A Simple, Precise, and High-Speed Die Edge Detection Framework for the Semiconductor Industry

XiaoJian Tan1*, WaiZhe Leow2, WaiLoon Cheor2

1Faculty of Engineering and Technology, Tunku Abdul Rahman University College
2Faculty of Electronic Engineering Technology, Universiti Malaysia Perlis
*corresponding author: xj_0506@hotmail.com

Abstract

This paper presents an automated high-speed die edge detection method that is meant to be implemented in non-destructive die-level defects inspection in the semiconductor industry. This paper specifically aims to achieve high accuracy (or yield) of greater than 99.95% with stable performance within a short computation time (i.e., less than 30 ms), provided the targeted unit-per-hour (UPH) is at 30,000. To demonstrate the applicability of the proposed method, the proposed method is validated using three production units (i.e., Production units A: 6000 units; Production unit B: 3500 units; Production unit C: 4000 units) and is benchmarked to two common edge detection methods that are widely implemented in the semiconductor industry nowadays (i.e., cross-correlation and normalized cross-correlation methods). The results obtained show that the proposed method is capable to perform die edge detection with promising accuracy and stable performance by achieving 100.0% yield in all three production units and outperformed the benchmarking methods. Also, the overall computation time (considering die edge detection and die rotation) of the proposed method is short, which is approximately 30 ms.

Keywords - non-destructive; automated optical inspection (AOI) system; edge detection; real-time; high-speed; precise detection

1. Introduction

The semiconductor industry is one of the most competitive and fast-evolving industries. In order to fulfill the high quality and consumption requirements, the product quality must be at an excellent level with mass production. Defective products are proven to impinge the manufacturing process and increase cost in general [1]. An automated optical inspection (AOI) system is a non-destructive approach that is widely implemented in the semiconductor industry typically for quality assurance purposes [2, 3]. Die edge detection is the first and most fundamental step in defects detection and dies coordination in an AOI system. Faulty edge detection can result in underkill (false negative) and overkill (false positive) which contributes to major end-product quality issues. In most cases, faulty edge detection can return the AOI system a wrong die coordinate (i.e., center coordinate) which results in a wrong picking center. Thus, a “flying die” phenomenon could occur whereby the die is detached from the picking collet during the high-speed machine motion.

For application in an AOI system, the edge detection method must fulfill the following criteria: (a) the accuracy must be high and performance must be stable, and (b) the computation time must be low. In most of the cases, the accuracy requirement (or commonly known as yield) for edge detection is at 99.95% to 99.99%. In terms of computation time, the unit-per-hour (UPH) requirement could be varying from 5,000 UPH to 50,000 UPH dependent on the machine capability. The higher the UPH, the better the throughput of the AOI system, the higher the machine price. Taking an AOI system with 30,000 UPH as an example, the total allowable time to process one die is 120 ms. Thus, the ideal computation time for edge detection would be 1/4 of the total allowable time (i.e., approximate 30 ms).

Edge detection is performed by identifying the discontinuities of the pixel intensities in an input image. Sobel [4, 5], Prewitt [6, 7], and Canny [8-10] are some common operators and filters which can effectively locate the boundaries of an object in the input image. To date, many researchers proposed various effective edge detection methods but with a tradeoff of high computation complexity which are not suitable for high-speed application. Chao and Tsai in [11] proposed an improved anisotropic diffusion method to detect defects on low contrast surface of a glass object. The anisotropic diffusion approach or commonly known as Perona-Malik diffusion [12] is a famous edge detection approach that can retain important lines and edges features while minimize the image noise. In [11], the authors combined the conventional anisotropic diffusion with an adaptive smoothing process to improve the defects detection process on the glass object. The reported computation time for the proposed method is approximately 300 ms. In [13], the same authors proposed another edge detection method by combining the conventional anisotropic diffusion with particle swarm optimization (PSO). The proposed method aims to detect defects on liquid-crystal display (LCD) in low contrast backlight. By using the improved method, the overall computation time is reduced to 60 ms. However, the algorithm is only applicable to non-textured surfaces which greatly limit its application in the semiconductor industry as the die material could possibly black-coated with textured surface. Chang et. al. proposed a simple edge detection method based on thresholding to identify cracks on touch panel [14]. The proposed method can achieve high accuracy (i.e., 94.0 to 99.0%) but with a tradeoff of long computation time.

