Group Assignment Template Matching

A. Group Identity

Class	RK-A1
Group	1
Group Member	1. Zinadine Zidan Alsyahana (163221014) 2. Jeffrey Nobel Martin (163221040) 3. Raden Fauzan Wisnuwardhana (163221049) 4. Affa Ndaru Rabbany Wijaya (163221061)

B. Case Title

Template Matching-Based Detection of 4-Pin EDG Components on PCBs

C. Case Description

In modern electronics manufacturing, the printed circuit board (PCB) production process encompasses several critical stages, including photomask pattern generation, naked PCB manufacturing, Surface Mount Device (SMD) mounting, reflow soldering, component insertion with leads, and flow soldering [6]. Quality control throughout these stages is crucial for ensuring product reliability and functionality.

Template matching has emerged as a fundamental image processing technique in the PCB industry, serving as an essential tool for component inspection and quality assurance. This method enables automated systems to determine the presence, position, and orientation of components within captured PCB images [7]. The significance of template matching lies in its ability to facilitate accurate defect detection through systematic comparison between template and target images.

A basic method in computer vision, template matching is frequently applied to industrial inspection, pattern recognition, and object detection applications. It entails using similarity metrics to identify areas in an image that correspond to a predefined template [1]. Techniques like sum of squared differences and cross-correlation are frequently used for the matching process [2] - [5].

Mathematically, the goal is to identify a region R in I that maximizes a similarity function $S(R,T)$ given an input image I of dimensions $m \times n$ and a template T of dimensions $p \times q$. The traditional method is to compute the normalized cross-correlation (NCC) at every potential location of R within I, which is defined as follows [6]:

$$NCC(R,T) = \frac{\sum_{x,y}(I(x,y)-\mu_I)(T(x-u,y-v)-\mu_T)}{\sqrt{\sum_{x,y}(I(x,y)-\mu_I)^2 \sum_{x,y}(T(x-u,y-v)-\mu_T)^2}} \tag{1}$$

where u, v indicate the offset position in I, and μ_I and μ_T represent the image and template's mean intensities, respectively. Although computationally demanding for large images, this measure is resilient to changes in lighting.

The effectiveness of template matching has been enhanced through various innovations. For instance, masked template matching reduces computational overhead by excluding non-component pixels from similarity calculations, thereby improving recognition accuracy. Furthermore, the integration of both 2D and 3D vision techniques has demonstrated superior capabilities in defect detection applications [8].

In PCB assembly, defects can manifest in various forms, primarily categorized into five types: missing components, misalignment, incorrect IC chip orientation, wrong parts installation, and poor solder joints [6]. Statistical analysis of SMD assembly defects reveals the following distribution pattern:

1. Poor Solder Joints (55%)
2. Missing Component (20%)
3. Wrong Orientation (15%)
4. Misalignment (10%)

These statistics underscore the importance of developing robust inspection systems capable of detecting and classifying these defects accurately, particularly for critical components such as 4-pin EDG (Edge Device Group) components, which are the focus of this study.

D. Methods

- **Setup**

  The detection process begins with the setup phase, which involves loading the PCB image and the template images from a specified folder. The PCB image represents the target area where the 4-Pin EDG components need to be detected, while the template images account for various orientations of the component, ensuring robustness during detection.

  Upon loading, all images are converted to grayscale. This step simplifies the detection process by reducing data complexity, focusing solely on intensity values. Grayscale conversion enhances computational efficiency and retains essential structural details critical for accurate template matching. By eliminating color information, the process ensures that the matching algorithm is unaffected by variations in color, improving overall reliability.

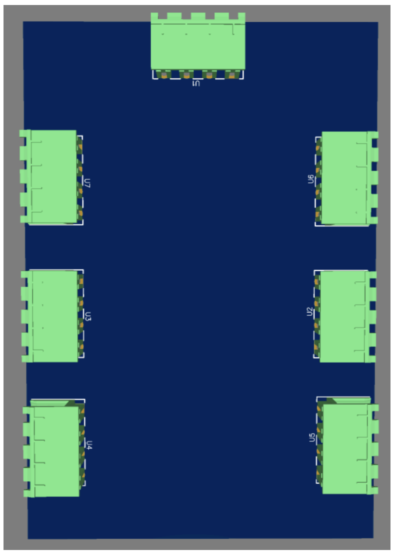

Fig 1. PCB Design Image for EDG Detection.

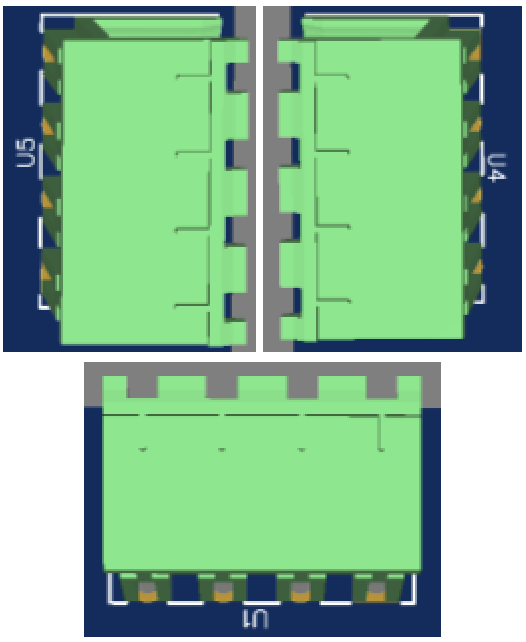

Fig 2. Template EDG Images

- **Template Matching Process**

1. Template Matching

   The core of the detection pipeline is template matching, which is performed for each template image using a normalized correlation method. This technique evaluates how closely each region of the PCB image resembles the template, generating a similarity map. High similarity scores in this map indicate regions that closely match the template.

