NEURAL IMAGE RESTORATION FOR DECODING 1-D BARCODES USING COMMON CAMERA PHONES

A. Zamberletti, I. Gallo, M. Carullo and E. Binaghi
Università degli Studi dell'Insubria, via Ravasi 2, Varese, Italy

Keywords: Barcode recognition, Image restoration, Neural networks.

Abstract: The existing open-source libraries for 1-D barcodes recognition are not able to recognize the codes from images acquired using simple devices without autofocus or macro function. In this article we present an improvement of an existing algorithm for recognizing 1-D barcodes using camera phones with and without autofocus. The multilayer feedforward neural network based on backpropagation algorithm is used for image restoration in order to improve the selected algorithm. Performances of the proposed algorithm were compared with those obtained from available open-source libraries. The results show that our method makes possible the decoding of barcodes from images captured by mobile phones without autofocus.

1 INTRODUCTION

In recent years the growth of the mobile devices market has forced manufacturers to create ever more sophisticated devices. The increasing availability of camera phones, i.e. mobile phones with an integrated digital camera, has paved the way for a new generation of applications, offering to the end users an enhanced level of interactivity unthinkable a few years ago. Many applications become possible, e.g. an instant barcode-based identification of products for the online retrieval of product information. Such applications allow for example, the display of warnings for people with allergies, results of product tests or price comparisons in shopping situations (Wachenfeld et al., 2008). In Figure 1 an illustration of a typical application that make use of a barcode identification.

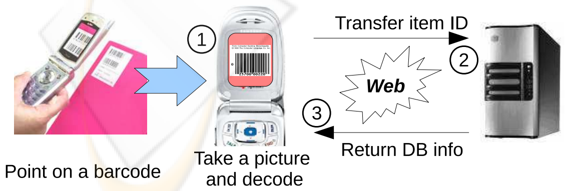


Figure 1: Graphical illustration of the process of a typical application that make use of a barcode identification.

There are many different barcode types that exist for many different purposes. We can split these into 1D and 2D barcodes. 1D barcodes are what most people think barcodes are: columns of varying width lines that are imprinted on the back of products. Within the 1D barcode we have EAN-13/UPC-A, Code 128, Code 39, EAN-8 etc. and today we know that billions of products carry EAN-13 bar codes. The two most important parameters influencing recognition accuracy on a mobile camera phone are *focus* and *image resolution*, with the former remaining the principal problem; instead low camera resolutions such as 640x480 pixels are not critical (Adelmann et al., 2006b). Figure 2 shows an example that highlights the difference between a barcode acquired with a device having autofocus (AF) and without AF. It is evident that images like that in Figure 2(b) present a high level of degradation that makes the decoding process very difficult or even worst, impossible.

Searching the Internet for camera phones with/without AF, we can estimate that about 90% of camera phones is without autofocus [1]. There are several libraries to decode 1-D barcode but if we analyze the most widespread of these available with an open-source license, all of them show serious difficulties in recognizing barcodes from images captured by devices without autofocus (see some results in Table 1).

Many studies have been made to develop applications for mobile devices capable to decode 1-D barcodes (Wachenfeld et al., 2008; Wojciechowski and Siek, 2008). Many studies have aimed to look for

[1]Based on data from http://www.shoppydoo.com

Zamberletti A., Gallo I., Carullo M. and Binaghi E. (2010).
NEURAL IMAGE RESTORATION FOR DECODING 1-D BARCODES USING COMMON CAMERA PHONES.
In *Proceedings of the International Conference on Computer Vision Theory and Applications*, pages 5-11
DOI: 10.5220/0002811600050011
Copyright © SciTePress

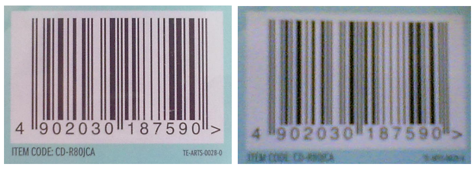

(a) (b)

Figure 2: A sample image captured by a device with autofocus (a) and without autofocus (b).

efficient and operative algorithms able to recognize a high percentage of codes in a limited time. Others have studied restoration techniques to improve the quality of codes acquired with sensors without autofocus or macro function, and the accuracy of the subsequent decoding (Simske et al., 2009).

