VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY
HO CHI MINH UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL & ELECTRONICS ENGINEERING
DEPARTMENT OF TELECOMUNICATION ENGINEERING

PROJECT COURSE REPORT
COURSE: DIGITAL IMAGE PROCESSING - EE3035
TOPIC: MULTIPLE CHOICE SCANNER

LECTURER: Dr. Vo Tuan Kiet

ID	**Fullname**	**Student ID**
1	Lam Thanh Phat	2111974
2	Quang Hoang Trung	1915701

Ho Chi Minh City, 2024

TABLE OF CONTENTS

Contents

List of Figures

1 OVERVIEW

1.1 Introduction to Multiple Choice Scanner

A multiple choice scanner is a device or software application designed to quickly and accurately scan and grade multiple choice answer sheets or forms. It is commonly used in educational settings such as schools and universities to automate the grading process for exams, quizzes, and surveys.

These scanners typically utilize optical mark recognition (OMR) technology, which involves scanning paper forms and detecting marks made by respondents in designated areas corresponding to answer choices. The scanner then interprets these marks and assigns scores accordingly. Multiple choice scanners can significantly reduce the time and effort required for manual grading, especially for large-scale assessments with numerous participants. They help ensure accuracy and consistency in grading, as well as provide detailed statistics and analysis of results.

2 IMAGE PROCESSING RELEVANT THEORY

2.1 Thresholding

Thresholding in image processing is a technique used to segment an image into regions based on intensity values. It involves setting a threshold value, which acts as a boundary separating pixels into two categories: those with intensity values above the threshold (foreground) and those below (background). There are several thresholding techniques in image processing. In this project, we use Adaptive Thresholding based on empirical experiments.

Adaptive thresholding is a method used in image processing to automatically determine local threshold values for different regions of an image. Unlike global thresholding, where a single threshold value is applied to the entire image, adaptive thresholding adjusts the threshold dynamically based on the local characteristics of the image.

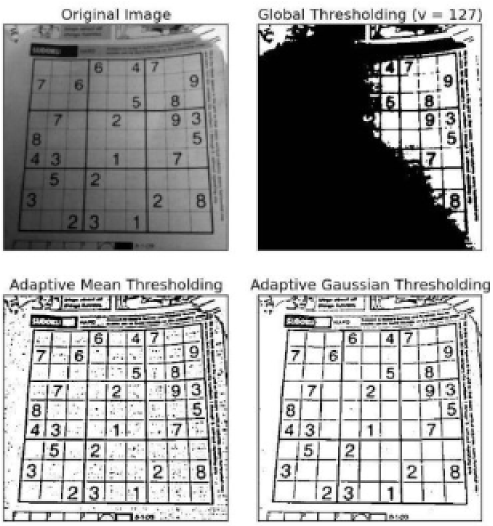


Figure 1: Examples of Adaptive Thresholding

2.2 Contours

In image processing, a contour refers to a curve joining all the continuous points along a boundary that have the same color or intensity. Contours are a fundamental concept used for various tasks such as object detection, shape analysis, and image segmentation. Contours represent the boundary of objects or regions in an image. They outline the shape and structure of the object, so we can use it to extract object based on its detected contours.

2.3 Morphological Operation

Morphological operations are a set of image processing techniques used to analyze and manipulate the structure of objects within an image based on their shapes. These operations are particularly useful for tasks such as noise reduction, feature extraction, image enhancement, and image segmentation. Morphological operations work by modifying the pixels in an image according to the shapes or patterns defined by a structuring element (also known as a kernel). There are some common morphological operations such as Dialation, Erosion, Opening, Closing, etc.

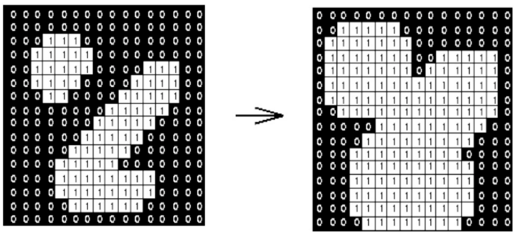


Figure 2: Morphological operation: Dialation

2.4 Image Alignment

Image alignment is a fundamental process in computer vision and image processing that involves aligning multiple images so that they share a common coordinate system or viewpoint. The goal of image alignment is to ensure that corresponding features in different images are spatially aligned, enabling further analysis or processing. In this project, we explore feature-based image alignment technique based on Homography transformation.

