View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by SZTE Publicatio Repozitórium - SZTE - Repository of Publications

A Novel Method for Barcode Localization in Image Domain

Péter Bodnár and László G. Nyúl

Department of Image Processing and Computer Graphics
University of Szeged
{bodnaar, nyul}@inf.u-szeged.hu

Abstract. Barcode localization is an essential step of the barcode reading process. For industrial environments, having high-resolution cameras and eventful scenarios, fast and reliable localization is crucial. Images acquired in those setups have limited parameters, however, they vary at each application. In earlier works we have already presented various barcode features to track for localization process. In this paper, we present a novel approach for fast barcode localization using a limited set of pixels in image domain.

Keywords: barcode localization, feature extraction, pattern recognition, UPC

1 Introduction

Barcodes are visual codes consisting of well-defined symbols that carry embedded data along one or two dimensions in the image domain. Most common 1D barcodes are a set of parallel lines with specific bar width, height and order. 2D visual codes are also often referred as barcodes, however, those visual codes are usually built by regular polygons instead of bars. Between these two groups, there are stacked codes as well, that can be considered as multiple 1D barcodes aligned next to each other perpendicularly to the axis carrying the data. In this paper, we restrict ourselves to the Code 128, EAN-13 and UPC codes.

Barcodes, like OCR-friendly fonts, aim the easy, reliable and automatic reading of embedded data with minimal or no human intervention. Those codes usually carry a short, unique numeric or alphanumeric product code that can be used at postal services, tickets of various type, or supermarkets. 2D barcodes usually carry more alphanumeric data, like URL-s, email addresses or character sequences for many different purposes like coupon codes or advertisements.

We have to take two steps to regain the data embedded in the visual code. First, we locate the barcode, recognize its size, orientation, and usually apply transformations, like noise reduction, sharpening, normalization, and correction of distortions. After this step, the processed image piece containing the code is passed to the detector that looks up the pattern for valid character data. When proper localization and preprocessing is applied, this step is relatively straightforward since characters are easily recognizable thanks to the maximized

hamming-distances between characters in most cases. Furthermore, most barcode patents provide additional, redundant data for error-correction purposes. Localization step has more difficulties due to the variety of code types, cameras and scenarios.

The basic idea of barcode localization is the scan-line method [1, 2]. It imitates the process how a laser scanner sweeps through an image, scanning lines and producing one-dimensional intensity profiles. The main idea is to find peak locations in blurry barcode models, then thresholding the intensity profile adaptively to produce binary values. Decoding part takes place on those processed profiles to recognize character sequence with various approaches. One example is a bayesian decoding, with 4 directional scan-lines and entropy filtering [3]. Speed is an advantage of this method, therefore it is appropriate for mobile phone application as Tekin et al. proposed in later works [4].

Several works assume that barcodes are aligned along the axes [5, 6], or they are already localized [7]. Those works propose accurate algorithms for decoding, that can be paired with proper localization approaches to solve the task of efficient barcode reading. Gallo et al. [6] also uses gradients, which also can be used for localization [8].

Algorithms with morphology [9–12] use the combination of basic morphological operations like erosion and dilation. White blobs on processed images show the possible barcode locations. Further processing, like segmentation and filtering of small blobs are required on these images. It can be used on both 1D and 2D barcodes. This group is considered as the slowest method of barcode localization, since morphological operations usually require convolution, which is a bottleneck at processing high-resolution images.

Multiple works propose Hough transformation for the localization step [9, 13, 14]. Those papers propose standard or probabilistic Hough transformation to extract lines or line segments of the image, and make further decisions by line lenght, proximity to each other, and orientation. The transformation needs an edge map as input, usually produced by convolution using Canny or Sobel kernels. Transforming the edge points to Hough space is also a time-consuming task, making the method comparable in runtime to morphology-based ones.

Tuinstra et al. uses both morphology and Hough transformation in the localization step, with bandpass filter and thresholding. They also experimented with gradient magnitude map and hit-or-miss transformation, as well as valley tracing, 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, recognize the ends of the valleys (bars).

Image partitioning to uniform tiles is also a wide-spread idea of pattern recognition, that also can be used as a base of barcode localization [14, 15]. 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 approach is partitioning the image into 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 of similar values defines regions of interest, representing barcodes with high possibility. This approach can be used for fast localization, however, image features have to be chosen properly to maintain accuracy at a reasonable level.

