Data Transfer Using a Camera and a Three-Dimensional Code

Jeton Memeti[1], Flávio Santos[2], Martin Waldburger[1], Burkhard Stiller[1]
[1] Communication Systems Group, Department of Informatics, University of Zurich
[2] Institute of Informatics, Federal University of Rio Grande do Sul, Brazil
[jeton.memeti¦martin.waldburger¦burkhard.stiller]@uzh.ch, flavio.barata@gmail.com

Abstract

One- and two-dimensional barcodes have become very popular in past years and are widely used to identify products as well as services. Recently, 2D barcodes, like QR codes, are also used to optically transfer a dedicated hyperlink. All 2D barcodes share one major limitation: the storage capacity. To overcome that, time can be introduced as a third dimension. Instead of one, a sequence of barcodes is used to transfer a larger amount of data. The main goal of the present work was to design, implement, and especially evaluate the entire workflow for such a QR code-based, three-dimensional (3D) transmission system on the receiver side, such as a SmartPhone, being able to "read" from a sender, represented by a screen. The steps comprise the capturing of the 3D barcode, the recognition and reading of the sequence of 2D barcodes, and the final retrieval of the original content. Furthermore, adversarial conditions had to be identified, tested, and documented in detail. The prototype achieves a theoretical throughput of 12,288 Bytes for 30 seconds transmission intervals, which results in approximately 3,280 bit/s. Future work may focus on increasing the throughput of the system as well as the transmission reliability by applying error detection and correction techniques.

1 Introduction and Motivation

Products and services are often identified by numeric codes. This is motivated by the need to quickly find and reference objects. In order to enable machines to read these identifiers, alternative methods were proposed to create codes using visual representation. Barcodes [1] are a successful example of codes that use visual representation. They use a one-dimensional code to represent numeric values.

In the last decade, two-dimensional codes started gaining popularity. The main motivation for their creation was the need for a more flexible representation, which allows storing also text and binary content. Thus, in order to extend the storage capacity of one-dimensional barcodes, the second dimension allows the use of a matrix barcode instead of aligned bars. The most popular example of 2D codes is known as QR (Quick Response) codes [2].

1.1 Motivation

The motivation of this thesis is to introduce a third dimension for circumventing the capacity limitation of 2D codes. Thus, a digital display can show a sequence of two-dimensional codes (through an animation), which uses the time as a third dimension. Since most of previous codes use black and white patterns to represent information, extended models can also consider the use of multiple colors to increase their storage capacity. Hence the investigation area is about how to represent animated 2D codes and which algorithms can be applied in their recognition.

Figure 1 depicts the workflow from content encoding to decoding. The sender must be able to reproduce, through a digital display, the sequence of 2D codes. The receiver must be equipped with a digital camera and a software to decode the animation. In the rest of this thesis, encoder defines the algorithm that reads a content (a) and generates an animated 2D code (b), the recognizer the algorithm that reads a sequence of frames (captured by the digital camera (c)) and generates a sequence of 2D matrices (d), and the decoder the algorithm that reads the sequence of 2D matrices and outputs the original content (e).

1.2 Thesis Goals

The work presented in this thesis is driven by the following key goals:

- Make the workflow work.
- Make it work fast.

These two overall goals can be divided into the following three more specific goals: (a) definition of the format of the 2D matrices, (b) implementation of the recognizer and the decoder, and (c) evaluation under adversarial conditions.

1.3 Outline

The next sections address the aforementioned goals. Section 2 shows related work with regard to 3D barcodes. Section 3 depicts the implemented research method which has been used to split the complexity of the overall system into smaller, manageable parts. The design choices and decisions concerning the first two specific thesis goals are covered in Section 4. The third specific thesis goal, namely the evaluation, is covered in Section 5.

To appear in" Praxis der Informationsverarbeitung und Kommunikation", Vol. 36, No. 1, pp. 31-37, February 2013.

