Vision-Based Localization and Text Chunking of Nutrition Fact Tables on Android Smartphones

Vladimir Kulyukin[1], Aliasgar Kutiyanawala[1], Tanwir Zaman[1], and Stephen Clyde[2]
[1]Department of Computer Science, Utah State University, Logan, UT, USA
[2]MDSC Corporation, Salt Lake City, UT, USA

Abstract—Proactive nutrition management is considered by many nutritionists and dieticians as a key factor in reducing and controlling cancer, diabetes, and other illnesses related to or caused by mismanaged diets. As more and more individuals manage their daily activities with smartphones, smartphones have the potential to become proactive diet management tools. While there are many vision-based mobile applications to process barcodes, there is a relative dearth of vision-based applications for extracting other useful nutrition information items from product packages, e.g., nutrition facts, caloric contens, and ingredients. In this paper, we present a vision-based algorithm to localize aligned nutrition fact tables (NFTs) present on many grocery product packages and to segment them into text chunks. The algorithm is a front end to a cloud-based nutrition management system we are currently developing. The algorithm captures frames in video mode from the smartphone's camera, localizes aligned NFTs via vertical and horizontal projections, and segments the NFTs into single- or multi-line text chunks. The algorithm is implemented on Android 2.3.6 and Android 4.2. Pilot NFT localization and text chunking experiments are presented and discussed.

Keywords—***computer vision; image processng; vision-based nutrition information extraction; nutrition management***

I. Introduction

According to the U.S. Department of Agriculture, U.S. residents have increased their caloric intake by 523 calories per day since 1970. A leading cause of mortality in men is prostate cancer. A leading cause of mortality in women is breast cancer. Mismanaged diets are estimated to account for 30-35 percent of cancer cases [1]. Approximately 47,000,000 U.S. residents have metabolic syndrome and diabetes. Diabetes in children appears to be closely related to increasing obesity levels. Many nutritionists and dieticians consider proactive nutrition management to be a key factor in reducing and controlling cancer, diabetes, and other illnesses related to or caused by mismanaged or inadequate diets.

Surveys conducted by the American Dietetic Association (http://www.eatright.org/) demonstrate that the role of television and printed media as sources of nutrition information has been steadily falling. In 2002, the credibility of television and magazines as sources of nutrition information were estimated at 14% and 25%, respectively. In contrast, the popularity of the Internet increased from 13% to 25% with a perceived credibility of 22% in the same time period. Since smartphones and other mobile devices have, for all practical purposes, become the most popular gateway to access the Internet on the go, they have the potential to become proactive diet management tools and improve public health.

Numerous web sites have been developed to track caloric intake (e.g., http://nutritiondata.self.com), to determine caloric contents and quantities in consumed food (e.g., http://www.calorieking.com), and to track food intake and exercise (e.g., http://www.fitday.com). Unfortunately, many such sites either lack mobile access or, if they provide it, require manual input of nutrition data. Manual input challenges on smartphones are well documented in the literatures (e.g., [2], [3]).

One smartphone sensor that can alleviate the problem of manual input is the camera. Currently, the smartphone cameras are used in many mobile applications to process barcodes. There are free public online barcode databases (e.g., http://www.upcdatabase.com/) that provide some product descriptions and issuing countries' names. Unfortunately, since production information is provided by volunteers who are assumed to periodically upload product details and to associate them with product IDs, almost no nutritional information is available and some of it may not be reliable. Some applications (e.g., http://redlaser.com) provide some nutritional information for a few popular products.

While there are many vision-based applications to process barcodes, there continues to be a relative dearth of vision-based applications for extracting other types of useful nutrition information from product packages such as nutrition facts, caloric contents, and ingredients. If successfully extracted, such information can be converted it into text or SQL via scalable optical character recognition (OCR) methods and submitted as queries to cloud-based sites and services.

