Efficient 1D and 2D Barcode Detection Using Mathematical Morphology

Melinda Katona and László G. Nyúl

Department of Image Processing and Computer Graphics
University of Szeged
Árpád tér 2, H-6720 Szeged, Hungary
katona.melinda@stud.u-szeged.hu, nyul@inf.u-szeged.hu

Abstract. Barcode technology is essential in automatic identification, and is used in a wide range of real-time applications. Different code types and applications impose special problems, so there is a continuous need for solutions with improved performance. Several methods exist for code localization, that are well characterized by accuracy and speed. Particularly, high-speed processing places need reliable automatic barcode localization, e.g. conveyor belts and automated production, where missed detections cause loss of profit. Our goal is to detect automatically, rapidly and accurately the barcode location with the help of extracted image features. We propose a new algorithm variant, that outperforms in both accuracy and efficiency other detectors found in the literature using similar ideas, and also improves on the detection performance in detecting 2D codes compared to our previous algorithm.

Keywords: barcode detection, morphological operations, bottom-hat filter, distance map.

1 Introduction

Barcode detection is required in a wide range of real-life applications. Computer vision algorithms vary considerably and each application has its own requirements. Many barcode localization methods have been developed for automatically segmenting barcode patterns from images.

The term barcode can be used for various types of visual codes. In this paper, we deal with classical 1D barcodes and stacked 2D barcodes. Barcodes are not human-readable and traditional devices have been widely adopted for personal use. The traditional barcode structure is simple: the variation of different thickness of parallel light and dark bars represent information. Such codes can be read optically by a machine. Code types vary from each other in what configurations of black and white bars correspond to a given character. The most frequent application of barcodes is the trade, e.g. in goods packing. The identification number visualized with symbols permit the use of electronic reading by machines which support and speed up the information streaming. Barcodes can have fix or variable length. In fix length codes, the standard specifies how many
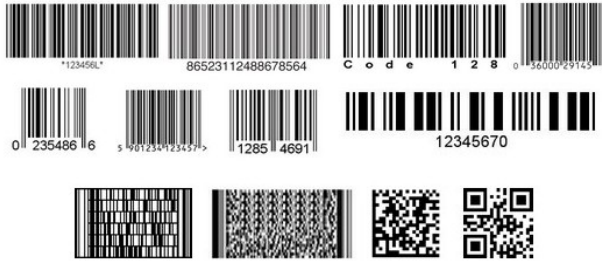
Fig. 1. Barcode patterns (from left to right). Top row (1D codes): Code39, Codabar, Code128, UPC-A; Middle row (1D codes): UPC-E, EAN-13, EAN-8, I2of5; Bottom row (2D codes): Codablock, PDF417, Data Matrix, QR.

characters are represented in a code, while the other type may encode an arbitrary number of characters. Different standard code types have specific features that help their localization. Figure 1 shows a set of standard 1D barcode types and some widely used 2D barcode types. [1]

Barcode localization methods have two competing objectives: detection accuracy and speed. For industrial environment, accuracy is crucial since undetected (missed) codes may lead to loss of profit. Processing speed is a secondary desired property of the detectors. On smartphones, accuracy is not so critical, since the device interacts with the user and reshooting is easily done, nevertheless, fast (and reasonably accurate) barcode detection is desirable. Various techniques are used to locate and decode barcodes from photographs: from the classical line scanning technique [2], through the widely studied morphological approaches [3–10], and recent studies using wavelets [11].

We propose a novel barcode detection algorithm based on bottom-hat filtering and other simple operations from mathematical morphology, which ensure high efficiency. We compare the effectiveness of the proposed method with several approaches from the literature and show that, in most cases, our algorithm outperforms the others. It also improves on the detection accuracy of our previous algorithm [10] for 2D stacked barcodes.

