Paper Money Recognizer Using Feature Descriptor

Nur Hadisukmana, Adri Yudianto N P
Faculty of Computing, President University, Cikarang, Bekasi, 17550, Indonesia

Article Info

Article history:

Received Mar 2, 2018
Revised Apr 21, 2018
Accepted Jun 14, 2018

Keywords:

Banknote Blind Code
Feature Descriptor
Feature Matching
ORB Algorithm
Paper Money Recognizer

ABSTRACT

People are still using paper money for daily transaction; this, however, will expose some difficulty for visually impaired people. Though they can still read the nominal value of the paper money by the help of other people or by touching the tactile feature, they cannot depend upon others all the time nor touching the tactile feature properly if the paper money is worn. Some alternatives have been proposed and conducted. One of them is using money value recognition application. The application will recognize nominal value of paper money comparing the image of the paper with database. This process is using a feature extraction algorithm called ORB feature descriptor. It has been used for six (6) different types of currencies that are 5 most traded currencies and Indonesia currency and also for different types of nominals (bills).

Corresponding Author:

Nur Hadisukmana,
Faculty of Computing,
President University,
Cikarang, Bekasi, 17550, Indonesia.
Email: nurhadisukmana@president.ac.id

1. INTRODUCTION

In everyday life, people are using banknotes for business. They play important role in financial transactions and are used as official currency and medium of exchange for goods and services. It is expected that every person can use it without needed helps from others. The nominal value of paper money or banknotes should be easily to recognize for all people.

They, however expose some difficulty to certain disable persons. Visually impaired (Blind) people hardly recognize the nominal of the banknotes or paper money so they need help from another person for reading the nominal of the banknotes.

Some states like Canada and Australia have been trying to add some blind code features to their paper money, such as tactile feature, a braille-like feature [1]. Whereas Indonesia has implemented the blind code in the form of the combination of braille feature and embossed triangle, circle, and rectangle shaped [2]. Unfortunately, these blind code features do not work properly whenever the banknotes are crumpled or the paper money are worn out.

To eliminate the problem, many researches have been conducted. Some the researches produced devices for identifying and recognizing banknotes (currency) [3], [4]. Though the devices work well, they are too rigid and dedicated devices, i.e., they can be used for only certain banknotes. Enhancement of device functionalities needs improvement of device circuitry.

Other researches produced applications such as a mobile application for currency recognition and authentication [5]. However, again it does work only for a specific and certain currency and it is not clear whether it uses some feature detection method for recognizing the object.

This paper introduces some application system to recognize the nominal of various banknotes or currencies for visually impaired people with high accuracy. It uses feature extraction algorithm called ORB feature descriptor [6].

2. THEORITICAL REVIEWS

This section discusses some fundamentals study on Computer Vision especially image processing, feature detector, feature descriptor, and matching algorithm.

2.1 Grey Scale

Image processing is a method to convert an image into a digital form and perform some operation on it. An image is an array or matrix of pixels represented in rows and columns. In which contains information of intensity represented by Blue-Green-Red (BGR) or gray-scale [7].

The BGR image contains information of blue, green, and red intensity. Meanwhile, gray-scale image only contain the intensity of white. Specifically, an 8-bit gray-scale image contains black intensity that the value ranged from 0 to 255. The lower its value, the darker the image, so a pixel with intensity of 0 is black, meanwhile a pixel with intensity 255 is white. An illustration of pixel in a gray-scale image can be seen at Figure 1.

Figure 1. Gray-Scale Illustration, from 0 (left) to 255 (right)

2.2 FAST Feature Detector

Feature is a piece of information in image that contained in key points that are relevant to the image. A key point itself is a point in an image that interesting enough to be analyzed. Usually, human recognize this "interesting point" as dot, cross, or corner. There are some algorithms to detect features in an image, one of which is FAST.

FAST (Features from Accelerated Segment Test) is an algorithm that used to detect a feature key point. FAST algorithm is very fast, because the idea is it works by only comparing the intensity of its 16 neighboring pixels as it is shown in Figure 2.