Another problem and challenge for mobile computing is eyes-free access to nutrition information for visually impaired (VI), blind, and low vision smartphone users. One tool that is frequently mentioned in the literature for eyes-free access to print matter is the K-NFB reader (www.knfbreader.com). The K-NFB reader is a mobile OCR software tool for Nokia mobile phones. Given lower incomes of many VI and blind individuals, the cost of this technology ($2,500 per phone installation), quite possibly, puts it out of reach for many VI users. K-NFB users are required to learn to effectively align print matter with the camera, which may not be a problem for dedicated users but may dissuade others from adopting this technology. More importantly, K-NFB users are required to use small mobile phone keys for navigation and input. The speaker volume is too low for use in outdoors and noisy places such as shopping malls.

In a series of evaluation experiments conducted by the K-NFB system's developers and published at the company's web site, the system accurately identified simple black on white text but did not perform well on documents with color graphics and images, large signs, mixed and italic fonts. The current version of the system cannot read round containers such as cans or products with colored fonts and images and can read flat top boxes only if the text is plain black on white, which is a serious limitation for grocery products, because most of grocery product packages contain colorful images and variable fonts.

The Utah State University (USU) Computer Science Assistive Technology Laboratory (CSATL) is currently developing a mobile vision-based nutrition management system for smartphone users. The system will enable smartphone users to specify their dietary profiles securely on the web or in the cloud. When they go shopping, they will use their smartphones to extract nutrition information from product packages with their smartphones' cameras. The extracted information includes not only barcodes but also nutrition facts, such as calories, saturated fat, sugar content, cholesterol, sodium, potassium, carbohydrates, protein, and ingredients.

Our ultimate objective is to match the extracted information to the users' dietary profiles and to make dietary recommendations to effect behavior changes. For example, if a user is pre-diabetic, the system will estimate the amount of sugar from the extracted ingredients and will make specific recommendations to the user. The system, if the users so choose, will keep track of their long-term buying patterns and make recommendations on a daily, weekly or monthly basis. Dieticians will also be able to participate in and manage the system's data flow. For example, if a user exceeds his or her total amount of saturated fat permissible for the specified profile, the system will notify the user and, if the user's profile has appropriate permissions, the user's dietician.

In this paper, we present a vision-based algorithm to localize aligned NFTs and to segment them into single- or multi-line text chunks. The algorithm captures frames in video mode from the phone camera, localizes aligned NFTs via vertical and horizontal projections, and segments text chunks from localized NFTs. The latter part is referred to in this paper as text chunking. These segmented text chunks can subsequently be input into OCR engines. However, scalable mobile OCR is beyond the scope of this paper. The algorithm has been implemented and tested on the Android 2.3.6 and Android 4.2 platforms.

The remainder of our paper is organized as follows. Section 2 presents related work. Section 3 discusses the localization of aligned NFTs. Section 4 covers how single- or multi-line text chunks are segmented from localized NFTs. Section 5 discusses NFT localization and text chunking experiments. In Section 6, the experimental findings are discussed and several future work directions are outlined.

II. Related Work

Many current R&D efforts aim to utilize the power of mobile computing to improve proactive nutrition management. In [4], the research is presented that shows how such mobile applications can be designed for supporting lifestyle changes among individuals with type 2 diabetes and how these changes were perceived by a group of 12 patients during a 6-month period. In [5], an application is presented that contains a picture-based diabetes diary that records physical activity and photos taken with the phone camera of eaten foods. The smartphone is connected to a glucometer via Bluetooth to capture blood glucose values. A web-based, password-secured and encrypted short message service (SMS) is provided to users to send messages to their care providers to resolve daily problems and to send educational messages to users.

The presented NFT localization algorithm is based on vertical and horizontal projections used by numerous computer vision researchers for object localization. For example, in [6], projections are used to successfully detect and recognize Arabic characters. The presented text chunking algorithm also builds on and complements multiple projects in mobile computing and mobile computer vision that capitalize on the ever increasing processing capabilities of smartphone cameras. In [7], a system is presented for mobile OCR on mobile phones. In [8], an interactive system is presented for text recognition and translation.