Homography, also referred to as planar homography, is a transformation that is occurring between two planes. In other words, it is a mapping between two planar projections of an image. It is represented by a 3x3 transformation matrix in a homogenous coordinates space. The planar homography relates the transformation between two points in two plains can be described as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

To compute a homography, you need corresponding points in both images. These points are typically manually selected or detected automatically using feature matching techniques such as SIFT (Scale-Invariant Feature Transform) or ORB (Oriented FAST and Rotated BRIEF). Once the corresponding points are identified, algorithms like Direct Linear Transformation (DLT) or RANSAC (Random Sample Consensus) can be used to estimate the homography matrix **H** that best maps the points. Finally, the homography matrix **H** is computed, it can be used to warp one image so that it aligns with the other image.

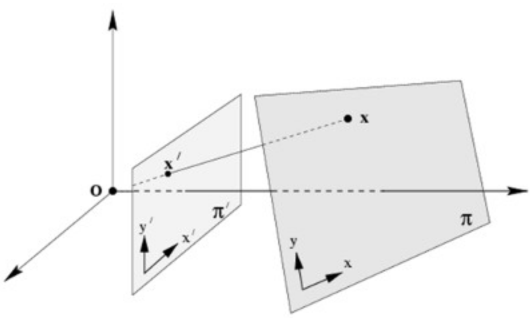


Figure 3: A transformation between two planes

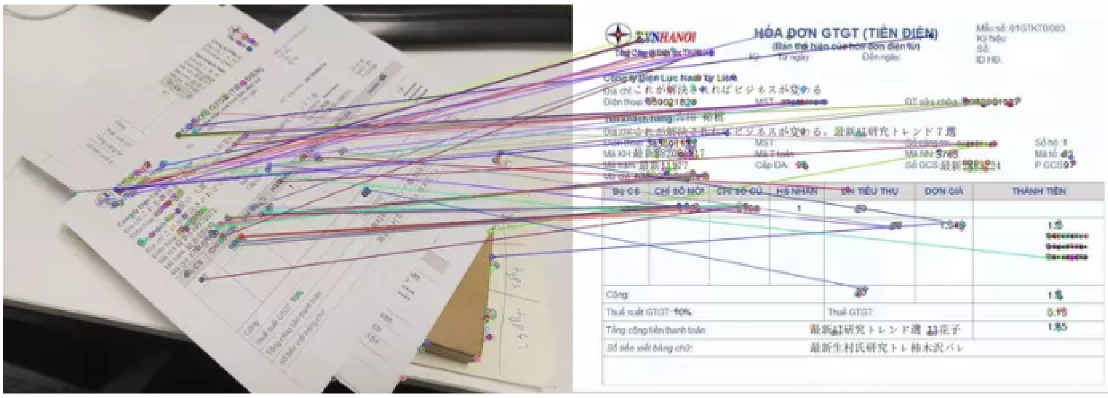


Figure 4: Image alignment based on Homography transformation

3 THE PROPOSED SYSTEM

In this project, we build a simple version of a Multiple Choice Scanner system mainly based on basic image processing techniques, combined with basic image classification techniques.

3.1 System Requirement

Here are some technical requirements for our system.

- Input image: Taken from smartphone.
- Accuracy: $\geq 90\%$.
- Response time for each image: $\leq 2s$.
- Capable of processing input images taken from various angles.
- Graphical User Interface (GUI) for interacting.

3.2 System Overview

The overall proposed system can be described shortly in the below diagram.

Figure 5: Proposed system diagram

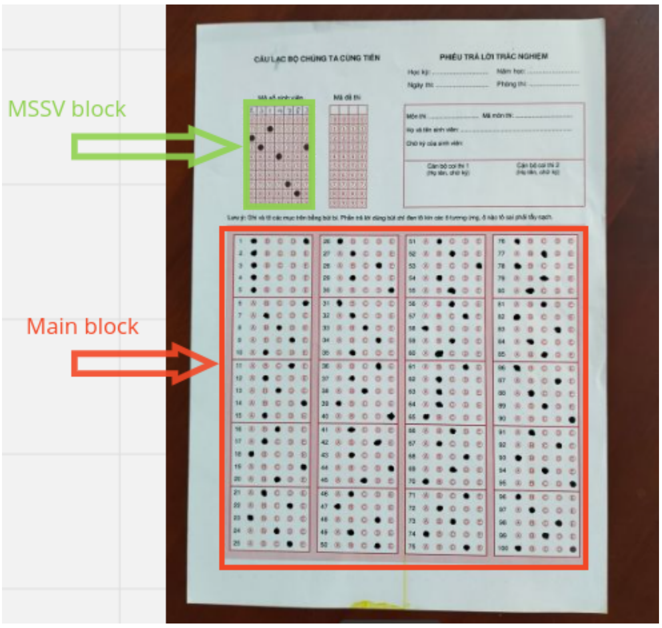


Figure 6: Input image

For simplicity, our system aims to extract two main information: Exam score (Score in Fig.5) and Student ID (MSSV in Fig.5). The system pipeline is indicated at the left corner of Fig.5, while the right corner shows some sub-blocks of intermediate blocks in main pipeline such as Preprocessing, Process main block, Process MSSV block.

