IMRESTAURANT()

MATLAB for Feature-based Restaurant Logo Recognition

Roshni Cooper
Department of Electrical Engineering
Stanford University
Palo Alto, CA, USA
rccooper@stanford.edu

Tony Hwang
Department of Electrical Engineering
Stanford University
Palo Alto, CA, USA
hwangt@stanford.edu

***Abstract*—This paper discusses the implementation of imrestaurant(), a MATLAB function for feature-based restaurant logo recognition. The algorithm works by first applying SIFT to a logo database and then creating a hierarchical vocabulary tree using K-means clustering. The input to the algorithm is a photograph of an image logo, which is first filtered to reduce noise. Then SIFT is applied to the input image, and the output descriptors are pushed to the vocabulary tree. Each image in the logo database is scored using TF-IDF weights and the ten highest scoring database logos are pairwise matched using the ratio test and RANSAC. The restaurant database image with the largest consensus set is searched for on yelp.com. This algorithm was shown to be robust against input image size, background noise, and shearing. Finally, future work in increasing the accuracy of feature detection as well as optical character recognition is discussed.**

I. Introduction

Searching for restaurant reviews online is one of the many conveniences afforded by today's ubiquity of the internet. This project aims to simplify the process of searching for a new restaurant by taking a snapshot of the restaurant's sign and invoking a web browser window displaying a search for the restaurant on yelp.com using a function called imrestaurant(). The most ideal system would be able to look at the photograph of any restaurant logo and segregate and recognize the letters in order to determine the restaurant's name. Initially, this project aimed to perform this optical character recognition (OCR). As further research and preliminary testing was performed, it was discovered that although OCR techniques have been successfully developed, segregating text from background is still a largely unsolved problem.

To align this project with the scope of the class, the project evolved from OCR to performing feature-based image matching between photographs of restaurant logos and a database of clean logos found online. The algorithm uses the scale-invariant feature transform (SIFT) feature detection and various feature-based image matching techniques including hierarchical k-means clustering, the ratio test, and RANSAC. .This paper outlines the preliminary experimentation with OCR, the successful implementation of feature-based logo recognition and its results, and finally touches on areas for future work.

II. Optical Character Regonition

The original goal of this project was to take a photo of a restaurant logo, segregate the text of the restaurant name from the background, and then perform optical character recognition to identify each character in the restaurant's name. The first step, text segregation, ensures that letters are distinct and aligned horizontally and vertically in order to standardize them as much as possible for recognition. Next, the algorithm characterizes distinctive properties of each character in the image and uses these properties to determine the best match for the character in a database. The database would have to contain characters from various training sets that have been similarly characterized..

In the early stages of the imrestaurant() project, the team attempted to implement the text separation portion of the algorithm on the DROID phone using Java. Due to the team's limited experience in Java, this portion was instead implemented in MATLAB. This section outlines the algorithm designed before the project changed.

Due to the complexity of OCR, some requirements were placed on the images to be used and the text in them:

- The only text contained in the photograph is the restaurant's name.
- Characters in a word are in a straight line.
- Characters are read from left to right, not top to bottom.
- No cursive or connected characters.
- Individual letters may only consist of contiguous segments.

The text separation algorithm involves many steps. It involves preprocessing to remove noise from the image, thresholding to create a binary image, slant and skew removal to aid in separation of characters and words, projection histograms to determine how many letters and words are in the image, and region labeling and counting to identify each character and each word [15].

First, the image is converted from color to grayscale using MATLAB's built-in rgb2gray function. Processing the image

in grayscale reduces the amount of computation since only one color channel needs to be processed.

Next, nonlinear noise reduction and sharpening are performed using soft coring. A block diagram of this method is shown in Fig. 1.

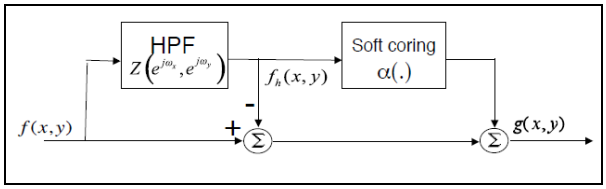

Figure 1. Nonlinear noise reduction and sharpening block diagram [6].

