Mobile Wine Label Recognition

Timnit Gebru, Oren Hazi, Vickey Yeh
Department of Electrical Engineering, Stanford University, Stanford, CA 94305

***Abstract* — In this project, we designed and implemented a system for wine label recognition on a mobile phone. Our algorithm was implemented in Matlab to facilitate testing and minimize development time. We built a prototype Android application that allows us to demonstrate and test our system on a Motorola Droid while the image processing is performed on a server running our Matlab scripts.**

Our visual search component computes SURF keypoints and descriptors on a test image. We compare these descriptors to a set of precomputed descriptors in a wine label database in order to find a rough correspondence map. We then use RANSAC to find a more accurate correspondence map and use the refined map to identify the wine label in the test image.

Images of a wine bottle taken from different angles will show a label that is distorted by the cylindrical projection from the bottle. RANSAC requires a model for reversing this distortion in order to find the optimal correspondence map. Rather than using a non-linear warping model for this transformation, we divided the test image into three regions where a linear projective map could be used instead. This simple technique allows us to find descriptor correspondences covering a greater fraction of the label than a simple homography or affine map, at the expense of processing time.

Keywords - wine label; wine recognition; label recognition; label features; visual search; cylindrical RANSAC; cylindrical approximation

I. Introduction

The motivation for this project arose when one of our group members was at a grocery store buying a bottle of wine for a dinner party. It quickly became apparant that colorful labels and superficial in-store descriptions did not provide sufficient information to make an informed wine selection.

Our group decided to explore image matching techniques that could be used to identify specific wines from an image of the label taken on a mobile device. Our long-term goal is to build an application for the Android platform that uses these techniques to provide a positive experience for a user by providing them with reviews, recommendations, prices at nearby stores, etc.

In [1], Chen et al. present an Android-based mobile asset tracking application for books. The application performs a visual search on a database of book spines. The matching is fast enough to be performed in real time as the user scans the camera across a bookshelf. Their interface even allows for information on the matched book spine to be overlaid on the viewfinder image in an augmented-reality mode so that the user does not have to deal with the shutter or waiting for a server response. This is the behavior that we would eventually like our application to emulate. A user should be able to scan their mobile phone along a shelf at a grocery store while the application identifies labels and displays ratings and prices at nearby stores. The user should be able to stop and select a result to see an in-depth review or recommendations for similar wines. Admittedly, we still have significant barriers to overcome before we can come close to doing this, the most critical being the speed of the visual search.

II. Prior Work

Although the field of mobile visual search has grown considerably in the past ten years due to the advent of mobile phones equipped with high resolution cameras and hardware accelerated graphics, there are many applications that have not yet been explored. Furthermore, the applications that utilize reverse image search algorithms are often not optimized for their specific configuration resulting in unnecessarily complicated implementations.

One such application is Snooth Wine Pro, a $4.99 iPhone application which incorporates image-based wine searching allowing users to scan wine labels and learn more about or purchase a particular wine of interest. Although the relatively constant geometry of wine bottles allows for the use of a simplified visual search algorithm, Snooth Wine Pro utilizes TinEye reverse image search engine built by Idee labs to perform image matching[2]. Since TinEye's feature matching method is not optimized for wine bottles, its use in the implementation of Snooth Wine Pro may result in an application that is slower and more costly than necessary.

By using a feature matching algorithm that takes advantage of known characteristics such as the cylindrical geometry of the bottle, small radius variation between brands and the general location of the labels, a fast and relatively simple wine label recognition application can be implemented.

III. System and Algorithm

A. Mobile Phone

Motorola Droid w/ Android v2.1

550 MHz ARM, 256 MB RAM

Captured Image Resolution 1280x960

Camera Parameters:

- Focus Mode: Macro
- White Balance: auto
- Scene mode: night

We have implemented an Android application with a simple user interface that allows a user to take a picture or select an existing image stored on the phone and upload it to an HTTP server where the computation is implemented in Matlab. The application displays "uploading" on the phone screen when the picture is being uploaded to the server and "finding matches" when the database is being searched. Once a match is found, the server returns the name of the top match along with images of the query and the top three matching wine labels.

