國 立 清 華 大 學

碩 士 論 文

自動光學檢測系統的演算法改良

An Algorithm improve on AOI System

系所別：資訊系統與應用研究所碩士班

學號姓名：105065527 呂昊叡 (Hao-Jui Lu)

指導教授：韓永楷 博士 (Prof. Wing-Kai Hon)

中 華 民 國 107 年 7 月

誌謝

感謝…

Acknowledgements

I’m glad to thank…

摘要

在工業檢測的場合中，自動化影像辨識 (AOI) 技術日漸成為一個重要的應用，而現成的影像處理軟體通常有授權費過高，且準確度與分析速度並不符合產線的預算與需求，對此問題，我們希望可以找到一個低成本的解決方案，同時滿足產線對於分析速度的需求。本研究所實作出的產品檢驗流程可以粗略分為兩步驟，第一步驟為樣板設定，此步驟會紀錄標準的產品特徵；第二步驟為樣本檢驗，此步驟會將樣本與第一步驟所記錄下的樣本進行比對，並判斷此背光鍵盤是否有瑕疵或故障。本論文主要討論自動化光學檢測系統及分析演算法的設計架構與分析過程中的演算法比較並加以改良。並在最後將嘗試過的各種方法以產線實際運作的標準下進行比較。

Abstract

In the industrial production site, Automatic-Optics-Inspection (AOI) techniques has become an importent application. Since the existing image processing software is not cost effective due to high licence fee, and doesn't meet the requirment of both speed and accuracy.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In production line, the quality check process is the final and the most important step before the products roll out. Having automatic inspection platform on the production line could speed up the production speed and reduce the chance that product with defect getting on the shelf. Witch is important for all manufacturers. In this thesis, we will introduce an AOI platform based on OpenCV 3.3.1 & EmguCV 2.4 special designed for self illuminated keyboard. And give a pros & cons analysis for our case.

1.1 Motivation

Automatic Optical Inspection (AOI) technique is now widely use in modern production lines. With these techniques applied, cost spent on quality check & chance of making mistake has significantly dropped. But the ready-made AOI solutions still cost much on license fee, take a lot of resource to running on production line. There is a demand to make an AOI platform special designed for our own need. Since the previous solution is obsoleted. It doesn't performs well on lettering defect detection and couldn't identify the color of LED back-light of the keyboard. In this thesis, we will introduce the construction and implementation of an AOI(Automatic Optics Inspection) system that is special designed to do the LED self illuminated keyboards.

1.2 Goal

Design an AOI(Automatic Optics Inspection) system is have ability to detect the *lettering defect* and *LED multifunction* from *given image* or from *camera*. Need to be *accurate*, *scalable* and both *time & cost effective* in order to meet the requirement of production line.

1.3 Organization

The organization of the paper is as follows. In Chapter 2, we introduce some background knowledge related works, previous solution & our testing platform structure. In Chapter 3, we will give a detailed description of our testing work-flows and inspection algorithm design. In Chapter 4, we will further introduce the difference on performance of each key points detection methods and their pros & cons. Finally, we conclude this thesis in Chapter 5.

1.3.1 Main Contribution of This Dissertation

Lower Cost

In this thesis, the implementation is based on OpenCV 3.3.1, and doesn't have any dependency with license fee required tools such as LabVIEW or Matlab. This brings huge benefit when the AOI system is deployed to the production line.

Accuracy

The accuracy issue of defect detection method in previous solution is now solved by improved implementation based on Puteras work [1] (2010).

Speed up

After solved the accuracy issue, we find out the bottle neck of the Inspection work flow. Thus, by reduce the input area size, only focus on the area that worth time to do

the inspection. Also use the new alignment method, witch will be discussed in Ch.3, instead of SURF to do the alignment speed up the process significantly.

Provide a updated baseline platform for later upgrade

This project provided a base model for future AOI tools development.

