TOWARD REAL TIME EYES-FREE BARCODE SCANNING ON SMARTPHONES IN VIDEO MODE

Aliasgar Kutiyanawala[1] Vladimir Kulyukin[1] John Nicholson[2]
[1]Department of Computer Science, Utah State University. [2]Department of Computer Science and Information Technology, Austin Peay State University

INTRODUCTION

In our previous research [1], we have shown that visually impaired (VI) individuals can shop independently by scanning MSI barcodes on shelves and UPC barcodes on products. This system was called ShopTalk and used a hand-held barcode scanner, a shoulder-mounted keypad and headphones connected to an ultra-portable OQO computer. ShopMobile-II [2, 3] is the next version of this system that reduces the system's hardware complexity by allowing VI users to shop independently using only a smartphone.

ShopMobile-II has three software modules – an eyes-free barcode scanner, an OCR engine and remote guidance. This paper describes our barcode scanning algorithm that operates in video mode on the Google Nexus One Android 2.2 smartphone. Two pilot experiments are presented: the first one, conducted in a grocery store evaluated the contributions of the algorithm's modules; the second one, performed in a laboratory with two VI users, evaluated the effectiveness of the algorithm in finding UPC barcodes on various grocery products.

EYES-FREE BARCODE SCANNER

RedLaser [4] and ZXing [5] are two applications for scanning barcodes on smartphones. However, they require users to carefully align the camera with the barcodes prior to decoding and cannot decode MSI barcodes. Our algorithm is designed to find both UPC and MSI barcodes in real video mode without prior camera alignment [6].

The eyes-free barcode scanning algorithm is comprised of three modules – interactive camera alignment loop, barcode localization, and barcode decoding. The barcode scanning algorithm operates in video mode. Images are taken continuously and processed by the barcode localization and decoding modules. If a barcode is decoded, the user is notified through text-to-speech.

Interactive Camera Alignment Loop

We have previously conducted a lab study with one VI participant and four blindfolded sighted participants, where participants had to find, retrieve and verify products from a simulated shopping aisle [2]. The participants scanned MSI barcodes on shelves to find product locations and UPC barcodes on products to identify them. MSI barcodes are a type of linear 1-D barcodes that are used mostly for inventory control, and marking storage containers and shelves in warehouse environments [7]. To scan a barcode, participants would first align the camera with the product or shelf and then slowly move the camera away. The VI participant would frequently misalign the camera in the pitch and yaw planes as he moved it away from barcodes. This misalignment caused skew distortions in barcode images, which resulted in several decoding failures.

The interactive camera alignment module is designed to minimize skew distortions by keeping the camera aligned with the barcode in the pitch and yaw frames. The shopper starts the loop by pressing the touch screen. The system captures the phone's orientation sensor's readings of the pitch and yaw planes for subsequent reference. The system takes these readings as the shopper moves the camera away from the barcode and compares them with the reference readings. When the readings deviate from the reference readings, the phone begins to vibrate until the camera is realigned.

Barcode Localization

The area of the barcode region in the image varies with the distance between the barcode and the camera. If the camera is held close to the barcode, the barcode region occupies a large area in the image and the barcode decoding stage can easily decode the barcode. However, if the camera is held at a distance to the barcode, the barcode region is small and there is a possibility of other components, such text and graphics, being alongside the barcode, which may cause a detection failure. Localizing and segmenting possible barcode regions prior to decoding increases the probability of accurate decoding.

A barcode is a small homogeneous region consisting of alternate black and white lines. We define two properties – *alternating frequency* and *vertical continuity* that characterize a region as a potential barcode. If a line is drawn horizontally across the barcode, alternating frequency is the number of black to white and white to black transitions along that line. Vertical continuity is the continuity of black and white lines along vertical lines. Vertical continuity is estimated by the similarity between two parallel horizontal lines, separated vertically, placed over the region. The longest common subsequence is used as the similarity measure.

The first step in the barcode localization is to scale the image down to the 320 x 240 resolution for efficiency. This scaled image is passed through a line filter presented in [2], which allows vertical lines to pass through and filters out everything else. The filtered image is scanned in a rasterized pattern to look for areas with high alternating frequencies and vertical continuities. These are translated into the corresponding areas of the original image and segmented from it.

Barcode Decoding

Our barcode decoding algorithm uses scanlines and consists of three main procedures – strip generation, strip processing, and scanline decoding. The strip generation procedure selects horizontal and diagonal *n*-pixel wide strips in the image. The strips are converted to one-pixel wide scanlines. This conversion is achieved through three methods – luminance based method (LM), two-level binary method (TLB) and a two-level binary method with alternating frequency filter (TLB-AF).