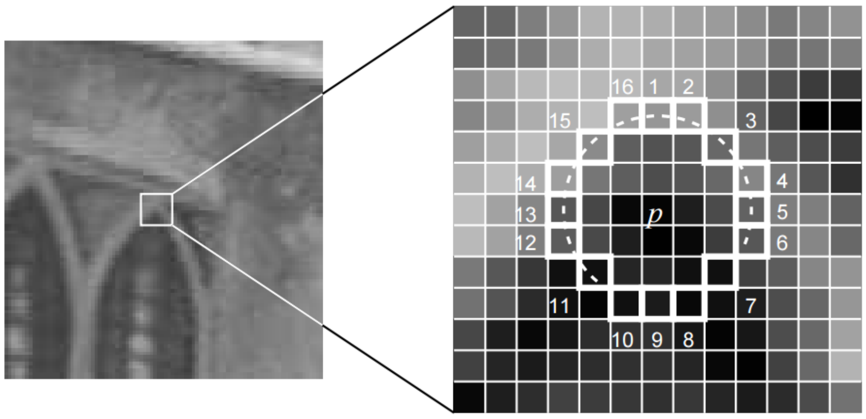

Figure 2. Test pixel (p) and its 16 surrounding pixels [8]

The first step is to take a test pixel p from the image. Then, assume that the intensity of this pixel is I_p. Set a threshold intensity value T. Assume that this threshold intensity value T is 20% of the text pixel.

Next, consider a circle of 16 pixels surrounding the test pixel with radius of 3. Then, check the intensity of those 16 pixels whether it is above or below the I_p by the value of T. If there are more than 12 pixels fit that criteria, then the test pixel can be considered as an interesting point. Repeat the procedure for all pixels from the image.

To fasten the algorithm, the computer can check the pixel umber 1, 5, 9, and 13 first. If at least three out of four pixels are above the threshold value, then check all the 16 pixels. With this additional step, the computer will have early information before checking all the 16 pixels [9].

2.3 BRIEF Feature Descriptor

After detecting key points from the image, the next step is to describe the key point itself. This description is to find the similarity between 2 key points. The process to find the similarity between features will be explained further in the matching algorithm.

There are terms such as scale invariant and rotation invariant. A scale invariant descriptor means that this descriptor will detect a same image with different scales as similar. Meanwhile, a rotation invariant descriptor means that this descriptor will detect a same image with different angle of rotation as similar.

There are two types of descriptor from how they contain the description data, which are binary and non-binary descriptor. One of binary feature descriptors is BRIEF (Binary Robust Independent Elementary Features).

BRIEF descriptor is more efficient than non-binary feature descriptors, because it is using a binary feature that are computational and space cheap. BRIEF works by randomly take a pair of points A and B in a key point, and then compare its intensity. If point A is having a greater intensity value than point B, then it will produce a binary of 1, else it will produce binary of 0. Repeat this process until 256 times. Finally, a 256 bit feature is produced. It uses random sample to select the test location [10].

2.4 ORB Algorithm

Even though the performance of BRIEF is better than non-binary feature descriptors, BRIEF is not classified as rotation invariance descriptor, which means that it cannot detect same result (called as invariance) if the object is rotated.

ORB (Oriented FAST and Rotated BRIEF) is an algorithm that combined the FAST detector and BRIEF descriptor. Similar to BRIEF, ORB also produces a 256 bit feature. The idea of this algorithm is to make a binary descriptor that both scale and rotation invariant.

ORB works by calculating the orientation when detecting the key point using FAST algorithm. It can find the orientation by simply calculating the centroid, center of mass, of the patches around each key points. Not only that, by making all the patches uniform with centroid, it turns out that there is an optimal value from BRIEF descriptor that can be trained instead of only using random pairs [7].

2.5 Feature Matching

Feature Matching is an algorithm that is using features to find the similarity between images. The idea of finding a match is by finding the least difference between features. This difference is called distance. Images to be matched can be classified as query image and train image. Query image is the image whose features to be matched in train image. Query image is captured directly from camera. Meanwhile, train image is a set of images in which the features of the images have been detected and extracted. The features of such images in the train image are usually saved into file or database. The relation between query and train images is shown by Figure 3.

It can be seen, from Figure 3, that each feature in query image can only have another one match with feature in train image. Meanwhile, each feature in train image can have multiple matches with features in query image.

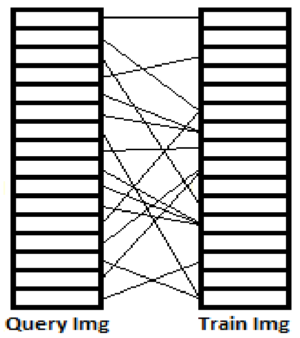

Figure 3. Matching algorithm demonstration

Brute force feature matching is a method to compare features in query and train images by searching the closest distance between a feature in query image and features in train image. This process will be done for all features in query image. So, if there is M number of data in query image and N number of data in train image, there will be M x N process of hamming distance.