III. NFT Localization

A. Vertical and Horizontal Projections

Images captured from the smartphone's video stream can be divided into foreground and background pixels. In general, foreground pixels are defined as content-bearing units in a domain-dependent manner. For example, content can be defined as black pixels, white pixels, pixels with specific luminosity levels, specific neighborhood connection patters (e.g., 4-connected, 8-connetected), etc. Background pixels are those that are not foreground.

Horizontal projection of an image (HP) is a sequence of foreground pixel counts for each row in an image. Vertical projection of an image (VP) is a sequence of foreground pixel counts for each column in an image. Figure 1 shows horizontal and vertical projections of a black and white image with three characters.

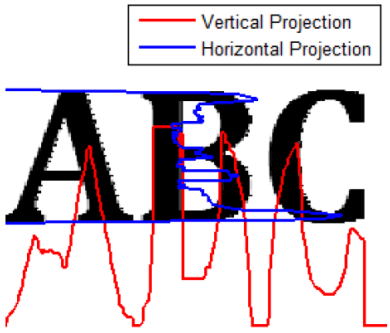

Figure 1. Horizontal & Vertical Projections.

Suppose there is an *m* x *n* image *I* where foreground pixels are black, i.e., $I(x,y)=0$, and the background pixels are white, i.e., $I(x,y)=255$. Then the horizontal projection of row *y* and the vertical projection of column *x* can defined as $f(y)$ and $g(x)$, respectively:

$$f(y)=\sum_{x=0}^{n-1}(255-I(x,y)); \qquad g(x)=\sum_{y=0}^{m-1}(255-I(x,y)). \tag{1}$$

For the discussion that follows it is important to keep in mind that the *x* axis in the image is the column dimension whereas the *y* axis is the row dimension. In other words, the vertical projections computed by $g(x)$ along the *x* axis are used in computing the vertical boundaries of NFTs while the horizontal projections computed by $f(y)$ along the *y* axis are used in computing the NFTs' horizontal boundaries.

B. Horizontal Line Filtering

In detecting NFT boundaries, three assumptions are currently made: 1) a NFT is present in the image; 2) the NFT present in the image is not cropped; and 3) the NFT is horizontally or vertically aligned. Figures 2 shows horizontally and vertically aligned NFTs. The detection of NFT boundaries proceeds in three stages. Firstly, the first approximation of vertical table boundaries is computed. Secondly, the vertical boundaries computed in the first stage are extended to the left and to the right. Thirdly, the upper and lower horizontal boundaries are computed.

The objective of the first stage is to detect the approximate location of the NFT along the horizontal axis (x_s', x_e'). This approximation starts with the detection of horizontal lines in the image, which is accomplished with a horizontal line detection kernel (HLDK) that we developed in our previous research and described in our previous publications [9]. It should be noted that other line detection techniques (e.g., Hough transform [10]) can be used for this purpose. Our HLDK is designed to detect large horizontal lines in images to maximize computational efficiency. On rotated images, the kernel is used to detect vertical lines. The left image of Figure 3 gives the output of running the HLDK filter on the left image shown in Figure 2.

Figure 2. Vertically & Horizonally Aligned Tables.

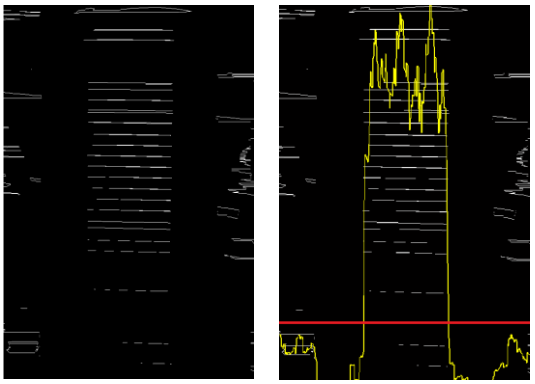

Figure 3. HLFI of Fig. 2 (left); its VP (right).