Firstly, the exam paper is extracted from the original input image. We then split the exam paper into two separated parts: The main block containing all the bubble choice answers, the MSSV block containing bubble choices to identify student ID. Consequently, each block is independently processed to extract the score and the student ID, respectively. Our system will be discussed more detailed in the next subsections.

3.3 Component Details

For convenience, we split our main system pipeline into three components: Preliminary component, Main block component and MSSV Block component. Each component is explored in the next parts.

3.3.1 Preliminary component

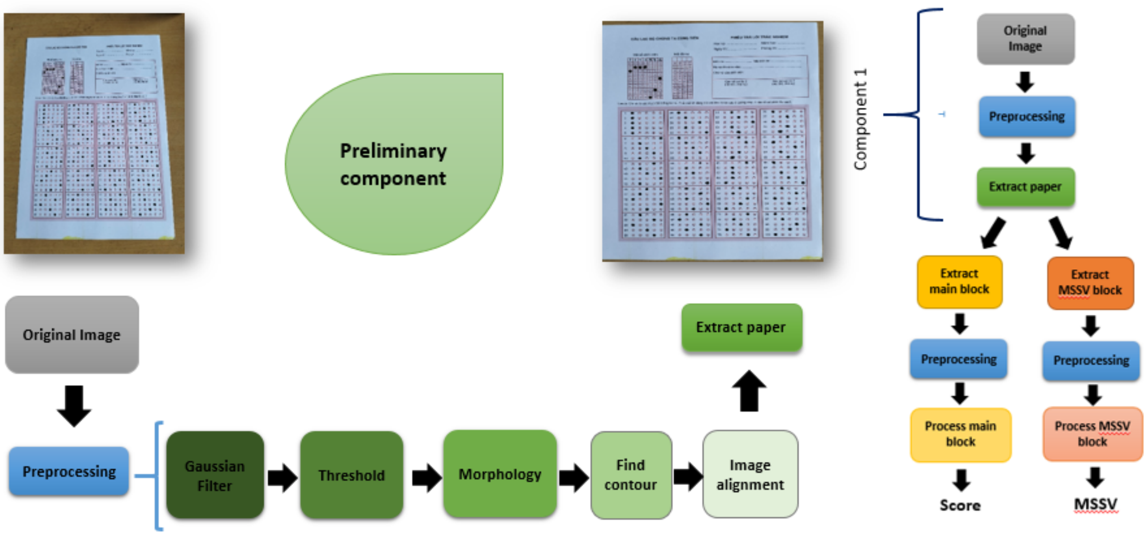


Figure 7: Premiliary component

The preliminary component aims to extract the paper exam from the background in the original input image. Notably, the input image is not necessarily taken from frontal perspective but from different perspectives. Additionally, the input image can include various type of noises. Hence, we need some intermediate steps in order to enhance the quality of the input image, which are presented in the **Preprocessing** block.

Firstly, Gaussian filter is used to reduce noises, especially Gaussian noise, and smoothen pixels to create a more uniform appearance for the image. Next, we apply Adaptive Thresholding to binarize the processed image into foreground and background regions. Following that, Morphological Operation (e.g. Dialation) is also utilized to highlight the object lines, especially boundary lines, for the purpose of better contour detection of the paper exam.

After detecting the paper exam through choosing the contour with the largest area, we apply image alignment technique to convert the current image perspective into the frontal one. From the current paper exam's contour, we extract the coordinate location of four corners of the contour $C_1(x_1,y_1), C_2(x_2,y_2), C_3(x_3,y_3), C_4(x_4,y_4)$, which represents the top left, top right, bottom left, bottom right corner, respectively. The four corresponding points of these corner of the aligned paper exam image is determined as $C'_1(0,0), C'_2(M,0), C'_3(0,N), C'_4(M,N)$, where M,N are the shape of the current paper exam image. Given the mentioned mapping, the homography matrix $\mathbf{H}$ is calculated for all N points of the current paper exam image through OpenCV library that satisfy:

$$C'_i = \mathbf{H}C_i \text{ for } i = 0,1,2,...N$$

The exam paper is now aligned to the frontial perspective. We then split the exam paper into two images that contain the Main block (refered as Candidate-Main block image) and the MSSV block (refered as Candidate-MSSV block image), which are described below. Consequently, the two above images are the inputs of two independent components: Main block component and MSSV Block component, respectively.