It is worthwhile to remark that the template matching method [15] is of the most common methods implemented in the semiconductor industry for edge detection purposes [3, 16-19]. The template matching method can locate the edge of an object...
by measuring the similarity between the template image and input image based on the difference in the Mean Squared Error (MSE) metric [20]. The feature-based template matching approach has better connection in terms of control points and features between the input image and the template image [20]. Cross-correlation and normalized cross-correlation methods are two typical examples in template matching approach. In cross-correlation method, the similarity is calculated basically by computing the sum of pair wise multiplications on the particular pixels in both the images. The larger the value, the greater the similarity. The normalized cross-correlation method is an improved method of the conventional cross-correlation method.

The main improvements lie within the ability to omit the changes in global brightness, such that regions with consistent bright and dark pixels will not affect the output of the calculation. Also, the output of the normalized cross-correlation method is always range from -1 to 1. For a better understanding in the cross-correlation and normalized cross-correlation methods, please refer to [21] and [22], respectively. Nevertheless, the template matching method can result in faulty edge detection which lead to an undesirable quality issue (i.e., overkill). Faulty edge is usually occurred when the edge detection is performed on die with markings, especially die with inconsistent marking location (refer to Figure 1 (a) and (b)). The overall similarity metric returns from the template matching method is predominant by the inconsistent marking location which act as the prominent features in template matching method. This similarity metric is then lead to a faulty edge detection (refer to Figure 1 (c)).

Fig. 1

In this paper, an automated high-speed non-destructive die edge detection method is proposed. The proposed method aims to be implemented in an AOI system for non-destructive optical inspection on die-level in a real manufacturing production line, typically in the semiconductor industry. Based on the recent literature, the various edge detection methods are either effective with long computation time [11-14] or short computation time with a trade-off of unstable accuracy and performance stability [16-19]. The main objectives of this paper are to address the research gap by proposing a die edge detection method which could achieve promising accuracy with stable performance within a short computation time. The proposed method consists of two main stages: (a) Stage 1: learn phase and (b) Stage 2: execution phase. The remainder of this paper is organized as follows: Section 2 detailed the proposed AOI system (i.e., image acquisition and detailed algorithm of the proposed method); Section 3 presents the experimental validation, and Section 4 remarks the conclusion of this paper.

2. The proposed AOI system

The proposed edge detection method first starts with the image acquisition stage, then followed by Stages 1 (learn phase) and 2 (execution phase), respectively. In image acquisition stage, the image of the die is captured and is stored in bitmap format (.bmp). In Stage 1, the user is required to select a non-defective die from the production unit to perform the learn phase. The main purpose of the learn phase is to compute the mean intensity difference ($u^\top$) which is an essential parameter in the next stage. It is important to remark that only one unit is required to perform the learn phase for each production unit. Stage 2 is the core image processing algorithm stage whereby termed as the execution phase. The images captured in image acquisition stage are used as input in this stage for edge detection and die rotation purposes. Figure 2 shows the overall block diagram of the stages involved in this paper. The methodology of each stage is detailed in the subsequent subsections.

Fig. 2

2.1 Image acquisition Stage

In a real manufacturing production line, the imaging environment and lighting configuration can substantially affect the accuracy of die edge detection [3, 23]. Figure 3 shows the architecture and hardware setup of the proposed AOI system. A Toshiba CMOS monochrome camera (2.2MP) is used to perform the image acquisition. The camera is mounted to a Navitar 12x zoom motorized lens together with a C-mount coupler, a mini adapter, and a 0.5x lens attachment. An external coaxial light (red) is fixed under the lens and followed by the low angle ring light (blue). The external coaxial light (red) and low angle ring light (blue) are hold in position using the mechanical arms with specific working distance (WD). The WD of the lens, external coaxial light (red), and low angle ring light (blue) are 165.0 mm, 56.0 mm, and 14.0 mm, respectively. The die is placed in a pocket of a carrier tape, whereby the carrier tape is hold by a bracket in a static position (refer to Figure 4).

