

AHSANULLAH UNIVERSITY

OF SCIENCE & TECHNOLOGY

Department of Computer Science & Engineering

Course No: CSE 4228

Course Title: Digital Image Processing Lab

Submitted To:

Mr. Sujan Sarker

Lecturer, CSE

[Ahsanullah University of Science & Technology](https://www.facebook.com/pages/Ahsanullah-University-of-Science-Technology/106443059391930?ref=br_rs)

Submitted By :

Name: Dewan Ziaul Karim

Id: 12.01.04.077

Sec: B

Group: B1

Date of Submission: 06.03.16

**1. Introduction**

The main objective of this project is to build a system that can detect & recognize Student ID number from ID card.

Many a time, we face a situation, when our ID cards can get really messy. This can happen for many reason. For example, ID cards can get old, so there may exist some unwanted spot on the id card. ID cards can get scratched or folded due to many reasons. They can even get wet due to rain or by some other reasons. But as we all know, ID card is a very important thing. It serves us in various helpful ways. So it is very much important to recognize ID from an ID card. But when they get messy, it is really tough to extract the exact information , so our aim is to build a system which can detect correct information (correct ID) from an ID card.

**2. Approach model**

-We start our approach by selecting an image from the real world. For example, I used the image my own ID card for this project.

-It should be noted that I used 2 types of images in my project -

1. Original Image

2. Input Image

The original image was taken so that I could compare the input image with respect to that. The input image may be in a rotated position. It may be rotated in different angles. So first I transformed the rotated image into shape same as my original image.

-During this process, both the original image and the input image were transformed into grayscale images.

-After rotating the input image (if needed), I resized the image to a suitable resolution. This is done because input images can be of different sizes, if we continue our experiment with different sizes of image, it will create problems later. So I transformed it into a specific size. So no matter what the size of the input image is, it will always be transformed into the size I declared.

-Then the noise removal process begins . I used median filter for removing noise.

-Then the ID part must be cropped from the whole card (as I need that part only).

-After that thresholding should be performed on the image so that the digits of student ID can be seen well.

There can be some unnecessary dots or small objects in the image which I don't need. So these should be removed.

-Now the image is ready for segmentation. Preprocessing is complete.

-So the image is compared with some images of letters/digits (which I am calling templates). This is done so that the digits from the ID part can be segmented individually.

-After recognizing the digits, small boxes are drawn along individual digits to segment them.

-Finally , these digits are extracted and written in a separate text file accurately (same as in ID card)

-So in a nut shell, the steps of this project are written below -

1. Take original image and Input image

2. Compare to correct the rotation (if the input image is rotated)

3. Resizing the input image

4. Noise removal

5. Crop the ID part

6. Thresholding

7. Removing unnecessary objects

8. Segmentation

9. Digit identification

**3. Tools + Matlab functions used**

1. Images of different sizes and rotations

2. Matlab R2015a

3. Various mat lab built in functions such as -

* imread
* rgb2gray
* imshow
* detectSURFFeatures
* extractFeatures
* matchFeatures
* imrotate
* imresize
* medfilt2
* imcrop
* size
* bwareaopen
* regionprops
* rectangle etc

**4. Implementation**

**4.1. Input Image, Original Image & Rotation**

We take two images -

* Original image
* Input image

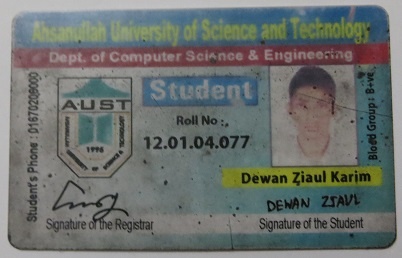


Fig : Original Image



Fig : Input Image (Grayscaled)

The input image may be in a rotated position. So firstly, I rotated the image back to 0 degree. This is done by comparing the rotated image with the original image.

Both the images are transformed into grayscale images at first. This is done using rgb2gray

function. Then from both images , I tried to extract the interest points and region of interest. Then they are matched with each other on the basis of interest point (region). For these, detectSURFFeatures and extractFeatures function were used. Then to match with one another, matchFeatures function was used. One image is mapped on the other to find out the similarity and dissimilarity. Finally with some normal mathematical calculations, I was able to rotate the input image correctly (same as the original image).

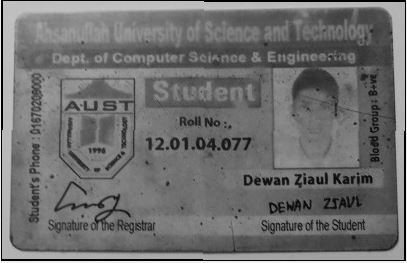


Fig : Corrected Input Image

**4.2 Resizing**

Then the corrected input image was resized using the imresize(A, [NUMROWS NUMCOLS]) function. imresize(A, [NUMROWS NUMCOLS]) resizes the image so that it has the specified number of rows and columns. Either NUMROWS or NUMCOLS may be NaN, in which case imresize computes the number of rows or columns automatically in order to preserve the image aspect ratio.

