The International Journal of Advanced Manufacturing Technology (2023) 129:1547–1565
https://doi.org/10.1007/s00170-023-12287-z

ORIGINAL ARTICLE

Defect inspection in semiconductor images using FAST-MCD method and neural network

Jinkyu Yu[1] · Songhee Han[2] · Chang-Ock Lee[1]

Received: 8 March 2023 / Accepted: 31 August 2023 / Published online: 3 October 2023

Abstract
Most defect inspection methods used in semiconductor manufacturing require design layout or golden die images. Unlike methods that require such additional information, this paper presents a method for automatic inspection of defects in semiconductor images with a single image. First, we devise a method to classify images into four types: flat, linear, patterned, and complex using a cosine similarity. For linear and patterned images, we obtain defect-free images that retain the structure. A flat image is then obtained by subtracting the defect-free image from the input image. The FAST-MCD method then estimates the parameters of the inlier distribution of the flat image and uses them to detect defects. A segmentation neural network is used to detect defects in complex images. Unlike conventional methods that only work on a specific structure, our method classifies structures and finds defects in each structure. We use **16** defective images in our experiments, where our method detects all **16** defective images, while the conventional methods detect fewer defective images.

Keywords Defect inspection · Semiconductor image · FAST-MCD method · Neural network · Structure classification

1 Introduction

Speed, accuracy, and repeatability are required for defect inspection in semiconductor manufacturing. These requirements are becoming more stringent as the fabrication process has become more sophisticated in recent years. Defects in semiconductors affect the appearance, functionality, efficiency, and stability of devices. Manual inspection is subjective, and its precision depends on the inspector's condition, such as eye fatigue. Therefore, automatic optical inspection continues to improve to detect defects and increase yield in semiconductor manufacturing [12, 20]. Non-destructive visual inspection is critical in the industry to assist or replace subjective and repetitive manual inspection processes.

✉ Chang-Ock Lee
colee@kaist.edu

Jinkyu Yu
hortensia@kaist.ac.kr

Songhee Han
shee33.han@samsung.com

[1] Department of Mathematical Sciences, KAIST, Daejeon 34141, Korea

[2] Samsung Electronics, Yongin, Gyeonggi-do 17113, Korea

Defect inspection methods in semiconductor images can be classified into four types: model-based algorithm, neural network, Die-to-Database (D2DB) method, and Die-to-Die (D2D) method. Many algorithms have been developed to find anomalies in various images, e.g., phase only transform [1], principal component analysis [6], self-similarity [9], discrete cosine transform [33], independent component analysis [34], and A-contrario detection [17]. Most of these methods assumed a specific structure, such as flat or patterned, and were proposed to fit the structure. Recently, as neural networks have shown good performances in imaging problems, many methods using neural networks have been proposed [10, 55, 57, 60]. However, unlike many neural network problems, semiconductor images do not have benchmark data. Neural networks also have the disadvantage in that they are ambiguous to interpret results. Most methods for finding defects, especially in semiconductor manufacturing, are D2DB methods or D2D methods. Traditional D2DB methods [31, 36, 48] require preprocessing to align the database and an image. Inspection is then performed using the aligned database. There have been attempts to apply neural network [30, 39, 42] to the D2DB method, but an alignment step is still needed. Traditional D2D methods [21, 50, 64] use golden die images to make a difference image to find defects. Like the D2DB methods, neural network [3] is applied to the

D2D method, but golden die images are still needed to train the network. In addition, the D2DB and D2D methods are very sensitive to the alignment process. Multiple scanning image method [37] is possible if other sensor images are available.

In this paper, we present a method for inspecting defects that removes the ambiguity of neural networks as much as possible with one image without additional information. First, we present a method for classifying images into four types: flat, linear, patterned, and complex using a cosine similarity. A flat image is an image in which the background excluding defects is almost constant with Gaussian noise. A linear image is an image that is shift invariant in a certain direction. If a particular shape appears repeatedly with a certain period, it is a patterned image. A complex image is the one in which all three of the above characteristics are absent. For linear images and patterned images, we reconstruct defect-free images. Then, a flat image is created by subtracting defect-free image from the input image. Under the assumption of Gaussian noise, a histogram of a flat image follows a normal distribution. Defects are considered as outlier in the distribution and could affect the parameters, so we need to minimize the influence of defects by estimating the inlier distribution. This distribution can be estimated using the minimum covariance determinant (MCD) method [46]. The MCD method is a highly robust estimator for multivariate location and scatter. The MCD method finds the part of the data with the minimum covariance determinant consisting only of inlier data. Then, defects are found by threshold from the inlier distribution. We use a segmentation neural network for complex images. Figure 1 shows the typical four images and their defect regions.

The rest of this paper is organized as follows: In Sect. 2, we briefly review the literature of model-based algorithm for the single image inspection and explain basic tools. Section 3 describes the classifier that classifies images into four types and two ways to remove the structure for linear and patterned images. A segmentation neural network is also described in Sect. 3. There are experimental results for several data sets in Sect. 4. We conclude this paper with remarks in Sect. 5.

2 Preliminaries

2.1 Previous works

This section briefly introduces conventional methods for finding anomalies in a single image $u \in \mathbb{R}^{h\times w}$. As mentioned in Sect. 1, most of these methods work on specific structures such as flat or patterned.

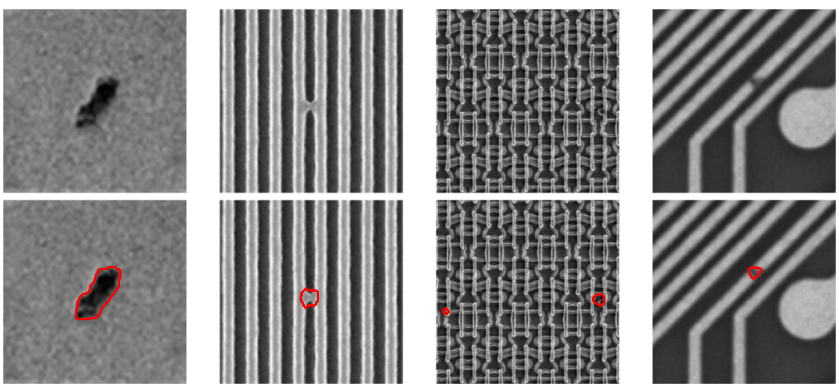
2.1.1 Works for flat images

For flat images, there are several ways to find defects. The simplest method [6] uses the mean and standard deviation of the image. Let μ_u and σ_u be the mean and standard deviation of the image u, respectively. Then, the binary image $\tilde{y}$ representing defects is obtained using the threshold as follows:

$$\tilde{y} = \begin{cases} 1 & \text{if } |u-\mu_u| > c\sigma_u\,, \\ 0 & \text{if } |u-\mu_u| \le c\sigma_u\,. \end{cases}$$

The constant c is usually assigned a value between 3 and 5. Because the mean and standard deviation of the entire image are used, the results will vary if the image has a large defect. Therefore, a method for estimating the inlier distribution without being affected by defects is needed, which is presented in Sect. 2.2.2.

Another method is to divide the image into two regions using linear discriminant analysis (LDA) [50]. This method finds the optimal threshold t^*. Let $C_0(t) = \{u_{ij} \mid u_{ij} < t\}$ and $C_1(t) = \{u_{ij} \mid u_{ij} \ge t\}$ where u_{ij} is the value of u at the pixel (i, j). Let $\mu_i(t)$ and $\sigma_i^2(t)$ be the mean and variance



(a) Flat [37] (b) Linear [11] (c) Patterned [67] (d) Complex [51]

Fig. 1 Typical four types of semiconductor images and defect regions. (First and third images have permission from IOP Science and IEEE, respectively)

for the set $C_i(t)$ for $i = 0, 1$. The farther apart the means of the two sets, $C_0(t)$ and $C_1(t)$, the smaller the variance of the sets and the better the division. That is, the object function $J(t)$ can be written as

$$J(t) = \frac{|\mu_0(t) - \mu_1(t)|^2}{\sigma_0^2(t) + \sigma_1^2(t)} .$$

Then, we find the value t^* which maximizes the object function $J(t)$. Since this method always divides the image into two sets, it is not suitable for applying to defect-free images.

2.1.2 Works for linear images

There is a method to find defects in directional textured images [6]. This method uses the principal component analysis (PCA) to separate defects and background structures. To apply PCA, first the average of the column vectors of the image matrix is set to zero. The normalized eigenvalues are then used to find the directional textured background. If the normalized eigenvalue is greater than 1, the principal component represents defect-free background. Otherwise, the principal component represents defects. This method is invariant to horizontal or vertical shifting, rotation, and illumination changes of the directional texture. However, in the case of an image with a vertical linear structure as shown in Fig. 1b, the linear structure is removed in the process of setting the average of the column vectors to zero, so the defect is judged to be the main structure. Therefore, this PCA-based method is not suitable for images with vertical linear structures.

2.1.3 Works for patterned images

There are several methods to find anomalies in patterned images. The simplest method [9] is to first choose an appropriate patch size for each image. Then, it checks how often the patch centered on each pixel appears in the image. For each patch q, we find the k most similar patches q_i for $i = 1, \ldots, k$. Then, the reconstructed patch $\hat{q}$ is obtained by averaging $\{q_i\}$ as

$$\hat{q} = \frac{1}{\sum_{i=1}^{k} \exp\left(-\frac{\|q-q_i\|}{a^2}\right)} \sum_{i=1}^{k} \exp\left(-\frac{\|q-q_i\|}{a^2}\right) q_i ,$$

where a is a constant. This method is highly sensitive to the patch size.