Fig. 3

Fig. 4

In this paper, the die is the object of interest and shall be prominent from the background. This could be done by fine-tuning the camera and light settings of the imaging configuration, which included the camera exposure time, camera gain, intensity of the external coaxial light (red), and intensity of the low angle ring light (blue). Table 1 shows the camera and light settings of the image configuration.

Table 1

The algorithm of the proposed method is developed and execute using a Windows 7 Professional personal computer with Intel (R) Core (TM) i7-6700 CPU @ 3.4GHz and equipped with installed memory (RAM) of 8.00 GB. The images captured in this stage are in gray scale with a resolution of 1088x1088 at a calibration value of 4.1360µm/ pixel.

2.2 Stage 1: Learn phase

Stage 1 (learn phase) starts by selecting a non-defective unit as the input image. The die and background are then characterized into two distinct clusters using an improved K-Mean clustering method, namely K-Mean with histogram-based initialization [24]. This method can effectively reduce the computation time
of the conventional K-Mean by providing the initial values based on the histogram analysis of the input image. The final centroid can be calculated with shorter iterations as compared to the conventional K-Mean. Also, the K-Mean with histogram-based initialization can eliminates inappropriate initial values which may lead to center redundancy, dead center, and possibility of trapping in the local minima [25].

Next, hole filling is performed on the output obtained from the K-Mean with histogram-based initialization. Figure 5 shows the output of the learn phase. The die and background are now white and black in color, respectively (refer to Figure 5 (c)). The coordinates of the detected die edge (i.e., edge of the white object) is used to calculate the mean intensity difference \( u \) in the input image using Equation 1 (refer to Figure 5 (d)).

\[
\bar{u} = \frac{\sum_{E} \left| \text{Intensity}_{ FTC} - \text{Intensity}_{ BTC} \right|}{E}
\]

where \( \text{Intensity}_{ FTC} \) denotes the intensity of the foreground transition coordinate (refer to Figure 5’ (d)); \( \text{Intensity}_{ BTC} \) denotes the intensity of the background transition coordinate (refer to Figure 5 (d)); and \( E \) denotes the total number of transition coordinates (i).

**Fig. 5**

### 2.3 Stage 2: Execution phase

Figure 6 shows the detailed flow chart of the proposed method: Stage 1 (learn phase) and Stage 2 (execution phase). The \( u \) obtained in Stage 1 is used in this stage for die edge detection purposes. In detailed algorithms of the execution phase is as follows:

Step 1: Landscape analysis in \( x \) and \( y \)-axis. Calculate the intensity difference for each pixel line in \( x \) and \( y \)-axis respectively. The pixel is set as foreground if and only if the intensity difference is greater than \( u \) which obtained from Stage 1.

Step 2: Grouping the coordinates. The coordinates of the foreground pixel are then characterized into four groups in order to identify which edge the specific coordinate is belonging to. This is done by calculating the minimum Euclidean Distance from four seed points, namely \( \text{Coordinates}_L \) and \( \text{Coordinates}_{SB} \) (for \( x \)-axis); \( \text{Coordinates}_T \) and \( \text{Coordinates}_{SB} \) (for \( y \)-axis). The coordinates of foreground pixel with minimum Euclidean Distance to the \( \text{Coordinates}_L \), \( \text{Coordinates}_T \), \( \text{Coordinates}_R \), and \( \text{Coordinates}_{SB} \) are respectively termed as group points left, top, right, and bottom. The \( \text{Coordinates}_L \), \( \text{Coordinates}_T \), \( \text{Coordinates}_R \), and \( \text{Coordinates}_{SB} \) are calculated using Equations 2 to 5, respectively. Figure 7 illustrates the coordination of each seed point.

\[
\text{Coordinate}_L = \begin{cases} 
  x_L = \frac{\text{Resolution}_x}{4} \\
  y_L = \frac{\text{Resolution}_y}{2}
\end{cases}
\]

\[
\text{Coordinate}_T = \begin{cases} 
  x_T = \frac{\text{Resolution}_x}{2} \\
  y_T = \frac{\text{Resolution}_y}{4}
\end{cases}
\]

