Proceedings of the IASTED International Conference
Computer Graphics and Imaging (CGIM 2013)
February 12 - 14, 2013 Innsbruck, Austria

BARCODE DETECTION WITH UNIFORM PARTITIONING AND DISTANCE TRANSFORMATION

Péter Bodnár and László G. Nyúl
Department of Image Processing and Computer Graphics
University of Szeged
Árpád tér 2., Szeged, H-6720 Hungary
email: {bodnaar, nyul}@inf.u-szeged.hu

ABSTRACT
Barcode detection is required in a wide range of real-life applications. Imaging conditions and techniques vary considerably and each application has its own requirements for detection speed and accuracy. In our earlier works we used uniform partitioning with several approaches for detection of various types of 1D and 2D barcodes and showed their behaviour on a set of test images. In this work, we extend the partitioning idea and replace scan-line based methods with distance transformation to improve accuracy.

KEY WORDS
barcode detection, computer vision, partitioning, feature extraction, distance transformation, data matrix

1 Introduction

Visual codes referred as barcodes have two major groups, 1- and 2-dimensional barcodes. 1D barcodes consist of a well-defined group of parallel lines aiming at easy automatic identification of carried data with endpoint devices such as PoS terminals, smartphones, or computers. Most two-dimensional barcodes serve the same purpose as their one-dimensional ancestors. They usually consist of simple picture elements (e.g. rectangles, dots, hexagons) organized in a 2D pattern layout. In this paper we restrict ourselves to codes like those in Fig. 1.

Barcode detection methods have two main objectives, speed and accuracy. On smartphones, fast detection of barcodes is desirable, but accuracy is not so critical since the user can easily reposition the camera and repeat the scan. Accuracy is critical for industrial environment (e.g. postal services), where false negatives cause loss of profit. Speed is also a secondary desired property in those applications.

For 1D barcodes, the basic approach for detection is scanning only one, or just a couple of lines of the whole image. This method is common for hand-held PoS laser scanners or smartphone applications. Scanned lines form an 1D intensity profile, and barcode-detector algorithms [1, 2] work on these profiles to find an ideal binary function that represents the original encoded data. The main idea is to find peak locations in blurry barcode models, then thresholding the intensity profile adaptively to produce binary values.

Valley tracing (or bar tracing) [1] is a method for finding barcodes in blurry, low resolution images, mostly on live smartphone camera frames. It consists of three steps. At first, we find starter points on the pictures, then follow the "valleys", and finally, recognizing the ends of the valleys (bars).

For 2D barcodes, most approaches involve the extraction of texture-like properties and detection of properties that refer to code-like appearance. Also, there are approaches for detection of specific types of codes with heavy perspective distortion, motion blur [3], and noise [4].

Algorithms based on morphological operations [5–8] produce a feature with combination of dilation, erosion, and operations derived from them. White blobs on these images show the possible barcode locations. Further processing, like segmentation and filtering of small blobs are required on these difference images. This approach can be used on both 1D and 2D barcodes with minor modifications.

Methods based on wavelet transformation [9] look at images for barcode-like appearance by a cascaded set of weak classifiers. Each classifier working in the wavelet domain narrows down the possible set of barcodes, decreasing the number of false positives while trying to keep the best possible accuracy.

Variants of Hough transformation [10] detect barcodes by working on the edge map of the image. The two most common methods are standard and probabilistic Hough transformation. Both transform edge points into Hough space first, and make decisions of line locations.

Other methods use various types of image filtering, like Kalman or Gabor-filters [11].

Our proposed method examines the image in small, disjoint or overlapping tiles, and make local measurements. We use distance transformation [12] for these measurements, a general operation of image manipulation. Code parts, like other textures, have well-traceable features. One of them is neighbour similarity, which means code parts in close proximity share similar local statistics with a well-chosen tile size. Thanks to the repeating patterns in the codes we can detect codes by observing how many code-like image parts (tiles) can be found. Lastly, we have to mention compactness of code parts, which influences the final decision about the code center.

DOI: 10.2316/P.2013.797-022

Figure 1: Barcode patterns. Row 1 (from left to right): Codabar, Code 11, Code 128, Code 39, Plessey, EAN-13, UPC, UPC-A. Row 2: Maxicode, Aztec code, Shotcode, Codablock, PDF417, Data matrix, QR code, Micro QR

2 The Proposed Barcode Detection Method

In this section we introduce a new approach for both 1D and 2D barcode detection with a combination of partitioning and distance transformation. First, we look at image parts locally for finding barcode-like features, then we examine the features at a higher level of hierarchy.

