Beteckning:________________

HÖGSKOLAN
I GÄVLE

Akademin för teknik och miljö

Currency recognition system using image processing

Yaojia Wang
Siyuan Lin

June 2010

Bachelor Thesis, 15 hp, C
Computer Science

Computer Science program
Examiner: Julia Åhlén
Co-examiner: Goran Milutinovic
Supervisor: Julia Åhlén

Currency recognition system using image processing

by

Yaojia Wang (870712 - 3298)
Siyuan Lin (880310 - 6734)

Akademin för teknik och miljö
Högskolan i Gävle

S-801 76 Gävle, Sweden

Email:
nfk08ywg@student.hig.se
nfk08sln@student.hig.se

Abstract

It is difficult for people to recognize currencies from different countries. Our aim is to help people solve this problem. However, currency recognition systems that are based on image analysis entirely are not sufficient. Our system is based on image processing and makes the process automatic and robust. We use SEK and Chinese RMB as examples to illustrate the technique. Color and shape information are used in our algorithm.

Keywords: currency recognition, image processing

Contents

1 Introduction

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. As we mentioned before, 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. In our system, we take Chinese RMB and Swedish SEK as examples. The system will be programmed based on MATLAB and include a user-friendly interface. 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.

There are some similar recognition systems, such as face recognition system, fingerprint recognition system. However the theories they use are similar but the techniques and approaches are different.

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 Figure 1, Flowchart of the process.

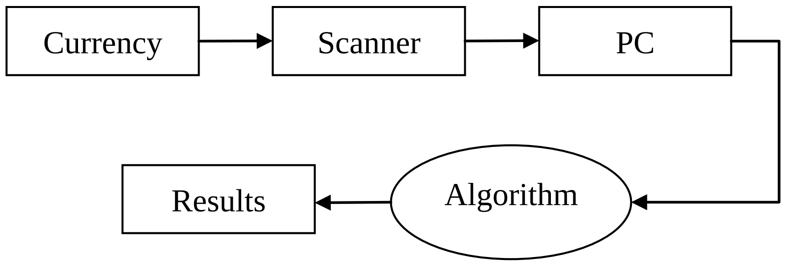


Figure 1, Flowchart of the process

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.

1.1.2 Currency Region Extraction

After our observation, we knew that no matter SEK or RMB, the basic distribution is similar, a head portrait in the right side, a white area in the left side. In the left white area, the water mark is located on it, with some texts on the top. The value's position of the currency is distributed at the corner, and some other patterns are distributed in the almost same region of these two kinds of currencies.

There are also some important characteristics to distinguish the different currencies. The head portrait of RMB is the same person, who is the former Chairman Mao Zedong. But the SEK is totally different with different person on each value. The texts on the top are in two different languages. The value's position of the currency is distributed at different diagonals. The silver thread for SEK is clear and broad. And the counterpart in RMB is inconspicuous and narrow. These differences might be used in our system to help us obtain a more accurate result.

The most obvious and unique characteristics are the colours and the patterns.

Figure 2, Chinese RMB and Swedish Kronor Shows the Swedish Kronor and Chinese RMB.

Figure 2, Chinese RMB and Swedish Kronor

After observing we extract the important features listed in Table 1 Currency features.

Table 1 Currency features

	Color	Value	Water mark	Text	Pattern
CNY	YES	NO	NO	NO	YES
SEK	YES	NO	NO	NO	YES

("YES" and "No" means have been used or unused)

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. [7]

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.[1] 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.[3]

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 3, Pattern in 500 Swedish Kronor is shown the pattern in 500 Swedish Kronor.

Figure 3, Pattern in 500 Swedish Kronor

1.2.4 Color

Color is a property of enormous importance to human visual perception. Hardware will generally deliver or display color via an RGB or HSV model.

RGB model is often used in computer graphics as the basis of a colour space.[2] The RGB model is an additive color model based on primary colours red, green and blue. Each color appears in its primary spectral components.

The main purpose of the RGB colour model is for the sensing, representation, and displays of images in electronic systems, though it has also been used in conventional photography.

HSV model (as shown by Figure 4, HSV model) is one of several color systems used by people to select colors from a color wheel or palette. This color model is considerably closer than the RGB model in the way how humans experience and describe color sensations.[8]