**Fig. 6**

**Fig. 7**

Step 3: Outlier elimination. The detected edge (i.e., foreground pixels in Step 1) may consist of outlier if the edge is a defective unit with defects such as chipping, foreign material, and/or burr edge. These defects can potentially impinge the die edge detection. In order to eliminate the outlier, the line of best fit must first be calculated for each group point (i.e., group points left, top, right, and bottom) by using Equations 6 to 8. Figure 8 illustrates the line of best fit of a specific edge.

\[
ax_r + by_r + c = 0
\]

\[
a = \frac{\sum_{r=1}^{n} (x_r - \bar{x})(y_r - \bar{y})}{\sum_{r=1}^{n} (x_r - \bar{x})^2}
\]

\[
\text{such that } \bar{x} = \frac{\sum_{r=1}^{n} x_r}{n} \text{ and } \bar{y} = \frac{\sum_{r=1}^{n} y_r}{n}
\]

\[
c = \bar{y} - a\bar{x}
\]

where \( a \) denotes the slope, \( b \) denotes the coefficient of \( y_r \), \( x_r \) and \( y_r \) denote the arbitrary coordinate \( (x_r, y_r) \) for each \( r \), \( c \) denotes the the y-intercept, \( \bar{x} \) denotes the mean of the x-values, \( \bar{y} \) denotes the mean of the y-values, and \( n \) denotes the total number of \( r \).

**Fig. 8**

Next, calculate the perpendicular distance \( d \) of an arbitrary coordinate \( (x, y) \) from the line of best fit using Equation 9. Eliminate the outlier using the boxplot method [26] and recalculate the line of the best fit using Equations 6 to 8. This line of best fit is optimized and is termed as optimized line of best fit in the subsequence section.

\[
d = \frac{|ax+y+b+c|}{\sqrt{a^2+b^2}}
\]

Step 4: Calculation of die corners. In this step, the die corners are determined based on the optimized line of best fit in Step 3. The die corners located at top left, top right, bottom right, and bottom left are termed as die corners P, Q, R, and S, respectively. The die corners P, Q, R, and S can be calculated by determining the intersection of two optimized lines of best fit as follows:

- Die corner P: intersection of top and left optimized lines of best fit;
- Die corner Q: intersection of top and right optimized lines of best fit;
- Die corner R: intersection of right and bottom optimized lines of best fit;
• Die corner S: intersection of bottom and left optimized lines of best fit.

The corners P, Q, R, and S can be calculated using Equation 10.

\[
\text{Corner } P/Q/R/S = \left( \frac{b\text{line } 1^2 \text{line } 2^2 \text{line } 1^2 \text{line } 2^2 \text{line } 1^2}{a\text{line } 1^2 \text{line } 2^2 \text{line } 1^2 \text{line } 2^2 \text{line } 1^2} \right)
\]

Step 5: Calculation of die center. An accurate die center is important for die picking purposes in an AOI system. The die center can be calculated by using the x and y coordinates of the detected edge. Equation 11 shows how the die center is computed. The die edge detection is now completed.

\[
\text{Center coordinate } (x_{center}, y_{center}) = \left( \frac{\sum_{i=1}^{n} x_i}{n}, \frac{\sum_{i=1}^{n} y_i}{n} \right)
\]

Step 6: Die rotation. Die rotation is not an essential step in die edge detection. However, it is important to remark that the die rotation can greatly reduce the complexity for majority of the defect detections. The defect detection algorithms can be implemented without considering the orientation and/or angle correction of the input die. This can substantially reduce the computation time in the subsequent defect detection steps. In order to perform the die rotation, the rotation angle and the direction of rotation must first be determined. Figure 9 illustrates how the die rotation is performed. From this figure, the rotation angle from the Coordinate E (the original arbitrary die corner) to the Coordinate G (the rotated arbitrary die corner) on the Coordinate F (the die center) can be calculated by using Equations 12 to 14, where the Coordinate E is obtained from Step 4 (in Equation 10), the Coordinate G is calculated using Equation 15, and the Coordinate F from Step 5 (in Equation 11). The direction of rotation is determined by using Equation 16.

