Abstract—This report details a mobile system capable of recognizing multiple foreign banknotes with varying levels of occlusion and converting the value of each note into U.S. dollars. The system relies upon a client-server architecture, in which the mobile device sends an image potentially containing banknotes to the server, the server applies the proposed algorithm to process the image for banknotes, and the server then sends the image back to the mobile device with an additional layer detailing banknote positions, countries of origin, and value in both local currency and USD. The recognition algorithm utilizes SIFT and RANSAC to detect banknotes in a variety of orientations, scales, and settings.

Keywords—Banknote Recognition; SIFT; RANSAC; nDLT;

I. INTRODUCTION

In recent years, the proliferation of and improvements in mobile devices have drastically altered the realm of achievable tasks while the user is on the go. Paired with high quality cameras and improved wireless data connectivity, the mobile device now offers near real-time solutions to a wide variety of common problems, hindrances, and inconveniences.

One particularly frustrating inconvenience involves keeping track of the value of foreign goods and services in terms of domestic currency, often United States dollars (USD), while abroad. This issue is amplified when the traveler must pay for these goods in the local currency and amplified further when the traveler moves between foreign nations and acquires currency from each nation. To combat this potentially frustrating situation, an automatic, mobile banknote recognition and currency conversion system would prove a useful tool for the traveler.

Several prior implementations of banknote recognition have proved to be successful, with algorithms involving SURF and wavelet transforms in applications ranging from improving the quality of life for the blind to increasing the accuracy of ATM counting [1]-[3]. However, these implementations have involved only a few banknotes from a single country.

This paper details a system capable of recognition of multiple foreign banknotes with reasonable occlusion and subsequent conversion of the value of the imaged banknotes to USD. The algorithmic recognition relies heavily upon the Scale-Invariant Feature Transform (SIFT) as detailed in [4], as well as Random Sample Consensus (RANSAC) [5].

II. ALGORITHM AND ANDROID IMPLEMENTATION

A block diagram of our algorithm is shown in Fig. 1, 2. The following sections detail the steps required for population of the banknote database and banknote recognition on a mobile device.

Prior to implementing a mobile banknote recognition system, a database of banknote images with which the mobile images will be compared must be assembled. Under the pretense that comparisons will be performed on keypoint feature descriptors, banknotes amassed for the database are first individually scanned into a computer and then pre-processed by applying SIFT to each side of the banknote separately. By applying a peak threshold of 0.5, SIFT will produce several thousand keypoints per image, of which only the top five hundred will be kept for computational efficiency during comparisons. The feature descriptors of these keypoints, along with the banknote country of origin, the side of the banknote imaged (front or back), and the local value of the banknote are stored in the database.

With the database populated, mobile banknote recognition is implemented via the following steps:

A. Mobile Device Photograph

With our Android application loaded onto and running on a user’s camera-equipped, mobile device, the user initiates the process of identifying and converting foreign currency into United States Dollars (USD) by taking a minimally-blurred photograph of the banknotes in question. This image is then uploaded to a remote server via the device’s wireless data service, predominantly Wi-Fi.

B. Perform SIFT

Once the image has been uploaded to the server, the server initiates a MATLAB script that performs SIFT on the received image. The top three thousand keypoints are kept for further processing. There is an unknown number of banknotes in the image, so a small peak threshold is necessary to capture the
target number of banknotes. Empirically, the peak threshold used for each image is 0.1.

![Feature Matching Diagram](image)

**C. Feature Matching**

With the largest image features captured, comparison to the banknotes in the database is now possible. In parallel, features from each side of each banknote in the database are compared to the image features using the VLFeat command `vl_ubcmatch` with a ratio distance threshold of two. The resulting set of feature matches are cleaned to remove matches in which different features in a database banknote are mapped to the same feature in the image. The remaining feature matches are then stored according to the associated database banknote.

**D. Geometric Consistency Checks via RANSAC**

The feature matches are then run through several checks, beginning with a geometric consistency check using an implementation of RANSAC, to iteratively find banknotes throughout the image. This series of checks can be seen in Fig. 2.

Taking the matched features associated with this database banknote, a variation of the RANSAC algorithm will be applied to these features four thousand times in parallel to determine the best homography between this banknote and the image.