Since we tested most of our code during a time when the wireless connection was fast, we only used the android phone to take pictures and transmit them to the server. However, this process can take a significant amount of time when the Internet connection is slow. The ideal implementation would down sample the image and extract as well as compress SURF based image descriptors before sending data to the server. However, the speed gain achieved by using the phone for image pre-processing versus sending an entire image to the server depends on the bandwidth of the Internet connection that is used.

B. Server

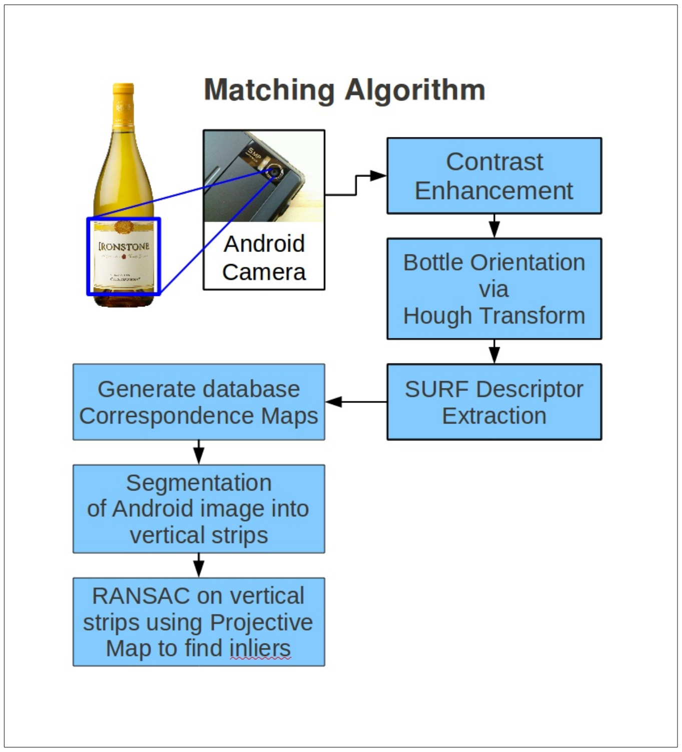
Stanford myth cluster, CentOS 5.3 x86_64

Intel Core2 Duo 3.16 GHz, 4 GB RAM

Matlab R2010a

Matlab OpenSURF Toolbox v0.1c (Dirk-Jan Kroon)

Matlab RANSAC Toolbox (Marco Zuliani)


Matching Algorithm
Android Camera
Contrast Enhancement
Bottle Orientation via Hough Transform
SURF Descriptor Extraction
Generate database Correspondence Maps
Segmentation of Android image into vertical strips
RANSAC on vertical strips using Projective Map to find inliers

Block diagram of visual search algorithm

C. Algorithm Overview

After the image is transmitted to the server a Matlab function reads the input file and begins implementing the wine label recognition algorithm. We have implemented multiple algorithms but some common steps used in all algorithms are:

- Image down sampling to speed up calculations
- Contrast enhancement to improve matching of blurry images
- Feature detection using SURF
- Feature descriptor matching using SURF
- Ratio test to decrease the number of features processed in the next step
- Geometric consistency check by RANSAC
- Sort images by the number of inliers found by RANSAC
- Return the top 3 images with the highest number of inliers

1) Feature Extraction and Matching Methods

Every visual search system needs to have a way to extract distinctive features from the query image and compare them with those found in the database. For our implementation of mobile wine recognition, we use Speeded Up Robust Features (SURF) to extract image descriptors from the query image and search for the best match from the database. SURF is a scale and rotation invariant feature detection algorithm developed in 2006 by by Herbert Bay, Tinne Tuytelaars and Luc Van Gool from Katholieke University[3]. According to [4], a comparison of SURF with other robust feature detection methods such as Scale Invariant Feature Transform (SIFT) and Principal Component Analysis (PCA)-SIFT showed that SURF is the fastest algorithm with the same matching accuracy as SIFT. Since speed is of the utmost importance in a mobile search application, we decided to use SURF although the same study also showed that SIFT is the most stable algorithm under varying imaging conditions.

