Currency Recognition System Using Image Processing

EEE F435

Digital Image Processing

Chinmay Patil
2014A3PS243P

Manish Sharma
2014A3PS181P

Background and Motivation

There are approximately 50 currencies all over the world, with each of them looking totally different. For instance the size of the paper is different, the same as the colour and pattern. The staffs who work for the money exchanging (e.g. Forex Bank) have to distinguish different types of currencies and that is not an easy job. They have to remember the symbol of each currency. This may cause some problems (e.g. wrong recognition), so they need an efficient and exact system to help their work. The aim of our system is to help people who need to recognize different currencies, and work with convenience and efficiency.

For bank staffs, there is a "Currency Sorting Machine" helps them to recognize different kinds of currencies. The main working processes of "Currency Sorting Machine" are image acquisition and recognitions. It is a technique named "optical, mechanical and electronic integration", integrated with calculation, pattern recognition (high speed image processing), currency anti-fake technology, and lots of multidisciplinary techniques. It is accurate and highly-efficient. But for most staffs, they have to keep a lot of different characteristics and anti-fakes label for different commonly-used currencies in their mind. However, each of them has a handbook that about the characteristics and anti-fakes labels of some less commonly-used currencies. Even for that, no one can ever be 100 per cent confident about the manual recognition.

Otherwise our system is based on image processing, techniques which include filtering, edge detection, segmentation, etc.
In order to make the system more comprehensive, we need to create a small database for storing the characteristics of the currency. The system will be programmed based on MATLAB and include a user-friendly interface.

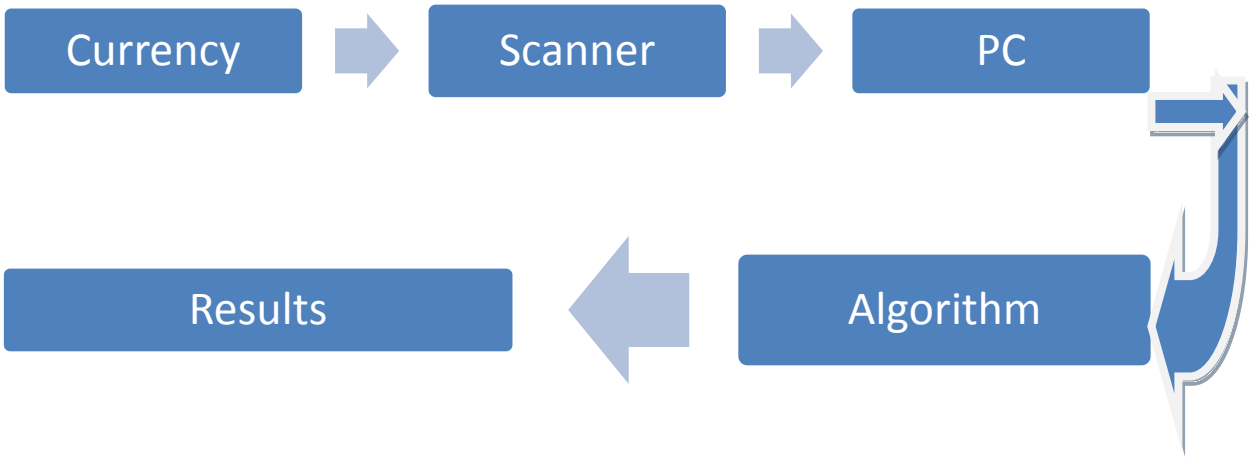
1. Project Tasks

The main steps in the system are:
1. Read image, reading the image we get from scanner as well as the format of the image is JPEG.
2. Pre-processing, removing noise, smoothening image.
3. Image process, edge detection, segmentation, pattern matching. 4. Results printing.

Basically the images are read from different derivations. However, we delimit our system which can read the currency from scanner. The device that the system needs is very common in our daily life, so we do not need to buy an extra device to realize the system.