For each iteration, RANSAC selects a random subset of matched features and estimates a homography from this subset. For speed, the subset consists of a minimum of four matched features. The homography is produced via the normalized direct linear transformation (nDLT) algorithm, as proposed in [6]. This homography is then applied to all features of this database banknote to determine its ability to accurately describe the transformation to the banknote in the image. This accuracy is quantified through the quantity of inliers, which are described in the equations below:

\[
\mathbf{v}_{est,i} = H \mathbf{v}_{db,i}
\]

\[
\|\mathbf{v}_{est,i} - \mathbf{v}_i\|_2 < \text{threshold} \Rightarrow \mathbf{v}_i \text{ is an inlier}
\]

where \(\mathbf{v}_i\) is the image descriptor of the \(i\)th matched feature, \(\mathbf{v}_{db,i}\) is the database descriptor of the \(i\)th matched feature, and \(\mathbf{v}_{est,i}\) is the estimate of the \(i\)th image descriptor as described by the homography, \(H\). Each homography is given a score equivalent to the number of inliers produced by it. In the implementation of this algorithm, threshold is empirically set to thirty.

Out of the four thousand transformations produced above, the homography with the largest score is assumed to be the most accurate transformation from the banknote in the database to the banknote in the image. This homography, its score and its associated inliers are stored for comparison. This above operation is done for every banknote in the database.

After producing the best homographies for each database banknote as described above, the banknote with the highest score, or number of inliers under its best homography, is assumed to be the most likely to appear in the image. Its homography, inliers, and database values are stored for further processing.

After determining which banknote has the highest probability of existing in the image, its score is checked against a threshold, empirically set at ten. This test determines whether any banknotes remain in the image, and causes section E to execute if no banknotes are present. If the test determines that a banknote does exist in the image, then the edges and centroid of the banknote are calculated using the optimal homography and stored for display, along with the information about the banknote in the database. Following this storage, the inliers are deleted from the global set of matched features, so that the banknote is not detected again.

One final check is then applied to the remaining image. If the ratio of the number of remaining features to the number of features before this set of consistency checks is greater than a threshold, empirically set to 0.2, then addition banknotes are assumed to exist in the image, and section D is run again on the remaining features. Otherwise, the remaining image is assumed to be devoid of banknotes, in which case section E is run.

For all future geometric consistency checks, the centroid check is now applied. The centroid check takes the best homography generated for each banknote in the database and calculates that banknote’s bounding box and centroid, in the sample image, according to the homography. If the homography’s centroid lies too close to any previously found banknote’s centroid, the homography’s score is set to 0. A bounding box of width and length 60 pixels is used. This was empirically found to be the best performing.

The effect of the above geometric consistency check can be seen in Fig. 3. It should be noted that performing this same set of geometric consistency checks a second time on only the inliers produced by the best homography would improve
accuracy but would also drastically increase the runtime, and therefore, this extra check has been omitted.

Fig. 3. Geometric Consistency Check result. Top: All matched features. Bottom: Inliers from highest scored homography.

E. Display Result

After capturing all detected banknotes in the image, a layer is painted on the original image. Using the location of each banknote in the image, the country of origin, and local currency value, the value in USD is calculated and shown at the centroid of each banknote. In addition, the corners of each banknote are determined from the stored data and highlighted in the image. Finally, the sum of the USD values of all detected banknotes is calculated and displayed in the upper left corner of the image.

With the image now highlighting detected banknotes and their values, the server sends the image back to the mobile device, which displays the image until the user clicks a button. An example of the final display image is shown in Fig. 4.

III. Experimental Results

A. Server and Database Setup

To optimize speed, the server is running Apache, PHP 5 and a Matlab Automation service. The Matlab Automation service allows an instance of Matlab to stay in memory, entirely eliminating delay due to boot time, and can be communicated with by using the COM protocol via PHP. This setup additionally allows for access to the server’s 4 cores without setup delay. The Android application uses a POST request to push images to the server and the server streams binary data back to the phone, following processing.

Our database includes 14 banknotes (front and back), totaling 14,000 keypoints. These banknotes span 8 currency types that can be used in a total of 34 countries.