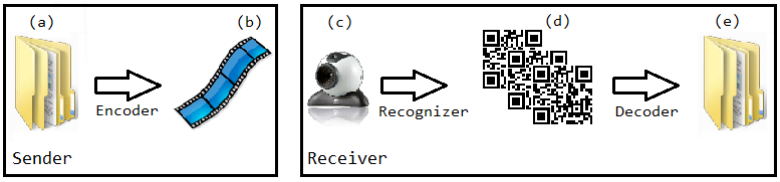

Figure 1: Workflow from content encoding to decoding

Finally, Chapter 6 summarizes the key findings and draws a conclusion. It also shows open issues concerning future work.

2 Related Work

In 2007 Langlotz and Bimber proposed a 3D barcode that uses the time as a third dimension to transmit an animated sequence of 2D barcodes. They called their approach *Unsynchronized 4D Barcodes* [3], which considered the color as an additional dimension. Using red, green and blue channels, they were able to encode three different 2D barcodes simultaneously into each frame of a displayed sequence. Their goal was to increase the robustness of the system by adding redundancy instead of increasing the transfer rate. Displaying every 2D barcode during three frames increases the recognition probability and is less fault-prone [3]. It is important to mention that this is the only 3D barcode approach so far.

This approach has unfortunately a low impact to this thesis. First of all, they used DataMatrix barcodes which prescribes the format and the decoder. This delimits the space of improvement concerning the throughput maximization. Second, the hardware has dramatically changed in the last five years, especially in the mobile phone sector. Therefore, the new hardware allows to research in other dimensions, e.g. increasing the storage capacity of a matrix by using a higher camera resolution. Nevertheless, the approach by Langlotz and Bimber has shown, that the first key goal of this thesis, namely to make the workflow work, is feasible.

3 Research Method

As mentioned in Section 1.2, the key goals for this thesis are on a high level view to (a) make the workflow work, and (b) make the workflow work as fast as possible.

Given that Langlotz and Bimber already implemented such an application for mobile phones in 2007 (Section 2), they have shown that it can be done, therefore, the goal (a) is not a research question anymore in principle. Hence, this thesis will focus on the second problem, i.e. the question of how high the maximum achievable transfer rate is. Nevertheless, reliability is still an issue as a sort of left-over from the first problem and has to be considered too.

The first step in answering the question of the maximum transfer rate is to apply an overall approach. In order to make the whole workflow as fast as possible, each sub problem needs to be analyzed and optimized. The complexity of the overall system needs to be broken down into smaller parts due to simplicity reasons, e.g., recognize a 2D matrix, find the modules, and identify the colors. Therefore, the divide-and-conquer approach is an ideal solution for this problem. By analyzing the workflow in Figure 1, the following list of functional steps can be derived: (a) encode data in a 3D barcode, (b) show 3D barcode on a screen, (c) record 3D barcode for a while with the camera of a SmartPhone, (d) find relevant content in the recorder material, and (e) decode data from the recorded material.

These functional steps can be split into even smaller steps, i.e. atomic steps. Since the first two steps in the list above belong to the sender, only the functional steps on the receiver side are analyzed. Using the divide-and-conquer approach, the following atomic steps can be identified for the receiver: (a) launch the application and initialize the camera, (b) request auto focus and auto white balance from the camera (in order to avoid blurry or too dark/bright images), (c) record a 3D barcode from a screen, (d) save the recorded data to the file system, (e) open the saved (or recorded if saving is not necessary) data in order to process it, (f) recognize the 2D matrices, and (g) decode the 2D matrices.

All these atomic steps are meaningful for the environment of a modern SmartPhone. The next section will shed light on the specific problems to be solved concerning these atomic steps and the range of options being analyzed and depicted.

4 Design Choices

This section addresses the problems described in the list of atomic steps in Section 3. It shows the range of options which have to be analyzed and which option was finally depicted to solve the underlying problem, that is, an atomic step. Section 4.1 addresses one of the thesis goals (Section 1.2),

i.e. the format of the 2D matrices. The core parts of this thesis are the recognizer and decoder module, explained in Section 4.2 and 4.3, respectively. Section 4.4 handles the different ways available to obtain data from the mobile phone's camera in order to record a 3D barcode from a screen.