Fig. 9

\[
\theta = \arccos \left( \frac{\vec{EF} \cdot \vec{FG}}{\| \vec{EF} \| \| \vec{FG} \|} \right)
\]

\[
\vec{EF} = F - E
\]

\[
\vec{FG} = G - F
\]

\[
\text{Coordinate } G(x_{\text{rotated}}, y_{\text{rotated}}) = \left( x_{center} - \frac{w}{2}, y_{center} - \frac{h}{2} \right)
\]

\[
\text{die rotation} = \begin{cases} 
0 & \text{if } (y_{\text{original}} - y_{\text{rotated}} > 0): \text{clockwise rotation} \\
0 & \text{else if } (y_{\text{original}} - y_{\text{rotated}} < 0): \text{anti-clockwise rotation} \\
0 & \text{else: no rotation}
\end{cases}
\]

where \(\vec{EF}\) denotes the vector from E to F, \(\vec{FG}\) denotes the vector from F to G, \(\| \vec{EF} \|\) denotes the magnitude of vector \(\vec{EF}\), \(\| \vec{FG} \|\) denotes the magnitude of vector \(\vec{FG}\), \(w\) denotes the width of the die, and \(h\) denotes the height of the die.

3. Experimental Validation

To validate the applicability of the proposed method in a real manufacturing production line, three production units namely production units A, B, and C are used. The die size of the production unit A is 1.3 x 1.3 mm with a total of 6000 units. The die size of production units B and C are 2.0 x 1.6 mm, whereby the total units are 3500 and 4000, respectively. The validation criteria are set based on the requirements as stated in Section 1 whereby the proposed edge detection method is evaluated in terms of: (a) the die edge detection accuracy and stability and (b) the computation time. The minimum edge detection accuracy (i.e., yield) is set as 99.95% and the computation time for edge detection shall be not more than 30 ms in general, provided the target UPH is at 30,000. For benchmarking purposes, the proposed method is compared to two common edge detection methods: (a) cross-correlation and (b) normalized cross-correlation methods. In order to establish a fair comparison, the input images for cross-correlation and normalized cross-correlation methods are captured using the same imaging environment, lighting configuration, as well as the same camera and light settings as in image acquisition stage in Section 2.1. All three methods are executed using the same computer environment. Figure 10 shows the selected results of die edge detection using the three methods (i.e., cross-correlation, normalized cross-correlation, and the proposed method). The overall die detection accuracy (i.e., yield) is depicted in Table 2. Table 3 shows the quality issues due to faulty die edge detection for correlation, normalized correlation, and proposed methods. Table 4 shows the comparison in mean computation time for all three methods.

In Figure 10, a sample of edge detection result is selected from each production unit. Based on the results, the cross-correlation and normalized cross-correlation methods show faulty edge detections. The faulty detected die edges located beyond the real die edge (Figure 10 (a), (b), (e), (i), and (j)) can potentially result in the “flying die” phenomenon and overkill in defect detection. For faulty edge detection by which the detected edge is located within the real die edge as shown in Figure 10 (f), the return die center is incorrect and can lead to a die picking issue in the later step. Also, the area between the faulty detected edges and real die edge would be characterized as defects such as foreign material and/ or burr edge. By comparing the edge detection obtained from the normalized cross-correlation method and the proposed method as in Figure 10 (j) and (k), the proposed method (Figure 10 (k)) shows better sensitivity and outperformed the normalized cross-correlation method with better accuracy in die edge detection. The superior die edge detection found in the proposed method can be explained by Step 3 in Stage 2 (execution phase). The optimized line of best fit is found proficient and contribute to better die edge detection in an AOI system.