Figure 4 shows the process of brute force feature matching algorithm. The first step (a) shows the process on finding the shortest distance between a feature in query image and all features in train image. Each line in the picture shows the hamming distance process occurs in the process. Meanwhile, the thick red line shows the minimum distance. Step (a) to step (c) shows that first, second, and third feature in query image done respectively. Finally, the last step (d) shows that each feature in query image already has its match, or finds its minimum distance.

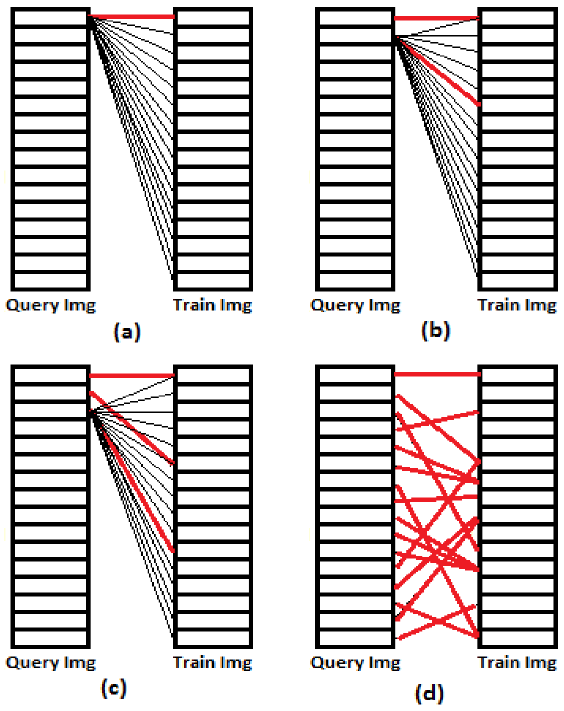

Figure 4. Brute Force Feature Matching

2.5.1 K-Majority Algorithm

The feature matching can be speed up by implementing K-Majority Clustering to reduce number of features in the list. K-Majority algorithm works by firstly choosing K number of data that will be used as centroid. Centroid is a center of cluster, which all of the corresponding data assigned to. The next step is to assign the corresponding data to the centroid by choosing the closest distance between the data and centroid using hamming distance. After assigning all the data to corresponding centroid, the next step is to find the new centroid from the corresponding data by using majority voting. Since the data is in binary string form, majority voting can works well by choosing the “winner” by counting the number of votes between 0’s or 1’s in each bit. The process of hamming distance, majority voting, and updating centroid will be repeated until the centroid is not changed anymore.

3. RESEARCH METHOD

The aim of the research is to develop an application that could detect the nominal of the banknotes and/or the paper money of several currencies implementing fundamentals that have been discussed in the previous section. This application will help and assist the visually impaired people to determine the values of the paper money they have. It, therefore uses Rapid Application Development (RAD) for the application.

The application consists of two main parts that are train image generator and nominal or image reader. The first part of the application will generate dataset from paper money images (train image) and save the set into database. The set will be used as references for the second part of the application to identify the image that is captured (query image) and give correct and valid solution of the image which is the nominal of the paper money. The application model is shown by Figure 5.

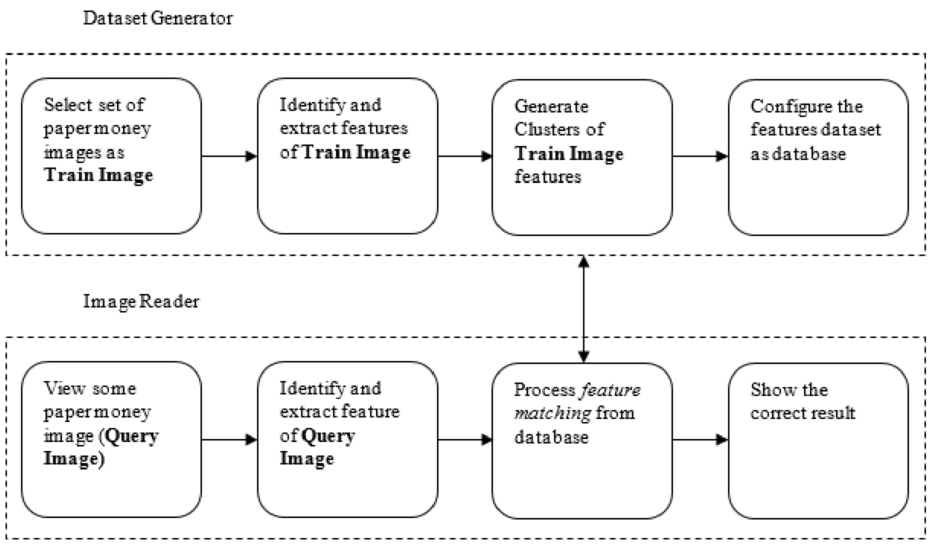
Dataset Generator
Select set of paper money images as Train Image
Identify and extract features of Train Image
Generate Clusters of Train Image features
Configure the features dataset as database
Image Reader
View some paper money image (Query Image)
Identify and extract feature of Query Image
Process feature matching from database
Show the correct result