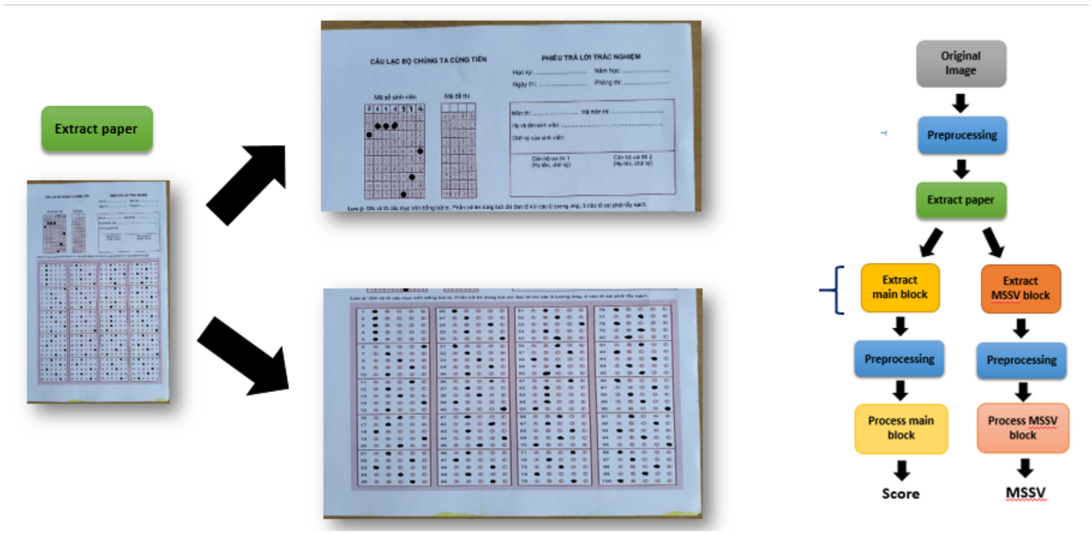


Figure 8: Extract paper exam into two blocks

3.3.2 Main block component

The main block component receives the Candidate-Main block image as input. Similarly to the premilinary component, the **Preprocessing** block here is also used to extract and align the main block. The main block is then resized to the standard pre-defined shape for the convenience of post-processing. The extracted main block includes four answer blocks. Leveraging contour-based boundary detection again, we extract the four largest contours of the main block to get the answer block. Then, answer lines and bubble choices of each answer block are also gathered, respectively.

At the next step, the CNN model is used to classify whether one bubble belongs to one of two classes: **choice** or **unchoice**. Then we can determine the answer of each question based on the location of the choice-classified bubble. By this way, all the answers are completely identified. Finally, the extracted answers are compared with the pre-defined true answer to get the exam score.