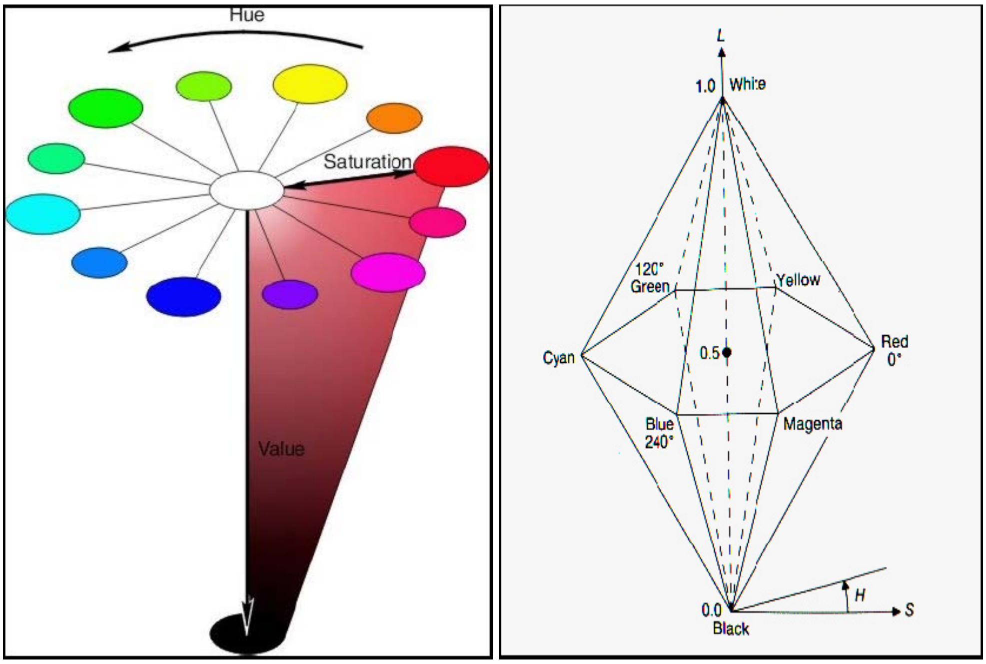


Figure 4, HSV model

HSV is abbreviated to Hue, Saturation and Value. Hue is pure color and is measured by degrees or percentage. Saturation is the radius in the circle. Value (V = 1 or 100%) corresponds to pure white (R = G = B = 1) and to any fully saturated color.

Gray describes the colors ranging from black to white. Gray paints are created by mixing the two colors. A gray scale image is the value of each pixel which is a single sample. That means it carries only intensity information.

Figure 5, Gray image and Gray scale show the gray image and gray scale. From the figure, we can know that, the gray image is based on black and white.

Figure 5, Gray image and Gray scale

1.2.5 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.

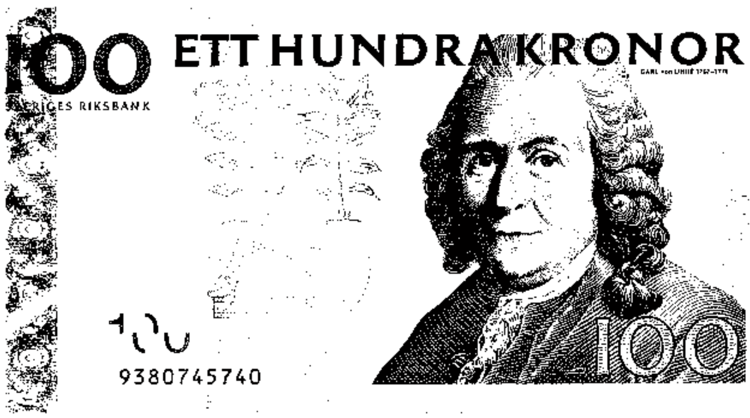


Figure 6, Binary image

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 1. In two dimensions is Equation 2.[1][5]

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \qquad \text{Equation 1}$$

Where x is the distance from the origin in the horizontal axis, and σ is the standard deviation of the Gaussian distribution. [5]

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \text{Equation 2}$$

Where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution. [5]

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.

Chinese CNY is illustrated as Figure 7, and the figure is cut off the white area after the pre-processing.

Figure 7, Chinese 100 Yuan

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. Figure 8 is the Swedish Crown we take by a scanner.

Figure 8, 100 Swedish Kronor

2.1.1 System launch

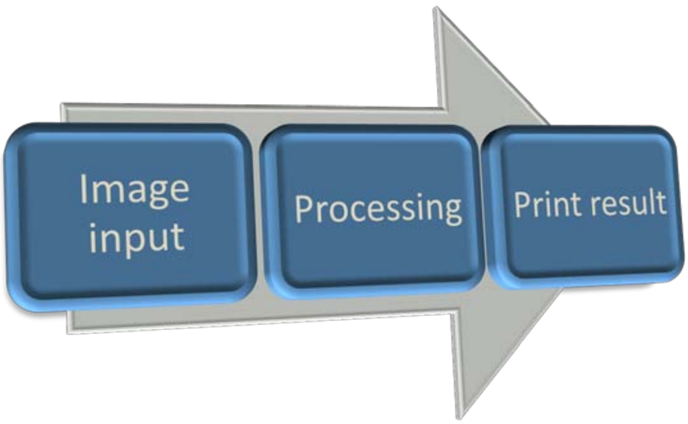


Figure 9, Flowchart of the system launch

The system contains a complete user interface, the user just needs to open the image, and the result will be shown after processing.

Figure 9 shows the process that the system performs.

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 as Figure 10.