Figure 5. Work flow of the Application

4. COMPONENT DETAILS

Figure 5 shows components of the two parts of the application. This section discusses briefly about roles of the components.

4.1 Dataset Generator

In order to generate dataset that is saved in database, this part has been divided into several components.

4.1.1 Select Train Image

This component will select a set of paper money images to be used as **train image.** Each of the images will be saved into a file. These file must be saved into a specific folder in which this component will easily pick all of the files. Every paper money has two images because paper money has two sides. The component also parses the properties of each image via its filename into nominal, currency, and side of the image and saved into some data structure called PDT (Picture DaTa) in dataset.

4.1.2 Image Identifier and Extractor

This component will identify or detect key points of all images in the train image. After that, it will transform these key points into the features of the images. These two important works are achieved by ORB algorithm. Key point identifications are done by ORB descriptor whereas feature extractions are conducted by implementing ORB descriptor. Figure 6 shows key point identification of Indonesia paper money (IDR) whose nominal is 20.000 rupiahs. The image features data are saved into a different data structures named as TC0 (Transform and Compile) in the dataset. The existence of TC0 is crucial especially in feature matching between query image and an image in train image.

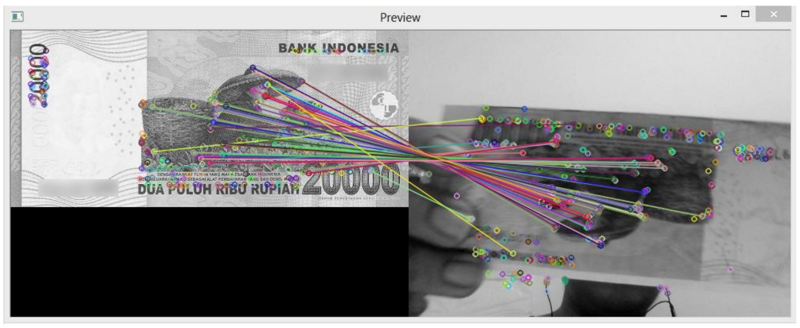

Figure 6. Key point Identification

4.1.3 Feature Cluster

In order to reduce number of features of train image, these features can be clustered by using K-Majority algorithm that is briefly explained in section 2.5.1. The result of feature clustering will be saved into another data structure, named as TC1 in the dataset. Using K-Majority algorithm, number feature entries in TC0 will be assumed points while the entries in TC1 will become centroids.

4.1.4 Dataset Configuration

All data that are produced and saved in appropriate data structures at previous stages must be configured as *dataset* in such a way that it will be easy to retrieve to match an image of train image and query image and therefore it could show the nominal value of the query image. The configuration of the dataset is shown by Figure 7.

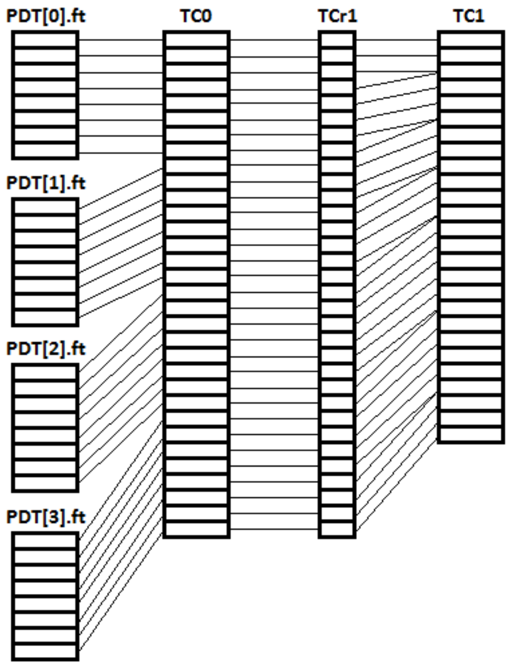

Figure 7. Dataset Configuration

Since number of entries in TC1 is less than that of entries in TC0 then an additional data structure must be added. This data structure, named as TCr1, has a function as routing table that maps one or more entries in TC0 onto one entry in TC1.