The high frequency and low frequency components of the image are separated, then the high frequency component is soft-cored using the following function α described by equation (1) [6], and whose transfer function is shown in Fig. 2.

$$\alpha\left(f_i(x,y)\right) = m \cdot f_i(x,y) \cdot \left(1 - e^{-\left(\frac{f_i(x,y)}{\tau}\right)^{\gamma}}\right) \qquad (1)$$

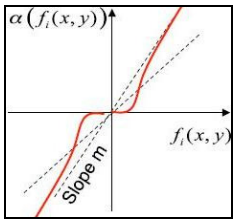

Figure 2. Soft coring transfer function [6].

This method of filtering allows for the reduction of noise while still emphasizing the edges in the image. For OCR, clear edges are necessary for determining character boundaries. After the initial filtering, the gray image is thresholded to produce a binary image. The threshold for the binarization of the image is calculated using Otsu's method which is implemented in the MATLAB function graythresh. A binary image is useful for performing region counting and labeling. But before region counting and labeling are performed, skew removal aligns the text in the image to horizontal lines. Skew removal is performed by taking the Hough transform of the image, finding the angle with the highest count, and then rotating the image by that angle.

Next the regions in the image are counted and labeled. In theory, now each character has a unique label. Even if the image text does not contain any connected characters as required, the region counting and labeling will fail if the text of the logo is the same color as the background of the image, for example in the Domino's logo shown in Fig. 3. In this case, all of the letters have the same label, and it is the same label as the background. This issue posed a significant problem that was not solvable given our limited background in image processing and text recognition. Another issue that we encountered during region counting and labeling is the labeling of elements that are not characters in the name; the circles in the Domino's logo would be identified as the letter "o" or the number zero, which would confuse the system.

Figure 3. Logo with same colors of text as background.

Assuming that the letters can be separated, and that only letters are remaining after the separation, vertical projection histograms are then computed for each character and are used to remove shearing. Realigning the characters to the vertical facilitates OCR by ensuring that the letter characterization is as similar to the database characteristics as possible. Various shear matrices are applied to each letter, and the shear transformation which generates the greatest peaks in the vertical project histogram is the one used to remove the slant of the letters [3][4]. This slant (shear) removal is the final stage of the text separation portion of the imrestaurant() algorithm.

Next, we planned to implement OCR by using projection histograms, profiling, zoning, and number of crossings to match database characters. [15].

The projection histograms provide useful information about the spatial distribution of pixels in a character. For profiling, zoning, and crossing counting, a rectangle of background bounding the character needs to be considered in addition to the character itself. Profiling counts the number of pixels from the edge of the box to the edge of the character. Profiling helps determine the differences between many letters, such as "p" and "q," and also can be used to determine the external contours of a character [15].Zoning divides the character's bounding box into smaller rectangular zones, and then counts the number of foreground pixels in each zone to create a different type of spatial histogram. The distribution of the zoning histogram helps to distinguish local features of the letter as opposed to global features [15].

Finally, a crossing is a transition from foreground to background pixels along horizontal and vertical lines throughout the character's bounding rectangle [15]. These numbers again provide useful information when searching for the best matching character in the database.

The database of characters would need to be very large because of the vast array of fonts used in restaurant logos. Additionally, the database would need to have both capital and lowercase letters, as well as punctuation. Each of the metrics described above would have to be calculated for each character in the database, and an efficient searching algorithm would be necessary to quickly find the best-matching letter in the database.

Optical character recognition is a very interesting problem, but because of the challenges posed in segregating characters from background the goal of imrestaurant() became detecting logos as a whole using feature-detection as suggested by the course teaching assistant. This new method would allow for

the algorithm to use more information than simply the text in the image, for example certain symbols unique to a restaurant's logo.