C. Detection of Vertical Boundaries

Let HLFI be a horizontally line filtered image, i.e., the image put through the HLDK filter or some other line detection filter. Let $VP(HLFI)$ be its vertical projection, i.e., the projections of white pixels computed by for each column of HLFI. The right image in Figure 2 shows the vertical projection of the HLFI on the left. Let θ_{VP} be a threshold, which in our application is set to the mean count of the white foreground pixels in columns. In Figure 3 (right), θ_{VP} is shown by a gray horizontal line in the lower part of the image. It can be observed that the foreground pixel counts in the columns of the image region with the NFT are greater than the threshold. Once the appropriate value of the threshold is selected, the vertical boundaries of an NFT are computed as follows:

$$x_l' = \min\{c \mid g_x(c) \geq \theta_{VP}\}; \qquad x_r' = \max\{c \mid g_x(c) \geq \theta_{VP} \;\&\; x_l' \leq x_r'\}. \tag{2}$$

The pairs of the left and right boundaries that are two close to each other, where 'too close' is defined as the percentage of the image width covered by the distance between the right and left boundaries. It has been experimentally found that the first

approximation along the vertical boundaries are often conservative (i.e., text is cropped on both sides) and must be extended left, in the case of x_l', and right, in the case of x_r'.

To put it differently, the left boundary is extended to the first column to the left of the current left boundary, for which the projection is at or above the threshold, whereas the right boundary is extended to the first column to the right of the current right boundary, for which the vertical projection is at or above the threshold. Figure 4 (left) shows the initial vertical boundaries (VBs) extended left and right.

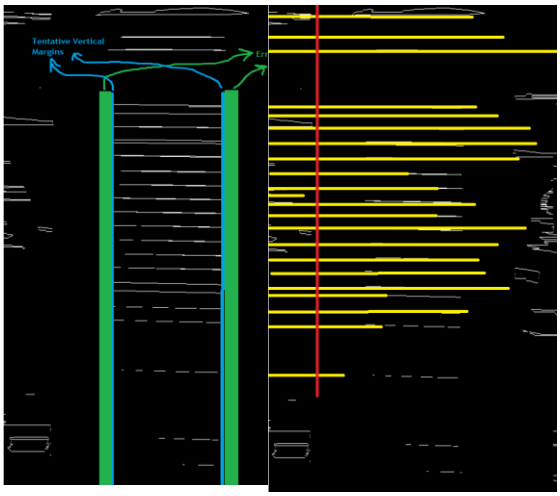

Figure 4. VB Extension (left); HP of Left HFLI in Fig. 2 (right).

D. Detection of Horizontal Boundaries

The computation of the horizontal boundaries of the NFT is confined to the image region vertically bounded by the extended vertical boundaries (x_l, x_r). Let $HP(HLFI)$ be the horizontal projection of the HLFI in Figure 2 (left) and let θ_{HP} be a threshold, which in our application is set to the mean count of the foreground pixels in rows, i.e., $\theta_{HP} = mean\{f(y) | f(y) > 0\}$. Figure 4 (right) shows the horizontal projection of the left HLFI in Figure 2. The gray vertical line in Figure 4 (right) shows θ_{HP}.

The horizontal boundaries of the NFT are computed in a manner similar to the computation of its vertical boundaries with one exception – they are not extended after the first approximation is computed. There is no need to extend the horizontal boundaries up and down, because the horizontal boundaries do not have as much impact on subsequent OCR of segmented text chunks as vertical boundaries. The horizontal boundaries are computed as follows:

$$r_u = \min\{r \mid f(r) \geq \theta_{HP}\}; \qquad (3)$$
$$r_l = \max\{r \mid f(r) \geq \theta_{HP} \;\&\; r \geq r_u\}.$$

Figure 5 (left) shows the nutrition table localized via vertical and horizontal projections and segmented from the left image in Figure 2.