   The matching process is repeated for all template images, one at a time. This iterative approach ensures that the detection system accounts for all possible orientations and slight variations in the shape or positioning of the 4-Pin EDG components, improving adaptability to real-world scenarios.

2. Candidate Filtering Using Thresholding

   To filter out low-confidence matches, the similarity map undergoes thresholding. A predefined threshold is applied, retaining only regions with high similarity scores as potential matches.

   Bounding boxes are created around these high-similarity regions, representing candidate detections. This step minimizes noise and false positives, ensuring that only the most relevant matches proceed to the next stages of the detection pipeline.

3. Storing Bounding Boxes

   Bounding boxes, defined by their coordinates and dimensions, are stored to mark the locations of detected components on the PCB image. To ensure detection reliability, all bounding boxes, including duplicates, are stored for further refinement.

   This redundant storage allows additional processing steps, such as grouping or filtering overlapping regions, to enhance the accuracy and clarity of the final detection results.

- **Visualization**

  After refining the bounding boxes through processes like non-maximum suppression (NMS), the final set of bounding boxes is visually represented on the PCB image. Colored rectangles are drawn around the detected regions, highlighting the locations of the 4-Pin EDG components.

  The annotated PCB image is then displayed, providing an intuitive visualization of the detection results. This step allows users to verify the success of the template matching method and evaluate the overall performance of the detection pipeline.

Algorithm 1: EDG 4-Pin on PCB Detection Using Template Matching.

```
1:  Load and Preprocess Images:
2:    Load the grayscale PCB image.
3:    Load all template images in grayscale format from the specified folder.
4:  Apply Template Matching:
5:    For each template image in the folder:
6:      Perform template matching between the PCB image and the template.
7:      Obtain a matching score map.
8:  Filter Matches:
9:    Define a threshold value for acceptable matches.
10:   Retain only those matches with a score above the threshold.
11:     Store Bounding Boxes:
12:   For each valid match, compute the bounding box coordinates on the PCB image.
```

```
13: Store all bounding boxes.
14:     **Apply Non-Maximum Suppression**:
15: Use non-maximum suppression to eliminate overlapping bounding boxes and retain
    the most prominent matches.
16:     **Draw Bounding Boxes**:
17: Draw the final bounding boxes on the grayscale PCB image to mark detected
    components.
18:     **Display and Save Results**:
19: Display the PCB image with marked components.
20: Save the processed image for further use or analysis.
```

E. Processing and Analysis Results

Template matching used in this program has been successful in detecting 4-Pin EDG components on a PCB design. The detection process is done by detecting a sample image or template given as in Fig. 2 then template matching will process the template that was given and will match with the given PCB design image as in Fig. 1 the detection results display a rectangle that includes the position of the component that matches the template. There are no significant stacked rectangles, because the rectangle grouping process using the cv2.groupRectangles function successfully eliminates duplication in the same area. The threshold used is 0.8, because after going through several experiments the most perfect result is 0.8. If it is below or above 0.8 then the results are less than optimal or there are some rectangles that collide so the results are not good.

Fig. 3 is the result of the template matching process performed. It can be seen that there is a rectangle surrounding the EDG pin, the rectangle is the identification result of the template given in Fig. 2. None of the rectangles overlap or collide with each other. This is due to the use of the 0.8 threshold earlier, if the threshold used is not 0.8 then the result is that there are some rectangles that collide and the result of the rectangle is truncated or imperfect.