4.1 Barcode Format

The analysis of the most common 1D and 2D barcodes revealed important characteristics for barcodes to be considered. These characteristics are: (a) quiet zone around the symbol or barcode, (b) an appropriate finder pattern, (c) orientation marks to compensate rotation, (d) the shape of the data modules within the barcode, and (e) the usage of colors in data modules.

The barcode format to be used for the matrices in this thesis has to meet three requirements. First, it should provide fast recognition and decoding. Second, it should provide a high storage capacity. Finally, it should be robust against false positives. Taking into account the lessons learned from the analysis of the existing one- and two-dimensional barcode formats, as well as the requirements specified above, a new barcode format has been designed. Figure 2 illustrates this barcode format.

In short, this barcode format has the following characteristics: (a) three black and two white bars alternating on each side of the barcode form the finder pattern, (b) one additional white bar with the double size of the bars in the finder pattern around the symbol represents the quiet zone, (c) square data modules (= shape), (d) use of 8 different colors to encode a bit sequence into a data module (resulting in 3 bits per module), (e) no usage of orientation marks (rotation with less than 45° is therefore theoretically supported), and (f) a module on each corner used to indicate the offset information when the barcode is not filled with modules (indicated by the letters A, B, C, and D).

Furthermore, the barcode format prescribes 4,096 modules (= 64 in width * 64 in height) for each 2D matrix. This is the largest amount of modules, which still can be decoded reliably by our prototype.

4.2 Recognizer Module

After defining the format of the 2D matrices, the next step according to the workflow in Figure 1 is to design the recognizer module. The recognizer reads a sequence of frames (captured by the camera) and generates a sequence of 2D matrices. Therefore, it obviously depends on the format specification and has to know how the format looks like in order to recognize a barcode within a frame.

Based on the format shown in Figure 2, the main idea for the recognizer is to find two points on each of the four sides of the barcode. If there can be found two points on each side which correspond to the finder pattern, then the actual frame contains a barcode and its exact location can be determined.

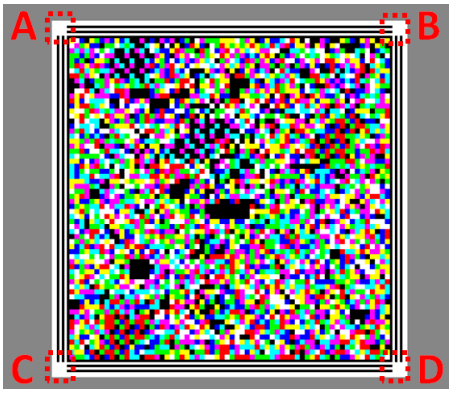

Figure 2: Barcode format designed and used for the prototype

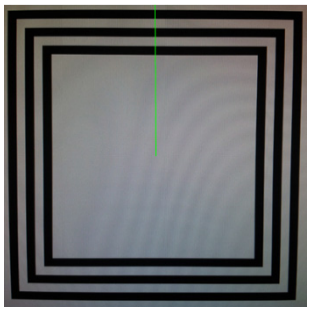

Figure 3: Color estimation algorithm

Once these points are found, one can draw a line through the two points on each side. The resulting four lines intersect in four points, namely the corner points which are important for the decoder module (see Section 4.3). These four lines also delimit the content of a barcode, so the decoder can process the payload.

Since the finder pattern consists of black and white bars, the recognizer module has to distinguish these two colors. The first approach was to use a fixed threshold for the color distinction, i.e., counting up the values of each single RGB value of a pixel and calculating the average and decide based on the static threshold if the analyzed pixel is black or white. The processing of pictures – or in this case matrices of a 3D barcode – captured with a digital camera is not so straight-forward. White areas often appear as grey and there are even other interfering color effects. The brightness of the picture also plays an important role when using fixed thresholds.

To overcome these difficulties another approach is needed. Instead of using a fixed threshold, the recognizer should estimate how dark (or bright respectively) a frame is and apply a dynamic threshold. This new algorithm starts with a small threshold, i.e. 20, the search for the finder pattern. Assuming that the barcode is at least 50% of the

frame's size, the recognizer algorithm hits the finder pattern if it analyses the pixels in the middle column (represented by the green line in Figure 3). In order to save time, the recognizer attempts to find only the top side of the barcode –the three other sides are ignored. Furthermore, it searches only in the upper 50% of the frame. This also requires that the barcode fills up at least 50% of the frame. If the recognizer cannot find the finder pattern with the given threshold, the threshold is increased by a value of 10. This is repeated until the finder pattern was found or the threshold exceeds the value of 160. Beyond this value, the frame is not expected to contain a barcode.

