DOCUMENT SCANNER BASED ON MATLAB

Chan Chi Weng, db92600@um.edu.mo (DB926002)
Ke Siyun, garfield.ke@connect.um.edu.mo (DB927871)

Abstract---- **This report presents a MATLAB project processing low-quality photographs of documents to get a scanner-style output. The algorithm adjusts leaning pictures with perspective warp and applies different operations like median filter, threshold (Otsu algorithm), and/or Gaussian smoothing, etc. This project comes with a MATLAB application with GUI for demonstration.**
Keywords- scanner, perspective warp, median filter, Gaussian filter, threshold, MATLAB App Design application.

Introduction

There are situations where we need to scan files but cannot find a scanner around, we sometimes have to use a picture taken from a camera or cellphone instead. However, those pictures might have some problems like shadows, low contrast, or leaning position.

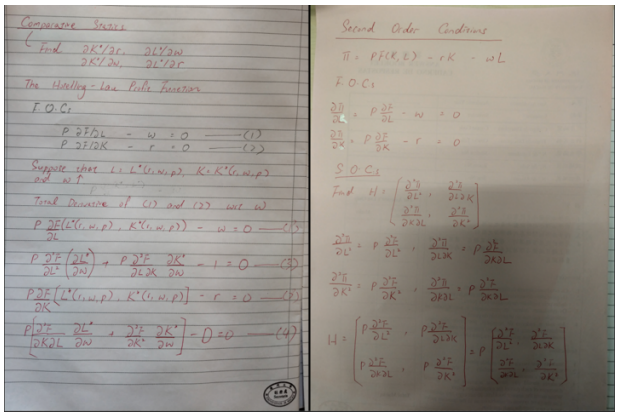


Fig. 1.1: Phone-taken photos with shadows and low contrast

This project aims to resolve these problems and implement a simple camera scanner. We code with MATLAB and generate a MATLAB APP, in which we may import photos taken from a mobile phone or camera and export grayscale xerox images.

Overview of program

A graphic interface will guide users to choose the type of document and import the original image file. Then the user should click the four corners of the document to select the boundary, the program then stretches the image for correction, resize it to the right size, and adjusts the image direction. Then the user can choose different image enhancement functions suiting their needs, including highlighting the image features by binarization (for text files), brightening (for photo images), as well as sharpening. The effect will be displayed in real-time. Once getting a satisfying result, the user could export the xerox style image.

There is also the possibility that applying the algorithm of this project to fields other than a simple scanner app. For instance, it may be used as a pre-processing algorithm of OCR readers.

Technique details

Perspective wrap

The photograph taken from cellphones could be leaned, which cannot be directly cropped and resized into the correct size and ratio. Thus we performed a perspective wrap before doing other modifications.

(i). Determine correct object size

We need to obtain the correct size of a document first. The program has three built-in sizes, i) standard (A4) paper files with a ratio of 1.41:1; ii) ordinary credit card size with a ratio of 8.56:5.5, and iii) passport size with a ratio of 125:88 (ICAO standard). The user will indicate the type he needs upon importing the original image.

(ii). Determine the boundary of document

We used a MATLAB build-in function ginput() to allow users to click points in a figure window to indicate the corner of the document. ginput() returns an array with coordinates of four points. Our algorithm checks the four points for their distances to the top left corner of the whole image, thus sorting the points in-order for later processes to use.

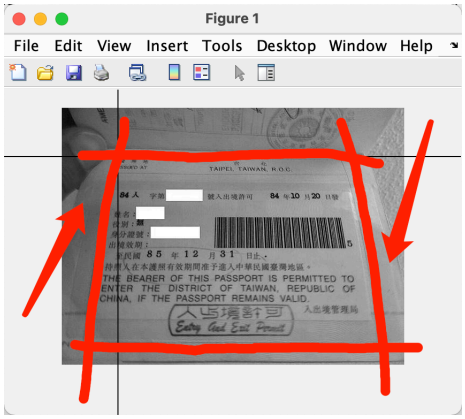


Fig. 2.1: Example of document boundary

(iii). Warp the image

A tform geometric transformation matrix is built based on the size of the document and boundary information obtained. Then we wrap the image with imwarp() using the tform matrix.

