Mobile Phone Based Interaction with Everyday Products – On the Go

Robert Adelmann
ETH Zurich
Institute for Pervasive Computing
Clausiusstrase 59, 8092 Zurich, Switzerland
adelmann@inf.ethz.ch

Abstract

Linking the physical world with information and services is a powerful concept. Even though RFID technology is very promising, the widespread use of RFID tags on retail products remains unlikely for the next years. In contrast, barcodes are ubiquitous – printed on virtually all consumer items world-wide. Recently there is an increasing interest in applications based on the recognition of standard 1D barcodes using the built-in cameras of mobile phones. Since many of the potential applications are especially useful when being "on the go", e.g., while shopping, a simple and fast user interaction is essential. In this paper we want to present two contributions that enable such a simple and fast interaction with everyday products and their services: A system for the recognition of 1D barcodes on mobile phones that differs in two aspects from already existing solutions: the fast and robust real-time recognition of codes on the camera's video images as well as the live detection of additional code parameters, like the barcode's relative orientation to the mobile phone. Besides that, we present and suggest the usage of these additional parameters in order to ease and accelerate the user interaction process – for example with the help of orientation sensitive menus.

1. Introduction

Many information resources containing data related to retail products are available today. For example, commercial product databases like SINFOS[1], an increasing amount of free databases like Codecheck[2], WikiFood[3], or web services, like offered by Amazon. All provide the possibility to get information about a certain product, once its worldwide unique EAN13 (European Article Number)[4] number is known. The information accessible by these resources is often more extensive than the data printed on the product package itself, representing an added value to the consumers. For example, for food related products there is information available regarding contained allergens or genetically modified ingredients. Even though such data might be highly relevant for some user groups, getting fast and easy access to this information is not readily granted, especially when it is needed most: when one is "on the go", e.g., when one is standing in the supermarket and deciding what to buy.

With the increasing mobility of powerful computing systems, e.g., mobile phones or handheld PDAs, this gap between the user and product related data as well as services can be bridged. Since 1998, when Barrett and Maglio already described a system for attaching information to real-world objects [2], several research projects have continued to explore this concept of "bridging the gap", i.e., the automatic identification of individually tagged real-world products in order to quickly look up information or initiate a specific action [6][11].

There are different technologies for the required automatic recognition of real world objects with mobile phones available, including NFC[5]/RFID, 2D codes and others. The main reason for our current interest in the 1D barcode recognition is the fact that in contrast to RFID technology and also 2D codes, it allows already today for the development and implementation of product based consumer services. All required components have not only been developed, but are already widely available: Virtually every one of the billions of products worldwide carries an EAN13 barcode, camera equipped mobile phones are becoming the standard and huge amounts of product information is available online.

In the next paragraph, prototypical applications are used to present the benefits of our two contributions, the real-time recognition as well as the detection and usage of ad-

[1] See `www.sinfos.com`
[2] See `www.codecheck.ch`
[3] See `www.wikifood.lu`
[4] See `www.barcodeisland.com/ean13.phtm`

[5] Near Field Communication (NFC) is a new standard for mobile phones that allows them to both act as an RFID reader and be read by other RFID readers (see www.nfc-forum.org).

Figure 1. Screenshots of the allergy assistant demo

ditional code parameters, regarding a fast and easy user interaction. Afterwards, related work will be discussed and the section "system" will provide an overview of the specific recognition conditions on mobile phones as well as the algorithms developed in order to meet them.

2. Demo Applications

We picked three already implemented prototypical applications in order to illustrate the advantages of our contributions.

2.1. Allergy Assistant

The first one is the so called "allergy assistant". It allows a user to predefine a profile containing the substances she or he is allergic to. If the user is then holding the mobile phone in front of a certain product, e.g., in the supermarket, he will be presented a simple "This product is fine for you.", "Take care, this product contains substances you are allergic to!" or a "No information found." message (see figure 1). Even though the idea for this simple application is not new, it illustrates well the general idea behind the "on the go" concept: provide fast, easy and direct access to personalized information and services on site.

The only necessary keystroke by the user is the one to start the application. Due to the real-time recognition of codes, the required information will be directly presented in the form of transparent overlays once a product's barcode is in range, providing the user with a simple answer to his question "Is that product fine for me?". This way, a very direct and natural link between the real world object and the according information can be established. This approach also allows for the fast checking of different products in close succession.