There are several libraries available to decode the multitude of barcode standards. Only a few of these libraries are open-source. One prominent open-source library is the ZXing project[2]. It has the capability to read not just 1D barcodes but also 2D barcodes. Although this library is widely used and has a great support by the community, it has the common weakness to expect a camera with autofocus and a relatively high resolution in order to work properly. For this reason we decided to work on ZXing library to make it a viable solution when working with devices without autofocus.

In particular, in this work we experiment with a novel restoration technique based on neural networks, in order to improve the quality of the images and therefore the recognition accuracy of 1D barcodes. Image restoration is a process that attempts to reconstruct an image that has been degraded by blur and additive noise(Perry and Guan, 2000; Asmatullah et al., 2003). The image restoration is called blind image restoration when the degradation function is unknown. In the present work we perform blind image restoration using a back-propagation neural network and we show how the proposed restoration technique can increase the performance of the selected open-source tool in order to use it with all types of camera phones.

Table 1: Results obtained using some open-source libraries on datasets acquired from devices with and without AF.

	with autofocus		without autofocus	
Sw	Precision	Recall	Precision	Recall
ZXing	1.00	0.64	1.00	0.04
BaToo	0.95	0.58	0.60	0.13
JJil	/	0.00	/	0.00

[2]`http://code.google.com/p/zxing/` a Java multi-format 1D/2D barcode image processing library

2 IDENTIFICATION AND DECODING TECHNIQUES

The present work focuses on a restoration algorithm to improve the accuracy of a generic 1D barcode decoding process. However, to make the work self-contained, a brief overview on the state of the art in decoding barcodes is given here.

Algorithms for decoding barcodes from digital images, can be broken down into two steps: *identification* and *decoding*.

An identification algorithm receives in input an image and provides as output the image coordinates that identify the region containing the barcode. There are many different algorithms to perform such operations and in the following we summarize some of them. In (Wachenfeld et al., 2008) the authors presented an algorithm we named *Scanline Detection* which selects a pixel in the center of the image to be analyzed. Assuming that the pixel belongs to the barcode to be extracted, the algorithm make a horizontal expansion that ends when it finds the ends of the bar code. In (Adelmann et al., 2006a) the scanline detection algorithm is applied to multiple rows drawn regularly throughout the image. We name *Expansion Detection* another algorithm presented in (Ohbuchi et al., 2004) that performs vertical and horizontal expansion starting from the center pixel of the image to be analyzed. Other two interesting algorithms are those presented in (Youssef and Salem, 2007) and (Basaran et al., 2006) using *Hough Transform* and *Canny Edge Detection* respectively.

A barcode decoding algorithm receives in input an image and a sequence of coordinates and returns as output one or more strings containing the values recognized. Unlike identification, the decoding process is standard and is greatly simplified by the redundant structure that every barcode has. Some interesting decoding algorithms are briefly described below. The algorithm proposed by (Adelmann et al., 2006a; Chai and Hock, 2005) we named *Line Decoding*, reads a line of barcode and makes the decoding. The algorithm is able to understand if a code has been read correctly by analyzing the control code contained within the barcode. *Multi Line Decoding*, proposed in (Wachenfeld et al., 2008), is an extension of the Line Decoding algorithm, where the Line Decoding is applied at the same time to a set of parallel image rows. The code will be constructed collecting the digits that appear several times in all the lines analyzed. Finally, a very different approach is based on neural networks trained to recognize the codes. This algorithm, which we called *Neural Net Decoding*, was presented in (Liu et al., 1993).

3 THE PROPOSED METHOD

The neural restoration algorithm we propose in this paper was added in the ZXing library, a library that was proved robust decoding of 1D barcodes.

ZXing uses a modified version of the algorithm proposed by (Adelmann et al., 2006a) and mentioned in the previous section. The changes allow ZXing to identify 1D barcodes placed in a non-horizontal position, partially missing and placed in non-central position within the image portion. The ZXing's identification and decoding process is summarized in Algorithm 1. A special parameter *try_harder* can be enabled to increase the number of lines considered in the process and the number of rotations of the input image, to search barcodes placed in a non-horizontal position. This latest process is done by rotating the image and re-applying the Algorithm 1 until the code is not identified.

The decoding is based on the *Line Decoding* algorithm described in the previous section.