B. Single Banknote Detection Results

All experimental results show that the algorithm is capable of identifying a single, non-blurry, banknote with 100% accuracy. In practice, with exposure to multiple backgrounds, low lighting conditions and partial occlusion the algorithm has not failed. In some cases the algorithm can even detect a folded banknote. However, the algorithm is known to fail for blurry images or images at extreme scales.

C. Multiple Banknote Detection Results

The algorithm has also proved to be effective for multiple banknotes including cases of identical banknotes, overlap and occlusion. Results have shown that the algorithm can detect up to twelve non-overlapping banknotes. Beyond twelve banknotes, important features become too small to detect due to camera resolution. The algorithm can additionally identify multiple identical overlapping banknotes. In all tested cases, for up to four banknotes with the current configuration, banknotes are properly identified 95% of the time. Table 1 shows a subset of our experimental results.

Table I. Repeatability Results

<table>
<thead>
<tr>
<th>Image Type</th>
<th># Samples</th>
<th># Correct</th>
<th># Incorrect</th>
<th># Missed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Banknote</td>
<td>40</td>
<td>40</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2 Banknotes</td>
<td>40</td>
<td>40</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2 Banknotes With Overlap</td>
<td>40</td>
<td>40</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3 Banknotes</td>
<td>60</td>
<td>60</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3 Banknotes With Overlap²</td>
<td>120</td>
<td>100</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>4 Banknotes</td>
<td>80</td>
<td>80</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig. 4. Single Banknote Detection

Fig. 5. Two banknote detection with overlap, occlusion, and noise
<table>
<thead>
<tr>
<th>Image Type</th>
<th># Samples</th>
<th># Correct</th>
<th># Incorrect</th>
<th># Missed</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Banknotes With Overlap</td>
<td>40</td>
<td>40</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5 Banknotes</td>
<td>50</td>
<td>50</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5 Banknotes Overlap</td>
<td>50</td>
<td>30</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>6 Banknotes</td>
<td>60</td>
<td>60</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6 Banknotes Overlap</td>
<td>60</td>
<td>50</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>9 Banknotes</td>
<td>90</td>
<td>81</td>
<td>0</td>
<td>9</td>
</tr>
</tbody>
</table>

*Results include failures due to the centroid check because of banknote orientation in the sample.

Fig. 6. Twelve banknotes detected with modified algorithm parameters which include seven thousand features

D. Design Tradeoffs

All of the previously mentioned, empirically determined thresholds in the algorithm can be adjusted. The current configuration is tuned to work best with 4 overlapping banknotes. It is possible to adjust the thresholds to detect many more banknotes as mentioned above. This, however, comes at the cost of longer runtime and oversensitivity to single banknotes.

The iterative nature of the algorithm causes the runtime to be a non-linear function of the number of banknotes detected. For 4 banknotes, including normal image upload and download times, a result is produced in about 10 seconds.

IV. CONCLUSION

This paper has demonstrated an effective, automated, mobile banknote recognition and currency conversion algorithm and application capable of discerning between multiple banknotes from multiple countries. It is, however, not a completed product, and several modifications can be undertaken to improve and expand upon the current system.

Perhaps the most obvious improvement entails increasing the database to include a wider array of banknotes from many more countries. This would require some algorithmic improvement to maintain similar runtimes, particularly in feature matching and RANSAC comparisons. One proposed solution involves storing features in a kd-tree and employing clustering techniques to reduce the number of comparisons.

It should be noted that many modern commercial scanners will not scan or copy banknotes from around the world, and the software interfaces will direct users to [7], where users can learn about anti-counterfeiting laws from around the world. This hindrance proves to be exceedingly problematic when attempting to populate the banknote database. One potential solution to this problem and another suggestion for improvement is to enable the mobile user to capture a high-quality image of a single banknote with their mobile device, input the required information about the banknote, and upload the banknote image to the server for processing via SIFT and storage in the database.

WORK BREAKDOWN

M. Digman implemented banknote recognition via SIFT and geometric consistency check, tuned the algorithm parameters, and designed poster. C. Elder implemented a banknote segmentation algorithm (unused in final algorithm), tested the algorithm, and wrote the report.

ACKNOWLEDGMENT

We would like to thank Mina Makar for his invaluable support and advice in all facets of algorithm development throughout the quarter.

REFERENCES