Chapter 2

Preliminaries

In this section, common techniques of object detection are mentioned. Preprocessing (thresholding, color model transformation,blob detection) and recognition (feature point detection, and matching) are included.

2.1 Common Image Processing Techniques

2.1.1 Color Models

In this thesis, two color model were utilized to do the lettering defect detection and LED color & function inspection. There is many different ways to describe colors in image processing.

RGB Color Model

The color model that most people known was 8 bits RBG color model. Which use 8 bits for each channel, could describe 256 X 256 X 256 colors. The idea of RGB color space is to separate colors to three primary color that human eyes could sense from light reflected from an object. By summing up three different color channel, we could simulate different colors in nature. Common used in color LED control and describing the color of a pixel. This color model is also the model used to describe raw image captured from camera we used.

Gray-scale

The color model we used to do lettering defect detection was 8 bit gray-scale color model, this color model could help us in production line to simply parameter setups. Also, gray-scale color model performs better then RGB to HSV color model when doing the lettering defect detecting. The conversion method from RGB color model to gray-scale color model built in OpenCV is described as below.

$$Y = 0.299 \times R + 0.587 \times G + 0.114 \times B$$

Here is an example conversion result

add example here

HSV Color Model

The other color model we used was HSV color model, an alternative of RGB color model, this color model describe colors in different way of RGB color model. HSV color model use Hue, Saturation and Value to describe different colors. This color space could describe different color in a more intuitive way. Hue stands for the color tone of a pixel. Saturation stands for how dense the color is. The more pure pixel color is, the higher value saturation will have. Value is used to describe how bright the pixel is. The brighter, the higher. The conversion method from RGB color model to HSV color model built in OpenCV is described as below.

$$V = \max(R, G, B)$$

$$S = \begin{cases} (V - min(R, G, B))/V, & \text{if } V \neq 0 \\ 0, otherwise \end{cases}$$

$$H = \begin{cases} 60(G-B)/(V-\min(R,G,B)), & \text{if } V = R \\ 120 + 60(B-R)/(V-\min(R,G,B)), & \text{if } V = G \\ 240 + 60(R-G)/(V-\min(R,G,B)), & \text{if } V = B \end{cases}$$

If $H < 0$, then $H = H + 360$. On output $0 \leq V \leq 1, 0 \leq S \leq 1, 0 \leq H \leq 360$

add HSV filtered example here

2.1.2 Thresholding & Binarization

Thresholding with different color model can help us select interested pixels in an efficient way. In this thesis, we used this idea to identify where the keyboard buttons are located by doing thresholding to the gray scale image. By set a threshold value between 0 & 255, we can get a binarized image. After apply blob or contour detection methods to that binarized image. When thresholding with HSV upper bounds & lower bounds, we can identify pixels with certain color or brightness. By HSV thresholding, we can identify keys with different color lightening. Witch is efficient for us to check if the LED back-light is malfunction.

add example here

2.1.3 Blob Detection

A blob is a group of connected pixels in an image that share some common property (e.g. gray-scale value, HSV value, RGB value, shapes, area, etc). The goal of blob detection method is to identify and mark these pixels in the given image. In this thesis, after we applied thresholding technique and defect detecting methods to the keyboard image

captured from camera and get the corresponding binary image. We apply blob detection to that binary image then get the key position, LED color and defect information from it.

add example here

2.1.4 Feature Point Detection

Feature point detection technique is widely used in object detection or tracking applications. Popular methods like SIFT feature points having the current state of the art accuracy, but its slow on execution. SURF is an improve based on SIFT feature point. SURF is much faster then SIFT without sacrificing much robustness. But still takes some time when dealing with large sized image. However, by restricting the area of interested with correct parameter setting, SURF feature point could be utilized in our project without issue. Other feature point detecting methods like FAST, ORB and FREAK will also be discussed in Chapter 4.

2.2 Related Works

This section will introduce where the main concept of each component comes from.