Algorithm 1 The ZXing's identification and decoding process.

```
Require: select a set of rows $H$ to be decoded
Require: select the number of rotations $R$ to be applied to the input image
 1: for all $r \in R$ do
 2:    for all $y \in H$ do
 3:       Select the image row $L_y$
 4:       Transform $L_y$ from RGB to gray levels
 5:       Apply to $L_y$ a high-boost filter (Gonzalez and Woods, 2001) with mask [-1 4 -1]
 6:       Apply an adaptive threshold to $L_y$
 7:       if decode($L_y$) is successful then
 8:          break loop
 9:       end if
10:    end for
11: end for
```

The barcode decoding process requires the image containing the code to be binarized. The image binarization is usually carried out using a thresholding algorithm. However there is an high chance that the image involved in the process is blurred and/or noisy, making the thresholding phase non-trivial. All the software tested in this work show their limits when faced with such images, with a high failure rate in the decoding process. The main contribution of this work is the definition and evaluation of a restoration technique that, complemented with an adaptive thresholding, can be proposed as an alternative to standard binarization. We base our strategy on the Multilayer Perceptron model trained with Backpropagation Momentum algorithm. The netwok has five input neurons, five output neurons and three hidden layers with two neurons in each layer. This configuration was chosen as it provides a high-speed in generalization combined with a high accuracy.

The network can learn how to restore degraded barcodes if trained with a proper amount of examples of the real system to be modeled. An example of degraded input image and its desired output is illustrated in Figure 3. The truth image (or output image) was created using a free online service to generate barcodes [3] and aligned to the input image using a computer vision algorithm called Scale-Invariant Feature Transform (SIFT) (Lowe, 1999) [4].

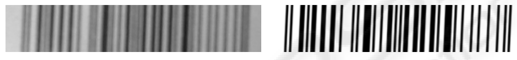


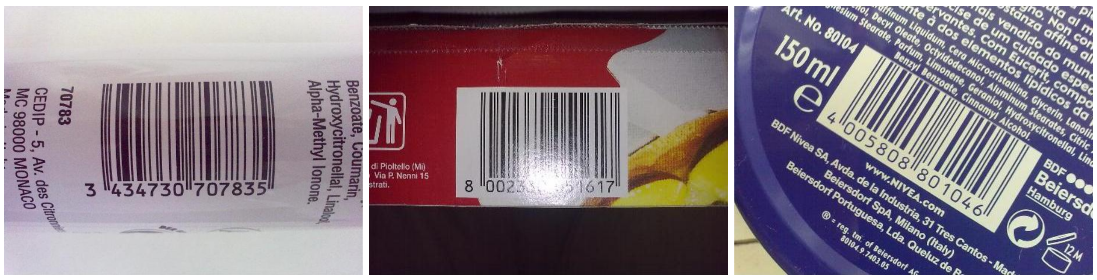
Figure 3: An example of training image captured by a device without autofocus (a), and its expected truth image (b). Only the rectangular section containing the barcode was extracted from the original image.

Training samples are presented to the neural network having the following form (P_{in}, P_{out}). The input pattern $P_{in} = \{L_y(x_i), \dots L_y(x_{i+S})\}$ is a sequence of S values, one for each input neuron, where $L_y(x_i)$ is the i^{th} pixel of the row L_y scaled in $[0,1]$. $P_{out} = \{L^o_y(x_i), \dots L^o_y(x_{i+S})\}$ is the expected output extracted from the truth image and then scaled in $[0,1]$, at the same position of the P_{in} pattern. Given a pair of training images having width W, we select only one line L_y and a number of patterns equal to $W - S + 1$, moving the input window $p = 1$ pixels forward for each new pattern. The training and test set creation was performed using degraded input images acquired by 1MP camera without autofocus at variable distances.