Once a code has been recognized, the according product information will be requested from a connected server. (We used the server that comes as part of the BaToo[1] system). This is done in the background, without requiring the attention of the user. The according socket connection to the server is established at the application's startup. Once the connection has been established, getting the data from the server is usually very fast and takes only a few milliseconds, allowing for real-time information to the current product in range.

Compared to the traditional approach of scanning a product's code by taking an image, followed by an often longer recognition process and the final presentation of the results on separate screens or the mobile phone's browser, the real-time recognition provides several advantages – at least regarding less complex interactions with products like in the considered "on the go" scenario: It eases and accelerates the recognition process (fewer keystrokes, no exact aiming at the code is necessary), increases the recognition performance (15 images/second are examined instead of only a single one) and allows for a very direct association of objects and information (e.g. by using overlays). These properties correspond with two important "learned design lessons" in a recent study conducted by Eleanor et al. regarding the interaction with mobile phones and services deployed in the real world [4]:

- "Keep data-entry on the phone's keypad to a minimum, rely on tag-based interaction instead whenever possible."
- "When small amounts of dynamic information are associated with visual tags, use overlays to display this information on the phone's screen".

2.2. Orientation Sensitive User Interaction

Figure 2 shows screenshots of two prototypical applications that illustrate how the detected orientation information can be used in order to accelerate and simplify the user interaction. On top, the "orientation sensitive menu" application can be seen. This application allows the user to select different menu items by rotating the phone in relation to a product's barcode. The menu items might represent information or certain services offered for that product. Consider the following example scenario: The user has found an old music CD at home. When he is holding his mobile phone straight in relation to the CD's barcode, its current price (as available from Amazon) will be displayed. If he turns the phone to the right, he will select the option to add this CD to his electronic music collection. If the phone is turned to the left, there will be the option to sell this product by means of the eBay platform. All required information for such an auction, like a product's name, images, price and a description can be obtained using the web services offered by Amazon. In combination with the web services provided by eBay, an according auction can be automatically created. (The price information application as well the automatic eBay auctions have already been implemented.)

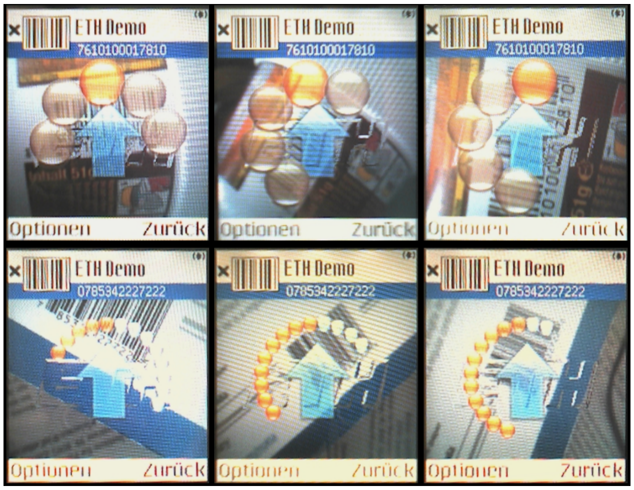

Figure 2. Screenshots of the orientation sensitive menu (on top) and slider (bottom)

The lower screenshots in figure 2 show the related ”orientation sensitive slider”, which can be used for example for the rapid rating of products. Further orientation sensitive applications like games [10] or the combination and enrichment of traditional user interface elements with orientation information are possible. These two example applications should only hint at the possibilities that the consideration of additional code parameters provides. It remains an open issue to examine if and under what circumstances they really ease and accelerate the user interaction process.

A precondition to such a kind of applications that is addressed by this paper is the very robust recognition of barcodes as well as their orientation under realistic conditions.

3. Related Work

Since 1D barcodes have been around for quite some time now,[6] a number of algorithms have already been implemented for their visual decoding on desktop computers.[7] Few specialized systems, for example in the logistics sector, also provide real-time recognition of codes. However, they often rely on specific conditions, including high-resolution, sharp images or constant lighting and their requirements in terms of system resources are much too demanding for an implementation on mobile phones, leave alone their execution in real-time on these devices.