Another method finds the lattice vectors which generate the pattern. If the lattice vectors generating the pattern are known, it is easy to remove the pattern. Traditional methods for detecting pattern repetition are to use autocorrelation [32] or fast Fourier transform (FFT) [53]. The autocorrelation of an image u, denoted by $ac \in \mathbb{R}^{h\times w}$, is defined as

$$ac_{xy} = \sum_{ij} u_{ij} u_{i+x, j+y} .$$

This autocorrelation ac has the largest value at the origin. Therefore, global thresholding is not suitable for finding peak points. There is a method [32] to find peak points, which uses local maxima of smoothed ac as the peak points. The basic idea of the method using the FFT is that the frequency with the maximum value of FFT is related to the number of repetitions. When the number of repetitions of the pattern is large enough, for example, if (x^*, y^*) is the index at which the FFT has the maximum value, the periods in the x and y directions can be approximated by h/x^* and w/y^*, respectively, so that $(h/x^*, 0)$ and $(0, w/y^*)$ can be used as lattice vectors. However, if the image has fewer repetitions of the pattern, (i.e., small x^*, y^*), then h/x^* and w/y^* cannot be said to be approximations of the periods. Therefore, this FFT-based method is not suitable for images with few repeated patterns.

2.2 Basic tools

2.2.1 Cosine similarity

In this section, we briefly review the cosine similarity which is widely used in image problems such as face verification and clustering [24, 40, 58, 62]. In this paper, it will be used to classify images and find periods in the case of patterned images.

Let $u \in \mathbb{R}^{h\times w}$ be an image, $K \in \mathbb{R}^{k\times l}$ be a kernel, and $\mathbb{1}_{k\times l}$ be a matrix of size $k \times l$ with all entries equal to 1. Then, the cosine similarity $CS \in \mathbb{R}^{h\times w}$ is calculated as

$$CS = \frac{K * u}{\sqrt{\mathbb{1}_{k\times l} * u^2}\|K\|} ,$$

where $*$ is the convolution and $\|\cdot\|$ is the Frobenius norm. When computing the convolution, we use reflection padding on u to get a cosine similarity CS of the same size as the image u. Note that the square, square root, and division operations are calculated on entry-by-entry. A large entry in CS means that the image u has a kernel-like structure near the same indices.

2.2.2 Minimum covariance determinant (MCD) method

This section describes a statistical technique for estimating inlier distribution. Since the average and covariance matrix

are extremely sensitive to outliers, a robust estimator is essential and MCD method [46] is one of the most widely used estimators [22, 23]. In this paper, for defective images, the MCD method will be used to estimate the inlier distribution without being affected by defects. For the sake of completeness, we introduce the MCD method.

Let $\{x_i\}_{i=1}^n$ be a finite sample of data in $\mathbb{R}^d$ with a distribution F, where d is the number of random variables. The MCD is determined by choosing a subset $S = \{x_{i_j}\}_{j=1}^s$ of size $n/2 \le s \le n$, which minimizes the determinant of covariance matrix computed from the subset S. Then, $\alpha = 1 - s/n$ is a portion of samples that is not contained in the subset S. Assume that the distribution F has a density of the form

$$f_{\mu,\Sigma}(x) = \frac{g((x-\mu)^T \Sigma^{-1}(x-\mu))}{\sqrt{\det(\Sigma)}},$$

where $g : \mathbb{R}_+ \to \mathbb{R}_+$ is a non-increasing function. Then, F is an elliptically symmetric, unimodal distribution. From the average μ_S and covariance matrix Σ_S of the MCD-solution S, the average μ and covariance matrix Σ of inlier distribution can be obtained by

$$\mu = \mu_S \quad \text{and} \quad \Sigma = c_\alpha \Sigma_S \tag{1}$$

with the consistency factor c_α as

$$c_\alpha = (1-\alpha)\left(\frac{\pi^{d/2}}{\Gamma(d/2+1)} \int_0^{q_\alpha} r^{d+1} g(r^2) dr\right)^{-1}, \tag{2}$$

where $q_\alpha > 0$ satisfies

$$\frac{2\pi^{d/2}}{\Gamma(d/2)} \int_0^{q_\alpha} r^{d-1} g(r^2) dr = 1 - \alpha\,.$$

Here, Γ denotes the gamma function (see [5] for more details). However, calculating all covariance determinants of $\binom{n}{s}$ subsets is too difficult.

2.2.3 FAST-MCD method

In this section, we introduce the FAST-MCD method [47] to quickly find the MCD-solution S. First, we consider the Mahalanobis distance which measures how much each sample point x_i deviates. For a given average vector μ and covariance matrix Σ, the Mahalanobis distance of a point x is defined as

$$d_M(x, \mu, \Sigma) = \sqrt{(x-\mu)^T \Sigma^{-1}(x-\mu)}\,.$$

The main part of the FAST-MCD method is called the concentration step (C-step), which is described in Algorithm 1. Through the C-step, it holds that $\det(\Sigma_k) \ge \det(\Sigma_{k+1})$. Since the sequence $\{\det(\Sigma_k)\}$ is monotone and bounded below, it converges. However, there is no guarantee that $\det(\Sigma_k)$ converges to $\det(\Sigma_S)$ for the MCD-solution S. Therefore, the FAST-MCD method has different limits for different choices of S_1 (see [47] for more details). Despite the lack of theory for the convergence to $\det(\Sigma_S)$, the FAST-MCD method has been applied to various fields [2] and empirically proven to produce good results [61]. In Table 1, we show by example that the FAST-MCD method estimates the inlier distribution well.

Algorithm 1 C-step in the FAST-MCD method.

Let S_1 be an initial subset of size s.
Compute the mean vector μ_1 and covariance matrix Σ_1 for S_1.
while $\det(\Sigma_k) \neq 0$ and $\det(\Sigma_k) \neq \det(\Sigma_{k-1})$ **do**
  Compute the Mahalanobis distance $d_M(x_i, \mu_k, \Sigma_k)$ for $i = 1, \ldots, n$.
  S_{k+1}: the subset of s vectors selected in order of smallest Mahalanobis distance.
  Compute the mean vector μ_{k+1} and covariance matrix Σ_{k+1} for S_{k+1}.
end while

3 Methods

In this section, we introduce a method of inspecting defects in a single image. First, we propose a method using the cosine similarity to classify images into four types: flat, linear, patterned, and complex. For linear and patterned images, we present how to reconstruct defect-free image. A flat image with the structure removed is obtained by subtracting defect-free image from input image. Then, we use the FAST-MCD method to detect defects in flat images. Finally, a segmentation neural network detects defects in complex images. Figure 2 shows the flow chart of our whole algorithm.

Table 1 Results of inlier distribution estimation using the FAST-MCD method for the image in Fig. 1a

$1-\alpha$	0.5	0.55	0.6	0.65	0.7	0.75	0.8	1.0	inlier
μ_S	0.484	0.483	0.483	0.482	0.482	0.482	0.482	0.472	0.483
σ_S	0.012	0.014	0.015	0.016	0.018	0.020	0.022	0.071	0.035
$\sqrt{c_\alpha}\sigma_S$	0.032	0.032	0.032	0.032	0.032	0.033	0.033		