III. Previous and Related Work

A. *Scale-invariant Feature Transform (SIFT)*

Modern feature detection employs a few general stages of techniques in series to observe and describe the features. Edges, corners, and blobs are a sampling of feature types. To detect any of these features, the image is convolved with a transfer function which emphasizes the intensity gradients in specific directions. In SIFT, scale-invariant feature detection is implemented by convolving images with scaled Gaussian functions to generate a scale space. Adjacent images in scale space are then subtracted which results in Difference of Gaussian (DOG) images. The DOG approximates the scale-normalized Laplacian of Gaussian, but is more efficiently computed [7]. The features are located by comparing each sample point of the DOG to its neighbors in space and scale. It is kept as a feature, also known as a keypoint, only if it is a local maxima.

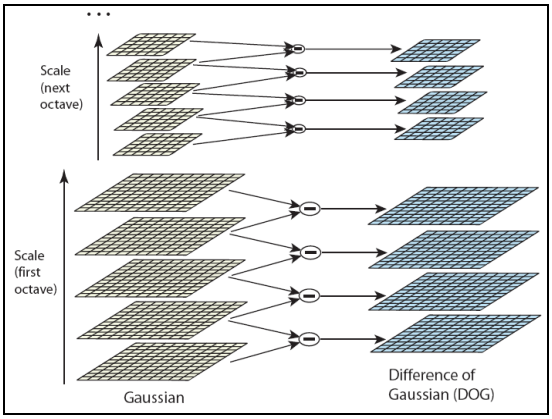

Figure 4. SIFT scale spaces are subtracted to calculate the DOG [7].

Once the keypoints have been determined, they are then further described by computing local descriptors. Each keypoint's descriptor comprises local gradient information stored in the form of a 128-dimensional vector. These gradients are consolidated as a 4x4 array of histograms with 8 orientation bins each, thus resulting in 128 values [7].

B. *K-means Algorithm*

One method of organizing feature descriptors for easier indexing is through hierarchical k-means clustering. The k-means algorithm has two steps, assignment and update. During the assignment step, k random descriptor values are chosen within the descriptor space. The data is then grouped with the centroid to which it has the lowest Euclidean distance. During the update step, the centroid of each group is then redefined as the mean value of the descriptors. These steps repeat until the centroids no longer change location [8].

Hierarchical k-means extends this concept such that k-means is applied to the clusters, creating a hierarchy of categorization. This hierarchy can then be organized into a tree, where each node branches out to k children nodes until the tree is complete [10]. In Fig. 5(a), the hierarchical k-means clustering of a 2-dimensional space is shown. See Fig. 5(b) for and illustrated representation of the tree.

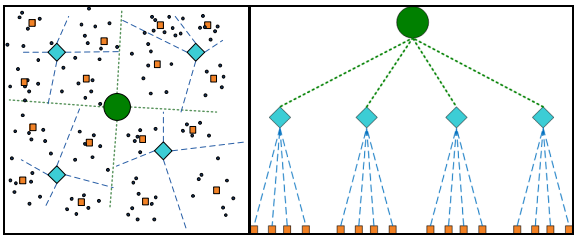

Figure 5. (a) Hierarchical K-means clustering of a 2D space with k=4. (b) Tree generated from hierarchical k-means clustering with k=4.

When a set of feature descriptors from a test image is pushed down the hierarchical tree, each resulting path corresponds, with some probability, to the test image matching a database image. Thus, scores for the likelihood that a test image matches a particular database image are incremented for each feature descriptor. Once all descriptors have been pushed, the highest of the resulting tallies will reveal the most likely image-to-database matches. There are several methodologies for determining the weights used in scoring, including hard binning and soft binning. Hard binning will only increment scores associated with the final node reached in a path. Soft binning is a more complex scheme that assigns fractional counts simultaneously to multiple possible nodes [1].

C. *Pairwise Feature Matching*

Pairwise matches between a test image and database image can be chosen and trimmed through ratio testing and the Random Sample Consensus (RANSAC) Algorithm [14].