Note that given the commercial potential of the 1D barcode recognition on mobile phones, it is not surprising that a number of commercial solutions exist. Scanbuy offers an SDK [8], which is capable of recognizing 1D barcodes. (Even though their own application for end users ”ScanBuyShopper” is missing this functionality and requires the manual input of the EAN13 numbers.) Applications for end users can be bought from Gavitec[9], ClickScan[10] and MediaStick[11], to name but a few. While informal trials with some programs from the above vendors showed a superior performance of our system, we explicitly abstained from conducting formal comparisons, as improving the recognition rate is not our primary goal. Instead we are focusing on the possibilities of the real-time recognition of codes as well as the live detection and usage of additional code parameters in order to simplify and accelerate the user interaction process.

Regarding the viewed solution, the one provided by Scanbuy comes closest to the notion of the real-time recognition, due to its auto-trigger functionality. This feature automatically detects if a code is in the image and, in case a candidate is found, starts the image capturing and recognition process. This process takes some time. However, without the full real-time recognition a fast scanning of consecutive products, the direct presentation of information using the overlay approach mentioned in the previous section, or the implementation of applications like the orientation sensitive menus is not possible. Besides that, the mentioned solutions differ in some other important aspects from our solution: The recognition of the codes is in some cases (ClickScan, partially Scanbuy) not performed on the mobile phone, but on a central server, which results in delays as well as costs when transmitting the images over the mobile phone network. Other applications exhibit a low recognition performance or require the special targeting of codes.

None of the currently available solutions for the recognition of 1D barcodes on mobile phones we know of, including freeware projects like BaToo[1], provides the real-time recognition of codes as well as the usage of additional code parameters for user interaction.

Even though Ohbuchi et al. [8] presented an algorithm capable of the real-time recognition of barcodes on a mobile phone, their solution differs in two important aspects from our system: The most important drawback of the system presented by them is the fact that it has been hand-tailored for one certain hardware device. It relies on access to a powerful but also very specific hardware element, which is not accessible by normal application developers: the built-in signal processor of a device. In contrast to that, our implementation has no hardware-specific requirements and is therefore able to run as a standard application on all Symbian-enabled mobile phones. Besides that, the algorithm proposed by Ohbuchi relies on two specific conditions

[6] The retail industry introduced EAN13 barcodes in the 1970s.
[7] See `www.charactell.com` or `www.axtel.com`

[8] See `www.scanbuy.com`
[9] See `www.mobiledigit.de`
[10] See `www.clickscan.eu`
[11] See `www.mediastick.co.jp`

regarding the barcode images that are in general not met in the "on the go" scenario: Prior to the code's decoding, the barcode's position is detected using a spiral scanning algorithm that runs on the device's signal processor. This algorithm makes the assumption that the point in the middle of the screen is located in the code. Like illustrated in section 4.1, this condition is easily violated in our considered scenario. Additionally, the spiral-scanning approach relies heavily on the clear detection of a single black bar based on the binarized image. Uneven lighting conditions, unclear camera images and especially the blurred images occurring during phone movements (see figure 3) render this approach too sensitive for the intended applications.

The detection and usage of additional code parameters, besides the encoded values, has been introduced by Michael Rohs for the 2D visual code symbology [9]. The developed system can't be transferred to 1D barcodes, since the visual code symbology has been specifically designed for camera phones and the detection of additional code parameters, featuring a special guiding bar and large elements that allow for a robust decoding even in very low resolution camera images. A big "drawback" regarding the usage of the system presented by Rohs remains the fact that in order to benefit from the advantages, the 2D codes first have to be printed and placed on objects, while 1D barcodes are already ubiquitous.

4. System

This section presents the algorithms used for the orientation detection as well as barcode recognition. The recognition is performed in two steps: First, a barcode's presence and orientation in the image is detected using an optimized algorithm based on the very robust Hough transform [5][3]. Once a code's orientation is known, its decoding is performed along several scan lines through the code.

In order to be able to motivate some design decisions and to illustrate the general challenges for the real-time barcode detection on mobile phones, first a brief overview of the recognition conditions in the "on the go" scenario will be given.