2.1 Preprocessing

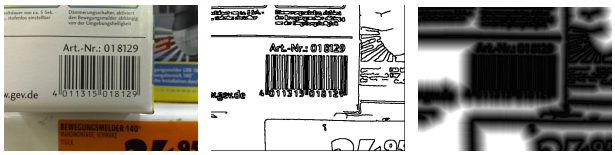
Digital images acquired from a camera often need preprocessing because of device flaws or environmental difficulties. In images having low contrast, intensity levels should be normalized.

Image resolution does not have to be high, barcodes having the narrowest line of two pixels is sufficient (3×3 px median filters can be applied to eliminate salt-and-pepper noise). For 2D barcodes, element size of 3×3 pixels is required for the same reason. Higher resolution yields to better results, but also increases computation time. The least time-consuming solution for downsampling such images is the nearest neighbour interpolation, which is also a good choice because it preserves strong edges. However, at least 3 px minimum line width (for 1D) and 5×5 block size (for 2D) is desired for accurate code detection.

2.2 Uniform Partitioning with Distance Transformation

Most barcodes, like regular textures, can be easily identified by observing only small parts of them. These barcode parts together form the desired barcode region with known height and width. The first part of the method is partitioning the image to square tiles and look at each tile for barcode-like appearance. Each tile is assigned a value that indicates the grade of the presence of this feature. Globally, a matrix is formed from these values. Texture parts have similar local statistics in their neighbourhood, so searching this matrix for compact areas defines image ROIs representing a barcode with high possibility.

The assigned value showing barcode-like appearance is based on distance transformation of the edge map (Fig. 2). Distance transformation is an operation that works with an initial set of points, like corners or edges. Value 0



(a) original image (b) edge map (c) distance map

Figure 2: Canny edge map (b) and distance map (c) of a real-life example image (a). Barcodes have compact dark areas. Note: the values in the distance map are scaled for visualization.

(a) original image (b) distance map (c) feature image

Figure 3: Uniform partitioning of a synthetic Aztec code (a). On the feature image (c), each tile has a value of mean distances of the distance map (b). The feature image has darker areas ("holes") where blocks with similar color are placed next to each other in the original image. (Brightness values of the feature image are for illustration purposes, based on the current tolerance to distance means.)

is assigned to these points, and any others get the distance value to the closest point of the initial set. We apply Canny edge detector for this point set to the transformation. For each tile of the distance map, means and standard deviations are calculated. For 1D codes, distance values spread between half of the minimum and half of the maximum line width. For 2D codes, these values usually stay below half of their block size, but higher values are also possible, since blocks of the same color can be next to each other in multiple directions. Having such code parts significantly raise the mean of distance values on the image tile. However, these "holes" in the feature matrix are rare enough so they do not split barcode-like areas (Fig. 3).

After evaluating all tiles locally, we look for clusters

(a) original image (b) colored feature image

Figure 4: Typical image of a barcode. The feature image is based on distance transformation of the edge map. As the last step, small clusters (shown in red), and shapes having low compactness (shown in blue) are dropped. Final important regions are colored green.

in the feature matrix. Experiments show that for 1D codes, mean of distance values spread around half of the average line width, and for 2D codes, the values spread around the square root of their element size. We applied a threshold for the values to classify whether or not an area contains a barcode segment. Letting a 25 % tolerance for these idealistic values, detection accuracy becomes satisfactory. For end-user applications we have to take noise, scratches and reflections into consideration. For images containing heavy noise, distance means drop. Barcodes suffering from scratches, dust, handwriting or reflections, change the distance means significantly according to the dark or bright intensity values of the flaw. Tolerance should be set according to amounts of these distracting properties and exact values can also be measured via trial and error. Tolerance value is a compromise between accuracy and the rate of false positive detections, so it should be set with respect to the final application.