The output of imwrap() function keeps all parts of the original image. Thus we need to crop out the useless area. Our solution is to mark the corners of the document in the original image and map them into the warped photo, then crop the latter according to the mapped top-left and lower-right corners.

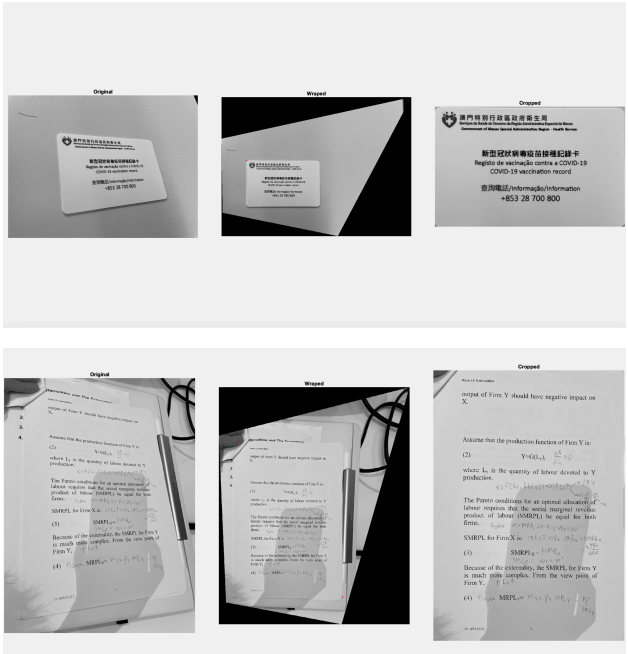


Fig. 2.2. Demonstration of perspective wrap

Image denoising: Median filter

After the angle adjustment(i.e, perspective wrap), apply the median filter to the image at first for noise reduction and better the image enhancement later. A median filter is very effective for denoising of speckle noise and salt and pepper noise, which are noises that may be generated by a camera. At the same time, it also

has the feature of preserving the edges, and will not blur the edges and affect the image effect. It is suitable for use in our scanner. In this system, the filter uses a 3*3 window to take the median value as the output.

Binarization

For the image enhancement function to text files image, it is using binarization to achieve. By changing the background into pure white and the text in black, the image presents an obvious black and white effect, this is in line with the documents file we usually see. In addition, by choosing an appropriate threshold value, binarization can also achieve the effect of shadow elimination (shadows are easy to appear when taking the device perpendicular above to the paper).

(i). Choosing threshold value: Otsu algorithm

We used the Otsu algorithm to decide the threshold value. It was introduced by Japanese scholar Nobuyuki Otsu in 1979. It is said to be the best algorithm for threshold selection in image segmentation. The principle is to divide the image into two parts, the foreground, and the background, according to the grayscale characteristics of the image. Then maximize the variance between the foreground and the background, because the greater the difference between the two parts that make up the image, the less likely it is to misclassify.

For the M×N size of the image, denote the threshold as T, and the proportion of the number of pixels belonging to the foreground to the entire image is denoted as ω_0, and its average gray level is μ_0; similar to background pixels give ω_1 and μ_1. The variance is denoted as g. The number of pixels in the image whose gray value is less than T is denoted as N_0, greater than T is denoted as N_1, then we have:

$$\omega_0 = \frac{N_0}{M \times N}$$

$$\omega_1 = \frac{N_1}{M \times N}$$

$$N_0 + N_1 = M \times N$$

$$\omega_0 + \omega_1 = 1$$

$$g = \omega_0 \times \omega_1 \times (\mu_0 - \mu_1)^2$$

(ii). Thresholding

Simply convert intensity value below the threshold as a_0=0 and above threshold as a_1=255 to achieve the image binarization.