The features extracted using SURF are compared with a database consisting of feature descriptors, and a RANSAC (Random Sample and Consensus) test is performed to discard feature correspondences that do not follow certain geometric transformations. The RANSAC algorithm was first introduced by Fischler and Bolles in 1981 as a method to estimate the parameters of a certain model starting from a set of data contaminated by large mounts of outliers [5].The percentage of outliers which can be handled by RANSAC can be larger than 50% of the entire dataset [6]. We have compared implementations of our wine label recognition algorithm using a RANSAC algorithm with an affine transformation and homography as well as a segmented affine model approximation of a cylindrical transform.

Finally, in addition to a pipeline consisting of SURF followed RANSAC we also considered using affine SIFT which is an affine invariant version of SIFT developed by Jean-Michel Morel and Goshen Yu [7]. The use of this algorithm could potentially eliminate the RANSAC parameter estimation step, significantly increasing the speed of the search. However, we were not able to test this implementation due to lack of time.

2) *Approximating a Cylindrical Correspondence Map for RANSAC*

The correspondence map between a flattened image of a label and a photograph of the label on a bottle is non-linear due to the cylindrical projection caused by the bottle. The map between two images of a label on a bottle taken from different angles is also non-linear. We wanted our algorithm to be invariant to the angle of the user's camera, so we needed to find a way to reverse this deformation

We analyzed the geometry of the cylindrical projection on paper and determined that a good model would require twelve degrees of freedom to achieve the transformation as long as we could assume that both images contained upright labels. We could therefore design a function to undo the projection given six independent point correspondences. One set of three correspondences uniquely determines the location of a parabola with an axis of symmetry parallel to the bottle. Another set of three correspondences determines a second parabola, also with a parallel axis of symmetry.

We can understand the vertical map using two parabolas to define the vertical location of the top and bottom edges of the wine label. We then take a linear combination of the two parabolas to define the vertical location of points lying on horizontal lines on the label.

To determine the horizontal map, we find the horizontal location of the peaks of the two parabolas and use the line between those points to define a principle axis for the bottle. We assume that the projection is symmetric about this axis, so we can use the horizontal locations of correspondences between two points to determine the radius of the bottle and invert the horizontal compression of the label near the edges of the bottle.

At this point, we had already discovered that the speed of our system was sub-optimal even when using RANSAC with a simple affine model. We were concerned that using this model would slow our system down even further, especially since the the large degree of freedom would force us to run more iterations of RANSAC.

We therefore decided to divide the test image into segments that were small enough to be approximated by a linear projective map. Our non-linear model assumes that vertical lines on the flat wine label are kept straight in the cylindrical projection. Although they are not mapped to vertical lines, we made an additional assumption that they would be mapped to nearly vertical lines based on the fact that it's easier for the user to focus the camera when there is little vertical perspective. We used this approximation to justify dividing the test image into three vertical strips of equal width and approximating the correspondence map within each segment as a homography.

$$S = \frac{\log(1-P)}{\log(1-q^k)} \qquad (1)$$

$P-$Probabilty of success

$q-$Probability of valid correspondence

$k-$Number of points needed to determine map

RANSAC requires S iterations to find a map. A projective map requires a minimum k = 4, and our higher order model requires k = 6. When q < 0.59, The number of iterations required for three projective RANSAC runs is less than the number of iterations required for the higher order model. When q < 0.74, two projective RANSAC runs would require fewer iterations than the higher order model. This suggests that comparable time performance can be expected from the simpler segmented linear model, not including any additional time required to compute the non-linear correspondences.