IV. Text Chunking

A typical NFT includes text chunks with various caloric and ingredient information, e.g., "Total Fat 2g 3%." To optimize the performance of subsequent OCR, which is beyond the scope of this paper, these text chunks are segmented from localized NFTs. This approach is flexible in that segmented text chunks can be wirelessly transmitted to cloud servers for OCR. As can be seen in Figure 5 (left), text chunks are separated by black colored separators. Formally, text chunks are defined as text segments separated by horizontal black separator lines.

Figure 5. Localized NFT (left); Text Chunks (right).

Text chunking starts with the detection of these separator lines. Let *N* be a binarized image with a segmented NFT and let $p(i)$ denote the probability of image row *i* containing a black separator. If such probabilities are reliably computed, text chunks can be localized. Toward that end, let l_j be the length of the *j*-th consecutive run of black pixels in row *i* above a length threshold τ_l. If *m* be the total number of such runs, then $p(i)$ is computed as the geometric mean of $(l_0, l_1, ..., l_m)$. The geometric mean is more indicative of the central tendency of a set of numbers than the arithmetic mean. If θ is the mean value of all positive values of normalized by the maximum value of $p(i)$ for the entire image, the start and end coordinates, y_s and y_e, respectively, of every separator along the *y* axis can be computed by detecting consecutive rows for which the normalized values are above the threshold as follows:

$$y_s = i \mid p(i-1) \leq \theta \;\&\; p(i) > \theta; \qquad (4)$$
$$y_e = j \mid p(j+1) \leq \theta \;\&\; p(j) > \theta.$$

Once these coordinates are identified, the text chunks can be segmented from either the binarized or grayscale image, depending on the requirements of the subsequent of OCR. As

can be seen from Figure 5 (right), some text chunks contain single text lines while others have multiple text lines.

V. Experiments

The NFT localization algorithm was implemented on Android 2.3.6 and Android 4.2. Forty five images were captured on a Google Nexus One (Android 2.3.6) in a local supermarket. The average running time of the algorithm is approximately one second per frame. All images were checked by a human judge to ensure that an NFT is present in the image, is not rotated, and is not cropped along any of its four sides. These images were then placed on a Google Nexus One smartphone (Android 2.3.6) and on a Galaxy Nexus (Android 4.2) smartphone with the installed application to obtain images with segmented NFTs and save them on the smartphones' SDK cards. The processed images were then analyzed by a human judge. On each original image, the four corners of an NFT were manually marked to obtain the ground truth to evaluate the segmentation process.

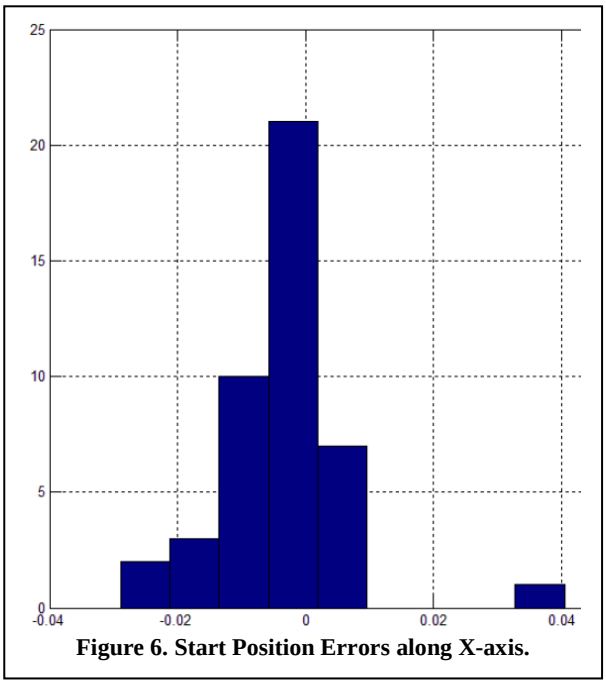

Figure 6. Start Position Errors along X-axis.