2.2.1 PCB defect detection

Putera et al., [1] (2010) proposed a computer vision inspection system implemented with MATLAB. The idea of updated defect detection method for keyboard lettering is from this work.

2.2.2 Computer-Vision-Based Fabric Defect Detection

Kumar et al.,(2008) [2] gives a design of CV based inspection platform for fabric defect detecting. We applied the idea connect multiple camera to one host machine in this thesis.

2.2.3 Image Alignment

On the website owned by Satya Mallick, he showed two different way to do the image alignment. One is by enhanced correlation coefficient (ECC) [3], the other is based on feature point matching [4].

2.3 Previous Solution

Previous solution applied SURF feature point on every ROI that represented a key, and we found that feature point based method have accuracy issue on small target searching at production line. Also, the lettering defect detecting method is not sensitive enough to deal with small defect. In this thesis, we'll focus on how to correct these problems.

Chapter 3

Methods

3.1 Inspection Platform Structure

Refer to Kumar's work [2], we designed a scalable platform could install multiple cameras. Use one PC as host machine to collect images from camera array. After retrieved images from camera, the defect detection methods (Lettering Defect Detection & LED Color inspection) described below will inspect whether the keyboard has any defect on it.

3.2 Lettering Defect Detection Introduction

In lettering defect detection, we want to check the layout of keyboard and quality of laser engraving on each key. We'll give a detailed explain of how the lettering defect detection is designed and how they work.

3.2.1 Learning Stage

Lettering defect detection can be separated into two main part. The first part is learning stage. This part reads the gold image from file and doing the feature point detection, and store the feature points & their descriptor for examining stage use.