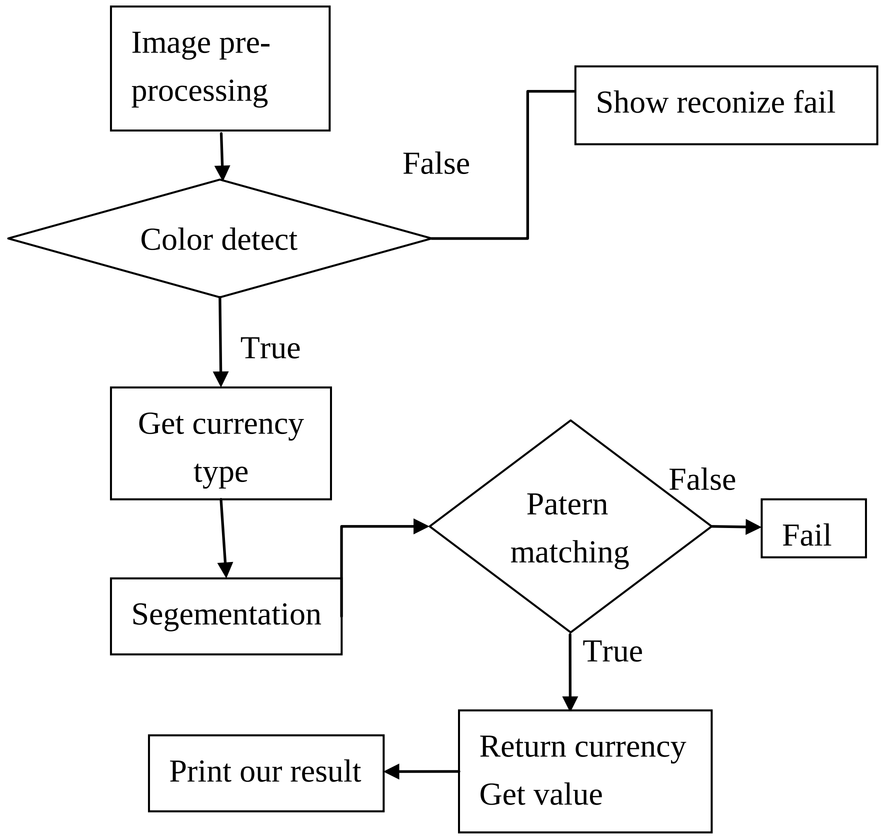


Figure 10, Flowchart of the algorithm

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.

Figure 11, the image being noise pollution

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. Figure 11 shows the image polluted by noise.

When performing smoothing process on a pixel, the neighbour of the pixel is used to do some transforming. After that a new value of the pixel is created. The neighbour of the pixel is consisting with some other pixels and they build up a matrix, the size of the matrix is odd number, the target pixel is located on the middle of the matrix.

Convolution is used to perform image smoothing. Figure 12 is showing the convolution. As the first step, we centre our filter over pixel that will be filtered. The filters coefficients are multiplied by the pixel values beneath and the results are added together. The central pixel value (as figure illustrate is marked with X) is changed to the new calculated value.

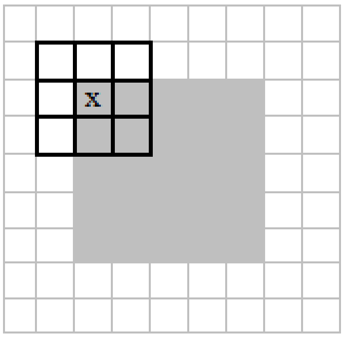


Figure 12, Convolution

As the last step, the filter is moved to the next pixel and the convolution process is repeated. New calculated values are not used in the next pixel filtering. Only old values are involved.

When the filter is centred over the pixel with the border, some parts of it will be outside the edge the image. There are some techniques to handle these situations:

1. Zero padding: all filter values outside the image are set to 0.
2. Wrapping: all filter values outside the image are set to its "reflection" value.
3. Start convolution in second raw and column.
4. The unfiltered rows and columns will be copied to the resulting image.

We use Gaussian operator to blur an image and suppress the noise, it could be seen as a perfect function which is easy to specify. As Equation 2 mentions, we create elements in Gaussian.

The standard deviation σ is a square root of the average of the *n* values squared deviations from its average value, or simply standard deviation of the distribution.

$$\sigma = \sqrt{\sigma^2} \qquad \text{Equation 3}$$

Where the σ^2 is the deviation expressed as average of sums of the average subtracted from each dimensions coordinates. Illustrated as Equation 4.

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 (y_i - \bar{y})^2 \qquad \text{Equation 4}$$

Figure 13, after Gaussian blur

Figure 13 shows the result after perform Gaussian blur.

Figure 14 shows the result of the matrix created by Gaussian filter.

0.57	0.67	0.73	0.75	0.73	0.67	0.57
0.67	0.78	0.86	0.88	0.86	0.78	0.67
0.73	0.86	0.94	0.97	0.94	0.86	0.73
0.75	0.88	0.97	1.00	0.97	0.88	0.75
0.73	0.86	0.94	0.97	0.94	0.86	0.73
0.67	0.78	0.86	0.88	0.86	0.78	0.67
0.57	0.67	0.73	0.75	0.73	0.67	0.57

Figure 14, Value of the matrix that created by the Gaussian filter.

Filter is produced by the code illustrate before.

However there are some other ways to smooth the image, such as median filter. Median filtering is a nonlinear operation. And this perform is often used in image processing to reduce "salt and pepper" noise. Besides the median filter is more effective than convolution when the goal is to simultaneously reduce noise and preserve edges.

Figure 15, Process after median filter

As the Figure 15 shown, 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.