Fig. 10

Table 2
An automated high-speed non-destructive die edge detection method for the semiconductor industry is proposed in this paper. The proposed method aims to be implemented in an AOI system in a real manufacturing production line to perform the non-destructive optical inspection on die for quality assurance purposes. The proposed method consists of two main stages: Stage 1 (learn phase) and Stage 2 (execution phase). In image acquisition stage, the imaging environment and lighting configuration have been proposed. The images captured in this stage are used as input to Stages 1 and 2. In Stage 1, a non-defective unit is selected to use as an input to the learn phase. The main purpose of Stage 1 is to compute the $u'$ which is an essential parameter in the next stage. In Stage 2, the core image processing algorithm is performed. The images captured in image acquisition stage are used as input in this stage. The main algorithms in Stage 2 are: (1) landscape analysis in the $x$ and $y$-axis, (2) grouping the coordinates, (3) outlier elimination and calculation of optimized line of best fit, (4) calculation of die corners, (5) calculation of die center, and (6) die rotation. The proposed method is validated using three production units namely production units A, B, and C. The proposed method is benchmarked to two common die edge detection methods in the semiconductor industry (i.e., cross-correlation and normalized cross-correlation methods). Based on the experimental validation, the proposed method found outperformed the cross-correlation and normalized cross-correlation methods by achieving promising yields in die edge detection of 100.0% for all three production units. The promising yields obtained also proven the stability of the proposed method in die edge detection. In terms of computation time, the die edge detection of the proposed method is not the shortest but comparative to the cross-correlation method (refer to Table 4). Nevertheless, taking UPH of 30,000 as the target, the mean computation time of the die edge detection of the proposed method for all three production units is acceptable as it is not more than 30 ms. Furthermore, when considering the computation time of the die rotation, the proposed method remarks a mean computation time of 29 ($\pm$ 2) ms, 30 ($\pm$ 3) ms, and 29 ($\pm$ 2) ms for production units A, B, and C, respectively. The obtained mean computation time (with die rotation) are very encouraging as the die rotation is expected to facilitate the defect detection algorithms in the later steps. Overall, the proposed method demonstrates its proficiency and applicability in die edge detection with promising accuracy or yield, stable performance with short computation time. By incorporating the proposed method in an AOI system for the semiconductor industry, the overall defect detection algorithms could be improved and the “flying die” phenomenon which potentially leads to quality issues can be eliminated.
Declarations of Competing Interest
We declare that we have no financial or non-financial conflicts of interest to disclose.