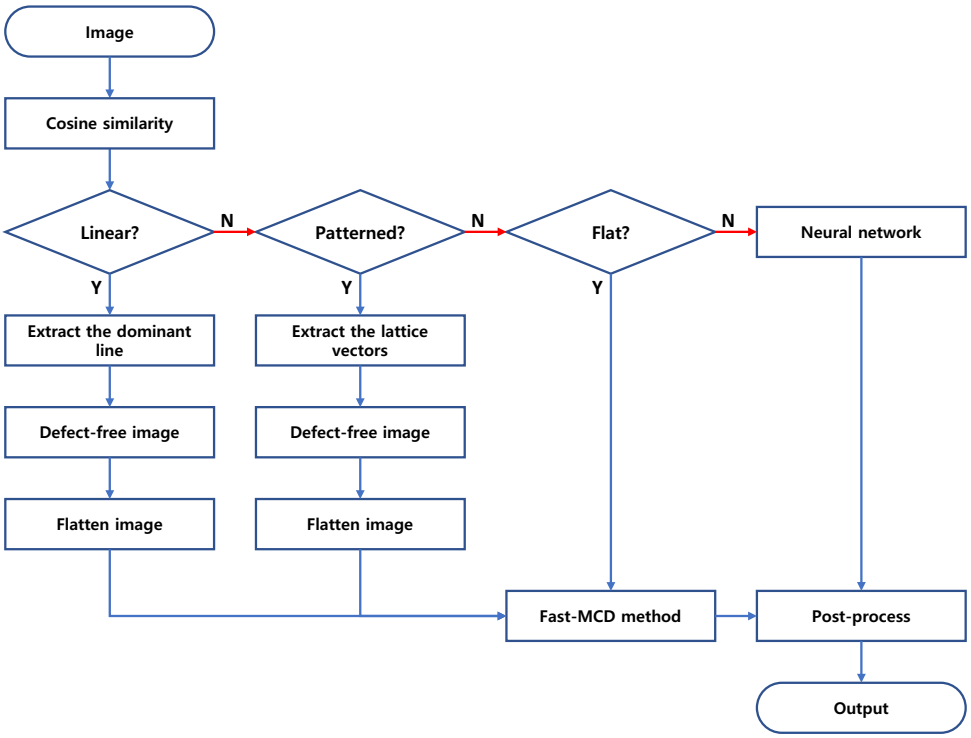


Fig. 2 The flow chart of our algorithm

3.1 Image classification

For convenience, we assume a gray scale image. First, we divide an image $u \in \mathbb{R}^{h\times w}$ into $M \times N$ subimages. Then, we calculate the cosine similarity CS_i with the kernel $K_i = i^{th}$ subimage for $i = 1, \ldots, MN$. A large entry in CS_i means that the image u has a K_i-like structure near the same indices. We find the region $P_i = \{(x, y) \mid CS_i(x, y) > t_i \text{ for } x = 1, \ldots, h \text{ and } y = 1, \ldots, w\}$ for a threshold t_i and call it the repeated region. The CS_i's of the flat image and the pattern image are different. Since CS_i depends on the structure of the image, the threshold t_i to obtain P_i must be set adaptively for the image. Therefore, t_i is selected a value between the maximum value 1 and the minimum value of CS_i, and the results for various ratios between the maximum and minimum values are in Appendix A.1. Here, we use $t_i = 0.85 + 0.15 \min CS_i$. For each P_i, we consider the centroid of K_i. Then, we overlap the repeated regions $\{P_i\}$ based on the centroid of each K_i and call it the overall repeated region $P \subset [1, \overline{h}] \times [1, \overline{w}]$ where $\overline{h} = 2h - \lceil \frac{h}{M} \rceil$ and $\overline{w} = 2w - \lceil \frac{w}{N} \rceil$. If M or N is so large that the kernel K_i becomes smaller than the repeating pattern, the cosine similarity CS_i cannot find the pattern. In Appendix A.2, there is a one dimensional example to show that a kernel with a small size cannot find the pattern.

In Appendix A.1, we compute the moment tensor I of the connected region R containing the center of the domain $[1, \overline{h}] \times [1, \overline{w}]$. If an image has a linear structure, P has a long connected region R, with the large axis ratio defined as the ratio of large and small eigenvalues of I, in the dominant direction defined as the direction of the major eigenvector. If the axis ratio is greater than 25, we determine that the image is linear as discussed in Appendix A.1.

For an image to have a pattern, it must be repeated at least three times. If the pattern is repeated three times in one direction, then P has five high value regions in a straight line. If the pattern is repeated three times in a triangular shape, then P has seven high value regions and can form three straight lines, each containing three high value regions. For each high value region, we can extract a peak point as the centroid of the region. That is, P of a patterned image has at least five peak points on one or two straight lines.

For a Gaussian noised flat image, the histogram of d_M^2 for the MCD-solution S follows the chi-squared distribution. Jensen-Shannon divergence (JSD) is commonly used to measure the distance between two distributions x and y [14]:

$$\mathrm{JSD}(x, y) = 0.5 \sum_i x_i \log\left(\frac{2x_i}{x_i + y_i}\right) + 0.5 \sum_i y_i \log\left(\frac{2y_i}{x_i + y_i}\right).$$

Note that this $\mathrm{JSD}(x, y)$ is bounded by $\log 2$. If the JSD between the histogram of d_M^2 for the MCD-solution S and the chi-squared distribution is less than $5 \log 2/100$, we determine that the image is flat (see Appendix A.3 for more details).

Now, images can be classified by using the information of the repeated region as follows:

1. Linear image: Axis ratio of the connected component $R \geq 25$.
2. Patterned image: At least 5 peak points in P forming one or two straight lines.
3. Flat image: JSD between the histogram of d_M^2 for the MCD-solution S and the chi-squared distribution $\leq 5 \log 2/100$.

Figure 3 shows the cosine similarities and overall repeated regions for four types of images. The axis ratio of R is displayed at the top of P. Note that the second row, which has a linear structure, shows a higher axis ratio than others. For the third row with a patterned structure, the yellow line in the last column represents the line passing through five or more peak points including the center of P. Figure 4 shows the results of our image classification method.

3.2 Defect inspection by image type

As mentioned in Sect. 1, for linear and patterned images, we present two methods to reconstruct defect-free images. The difference between the defect-free image and the input image becomes a flat image containing defects. For flat images, we use the FAST-MCD method to estimate the inlier distribution and find the defects. The segmentation neural network is applied to inspect complex images.

3.2.1 Removal of structure in linear images

An image in which the axis ratio of the connected component R is greater than 25 is judged to have a linear structure. For linear images, we compute the direction of the major

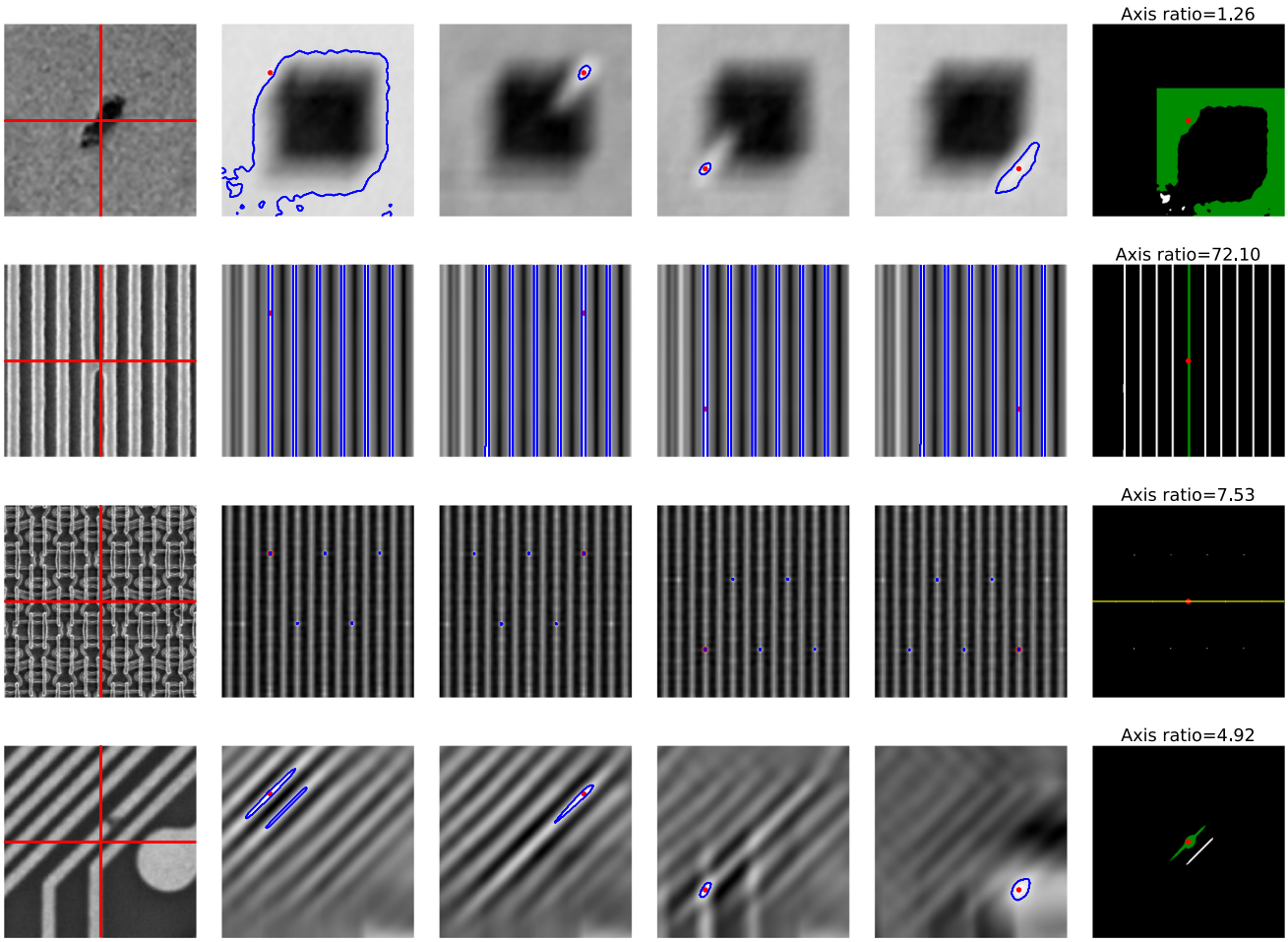


Fig. 3 Cosine similarities and repeated regions for four types of images. First column shows the input images with 2×2 partitions. Columns 2–5 show each cosine similarity CS_i, and blue contours show the repeated region P_i. The centroids of the subimages used as kernels are indicated by red dots. Last column shows the overall repeated region P, and red dot indicates the center of the domain. The green region shows the connected region R containing the center of the domain

