Barcodes Detection and Description

Cláudia Rodrigues
up201508262@fe.up.pt
João Barbosa
up201406241@fe.up.pt
Manuel Curral
up201202445@fe.up.pt

Faculdade de Engenharia
Universidade do Porto
Porto, Portugal

Abstract

Barcode is a simple and widely adopted method of representing machine-readable data in a visual form. Scanners do a good job at reading barcodes and transmitting its information, but they're not readily available to every person. Thus, a software-based scanner would allow new possibilities for their uses.

We propose a new method to find the barcodes' position and describe its bars distribution by analyzing both contours and edges - robust against multiple problems that arise from camera scannings, such as illumination, rotation, perspective, cluttered backgrounds and multiple barcodes in one image.

1 Introduction

Barcodes are an old technology, but widely used due to the improvement in speed and accuracy of data entry. Even though barcodes are used with great success in warehouses, pharmacies or retail stores, they usually need appropriate hardware to be efficient. Allowing barcodes to be read from everyday technology would open new ways to benefit from them, such as accessing more information about a piece of art in a museum or tracking information about the food we eat.

The key problem is that, in an everyday scenario, many factors can influence the image we can capture and thus scan [1]. Cluttered backgrounds can make edge detection difficult to analyze, different lighting can cause gradient detection to be highly irregular, etc. Using common barcode placement guidelines and assumptions, our localization method uses contours and line detection to track the barcode's region in the image.

According to Standard Guidelines for Barcode Placement from GS1, a barcode must always be placed with a fence (vertical) or ladder (horizontal) orientation and should always be surrounded by a Quiet Zone (white background); there should be no other graphics encroaching on the barcode or its Quiet Zone; it is also not advised to truncate the barcode i.e. reduce its height relative to its length [5]. We can see from everyday objects that this isn't always the case, so it has to be accounted for.

2 Barcode Localization

In order to explain what the method does, assume an image with a barcode in the fence orientation. The same procedure applies on a ladder orientation by rotating the image 90 degrees. Skewed barcodes can also be detected: until 45^o we assume its in the fence orientation; from 45^o to 90^o the ladder orientation should be used.

The algorithm is divided into two parts: it starts by focusing on finding the black bars, so it picks vertical features using contour detection. In the second step, it groups regions according to the amount of features previously found, and runs Hough transform to detect if the number of lines is above a threshold. If it is, then the region is considered to have a barcode.

2.1 Pre-processing

We start by applying Contrast Limited Adaptive Histogram Equalization (CLAHE) to improve contrast in the image and, therefore, enhance the differences between the black bars and their white background.

2.2 Segmentation

Following the previous step, an adaptive thresholding algorithm is applied to inversely binarize the image, that is, low gray levels (such as the black bars) become white and high gray levels become black.

After the threshold is applied, a small opening morphological operation in the vertical direction is used to remove some noise in the image that might hinder the contour detection.

2.3 Contour Detection

In this step, we focus on finding shapes that would fit an expected black bar of a barcode. The following assumptions are used:

- The black bar has a height of at least 20 pixels. Consequently, its area must also be at least 20 pixels.
- Considering the average picture taken by a user, the black bar also has a limit of 350 pixels to its maximum height.
- The height to width ratio of the contours is at least 2:1, that is, the height must be at least the double of the width.

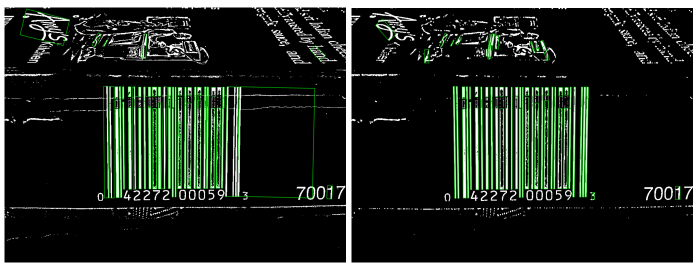
Contours that don't fit these criteria are labelled as "bad" and therefore removed from the image. This step will leave us with vertical contours, eliminating much of undesired noise surrounding the position of the barcode.

(a) (b)

Figure 1: Contour detection using predefined restrictions removes many undesired features of the image: (a) before applying contour filtering; (b) after applying contour filtering.

The contours are analyzed in descendent order of area. This allows the algorithm to stop after it finds a feature with less than the minimum area (20 pixels), improving its overall performance.

Notice also how the small opening operation referenced in section 2.2 makes a noticeable difference in the detection of some contours.



(a) (b)

Figure 2: Green bars represent contours being kept in the image: (a) No noise removal (some contours might be incorrectly detected); (b) With noise removal.