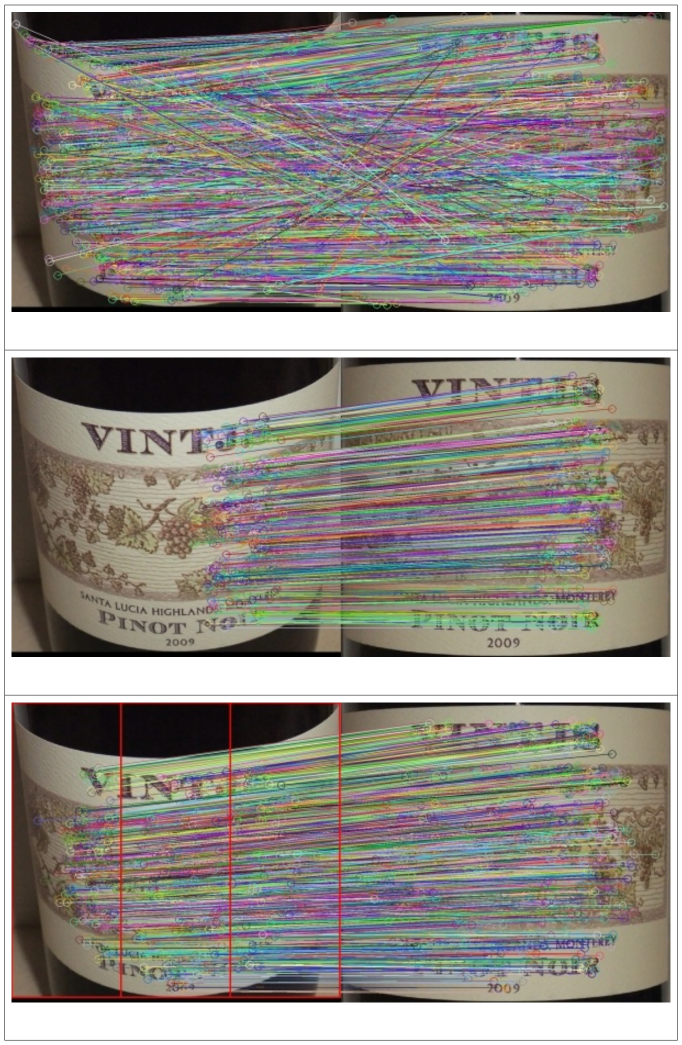


Figure 1. (a) Correspondence map before RANSAC. (b) RANSAC using projective map to find inliers. (c) Vertical segmentation followed by RANSAC on each segment to find three distinct projective maps and achieve greater physical coverage of the label.

IV. Experimental Results

A. *Down sampling and contrast enhancement*

The uploaded and database images are first down sampled to approximately VGA resolution before further processing in order to reduce the number of feature comparisons. Initial testing performed with high resolution (>1000x1000) query and database images resulted in nearly 100% accuracy under various imaging conditions. Figure 2 below shows an example of wine recognition using SURF based feature extraction followed by an affine RANSAC model.

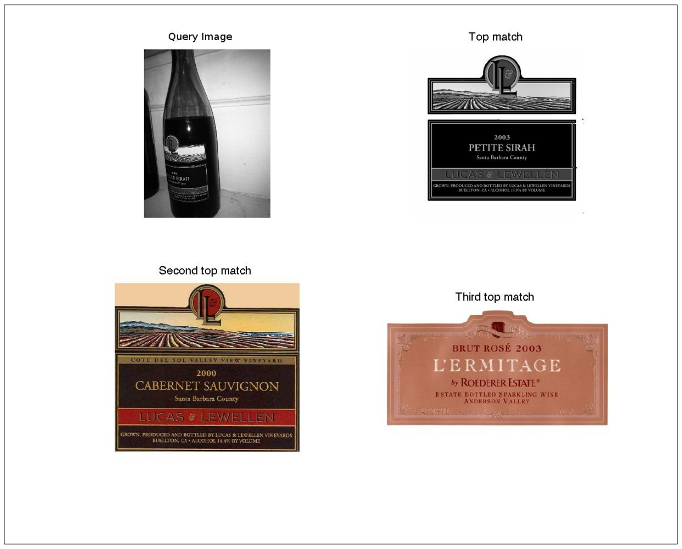


Figure 2.