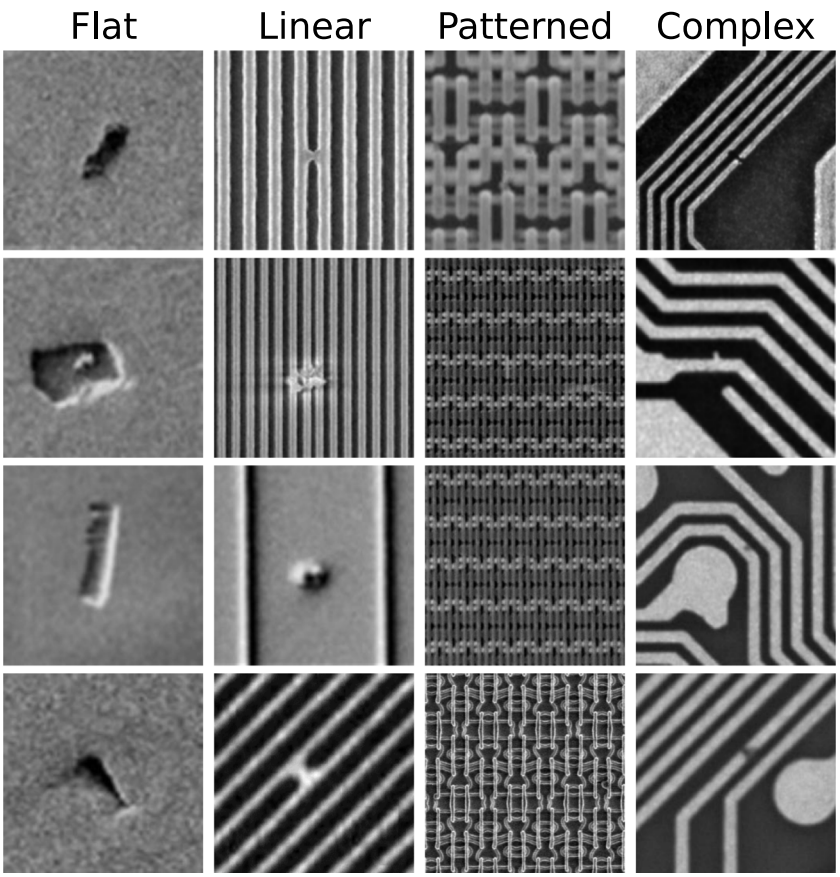


Fig. 4 Examples of images for four types: flat [37], linear [11, 37, 52], patterned [41, 63, 67], and complex [18, 50, 51]. (All flat images and third linear image have permission from IOP Science, and first and second complex images have permission from Elsevier and Springer, respectively)

eigenvector of the moment tensor I to get the dominant direction. The defect-free image can be obtained by taking the median of average intensity of the dominant line and the intensities at both ends along the dominant line. If the line has no defects, the average value is chosen as the median. Otherwise, one of the two end values is chosen as the median. Then, we can obtain a flat image by subtracting the defect-free image from the input image. Finding defects in the flat image can be done as in Sect. 3.2.3. Figure 5b shows a long connected region R colored by green. It has an axis ratio $77.039 > 25$ and a vertical major eigenvector. Hence, it is classified as a linear image. Figure 5d shows the defect-free image obtained using the dominant lines with the same direction as the major eigenvector. In Fig. 5e, the defects are prominent in the flattened image where the linear structure is removed.

3.2.2 Removal of structure in patterned images

If an image is not linear, we consider a straight line passing through the center of overall repeated region P. As mentioned in Sect. 3.1, if there exist one or two straight lines passing through at least five peak points, the image is judged to have a patterned structure. For a patterned image, we extract two lattice vectors $\{w_1, w_2\}$ (see Appendix A.4 for details on how to extract the lattice vectors). Depending on the pattern of the image, one lattice vector can be the zero vector. Using the two lattice vectors $\{w_1, w_2\}$, we create lattice points (see Algorithm 2). We overlap the image $u \in \mathbb{R}^{h\times w}$ so that the top left of the image is located at each lattice point. After taking the average values of the overlapped images, a defect-free image can be obtained by cropping the averaged image of size $h \times w$ in the middle. Since the defects do not appear repeatedly, the average image gives a defect-free

Algorithm 2 Lattice points generation.

Assume $w_1 \neq 0$ and image $u \in \mathbb{R}^{h\times w}$.
if $\|w_2\| \neq 0$ **then**
  $n_1 = \left\lceil \frac{\sqrt{h^2+w^2}}{\|w_1\|} \right\rceil$, and $n_2 = \left\lceil \frac{\sqrt{h^2+w^2}}{\|w_2\|} \right\rceil$, with the notation $\lceil x \rceil$ for the smallest integer greater than x.
else[$\|w_2\| = 0$]
  $n_2 = 0$
end if
for $i = -n_1 : n_1$ **do**
  for $j = -n_2 : n_2$ **do**
    $z = iw_1 + jw_2 + (h+1, w+1)$
    if $z \in [1, 2h] \times [1, 2w]$ **then**
      Select z as a lattice point.
    end if
  end for
end for

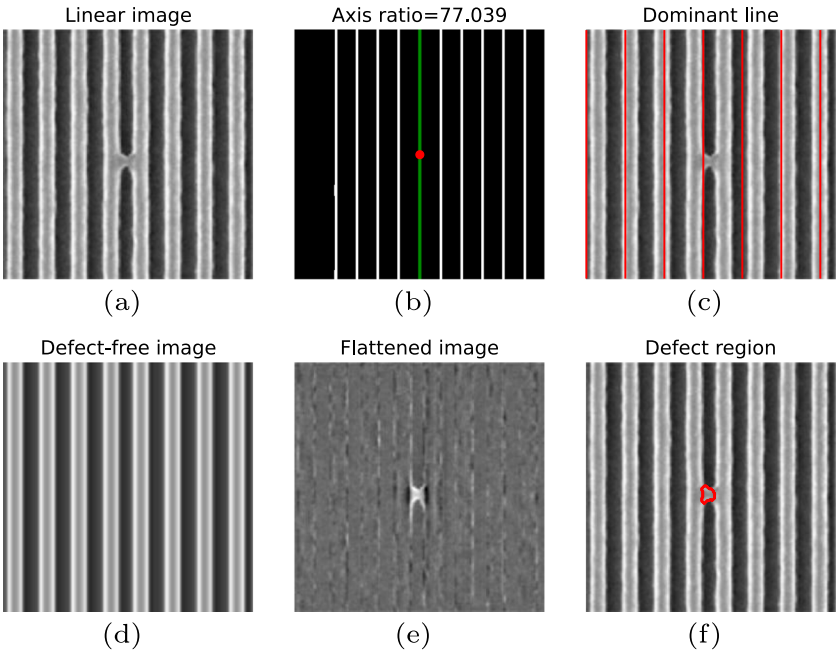


Fig. 5 Inspection process of a linear image. **a** Input image, **b** overall repeated region P. The red dot indicates the center of P, and the green region shows the connected region R containing the center of the domain. The red lines in **c** show the dominant lines. The defect-free image is shown in **d**. **e** shows the flattened image with linear structure removed. The defects detected by the FAST-MCD method are shown in **f**

image. Then, a flat image can be obtained by subtracting the defect-free image from the input image. Finding defects in the flat image can be done as in Sect. 3.2.3. Figure 6 shows a graphical description of lattice point generation. We overlap the image $u \in \mathbb{R}^{h\times w}$ so that the top left of the image is located at each lattice point. The figure shows when the top left of the input image is placed on the orange dot. After the overlapping process, the averaged image is obtained. Then, we can obtain a defect-free image by cropping the green box. Figure 7c shows the lattice points generated from Algorithm 1. The lattice points appear regularly in the upper right corner of each patterned circle. In Fig. 7e, the defects are prominent in the flattened image where the patterned structure is removed.

3.2.3 Detecting defects in flat images

For the image which is not linear or patterned, we check whether the image is flat. To do this, we compute the JSD between the histogram $h_S(x)$ of d_M^2 for the MCD-solution S and the probability density function of chi-squared distribution. If the JSD is less than $5\log 2/100$, then the image is judged to have a flat background. This section describes how to find defects in flat images using the FAST-MCD method in Sect. 2.2.3.

A histogram of a flat image with Gaussian noise follows a normal distribution $N(\mu, \Sigma)$ with $g(r^2) = \frac{e^{-0.5r^2}}{(2\pi)^{d/2}}$ having a negative derivative. Therefore, the consistency factor c_α in (2) can be used to estimate the inlier distribution. Table 1 shows the estimate results of the gray scale flat image in Fig. 1a with several $(1-\alpha)$ from 0.5 to 0.8. When $1-\alpha = 1.0$ meaning that all pixel data is used, the mean and standard deviation are 0.472 and 0.071, respectively. The standard deviation σ_S depends on α, but $\sqrt{c_\alpha}\sigma_S$, which is estimated as an inlier standard deviation, gives similar values to the standard deviation of inlier distribution even if α is different. These results experimentally show that the aforementioned relationship (1) holds between the inlier distribution and MCD-solution S distribution is established. Hence, we use $\alpha = 0.25$ which is a default value in [47]

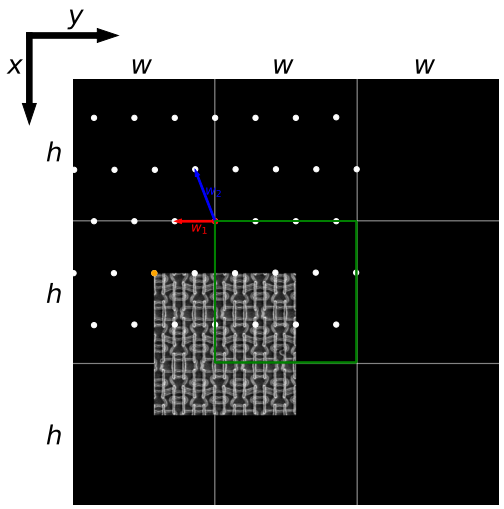


Fig. 6 Graphical description of lattice point generation using two lattice vectors w_1 and w_2 which are denoted as red and blue arrows, respectively. The white dots are the lattice points in $[1, 2h]\times[1, 2w]$ generated from these two lattice vectors