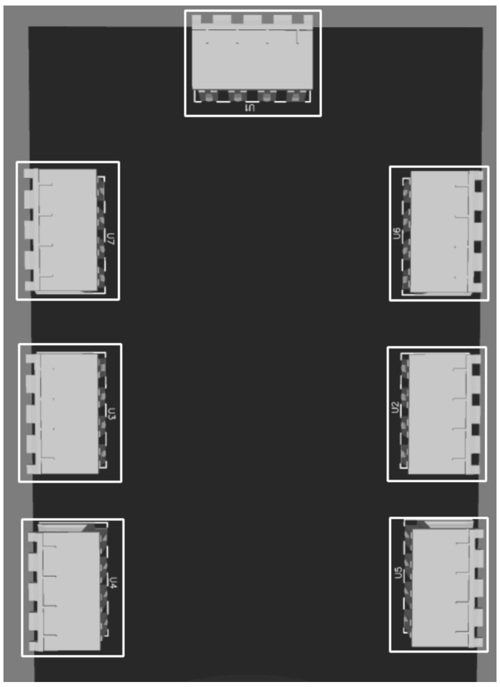

Fig 3. The Result of EDG 4-Pin on PCB Detection Using Template Matching

F. Conclusions and Suggestions

The study demonstrates the efficacy of template matching as a robust technique for detecting 4-Pin EDG components on PCBs. By utilizing grayscale conversion and a normalized correlation method, the detection pipeline achieves high accuracy in identifying component positions, orientations, and potential defects. The iterative matching process ensures adaptability to varying orientations, while thresholding and bounding box refinement enhance the precision of the results. The implementation of the cv2.groupRectangles function effectively eliminates overlapping detections, leading to a clean and reliable output. A threshold of 0.8 was found to yield the most optimal results, as values above or below this benchmark either increased noise or reduced detection performance.

Future work could explore integrating masked template matching or 3D vision techniques to further improve the system's accuracy and efficiency, especially in handling occlusions or varying lighting conditions. Additionally, the incorporation of machine learning methods, such as convolutional neural networks (CNNs), may enhance the system's adaptability and capability to detect a broader range of components or defects. Expanding the application to include real-time detection systems could also bring significant advantages to modern PCB manufacturing processes, further contributing to quality assurance and production reliability.

G. References

[1] J. Ma, X. Jiang, A. Fan, J. Jiang, and J. Yan, "Image Matching from Handcrafted to Deep Features: A Survey," *International Journal of Computer Vision*, vol. 129, no. 1, pp. 23–79, 2021, doi: https://doi.org/10.1007/s11263-020-01359-2.

[2] M. B. Hisham, S. N. Yaakob, R. A. A. Raof, A. B. A. Nazren, and N. M. Wafi, "Template Matching using Sum of Squared Difference and Normalized Cross Correlation," *2015 IEEE student conference on research and development (SCOReD)*, pp. 100–104, 2015, doi: https://doi.org/10.1109/SCORED.2015.7449303.

[3] J. P. Kern and M. S. Pattichis, "Robust Multispectral Image Registration Using Mutual-Information Models," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1494–1505, 2007, doi: https://doi.org/10.1109/tgrs.2007.892599.

[4] S. Suri and P. Reinartz, "Mutual-Information-Based Registration of TerraSAR-X and Ikonos Imagery in Urban Areas," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 2, pp. 939–949, 2010, doi: https://doi.org/10.1109/tgrs.2009.2034842.

[5] Z. Cui, W. Qi, and Y. Liu, "A Fast Image Template Matching Algorithm Based on Normalized Cross Correlation," *Journal of Physics: Conference Series*, vol. 1693, 2020, doi: https://doi.org/10.1088/1742-6596/1693/1/012163.

[6] N. H.-H. Loh and N. M.-S. Lu, "Printed circuit board inspection using image analysis," pp. 673–677, 2002, doi: https://doi.org/10.1109/iacet.1995.527640.

[7] A. J. Crispin and V. Rankov, "Evolutionary algorithm for PCB inspection," *International Journal of Knowledge-based and Intelligent Engineering Systems*, vol. 13, no. 3–4, pp. 91–101, 2009, doi: https://doi.org/10.3233/kes-2009-0177.

[8] L. Hua, K. Li, L. Cheng, Y. Chen, D. Yin, and F. R. Yu, "Research on PCB Defect Detection Using 2D and 3D Segmentation," *Communications in computer and information science*, pp. 16–28, 2024, doi: https://doi.org/10.1007/978-981-97-1280-9_2.

H. Code

```
import cv2
import numpy as np
import os

# Paths for the PCB image and template folder
pcb_image_path = 'path/to/pcb_image.jpg'
template_image_path = 'path/to/4pin_template.jpg'

# Load the PCB image in grayscale mode
pcb_image = cv2.imread(pcb_image_path, cv2.IMREAD_GRAYSCALE)

# Threshold for template matching confidence
threshold = 0.8

# List to store rectangles corresponding to matching regions
all_rectangles = []

# Iterate through all files in the template folder
for filename in os.listdir(template_folder_path):
    template_path = os.path.join(template_folder_path, filename)

    # Check if the file is a valid image
    if os.path.isfile(template_path) and
template_path.lower().endswith(('.png', '.jpg', '.jpeg', '.bmp', '.tiff')):
        # Load the template image in grayscale mode
        template = cv2.imread(template_path, cv2.IMREAD_GRAYSCALE)

        # Get dimensions of the template
        template_height, template_width = template.shape

        # Perform template matching
        result = cv2.matchTemplate(pcb_image, template,
cv2.TM_CCOEFF_NORMED)

        # Get locations where matching confidence exceeds the threshold
        locations = np.where(result >= threshold)

        # Add matching rectangles to the list
        for point in zip(*locations[::-1]):  # Reverse to get (x, y)
coordinates
            # Append the matching rectangle twice for further processing (if
needed)
```

```
                all_rectangles.append([int(point[0]), int(point[1]),
int(template_width), int(template_height)])
                all_rectangles.append([int(point[0]), int(point[1]),
int(template_width), int(template_height)])

   # At this point, `all_rectangles` contains rectangles for all matching
regions
```