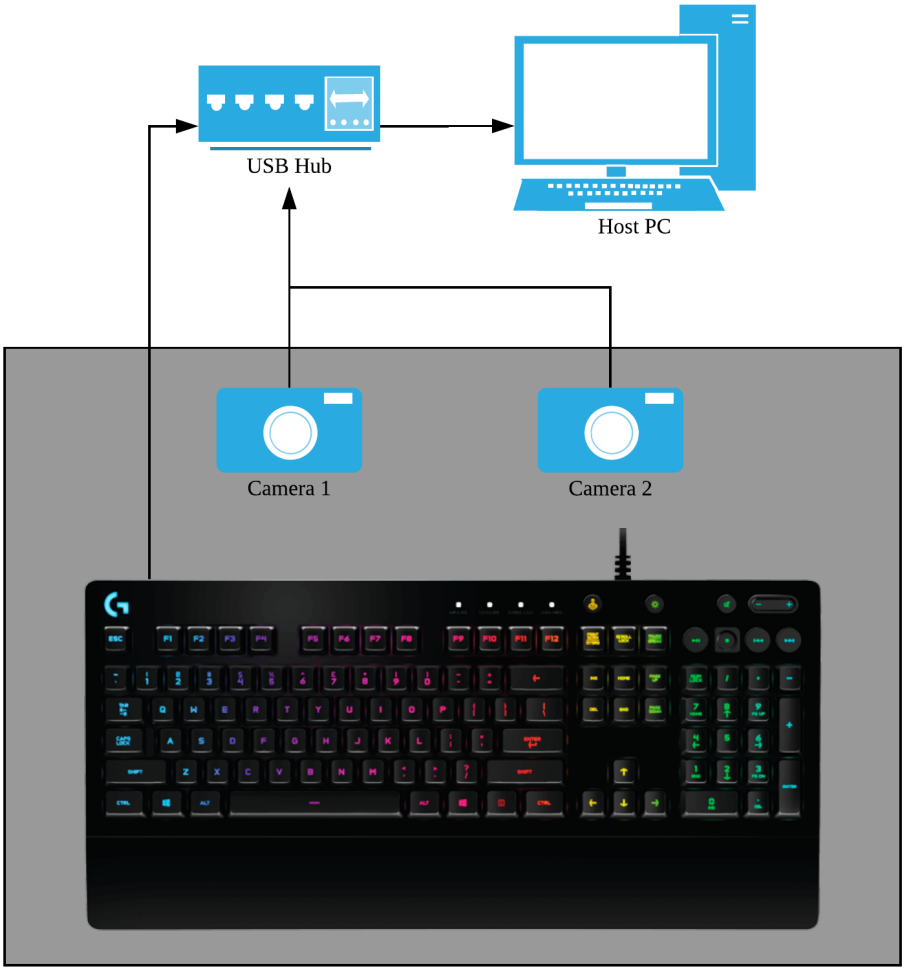

Figure 3.1: The Diagram of SystemStructure

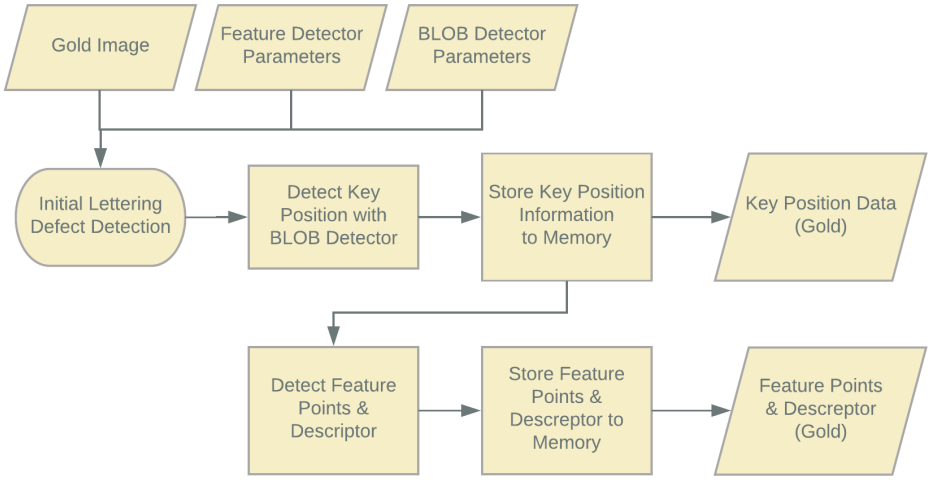

Figure 3.2: The Diagram of Lettering Initialize Stage

3.2.2 Inspection Stage

In lettering inspection stage, we did same steps on the keyboard sample image we want to inspect, doing the feature point detection and store the feature points & their descriptor. After having the feature point and descriptor from both gold & keyboard sample image, we do the feature point matching to find the homography matrix then we can align sample image with gold image with geometric transformation. With both image aligned, we apply the defect detection method to aligned images, and get the marked defect information. And judge the sample keyboard is a defected on or a good one.

3.2.3 Defect Detection Method

In lettering defect detection, after gray-scale gold & sample image aligned, we calculate absolute difference of two image, and get a new gray scale image, say M_{Diff}, that contained defects information. To detect the defect, we could apply simple blob detection with suitable parameters to identify whether a bright spot is defect or noise.