2.4 Target Detection

After removing undesired contours in the image, the next step is about trying to find possible targets, that is, regions with a high concentration of vertical contours.

We start by applying a closing morphological operation with a structuring element spanning mainly horizontally, but also vertically. The vertical direction helps further connecting contours of the barcode if the latter is slanted.

After the morphological operation, the image is left with blobs that are possible barcode regions. Again, contour detection is used to detect these blobs. As in the previous step, some assumptions are now made about the possible targets:

- The blob must have an area of at least 2000 pixels.
- The width to height ratio must be less than or equal to 3:1.
- The height to width ratio must be less than or equal to 2:1.

As in the contour detection step, blobs that don't respect these criteria are classified as "bad" and removed from the image. The algorithm also stops after it finds an area of less than 2000 pixels in order to improve computational speed.

(a) (b) (c)

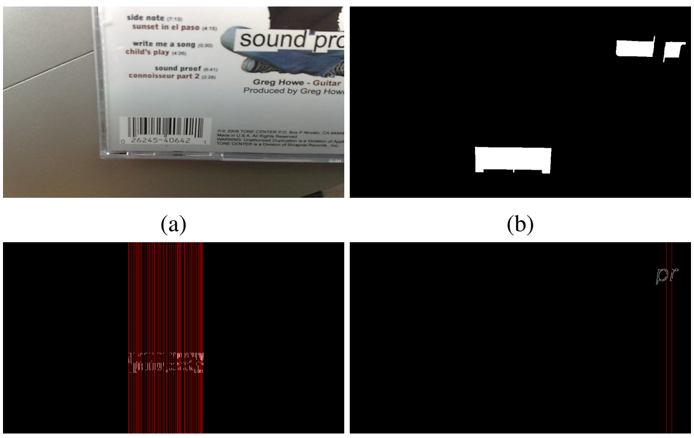
Figure 3: Target creation and filtering: (a) Image with contours; (b) After applying a closing morphological operation; (c) After filtering according to criteria.

2.5 Barcode Confirmation

After the target detection step, we are left with regions that are likely to have a barcode. However, there may still be some undesired blobs.

The following and final step is about iterating each of these blobs and decide if it should be considered a barcode or not. Our method to do so is about detecting and counting the number of lines in that region. In order to do that, for each of the regions we apply the Canny edge detection algorithm and then the Hough transform to check if the number of lines in the image is above a certain threshold [3] [4].

This method also allows to pick up multiple barcodes, if that is the case in the image.



(a) (b)

(c) (d)

Figure 4: Regions with barcodes will have a high number of lines detected: (a) Original image; (b) Filtered targets; (c) Barcode region; (d) Non-barcode region.

3 Barcode Decoding

After the region is identified, the barcode is decoded. We use the term "decoding" here to refer to separating the black bars from the whites spaces describing their widths (and not the decoding of the barcode's content).

In this step, the bars are reported with their sequence and width. A scan line also displayed along the width of the code.

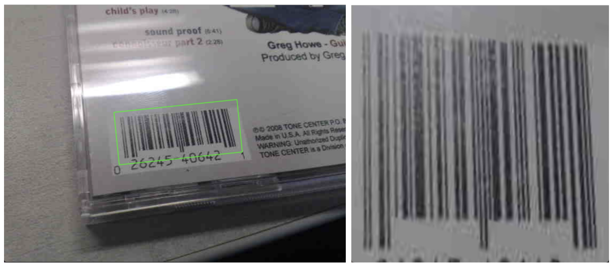
3.1 Isolation

First the barcode area is isolated, using the region found in section 2. The morphological operation dilation was applied to ensure the barcode was not cut due to its orientation, since the region was an estimation.

3.2 Alignment

In this step, we apply the alignment of the image, so that the barcode can be seen in front view. The contour corners of the barcode region are used to apply this perspective transformation.

As the result, the barcode is vertically displayed in a square image with fixed dimensions. The digits are removed using a small offset from the bottom.

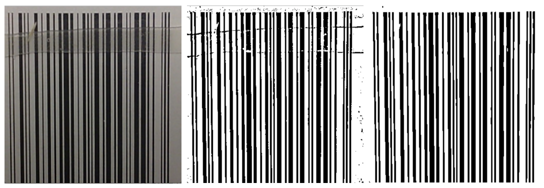


(a) (b)

Figure 5: Image alignment to get the barcode in frontal view: (a) Original image with barcode area identified; (b) after applying perspective transformation.

3.3 Segmentation

We start by using CLAHE to improve the constrast between the black bars and background. Then, we applied an adaptive threshold to binarize the image. In order to remove the noise, open and close morphologic operators were applied, mainly in the vertical direction.