2   Methods

In this section, we present several barcode detection algorithms that use different approaches to determine the barcode location in an image. The methods presented in Sections 2.1, 2.2, and 2.3 were re-implemented according to the original papers. Where some details were not available in the papers, we made our best effort to fill in the gaps, and this is also noted in our description. The new proposed algorithm is presented in full detail in Section 2.4. For visual reference, Figure 2 illustrates intermediate stages of the processing by the described algorithms.
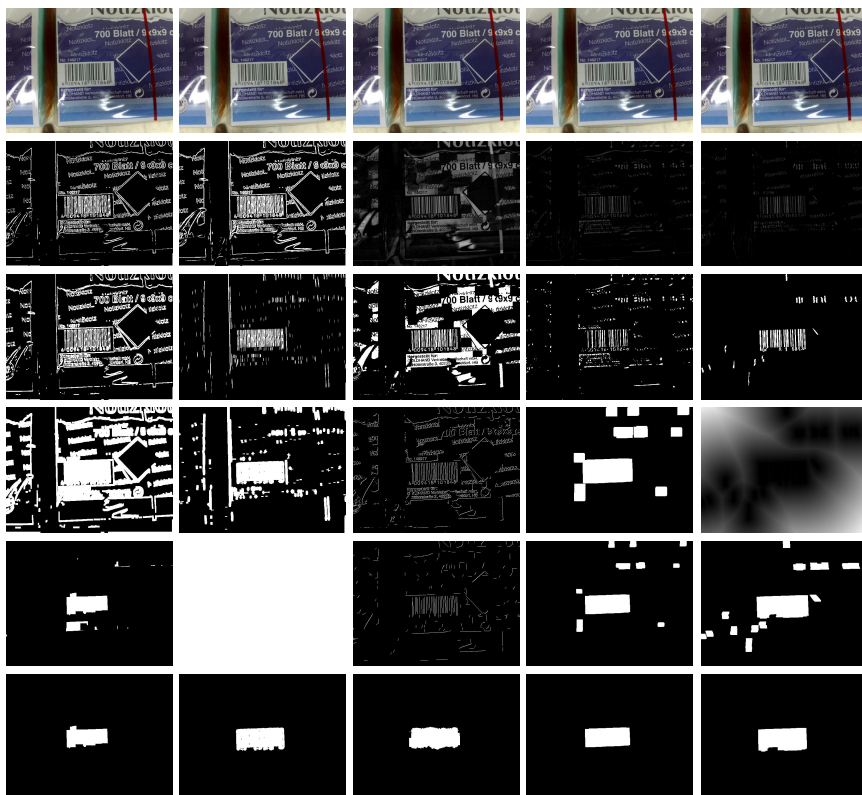
Fig. 2. Intermediate stages of the processing by the described algorithms. Columns: TT, ETJC, JJXQ, MKLN, and Proposed (see text). First row: original image, last row: final output.

Nowadays, the quality of digital images are usually very good, although low-quality recordings also exist. The reason for lesser quality may be e.g. the capturing device, and the environment may also impose problems. Therefore, often there is a need for correcting (improving) the image quality before the particular detection process takes place.

2.1   Method Based on Basic Morphological Operations

In Tuinstra's [3] algorithm, the author relies on that in the barcode region, the intensity difference between the stripes is high, so the gradient would highlight the bars. Sobel kernels are used to estimate the gradient in the x and y directions. Next, the gradient image is thresholded, pixels having a high gradient value are selected. On the binary image, first a hit-or-miss transformation is performed with a line structuring element, which, unfortunately, is not specified in the original article [3]. Next, morphological dilation is performed in order to merge

nearby but not necessarily connected objects to be able to compose a region. The structuring element is square shaped, but its exact size was not fixed in the paper. We used a 10×10 block SE matching the size used for the previous step. In practice, the image size determines this parameter. Morphological erosion follows to discard thin objects from the image and remove undesired segments which were fused by the dilation. Here, the structuring element is greater than that used for dilation. We used a 20×20 block SE. The final operation is a solidity test that compares the number of pixels turned on in a region to the convex hull of the region. After this step, likely false positive objects are removed and only barcode regions remain.

2.2   Method Based on Image Scanning

The procedure of Telkin and Coughlan [4] was designed for visually impaired or blind people to facilitate their everyday lives. It is presented in two main phases. First, the noise level is reduced by Gaussian smoothing. Since the original paper did not specify the σ parameter, we used the same value of $\sigma = 0.3$ as in our Proposed method. Then, pixel gradient values are calculated using the Sobel operator, thus yielding a kind of edge enhancement in the image. Next, binarization is done so that pixels having a gradient above the threshold become white and the others black. We used a threshold value set to 95 % of the maximal gradient value. In the detection phase, first, the image is scanned in four directions (horizontally, vertically, and in direction of the diagonals ($\pm 45°$)). The horizontal scan comes first, wherein those edge pixels are found whose orientation is vertical. The method looks in the vicinity of each edge pixel for opposite polarity pixels. An area will be kept if there is a sufficient number of such pixels in the area (as if they were part of a line segment). Vertical scan follows, when those segments remain which have almost the same beginning and end so they most likely belong to barcode areas. At the final stage of the detection the entropy value is calculated for each pixel in the resulting image that describes the disorder of intensities within a given neighborhood around each pixel.