The ratio test forms a set of correspondences between descriptors of the query image and the database image. For each query descriptor, the Euclidean distance to all database descriptors is calculated, and the ratio of the two minimum distances is compared to a threshold. If the comparison passes, then the correspondence between the test keypoint to the database keypoint with the minimum Euclidean distance is stored. Equation (2) shows the minimum distance ratio equation.

$$\frac{\min_{i} \sqrt{(q_i - d_i)^2}}{\min_{j, j \neq i} \sqrt{(q_j - d_j)^2}} < threshold \qquad (2)$$

Once a set of correspondences is generated from the ratio test, RANSAC is applied to trim potential outliers from the set. The algorithm selects several correspondences at random, and generates an affine model to fit these correspondences. Then, the remaining correspondences are fitted against the model, and those that have a small enough error are kept in a consensus set. RANSAC stores the largest consensus set from all iterations, and if the set is large enough, it is returned after the

last iteration to be used as a feature correspondence map between the two images [5].

IV. IMRESTAURANT() ALGORITHM

A system-level block diagram of the imrestaurant() algorithm is shown in Fig. 6. First, the algorithm trains the program using a database of logos. The first step of the training process is to convert each logo in the database from a color image to a gray image using MATLAB's rgb2gray function. Then, SIFT is performed on each database logo and a hierarchical k-means tree is built from the keypoint descriptors using the vlfeat and sift open source libraries [16][17]. Next, the program asks the user for a zip code and the filename of the photograph containing the restaurant logo. Once the program reads the photograph file it converts the color image to gray using rgb2gray. The gray query image also undergoes SIFT, and its descriptors are pushed down the tree, resulting in score tallies for matches with database images. An example image is shown in Fig. 7 with the frames and descriptors superimposed. The top ten scorers are then pairwise matched with the query image using the ratio test and RANSAC, and the one with the largest consensus set is deemed the best match. Its filename is then parsed and opened directly on yelp.com in a web browser.

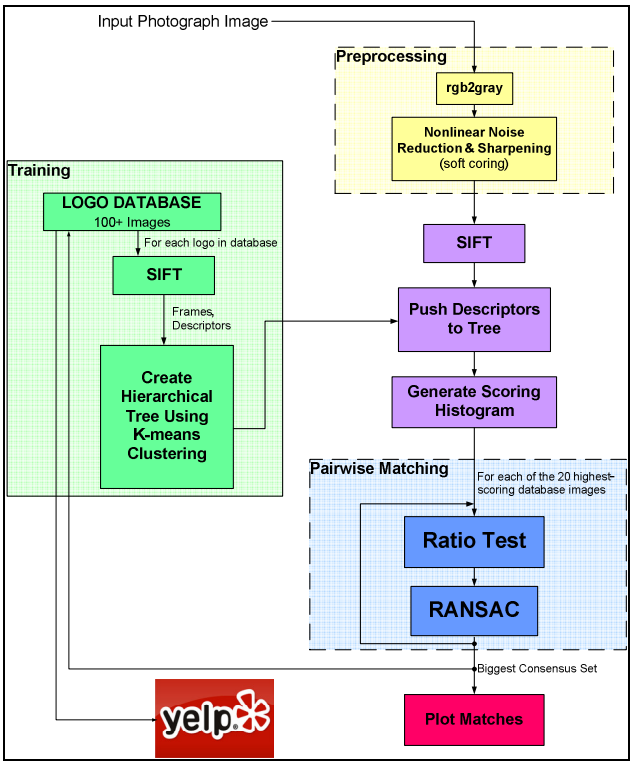

Figure 6. System-level block diagram of imrestaurant() algorithm.

The hierarchical K-means tree is structured to fit our data set. There are approximately 100 images in our database. Accordingly, we choose k=10, and create enough levels of depth such that there are approximately as many leaves at the lowest level as there are descriptors. This tree structure will attempt to maximize precision of our scoring methodology described below.