Figures 6 and 7 show the error histograms for the starting and ending positions of the segmented NFTs along the images' *x*-axis, respectively. In both figures, the *x*-axis encodes the error as a fraction of the NFT width while the *y*-axis encodes the number of images with a specific error value. Figures 8 and 9 show the error histograms for the starting and ending positions of the segmented NFTs along the images' *y*-axis, respectively. In Figure 8 and 9, the *x*-axis encodes the error as a fraction of the NFT height. Positive errors occur in segmented images where segmented NFTs contain extra background pixels whereas negative errors occur when NFTs are cropped.

In general, positive errors are better for our purposes than negative ones because negative errors signal information loss that may result in subsequent OCR or image classification errors. It should be observed that the performance of the NFT localization algorithm has a mean error of 1% on the sample of images. There was one notable outlier, for which the start position on the *x*-axis error was 12.5% and the end position error on the *x*-axis was 14%. The same image was the outlier for the segmentation errors of the start and end positions along the *y*-axis.

Figure 10 shows the outlier image that caused the segmentation errors along both axes. It can be observed that the NFT in this image lacks a black colored bounding box that is usually present around nutrition fact tables. It is the absence of this box that caused the algorithm to fail to obtain the exact location of the NFT in the image.

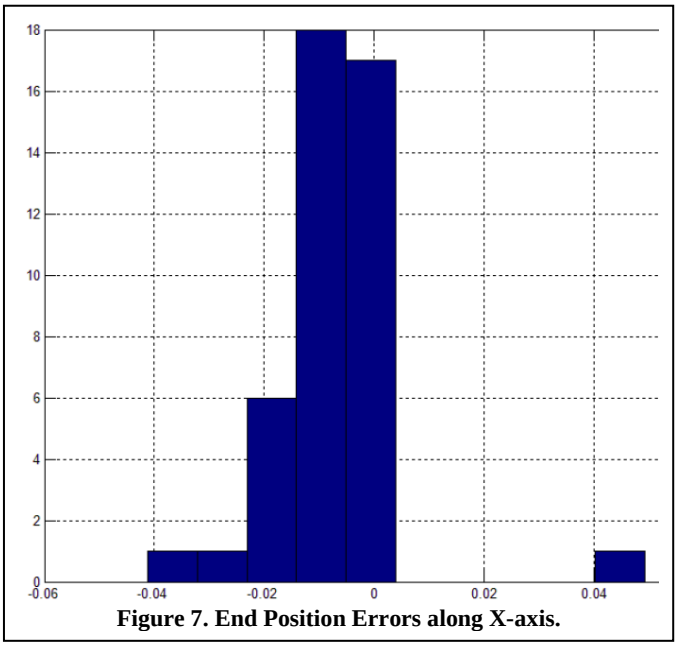

Figure 7. End Position Errors along X-axis.

The performance of the NFT localization algorithm along the *y*-axis has the mean errors of 5% and 7% for the start and end positions, respectively. Most errors, along both axes, are caused by NFTs that are not bounded by boxes, one of which is shown in Figure 10.

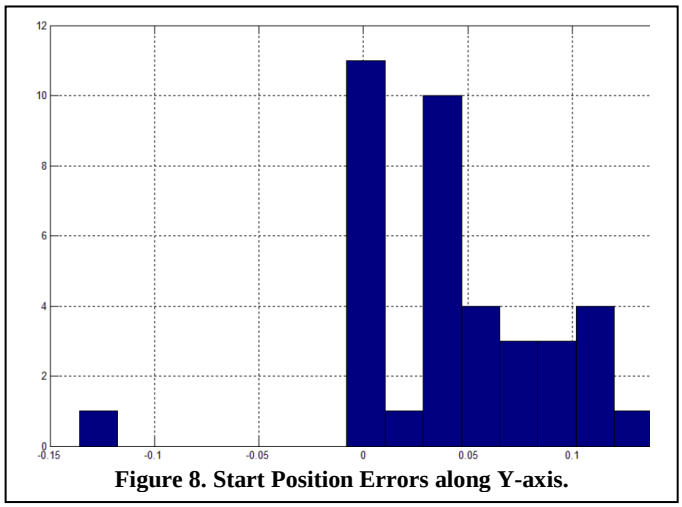

Figure 8. Start Position Errors along Y-axis.