Despite the high matching accuracy, each search took longer than 10 minutes to complete. After performing various tests, we determined the lowest possible database image resolution that results in an accurate match to be less than 90000MPixels so we used images down sampled to approximately VGA resolution to be safe. During testing, it was found that simply resizing the image to 640x480 resolution resulted in distortion severely degrading the accuracy of the search. So both the database and query images were resized by a scale of sqrt((640*480)/(image_row_size*image_col_size)).

After resizing the query image, contrast enhancement through adaptive histogram equalization was performed in order to improve the matching accuracy of blurry images. Figures 3a and 3b below show a wine a label that was incorrectly matched without contrast enhancement and was correctly recognized when the histogram equalization step was incorporated into the algorithm.

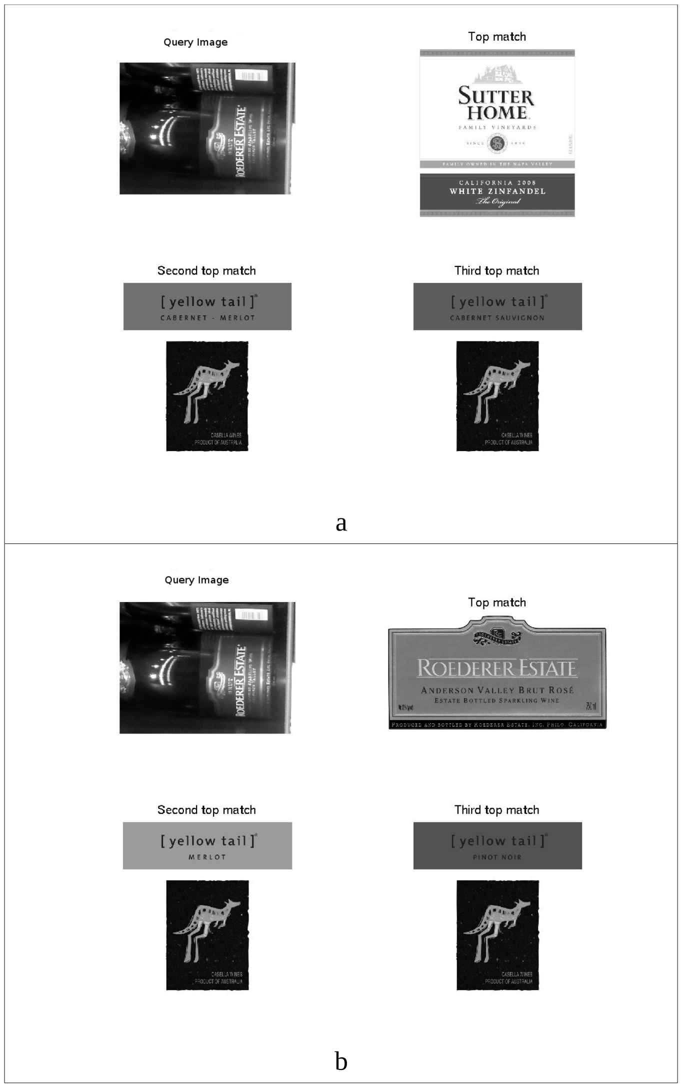


Figure 3. (a) Wine not recognized before contrast enhancement. (b) Wine recognized after contrast enhancement.

B. Feature extraction

Feature extraction and matching were implemented using the matlab openSURF toolbox version of SURF developed by Dirk-Jan Kroon and adapted from a C# implementation written by Chris Evans. 64 feature descriptors extracted from the query image are compared to descriptors corresponding to each stored database image. A descriptor in the query is found to have a matching descriptor in the database when the euclidean distance between the two is less than the euclidean distance between the query descriptor and all other image descriptors in a database image. In addition, the euclidean distance between the two closest descriptors has to be greater than 1.25 times the distance between the query descriptor and its second closest neighbor in the database image. This ensures that 90% of the false matches are eliminated while only discarding less than 5% of the correct matches [8]. Figure 4 shows an example of the matched descriptors after a ratio test.