2.3   Method Based on Bottom-Hat Filtering

The algorithm by Juett and Qui [5] is based on bottom-hat filtering. In preprocessing, the method corrects the non-ideal image with simple contrast stretching in order to highlight differences between light and dark areas. Next bottom-hat filtering is applied, wherein the size of the structuring element depends on the widest bar in the barcode to be detected. The article specified 25×25 block SE for images of size 720×480 pixels. After binarization, the contour is defined. It is followed by the step to determine the orientation of the bars, which is performed by directional image openings using a relatively large linear structuring element. These openings are performed at 16 different orientations, with a step of 11.25°. The directional opening images are summed and a low resolution density image is calculated, which is then converted back into binary. Each region represents a

potential barcode region. In the last phase, objects whose area is smaller than a given threshold are eliminated.

2.4 Proposed Method Using Bottom-Hat Filtering

Our aim was to devise an algorithm with sufficiently high speed and accuracy. The presented algorithm is a modified version of our previously published method [10], and is summarized in Figure 3. Similarly to the above described

---

ALGORITHM:

$f := (f \bullet SE_1) - f$
for all pixels **do**
    if $f_{i,j} > i_t$ **then** $f_{i,j} \leftarrow 1$
    else $f_{i,j} \leftarrow 0$
Calculate Euclidean distance map
Compute the minimum X of row averages in the distance map
Drop those pixels, whose distance value is greater than X
$f := f \oplus SE_2$
$f := f \ominus SE_3$
for all components **do**
    if component area $> min_a$ **then** Record this component as a barcode segment
    else Discard this component

Fig. 3. Pseudocode of the Proposed algorithm

methods [3–5], our algorithm consists of two main phases. In the preprocessing phase, the input image is converted to grayscale, because the pattern of dark bars on a light background is apparent regardless of some codes being printed using colors in certain applications.

To reduce the image noise, we smooth with a Gaussian kernel using $\sigma = 0.3$. Edge enhancement follows. Instead of calculating grafient values (like done in most barcode localization methods), we use bottom-hat filtering that is also based on intensity differences. Although bottom-hat filtering is less attractive regarding operation time than other non-directional edge enhancement operations, its accuracy is higher. The proposed algorithm calculates the frequency of the most frequently occuring element. Later we refer to this value as $MaxFreq$. We defined a linear structuring element and its rotated version by 90°. We perform bottom-hat filtering using both structuring elements and later use the one that produces stronger matches. The grayscale images are then converted to binary, using a standard thresholding technique. We used $MaxFreq$ value and image size for calculating this threshold value. We assumed that, if $MaxFreq$ is very high, the image is noisy or blurred or both. In this case, we set the threshold to 95 % of the maximal value, otherwise the threshold is set to 80 % of the maximal value.

After preprocessing, the digital image is suited to finding segments which contain barcode. The resulting image so far contains many false positive regions. There are many small connected components which satisfy the criteria but do not belong to barcode regions. These false regions are eliminated using an area threshold. Selecting a too high threshold value could remove small barcodes from high-resolution images. If the image size is higher than 800×800, we set the threshold to half the size of the largest component, otherwise the threshold is set to 1/4 of the size of the largest component.

We take advantage of the structure of the barcode, the fact that it consists of approximately regularly spaced parallel stripes. So, for each pixel, we calculate the Euclidean distance of the pixel from the nearest nonzero pixel. Using the distance map, objects that are far from other objects can be easily dropped, and only nearby objects (sort of a cluster of bar segments) will be kept. The distance threshold is selected as the minimum of the row averages of the distance map.

Our observations showed, that for the final detection steps, mostly dense text regions remain along with the supposedly barcode areas. Since a barcode consists of a sequence of parallel bars that are located at varying distance from each other, they do not compose a connected component. Therefore, to merge these patterns, we use dilation with a square structuring element whose size is defined as $S = \max(40, \text{width of the widest bar} * 3)$. Nevertheless, this dilation may also thicken and merge unwanted, non-barcode locations as well. To cope with this problem, we also use the dual operation, i.e. erosion. Here, the structuring element is linear, consistent with the size of the 1D barcodes. The structuring element size is less than (about 1/3 of) the one used for dilation, i.e., it matches the width of the widest bar. After erosion, those areas that likely contain barcodes can be found. Of course, there may still be false positive objects. These are removed in the last step on the basis of their size and proportions. Similarly to the previous step, if the image size is higher than 800×800, objects smaller than half the size of the largest object are removed, otherwise objects smaller than 1/4 of the size of the largest object are removed.