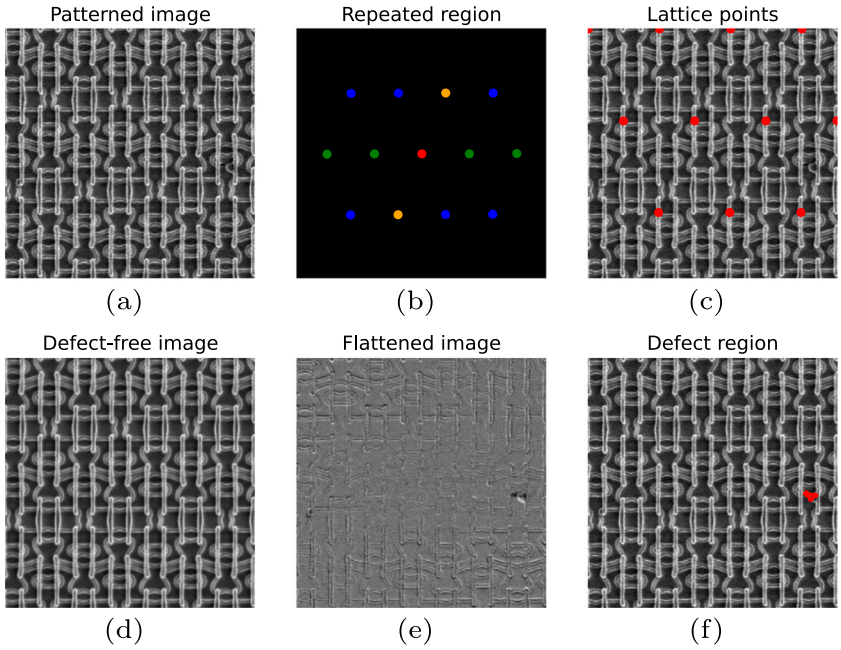


Fig. 7 Inspection process of a patterned image. **a** Input image. **b** Peak points of repeated region. The green dots with red dot (center) give first lattice vector. The orange dots with red dot give second lattice vector. The lattice points generated from the two lattice vectors are indicated as red dots in **c**. A defect-free image is shown in **d**. **e** shows the flattened image with patterned structure removed. The defects detected by the FAST-MCD method are shown in **f**

for estimating the inlier distribution. Since we assume the Gaussian noise, the square of the Mahalanobis distance, $d_M^2(x_i, \mu, \Sigma)$, of inlier part follows a chi-squared distribution. We find defects with a threshold $d_M^2(x_i, \mu, \Sigma) > \chi^2_{1,p}$. Here, p can be adjusted according to the level of defect detection. For example, $p = 0.99$ means that approximately 1% of the area is detected in the defect-free flat image. For our purpose, a defect-free image should be judged to be defect-free. Therefore, we use the threshold for detecting defects as $p = 1 - 1/(4hw)$ for an $h \times w$ image. Here, the use of $p = 1 - 1/(4hw)$ means that about 0.25 pixel is detected in a defect-free flat image, regardless of the size of the image. From now on, we will use $\alpha = 0.25$ and $p = 1-1/(4hw)$ for gray scale images. Figure 8b shows that the MCD-solution S contains no defects. Figure 8d shows that the histogram of d_M^2 for S is similar to the chi-squared distribution. (i.e., MCD-solution S follows a Gaussian distribution).

3.2.4 Detecting defects in complex images

An image that is not judged flat, linear, and patterned is called complex and inspected for defects through a segmentation neural network. Let $f_\theta : \mathbb{R}^{h\times w} \to [0, 1]^{h\times w}$ be a segmentation network with parameters θ, which gives a probability output. Let $y = f_\theta(x)$ be the probability output of an input image x. Let $\hat{y}$ be a ground-truth (label) segmentation region of the input x: $\hat{y}_{ij} = 1$ if (i, j) belongs to the target region and 0 otherwise.

The dice score (DS) is used to measure the performance of segmentation problems, which is defined by

$$DS(A, B) = \frac{2 \mid A \cap B \mid}{\mid A \mid + \mid B \mid} = \frac{2\sum_{ij} A_{ij} B_{ij}}{\sum_{ij}(A_{ij} + B_{ij})}.$$

It takes maximum value 1 when $A = B$. Similarly, $DS(1 - A, 1 - B)$ gives the performance of background

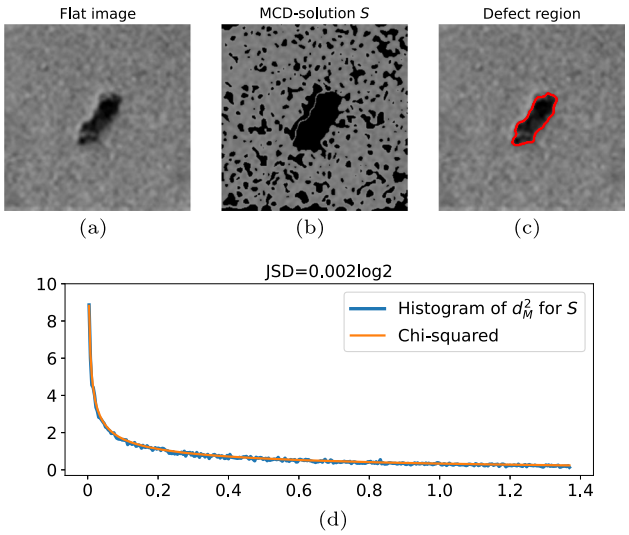


Fig. 8 Inspection process of a flat image. **a** Input image. **b** MCD-solution S. The defects detected by the FAST-MCD method are shown in **c**. The histogram of d_M^2 for the MCD-solution S and the probability density function of chi-squared distribution are shown in **d**

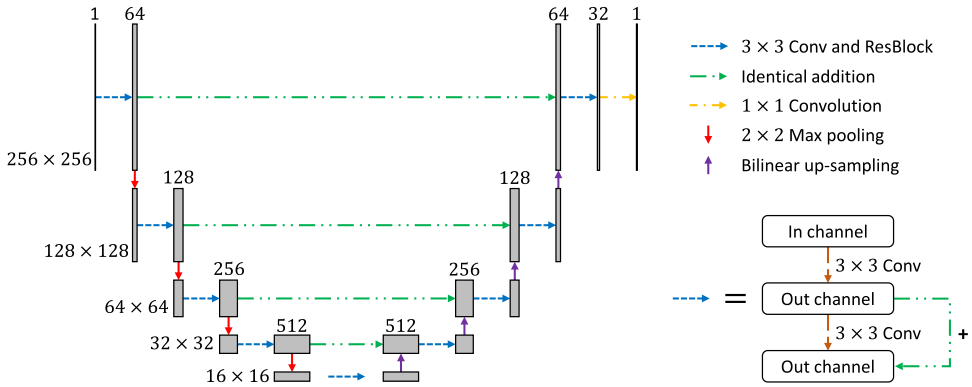


Fig. 9 Segmentation network architecture; it has a U-Net structure with a ResBlock form in a skip connection

segmentation. The generalized dice score (GDS) given by

$$GDS(A, B) = 2\frac{w_D \sum_{ij} A_{ij}B_{ij} + w_B \sum_{ij}(1 - A_{ij})(1 - B_{ij})}{w_D \sum_{ij}(A_{ij} + B_{ij}) + w_B \sum_{ij}(2 - A_{ij} - B_{ij})} \quad (3)$$

is used to evaluate multiple class segmentation [8], and it can be used to measure the tiny segmentation. It reduces the well-known correlation between the dice overlap and region size. From this GDS, the generalized dice loss is widely used in small segmentation problems in the form [54]:

$$L_{GD}(y, \hat{y}) = 1 - GDS(y, \hat{y}) .$$

Originally, the weights w_D and w_B are determined by the ratio of the total training dataset, like weighted cross entropy loss, thus the same weights w_D and w_B are used for all images. But there is a difference between cross entropy loss and dice loss. Since the cross entropy loss is calculated for each pixel, weights can be given using the number (area) of pixels of each class in the total training dataset. On the other hand, as the dice loss is calculated for each image, it is not appropriate to use fixed weights for dataset with various sizes of class areas. Our training dataset contains images with various defect sizes, such as $0.0002 \le \frac{1}{hw}\sum_{ij} \hat{y}_{ij} \le 0.3889$. So, instead of using fixed weights, we use adaptive weights for each image

$$w_D = \left(\sum_{ij} \hat{y}_{ij}\right)^{-2} \text{ and } w_B = \left(\sum_{ij}(1 - \hat{y}_{ij})\right)^{-2} . \quad (4)$$

We also use the boundary loss [29]

$$L_B(y, \hat{y}) = \sum_{ij} \phi(\hat{y})_{ij}(y_{ij} - \hat{y}_{ij}) ,$$

where $\phi(\hat{y})$ is the signed distance function as

$$\phi(\hat{y})_{ij} = \begin{cases} -d((i, j), \partial\hat{y}) & \text{if } \hat{y}_{ij} = 1 , \\ d((i, j), \partial\hat{y}) & \text{if } \hat{y}_{ij} = 0 . \end{cases}$$

Here, $\partial\hat{y}$ is the boundary of $\hat{y}$. Then, our loss function is a weighted sum of these two losses [29]:

$$L(y, \hat{y}) = \lambda L_{GD}(y, \hat{y}) + (1 - \lambda)L_B(y, \hat{y}).$$

During training, the weight λ was initially set to 1 and decreased gradually to 0.5 at the end of training. The optimizer Adam is used to minimize the loss function with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and learning rate 0.001. The network architecture based on U-Net [45] with ResNet [19] is shown in Fig. 9.