Our work involves overlapped partitioning with a modified idea of scan-line analysis and replacement of the sweeping lines with circular pattern. A reliable measure is obtained that way, indicating possible barcode regions. The output of the localization process, cropped regions can be passed to decoding algorithms [5,7], thus making applicable solutions for scenarios that require fast and accurate processing.

2 The Proposed Method

The proposed localization algorithm starts with preprocessing. First, we convert the image to binary by thresholding. With images having even illumination, a simple threshold is satisfactory, otherwise adaptive threshold is required. Binary images are divided into square tiles with overlapping by half the tile size. Each tile is processed individually at first, and a measure is assigned by evaluating pixels in a circular pattern, with the tile size as diameter. A one-dimensional profile is obtained, which has zero-crossings of various densities. After this step, the circle forming the intensity profile is divided into four equal sized quadrants by density of zero-crossings. Image parts representing a barcode have equally low or high number of crossings at opposite quadrants, and significant difference in the number of crossings at neighbouring quadrants. "Wild" and "calm" quadrants are separated, as shown in Figure 1.

Wild quadrant centers of a tile are placed by maximum of the following formula:

$$D(T) = \max\left(\frac{1}{2q}\left((Z_{w1} + Z_{w2}) - (Z_{c1} + Z_{c2})\right)\right) \tag{1}$$

where Z holds for zero-crossing, w and c for wild and calm zones respectively, and q for quadrant size in pixels. D shows the direction of the first wild quadrant in the best fitting placement of quadrants along the circle. According to symmetry of the circular pattern, we use $D - 180°$ where D is greater than 180 degrees.

Connecting the wild zone centers, defines a dominant direction of the possible barcode texture in each tile, and neighbouring tiles will have the same dominant direction when containing barcode parts. We also have to take the strength of the dominant direction into consideration. The maximum at Eq. 1 is small at tiles having very small or high level of intensity entropy, and there can be multiple possible dominant directions as well. According to image flaws, noise, reflections and distortions, a level of tolerance should be introduced at neighbouring tiles. Within that range, dominant directions must be considered equal. Furthermore, having all neighbouring tiles with similar dominant directions and different from

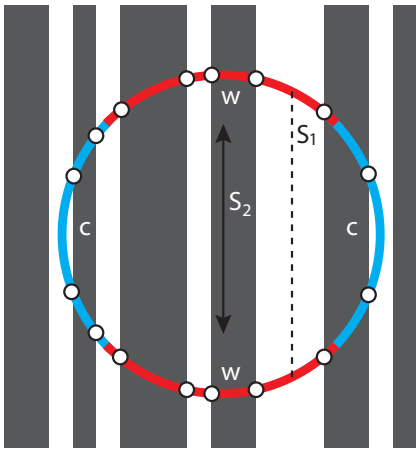


Fig. 1: Zones and symmetries of the circular intensity profile. Wild (w) and calm (c) zones, symmetry at pixel level (S_1) and between quadrants S_2

the examined tile should suppress the assigned direction and make the alignment be reconsidered.

Symmetries also form a well-traceable feature for searching barcode parts. The level of symmetry at pixel level and between quadrants are computed with Eq. 2 and Eq. 3, respectively. These symmetries can only be detected at wild quadrants due to the axial symmetry of barcode patterns.

$$S_1(T) = \frac{1}{q} \sum_{i=0}^{q} 1 - |W_1(i) - W_2(q-i)| \tag{2}$$

$$S_2(T) = 1 - \frac{|Z_{w1} - Z_{w2}|}{q} \tag{3}$$

We look for the smallest difference in dominant direction at T and its 4-adjacent neighbourhood T^* as expressed by Eq. 4, and the final measure assigned to each tile is by Eq. 5. We use δ as the weight of D_m, since neighbourhood similarity is more important than symmetry in tiles ($\delta = 2$ is our proposed value).

$$D_m(T) = 1 - \frac{1}{90} \min\{U \in T^* \mid |D(T) - D(U)|\} \tag{4}$$

$$V(T) = \frac{1}{4} \left(\delta D_m(T) + S_1(T) + S_2(T)\right) \times D(T) \tag{5}$$

The feature matrix formed by $V(T)$ values. The thresholded matrix is further analysed via Connected Component Labelling. Components smaller than half of the expected barcode size are dropped. The bounding boxes are calculated for remaining blobs, and image regions are passed to the decoder. Convexity of the

returned blobs could also be examined, however, it would decrease localization performance. Since low rate of false positives are not crucial, a better solution is to let the decoder decide whether concave regions contain valid barcode. It is also possible to decrease the size of the passed image parts by defining rotated bounding rectangle that fits more tightly to the possible barcode.