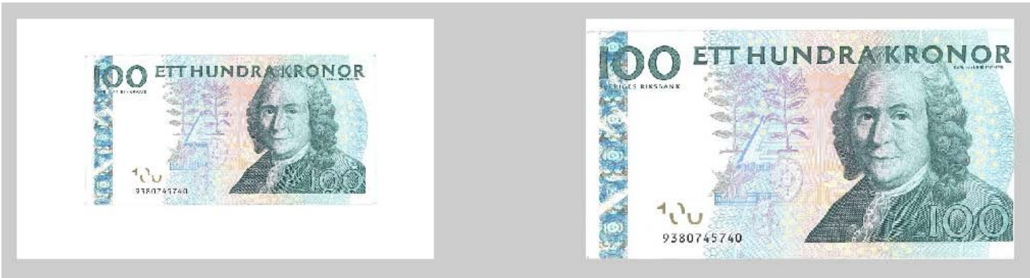
Median filter replaces a pixel via the median pixel of all the neighbourhoods:

$$y[m,n] = median\{x[i,j], (i,j) \in w\} \qquad \text{Equation 5}$$

Where *w* represents a neighbourhood centred around 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 16 shows before cutting and after cutting.



A. Before cutting B. After cutting

Figure 16, before cutting and 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.

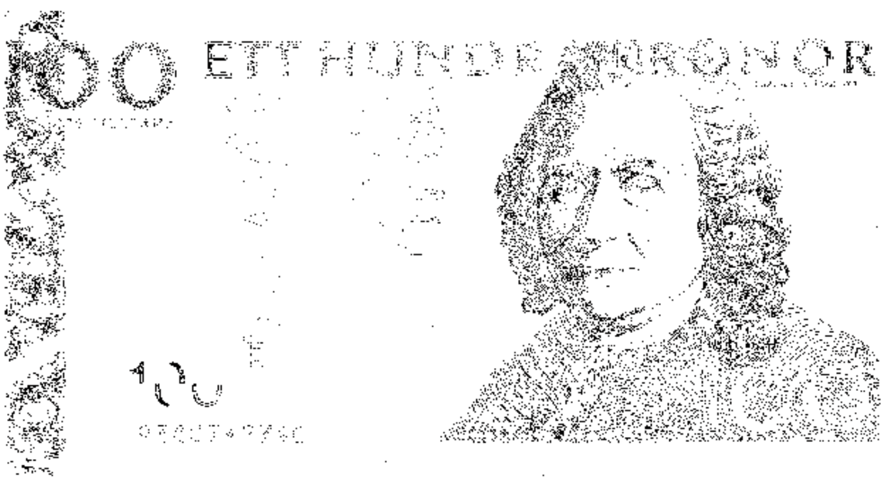


Figure 17, After RGB to Binary translation

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 0.50 and 0.60 for Swedish Kronor. The result is shown as Figure 17.

Here is our algorithm of image binarization:

```
K=find(J>=0.60);
J(K)=1;
K=find(J<=0.5);
J(K)=1;
K=find(J~=1);
J(K)=0;
```

J is the image, from the code we can know to find the pixel which are between 0.5 and 0.6 than set to 0, otherwise set to 1.

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
```

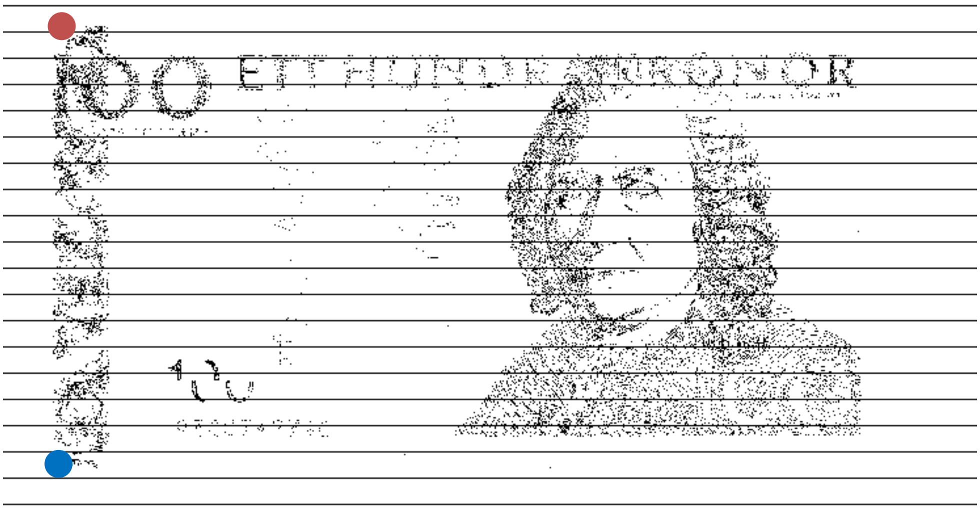


Figure 18, Scan line by x direction

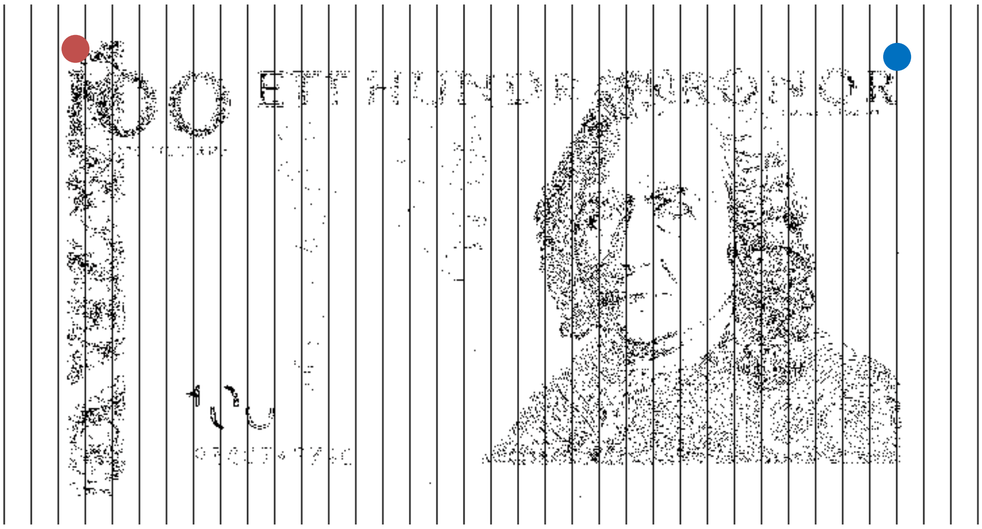


Figure 19, Scan line by y direction

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

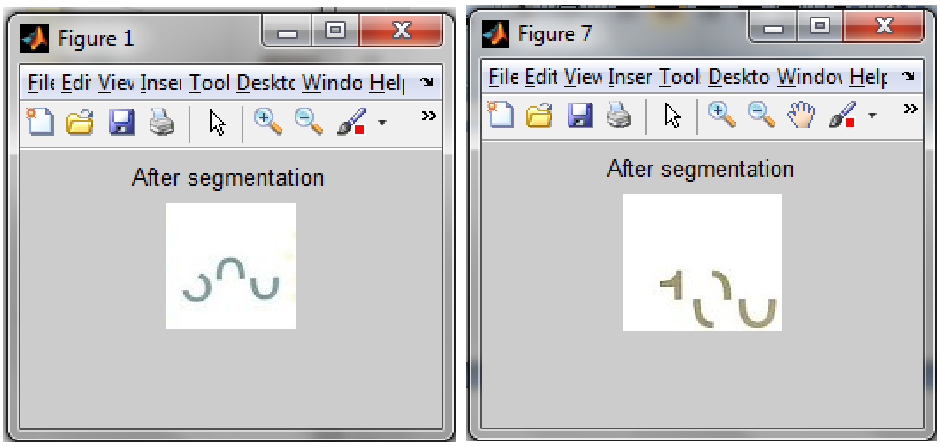
After observation we knew that each currency has one or more unique patterns. So these unique patterns can be used to distinguish different types of currencies.

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
```