The recognizer module has also to support rotated pictures (or frames). If the user rotates the mobile phone while recording the 3D barcode, the 2D matrices will be rotated. An early version of the recognizer module was fault-prone to this kind of effects. Figure 4 shows the adjusted recognizer algorithm for the top side of the barcode. The brown line indicates where the algorithm starts to find the finder pattern after the picture's brightness has been estimated. The algorithm then continues searching the border between the finder pattern – more precisely the third black bar – and the payload area, indicated with the green lines. Finding points – or in this context lines – which are farther away assures that the calculation of the edge points is more accurate. The yellow lines indicate where the algorithm stops. The last green lines are then taken into consideration to draw a line threw the top border. The yellow line on the left side shows why rotation has been a problem. This yellow line has obviously found the finder pattern, but the point belongs not to the top side. Comparing the length of the yellow line with that of the previous green one reveals the error. The algorithm proceeds then similarly for the other three sides of the matrix.

4.3 Decoder Module

Once a sequence of 2D matrices is recognized in a sequence of frames, the last step according to the workflow in Figure 1 is to read that sequence of 2D matrices and output the original content. The latter task is accomplished by the decoder module.

First, the decoder depends on the format specification. It has to know how much modules are in the width and height of the barcode, how many colors are used to encode information and which color represents which bit sequence. Furthermore, it has to know if it should read all modules or just a part of them, which is indicated by the offset information. Second, the decoder depends on the corner points of the 2D matrix calculated by the recognizer module.

For each module of the 2D matrix, the decoder has to solve the following two problems: (1) determine the position of a module and (2) read that module. These two sub problems are described in Subsection 4.3.1 and 4.3.2, respectively.

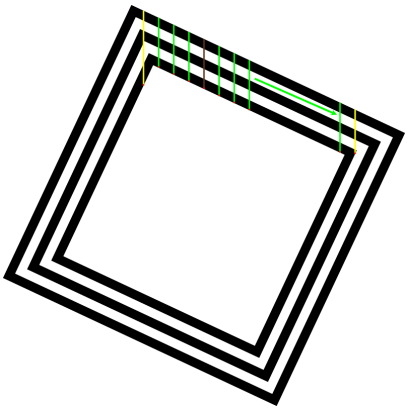

Figure 4: Finding two edge points on one side

4.3.1 Determine the Position of a Module

Since each module has to be read, the position of the modules has to be determined afore. 2D matrices created on and loaded from a PC rather than captured by a camera follow a linear rule. This means that each module of a 2D matrix has the same height and width. Rotated barcodes are more difficult to determine the modules positions. But simple trigonometric functions solve these problems. The problem of determining the position of a module arises, when dealing with 2D matrices captured by a camera. Besides rotation issues, there is another problem to be solved, namely distortion. Since it cannot be assumed that the user holds the mobile phone parallel to the sender, i.e. the screen, the captured barcodes might be (and in nearly all cases are) distorted.

To solve the distortion problem, a geometrical approach has been chosen. An interesting fact concerning distorted squares is that the diagonals intersect in the balance point. Drawing a line through the diagonal intersection and the horizontal vanishing point and another line through the diagonal intersection and the vertical vanishing point quarters the square exactly. In order to illustrate this approach a barcode with 16 modules has been created and captured with a digital camera. To keep the example clear, only the third black bar of the finder pattern has been drawn. The offset information has been neglected as well. Furthermore, the barcode has been captured from an extreme angel in order to have the vanishing points not too far away. Figure 5 illustrates this idea. These points A, B, C, and D are again the corner points of the barcode determined by the