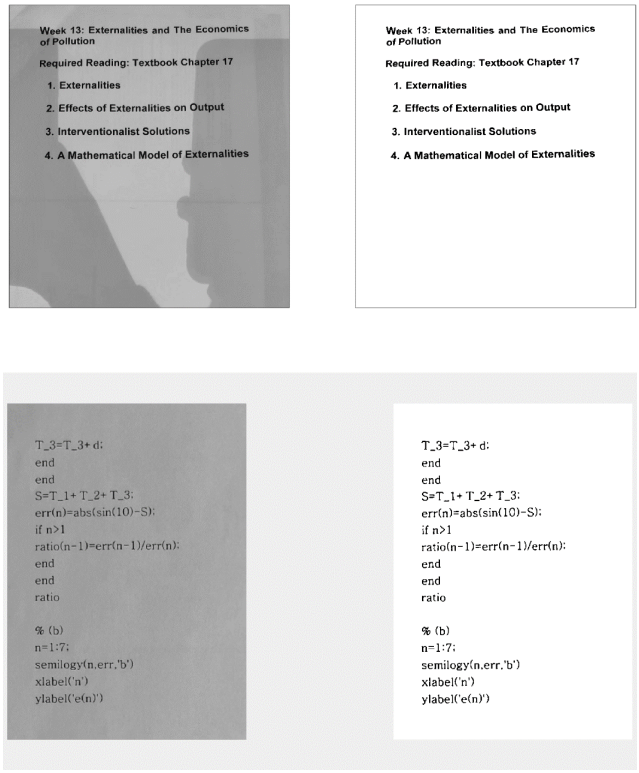


Fig. 2.3. Images before and after the Binarization

(iii). Gaussian filter

After binarization, there may exist some pixel loss in some texts, we decide to use the Gaussian filter to repair it. After testing different sigma values, sigma=0.4 was chosen as the default value in the system. It can have a repair effect without making the overall image become blurred.

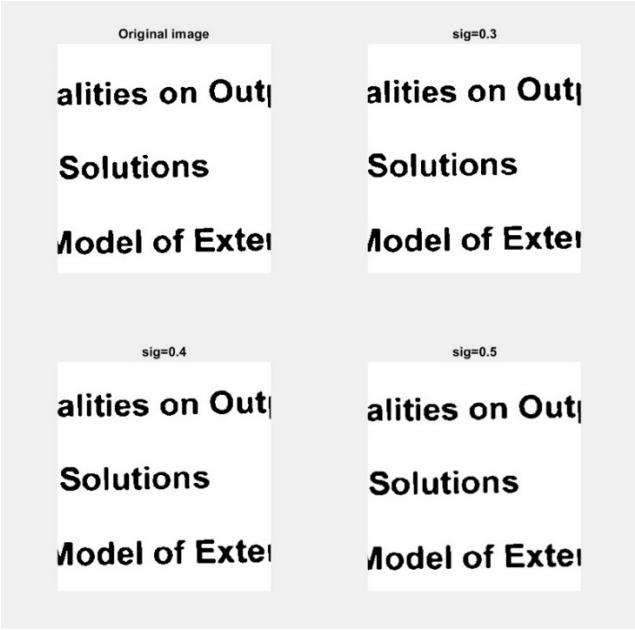


Fig. 2.4. Images processed by Gaussian filter under different sigma value

In addition to document scanning, simple sharpening and brightening functions are also implemented.

Sharpening

For photo scanning, the image sharpening function is used to improve the problem of blurred pictures in the system. It is implemented by a 3*3 Laplace filter and w=1 as the default value.

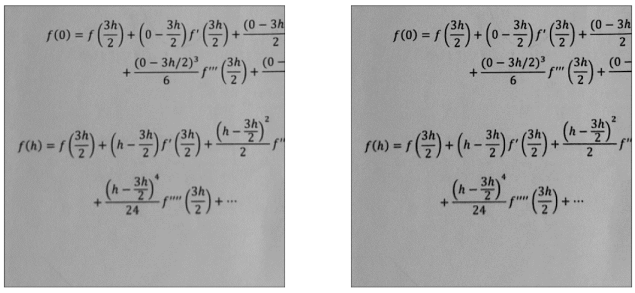


Fig. 2.5. Images before and after processing of Laplace filter

Brightening

The adjustment of the brightness of the image also needs to be considered. As an auxiliary function, whenever the user clicks the ‘Brighten’ button, the intensity value of all pixels of the image is added by 25 to increase the average brightness. This function can be superimposed to meet the needs of the user.