1.1 **Theoretical background**

The system is based on scanner, PC, and algorithm. The aid of the algorithm is located in the unique figure, RGB to Gray, image binarization, noise elimination, segmentation, pattern matching, etc. We realize there by programing with MATLAB®. Flowchart of the system is described in the following figure-



1.1.1 Image format

The image we get from scanner is formatted by JPEG. JPEG (Joint Photographic Experts Group) is a standard for destructive or loss compromising for digital images. When you save the image as JPEG, the image will lose some information, and this cannot be recovered.

1.2 Technical background

The major technique of this system is image analysis and image processing, which are part of cognitive and computer science. Image processing is a signal processing after pre-processing. The output can be either an image or a set of characteristics or parameters related to the image. Actually the image is treated as 2-dimensional signal and applies some standard signal processing techniques with image-processing techniques involved. Image analysis is a means that the meaningful information from an image is extracted mainly from digital images by means of digital image processing techniques. Image analysis tasks can be as simple as reading bar coded tags or as sophisticated as identifying a person from their face.
In order to recognize the currency, the system should contain the techniques which include image pre-processing, edge detection, segmentation, pattern matching.

1.2.1 Image pre-processing

Image pre-processing is used for operations on images at the lowest level of abstraction. The pre-processing do not increase image information content but decrease it if entropy is an information measure. For example as Histogram equalization, it modifies the brightness and contrast of the image, making the image look more clear. The other example is to remove the noise on the image and improve the quality of edge detection (image).

1.2.2 Edge detection

Edge detection is used in image analysis for finding region boundaries.
Edge and contours play a dominant role in human vision and probably in many other biological vision systems as well. Not only are edges visually striking, but it is often possible to describe or reconstruct a complete figure from a few key lines.

1.2.3 Segmentation

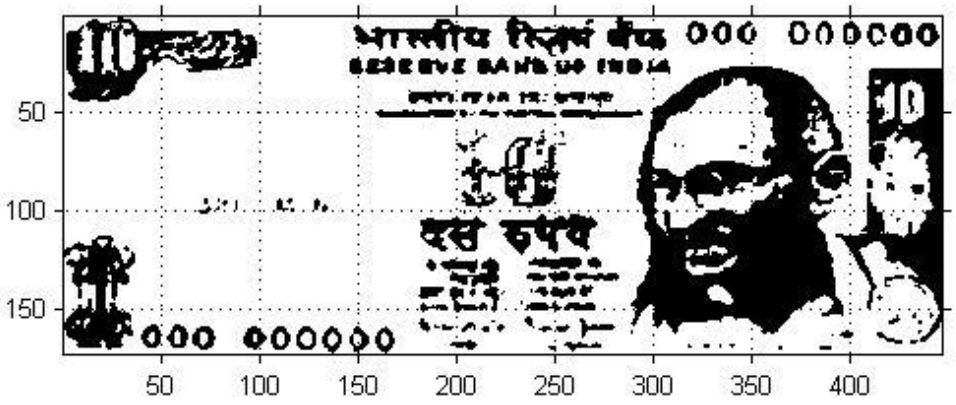
Segmentation is one of the important parts to process image data. It aims to divide an image into parts that have a strong correlation with

objects. In our project, we should use it to segment different parts of the feature.

Figure 1-Indian Rs 10 Note

1.2.4 Binary image

Binary image is the image that only contains two possible intensity values. Usually black and white are used in binary image, and the value of each pixel is stored as 0 or 1. The example of binary image is shown as Figure 6, Binary image.



1.2.6 Gaussian blur

In order to remove noises on the image, Gaussian blur is performed. Convolving the image with a Gaussian function is performed when a Gaussian blur is applied. The equation of a Gaussian function in one dimension is Equation.

$$G(x,y) = \frac{e^{-\frac{x^2+y^2}{2\sigma^2}}}{2\pi\sigma^2}$$

Where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and □ is the standard deviation of the Gaussian distribution.

1.2.7 Histogram equalization

Histogram equalization is used to adjust contrast based on the image's histogram. This method is useful for the image with background and foreground that are both bright or both dark.

2. Method

We built the system by MATLAB®. As the description on its website "MATLAB® is a high-level technical computing language". It has variety API for image processing, so it makes our work become simple and effective.