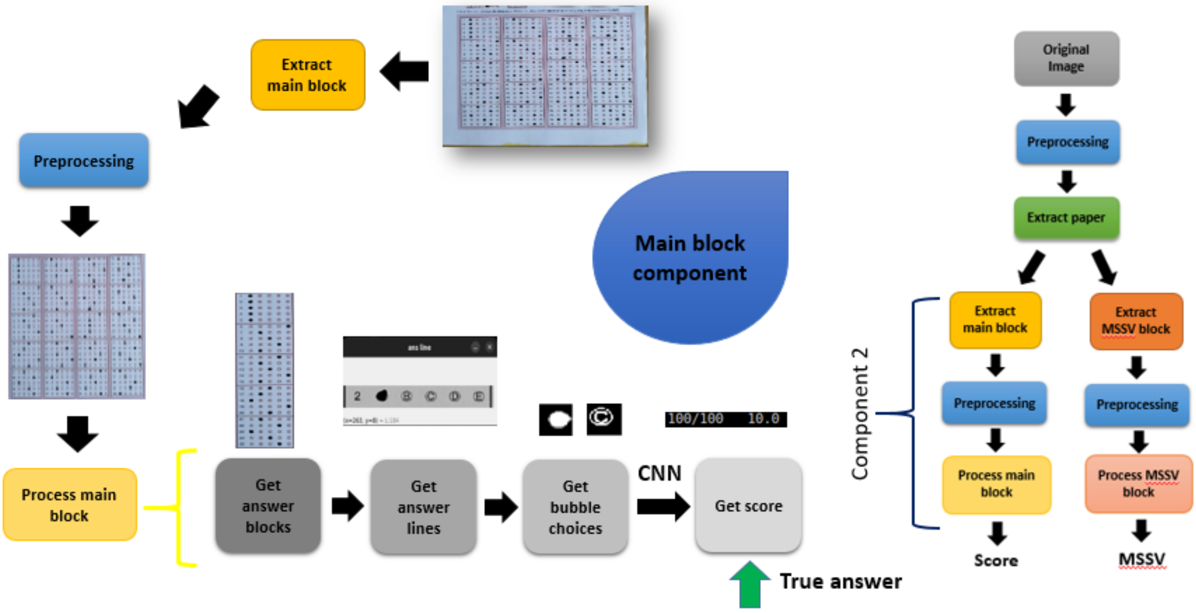


Figure 9: Main block component

3.3.3 MSSV block component

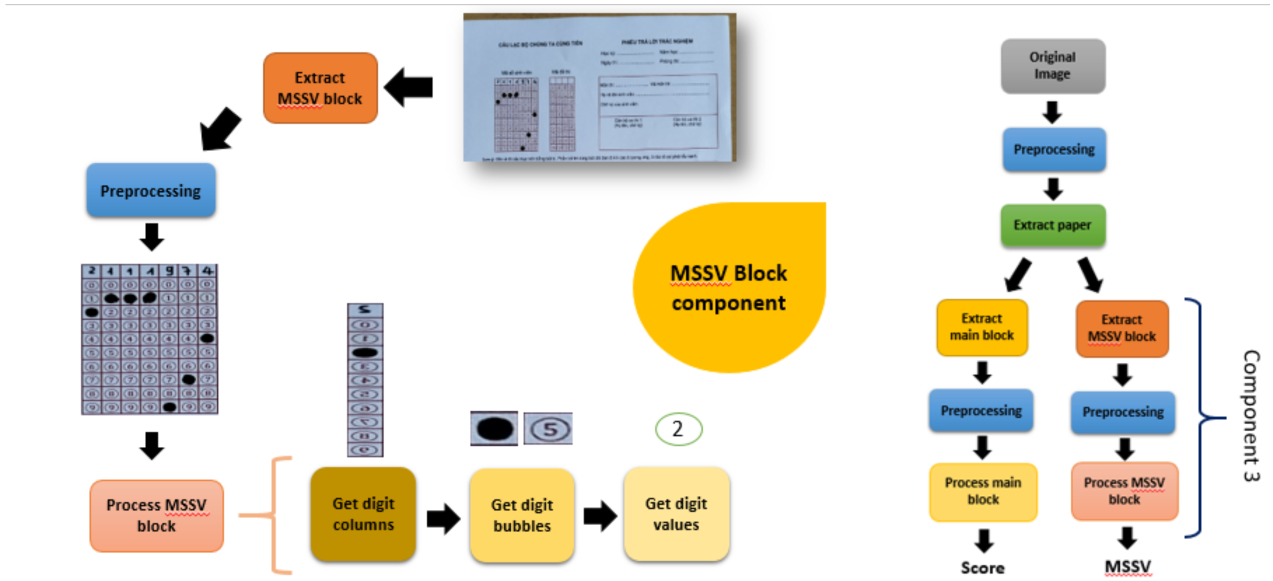


Figure 10: MSSV block component

The MSSV block component receives the Candidate-MSSV block image as input. The MSSV block is extracted and align as the result of the **Preprocessing** block. The MSSV block contains seven columns corresponding to seven digits of student ID. We extract these columns and then extract bubbles based on basic image slicing. Once again, the **choice** or **unchoice** status of one bubble is determined with the support of the CNN model mentioned at above component. From the choice-classified bubbles and their location in columns, the digit value of each column can be easily inferred.

4 RESULT ILLUSTRATION AND DEMO

4.1 Result illustration

Some figures below illustrate the results of several processes within our system.