Figure 10d shows that the histogram of d_M^2 for S and the chi-squared distribution are different. This means that some structure is present in the input image.

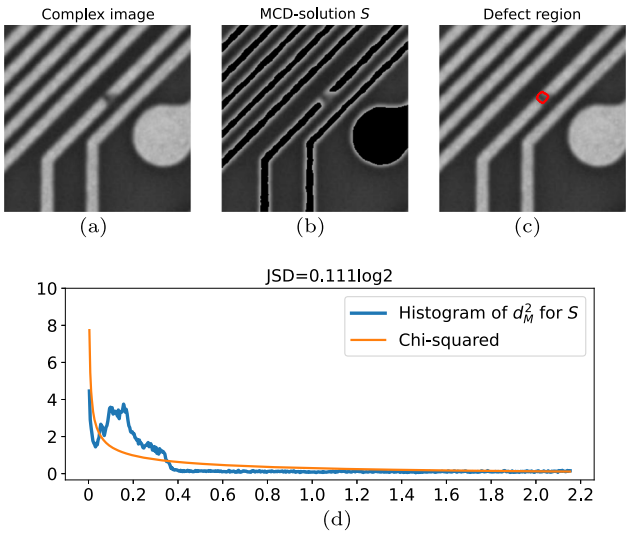


Fig. 10 Inspection process of a complex image. **a** Input image. **b** MCD-solution S. The defects detected by the segmentation network are shown in **c**. The histogram of d_M^2 for the MCD-solution S and the probability density function of chi-squared distribution are shown in **d**

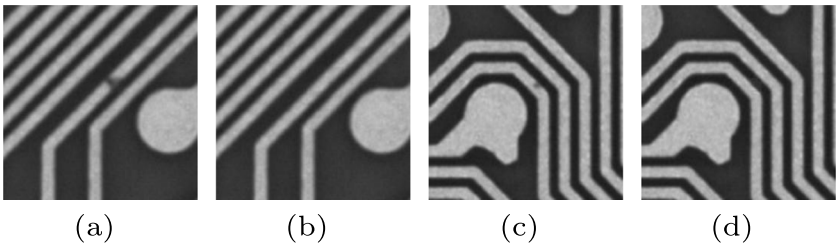


Fig. 11 Examples of defect-free images generated by graphical tools. **a** and **c** are defective images, and **b** and **d** are defect-free images

Remark 1 We might consider applying the segmentation network to all images. We observed that in most images, the neural network does not give as accurate results as our proposed mixed method. Also, we do not know what the neural network will do for new, untrained structural images.

3.3 Pre-processing and post-processing

Before applying the proposed method, we take a denoising step. In denoising methods based on isotropic diffusion, diffusion at the edge can smear the edge and remove the texture of the object. However, denoising methods based on anisotropic diffusion consider both spatial distance and intensity difference, thus preserving edges while reducing noise in non-edge regions. We use Perona-Malik anisotropic diffusion [44], the most popular model, to denoise the image:

$$u_t = \nabla \cdot \left(\frac{u}{1 + (|\nabla u|/\kappa)^2} \right),$$

where κ is a constant. We use $\kappa = 0.14$ and the time increment$= 0.1$ with five iterations.

If a defect appears on the boundary of the image, it is not known whether it is an actual defect or a part of the structure. Therefore, if a defect appears on the boundary of the image, it is excluded. For the remaining defects, we perform the morphological opening and closing to remove the dot defects (noise) and to connect nearby defects, respectively. We use structuring elements with a radius of 1 pixel for opening and a radius of 5 pixels for closing for 256×256 images. If there is a hole inside the defect, we fill it in during the post-process.

4 Experimental results

The proposed method was implemented to evaluate the performance of defect inspection for images with various structures. Since there is no testing database for semiconductor defects, we used 171 images in the literature [4, 11, 13, 15, 16, 18, 21, 25–28, 35, 37, 38, 41, 49–52, 56, 59, 63, 65, 67]; see [66] for specific image information. The size of images under inspection is 256×256. To train the network, we use the following data augmentation strategy:

- Normalization and the use of complement,
- Eight types of rotation and flipping.

For a 256×256 gray scale image u, we use the normalized image $\bar{u} = (u - \mu)/(10\sigma) + 0.5$ and the complementary image $\bar{u}^c = 1 - \bar{u}$, where μ and σ^2 are the average and the variance of u, respectively. Then, we rotate the image by $\pi/2, \pi$ and $3\pi/2$ radians and flip these vertically. The network is trained with $155 \times 2 \times 8 = 2,480$ defective images, and the validation dataset has 16 defective images and corresponding defect-free images. We created the defect-free images using the exemplar-based inpainting method [7] and manual processes with some graphical tools. Figure 11 shows the examples of defect-free images generated with graphical tools. We chose the parameters of the neural network with the highest $GDS(\tilde{y}, \hat{y})$ (3) for the 16 defective images in the validation dataset, where $\tilde{y}$ is a threshold result for $y \geq 0.5$. The weights are the same as in (4). The network is trained on 200 epochs with a batch size of 64.

Figure 12 shows the defect inspection results for defective images. We show the input images, ground truths, results of our method, neural network-1 [55], neural network-2 [57], and self-similarity method [9] in order from left to right column. We generated the ground truth with threshold and some manual process. Since we do not have a design layout, we cannot implement the D2DB inspection methods. The neural network-1 (NN-1) extracts features using $2D$ convolution with kernel size of 5×5, ReLU activation function, and 2×2 max pooling. The NN-1 method gives 32×32 outputs. To find the region in 256×256 input images, we use the bicubic interpolation. The second neural network-2 (NN-2) method which takes 512×512 images as inputs has a W-shape cascaded autoencoder architecture. Each U-shaped autoencoder uses a dilated $2D$ convolutions and ReLU activation function to extract features and has skip connections. To use this network architecture, we obtained 512×512 images using the bicubic interpolation again. We trained the parameters in the networks with our dataset. The number of parameters in the network is 15.4M, 61.9M, and 11.7M in order of NN-1, NN-2, and our method. NN-2, a full machine learning method, is inferior to our method despite using 5 times more parameters. For the self-similarity method, NFA$= 10^{-10}$ was used. The self-similarity method is known to work well for images with repetitive structures. However, if a specific structure

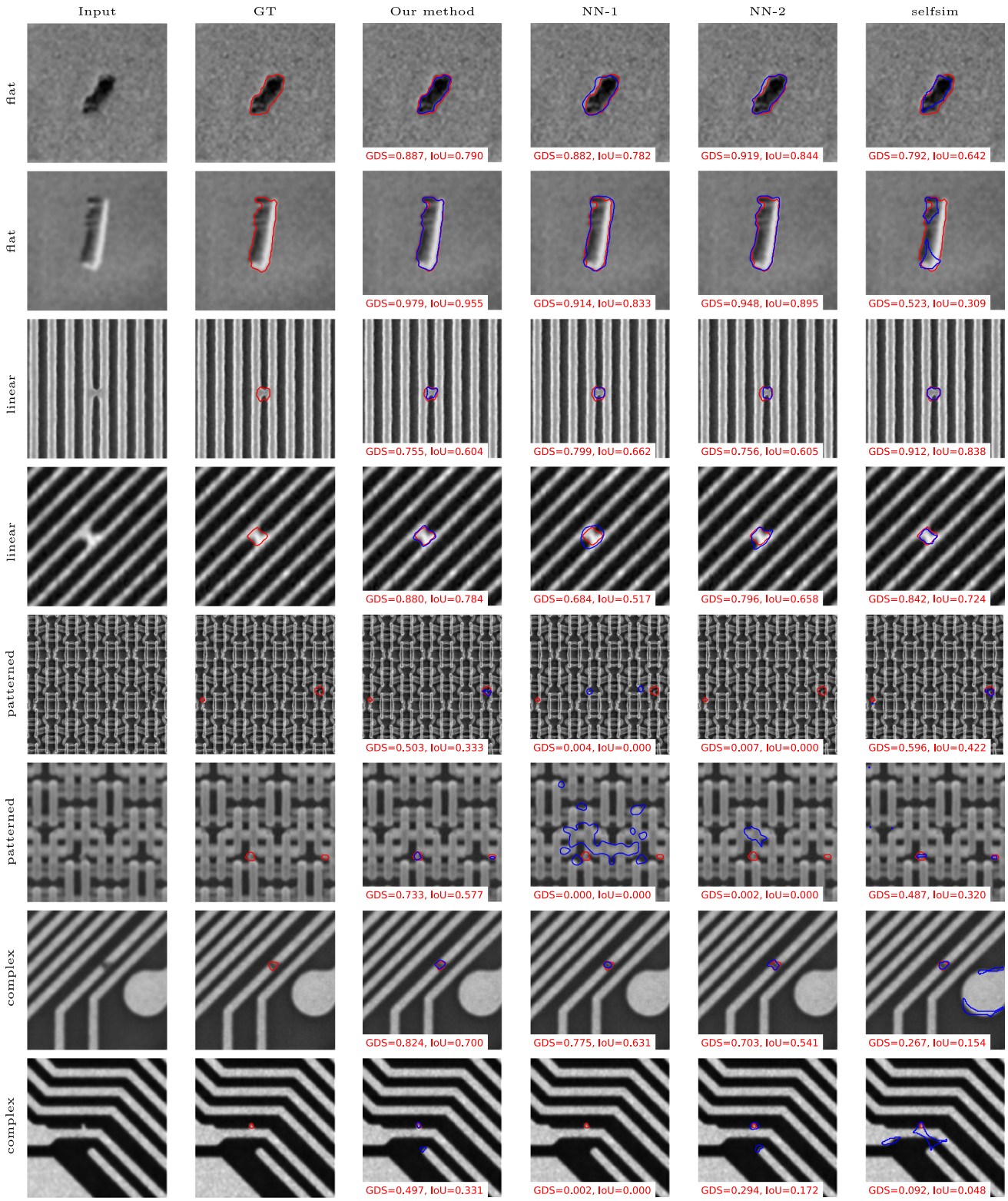


Fig. 12 Comparison of several methods for defective images: Our proposed method classifies images into four types: flat, linear, patterned, and complex. The red and blue contours show the ground truth and segmentation region, respectively