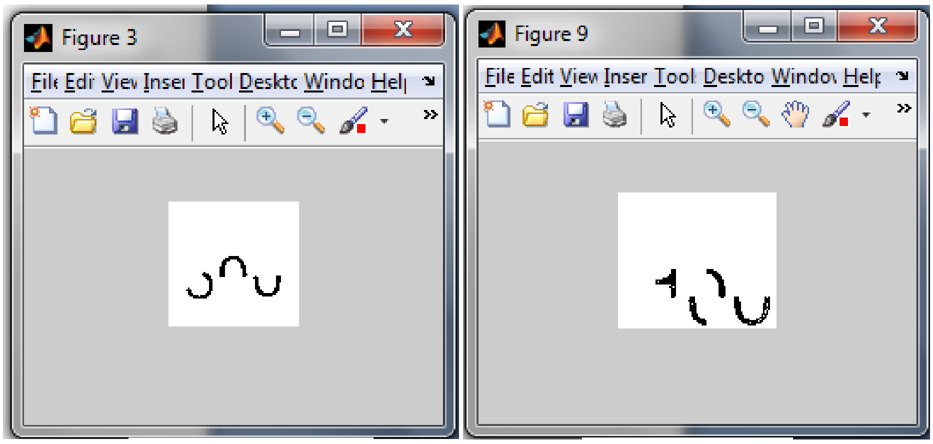
Actually *PY1*, *PY2* is the boundary of top and bottom. *PX1* and *PX2* is the boundary of left and right. After calculation, we get the new boundary of the pattern. That is the way we segment the pattern.



A. Pattern from 500 SEK   B. Pattern from 100 SEK

Figure 20, Pattern from SEK

From Figure 20 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.



A. 500 SEK B. 100 SEK

Figure 21, RGB to Binary image

From Figure 21, we can see that, there is nothing except the pattern.

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 22 shows the RGB histogram and the image.