References
1. Soto, J.A.C., Tavakolizadeh, F., Gyalui, D.: An online machine learning framework for early detection of product failures in an industry 4.0 context. Int. J. Comput. Integr. Manuf. 32, 452–465 (2019)
2. Ahmadi, B., Heredia, R., Shahbazizadeh, S., Shahbazi, Z.: Non-destructive automatic die-level defect detection of counterfeit microelectronics using machine vision. Microelectron. Reliab. 114 (2020). https://doi.org/10.1016/j.microrel.2020.113893
3. Ebayeeh, A.A.R.M.A., Mousavi, A.: A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry. IEEE Access. 8, 183192–183271 (2020). https://doi.org/10.1109/ACCESS.2020.3029127
4. Jing, M., Du, Y.: Flank angle measurement based on improved Sobel operator. Manuf. Lett. 25, 44–49 (2020). https://doi.org/10.1016/j.mfglet.2020.07.002
5. Ravivarma, G., Gavaskar, K., Malathi, D., Asha, K.G., Ashok, B., Aarthi, S.: Implementation of Sobel operator based image edge detection on FPGA. Mater. Today Proc. (2020). https://doi.org/10.1016/j.matpr.2020.10.825
6. Zhang, H., Zhu, Q., Fan, C., Deng, D.: Image quality assessment based on Prewitt magnitude. AEU - Int. J. Electron. Commun. 67, 799–803 (2013). https://doi.org/10.1016/j.aeue.2013.04.001
7. Wang, J., Fu, P., Gao, R.X.: Machine vision intelligence for product defect inspection based on deep learning and Hough transform. J. Manuf. Syst. 51, 52–60 (2019). https://doi.org/10.1016/j.jmsy.2019.03.002
8. Zhang, X., Zhang, Y., Zheng, R.: Image edge detection method of combining wavelet lift with Canny operator. Procedia Eng. 15, 1335–1339 (2011). https://doi.org/10.1016/j.proeng.2011.08.247
9. Jiang, W., Zhou, H., Shen, Y., Liu, B., Fu, Z.: Image segmentation with pulse-coupled neural network and Canny operators. Comput. Electr. Eng. 46, 528–538 (2015). https://doi.org/10.1016/j.compeleceng.2015.03.028
10. Tong, J., Shi, H., Wu, C., Jiang, H., Yang, T.: Skewness correction and quality evaluation of plug seedling images based on Canny operator and Hough transform. Comput. Electron. Agric. 155, 461–472 (2018). https://doi.org/10.1016/j.compag.2018.10.035
11. Chao, S.M., Tsai, D.M.: Anisotropic diffusion-based defect detection for low-contrast glass substrates. Image Vis. Comput. 26, 187-200 (2008).
12. Pietro, P., Jitendra, M.: Scale-space and edge detection using anisotropic diffusion. IEEE Trans. Pattern Anal. Mach. Intell. 12 (2002).
13. Chao, S.M., Tsai, D.M.: Anisotropic diffusion with generalized diffusion coefficient function for defect detection in low-contrast surface images. Pattern Recognit. 43, 1917–1931 (2010). https://doi.org/10.1016/j.patcog.2009.12.005
14. Chang, M., Chen, B.C., Gabayno, J.L., Chen, M.F.: Development of an optical inspection platform for surface defect detection in touch panel glass. Int. J. Optomechatronics. 10, 63–72 (2016). https://doi.org/10.1080/15599612.2016.1166304
15. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: A review. IEEE Trans. Pattern Anal. Mach. Intell. 22, 4–37 (2000). https://doi.org/10.1109/34.824819
16. Zhong, F., He, S., Li, B.: Blob analysis-based template matching algorithm for LED chip localization. Int. J. Adv. Manuf. Technol. 93, 55–63 (2017). https://doi.org/10.1007/s00170-015-7638-5
17. Chen, F., Ye, X., Yin, S., Ye, Q., Huang, S., Tang, Q.: Automated vision positioning system for dicing semiconductor chips using improved template matching method. Int. J. Adv. Manuf. Technol. 100, 2669–2678 (2019). https://doi.org/10.1007/s00170-018-2845-5
18. Cho, H.J., Park, T.H.: Wavelet transform based image template matching for automatic component inspection. Int. J. Adv. Manuf. Technol. 50, 1033–1039 (2010). https://doi.org/10.1007/s00170-010-2567-9
19. Fonseka, S., Jayasinghe, J.A.K.S.: Feature extraction and template matching algorithms classification for PCB fiducial verification. J. Achievements Mater. Eng. 86, 14–32 (2018)
20. Swaroop, P., Sharma, N.: An overview of various template matching methodologies in image processing. Int. J. Comput. Appl. 153, 975–987 (2016)
21. Li, W., Huang, Y.: A combined method of cross-correlation and PCA-based outlier algorithm for detecting structural damages on a jacket oil platform under random wave excitations. Appl. Ocean Res. 102 (2020). https://doi.org/10.1016/j.apor.2020.102301
22. Nakhmani, A., Tannenbaum, A.: A new distance measure based on generalized image normalized cross-correlation for robust video tracking and image recognition. Pattern Recognit. Lett. 34, 315–321 (2013). https://doi.org/10.1016/j.patrec.2012.10.025
23. Miao, Y., Jeon, J.Y., Park, G.: An image processing-based crack detection technique for precision panel products. J. Manuf. Syst. 57, 287–297 (2020). https://doi.org/10.1016/j.jmsy.2020.10.004
24. Tan, X.J., Mustafa, N., Mashor, M.Y., Ab Rahman, K.S.: An improved initialization based histogram of k-mean clustering algorithm for hyperchromatic nuclei segmentation in breast carcinoma histopathological images. (2019). https://doi.org/10.1007/978-981-13-6447-1_67
25. Mashor, M.Y.: Hybrid training algorithm for RBF network. Int. J. Comput. Internet Manag. 8, 50–65 (2000). https://doi.org/10.1080/00207179208934272
26. Schwertman, N.C., Owens, M.A., Adnan, R.: A simple more general boxplot method for identifying outliers. Comput. Stat. Data Anal. 47, 165–174 (2004). https://doi.org/10.1016/j.csda.2003.10.012