The preliminary evaluation of the text segmentation algorithm was done on a set of 15 NFT images. A total of 303 text chunks (text segments between separator bars) were manually identified. Of these manually detected chunks, the algorithm detected 163, which gives a detection rate of 53.8%. The average running time of the text chunking algorithm is approximately half a second per localized NFT.

A statistical analysis of text chunk segmentation was executed. All text chunks readable by a human judge were considered as true positives. There were no true negatives insomuch as all text chunks had text. Text chunks which could not be read by a human judge were reckoned as false positives. False positives also included detection of separator bars between text chunks. Figure 11 contains a true positive example and two examples of false positives.

There were no false negatives in our sample of images, because all text chunks either contained some text or did not contain any text. The performance of the text chunking algorithm was evaluated via precision, recall, specificity and accuracy were calculated. The average values for these measures over the entire sample are given in Table 1.

VI. Discussion

The NFT localization algorithm had a mean error of 1% on the sample of NFT images. The average accuracy of the text chunking algorithm on the sample of images with localized NFTs is 85%. While we readily acknowledge that these results must be interpreted with caution due to small sample sizes, we believe that the approaches presented in the paper show promise as a front end vision-based nutrition information extraction module of a larger nutrition management system.

One limitation of the presented approach to NFT localization is that an image is assumed to contain a horizontally or vertically aligned NFT. We are currently working on relaxing this constraint to localize skewed NFTs in captured frames.

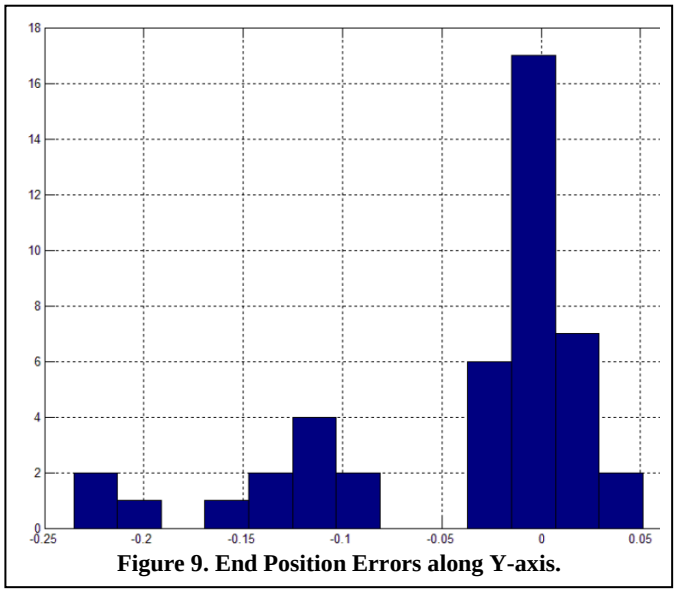

Figure 9. End Position Errors along Y-axis.

Figure 10. NFT Localization Outlier.

Table I. Average Recall, Precision, Specificity, Accuracy.

Precision	Recall	Specificity	Accuracy
0.70158	0.9308	0.88978	0.8502

The detection of skewed NFTs will make the system more accessible to users that require eyes-free access to visual information. However, it should be noted in passing that,

while visually impaired, low vision, and blind populations continue to be a major target audience of our R&D efforts, nutrition management is of vital importance to millions of sighted individuals who will not have a problem aligning their smartphone cameras with NFTs on product packages.