add a flow chart here

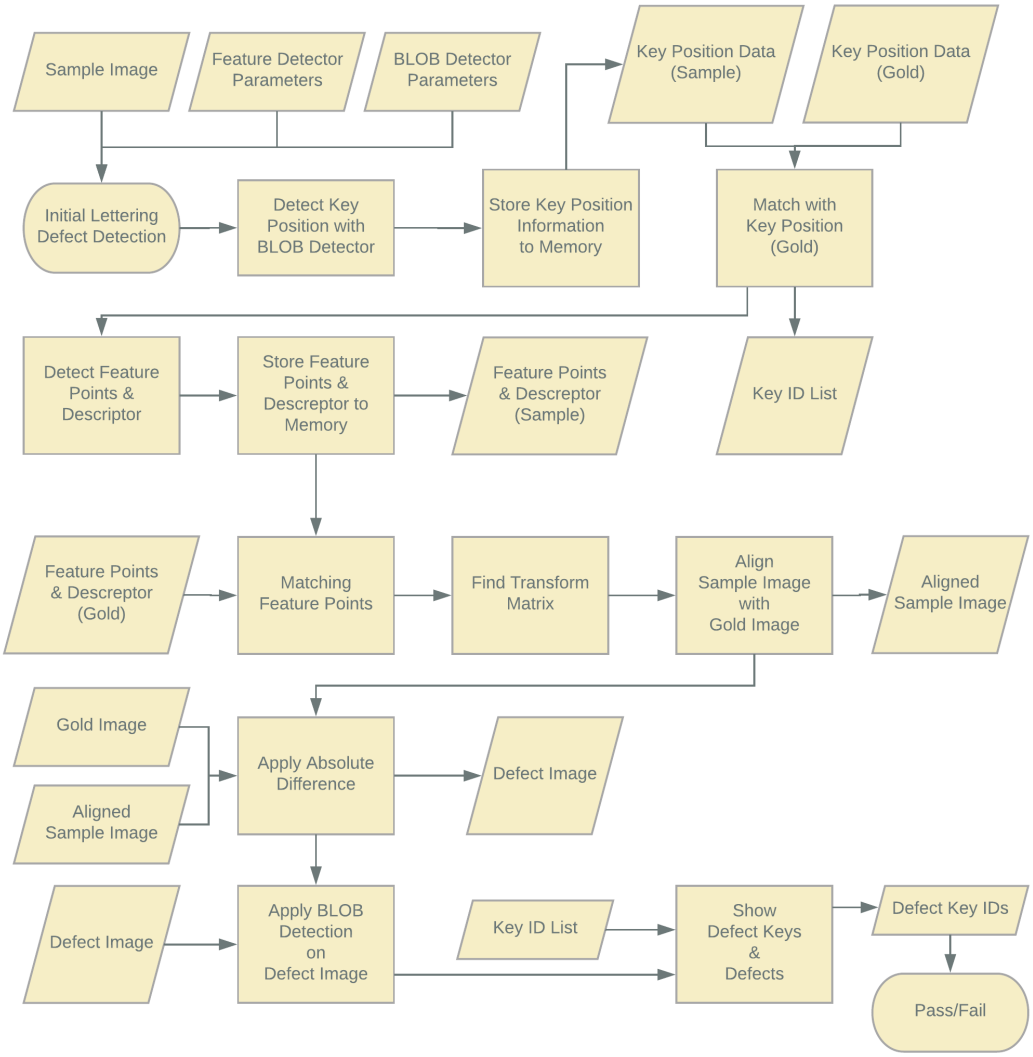

Figure 3.3: The Diagram of Lettering Inspection Stage

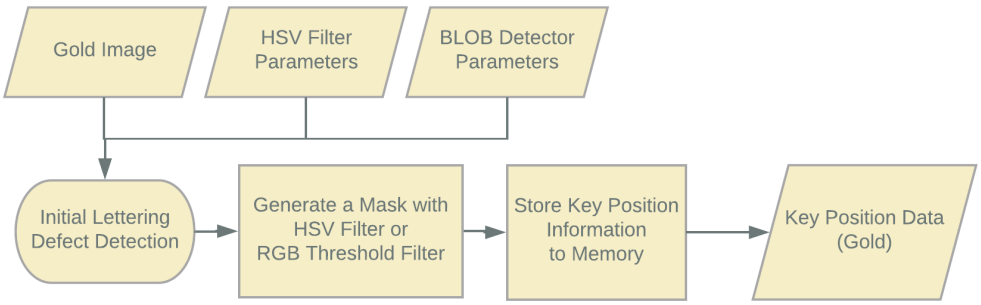

Figure 3.4: The Diagram of LED Initialize Stage

add example image of absdiff result after alignment

3.3 LED Malfunction Detection Introduction

In LED function inspection, we want to check if each LED embedded in the circuit of keyboard functions as expect. In this stage, we'll capture three pair of gold and sample image, $(gold_R, sample_R), (gold_G, sample_G), (gold_B, sample_B)$, used to inspect three different color channel of LED light source. The detailed method will be described in following subsections.