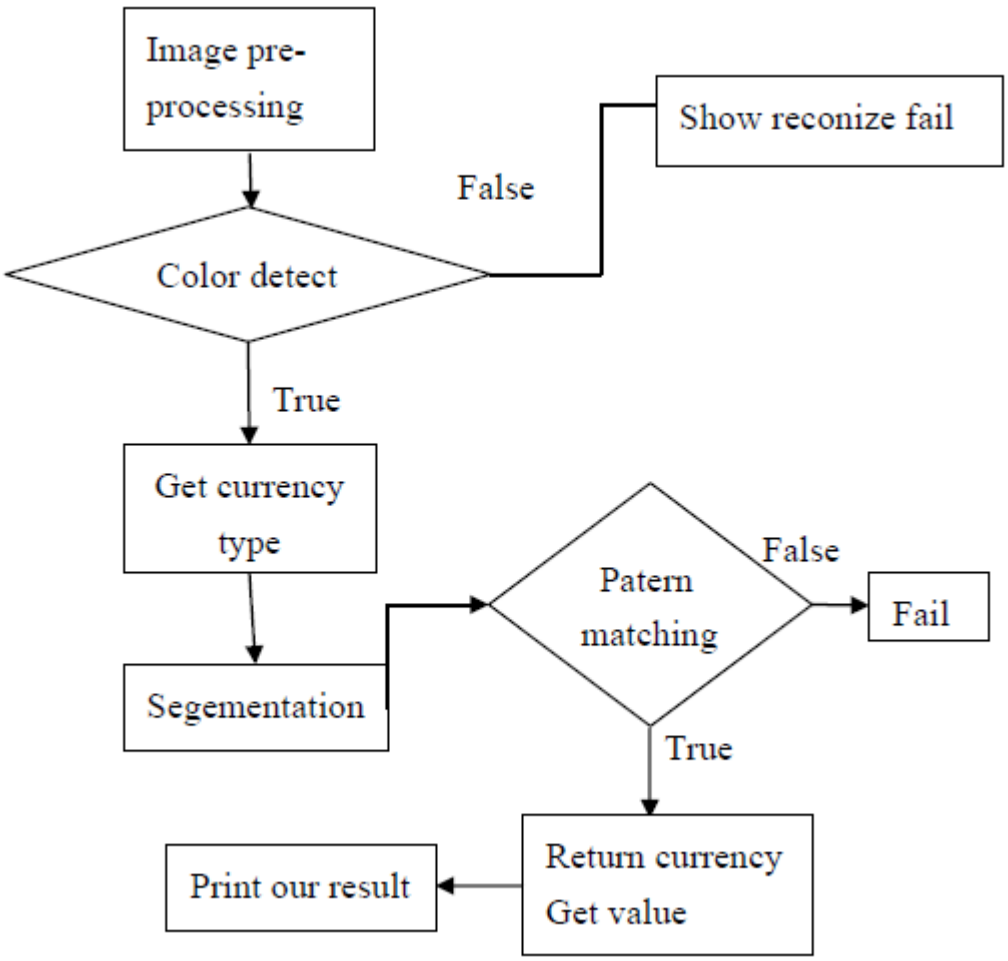
All the currencies that we test are taken from the scanner. The resolution is set to 600 DPI (Dots Per Inch). DPI is means that the number of pixels per unit area, that is a scan precision. The smaller dpi is the lower-resolution scans perform. Otherwise higher-resolution scans perform. As we know the size of A4 paper is 21cm × 29.7cm, 600 dpi means that each inch contain 600 pixels. So after scanning, we get the size of the image is 7016 × 4961 pixels. In order to decrease the computation, the system will reset the original image to size 1024×768 pixels. This work is done in pre-processing.

2.1 System

The system will ask the user to take the image of currency when launching. After that, the system tries to recognize the currency. When the recognition processing starts, the system inside will do some image processing with the image (pre-processing, segmentation, edge detection and so on). If the image exhibit information loses such as surface damage, noise level, sharpness issues and so on, the recognition may fail and the user has to do the processing again.

The system do not need extra device, our algorithm relies on visual features for recognition. It can recognition the currency, and print out the result by text.