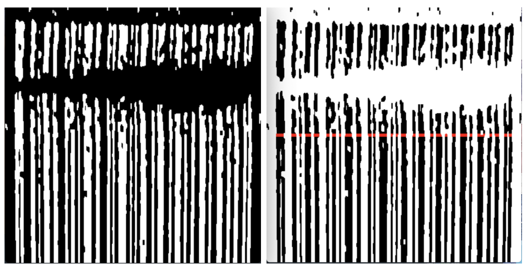


(a) (b) (c)

Figure 6: Barcode segmentation: (a) Original image; (b) After applying binarization; (c) After removing noise.

3.4 Bars identification

The image is inverted and two lines are drawn parallel to the barcode, where the intersection creates the scan. The line with more intersections is chosen, which can minimize some error in the case where the bars are not completely filled in the previous step, for example, due to some specularity.



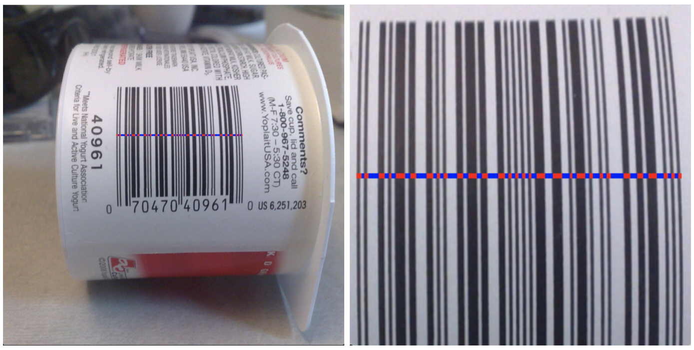
(a) (b)

Figure 7: Barcode scan: (a) Inverse binarization; (b) Black bars are correctly identified.

3.5 Scan line

With the scan line in the bars obtained in the previous step, we extracted the scan line over the spaces.

We plot a colored line along the width of the code, composed of red segments over the bars, and blue segments over the spaces.



(a) (b)

Figure 8: Scan result: (a) Original image; (b) Perspective barcode.

3.6 Decoding

The last step is to describe the bars. We applied contour detection to the scan line in the bars, and to the scan line in the spaces. Each segment was identified and its length was calculated. Dividing this length with the total scan line generates its percentage of width.

4 Results

The algorithm was tested in a dataset of 27 images plus others captured by videocamera, containing multiple scenarios, from cluttered backgrounds to specular light. Out of the 27 images, we were able to successfully detect 25 (the other 2 images still contained the barcode in the region, but its isolation was not possible due to noise surrounding it).

When the barcode region was correctly identified, we were able to describe the bars and spaces with success. Among the different conditions tested, accuracy may decrease slightly when the barcode is not focused enough.

(a) (b)

Figure 9: Some results. (a) barcode detection; (b) barcode description.

5 Conclusions and Future Work

We've presented a solution for barcode detection and description, robust in conditions where:

- There are multiple barcodes.
- The barcode is skewed or in perspective.
- The barcode is occupying a small part of the image.
- The image is blurred.
- There is strong illumination and even specular light.
- The background is cluttered.

It is particularly difficult to detect the barcode's region when its placement doesn't respect standard guidelines. Specifically, we found problems when the barcode has a very small Quiet Zone (white background) or if there are a lot of graphics around it, making the removal of the barcode's surrounding a hard task. Both situations should be avoided by product manufacturers [5]. A possible solution for this problem is, during the Hough transform (section 2.5), remove lines that are too tall and too short compared to the average - this wasn't possible in our work due to limitations in the framework used. Still, the algorithm was able to surpass these issues in most images.

Image conditions can affect barcode decoding, especially when the image is blurry. Most modern barcode scanning technologies are not capable of handling out-of-focus image blur and rely on camera systems to capture good quality, well-focused images. Nevertheless, there are algorithms that can estimate and remove blur when recognizing barcodes in images [2].

References

[1] S. Han M. Yi Z. Zhang C. Zhang, J. Wang. Automatic real-time barcode localization in complex scenes, 2006. Proc. of the International Conference on Image Processing (ICIP).

[2] Man H. Jia H Chen, L. On scanning linear barcodes from out-of-focus blurred images: A spatial domain dynamic template matching approach, 2014.

[3] A. Otero R. Muniz, L. Junco. A method and means for recognizing complex patterns, 1999. US Patent 3069654.

[4] A. Otero R. Muniz, L. Junco. A robust software barcode reader using the hough transform, 1999. Proc. International Conference on Information Intelligence and Systems.

[5] GS1 US. Guidelines for bar code symbol placement, 2007.