Table 2 Inspection results for validation dataset

	Our method	NN-1	NN-2	Selfsim
Mean of GDS	0.7353	0.4462	0.5106	0.5510
Std of GDS	0.1772	0.3936	0.3887	0.2553
Mean of IoU	0.5952	0.3653	0.4310	0.4122
Std of IoU	0.2166	0.3320	0.3555	0.2474
TP for 16 defective images	16	9	11	15
FP for 16 defect-free images	1	5	8	11

does not appear repeatedly even though it is not a defect, the self-similarity method judges it as a defect. Table 2 shows the mean of GDS, standard deviation (std) of GDS, mean of IoU, and std of IoU for 16 defective validation images. Here, IoU score is another metric mainly used in segmentation problems with the formula:

$$\mathrm{IoU}(\tilde{y}, \hat{y}) = \frac{|\,\tilde{y} \cap \hat{y}\,|}{|\,\tilde{y} \cup \hat{y}\,|},$$

where $|\cdot|$ denotes the number of pixels. We also provide the number of true positives for 16 defective validation images and the number of false positives for 16 defect-free validation images.

All model-based algorithms were implemented in MATLAB. All neural networks were implemented in Python with PyTorch [43], and all computations were performed on a cluster equipped with Intel Xeon Gold 6148 (2.4GHz, 40C) CPUs, NVIDIA RTX 3090 GPUs, and the operating system Ubuntu 18.04 64 bits.

5 Conclusion

Unlike the defect inspection methods mainly used in semiconductor manufacturing, we proposed a method of inspecting defects in a single image. Our method consists of classifying images and detecting defects according to each type. The cosine similarity, the moment tensor, and JSD were used to classify the image types. We proposed two methods for removing structures: one for a linear structure and the other for a repeated pattern. For the linear structured image, we found the dominant angle and removed the linear structure by subtracting the median of the average intensity and those at both ends on the dominant line. For the repeated patterned image, we selected two lattice vectors and made the lattice points. By overlapping the input image at each lattice points and averaging them, we obtained the defect-free reference image. From the difference image between the input image and the reference image, we found a flat image. The FAST-MCD method is used to detect defects in flat images. For an image with complex structure, we found the defects using a segmentation network.

Among the existing methods, most model-based inspection methods for a single image assume a special image structure (e.g., flat or patterned). Our method has the advantage of being more general in that it classifies the types of images and finds defects according to the types. Depending on the type of image with defect, it will be possible to know in which process the defect occurred. Machine learning methods have a disadvantage that it is difficult to explain the reason for the result. However, our method reduced the ambiguity of the results by classifying images into four types and then detecting defects in flat, linear, and patterned images by statistical methods and applying machine learning only to complex images.

Appendix A

A.1 Selection of the parameter t_i

Here, we experimentally show the role of the parameter t_i used in the image classification in Sect. 3.1. We obtain the repeated region $P_i = \{(x, y) \mid CS_i(x, y) > t_i \text{ for } x = 1, \ldots, h \text{ and } y = 1, \ldots, w\}$ using the threshold t_i. We overlap P_i based on the centroid of each kernel and call it the overall repeated region $P \subset [1, \overline{h}] \times [1, \overline{w}]$. Since the value of CS_i at the centroid of K_i is always 1, P_i contains the centroid of K_i. It means that P contains the center of domain $[1, \overline{h}] \times [1, \overline{w}]$. For R, the connected region containing the center of domain, we compute the moment tensor I as

$$I = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix},$$

where the components are defined as

$$I_{xx} = \int_R x^2 dS\,,\ I_{yy} = \int_R y^2 dS\,,\ I_{xy} = I_{yx} = \int_R xy dS\,.$$

For the moment tensor I, we compute eigenvalues and corresponding eigenvectors. If an image has a linear structure, P has a long connected region R, with the large axis ratio defined as the ratio of large and small eigenvalues, in the

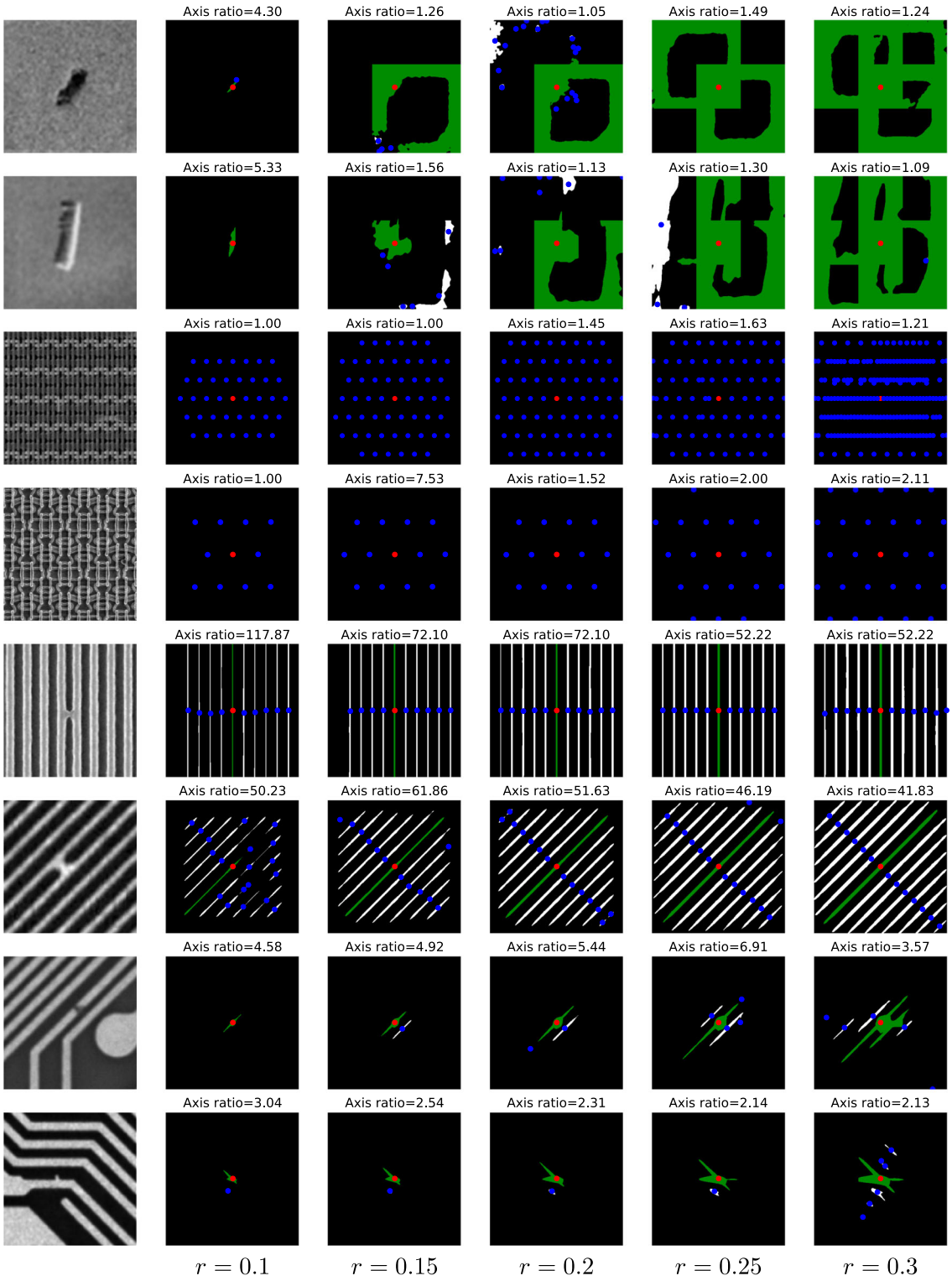


Fig. 13 Experiment results for various t_i. The first column shows the input images. Columns 2 through 6 show P for $r = 0.1$ to $r = 0.3$. The red dot indicates the center of domain, and green region shows the connected region R containing the center of domain. The blue dots indicate the peak points