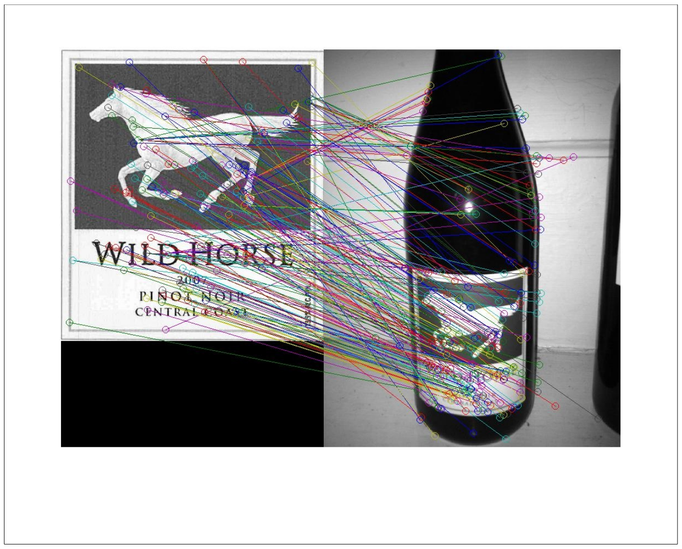


Figure 4. Output after SURF + ratio test

C. Geometric Consistency Test

An affine RANSAC model is used on the retained features to remove matches that do not obey an affine transformation. This section is implemented using the Matlab RANSAC Toolbox by Marco Zuliani. The parameters chosen for the RANSAC model are ebs=1e-3, q=0.3, k=5, tolerant noise=10 pixels and maximum number of iterations=500. Figure 5 shows an example of the matching features retained after a geometric consistency check.

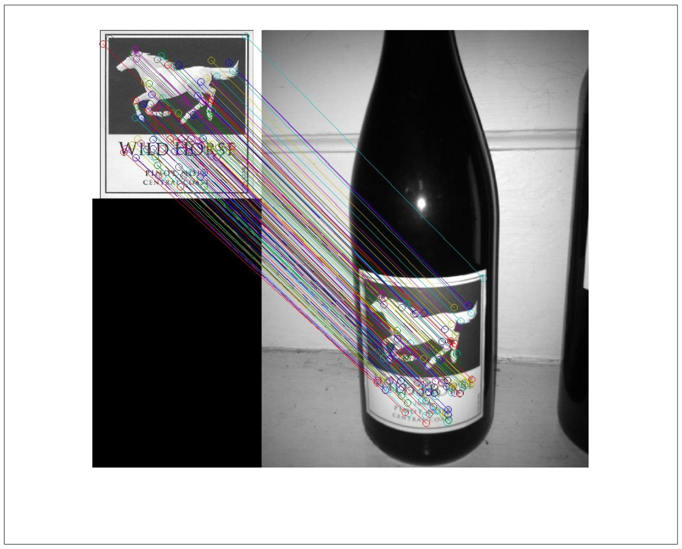


Figure 5. Output after RANSAC

In addition to an affine RANSAC model, one with homography was also tested. However, a model utilizing homography resulted in minimal improvement in matching accuracy and a higher latency due to its increased number of parameters. This is because most of the wine labels are fairly flat. Labels that were consistently not recognized by an affine model were ones that spanned a large portion of the bottle, resulting in significant distortion. A model utilizing a projection transformation is unlikely to detect these significantly warped labels. Figure 6 shows one of the two wine labels that were correctly recognized using homography while an affine model failed to return the correct match. As can be seen from the figures, the query image was significantly foreshortened which is why a projection model worked better than an affine model.