3   Evaluation

In this section, we compare the discussed methods' effectiveness under specific conditions. We use the following acronyms for referring to the algorithms: TT (Timothy R. Tuinstra's method [3]), ETJC (Ender Telkin és James M. Coughlan's method [4]), JJXQ (James Juett és Xiaojun's method [5]), MKLN (our previous method [10]), and Proposed (Prop.) for our new proposed algorithm.

3.1   Test Suite, Test Environment, and Implementation

We generated barcodes digitally with the types shown in Figure 1. Only one base image was chosen for each code type. Test images contained one or three barcodes from each types, respectively, and images were affected by distortions. For each base image, we generated all combinations of the following properties:

rotation in every 15° from 0° to 180°, Gaussian blur filter using a 3×3 kernel with 6 different values of σ, additive noise from 0 % to 50 % with a step of 10 %, and an optional 15 % shear. In summary, the test set contained images presenting either 1 or 3 codes of 10 different barcode types, with 12 orientations, 6 different blur filters, 6 different rates of additive noise, with or without shear, totaling in 17,280 images. Figure 4 shows a selection of generated test images with various code content and applied distortions.
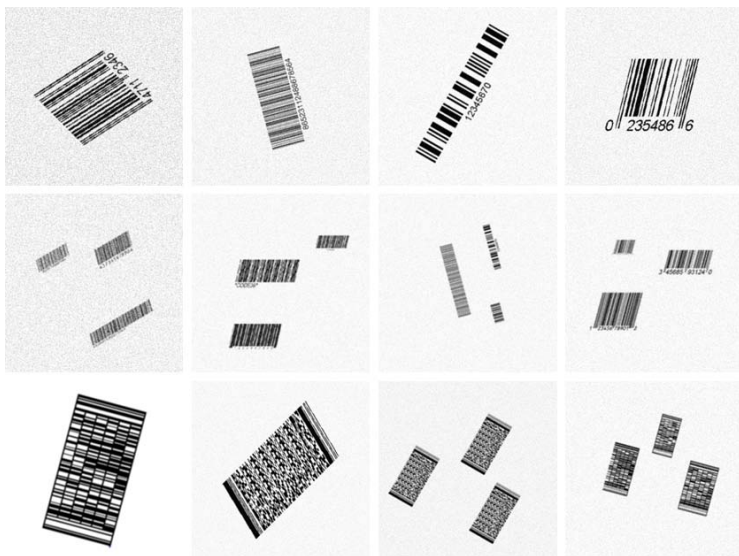
Fig. 4. A selection of generated test images with various code content and applied distortions

Another 100 images containing barcodes were collected from real-life images without any modifications. These images presented scratches, blur, minor light reflections and distortions also.

The methods were implemented in MATLAB using the Image Processing Toolbox. Evaulation was performed on a computer with Pentium(R) Dual-Core 2.30 GHz CPU.

3.2 Results and Discussion

In this section, we show how effective the implemented algorithms are on images with various characteristics, and compare their running time also. For calculating accuracy we used the Jaccard coefficient of similarity, measuring the overlap of the bounding boxes of the real and the detected barcode region. $J(A, B) = \dfrac{|A \cap B|}{|A \cup B|}$, where A denotes the bounding box of the real barcode and B that of

the detected code. This not only measures the success of detection, but also considers the delineation accuracy of the methods.

The performance of the algorithms do not seem to depend on the applied stretching in the test images and also seem to be insensitive to rotation as well.

We can conclude that all five algorithms are capable of effectively localizing barcodes. ETJC method is fast, but it looses efficiency when there are several barcodes in the image and noise is also present. TT algotithm is slower than the others, which can most probably be attributed to that here the image is scanned in four different orientations. JJXQ, the method based on bottom-hat filtering falls off to the discussed fast algorithms with respect to running time. The process is slower than the others because in the detection phase the image is scanned in 16 different orientations. However, this brings better accuracy to the process at the expense of more computation.