4.2 Image Reader

This second part of the application has also several components; each of which has different and specific role as discussed briefly below.

4.2.1 View Image

The second part of the application starts by viewing an image of paper money of the selected currency. This action is achieved by activating the camera that is embedded in a mobile device. This process will take some duration of time since it also activates the second component of Image Reader to identify key points of the image.

4.2.2 Image Identifier and Extractor

After activating the mobile device camera which views a paper money image, this component is automatically activated. It starts detecting key points of the image and then creates and extracts features of the image from the key points. The features of the image are sent the next component.

This process uses similar algorithm as described in section 4.1.2. The difference is that this component identifies key points and extracts them into features of single image while the component in section 4.1.2 implements the same algorithm for a set of images.

4.2.3 Feature Matcher

This component runs whenever the features of image generated by the previous component are ready. The role of this component is to find an image in the dataset that has similar features with the image that is viewed by the camera. It searches the features in TC1 by finding the smallest hamming distance. This is done by using Hamming brute force algorithm. Once this step is completed, the process continues finding the index of the features in TC0 through TCr1. Then it uses the TC0 index to get the correct data that are nominal, side and currency of the image in PDT.

4.2.4 Show Result

This component has role to show the result from the previous stage (component). The result is actually data text containing the information of the nominal, the side and the currency of the paper money image that is viewed by the camera of a mobile device. This data text is then passed to an API that could convert the data text into a speech. Eventually the application is using English-based text to speech API.

5. RESULT AND ANALYSIS

The development of the application has shown several things:

a. It processes a set of images from 6 different currencies, i.e., Indonesian Rupiah (IDR), the United Stated of American Dollar (USD), Australian Dollar (AUD), European Community Euro (EUR). United Kingdom Poundsterling (GBP), and Japan Yen (JPY).
b. It produces around 2500 ORB features for the images.

Results of the application have been compared to others, especially [4] for a set of nominals IDR currency. Comparisons are done in two aspects: the accuracy of recognizing the nominals and response time of the applications.

This section would also show the results of the application in the two aspects for all other currecies on the average.

5.1 Recognition Accuracy of Two Methods

Figure 8 shows the application accuracy in detecting and recognizing the currency nominals. It also shows the accuracy of [4] to recognize the nominals. Figure 8, and also figure 9, use OCR and ORB in the legend to represent this application and [4].

Several things can be pointed out from figure 8.

a. Legend ORB represents recognition methods that used by this application whereas OCR is used by [4].
b. Nominals of IDR currency are 1,000, 2,000, 5,000, 10,000, 20,000, 50,000, and 100,000 rupiahs, which become X-axis.
c. The minimum accuracy value that is obtained by ORB method is 87% for 5000 rupiah, whereas OCR produces minimum value around 55% for the same nominal.
d. The maximum value obtained by ORB is around 97% which happens for 100000 rupiah. The same thing also happens in OCR for 20000, 50000 and 100000 rupiahs.
e. ORB method produces more stable result in recognizing the nominals compared to OCR. ORB gives the difference around 10% between minimum and maximum values. OCR produces a bigger difference value that is 42%.
f. On the average, ORB produces around 92.86% of accuracy which is better than OCR that gives around 85.71%.