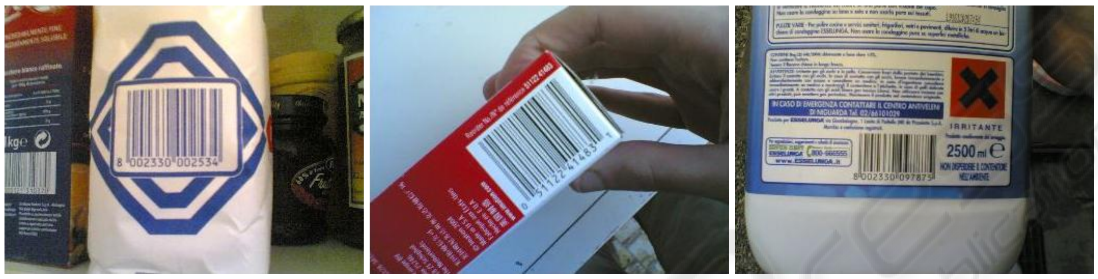
The trained neural network performs the restoration for never seen input patterns P_{in} transforming the input pixel values to gray levels without blur and noise. The algorithm is applied on each single line L_y selected by the identification algorithm. During the restoration phase, according to the step value p and size S of the input window, each pixel can be classified more than once. A decision rule must be accomplished to compute the final value of the restored image. In the present work for each pixel $L_y(x_i)$ the value of average output activation $\overline{o}_i = (\sum_{n=1}^{N_i} o_{i,n})/N_i$

[3] `http://www.terryburton.co.uk/barcodewriter/generator/` a free web-based online barcode generator

[4] We have used the plugin JavaSIFT (`http://fly.mpi-cbg.de/~saalfeld/javasift.html`) for ImageJ (`http://rsb.info.nih.gov/ij/`)



(a) Dataset1



(b) Dataset2

Figure 4: Three sample images of Dataset1 (a), and three of Dataset2 (b).

is calculated, where N_i is the number of times in which the pixel has been classified by the neural model and o_i is the activation of the i^{th} output neuron. Binarization process adopted for the restored images simply evaluate the average activation value $\overline{o}_i$ and sets a threshold at 0.5.

We named ZXing-MOD the library ZXing with the addition of our neural restoration process. The new algorithm is very similar to the original described in Algorithm 1, only the lines 5 and 6 have been replaced respectively by the neural restoration process and the binarization technique described above. The number of lines to be analyzed and decoded is deduced from a parameter called *rowStep*. This parameter specifies the number of image lines to be skipped between two subsequent scanning L_{y_i} and $L_{y_{i+1}}$.

4 EXPERIMENTS

The performance of ZXing-MOD was evaluated and compared with the original ZXing and with other two open-source libraries: BaToo[5] and JJil[6]. System performances are compared using precision P and recall R (Frakes and Baeza-Yates, 1992),

$$P = \frac{correct}{actual}, \quad R = \frac{correct}{possible} \tag{1}$$

where *correct* is the number of barcodes correctly recognized by the system, *actual* is the total number of barcodes recognized by the system, and *possible* is the total number of barcodes we expected from system. A Precision score of 1.0 means that every code recognized is correct, but says nothing about the number of codes that were not recognized correctly. A high Precision guarantees that there are few false positives. Whereas a Recall of 1.0 means that every barcode was correctly recognized, but says nothing about how many other barcodes were incorrectly recognized. In addition to precision and recall were also evaluated minimum and maximum execution time.

Analyzing the literature we realize that there is no dataset available to evaluate a barcode recognition system. For this reason we create two datasets of images, one with pictures of barcodes taken from devices with the AF function (Dataset1) and a second dataset with photos taken by devices without AF (Dataset2). The Dataset1 contains 215 color images all taken with a Nokia 5800 mobile phone, while Dataset2 contains 215 images all acquired by a Nokia 7610 mobile. The two datasets were acquired varying the rotation angle of the barcode, the distance between camera and object, the lighting conditions and the resolution.

The training set used to train the neural network is built considering 14 pairs of images as shown in Fig-

[5]`http://people.inf.ethz.ch/adelmanr/batoo/` Barcode Recognition Toolkit

[6]`http://code.google.com/p/jjil/` Jon's Java Imaging Library, for mobile image processing