Our proposed algorithms are also very fast. In most cases, faster procedures locate barcodes less accurately or not at all, as if loss of accuracy were the price for the speed gain. However, in our case, speed is not coupled with a significant cost in accuracy. From the evaluation output we can conclude that the proposed procedure has second best running efficiency for all test images (only our previous version executes faster), and in many cases it also shows best accuracy. The weakness of the method appears when the images is very noisy or when there are such image areas which are similar to a barcode.

The efficiency of the methods, i.e. processing time is also an important aspect. Table 1 presents the execution time of the detection methods for images with 1 or 3 barcodes. Here, one can easily appreciate the significant differences between the different approaches.

Table 1. Running time of the algorithms. Mean values (± standard deviation) (expressed in seconds) for all test images of a certain class

		TT	ETJC	JJXQ	MKLN	Proposed
1 code	1D	0.23±0.13	0.67±0.22	0.81±0.43	0.13±0.10	0.16±0.08
	2D	0.21±0.14	0.77±0.47	0.73±0.44	0.12±0.12	0.14±0.09
3 codes	1D	0.54±0.35	1.73±0.66	1.86±0.78	0.18±0.09	0.35±0.12
	2D	0.46±0.21	1.70±0.54	1.68±0.71	0.18±0.10	0.28±0.07

Subsequent tables show, how the algorithms behave on images which contain either one or three code pieces of various 1D and 2D barcodes types. The structure of barcode types varies, which also has an effect on how well the algorithms can perform. Accuracy of the detection methods for images containing a single or 3 pieces of codes for various code types is presented in Table 2. TT, ETJC, and JJXQ has very bad performance for Code128, however MKLN and the Proposed algorithm handle this variable-length code as well as the fixed-length types. For the 2D codes, ETJC shows exceptionally good accuracy, but the other methods also perform well on these stacked barcodes. There is more variance in performance on images with 3 codes, and some methods (TT, MKLN, Proposed)

Table 2. Accuracy of the algorithms for various types of codes, for images containing 1 or 3 codes. Mean values (expressed in percent) for all distorted test images with 1D barcodes and for all test images with 2D codes.

		1 code					3 codes				
		TT	ETJC	JJXQ	MKLN	Prop.	TT	ETJC	JJXQ	MKLN	Prop.
1D	Codabar	99.0	98.9	99.2	99.9	98.6	83.8	56.2	52.7	92.8	87.5
	Code128	50.4	63.6	70.9	95.7	93.7	83.6	95.0	79.6	80.1	84.8
	Code39	98.7	94.8	86.3	99.4	100.0	83.3	90.5	67.3	94.7	90.5
	EAN-13	98.8	94.7	91.7	98.9	99.3	83.3	91.4	76.6	99.3	99.2
	EAN-8	98.7	99.6	84.6	99.6	98.2	83.3	97.2	81.8	99.4	95.9
	I2of5	98.6	84.6	84.4	93.1	92.5	78.8	35.5	93.8	99.2	84.9
	UPC-A	99.1	98.2	94.9	86.4	100.0	74.6	65.5	92.6	99.9	95.4
	UPC-E	82.7	92.6	76.0	99.6	100.0	98.7	77.5	96.7	99.3	92.5
	all 1D	91.8	86.4	78.9	96.6	96.8	74.0	74.7	77.4	95.6	89.2
2D	Codablock	87.0	99.8	83.3	87.7	100.0	86.6	100.0	80.0	90.7	94.2
	PDF417	81.7	99.6	79.7	93.9	97.6	83.3	100.0	83.9	91.7	90.8
	all 2D	85.0	99.7	82.5	90.8	99.1	85.0	100.0	82.4	91.2	92.5

show a solid performance while the others very much depend on the type of code present in the image.

The accuracy of the algorithms degrades on distorted images. Next, we analyze the effect of distortions on efficiency. Table 3 shows the methods' behavior with respect to the level of blur applied to the test images. Image smoothing does not change the performance considerably. Although there is a noticeable inverse relation between values of σ and accuracy, this is not significant. It is interesting to note that in some cases the barcode detection accuracy is higher in images with three barcodes than for images with a single code present. This can be due to the fact that relative size (w.r.t. image size) of individual codes are smaller in the 3-code examples. ETJC, MKLN, and the Proposed method show outstanding accuracy in images which contain 2D barcodes and there are not considerable differences between the precision values.