The system contains a complete user interface, the user just needs to open the image, and the result will be shown after processing.
The most important part of this system is pattern matching. Based on edge detection algorithm, the system performs pattern matching. When some error occurs, the system will emerge some exception, which may cause the exceptions like “the image not complete”, “failed recognition”, etc. The main steps of the algorithm are illustrated in the figure below.



2.1.2 User interface

The user interface is for printing out the original image and special features.

2.1.3 Read in image

The system can read not only JPEG (JPG) format but others. Our image was obtained from a scanner. As mentioned before, the resolution is set to 600 DPI. But this will make the image a big size. So after reading in the image, the system will reset the image to size 1024 by 768 pixels and this work will refer to image pre-processing.

2.1.4 Image pre-processing

The aim of image pre-processing is to suppress undesired distortions or enhance some image features that are important for further processing or analysis.

In our work, image pre-processing includes these parts:

- Image adjusting
- Image smoothening (removing noise).

When we get the image from a scanner, the size of the image is so big. In order to reduce the calculation, we decrease the size to 1024×768 pixels.

When using a digital camera or a scanner and perform image transfers, some noise will appear on the image. Image noise is the random variation of brightness in images. Removing the noise is an important step when image processing is being performed. However noise may affect segmentation and pattern matching.

After processing with median filter, the noise is removed so well, and some detail is described so well on the image. The pattern which is the most important thing that we want to find is also clear.

Median filter replaces a pixel via the median pixel of all the neighbourhoods:

$$y[m,n]=median\{x[i,j],(i,j)\in w\}$$

Where w represents a neighbourhood centred on location. After removing the noises, the next step is to cut off some useless area. Sometimes, for some reasons, some black lines will appear on the edge of the original image, which will affect the next operation.

To avoid this problem, we cut each side down by 10 pixels.

Compared with an A4 size paper, the currency is so small. However, when we get the image from the scanner, the image we get is a picture like an A4 paper. So after scanning, the image will have lots of white area surrounding the currency. Actually this is useless part for recognition. In order to make the system efficient, the white area part will be cut entirely.

Figure shows before cutting and after cutting.

Figure 2 : Before Cutting

Figure 3 After Cutting

Because the light condition, when getting the image from digital camera, we need to perform histogram equalization.
Histogram equalization is used to adjust the contras and brightness of the image, because some part of the recognition based on color processing. Different light conditions may affect the result. So histogram equalization is needed to perform
For segmentation, we removed more things that are not expected by binarizing the image. We had to set a threshold to decide which one is set to “0” (black) and which one is set to “1” (white). Actually the thing we don’t need is set to “1”. We set two values for the threshold, the value of the pixel between those two values is set to “0”, and others will set to “1”. After many times of testing, we set the threshold between 0.05 for Rs.10 note. The result is shown as Figure-