Scanning the image parts in circular pattern instead of a set of lines with fixed number and orientation is more efficient, because it is fully direction-independent, uses symmetry to reinforce decision, and processes less pixels.

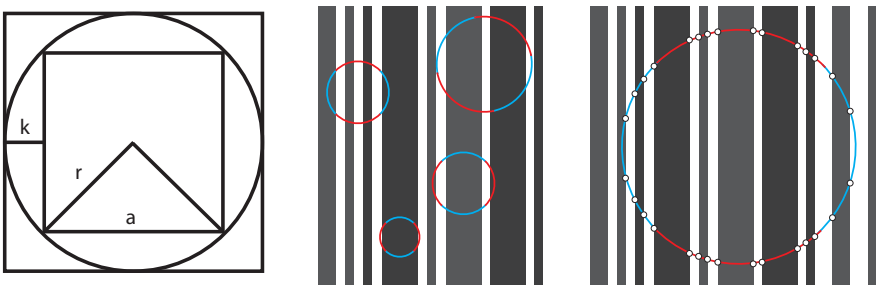
Downsampling of high resolution images having low noise level, good contrast and crisp edges is allowed and recommended, but all setups should be examined for minimum resolution of the input for accurate localization. In idealistic images, like artificially rendered ones, 1 unit[1] as 1 px is satisfactory, however, at least 2 px as unit size is recommended for real-life applications. Strict industrial setups can be considered as idealistic, as they have even illumination and quality camera. The proposed downsampling to obtain desired unit size is nearest neighbour interpolation, since it preserves hard edges of barcodes.

The optimal tile size is about 25 units. UPC-A codes have a total width of 95 units and a height of 78 units. Upper bound to tile size is 36 units, since larger tiles would not fit along the smaller dimension of the code, thus reducing accuracy dramatically, while lowering the minimum number of tiles required to form a barcode to compensate the effect, would raise false positives. The worst case scenario is digits 6 and 3 are besides each other, which causes 1-4-4-1 width pattern. As it is illustrated in Fig. 2a, wild zone width is $\sqrt{2} \times r$, and calm zone width is $1/2 \times r(2-\sqrt{2})$. At Fig. 2b, some badly chosen tile sizes are present. The zones are eventually defined well (top left circle), but the same sized circle fails at thicker bars (bottom right circle). To make sure about the worst aligned tile is able to indicate sufficient zero-crossings to align wild and calm zones correctly, tile size has to be at least 13 units. According to these bounds and letting the maximum variance in expected barcode size while keeping high accuracy, 24–26 units is the recommended tile size (Fig. 2c). Computing with an optimal scenario where calm zones contain no zero-crossings at intensity values, leads to the same conclusion, since width of k using 26 units as tile size is $(26 - 13\sqrt{2})/2$, which is about 4 units, the thickest bar width of UPC-A patent. However, zero-crossings in the calm zone are not problematic over the lower bound of tile size.

3 Evaluation

Since we have not found many official barcode detection test image databases, we took about 100 images of grocery product barcodes with a Nokia N95 smartphone camera. We downsampled those images to 640×480 px with bilinear interpolation. Minor reflections, blur, scratches and distortions were present in these images. We also found one barcode image database for comparative assessment,

[1] In patents, barcode dimensions are often expressed in units, a resolution-independent representation relative to the smallest bar width.



(a) geometry of a tile and zone widths (b) badly chosen tile sizes (c) well-chosen tile size, 26 units, worst case scenario

Fig. 2: Tile geometry and size

which was created by Tekin and Coughlan[2]. Ground truth to those images had been made manually, without marking the quiet zones and the digits that belong to the code.

There are several barcode detection software and frameworks on the market, like the DTK Barcode Reader SDK[3], BC Tester[4], and Barcode Recognition SDK of DataSymbol[5], however, they do not indicate the applied theory behind their detection mechanism.

Runtime evaluation is performed on a computer with Intel Core 2 Duo 3.00GHz CPU. 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 standard image processing and manipulation.