LM populates the scanline with the central ($n/2^{th}$) row of the strip, and each pixel in the scanline $p_i \in [0, 255]$. TLB binarizes the strip using a modified Niblack filter [8] before populating the scanline with the central row of the strip. Each pixel in the scanline pi $\in \{0, 255\}$. TLB-AF takes this process a step further by filtering out areas with low alternating frequencies in the binarized strip before populating the scanline.

The processed scanline is now converted to a line-widths representation $W = \{w_1, w_2 \ldots w_k\}$. To obtain *W*, strings of consecutive zeros and ones in the scanline are replaced with their run lengths. A separate data structure $C = \{c_1, c_2, \ldots, c_k\}$ records the color associated with each line-width w_i. For example, if the scanline $S = \{255, 0, 0, 0, 255, 255\}$, $W = \{1, 3, 2\}$ and $C = \{1, 0, 1\}$.

Table 1: UPC barcode symbology in line-widths representation.

Pattern / Digit	**Template of line widths T^m**
Start Pattern	111
Middle Pattern	11111
End Pattern	111
Digit 0	3211
Digit 1	2221
Digit 2	2122
Digit 3	1411
Digit 4	1132
Digit 5	1231
Digit 6	1114
Digit 7	1312
Digit 8	1213
Digit 9	3112

A UPC barcode consists of two sets of six digits *D* enclosed by start (*S)*/end (*E)* patterns and separated by a middle pattern (*M)* as *SDDDDDDMDDDDDDE* [9]. The start, middle and end patterns as well as the digits are encoded by a series of alternating black and white lines of varying widths. Table 1 shows the

UPC barcode symbology in the line-widths representation. To decode the barcode in the scanline *W*, we find the start pattern index (*s*) and the end pattern index (*e*) within *W* using the following equations:

$$s = \arg\max(w_{i-1} - std(w_i, w_{i+1}, w_{i+2}))$$
$$e = \arg\max(w_{i+3} - std(w_i, w_{i+1}, w_{i+2}))$$

where, $c_i = 0$.

Each digit *d* is encoded by four lines and the index *i* of the j^{th} digit d^j_j within *W* can be found as:

$$i = s + (j-1) \times 4, if\ j \le 6$$
$$i = s + (j-1) \times 4 + 5, if\ j > 6$$

If T^m represents the template of the digit *m* in Table 1., the value of the j^{th} digit d^j can be found as follows:

$$d^j = \arg\min(\sqrt{\sum_{k=0}^{3}(w_{i+k} - T_k^m)^2})$$

where, $0 \le j \le 12$ and $0 \le m \le 9$.

The twelfth digit is a checksum digit to verify accurate decoding.

EXPERIMENTS

The objective of the first experiment was to find the contribution of barcode localization as well as the contributions of the three strip processing methods. Our claim is that barcode localization prior to decoding increases the decoding rate. To test this claim, a database of 68 product images, which included boxes, cans, and bottles, was obtained in a grocery store with the Google Nexus One smartphone with Android 2.2. We then logged the number of barcodes decoded in the scans with and without the localization stage. As Fig. 1 shows, localization prior to decoding resulted in an increase of 14 (20.59%) decoded barcodes.

As mentioned earlier, the three methods are used to convert n-pixel wide strip to one-pixel wide scanlines. We wanted to test the contributions of each method to decoding. As shown in Fig. 2, we found that the LM decoded the most barcodes (53), followed by TLB (26) and finally TLB-AF (15). Only one barcode out of the 15 decoded by TLB-AF is not decoded by the other methods. Hence, this method can be eliminated without a significant impact on the overall decoding rate.

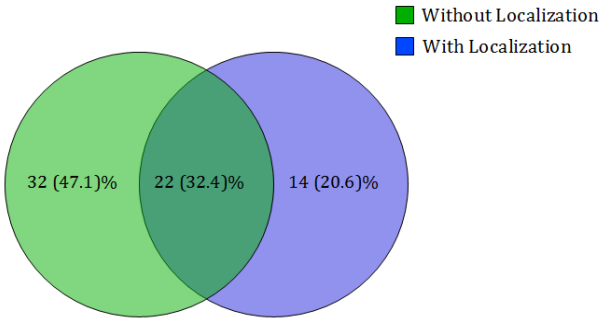

Figure 1. Contributions of the barcode localization stage.