In a similar manner, we analyzed how the algorithms perform on images having different levels of noise added. We can see from Table 4 that each algorithm is somewhat sensitive to noise. The studied literature methods loose much of their accuracy as the noise level increases, however, our methods only degrade to a lesser extent.

We also tested the methods on a set of 100 real-life images. These images contain 1D barcodes each. The accuracy of the algorithms are respectively 92 % (TT), 91 % (ETJC), 92 % (JJXQ), 94 % (MKLN), and 96 % (Proposed).

In Figure 5 we show two result images of our algorithm for 2D barcodes. Here, we have to locate all parts of the barcode, because stacked barcodes contain data everywhere and there is a much smaller visual redundancy as in 1D barcodes.

For the digitally generated images maximal size was 800×800. At this resolution TT and JJXQ cannot compete with the other two methods in speed. MKLN and the Proposed method is not far better on 800×800 images than ETJC, but

Table 3. Accuracy of the algorithms for different blur levels, for images containing a single code and for those containing 3 codes. Mean values (expressed in percent) for all distorted test images with 1D barcodes and for all test images with 2D codes.

		1 code					3 codes				
		TT	ETJC	JJXQ	MKLN	Prop.	TT	ETJC	JJXQ	MKLN	Prop.
1D	0.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	0.5	91.2	88.4	91.3	96.1	95.7	93.8	94.4	92.4	98.0	100.0
	1.0	90.4	88.5	89.4	97.1	95.4	92.4	86.7	89.1	98.0	100.0
	1.5	90.4	88.5	89.4	96.8	95.4	91.7	86.7	89.1	97.0	100.0
	2.0	90.3	90.4	87.5	97.1	93.3	86.3	82.9	90.2	97.0	100.0
	2.5	90.4	90.4	87.5	97.1	93.3	82.7	82.0	90.2	97.0	100.0
	all 1D	90.5	89.1	89.6	96.7	94.3	82.7	77.9	90.6	97.5	100.0
2D	0.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	0.5	88.8	100.0	83.1	95.8	100.0	89.1	100.0	83.2	100.0	100.0
	1.0	88.4	100.0	82.6	95.7	100.0	87.7	100.0	82.5	100.0	100.0
	1.5	87.7	100.0	82.6	95.7	100.0	87.0	100.0	82.5	100.0	100.0
	2.0	83.6	100.0	82.6	95.7	100.0	85.9	100.0	74.5	100.0	100.0
	2.5	83.6	100.0	82.6	95.7	100.0	85.0	100.0	74.5	100.0	100.0
	all 2D	87.5	100.0	83.3	95.8	100.0	87.0	100.0	82.9	100.0	100.0

Table 4. Accuracy of the algorithms for different noise levels, for images containing a single code and for those containing 3 codes. Mean values (expressed in percent) for all distorted test images with 1D barcodes (top), and for all test images with 2D codes (bottom).

		1 code					3 codes				
		TT	ETJC	JJXQ	MKLN	Prop.	TT	ETJC	JJXQ	MKLN	Prop.
1D	0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	10	90.2	89.5	93.9	97.4	97.1	94.8	97.5	92.6	96.0	95.8
	20	89.7	86.9	87.5	97.4	97.1	89.4	87.6	89.5	96.3	91.6
	30	89.4	86.2	79.2	97.4	92.5	84.6	86.3	82.7	96.7	95.8
	40	90.1	90.1	78.5	97.8	92.5	87.2	84.9	79.9	96.7	95.8
	50	93.3	84.2	69.9	94.9	91.6	73.8	82.7	64.4	90.6	91.6
	all 1D	90.5	88.6	83.5	96.8	95.4	87.6	87.3	84.7	95.6	95.1
2D	0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	10	87.7	100.0	84.7	94.4	100.0	86.3	100.0	84.0	100.0	100.0
	20	87.7	98.6	81.9	94.2	100.0	85.8	100.0	78.9	100.0	100.0
	30	87.7	100.0	79.3	94.2	100.0	90.4	100.0	73.0	100.0	100.0
	40	85.7	100.0	76.7	94.2	100.0	86.3	100.0	72.6	100.0	100.0
	50	77.8	98.6	69.6	92.3	100.0	84.9	100.0	71.4	85.7	100.0
	all 2D	85.8	99.5	77.3	94.8	100.0	86.4	100.0	73.3	95.6	100.0
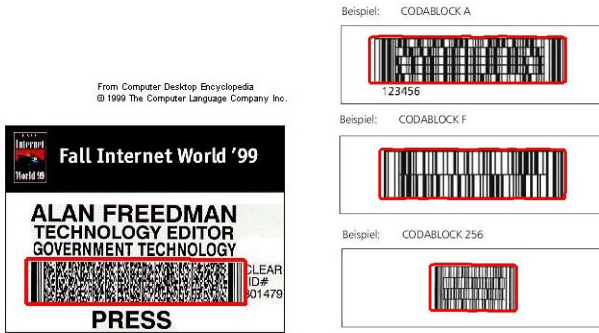
Fig. 5. Display examples of detection output by the propsed algorithm for PDF417 (left) and Codablock (right) code samples