The resulting binary matrix can be further analized via connected component labelling [13]. Finally, components showing small size or low compactness are dropped, and momentums of the remaining components are returned (Fig. 4). A component is considered small if it contains less than N tiles (Eq. (1)), where h is barcode height, w is barcode width and s is the tile size.

$$N = \max\left(4, \frac{|h-s| \times |w-s|}{s^2}\right) \tag{1}$$

We obtained the useful range of tile size by experimenting. If we consider the tightest bounding box for a barcode, we choose its smaller dimension as a reference. Tile height greater than half of this value leads to poor detection accuracy, and choosing the tile size too small, such as the block size of a QR code leads to losing this measure because of high differences in means. The best tiling size appears to be about $1/3$ of the smaller dimension of the bounding box.

It is possible to run the partitioning with disjunct or overlapping tiles, but overlapping does not improve the method's accuracy, only increases computation time. Offsetting the tiles has no significant effect either, since it just produces some blocks to be more and others to be less "barcode-like" at the barcode's opposite sides.

This method gives a noticeable amount of false positives, since text can also satisfy well the condition of having similar distance averages than a barcode. However, it can be used as a weak classifier of image areas, and its output is a good starting point for more accurate methods.

3 Evaluation

The discussed methods were tested on a fair amount of images of different types of barcodes, both synthetic and real.

3.1 Test Suite

Due to the absence of public 2D barcode databases, we had to run the quantitative test on our own data set. We digitally generated barcodes of the types shown in Fig. 1. For every type of code, five base test images were created with different embedded data. We generated every combination on the base image set of the following properties: rotation at 0, 30, 60, and 90 degrees; Gaussian blur filter with 6 different σ values as well as without any smoothing; uniform noise from 0 % to 50 % with a step of 10 %. The test set contained 16 different types of barcodes, with 4 orientations, 7 different smoothing filters, and 6 different levels of noise.

For more realistic examples, we googled up images of barcodes that had no distortion, good contrast and minor or no reflections. We also googled up abstract wallpapers and made 500 randomly cropped grayscale images from them. We embedded the cropped barcodes within the background images using randomly selected rotations from the 0–30° angular range around X, Y and Z axes, allowing for affine distortions.

Another 100 images containing barcodes were collected from real-life examples (Fig. 5) without any modifications to the images. Some images contain high percent of text content (Fig. 6), while other examples suffer from reflections (Fig. 7), scratches and distortions (Fig. 8).

3.2 Implementation and Test Environment

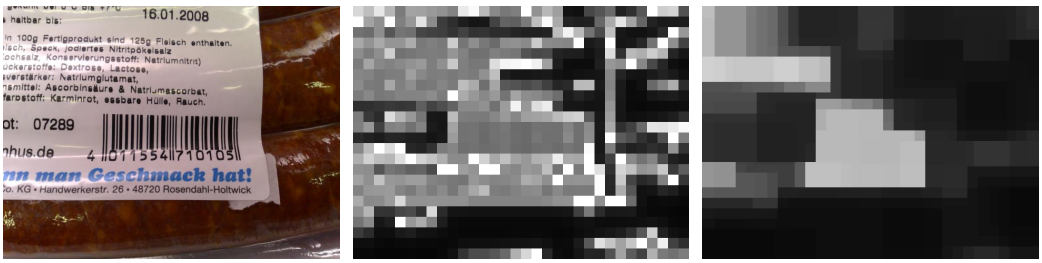
We implemented the method in `C++`, with the help of the `OpenCV` library. `C++` provides convenient OOP approach and fast code execution, while `OpenCV` has all the functions needed for image preprocessing and manipulation. Evaluation is performed on a computer with Intel(R) Core(TM)2 Duo 3.00GHz CPU.

3.3 Accuracy and Detection Speed

The proposed detection method consist of three main steps: producing the edge map, applying distance transformation and finally evaluating image tiles with the feature matrix. Edge detectors can be a bottleneck due to the convolution they are using, other steps can be executed in linear time.

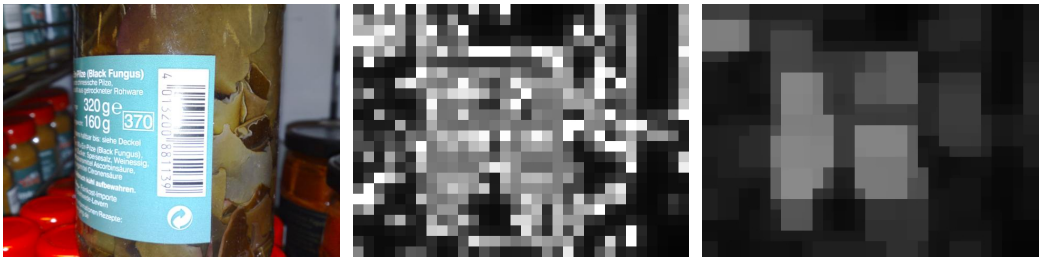
(a) original image (b) distance map (c) detected rectangular ROIs