4.1. Recognition Conditions

The recognition of a barcode and its orientation in an image is quite simple, given we have a sharp code present on an empty white background. The situation on mobile phones differs substantially from such perfect conditions and is challenging due to the following points:

Given the still limited processing power of mobile phones, even an appropriate binarization[12] of a 640x480 pixels image, e.g. according to an adaptive thresholding algorithm presented in [12], at a reasonable frame rate is already too much for standard mobile phones. Often the code itself takes up only a small portion of the whole image and the background contains a lot of features that are not limited to certain well-defined elements (see figure 3, image 1) – like for example in a scenario where barcodes in text documents are recognized. Today, the recognition of barcodes on mobile phones requires either the use of macro lenses[13] or mobile phones with autofocus cameras [1]. Both alternatives easily result in unsharp images: When using a macro lens, if the optimal distance between the phone and camera has been left and in case of an autofocus device due to the slow re-focusing (image 2). In a realistic environment, the used algorithms also have to be able to cope with difficult lighting conditions as presented in image 3 and 6). Finally, another problem arises when considering applications such as orientation sensitive menus, that require the rotation of the phone or the product: codes in the images will be blurred due to the rotation (image 4) and will not always be located in the image's center because of hand movements (image 5). The following outlined algorithms are specifically designed to cope with these conditions.

Figure 3. Recognition conditions: 1. Small code, 2. Unsharp image, 3. Reflections, 4. Fuzziness due to rotation, 5. Code movements, 6. Difficult lightning conditions

4.2. Orientation Detection Algorithm

The orientation detection can be broken down into the following three steps, that are illustrated in figure 4:

1. Image Preprocessing: Starting point is a live image from the mobile phone's camera with a resolution of 640x480 pixels. This resolution has been found to be sufficient for the robust decoding of EAN13 barcodes [1]. In a

[12] The conversion of the original image into one containing only black and white pixels.

[13] A small plastic lens that ensures that close-up images are sharp. Sharp images of the barcode are necessary because the white and black bars of EAN13 codes can have four different widths that have to be distinguished reliably for robust decoding.

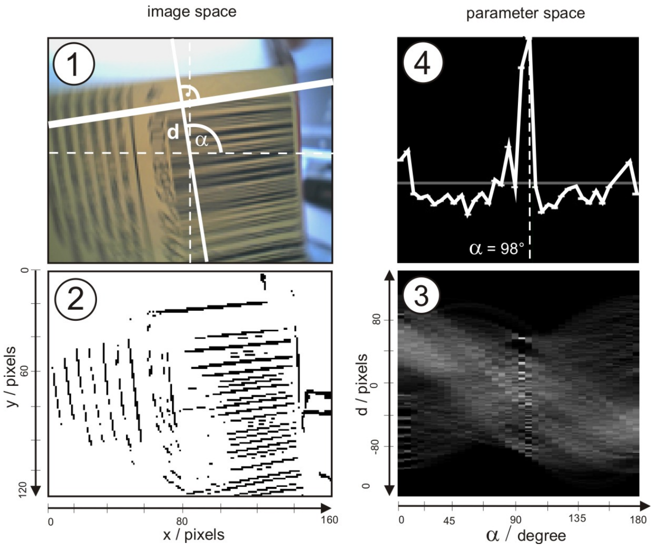

Figure 4. Orientation detection process: 1. Camera image 2. Fast edge detection 3. Parameter space of the Hough transform 4. Search for parallel lines

first step, a very fast edge detection algorithm based on simple threshold values is performed: A pixel in the resulting image will be black if the difference to its preceding pixel in x or y direction in the start image is higher than a predefined value. We use the standard RGB color space and determine the distance of two pixels by calculating the difference of their grey values, which are in turn computed as averages of the pixels' red, green and blue values. This is not optimal regarding different lighting conditions or unsharp images, but it can be performed very fast and the following Hough transform is more than robust enough to handle the imperfect results. Due to the limited computing resources, this edge detection is performed only on each 4th pixel of the original image, resulting in a 160x120 image containing the most distinct edges. Figure 4 shows the very blurry start image (image 1) as well as the resulting image (image 2).

2. Applying the Hough Transform: In a second step, the Hough transform is used in order to detect straight lines in image 2. The Hough transform[5][3] is a standard method in image recognition that allows the detection of analytically representable features in images, such as straight lines, circles or ellipses. The recognition of these global patterns in the image space is based on the identification of local patterns in a transformed parameter space. We use this method for the detection of the barcode's lines, since it is very robust regarding noise, partially hidden or incomplete lines, even if the image resolutions are very low (in our case 160x120 pixels).

$$d = x \cdot \cos(\alpha) + y \cdot \sin(\alpha) \quad (1)$$

For the detection of the lines, we use the line equation in its normal form (1), where *d* is the distance from the origin and α the angle with the normal. (Image 1 in figure 4 illustrates the association.) Using this formula, the Hough transform will in principle map all collinear points lying on the line described by equation 1 in the image space, to sinusoidal curves in the parameter space that intersect at the point (α,d).