when executing them on 2500×1722 images, ETJC runs for 3.2 sec, while our algorithms finish under 1.5 sec. The Proposed algorithm is somewhat slower than the MKLN version due to the extra computation of the distance map. We can state, that for larger images there are considerable execution time differences between the studied algorithms.

Our algorithms outperform the three algorithms from the literature both in terms of accuracy an speed. Note, that methods which use bottom-hat filtering in the preprocessing phase, have higher accuracy than the others, but their running times are quite different from each other. JJXQ looses its accuracy proportional to image degradation, while our proposed algorithms maintain higher accuracy for those cases as well. The tests also demonstrated that in all cases the methods' running time significantly grows with increasing image size. Among the two methods of ours (MKLN and Proposed), MKLN accuracy is slightly better than that of the Proposed version for the entire pool of our synthetic test images, including many 1D as well as 2D code examples. Regarding images with 2D stacked codes only, the newly proposed algorithm clearly outperforms the MKLN version.

4   Conclusion

We have presented a new version of our barcode detection algorithm and compared its performance (in terms of accuracy and speed) with our previous version and three other methods from the literature using similar ideas. We concluded that these algorithms do not specialize for individual barcode types, they can efficiently detect various types of 1D and stacked 2D barcodes, but there are subtle differences in performance. We have built a test database containing 17,280 synthetic images representing various degradations (blur, noise, shear) as well as collected 100 real images for evaluation. We demonstrated that the proposed new methods are less sensitive noise than those from the prior art. Our proposed methods outperform the other three in detecting 1D codes both in terms

of accuracy and speed, and the new proposed version has better accuracy for 2D stacked barcode examples.

References

1. Palmer, R.C.: The Bar Code Book: Reading, Printing, and Specification of Bar Code Symbols. Helmers Pub. (1995)
2. Adelmann, R.: Toolkit for bar code recognition and resolving on camera phones - jump starting the internet of things. In: Informatik 2006 Workshop on Mobile and Embedded Interactive Systems (2006)
3. Tuinstra, T.R.: Reading Barcodes from Digital Imagery. PhD thesis, Cedarville University (2006)
4. Tekin, E., Coughlan, J.M.: An algorithm enabling blind users to find and read barcodes. In: Workshop on Applications of Computer Vision (WACV), pp. 1–8 (2009)
5. James Juett, X.Q.: Barcode localization using bottom-hat filter. NSF Research Experience for Undergraduates (2005)
6. Bodnár, P., Nyúl, L.G.: Efficient barcode detection with texture analysis. In: Proceedings of the Ninth IASTED International Conference on Signal Processing, Pattern Recognition, and Applications, pp. 51–57 (2012)
7. Bodnár, P., Nyúl, L.G.: Improving barcode detection with combination of simple detectors. In: The 8th International Conference on Signal Image Technology (SITIS 2012), pp. 300–306 (2012)
8. Chai, D.: Locating and decoding ean-13 barcodes from images captured by digital cameras. In: Fifth International Conference on Information, Communications and Signal Processing, pp. 1595–1599 (2005)
9. Lin, D.T., Lin, M.C., Huang, K.Y.: Real-time automatic recognition of omnidirectional multiple barcodes and dsp implementation. Machine Vision and Applications 22, 409–419 (2011)
10. Katona, M., Nyúl, L.G.: A novel method for accurate and efficient barcode detection with morphological operations. In: The 8th International Conference on Signal Image Technology (SITIS 2012), pp. 307–314 (2012)
11. Wachenfeld, S., Terlunen, S., Jiang, X.: Robust recognition of 1-d barcodes using camera phones. In: 19th International Conference on Pattern Recognition (ICPR 2008), pp. 1–4 (2008)