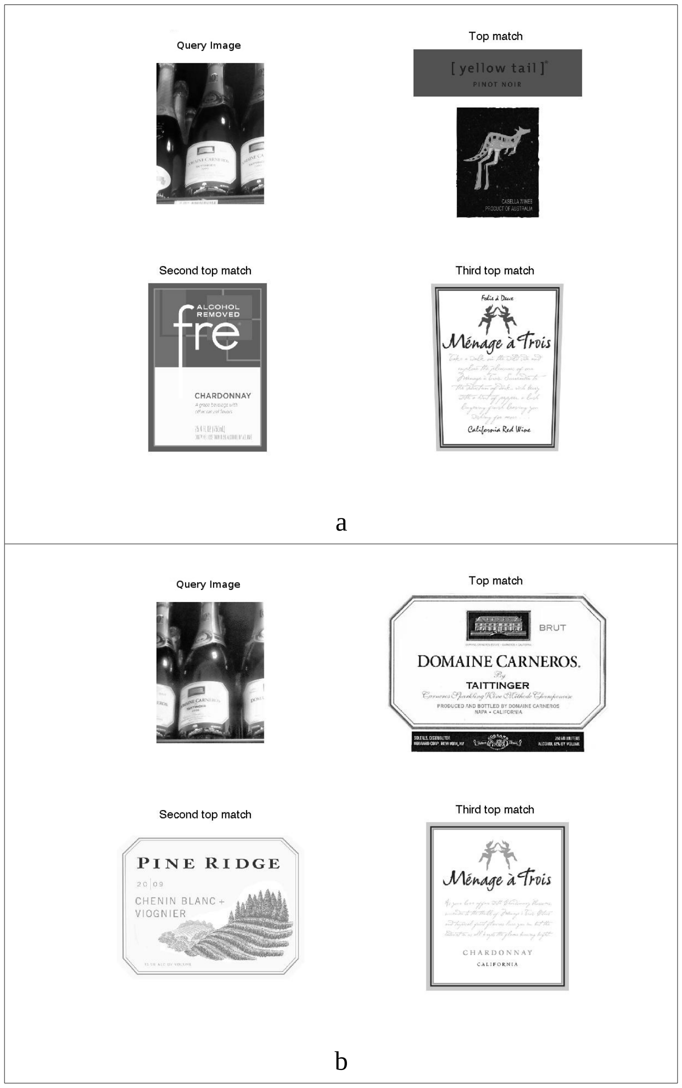
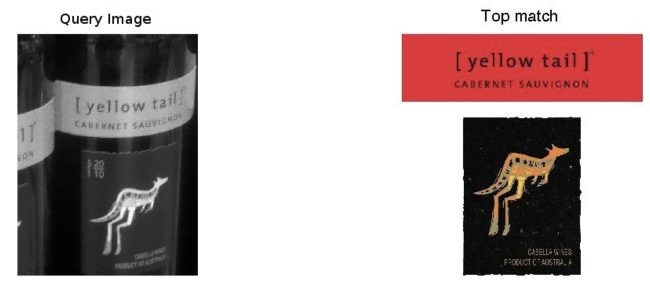


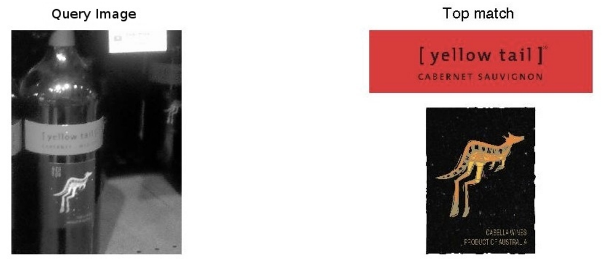
Figure 6. (a) Image unrecognized with an affine model. (b) Image recognized with a homography.

D. Performance statistics

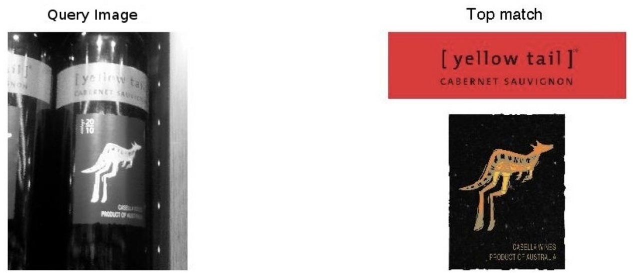
All testing was performed with a database of 43 wine labels and 63 search images consisting of pictures of wine bottles on the shelf of a local grocery store. A database of flat labels was compared with one consisting of labels taken from bottles. While 70% of the images were recognized with a database of flat labels, 80% were correctly matched with a database consisting of labels on bottles. The search time for each query was between 30 to 40 seconds. Figure 7 shows examples of a wine label recognized under various imaging conditions. As can be seen from the figure, the brand of the wine as well as the type were correctly identified.