The algorithm used to calculate the parameter space visible in image 3 is quite simple: All the pixels in image 2 are processed one by one. For each black pixel (x,y), we are going to obtain a set of (α,d) pairs, by using different values for α in equation 1. For each resulting point (α,d) the corresponding entry at that position in the parameter space is incremented by one unit. When all pixels are processed, the parameter space contains the number of collinear pixels for all the lines found in the image. The resulting parameter space for our example is displayed in image 3.

In order to be able to consider all possible lines in the image space, the entered values for α usually have to range from $0\,^\circ$ to $360\,^\circ$. We can apply a simple trick and save half the necessary computations by allowing negative values for d. This way we have to evaluate only α values from $0\,^\circ$ to $180\,^\circ$. The amount of evaluated angles (the entered values for α) for each pixel is variable and corresponds with the width of our parameter space. The more angles we use, the more accurate and robust the orientation detection becomes, but also the more time it takes. Increasing the amount of considered angles increases the time needed to calculate the parameter space, since these calculations are executed for each black pixel in the image space. On the Nokia N70 mobile phone we currently use 36 angles, evenly distributed from $0\,^\circ$ to $180\,^\circ$ at $5\,^\circ$ steps (the left diagram in figure 5 shows this uniform angle distribution). In order to increase the accuracy as well as robustness of the orientation detection, we change the distribution of angles if a barcode's orientation has already been recognized in previous images in such a way, that there is a higher angle density around the last detected orientation. For example, the right diagram in figure 5 shows the new angle distribution if the last detected orientation angle was $90\,^\circ$. This way we can achieve a $1\,^\circ$ accuracy and a higher robustness while maintaining the fast execution speed of the algorithm, since the total amount of considered angles (36) remains the same.

In order to accelerate the calculation of the parameter space further and therefore allow for an execution of the algorithm in real-time, several other means have been taken, including the following major one: We use an optimized integer based version of the Hough transform that abstains from using time consuming floating point operations and heavily relies on lookup tables. Such an integer based im-

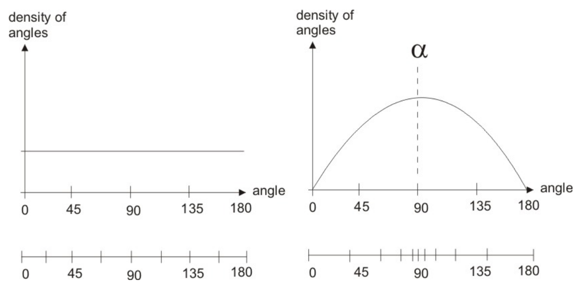

Figure 5. Adjustment of the angle distribution after an orientation has been found, in order to increase the algorithm's robustness and accuracy while maintaining overall speed

plementation accelerates the algorithm considerably, while Magli et al. [7] showed that the negative effects on the accuracy, compared to the floating point based version, are negligible. It is also no coincidence that we use the already described edge detection prior to the Hough transform instead of a simple binarization of the original image or a more "correct" edge detection that returns more edges. This way the amount of black pixels in the resulting image is much smaller and therefore the computations necessary during the Hough transform can again be reduced considerably.

3. Search for Barcode's Orientation: After the parameter space has been calculated, it is searched for a set of parallel lines belonging to the barcode. In order to do this very fast, we can use the handy fact that all maxima belonging to parallel lines in the image space will show up in the same column in the parameter space, since all have the same α parameter. For each column of the parameter space a value is calculated, basically by adding up the differences of adjacent entries in that column. Image 4 in figure 4 shows the resulting values of the different columns in our example. Determining the barcode's orientation is now reduced to the determination of the column that contains the highest value – in our example 98 °.

Using the parameter space, additional code parameters besides the orientation can determined. E.g., the position of a barcode's first and last bar or its tilt in relation to the mobile phone. These parameters can also be used for the user interaction. So far we have limited our application to the orientation detection, since the barcode's orientation can be determined very robust even under difficult conditions. (See figure 6 for some examples.)

4.3. Barcode Recognition Algorithm

We will mention the used decoding algorithm only briefly, since it is basically an extended and optimized version of the one already used in the BaToo[1] toolkit.

Figure 6. Screenshots of the orientation detection

Due to the strict timing constraints regarding a real-time recognition, the algorithm is scan line based. It basically works in four step: First, the image pixels along the scan line are binarized. Since we only have to consider the data along the scan lines, this can be done very fast, even if advanced algorithms are used. The applied binarization algorithm is based on a moving threshold method presented in [12] and can handle uneven image illuminations as well as slightly blurred images.