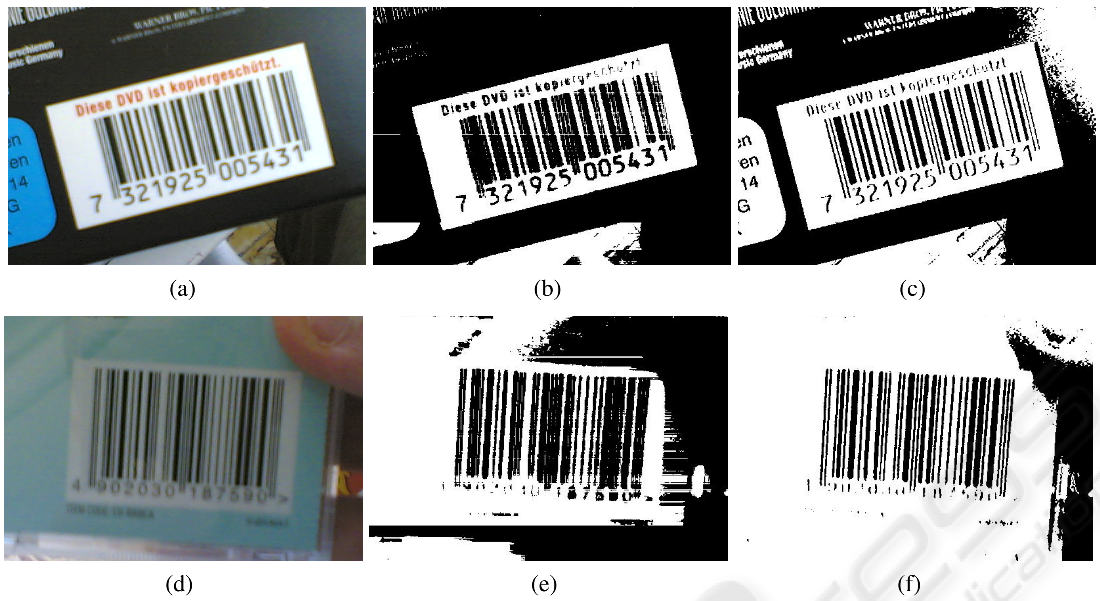


Figure 5: Two examples of barcode contained in blurred images captured with a device without AF (a)(d). Thresholding results obtained with the library ZXing (b)(e), and corresponding threshold obtained with ZXing-MOD (c)(f).

Table 2: Comparison results between the three tested algorithms and our strategy. The table shows the Precision, Recall and execution time computed on Dataset1.

				Time (ms)		
Library	Parameter	Precision	Recall	Min	Max	Average
ZXing		1.00	0.64	0.26	12.80	1.77
	try_harder	1.00	0.82	0.26	244.40	26.59
BaToo		0.95	0.58	1.80	29.64	13.49
JJil		/	0.00	219.26	966.70	470.73
ZXing-MOD	rowStep=1	1.00	0.87	1.74	2777.12	353.78
	rowStep=5	1.00	0.83	1.74	559.48	90.22
	rowStep=40	1.00	0.70	1.74	68.87	17.71

ure 3. All the input images have been cropped from photos belonging to Dataset2.

The two datasets and the training set used for our experiments are available online[7] allowing to compare our algorithm with other research works.

The quantitative results are shown in Tables 2 and 3. We must start by saying that the library Jiil, although much-quoted, did not recognize any code in the two datasets used. This could be caused by the fact that the library is currently under development and could therefore be in a state of low functionality. Conversely, libraries ZXing and BaToo showed good performances on Dataset1 but, as introduced earlier, collapsed on images of Dataset2. Analyzing the results obtained on Dataset1, we note that the parameter *try_harder* of ZXing increases significantly Recall although this advantage also leads to an increase in computation time. ZXing with the parameter *try_harder=true* becomes the best library to use with high resolution and not degraded images. The recall of ZXing-MOD is slightly lower than that of ZXing, this because the rotation of the analyzed image is not carried out and then some codes placed in a position not horizontal are not recognized. BaToo is efficiently fast even if the percentage of recognized images is rather low compared with results obtained with ZXing and ZXing-MOD.

[7]`http://www.dicom.uninsubria.it/arteLab/ricerca.html`

The results observed in tests carried out on Dataset2 show how our proposed neural restoration strategy makes ZXing-MOD the only viable solution that works on blurred and low resolution images. The *rowStep* parameter determines the number of lines to be analyzed, the greater the value of *rowStep* the lower the number of lines analyzed. ZXing-MOD maintains good results even when the value of *rowStep* increases, passing from a value equal to 1 to 5 the

Table 3: Comparison results between the three tested algorithms and our strategy. The table shows the Precision, Recall and execution time computed on Dataset2.