3.3.1 Learning Stage

In this step, we want to build the pass standard with given gold image. By apply corresponding HSV filter, $filter_r, filter_g, filter_b$, to gold images, we can obtain three binary image, witch can use to identify if a testing object functions as this binary image.

3.3.2 Inspection Stage

In this step, we are going to check if a testing object functions as the passing standard. We applied same filter, say $filter_c$, on the image captured from camera. And get a binary image of testing object. We compare the blob size and shape between the binary

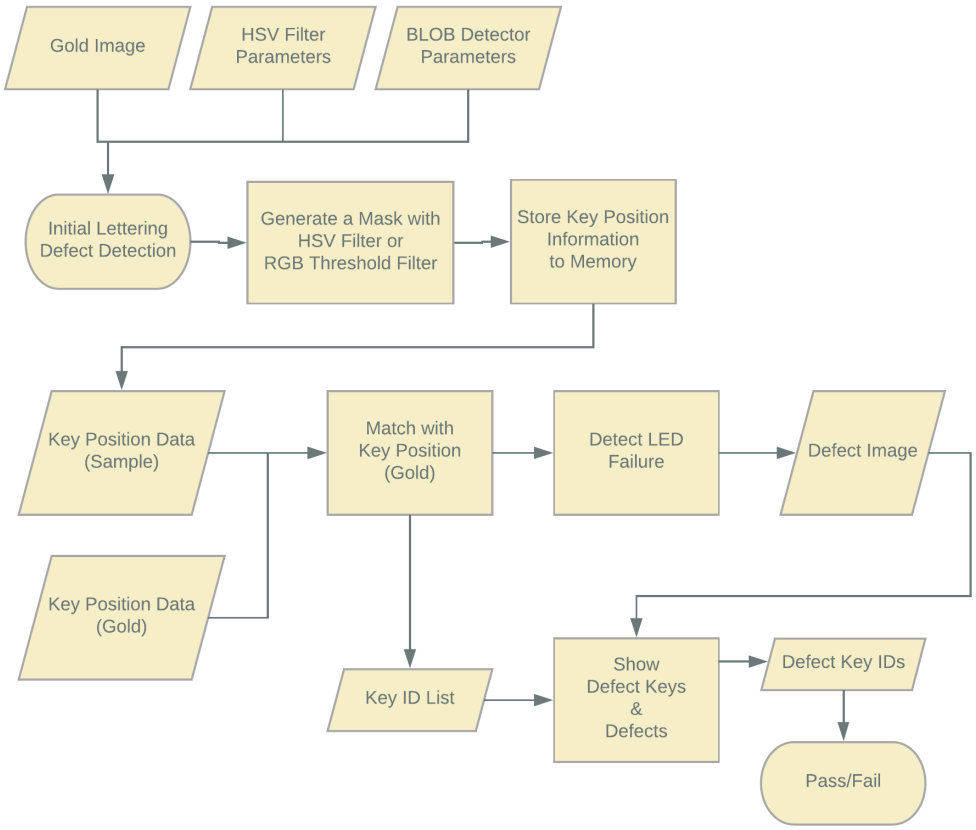

Figure 3.5: The Diagram of LED Inspection Stage

image of gold and sample, to see if there is any malfunctioning on the LED lights.

3.4 Other Functions Implemented

3.4.1 Stamp Detection

Stamp detection method is implemented by solving planar object detection problem. Since the steps doing planar object detection and feature based image alignment are very similar. We first applied feature point detect method, surf in this case, on the stamp image and image that we want to inspect. After applied point matching methods, we could find the possible area for the stamp. Finally, we compare the similarity [1] between the possible area, make sure that there is no misjudge on the inspection.