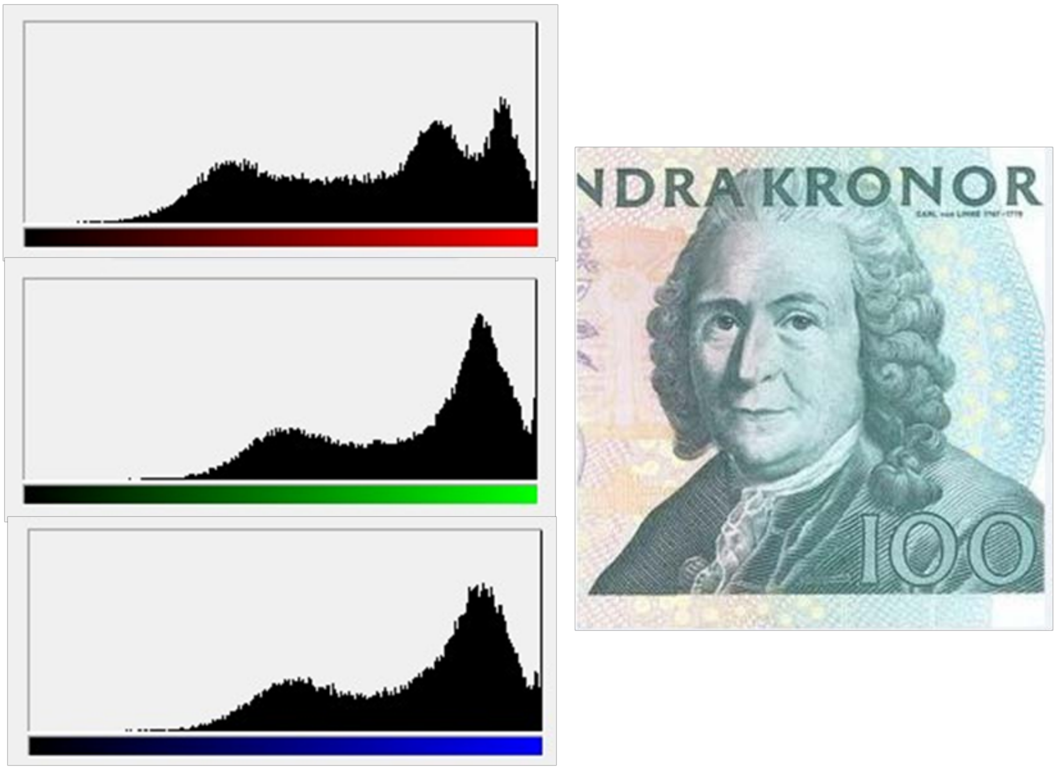


Figure 22, RGB Histogram and the image

2.1.7 Pattern matching and edge detection

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. Figure 23 show the comparison between correlation and convolution.

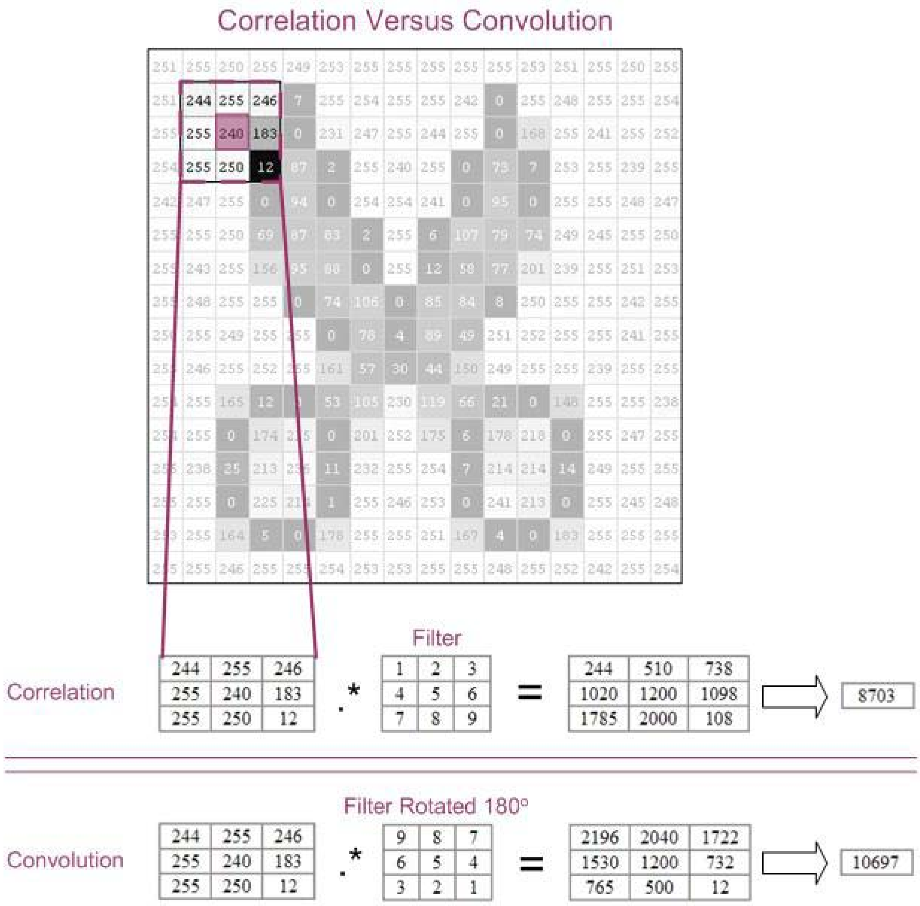
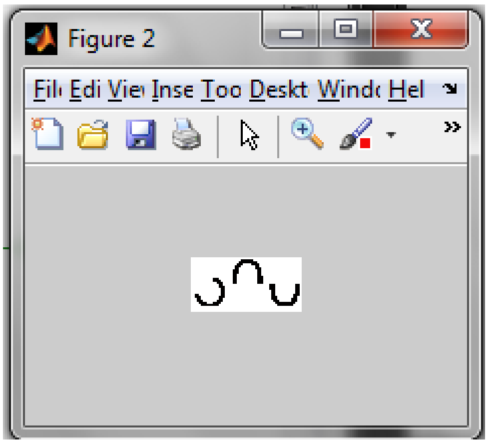


Figure 23, Correlation and Convolution

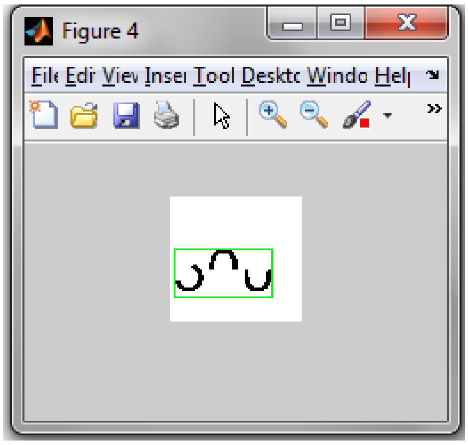
Here are some parts of code for cross-correlation:

```
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
```

The result is shown as Figure 24.