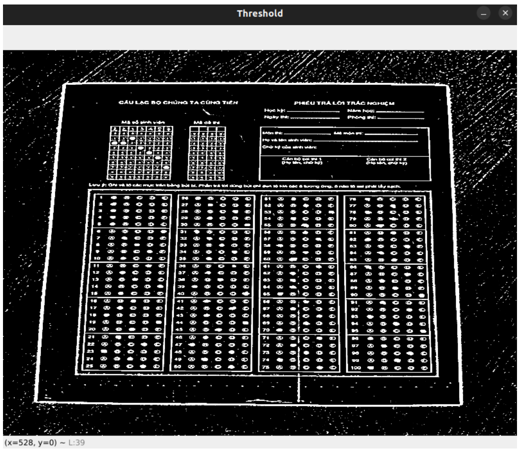
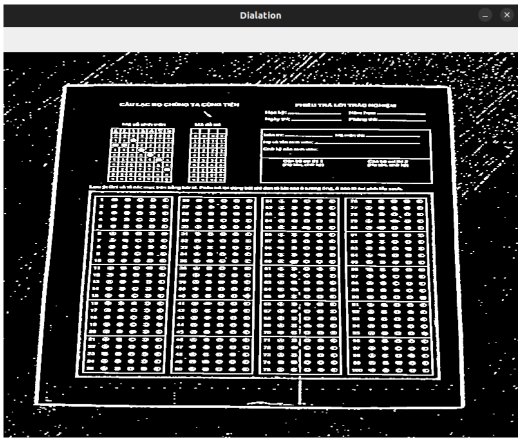



Figure 11: Preprocessing: (left)- Thresholding; (right)-Dialation

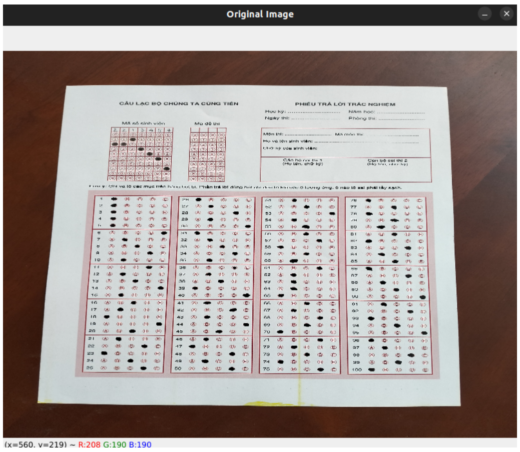
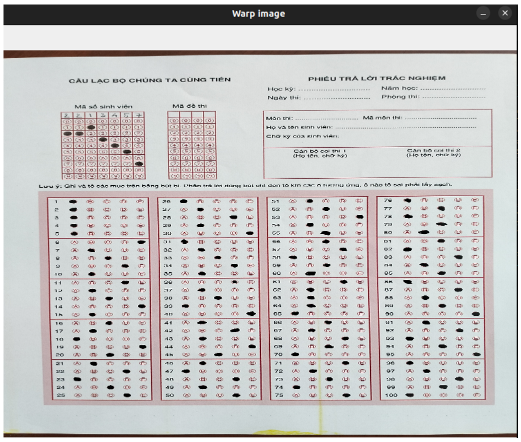



Figure 12: Extract paper exam: (left)-Original image; (right)-Aligned image

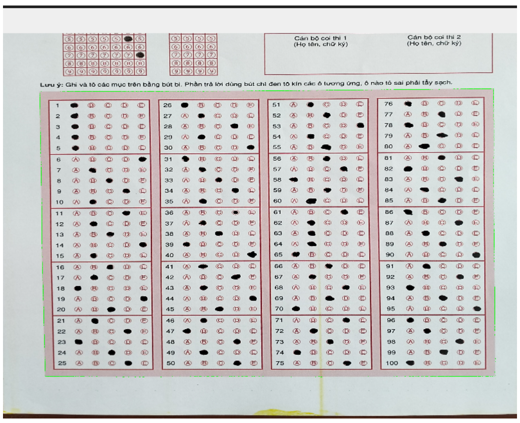
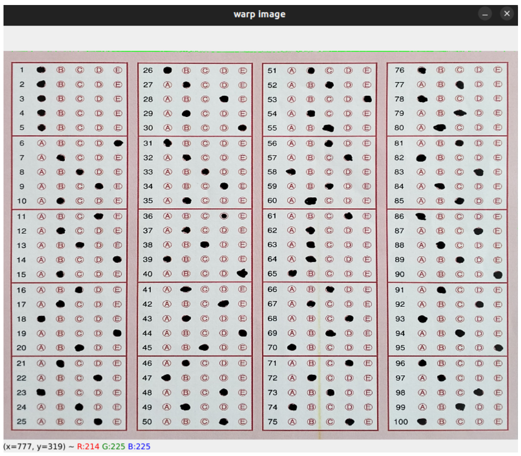