Another important improvement is to couple the output of the text chunking algorithm to an OCR engine (e.g., Tesseract (http://code.google.com/p/tesseract-ocr) or OCRopus (https://code.google.com/p/ocropus)). We have integrated the Android Tesseract library into our application and run several tests but were unable to analyze the collected data before the submission deadline. We plan to publish our findings in a future publication.

Finally, we would like to bring the combined frame processing time to under one second per frame. This will likely be accomplished by moving current code bottlenecks to the Android NDK or using OpenCV Android libraries.

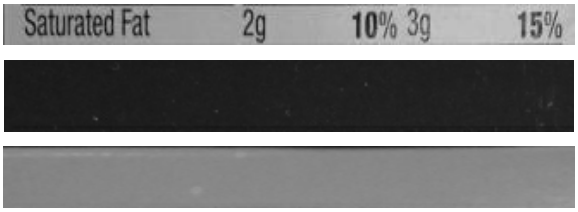

Figure 11. True Positive (top); Two False Positives (middle, bottom).

Acknowledgment

This project has been supported, in part, by the MDSC Corporation. We would like to thank Dr. Stephen Clyde, MDSC President, for supporting our research and championing our cause.

References

[1] Anding, R. *Nutrition Made Clear*. The Great Courses, Chantilly, VA, 2009.

[2] Kane S, Bigham J, Wobbrock J. (2008). Slide Rule: Making M obile Touch Screens Accessible to Blind People using Multi-Touch Interaction Techniques. In *Proceedings of 10-th Conference on Computers and Accessibility (ASSETS 2008)*, October, Halifax, Nova Scotia, Canada 2008; pp. 73-80.

[3] Kulyukin, V., Crandall, W., and Coster, D. (2011). Efficiency or Quality of Experience: A Laboratory Study of Three Eyes-Free Touchscreen Menu Browsing User Interfaces for Mobile Phones. *The Open Rehabilitation Journal*, Vol. 4, pp. 13-22, 2011, DOI: 10.2174/1874943701104010013.

[4] Årsand, E., Tatara, N., Østengen, G., and Hartvigsen, G. 2010. Mobile Phone-Based Self-Management Tools for Type 2 Diabetes: The Few Touch Application. *Journal of Diabetes Science and Technology*, 4, 2 (March 2010), pp. 328-336.

[5] Frøisland, D.H., Arsand E., and Skårderud F. 2010. Improving Diabetes Care for Young People with Type 1 Diabetes through Visual Learning on Mobile Phones: Mixed-methods Study. *J. Med. Internet Res.* 6, 14(4), (Aug 2012), published online; DOI= 10.2196/jmir.2155.

[6] Al-Yousefi, H., and Udpa, S. 1992. Recognition of Arabic Characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14, 8 (August 1992), pp. 853-857.

[7] Bae, K. S., Kim, K. K., Chung, Y. G., and Yu, W. P. 2005. Character Recognition System for Cellular Phone with Camera. In *Proceedings of the 29th Annual International Computer Software and Applications Conference*, Volume 01, (Washington, DC, USA, 2005), COMPSAC '05, IEEE Computer Society, pp. 539-544.

[8] Hsueh, M. 2011. *Interactive Text Recognition and Translation on a Mobile Device*. Master's thesis, EECS Department, University of California, Berkeley, May 2011.

[9] Kulyukin, V., Kutiyanawala, A., and Zaman, T. 2012. Eyes-Free Barcode Detection on Smartphones with Niblack's Binarization and Support Vector Machines. In *Proceedings of the 16-th International Conference on Image Processing, Computer Vision, and Pattern Recognition*, Vol. 1, (Las Vegas, Nevada, USA), IPCV 2012, CSREA Press, July 16-19, 2012, pp. 284-290; ISBN: 1-60132-223-2, 1-60132-224-0.

[10] Duda, R. O. and P. E. Hart. 1972. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Comm. ACM*, Vol. 15, (January 1972), pp. 11-15.