mark Fiala. ARTag, a Fiducial Marker System Using Digital Techniques. In *Computer Vision and Pattern Recognition*, pages 590–596. IEEE Computer Society, 2005.

[4] Tobias Langlotz and Oliver Bimber. Unsynchronized 4D Barcodes: Coding and Decoding Time-Multiplexed 2D Colorcodes. In *3rd Int. Conference on Advances in Visual Computing*, pages 363–374, 2007.

[5] Devi Parikh and Gavin Jancke. Localization and Segmentation of a 2D High Capacity Color Barcode. In *Workshop on Applications of Computer Vision*, pages 1–6, 2008.

[6] Samuel David Perli, Nabeel Ahmed, and Dina Katabi. PixNet: Interference-free Wireless Links using LCD-camera Pairs. In *16th Int. Conference on Mobile Computing and Networking*, pages 137–148, 2010.

[7] Jun Rekimoto and Yuji Ayatsuka. CyberCode: Designing Augmented Reality Environments with Visual Tags. In *Designing Augmented Reality Environments*, pages 1–10, 2000.

[8] Constantin Scheuermann, Martin Werner, Moritz Kessel, Claudia Linnhoff-Popien, and Stephan A. W. Verclas. Evaluation of Barcode Decoding Performance using ZXING Library. In *Second Workshop on Smart Mobile Applications*, 2012.

[9] Genta Suzuki, Shun Aoki, Takeshi Iwamoto, Daisuke Maruyama, Takuya Koda, Naohiko Kohtake, Kazunori Takashio, and Hideyuki Tokuda. u-Photo: Interacting with Pervasive Services Using Digital Still Images. In *3rd Int. Conference on Pervasive Computing*, pages 190–207, 2005.

[10] Su X., Riekki J, and Haverinen J. Entity Notation - Enabling Knowledge Representations for Resource-Constrained Sensors. *Personal and Ubiquitous Comp.*, pages 1–16, 2011.