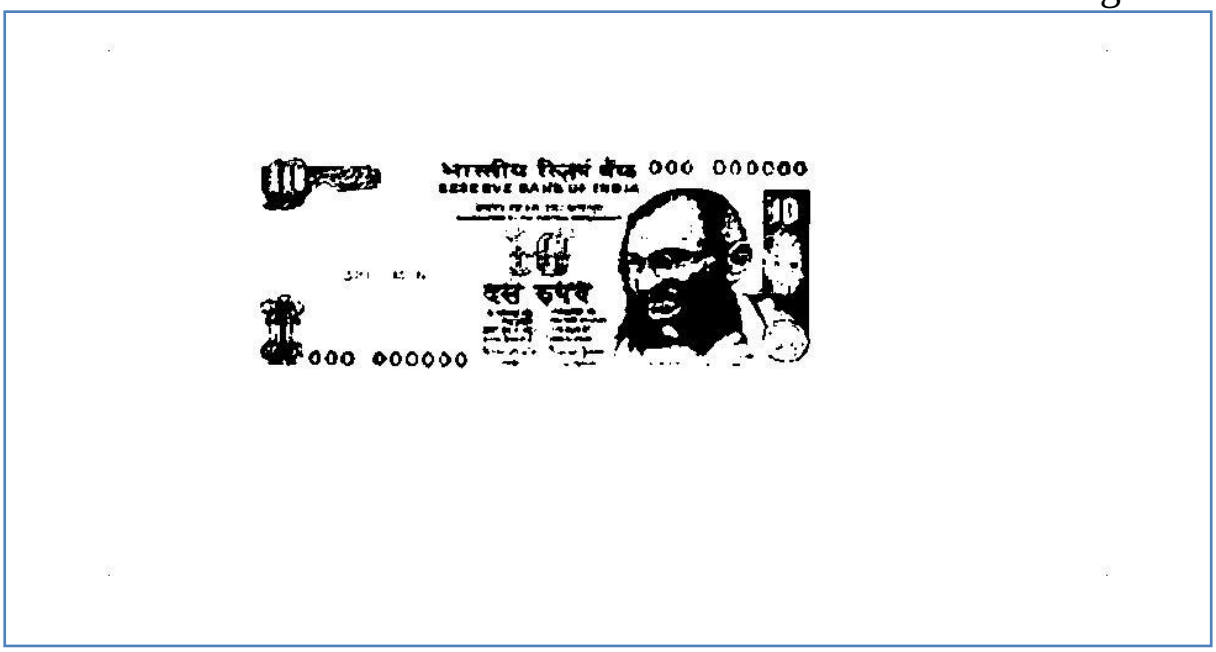


Figure 4-Binary Image

In order to remove the white area, we create our algorithm.
Scan the image by x direction and y direction, and detect each pixel’s value, if the value does not equal to 0 (that means is not a white point), record this point, then continue detecting, if the value equals to 1 (that means is a white point), record this point and break the loop. We set a flag, when flag equals to 0 that means this row or line has been checked, and it contains a black point. So this row or line can be skipped and check next row or line. When meet the row or line which

only consist by white point, we record this row or line. When finishing y direction, do the same thing with the x direction.
Because the system records the points which is first time hit the black one and the white one. We can get the boundary of top and bottom, left and right.
As Figure 18 and Figure 19 illustrate, the first recorded pixel we labelled a red point, and the blue point is the last one.
The core codes for this part are:

```
for i =1:y % The other direction j = 1 : x
        flag=1;
    for j =1:x % The other direction i = 1 : y
        if((J(i,j)==0)||flag==0)
            flag=0;
            count=count+1;
            y1(count)=i; % The other direction
        x1(count)=j
        break
        end
    end
end

PY1 = min(y1); % First record
PY2 = max(y1); % Last record
PX1 = min(x1); % First record
PX2 = max(x1); % Last record
```

When we get the boundary of left, right, top and bottom, we get coordinate of each point, and then the currency is separated successfully

2.1.5 Segmentation

After observation we knew that each currency has one or more unique patterns. So these unique patterns can be used to distinguish different types of currencies.
Because the position of the pattern is aptotic, so we can segment it proportionally and finally get what we want. We set scales for each side of the image, after that, the pattern is segmented.
Here are the core codes for segmentation.

```
PY1 = round((PY2-PY1)*scale_y1)+PY1; % Set the
boundary by proportional
PY2 = PY2 - round((PY2-PY1)* scale_y2);
PY1 = PY1 -10;
PY2 = PY2 +10;
PX1 = round((PX2-PX1)* scale_x1)+PX1;
PX2 = PX2 - round((PX2-PX1)* scale_x2); %Set the
boundary by proportional
```

Actually PY1, PY2 is the boundary of top and bottom. PX1 and PX2 is the boundary of left and right. After calculation, we get the new boundary of the pattern. That is the way we segment the pattern.

Figure 5-After Segmentation

From Figure 5 we can see that not only the pattern, but other thing include the image, so transform this image to binary to remove the thing we did not need.