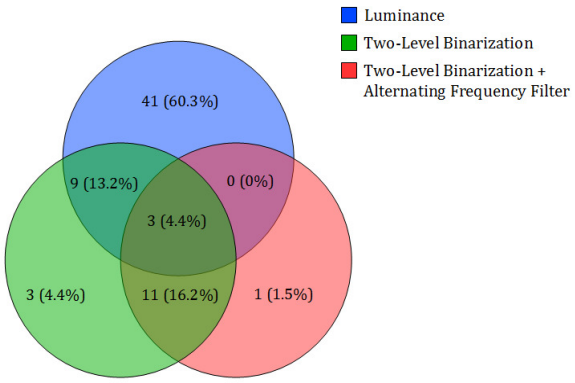

Figure 2. Contributions of the different strip processing methods.

To test the accuracy of barcode recognition in video mode, we conducted two single subject experiments at the Smith Kettlewell Eye Research Institute in San Francisco, CA. The software was implemented on the Google Nexus One smartphone with Android 2.2. Both subjects were completely blind staff electrical engineers. One had a cellphone; the other never owned or used one. The camera alignment module was not tested. The camera operated in video mode.

The experiment consisted of a tutorial and an actual test. In the tutorial, each subject was given two sample products (a plastic bottle and a juice box) and was shown how to move the camera along the surface of the product. The tutorial continued until each subject could detect the barcodes on both products without the experimenter's help. The first subject's tutorial took approximately ten minutes. The second subject's tutorial took approximately eight minutes. During the test, each subject was given ten products: a cereal box, a small tea box, two small juice bottles, a small milk carton, a Pringles tube, a toothpaste box, a larger juice bottle, a small water bottle, and a yogurt cup. The detection time for each product was recorded on the phone. When the subject could not detect the barcode for over five minutes, the detection was considered a failure and the subject was given the next product. The order of the products was randomized for each subject.

Table 2: Results of barcode scanning experiments with VI participants.

Product	**Participant 1**		**Participant 2**	
	Time Taken	**False Positives**	**Time Taken**	**False Positives**
Cereal box	33	0	40	0
Tea box	140	0	100	1
Small juice bottle 1	15	0	Fail	0
Small juice bottle 2	142	0	151	0
Milk carton	56	0	51	1
Pringles	37	0	111	0
Toothpaste	184	0	Fail	0
Large juice bottle	125	0	140	0
Water bottle	87	0	121	0
Yogurt cup	17	0	41	0

The average barcode recognition times were 83.6 seconds and 93.4 seconds for subject 1 and 2, respectively. Subject 2 had two failures where the software failed to detect the barcode in five minutes. There were also two false positives when the software detected a barcode over printed text.

In informal feedback collected from both subjects after the experiments, they indicated a preference for run time image alignment signals when the camera is misaligned with the object. They also said that the camera should work over a wider range of positions and the software must operate much faster.

CONCLUSION

Our experiments indicate that smartphones can be used for real time eyes-free barcode scanning in video mode. Barcode localization was observed to improve the decoding rates by as much as 20%. The TLB-AF can be eliminated without a significant reduction in barcode decoding rates. Two VI participants were able to successfully scan most UPC barcodes on various grocery products using our algorithm.

REFERENCES

[1] J. Nicholson, V. Kulyukin, and D. Coster, "ShopTalk: independent blind shopping through verbal route directions and barcode scans", The Open Rehabilitation Journal, ISSN: 1874-9437 Volume 2, 2009, pp. 11-23, DOI 10.2174/1874943700902010011.

[2] V. Kulyukin, and A. Kutiyanawala, "Eyes-free barcode localization and decoding for visually impaired mobile phone users", Proceedings of the 2010 International Conference on Image Processing, Computer Vision, and Pattern Recognition, 2010.

[3] V. Kulyukin, and A. Kutiyanawala, "From ShopTalk to ShopMobile: vision-based barcode scanning with mobile phones for independent blind grocery shopping", Proceedings of the 33-rd Annual Conference of the Rehabilitation Engineering and Assistive Technology Society of North America, June 2010, Las Vegas, Nevada.

[4] Occipital, LLC. RedLaser http://redlaser.com/

[5] The Zebra Crossing Barcode Decoding Library. http://code.google.com/p/zxing/

[6] Eyes-Free Barcode Scanning in Video Mode on Google Nexus One Smartphone.
http://www.youtube.com/user/csatlusu#p/u/0/-aiT-Zan7AE

[8] Wikipedia, MSI barcode.
http://en.wikipedia.org/wiki/MSI_Barcode.

[8] W. Niblack, "An introduction to image processing", Prentice-Hall, Englewood Cliff, NJ, pp. 115-1 16, 1986.

[9] Wikipedia, UPC barcode.
http://en.wikipedia.org/wiki/Universal Product Code.