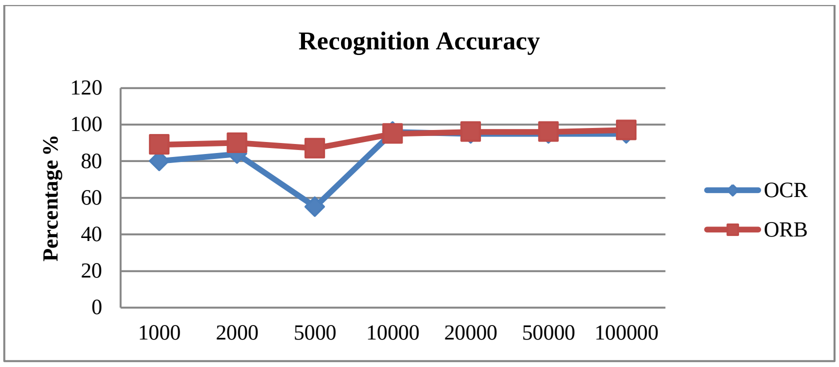

Figure 8. IDR Nominals-Recognizing Accuracy

5.2 Response Time of Two Methods

Figure 9 shows the application response time in detecting and recognizing the currency nominals. It also shows the accuracy of [4] to recognize the nominals.

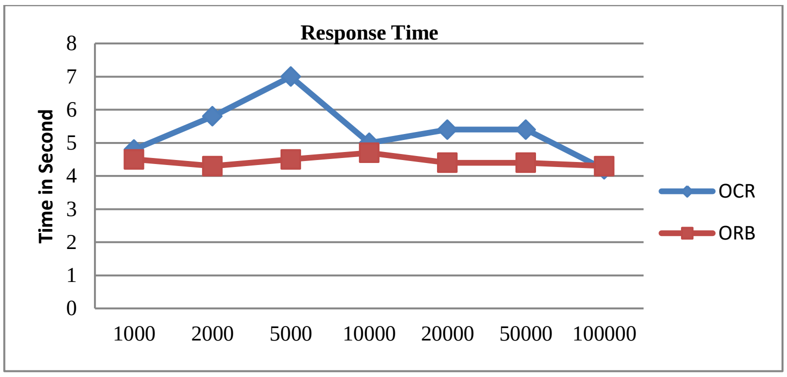

Figure 9. IDR Nominals-Recognizing Response Time

Several things can be pointed out from Figure 9.

a. Nominals of IDR currency are 1,000, 2,000, 5,000, 10,000, 20,000, 50,000, and 100,000 rupiahs. They function as X-axis.
b. The longest response time produced by ORB method is around 4.7 second for 10000 rupiah, whereas OCR produces the longest response time is 7 second, which happens for 5000 rupiah.
c. The fastest response time produced by ORB is around 4.3 second which happens for 2000 and 100000 rupiahs. The OCR produces the fastest response time of 4.2 second, which happens for 100000 rupiah.
d. ORB method produces more stable performance in response time compared to OCR. ORB gives the difference around 0.4 second between minimum and maximum values. OCR gives a bigger difference value that is 2.8 second
e. On the average, ORB produces response time around 4.4 second of accuracy and OCR gives around 5.4 second.
f. One keypoint of stable performance of ORB is that the use of K-Majority algorithm for feature matching process

.

5.3 Currencies Recognition Accuracy

Figure 10 shows the accuracy performance of the application to detect and recognize for all currencies. It gives average values from set of nominals from each of currencies.

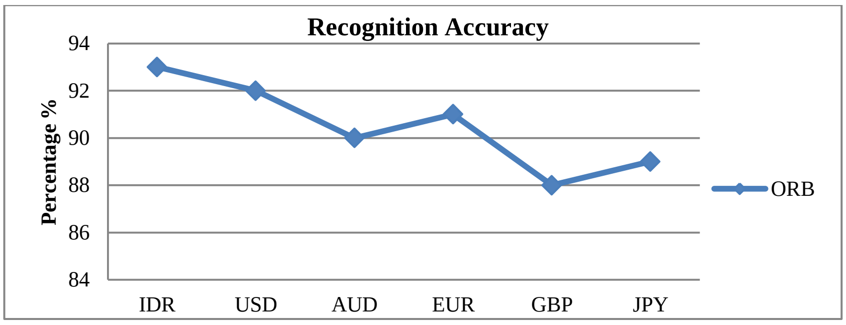

Figure 10. Currencies-Recognizing Accuracy

Figure 10 also shows that:

a. Currencies IDR, USD, AUD, EUR, GBP, and JPY function as X-axis.
b. The minimum accuracy value that is obtained by ORB method is 88% for GBP.

c. The maximum value is around 93% for IDR.
d. Eventhough the accuracy of ORB method tends to be decreased, but the performance of the application is considered to be stable because ORB gives small difference range around 15% between minimum and maximum values.
e. On the average, ORB produces around 90.5% of accuracy for currencies applied.
f. The application, on the average, gives high accuracy in detecting and recognizing the currencies.