A. Template



B. After correlation perform

Figure 24, Pattern Matching

The implementation closely follows following equation from

$$\gamma(u,v) = \frac{\sum_{x,y}[f(x,y)-\overline{f_{u,p}}][t(x-u,y-v)-\bar{t}]}{\{\sum_{x,y}[f(x,y)-\overline{f_{u,p}}]^2 \sum_{x,y}[t(x-u,y-v)-\bar{t}]^2\}^{0.5}} \qquad \text{Equation 6}$$

Where

- f is the image
- $\bar{t}$ is the mean of the template
- $\bar{f}_{u\,p}$ is the mean of f(x, y) in the region under the template.

In order to inspect the result, edge detection is performed. After performing correlation, we can get the coordinates of the subject in the origin image. We perform segmentation again, and get the image.

The next step is performing edge detection. There are some way to perform edge detection, Sobel, Canny, Prewitt and so on.

Sobel is used for edge detection in our system. It computes an approximation of the gradient in the image. Sobel operator is the operator in image processing, one mainly used for edge detection. It is used for computing the gradient of image brightness function approximation. Any point in the image using this operator will have a corresponding gradient vector or the vector of its law.

There are two Sobel operators, one is used to detect the x direction and the other is used to detect the y direction. Equation 7 is the example of Sobel operator.

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \times a \qquad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \times a \qquad \text{Equation 7}$$

Where a is the image, G_x and G_y are the image that after detected.

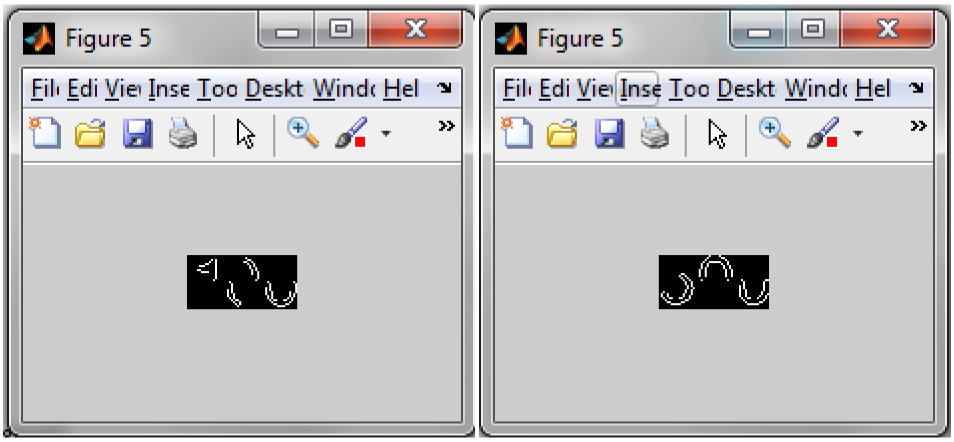
$$G = \sqrt{G_x^2 + G_y^2} \qquad \text{Equation 8}$$

After that, Equation 8 to calculate the gradient direct is used.

But Sobel operator cannot distinguish background and foreground, so we set a threshold and transform the image to binary image before perform the Sobel operator.

The result we get is illustrated as Figure 25.

.



A. 100 SEK B. 500 SEK

Figure 25, Sobel edge detection

The next step is calculation. 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)==1)
           white_points = white_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.

2.2 Implementation

In this part, we will illustrate how the system works.

The system is constituted by eight files:

- Main.m
- Segmentation.m
- PreProcessing.m
- Matching.m
- ColorCheck.m
- LoadCurrency.m
- Data.m
- Template.m

Main.m is the entrance of the system. It calls other functions and prints out the result.

PreProcessing.m is used to do some pre-processing of the image. After calling it, it will return the image which has been process.

After calling the function PreProcessing, ColorCheck is called. ColorCheck is used to check the color and it will return the primary colour of the image, actually it's the value that after calculating the mean of each channel of the image.

And then we get the primary color of the image, we need to know if the color fits for any currencies and what the currency actually is. So call the function Data. This function contains a series dataset of the currency. After that, it returns a value representing one of the currencies and prepares for doing next processing.

The next step is color detection. We probably know what this currency is. But it is not ensured because some other images are primarily colours in the same way. Actually it's a currency or not. But the other one is similar to it. In order to confirm it, pattern matching is performed. But not every currency has the same structure, maybe the pattern is located on the right side, maybe the pattern is located on the right side. It depends what currency it is. It's a different way to segment different currencies. Since we got the probable result, we segment the image based on this result (e.g. if we check that is 500 Swedish Kronor, we use the way which fits for 500 Swedish Kronor to segment the image.), it's a series of judgment statements. When it fits for the condition, the way for segmentation is returned. After the processing, the pattern that we segmented is returned.