Figure 7. Sample photo with SIFT frames and descriptors. Green dots are the keypoints and the blue squares show the scaled descriptors.

Once the tree is generated from the database images, each feature descriptor from the test image is pushed through the tree. It ends up at a leaf which is associated with a set of TF-IDF weighted scores for each database image. Term frequency-inverse document frequency (TF-IDF) weights are a form of hard-binning that assigns greater weight to nodes that are unique, and thus is useful when trying to weigh the value of descriptor matches [11][2]. The equation for calculating the weights is showing in equation (3), where N is the total number of leaves, and N_i is the number of descriptors mapped to the particular leaf. The total score for each database image is then increased accordingly for every feature descriptor pushed through the tree.

$$W_i = \log\left(\frac{N}{N_i}\right) \qquad (3)$$

The top ten scorers are pairwise matched against the query image to save the computation of pairwise matching against all 100 images. The ratio test is defined with a threshold of 0.8. In our implementation of RANSAC, we initially choose 3 correspondences randomly to generate the affine model. We also add in protection against choosing points too close to each other by ensuring a minimum Euclidean distance among them of 20. If these points are too close together, the iteration is passed and a new set of random correspondences is used. Once acceptable points are chosen and the affine model is found, the expected descriptors are compared against the actual descriptors, and if they are similar enough (closer than 20 in Euclidean distance), they are added to a consensus set. The consensus set is then used to generate a better affine model, and the expected and actual correspondences are compared to calculate the error of the model. The RANSAC procedure iterates 400 times, and the best consensus set is kept. The block diagram for the RANSAC algorithm is shown in Fig. 8.

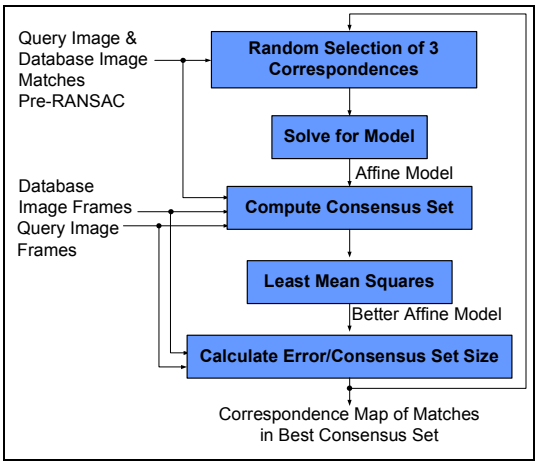

Figure 8. RANSAC algorithm block diagram.

After pairwise matching is completed for the ten best-matched images, the algorithm deems the database image with the biggest consensus set to be the best match for the query image. The filename of the database image is then parsed out and is used with the zip code provided at the beginning of the algorithm to send a URL to the default web browser.

V. Experimental Results

The algorithm used in this project exposed the strengths and weaknesses of the techniques employed. For example, the algorithm was very capable of dealing with variation of scale due to the scale-space feature detection of SIFT. We were able to match query photos of 640x480, 1600x1200, and 3648x2736 to logos of various sizes from the Internet. See Fig. 9 for an example.

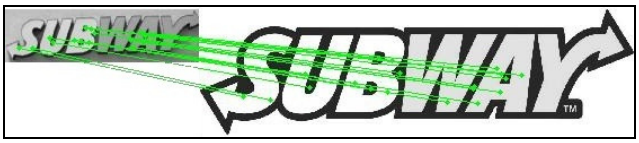

Figure 9. Descriptor matches mapping illustrating scale-invariance of imrestaurant() algorithm.

The algorithm also demonstrated some robustness against shearing up to about 40° from the perpendicular, as shown in Fig. 10. The SIFT detector emphasizes blob detection by using the Gaussian filter, thus it is still able to find the regions shown from an angle as long as they do not change too much in shape.