For comparing the effectiveness of the proposed methods, we used the most common measures like precision, recall, accuracy and F-measure. The values are based on the Jaccard index

$$J(G, D) = \frac{\sum_{x,y}(G(x, y) \wedge (D(x, y))}{\sum_{x,y}(G(x, y) \vee (D(x, y))} \tag{6}$$

where G and D give binary 0–1 values based on the pixel intensity of the ground truth and the binarized detector output images respectively.

The average performance indicators of the detectors are shown in Table 1.

A more accurate method for finding the dominant pattern direction is the fast Hough transformation. It provides the most reliable results, however, it slows down the localization algorithm to a level comparable to morphological operations.

[2] http://www.ski.org/Rehab/Coughlan_lab/Barcode
[3] http://www.dtksoft.com/
[4] http://www.bctester.de/
[5] http://www.datasymbol.com/

Table 1: Average detection performance of the proposed method

Algorithm	Precision	Recall	Accuracy	F-measure	Runtime
Tuinstra [2]	57.08 %	85.29 %	84.19 %	48.39 %	160 ms
Juett et al. [11]	34.26 %	94.08 %	72.76 %	36.13 %	230 ms
Katona et al. [12]	69.60 %	84.67 %	86.61 %	61.86 %	80 ms
Proposed	23.80 %	92.70 %	71.33 %	25.39 %	50 ms

Only minimal trigonometric calculations are necessary at the initialization step, since discrete relative coordinates to the points of the sampling circle have to be calculated only once, and can be applied by offsetting with center positions. Furthermore, if a map is stored with the corresponding angle to each relative coordinate, calculation of the dominant direction does not require arcus tangent.

4 Concluding Remarks

We presented a novel method for barcode localization and evaluated its performance on a set of real-life example images. Results show that uniform partitioning of an image, and scanning parts in circular pattern leads to a trustworthy approach for barcode localization.

In industrial setups, runtime can be further decreased by parallel execution. Tile evaluation can be fully parallelised.

Our future work includes extending the presented approach to 2D barcodes by finding strong features having high sensitivity to barcode patterns.

References

1. Robert Adelmann. Toolkit for bar code recognition and resolving on camera. In *Phones – Jump Starting the Internet of Things. In: Informatik 2006 workshop on Mobile and Embedded Interactive Systems*, 2006.
2. Timothy R. Tuinstra. *Reading Barcodes from Digital Imagery.* PhD thesis, Cedarville University, 2006.
3. Ender Tekin and James M. Coughlan. An algorithm enabling blind users to find and read barcodes. In *Applications of Computer Vision (WACV), 2009 Workshop on*, pages 1–8, dec. 2009.
4. Ender Tekin and James M. Coughlan. A mobile phone application enabling visually impaired users to find and read product barcodes. In *Proceedings of the 12th international conference on Computers helping people with special needs*, pages 290–295, Berlin, Heidelberg, 2010. Springer-Verlag.
5. Ender Tekin and James Coughlan. A bayesian algorithm for reading 1d barcodes. In *Proceedings of the 2009 Canadian Conference on Computer and Robot Vision*, CRV '09, pages 61–67, Washington, DC, USA, 2009. IEEE Computer Society.
6. Orazio Gallo and Roberto Manduchi. Reading 1d barcodes with mobile phones using deformable templates. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(9):1834–1843, September 2011.

7. Kongqiao Wang, Yanming Zou, and Hao Wang. Bar code reading from images captured by camera phones. In *Mobile Technology, Applications and Systems, 2005 2nd International Conference on*, page 6, nov. 2005.
8. R. Shams and P. Sadeghi. Bar code recognition in highly distorted and low resolution images. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 1, pages I–737 –I–740, april 2007.
9. 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)*, pages 300–306, 2012.
10. 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.
11. Xiaojun Qi James Juett. Barcode localization using bottom-hat filter. *NSF Research Experience for Undergraduates*, 2005.
12. Melinda Katona and László G. Nyúl. A novel method for accurate and efficient barcode detection with morphological operations. In *The 8th International Conference on Signal Image Technology (SITIS 2012)*, pages 307–314, 2012.
13. Sherin M. Youssef and Rana M. Salem. Automated barcode recognition for smart identification and inspection automation. *Expert Systems with Applications*, 33(4):968–977, 2007.
14. 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.
15. Pavel Šimurda. Barcode localization in image. In *Information Sciences and Technologies Bulletin of the ACM Slovakia*, volume 3, pages 55–56, 2011.