Figure 5: Real-life example of a product case with uneven illumination. Original image (a), distance transformation (b), and the detected rectangular ROIs based on the distance map (c)



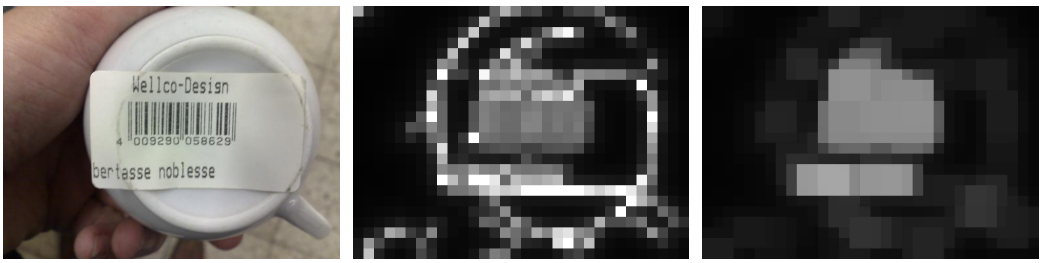
(a) original image (b) feature image (c) closed feature image

Figure 6: Grocery product example with high amount of text and barcode-like frequencies. Feature images contain such percent of feature responses needs further processing, like morphological closing



(a) original image (b) feature image (c) closed feature image

Figure 7: Another image of a grocery product with high amount of text and reflection at the barcode area. Distance transformation does not differentiate well between text and barcode in this case, but closing the feature image may lead to valuable localization result with a reasonable amount of false positive areas.



(a) original image (b) feature image (c) closed feature image

Figure 8: Real-life example with noise and minor distortions at the barcode area.

Detection accuracy is compared in Table 1 with a general barcode detection method using edge detection and probabilistic Hough transformation. Increased miss rate can be noticed with this method on 2D codes because it detects them by line directions. For accuracy formula, Jaccard index can be used on the pixels of the original and detected bounding boxes (Eq. 2)

$$J(O,D) = \frac{\sum_{x,y}(O(x,y) \wedge (D(x,y))}{\sum_{x,y}(O(x,y) \vee (D(x,y))} \tag{2}$$

where O and D are binary functions giving 1 on the inside of the original and detected bounding boxes respectively.

The proposed method shows no significant difference in detection of 1D and 2D codes. Minor differences in accuracy may occur due to the element layout caused by embedded data. The probabilistic Hough method performs better on 1D codes, because on 2D codes, there are more line directions which causes an increased miss rate. Both overall accuracy and runtime of the proposed method is comparable to the probabilistic Hough method.

On 1000×1000 px images, Canny edge detection with probabilistic Hough transformation takes 850 ms, while the proposed method takes about 650 ms measured on the above configuration.

Compared to other detection methods, uniform partitioning with distance transformation takes more time than the scan-line approach, but also provides better detection accuracy. Like the morphological MIN-MAX method, it also produces a noticeable amount of false positives, but also seems to have high tolerance to noise and blur. Perspective distortions are not handled well by some methods. Thanks to the nature of the proposed method, it is reliable on images with minor distortions, unlike the Hough transformation, which detects barcodes based on line angles.

4 Conclusion

We have presented an approach to detect both 1D and 2D barcodes in raster images using the idea of uniform partitioning and the statistics-based idea of distance transformation. We studied its behaviour on a set of images showing various barcode types. We found that there are two major groups of detection approaches, one concentrates on fast execution, while the other group concentrates on accuracy. Our proposed method is a compromise between speed and detection accuracy.

Distance transformation can also be considered as a preprocessing step for more complex localization algorithms, which will work on a subset of the original image pixels. Our future work includes further experiments in the Hough space, and aggregation of the Distance Transformation with other features.

In industrial setups parallel execution may also be possible for further improve detection speed. Furthermore, the data of the proposed algorithm, like the edge map can be re-used as the input for other, more accurate classifiers discussed in the first section.

References

[1] Timothy R. Tuinstra. *Reading Barcodes from Digital Imagery*. PhD thesis, Cedarville University, 2006.