				Time (ms)		
Library	Parameter	Precision	Recall	Min	Max	Average
ZXing		1.00	0.04	0.24	12.15	1.87
	try_harder	1.00	0.09	0.24	106.16	58.41
BaToo		0.60	0.13	9.29	19.19	10.10
JJil		/	0.00	120.41	253.26	146.89
ZXing-MOD	rowStep=1	0.99	0.70	1.34	439.36	156.59
	rowStep=5	0.99	0.64	1.34	101.41	37.52
	rowStep=40	1.00	0.47	1.34	27.18	6.62

Recall drops of 0.06, and the execution time become four times lower than the initial one (see Table 3).

Tests were performed on a computer with the following configuration: Intel Core 2 Quad Q6600, 2GB RAM and the Windows XP Professional OS. Although the processor is multi-cored, all implemented software is single threaded.

A qualitative assessment was done by implementing a simple J2ME application and installing it on different camera phones. Evaluating the application on a Sony Ericsson v800 mobile phone reported the operativeness of the approach with a mean recognition time of 4 seconds.

5 CONCLUSIONS

In this paper, we proposed a general purpose solution to the problem of recognizing 1D barcodes from blurred images. This solution adopted is applicable to any decoding strategy and is based on supervised neural networks.

The algorithm has shown excellent experimental results and is therefore a valid alternative to standard methods that try to improve the quality of the images before decoding.

We implemented a version of ZXing-MOD in J2ME testing it on different types of phones with very encouraging results.

REFERENCES

Adelmann, R., Langheinrich, M., and Floerkemeier, C. (2006a). A toolkit for bar-code-recognition and -resolving on camera phones – jump starting the internet of things. In *Proceedings of the workshop on Mobile and Embedded Interactive Systems (MEIS'06) at Informatik 2006. GI Lecture Notes in Informatics Series (LNI)*, Dresden, Germany.

Adelmann, R., Langheinrich, M., and Floerkemeier, C. (2006b). Toolkit for bar code recognition and resolving on camera phones - jump starting the internet of things. In Hochberger, C. and Liskowsky, R., editors, *GI Jahrestagung (2)*, volume 94 of *LNI*, pages 366–373. GI.

Asmatullah, Mirza, A., and Khan, A. (2003). Blind image restoration using multilayer backpropagator. In *Multi Topic Conference, 2003. INMIC 2003. 7th International*, pages 55–58.

Basaran, E., zgr Uluay, and Erturk, S. (2006). Reading barcode using digital cameras through image processing. In *Proceedings of 5th International Symposium on Intelligent Manufacturing Systems*.

Chai, D. and Hock, F. (2005). Locating and decoding ean-13 barcodes from images captured by digital cameras. In *Information, Communications and Signal Processing, 2005 Fifth International Conference on*, pages 1595–1599.

Frakes, W. B. and Baeza-Yates, R. A., editors (1992). *Information Retrieval: Data Structures & Algorithms*. Prentice-Hall.

Gonzalez, R. C. and Woods, R. E. (2001). *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Liu, S.-J., Liao, H.-Y., Chen, L.-H., Tyan, H.-R., and Hsieh, J.-W. (1993). Camera-based bar code recognition system using neural net. In *Neural Networks, 1993. IJCNN '93-Nagoya. Proceedings of 1993 International Joint Conference on*, volume 2, pages 1301 – 1305.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2*, page 1150, Washington, DC, USA. IEEE Computer Society.

Ohbuchi, E., Hanaizumi, H., and Hock, L. A. (2004). Barcode readers using the camera device in mobile phones. In *Cyberworlds, 2004 International Conference on*, pages 260–265.

Perry, S. W. and Guan, L. (2000). Weight assignment for adaptive image restoration by neural networks. *IEEE Trans. on Neural Networks*, 11:156–170.

Simske, S. J., Sturgill, M., and Aronoff, J. S. (2009). Effect of copying and restoration on color barcode payload density. In Borghoff, U. M. and Chidlovskii, B., editors, *ACM Symposium on Document Engineering*, pages 127–130. ACM.

Wachenfeld, S., Terlunen, S., and Jiang, X. (2008). Robust recognition of 1-d barcodes using camera phones. In *ICPR 2008*, pages 1–4. IEEE.

Wojciechowski, A. and Siek, K. (2008). Barcode scanning from mobile-phone camera photos delivered via mms: Case study. In *ER Workshops*, volume 5232 of *Lecture Notes in Computer Science*, pages 218–227. Springer.

Youssef, S. M. and Salem, R. M. (2007). Automated barcode recognition for smart identification and inspection automation. *Expert Syst. Appl.*, 33(4):968–977.