Figure 13: Extract main block:(left)-Original image; (right)-Aligned image

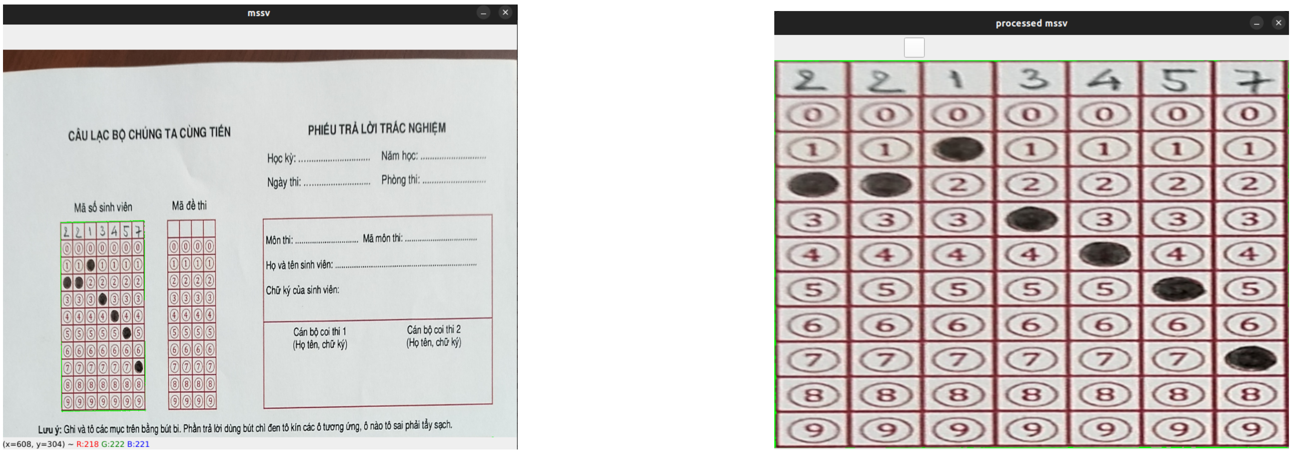


Figure 14: Extract MSSV block:(left)-Original image; (right)-Aligned image

4.2 DEMO

For the purpose of conveniently interacting with the system, we build a simple GUI using Pyqt5 tool. The image below shows a simple UI of our system.

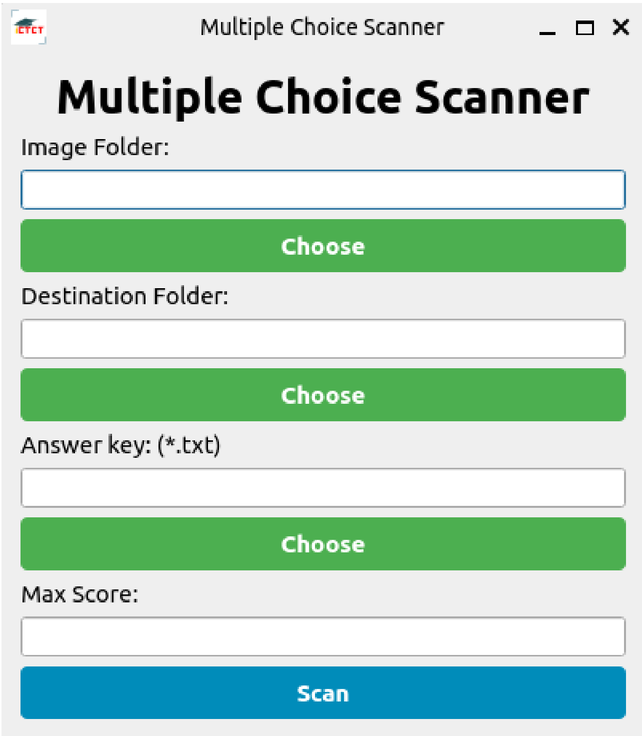
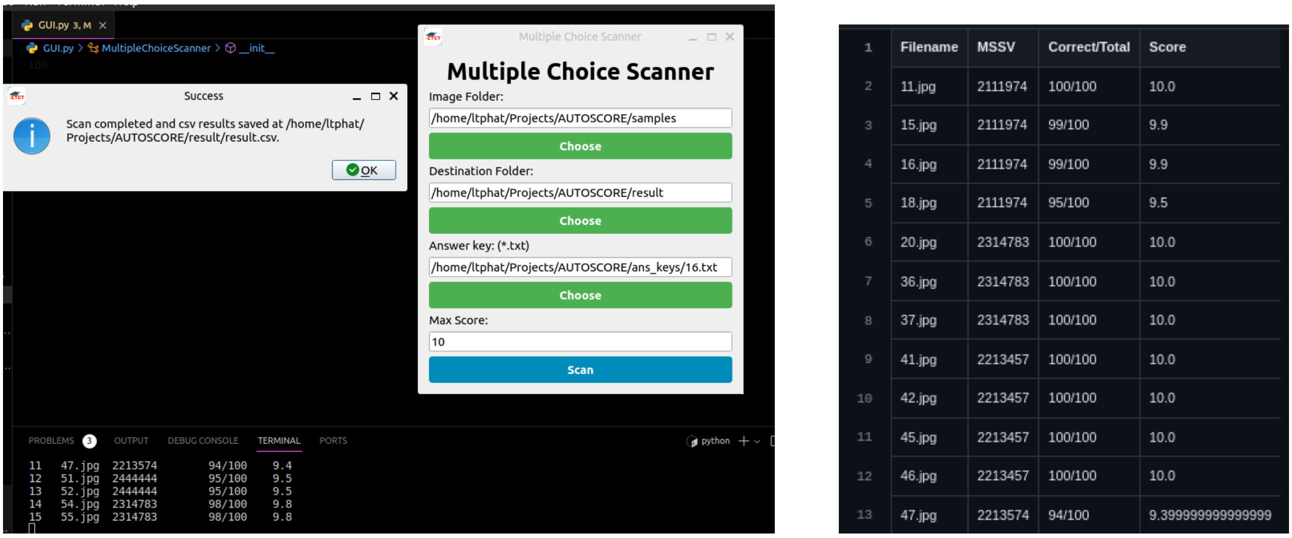