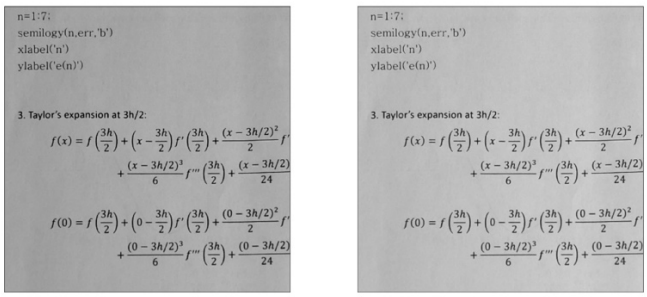


Fig. 2.6. Images before and after brightening

MATLAB GUI

We built a MATLAB APP (*.mlapp) to guide users in performing tests. This app was implemented using MATLAB App Designer.

In case users have only access to an old version of MATLAB, which does not support the current

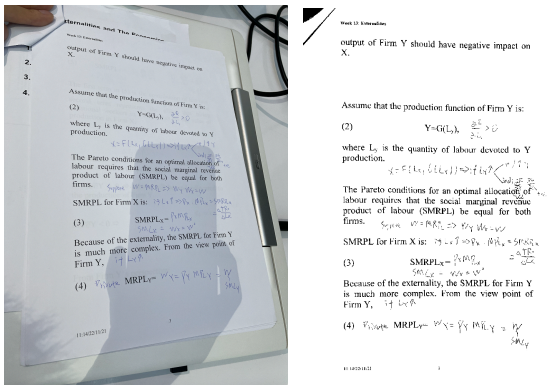
MATLAB APP. Another command-line tool can be found in the deliverable folder.

You may examine the source code of our algorithm, in both *.m files and *.mlapp files. Their ideas are similar; however, some details were modified for better performance in the graphic interface.

Test and demonstration

We take three images taken by phone for demonstration, one in each type of document.

Standard files



Card

Passport

Fig. 3.1. Demonstration of test images

More details and instructions about using the demonstration app can be found in the deliverable folder.

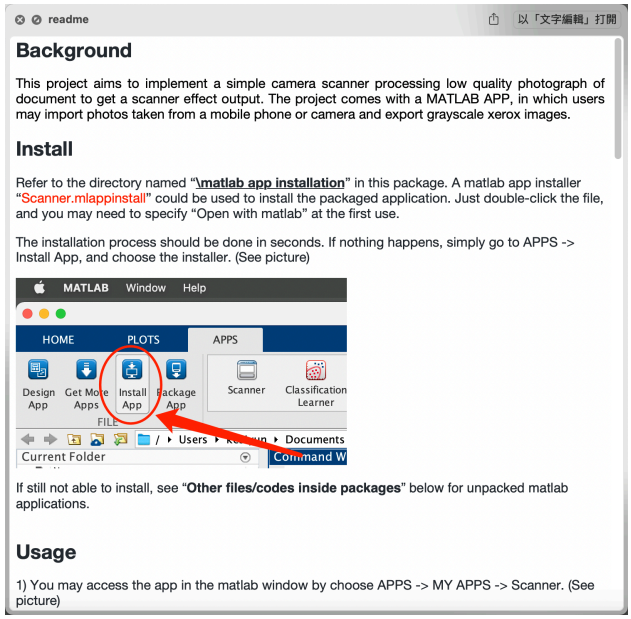


Fig. 3.2. User guidance in deliverable folder

Remarks

This project was completed by Chan Chi Weng and Ke Siyun. Specifically, their contributions to the project are:

Contributions by Chan Chi Weng:

1) Image denoising
2) Binarization
3) Sharpening and Brightening
4) Design of user interface of the MATLAB app

Contributions by Ke Siyun

1) Perspective Wrap
2) Implementing MATLAB app
3) Organizing deliverables(Test images, readme etc.)

References

Nobuyuki Otsu (1979). "A threshold selection method from gray-level histograms". DOI:10.1109/TSMC.1979.4310076.

W. (n.d.). GitHub - wuxiaolang/ WarpPerspective_MarkerArDemo_wu: A demo for Affine transformation and perspective transformation. GitHub. Retrieved December 15, 2021, from https:// github.com/wuxiaolang/ WarpPerspective_MarkerArDemo_wu

Crop image - MATLAB imcrop - MathWorks Australia. (n.d.). Mathworks. Retrieved December 15, 2021, from https://au.mathworks.com/help/images/ref/imcrim.html?requestedDomain=

ginput (MATLAB Functions). (2005). Izmiran. Retrieved December 15, 2021, from http://matlab.izmiran.ru/help/techdoc/ref/ginput.html