recognizer module. Once the decoder algorithm receives these four points, it can proceed determining the modules. The intersection of the line through $\overline{AB}$ and the one through $\overline{CD}$ is the horizontal vanishing point, indicated with the letter H. The vertical vanishing point V is the intersection of the lines through $\overline{AC}$ and $\overline{BD}$, respectively. These two vanishing points are crucial for the proceeding of the algorithm and are used for the next steps. The next task of the decoder algorithm is to calculate the balance point. As mentioned before, the balance point is the intersection of the diagonals. Therefore, the intersection of the lines through $\overline{AD}$ and $\overline{BC}$ returns the point E. By drawing a line through $\overline{VE}$ and another one through $\overline{HE}$ the barcode gets quartered, this means there are now four smaller matrices. Using this technique recursively one can determine the edge points for each module in a 2D matrix. An advantage of this approach is that rotation is handled and solved implicitly.

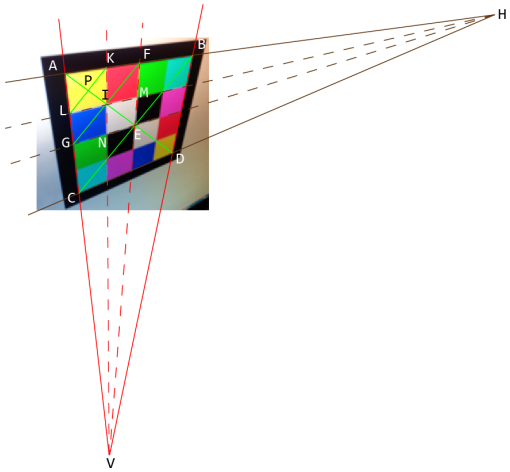

Figure 5: Determining the position of modules in distorted barcodes

4.3.2 Read a Module

Each module determined in the manner described above needs to be read in order to output the original content of the decoded data. To avoid keeping to much information in the memory of the mobile phone each module is subsequently read after its position is determined. Based on the approach with the vanishing points described above, the decoder algorithm uses a similar approach to read a given module. Figure 5 illustrates how the algorithm proceeds to read the top left module – the yellow module A, K, L, I – in the 2D matrix. The two lines through $\overline{AI}$ and $\overline{KL}$ intersect in the point P. This is again the balance point of the given quadrangle. Once the balance point of a given module is calculated, one can apply a matrix to read an amount of pixels. The more pixels of a module are read, the higher is the chance to determine its color correctly. Using a reading matrix which is too big increases however the chance to cross the modules border and to start reading pixels of an adjacent module. Based on characteristics like the picture resolution of the mobile phone, the supported distance between sender and receiver, and the number of modules within a 2D matrix, a 7*3 pixels matrix is used to read the pixels in a module in order to determine its color. The usage of such a flat reading matrix insures that interference lines as observed in pictures captured from a LCD display are crossed. Test results which supported this reading matrix size can be found in [4].

The distinction between the 8 different colors used to encode information is another challenge when reading a module. Reference values for black and white help adjusting the brightness of the image and therefore allow using a dynamic threshold for each captured 2D matrix. Since the recognizer module reads black and white pixels when recognizing a 2D matrix, i.e., the black and white bars, these values can be used. After adjusting the color brightness of a read matrix, the values of each color channel can be compared to determined thresholds. This procedure determines the modules colors in a high accuracy. However, it has to be mentioned that the different thresholds were tested on only one screen (sender) and mobile phone (receiver), respectively. In order to specify whether these thresholds are generally valid or only for the hardware in place, further tests are inevitable.

4.4 Obtaining Camera Data

There are three different ways to obtain data from the built-in camera of an Android mobile phone, namely via: (a) preview frames, (b) image capturing, or (c) video recording. In order to achieve the highest possible throughput, each of this three approaches needs to be tested and experimented with. The main idea is to slice each of the three possible ways to obtain data from the camera in its atomic steps, to optimize them and to measure the output for a given time period. For these three tests the atomic steps of each approach were put together into a sequence of 30 seconds to make the throughputs comparable. To keep the test between the three approaches fair, each approach needs a barcode with an amount of modules related to its maximum resolution. The following gives only a summary of each approach. Detailed descriptions can be found in [4].