[2] Eugene Joseph and Theo Pavlidis. Bar code waveform recognition using peak locations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(6):630–640, 1994.

[3] Chung-Hua Chu, De-Nian Yang, Ya-Lan Pan, and Ming-Syan Chen. Stabilization and extraction of 2d barcodes for camera phones. *Multimedia Systems*, 17:113–133, 2011. 10.1007/s00530-010-0206-9.

[4] Jong-Eun Ha. A new method for detecting data matrix under similarity transform for machine vision applications. *International Journal of Control, Automation and Systems*, 9:737–741, 2011. 10.1007/s12555-011-0415-9.

[5] Péter Bodnár and László G. Nyúl. Barcode detection with morphological operations and clustering. In *Signal Processing, Pattern Recognition, and Applications, Proceedings of the Ninth IASTED International Conference on*, pages 51–57, 2012.

[6] Daw-Tung Lin, Min-Chueh Lin, and Kai-Yung Huang. Real-time automatic recognition of omnidirectional multiple barcodes and dsp implementation. *Machine Vision and Applications*, 22:409–419, 2011. 10.1007/s00138-010-0299-3.

[7] Daw-Tung Lin and Chin-Lin Lin. Multi-symbology and multiple 1d/2d barcodes extraction framework. In Kuo-Tien Lee, Wen-Hsiang Tsai, Hong-Yuan Liao, Tsuhan Chen, Jun-Wei Hsieh, and Chien-Cheng Tseng, editors, *Advances in Multimedia Modeling*, volume 6524 of *Lecture Notes in Computer Science*, pages 401–410. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-17829-0-38.

[8] Péter Bodnár and László G. Nyúl. Improving barcode detection with combination of simple detectors. In *The 8th International Conference on Signal Image Technology (SITIS 2012)*, 2012.

[9] Rusen Oktem. Bar code localization in wavelet domain by using binary morphology. In *Signal Processing and Communications Applications Conference, 2004. Proceedings of the IEEE 12th*, pages 499–501, april 2004.

[10] Dana H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[11] Anil K. Jain and Yao Chen. Bar code localization using texture analysis. In *Document Analysis and Recognition, 1993., Proceedings of the Second International Conference on*, pages 41–44, 1993.

[12] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science, 2004.

[13] Michael B. Dillencourt, Hannan Samet, and Markku Tamminen. A general approach to connected-component labeling for arbitrary image representations. *J. ACM*, 39:253–280, April 1992.

Table 1: Accuracy of the proposed detection method compared to the reference method (mean ± sd, expressed in percent). Note: Probabilistic Hough method is unable to detect shotcodes

Code type	Code size	Code dim.	Proposed method	Hough transform.
Codabar	188 × 100	1	83.7 ± 36.9	96.7 ± 17.1
Code 11	176 × 64	1	87.5 ± 33.0	98.7 ± 11.7
Code 128	180 × 150	1	83.7 ± 37.1	97.2 ± 16.4
Code 39	332 × 100	1	87.9 ± 32.6	60.7 ± 48.8
EAN-13	190 × 138	1	100.0 ± 0.0	98.5 ± 12.2
Plessey	402 × 80	1	78.5 ± 41.1	76.1 ± 42.2
UPC	187 × 96	1	98.8 ± 10.6	99.7 ± 5.0
UPC-A	190 × 120	1	100.0 ± 0.0	98.7 ± 11.1
Maxicode	300 × 300	2	99.7 ± 7.4	34.2 ± 47.4
Aztec code	660 × 660	2	71.9 ± 45.0	78.3 ± 41.2
Shotcode	500 × 500	2	84.0 ± 31.4	N/A
Codablock	360 × 120	2	74.4 ± 35.2	98.8 ± 11.1
PDF417	360 × 120	2	100.0 ± 0.0	100.0 ± 0.0
Data matrix	420 × 420	2	100.0 ± 0.0	60.6 ± 48.9
QR code	310 × 310	2	96.7 ± 11.2	60.8 ± 48.8
Micro QR	340 × 340	2	100.0 ± 0.0	88.1 ± 32.3
All 1D codes			89.9 ± 23.9	90.8 ± 20.6
All 2D codes			90.8 ± 16.3	74.4 ± 32.8
All together			90.4 ± 20.0	83.1 ± 26.3