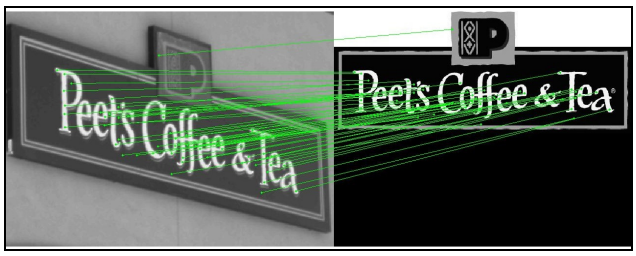

Figure 10. Descriptor matches mapping illustrating robustness against shearing of imrestaurant() algorithm.

There was some robustness against noisy backgrounds, as Fig. 11 illustrates. Feature matching worked well when there was only a constant solid background, and we also saw matches with noisy backgrounds when the restaurant in the logo was clear and not affected by lighting.

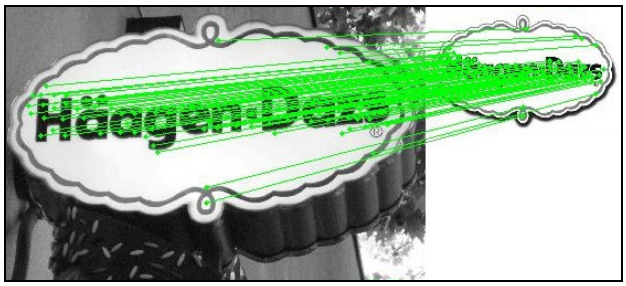

Figure 11. Descriptor matches mapping illustrating robustness of imrestaurant() algorithm against noisy backgrounds.

Incorrect image matches were caused mostly by severe shearing, noisy backgrounds, rotation, and blurry images. Although SIFT feature detection itself is rotation invariant, the ratio test and RANSAC are not. Blurry images were somewhat corrected by our non-linear noise reduction and sharpening, but it still could not accommodate for all blurry images.

While optimizing our algorithm, we experimented with the parameters that defined the feature detection and matching. We increased the EdgeThresh variable to 1000 in the vl_sift() function because we suspected that we were not starting with enough features for our tree. Since computation time was not an issue, we felt that this was a sufficient method to increase the usefulness of our hierarchical tree. We also experimented with values for the ratio test threshold (0.7-0.9), and settled upon 0.8 as the best threshold value to use across our set of test images. In RANSAC, at first we chose the best consensus set based on finding a model with the lowest error. However, as we saw that RANSAC was eliminating too many correspondences, we changed the algorithm to pick the biggest consensus set regardless of error, banking on the fact that the ratio test would sufficiently eliminate most erroneous matches.

VI. Conclusion

This project successfully took a photograph of a restaurant logo and matched the logo to a clean version of the same logo in a database of images using scale-invariant feature detection (SIFT), and feature-based image matching algorithms (hierarchical K-means clustering, the ratio test, and RANSAC).

The imrestaurant() function had greater success with some images than others. Regarding color, photographs with similar contrast and intensity as their database counterparts performed well. Photographs of logos on glossy surfaces were not matched successfully because of the reflections, as expected [12]. The ratio test is not invariant to shearing, so images taken at a large angle did not always match well with the database logos. The RANSAC algorithm generated an affine model for the relationship between the photograph and the logo, so if the photograph was of a logo projected with a different correspondence, RANSAC was not as successful.

To improve the imrestaurant() algorithm, the k-means clustering and ratio test could be performed with different metrics than the Euclidean distance. For example, pixel coordinates, color, intensity, and image texture could be used instead [13].

In the future, in order for users to run imrestaurant() to instantly access restaurant reviews, the algorithm can be ported from MATLAB to a cellular phone.

Other future work involves learning more about text separation and OCR in order to develop a more generic algorithm for determining the restaurant name from a logo or a sign instead of depending on the entire logo. Selecting the characters accurately from a photograph is very challenging because of other components in an image, and it would be an interesting problem to solve in the future.