When the pattern is segmented, we need to check if it is the one we want to get, so pattern matching is performed. This function needs to pass two arguments, one is the serial number of the currency, and the other is the pattern when the function

Matching is called function template, because we need to get the template. In function Template, it contains a series of templates for different currencies; we get the template based on the serial number. After the template is finished, matching is performed and the final result is to come immediately.

3 Result

For now, the system can check Swedish 100 and 500 kronor and Chinese 100 Yuan. If we add more data to our database that are the currency's primary colour, unique pattern. The system can 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. This is a problem, but we need more time to modify it.

Table 2 and Table 3 show the statistical results of our tests.

Table 2 Banknote classification results

	SEK 100	SEK 500	CNY 100	Total
Quantity	5	5	2	12
Success	5	5	2	12
Error	0	0	0	0
Unknown	0	0	0	0
Accuracy (%)	100	100	100	100

(The entire images get from the scanner)

Table 2 illustrates the accuracy that classification different currency, as example, we test Swedish Kronor and Chinese RMB. As the results shown from Table 2, the correct rate is pretty high. That is because we just take few images for testing, and the entire images are all from the scanner. These means the light conditions are similar, the images are high quality. As we mention before, if we take digital image from camera, this good performing will be changed. That's because it's hard to distinguish color when in different light condition. However it's very hard to realize a 100% correct recognition. There is some delimitation. For instance, the quality of the image, the light condition, the old level of the currency, these are the important characters for recognition.

Table 3 Distinguish between currency and other image

	SEK 100	SEK 500	Other image
Success	5	5	5
Fail	0	0	0

(The entire images get from the scanner)

Table 3 also shown pretty good results. The system distinguishes if the image is a currency. We use 5 other different images, such as the images of cartoon, car or some words. However, this test is check if the system can distinguish the currency or not. As similar as last test, all the images are getting from the scanner. The quality is pretty high.

Through these two 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 thesis 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.

5 Discussion

In our system, the format of the input image we used is JPEG (JPG). There are two types of image format that can be received from the scanner, JPEG and TIF. As is known, TIF contains more details, and high definition, but the restriction is the capacity of TIF image is too large. For JPEG image, the restriction is the details losing, and less details contains, but even for that, those details is enough for our system, and the better thing is the capacity is small. As our comparison, even the same resolution, the capacity of TIF image is several times larger than JPEG image, and in details, the details in JPEG image it's not plain to see any markedly different to TIF image. So we think it more efficient that we use JPEG instead of TIF.

We realize our system by using MATLAB®. That is because MATLAB® is a high –level technical computing language and it has varied API for image processing, the algorithms are terseness and succinctness, speeding up the image processing, making it more efficient. Users won’t feel any time hysteretic, which can satisfy and achieve currency real-time detection.

We test a lot of scanning images, the rate of accuracy can be reached almost 100%, and we are encouraged by this result. And then, we test some images from a digital camera, but this time, the result is not like what we expect. Different illumination condition (under fluorescent light or under incandescent lamp), too dark and too bright, underexposure and overexposure, too many noises, and things surrounding the currency, all of these unexpected situations require us do different extra processing work, like enhancing the contrast, suppressing noises, more accurate segmentation and so on.

We think we are capable to fix those problems and make the system generally used. But we need more time. So, for the present, for scanning image, we are proud of our system.

6 Conclusion

This paper 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.

In the future, we are going to modify our system, overcome some limitation, especially the problem that we get the image from digital camera and complete our data base for recognizing more currencies.

Through this thesis, we know more about image analysis. It's a very meaningful and interesting subject. And here we’d like to thank our supervisor Dr Julia Åhlén. Her enjoyable and helpful meeting with us in past two months gave us a lot of advices and helps. Her earnestness and patience make our work efficient and regular.

.

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] Andrew S. Glassner. *Principles Of Digital Image Synthesis*. Morgan Kaufmann Publishers, 1995.

[3] Burger, W., Burge, M..J. *Digital Image Processing: An Algorithmic Introduction Using Java.* Springer, New York, 2007.

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

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

[6] Milan Sonka, Vaclav Hlavac, Roger Boyle. *Image Processing, Analysis, and Machine Vision Third Edition*. Thomson, 2008

[7] Rafael C. Gonzalez, Richard E. Woods. *Digital Image Processing 2nd edition.* Pearson Education, New York, 2001.

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

[9] Shapiro, L. G. & Stockman, G. C. *Computer Vision*. Prentence Hall, 2001

[10] Shyang-Lih Chang, Li-Shien Chen, Yun-Chung Chung, Sei-Wan Chen, *Automatic license plate recognition*, IEEE Transactions on Intelligent Transportation Systems, Vol.: 5, Issue: 1, (2004) 42-53

[11] Xifan Shi, Weizhong Zhao,and Yonghang Shen, *Automatic License Plate Recognition System Based on Color Image Processing.*Springer Berlin / Heidelberg, 2005

Appendix 1: The functuin we use in MATLAB

```
imread ('file name');

rgb2gray(Image);

size();

find();

round();

figure

imshow(image);

imhist(image);

im2bw();

normxcorr2(A1,A2);

ind2sub(size,IND);

retangle();

edge();

abs();

zeros();

mean2();
```