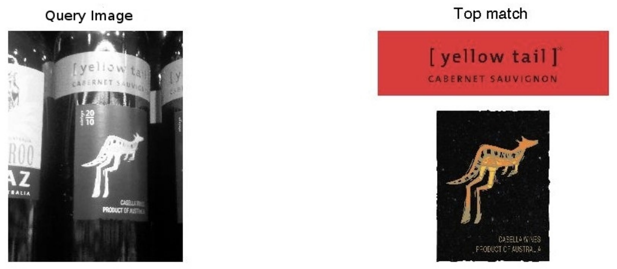
a



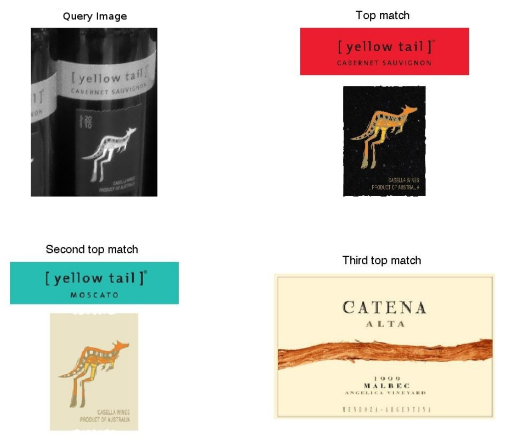
b



c



d



e

Figure 7.

V. Conclusions

One of our primary obstacles in this project was the speed of the image search. Early tests took nearly 15 minutes to complete a search in a database containing only 43 descriptors. This was primarily due to the fact that we were trying to extract SURF features from our test image at the camera resolution (1280x960). Down-sampling to VGA resolution reduced this time to about one minute at the expense of accuracy. From a test set of 63 image searches, 95% of the labels were correctly identified when the full image resolution was used where as 80% were recognized with down-sampling.

We used the open source Matlab toolboxes “OpenSURF” and “RANSAC Toolbox” to implement our search algorithm. Since these toolboxes include a significant amount of debug and profiling code, we would expect to see a nontrivial reduction in search time by implementing our algorithm with an optimized library such as OpenCV. For this application to be practical, the search database would have to grow to include many thousands of labels in which case pairwise matching would be impractical. This can be overcome by creating a database whose entries are part of a vocabulary tree based on feature histograms.

Division of Labor

Timnit Gebru worked on the algorithm pipeline and performed many of the speed and accuracy tests. Oren Hazi developed and tested the method for approximating a cylindrical correspondence map for RANSAC. Vickey Yeh worked on the prototype Android application.

Acknowledgment

We would like to thank Professor Berndt Girod, David Chen, and Derek Pang for teaching EE 368 and providing additional help after class. We would also like to thank Gabriel Takacs and Vijay Chandrasekhar for mentoring us during the course of this project.

References

[1] Chen, D., Tsai, S., Kim, K.-H., Hsu, C.-H., Singh, J. P., Girod, B., "Low-Cost Asset Tracking using Location-Aware Camera Phones," SPIE Workshop on Applications of Digital Image Processing (ADIP), San Diego, California, 2010.

[2] http://ideeinc.com/products/tineyemobile/

[3] Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., "SURF: Speeded Up Robust Features," Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346-359, 2008.

[4] Luo Juan & Oubong Gwun, A Comparison of SIFT, PCA-SIFT and SURF. International Journal of Image Processing (IJIP) Volume(3), Issue(4), pp. 143-152. 2009.

[5] M. A. Fischler and R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (1981), 381–395.

[6] Marco Zuliani, RANSAC for dummies, unpublished, 2009

[7] Jean-Michel More and Guoshen Yu, A New Framework for Fully Affine Invariant Image Comparison, Siam J. Imaging Sciences Vol. 2, No.2, pp. 438-469