5.4 Response Time for Currencies

Figure 11 shows the application's average response time performance in detecting and recognizing set of nominals of currencies that are applied. This could also show:

a. Currencies IDR, USD, AUD, EUR, GBP, and JPY function as X-axis.
b. The maximum response time that was obtained by ORB method is 5.2 second, which occurs for GBP.
c. The minimum response time is around 4.4 second for IDR.
d. Eventhough the resnponse time of ORB method tends to be incresed, but the performance of the application is considered to be stable because ORB gives small difference range of response time, which is around 0.8 second between minimum and maximum values.
e. On the average, ORB gives response time of 4.9 second for currencies applied.
f. The application, on the average, produces quite fast response time, which is less than 5 second, in detecting and recognizing the currencies.

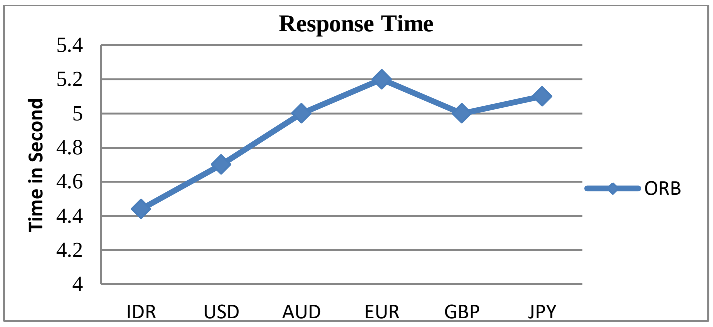

Figure 11. Currencies Response Time

6. CONCLUSION

The research on the problem of the visually impaired people in recognizing the paper money, especially the worn out paper money, has enabled the development of mobile application utilizing the camera of the mobile device. The application is able to able recognize the nominal of the paper money from any different angle even if the paper is rotated. It also recognizes the nominal of the paper money even though the paper is folded or worn out. The response time to recognize the nominal of the paper money on the average is less than 5 second with the accuracy averagely 90%.

An extension of the research can be conducted to improve the following.

a. Increasing the accuracy of the application up to 100%.
b. Minimizing the response time to at most 3 second.
c. Providing more text to speech API than English-based only.

REFERENCES

[1] Samuel, C. Making Bank Notes Accessible for Canadians Living with Blindness or Low Vision. *Bank of Canada Review (Winter).* 2009-10; 29–36.
[2] Hartono Try, Cahaya Fitra Roman. The Effectiveness of Blind Codes and Security Features for The Blind and The Prevention of Fraud on The Indonesian Rupiah. *Asia Pasific Fraud Journal*. 2017; 2(2)
[3] A. Hinwood,P. Preston, G. J. Suaning, N. H. Lovell. Bank note recognition for the vision impaired. *Australasian Physical & Engineering Sciences in Medicine*. 2006; 29(2): 229-233.
[4] Gunadhi Albert, Ivan Daniel, Lestariningsih Diana, Andyardja Widya, Yuliati. Identification Tool of Rupiah Banknote for Blind People with Audio Output Using Optical Character Recognition (OCR) Method. *ARPN Journal of Engineering and Applied Science.* 2016; 11(13): 8113-8121.

[5] Sagar Swati, Modal Shaheen, Seth Apoorva, Shah Roopvati, Deshpande Akanksha. An Android Application for Indian Currency Recognition and Authentication for Blind. *International Journal of Computer Science and Network*. 2016: 5(2): 324-329.
[6] Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary R. Bradski: ORB: An efficient alternative to SIFT or SURF. *ICCV* 2011: 2564-2571.
[7] Christopher Kanan, Garrison W. Cottrell, Color-to-Grayscale: Does the Method Matter in Image Recognition?, *PLoS ONE*. 2012; 7(1): 1-7.
[8] Rosten E, Drummond T, *Machine learning for high speed corner detection*. 9th Euproean Conference on Computer Vision. 2006; 1: 430–443.
[9] Viswanathan, Deepak Geetha. Features from Accelerated Segment Test (FAST). 2011; 1-5.
[10] Calonder Michael, Lepetit Vincent, Strecha Christoph, Fua Pascal. *BRIEF: Binary Robust Independent Elementary Features*. 11th European Conference on Computer Vision. Heraklio. 2010; 778-792.