2.1.6 Colour detection

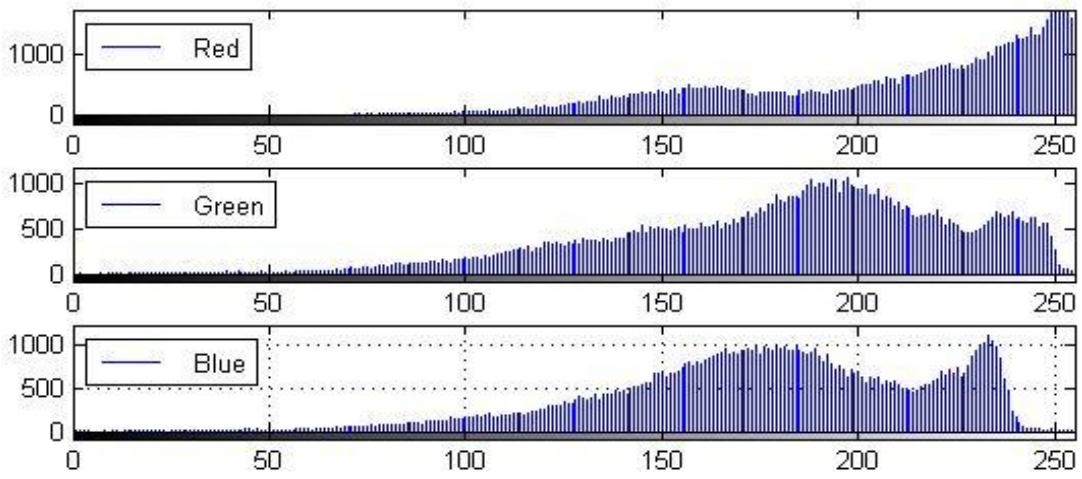
In this part, we are going to describe how to detect the primary colour of the images.
There are too many types of color model we can use, like RGB, HSV, and GREY.

We use RGB model because we need to calculate the mean of the colour. The image is presented as x by y by 3 matrix (here x is the width of the image, y is the height of the image), iteration each pixel and store the value of R, G, and B. After that, the mean of each channel will be calculated.

We do not calculate the whole primary color of the currency. We cut half of the currency, because most of the currencies are dividing into

two parts. And the left part is mostly white area, while the right part has some patterns or portrait.

The primary colour of the image is used to check what this currency is and this is one of the important characters for recognizing the currency. Figure shows the RGB histogram and the image.



2.1.7 Pattern matching

After the pattern is segmented, we need to recognize the pattern. In this part, correlation is performed. Correlation is used to measure the similarity between images and parts of image. When the image consists some objects and regions, and there is an image name template, the template is used to search the object which consists in the origin image. So correlation can use to investigate whether the object is presented in the image.

Actually cross-correlation and convolution is the same procedure. The difference between cross-correlation and convolution is that the

template is not flipped before usage in the correlation case. And the resulting image is larger than original one.
If something matches between the original image and the template image, cross-correlation function will have its maximum value at the position of the object.
The original image and the template image do not need to be of the same size.
Each kernel coefficient in turn and multiplied by a value from the neighbourhood of the image lying under the kernel when convolution is applied.
Top-left corner of the kernel is multiplied by the value at the bottom-right corner of the neighbourhood when we apply the kernel in such a way.
When we rotate symmetric less than 180 degree rotation, the difference will be shown between convolution and correlation. We perform normalized cross-correlation by diving 1 by the result.

Here are some parts of code for cross-correlation:

```
c = normxcorr2(T1,S1); % T1 is the template, S1 is
the image
[max_c, imax] = max(abs(c(:)));
[ypeak, xpeak] = ind2sub(size(c),imax(1));
offset = [(xpeak-size(T1,2)) (ypeak-size(T1,1))];
xoffset=offset(1)+1; % Get the coordinate of X
yoffset=offset(2)+1; % Get the corrdinate of Y
```

Figure 6- Template

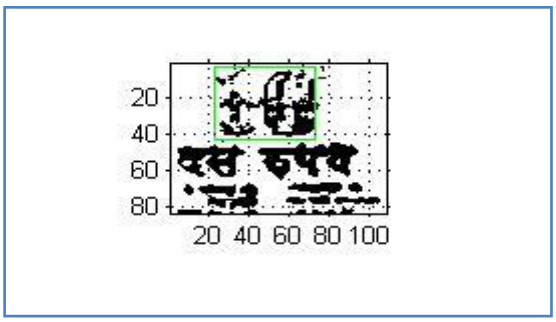


Figure 7 After correlation

We calculate how many white points in the segmented image and the template image.