[1] Similarity is determined by Pearson correlation coefficient.

Chapter 4

Comparison Between Other Methods

4.1 Different Feature Point Extraction & Point Matching Methods

In this section, we'll discuss how different feature point performs in keyboard defect detection scenario. Since the bottleneck of this project is the feature point detection method, we tried some common methods, and below is the observation & time comparison between different approach.

4.1.1 Simple Blob Detector

Introduction

The blob feature detector is an simple approach for feature point detection. We choose the centroid of each selected blob as the feature point. Matching feature points with nearest neighborhood. Since we have a fixture that can guarantee the keyboard does not have shift or rotation more then a certain amount, we can assume that the feature point matched is the nearest one.

Performance

Add time data

Pros & Cons

This method has lowest robustness, but fastest in execution.

4.1.2 SIFT

Introduction

SIFT feature is proposed by Lowe [5] et al., having the state of the art robustness in most scenarios [6], but slower than any other methods on execution.

Performance

Add time data

Pros & Cons

4.1.3 SURF

Introduction

SURF is a upgrade version based on SIFT, witch is designed to solve the speed issue of SIFT. And in the scenario of this thesis, this method meets the requirement on the stability without wasting too much time on calculation.

Performance

Add time data

Pros & Cons

4.1.4 ORB

Introduction

ORB is a

Performance

Add time data

Pros & Cons

4.1.5 FREAK

Introduction

Performance

Add time data

Pros & Cons

4.2 Comparison between each methods

crest a chart here

Chapter 5

Conclusions & Future Works

5.1 Conclusions

In this thesis, we have discussed the performance bottleneck of the inspection methods and improved the inspection methods. Thought the current implementation solved the performance issue of the previous solution, increased the accuracy and precision of the judgment. The implementation doesn't consider the case of conventional keyboards, also doesn't optimized or parallel computing. We will discuss about the future work in the following section.

5.2 Future Works

5.2.1 Ability To Inspect Non-illuminated Keyboards

For the key board lettering inspection method, since the current method cannot handle the shading that is too complex.

5.2.2 Multi-camera and controls optimization

The current version of implementation using the multi-camera in sequential order. Parallel processing techniques is not utilized into this project yet. And every camera we

using is now using a manual adjusted parameter setup. Witch means the system performance is highly depends on the experience of the user of the system.

5.2.3 Parallel computing is not utilized in this project

In this thesis, we didn't fully utilize the computation power of the host PC, only use the sequential processing technique in the current implementation to ensure the stability of the system. The system may have an acceleration about the number of CPU cores in the ideal situation. Witch is a great improvement for the production line application. And GPU is also a powerful tool witch we didn't introduce to this project since the budget issue. With good implementation, GPU may make the real time inspection possible.

5.2.4 Smarter Algorithm

In current implementation, we still need to setup up to ten parameters manually,

References

[1] S. I. Putera and Z. Ibrahim, "Printed circuit board defect detection using mathematical morphology and matlab image processing tools," in *Education Technology and Computer (ICETC), 2010 2nd International Conference on*, vol. 5, pp. V5–359, IEEE, 2010.

[2] A. Kumar, "Computer-vision-based fabric defect detection: A survey," *IEEE transactions on industrial electronics*, vol. 55, no. 1, pp. 348–363, 2008.

[3] S. Mallick, "Image alignment (ecc) in opencv (c++ / python)." https://www.learnopencv.com/image-alignment-ecc-in-opencv-c-python/. Accessed:JULY 1, 2015.

[4] S. Mallick, "Image alignment (feature based) using opencv (c++/python)." https://www.learnopencv.com/image-alignment-feature-based-using-opencv-c-python/. Accessed:MARCH 11, 2018.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] E. Karami, S. Prasad, and M. Shehata, "Image matching using sift, surf, brief and orb: Performance comparison for distorted images," *arXiv preprint arXiv:1710.02726*, 2017.