Figure 15: GUI

The system requires four inputs:

- **Image folder:** The directory containing all the input images.
- **Destination folder:** The directory where the scanning result is stored.
- **Answer key:** The directory where the answer key for the exam (*txt) is placed.
- **Max score:** The maximum score of the test.

After filling in all fields above, we click the "Scan" button, the result is described below. The scanning result is store as *.csv file at the destionation folder.



1	Filename	MSSV	Correct/Total	Score
2	11.jpg	2111974	100/100	10.0
3	15.jpg	2111974	99/100	9.9
4	16.jpg	2111974	99/100	9.9
5	18.jpg	2111974	95/100	9.5
6	20.jpg	2314783	100/100	10.0
7	36.jpg	2314783	100/100	10.0
8	37.jpg	2314783	100/100	10.0
9	41.jpg	2213457	100/100	10.0
10	42.jpg	2213457	100/100	10.0
11	45.jpg	2213457	100/100	10.0
12	46.jpg	2213457	100/100	10.0
13	47.jpg	2213574	94/100	9.399999999999999

Figure 16: Results

5 SYSTEM EVALUATION

By conduction extensive experiments on different samples, our system achieve the technical requirement mentioned at section 3. Through our experiment, the most common cases that our system failed to process are attributed to some below reasons:

- **Low resolution image:** Low-resolution images have fewer pixels, which means less detail is available for analysis. Hence, our system is recommended to process input images taken from smartphone, which have high resolution.
- **Images taken from an angle that was too inclined:** Although image alignment techniques are equipped, the efficiency of the aligned images can be affected when the input image's perspective are very far from the standard images.
- **Images with low brightness:** These kinds of images make the preprocesing block difficulty to reduce noises, detect contours, etc, which result in failure of extracting necessary information.

6 REFERENCES

References

[1] OpenCV, "Image Thresholding", "https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html"

[2] OpenCV, "Basic concepts of the homography explained with code", https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html

[3] Adrian Rosebrock, "Image alignment and registration with OpenCV", https://pyimagesearch.com/2020/08/31/image-alignment-and-registration-with-opencv/

[4] Adrian Rosebrock, "Adaptive Thresholding with OpenCV (cv2.adaptiveThreshold)" https://pyimagesearch.com/2021/05/12/adaptive-thresholding-with-opencv-cv2-adaptivethreshold/,

[5] OpenCV, "Contour Feature" https://docs.opencv.org/4.x/dd/d49/tutorial_py_contour_features.html

[6] PyQt5 Tutorial, https://www.tutorialspoint.com/pyqt5/index.htm