Here are the core codes:

```
for aa = 1:T1_y
    for bb = 1:T1_xd
        if(edge_T1(aa,bb)~=255)
        black_points = black_points+1;
        end
    end
end
```

Where T1_y and T1_x are the height and the width of the image. After the calculation, we need calculate the difference between those two values. The result we get is used to check if matching is correct

3 Result

For now, the system can check Indian Rs10 and Rs 20 note. If we expand our project by adding more data to our database that is the currency's all unique patterns, then the system can more easily check what the type and value currency is.

After our system is accomplished, we scan more currencies for testing. As far as our test, all the currency whose images are from the scanner can be recognized. And the results show pretty well. Because of the light condition and other condition, the image taken by digital camera cannot be recognized as well. We obtained a 100% result on the scanned images we performed our test. Through these results, with some delimitation, the system performs pretty well. The correct rate is quite high. But if remove these delimitation, we can't get such a high correct rate. We are trying to modify our algorithm to overcome these delimitation, these are the challenges for us to our knowledge.

4 Relevant

Although the aim of our project is for currency recognition, the framework we build also work for recognizing other things, such as

an area in the map, book searching system (Provide the cover of the book, then search from data base).
In our solution, we build the system for someone who needs to distinguish different currencies. But the system can also service someone who is visually impaired, if we put out the result by voice instead by text. They can hear through some output device and get the result.
The system can also be performed based on a website, it can service all the people that need to recognize which currency they have, it is so easy that they just need to get the image of the currency, then upload to the website, and the result will be shown on a web page, it can also show all the information about that currency and how much the currency rate is.

6 Conclusion

This project proposes an algorithm for recognizing the currency using image processing. The proposed algorithm uses the primary color and a part of currency for recognition. We differentiated the denomination of currency using mean value of brightness of R, G and B. This is the first condition to recognize the currency. Following, we segmented the pattern from the currency and performed template matching to check the currency.

The experiment performed by program based on aforesaid algorithm indicates that our currency recognition system based on image processing is quite quick and accurate.

However such system suffers from many drawbacks. The quality of sample the currency, the damage level of the paper currency will affect the recognition rate. And our system still has some limitations, such as the light condition.

7. References

[1] Ahmed, M. J., Sarfraz, M., Zidouri, A., and Alkhatib, W. G., License Plate Recognition System, The Proceedings of The 10th IEEE International Conference On Electronics, Circuits And Systems (ICECS2003), Sharjah, United Arab Emirates (UAE), 2003.

[2] John C.Russ. The image processing Handbook Fifth Edition. Taylor & Francis, North Carolina, 2006.

[3] Mark S. Nixon and Alberto S. Aguado. Feature Extraction and Image Processing. Academic Press, 2008

[4] Rafael C. Gonzalez, Richard E. Woods, Steven L.Eddins. Digital Image Processing Using MATLAB. Pearson Education, 2004.