**4.3 Noise Removal**

The noise was removed from the input image using medfilt2(Resized,[3 3]) function. This indicates that I used median filter. [3 3] indicates 3 by 3 neighborhood. There are many types of filters. Among all of them, median filter is the best as it takes the median value. So there is little chance of getting any unrepresentative or unrealistic value.

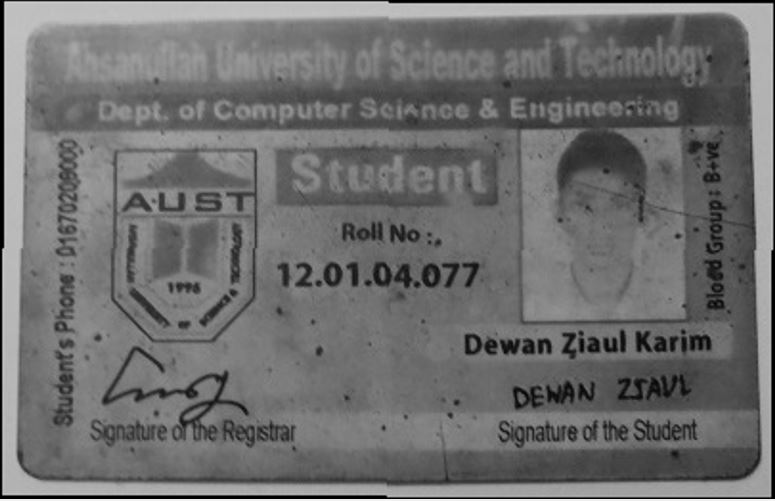


Fig : Resized and Noiseless Image

**4.4 ID part Cropping**

Then the id part was cropped from the input image. It was done using imcrop(NoiseRemoved,[543.5 499.5 438 90]) function. here [543.5 499.5 438 90] indicates the coordinates of the area I want to crop.

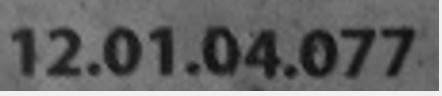


Fig : ID part Cropped

**4.5 Thresholding**

Then thresholding was performed on the cropped ID part. In this part, I compared the image with a value of 70. Wherever the intensities are equal to 70 or greater, I transformed those values into 0, and 255 otherwise. So I made the background black & the digits white. The images I used for individual digits are white in color. That's why I chose to select the colors in this order.



Fig : Thresholding to get the digits clearly

**4.6 Removing unnecessary objects**

After that, some unnecessary small objects were removed from the input image (such as small dots) as I didn't require them. This was done using bwareaopen(G,200) function. This means that it would remove all the objects that are less than 200 pixels.



Fig : Thresholding to get the digits clearly

**4.7 Segmentation**

Then the segmentation part begins. The cropped ID part was compared with some individual images of letters/digits. These were read from a different function. In my main code, I called that function to do that job.

Then I found out the necessary matrix which would help me in segmentation. I counted the rows of that matrix and executed a for loop. Thus I was able to bound all individual digits of my id with rectangles. I decorated them with necessary colors and stuffs.



Fig : Segmentation of digits

**5. Result Analysis**

I faced some problems while doing this project. First of all, whenever the input image is rotated at any angle except 90 or 180 degree (left or right) , it occupies a background which creates problem while rotating the image back to original form. The background cannot be removed easily. So when I wanted to crop the necessary part, the coordinates became messy. The background takes some extra spaces, so the size of the image changes automatically. As a result when I wanted to crop that image, it became problematic for me to crop the ID part accurately.

So the performance will drastically drop whenever a rotated image (other than 90 or 180 degree) is used.

Another problem I faced is that, I couldn't identify the digits after segmentation. The methods I used provided me with incomplete result. For example, I couldn't extract 120104077, rather it gave me '04077' or '12 0 077'.

This model will work best if the rotation problem I mentioned can be solved. The rotation should be done accurately so that the model can work for any type of pictures. Pictures can be rotated at any angle. The model should be made in such a way which can detect ID from the images of any angle or any size. Some tweaking should also be made into digit identification to make that process more efficient.

**6. Discussion**

The performance would have been much better if the rotation problem could have been solved. Another thing which I wasn't able to do is digit identification. That part is a very important part of this model. So problems regarding this sector should also be solved.

The problems with rotation can be solved in the following way -

If somehow, The extra background of the rotated image can be removed, then the image will again be same size as our declared size. That way, when we would crop the ID part, it would not create any problems (as the size would be same as we want).

If we can somehow remove the extra background, it would also allow us to take images of ID card that have extra background from the very beginning of the operation. It should be noted that the images we are working now don't have any extra background at the very beginning. The extra background appears when we take a rotated image as our input image.