Putting all atomic steps needed for the preview frames approach into a sequence of 30 seconds revealed that 38 barcodes could be processed in

that time. Having 38 barcodes each with 64 modules and 3 bit per module this results in a throughput of 912 Byte per 30 seconds for the preview frames.
Based on the hardware and software (Samsung Galaxy SIII with Android 2.3.3) used for the test series, pictures have a much higher resolution than preview frames. On the other hand, it takes longer to take a picture and to process the other atomic steps required for this approach. Putting these atomic steps again into a sequence of 30 seconds, it could be calculated that the picture taking approach achieves a throughput of 12,288 bytes per 30 seconds.
The last approach, i.e. recording a video and grabbing the frames in order to decode a 3D barcode, has been hard to test. First of all, the Android API does not offer a functionality to grab frames from a video file (neither off-line nor while recording). Second, the hardware did not allow to record videos in full HD quality, which is a drawback for this approach. The throughput has nevertheless been theoretically calculated in order to see if this approach is auspicious and worth investigating in the direction to solve the problems mentioned. Resulting in a throughput of 7,296 Byte per 30 seconds it showed that the picture taking approach is the way to go or the prototype.

5 Evaluation

After having successfully implemented the algorithms described in Section 4, the prototype has been evaluated in order to fulfill the last thesis' goal, namely to show the boundaries of the prototype. The description of the test environment, the test cases and the results can be read in detail in [4]. Summarizing the test results, the following specify the limits of the prototype:

- Rotation is supported up to ± 30°.
- Over and under-exposed pictures are not a problem.
- Horizontal and vertical distortions are supported up to ± 15°.
- Light reflections are no problem when the sender is not too glossy.
- The receiver has not to be fixed for the prototype to work reliably. It can be held with the hands (the same as when taking pictures with a mobile phone).
- The distance to the screen should be as short as possible.

Since these interferences do not appear segregated but several at the same time, further tests are needed in order to specify the limits more accurate. In all likelihood these limits will have to be downgraded.

6 Summary and Conclusions

Section 1 specified two areas of work for this thesis, namely the design and implementation of the recognizer as well as decoder module. This has been formulated as the following key goals: (a) make the workflow work, and (b) make it work fast.
It is safe to say that the first goal has been achieved on the condition that a reliability of 100% could not be accomplished. Adversarial conditions like distortion, rotation, and light reflectance among others appear mostly all at the same time and have an impact on the decoded content. This is not fatal when using the prototype for streams. File transfer on the other hand is nearly impossible.
The maximum throughput achieved by this prototype is 12,288 Byte per 30 seconds, which means about 430 Byte/s. Assuming that the recognition and decoding is executed in parallel, the prototype can at most double this throughput, which would result in 860 Byte/s. Compared to speech (telephone quality) which requires a data rate of 64 kbit/s, this prototype could not be used to transfer some voice for example. Therefore, it is doubtful that 3D barcodes as implemented in this thesis will establish itself.
A glimmer of hope concerning the throughput is that Android 4.0 is putting an emphasis on taking images faster [5]. Since the bottleneck of the prototype is the capturing of images, this could help increase the throughput. In order to evaluate the thresholds used for the color distinction, further tests on different devices on the sender as well as on the receiver side are inevitable. The actual prototype uses the thresholds evaluated only on one display and digital camera. And finally, to address the reliability problem of the actual prototype, future versions could apply error detection and correction techniques.

Bibliography

[1] Global Standards 1, "GS1 Bar Code Verification for Linear Symbols," http://www.gs1.org/docs/barcodes/ GS1_Bar_Code_Verification.pdf, May 2009.

[2] Denso Wave, "Version and Maximum capacity table," http://www.denso-wave.com/qrcode/ index-e.html. Last accessed October 2011.

[3] T. Langlotz and O. Bimber, "Unsynchronized 4D Barcodes," *International Symposium on Visual Computing,* pp. 363-374, 2007.

[4] J. Memeti, "Data Transfer Using a Camera and a Three-Dimensional Code," http://www.merlin.uzh.ch/publication/ show/7197. April 2012.

[5] Android Developers, "Ice Cream Sandwich," http://developer.android.com/about/versions/android-4.0-highlights.html. Last accessed January 2013.