8. PROGRAM CODE

```
%read image
[note10,map] = imread('ten.png');
figure,imshow(note10)

%Colour detection

count=0;
for i =10:416 % The other direction j = 1 : x
    flag=1;
    for j =10:801 % The other direction i = 1 : y
        if((note10(i,j)~=255)||flag==0)
            flag=0;
            count=count+1;
            y11(count)=i; % The other direction
x1(count)=j
            break
        end
    end
end


count=0;
for j =10:801 % The other direction j = 1 : x
    flag=1;
    for i =10:416 % The other direction i = 1 : y
        if((note10(i,j)~=255)||flag==0)
            flag=0;
            count=count+1;
            x11(count)=j; % The other direction
x1(count)=j
            break
        end
    end
end

PY11 = min(y11) % First record
PY21 = max(y11) % Last record
PX11 = min(x11) % First record
PX21 = max(x11) % Last record
```

```
note10_rgb_crop = imcrop(note10,[PX11 PY11 (PX21-
PX11) (PY21-PY11)]);
figure,imshow(note10_rgb_crop);

figure,
subplot(3,1,1);
imhist(note10_rgb_crop(:,:,1));
legend('Red','Location','NorthWest');

subplot(3,1,2);
imhist(note10_rgb_crop(:,:,2));
legend('Green','Location','NorthWest');

subplot(3,1,3);
imhist(note10_rgb_crop(:,:,3));
legend('Blue','Location','NorthWest');




%Pre-processing
note10 = medfilt2(rgb2gray(note10));
note10 = histeq(note10);




note10_bw = im2bw(note10,map,0.05);
grid on;
axis on;
figure,imshow(note10_bw)

%cutting
count=0;
for i =10:416 % The other direction j = 1 : x
    flag=1;
    for j =10:801 % The other direction i = 1 : y
        if((note10_bw(i,j)==0)||flag==0)
            flag=0;
            count=count+1;
            y1(count)=i; % The other direction
x1(count)=j
            break
        end
```

```
    end
end


count=0;
for j =10:801 % The other direction j = 1 : x
    flag=1;
    for i =10:416 % The other direction i = 1 : y
        if((note10_bw(i,j)==0)||flag==0)
            flag=0;
            count=count+1;
            x1(count)=j; % The other direction
x1(count)=j
            break
        end
    end
end

PY1 = min(y1) % First record
PY2 = max(y1) % Last record
PX1 = min(x1) % First record
PX2 = max(x1) % Last record

note10_crop = imcrop(note10_bw,[PX1 PY1 (PX2-PX1)
(PY2-PY1)]);
figure,imshow(note10_crop);
grid on;
axis on;

GX1 = 200;
GX2 = 250;
GY1 = 60;
GY2 = 100;

note10_template = imcrop(note10_crop,[GX1 GY1 50
40]);
figure,imshow(note10_template);

scale_y1 = 0.4;
scale_y2 = 0.4;
scale_x1 = 0.4;
scale_x2 = 0.6;

SX1 = 0;
```

```
SY1 = 0;
SX2 = PX2 - PX1;
SY2 = PY2 - PY1;

SY1 = round((SY2-SY1)*scale_y1)+SY1; % Set the
boundary by proportional
SY2 = SY2 - round((SY2-SY1)* scale_y2);
SY1 = SY1 -10;
SY2 = SY2 +10;
SX1 = round((SX2-SX1)* scale_x1)+SX1;
SX2 = SX2 - round((SX2-SX1)* scale_x2); %Set the
boundary by proportional

note10_segment = imcrop(note10_crop,[SX1 SY1 (SX2-
SX1) (SY2-SY1)]);

c = normxcorr2(note10_template,note10_segment); % T1
is the template, S1 is the image
[max_c, imax] = max(abs(c(:)));
[ypeak, xpeak] = ind2sub(size(c),imax(1));
xpeak
ypeak
offset = [(xpeak-size(note10_template,2)) (ypeak-
size(note10_template,1))];
xoffset=offset(1)+1 % Get the coordinate of X
yoffset=offset(2)+1 % Get the corrdinate of Y

figure,imshow(note10_segment);
grid on;
axis on;
hold on;
rectangle('Position',[xoffset yoffset xpeak-xoffset
ypeak-yoffset],'EdgeColor','g')

note10_segment_crop = imcrop(note10_segment,[xoffset
yoffset xpeak-xoffset ypeak-yoffset]);
figure,imshow(note10_segment_crop);

black_segment=0;
for i=1:size(note10_segment_crop,1)
    for j=1:size(note10_segment_crop,2)
        if(note10_segment_crop(i,j)==0)
            black_segment = black_segment + 1;
        end
```

```
    end
end

black_template=0;
for i=1:size(note10_template,1)
    for j=1:size(note10_template,2)
        if(note10_template(i,j)==0)
            black_template = black_template + 1;
        end
    end
end

diff = abs(black_template - black_segment);
if(diff < 50)
    result = 'The currency is verified.';

else
    result = 'The currency is not verified'
end

disp(result);
```