ACKNOWLEDGMENT

We would like to thank the teaching assistant David Chen for his incredible patience and desire to share his knowledge about image processing in general and specifically feature detection during the course of this project. David was a very helpful and responsive resource. Additionally, we would like to acknowledge Vijay Harid for his willingness to share ideas about the RANSAC algorithm.

REFERENCES

[1] D. M. Chen, S. S. Tsai, V. Chandrasekhar, G. Takacs, R. Vedantham, R. Grzeszczuk, and B. Girod, "Inverted Index Compression for Scalable Image Matching," *Proceedings of Data Compression Conference (DCC)*, Snowbird, Utah, March 2010.

[2] O. Chum, J. Philbin, and A. Zisserman, "Near-duplicate image detection: min-hash and tf-idf weighting," In *Proceedings of British Machine Vision Conference*, 2008.

[3] F. de Zeeuw, "Slant Correction Using Histograms," Undergraduate Thesis.http://www.ai.rug.nl/~axel/teaching/bachelorprojects/zeeuw_slantcorrection.pdf

[4] N.Fakotakis, E.Kavallieratou , and G.Kokkinakis, "New Algorithms for Skewing Correction and Slant Removal on Word-level," proc IEEE, 1999.

[5] M. Fischler and R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. Of the ACM*, vol. 24, pp. 381–395, June 1981.

[6] B. Girod, "Image Filtering and Deconvolution," EE368 Lecture Notes. http://stanford.edu/class/ee368/Handouts/5-Filtering_April9_12_14.pdf

[7] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, 60, 2 (2004), pp. 91-110.

[8] D. MacKay, "Chapter 20. An Example Inference Task: Clustering," *Information Theory, Inference and Learning Algorithms*. Cambridge University Press. pp. 284–292, 2003.

[9] P. Nagabhushan and N. Shivananda, "Separation of Foreground Text from Complex Background in Color Document Images," IEEE Advances in Pattern Recognition, vol. 7, pp. 306-309, Feb. 2009.

[10] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 2161-2168, June 2006.

[11] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for IDF," *Journal of Documentation*, vol. 60, pp. 503-520, 2004.

[12] C. Roduner and M. Rohs, "Practical Issues in Physical Sign Recognition with Mobile Devices," Pervasive 2006 Workshop Proceedings (Workshop on Pervasive Mobile Interaction Devices, PERMID 2006.

[13] L. Shapiro and G. Stockman, *Computer Vision,* Upper Saddle River, NJ: Prentice Hall, 2001.

[14] G. Takacs, V. Chandrasekhar, H. Chen, D. Chen, S. Tsai, R. Grzeszczuk, B. Girod, "Permutable Descriptors for Orientation-Invariant Image Matching," *Proceedings of SPIE Conference on Optical Engineering and Applications*, San Diego, August 2010.

[15] Vamvakas, "Optical Character Recognition for Handwritten Characters." http://www.iit.demokritos.gr/IIT_SS/Presentations/Off-Line%20Handwritten%20OCR.ppt.

[16] A. Vedaldi and B. Fulkerson, "VLFeat: An Open and Portable Library of Computer Vision Algorithms," 2008, http://www.vlfeat.org.

[17] A. Vedaldi, "SIFT for MATLAB," 2006, http://www.vlfeat.org/~vedaldi/code/sift.html.

APPENDIX

Tony Hwang and Roshni Cooper both researched various algorithms, attempted to implement OCR algorithms on the DROID phone, successfully implemented them in MATLAB, and worked on various sections of the poster and paper. The table shows which functions each student helped to implement.

TABLE I. DISTRIBUTION OF WORK

MATLAB Function Name	**Team Member**	
	Tony Hwang	***Roshni Cooper***
corr2frames		X
eucdist		X
getModel_r	X	
imrestaurant	X	X
preproc	X	
projectionHistogram	X	
RANSAC	X	X
ratio_test	X	X
region_label		X
run_imrestaurant	X	X
sharpen	X	
shear		X
skew_removal		X
training	X	X