After binarization, the resulting set of black and white pixels is searched for the start and end delimiters[14] of the EAN13 code. If these have been found, the digits encoded in the sets of four bars will be determined by comparing the bar lengths to a lookup table, containing the digits encodings. In order to increase the robustness of the decoding process, not only perfect, but also close matches are considered. Finally, the high sensitivity of a scan line based approach to local image distortions is compensated for by the use of multiple scan lines and by the combination of the results of different scan lines in a majority voting fashion. False detections are reduced by weighting the digits returned by completely recognized scan lines higher than results of only partially recognized lines. Controlling a barcode's checksum digit provides an additional mean to reduce false detections.

Our current implementation allows for different decoder classes to be used in order to support the recognition of different types of barcodes. Currently we have implemented only support for EAN13 barcodes, since those are the most widely used product information codes today.

The described orientation detection algorithm as well as the barcode recognition algorithm have been implemented in C++ for Symbian and tested on several Nokia devices, including the Nokia N70, 6680 as well as the N90. Since no hardware specific functionality is used, the proposed algorithms and programs are executable on all Symbian devices. The implementation of a real-time recognition on J2ME devices is limited by the fact that it is currently in general not possible to get fast access to higher resolution images from a mobile phone's camera using the J2ME APIs. On the N70,

[14] A black-white-black series of bars of one unit width.

a Nokia Series 60 device featuring Symbian OS 8.1 that has been now already for two years on the market, the orientation detection in combination with the barcode recognition (20 scan lines) as well as several transparent overlays on the screen result in an average frame rate of 15 frames/second. This indicates that the proposed algorithms are well suited for the application in today's mobile phones.

5. Conclusion

Applications based on the recognition of barcodes using mobile phones are becoming increasingly attractive, given the fact that all required elements are already present. Since access to information and services to real world products will often be very useful when being "on the go", corresponding applications have to be very simple and fast to use. We presented a mobile phone application for standard mobile phones that is capable of the robust real-time recognition of EAN13 barcodes under realistic conditions as well as the very robust detection of the code's orientation relative to the mobile phone. Arising interaction possibilities have been hinted at using the allergy assistant demo as well as the orientation sensitive menus.

References

[1] R. Adelmann, M. Langheinrich, and C. Floerkemeier. A toolkit for barcode recognition and resolving on camera phones - jump starting the internet of things. Workshop Mobile and Embedded Interactive Systems (MEIS'06) at Informatik 2006, Oct. 2006.

[2] R. Barrett and P. P. Maglio. Informative things: how to attach information to the real world. In *UIST '98: Proceedings of the 11th annual ACM symposium on User interface software and technology*, pages 81–88, New York, NY, USA, 1998. ACM Press.

[3] R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, 1972.

[4] A. M. D. S. E. U. Eleanor Toye, Richard Sharp and A. Blackwell. Interacting with mobile services: an evaluation of camera-phones and visual tags. *Personal and Ubiquitous Computing*, Volume 11, Number 2 / February, 2007(2):97–106, February 2006.

[5] P. Hough. Methods and means for recognizing complex patterns. U.S. Patent 3,069,654, 1962.

[6] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, and M. Spasojevic. People, places, things: web presence for the real world. *Mobile Networks and Applications*, 7(5):365–376, 2002.

[7] E. Magli and G. Olmo. All-integer hough transform: performance evaluation. In *ICIP (3)*, pages 338–341, 2001.

[8] E. Ohbuchi, H. Hanaizumi, and L. A. Hock. Barcode readers using the camera device in mobile phones. In *CW*, pages 260–265. IEEE Computer Society, 2004.

[9] M. Rohs. Real-world interaction with camera-phones. In *2nd International Symposium on Ubiquitous Computing Systems (UCS 2004)*, pages 39–48, Tokyo, Japan, Nov. 2004.

[10] M. Rohs. Marker-Based Embodied Interaction for Handheld Augmented Reality Games. *Journal of Virtual Reality and Broadcasting*, 4(5), Mar. 2007.

[11] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison. Bridging physical and virtual worlds with electronic tags. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 370–377, New York, NY, USA, 1999. ACM Press.

[12] P. Wellner. Adaptive thresholding on the digitaldesk. EuroPARC Technical Report EPC-93-110, 1993., 1993.