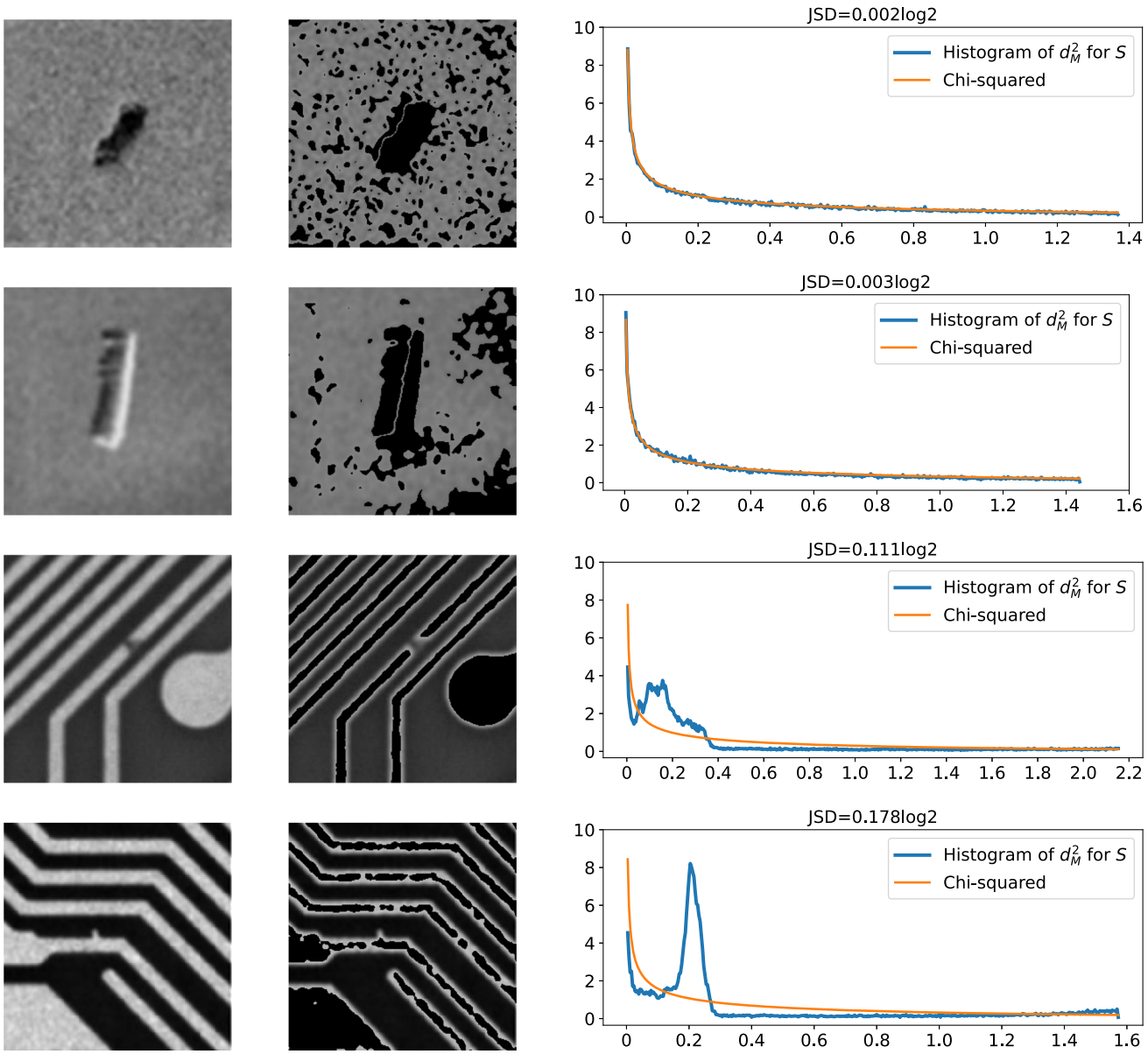


Fig. 14 JSDs for flat and complex images. First and second columns show the input image and the MCD-solution S, respectively. Third column shows the histogram of d_M^2 for the MCD-solution S and the chi-squared distribution

dominant direction defined as the direction of the major eigenvector.

Figure 13 shows the overall repeated region P with various $t_i = (1-r)+r \min CS_i$ for $r = 0.1, 0.15, 0.2, 0.25, 0.3$. The axis ratio of R is displayed at the top of each P. The maximum axis ratio of nonlinear images is 7.53, and the minimum one of linear images is 41.83. Therefore, we take the value 25 as the threshold for determination of linear images. Linear images always have an axis ratio greater than 25, regardless of t_i values. The peak points of the patterned images are aligned on specific lines.

A.2 Kernel size for cosine similarity

Here, we give a one-dimensional example of the cosine similarity for different kernel sizes. Consider a long vector in which (10110) is repeated. Then, the cosine similarities for the three types of one-dimensional kernels are as follows:

$$(\cdots 101101011010110\cdots) \text{ with } K = (101)$$
$$\Longrightarrow CS = (\cdots * 1 * *1 * 1 * *1 * 1 * *1\cdots)$$
$$(\cdots 101101011010110\cdots) \text{ with } K = (10110)$$

Algorithm 3 GHT method of extracting the lattice vectors [32].

Let $\{v_i\}_{i=1}^n$ be the peak points with vector representation in an ascending order of lengths, where $v_1 = (0, 0)$.
Set score matrix $L = [0]_{n\times n}$.
for $i = 2 : n$ **do**
  for $j = 2 : n$ **do**
    if v_i, v_j are collinear **then**
      $a = \frac{\|v_j\|}{\|v_i\|}$.
      $L_{1i} = L_{1i} + \frac{1-2|a-[a]|}{\|v_i\|}$, with the notation $[a]$ for the rounded integer of a.
    else
      for $k = 2 : n$ **do**
        Solve $v_k = av_i + bv_j$.
        $L_{ij} = L_{ij} + \frac{1-2\max(|a-[a]|,|b-[b]|)}{\max(\|v_i\|,\|v_j\|)}$.
      end for
    end if
  end for
end for
Let the entry with the highest value in L be $\left(\hat{i}, \hat{j}\right)$.
Select the pair of vectors with the smallest length in $\{v_{\hat{i}}, v_{\hat{j}}, v_{\hat{i}} + v_{\hat{j}}, v_{\hat{i}} - v_{\hat{j}}\}$ as lattice vectors $\{w_1, w_2\}$.

$$\Longrightarrow CS = (\cdots * * 1 * * * * 1 * * * * 1 * * \cdots)$$

$(\cdots 101101011010110\cdots)$ with $K = (1011010)$

$$\Longrightarrow CS = (\cdots * * * * 1 * * * * 1 * * * * 1 * \cdots)$$

If the kernel does not include a pattern as in the first case, the repeated region of CS cannot find the pattern. On the other hand, if the kernel represents a pattern as in the second and third cases, the repeated region of CS finds the pattern. Therefore, we suggest small M and N values like 2, 3, and 4 so that the kernel includes the pattern.

A.3 JSD between histogram and chi-squared distribution

We explain the details of the JSD between the histogram of d_M^2 for the MCD-solution S and the chi-squared distribution. Let $h_S(x)$ be the histogram of $d_M^2(x_i, \mu, \Sigma)$ for the MCD-solution S. For a defect-free image, the maximum value of d_M^2 in the MCD-solution S is close to $\chi^2_{1,1-\alpha}$ where the quantity

Algorithm 4 Our method of extracting the lattice vectors.

Let $\{l_i\}$ be a set of straight lines passing through the center of P, in descending order of the number of peak points in l_i.
For the same number of peak points, the line with the smaller distance between the points comes first.
for $i = 1 : 2$ **do**
  Let $\{v_j \mid j = 1, \ldots, n_i\}$ be the vectors in the line l_i.
  Find the minimum length vector $\hat{v}_i$ in l_i.
  for $j = 1 : n_i$ **do**
    $a_j = \frac{\hat{v}_i^T v_j}{\hat{v}_i^T \hat{v}_i}$.
  end for
  Let $\hat{w}_i = \mathbb{E}\left(\frac{1}{[a_j]} v_j\right)$.
end for
Select the pair of vectors with the smallest length in $\{\hat{w}_1, \hat{w}_2, \hat{w}_1 + \hat{w}_2, \hat{w}_1 - \hat{w}_2\}$ as lattice vectors $\{w_1, w_2\}$.

$\chi^2_{1,p}$ is defined to satisfy $P(X > \chi^2_{1,p}) = p$ for $X \sim \chi^2_1$. However, for defective images, the MCD-solution S appears outside the defects, and the maximum value of d_M^2 in the MCD-solution S increases. The increment depends on the size of the defects. Therefore, we use a slightly modified chi-squared distribution as follows. Let $f_1(x)$ be the probability density function for χ^2_1 distribution and l be the maximum value of d_M^2 in the MCD-solution S. Then, the cropped probability density function $\overline{f}_1(x)$ is

$$\overline{f}_1(x) = \frac{1}{\int_0^l f_1(x)dx} f_1(x)\,.$$

In Sect. 3.1, we measure the JSD between $h_S(x)$ and $\overline{f}_1(x)$ to check whether $h_S(x)$ is close to $\overline{f}_1(x)$ or not.

Figure 14 shows the JSDs for flat and complex images. For flat images, JSD is below $0.01 \log 2$. For complex images, JSD is greater than $0.1 \log 2$. Therefore, we take the value $0.05 \log 2$ as the threshold for determination of flat images.

A.4 Lattice vector extraction

In this appendix, we briefly describe how to extract the lattice vectors from the overall repeated region P in Sect. 3.2.2. Before starting, the locations of peak points are regarded as vectors originating from the center of domain $[1, \overline{h}] \times [1, \overline{w}]$.

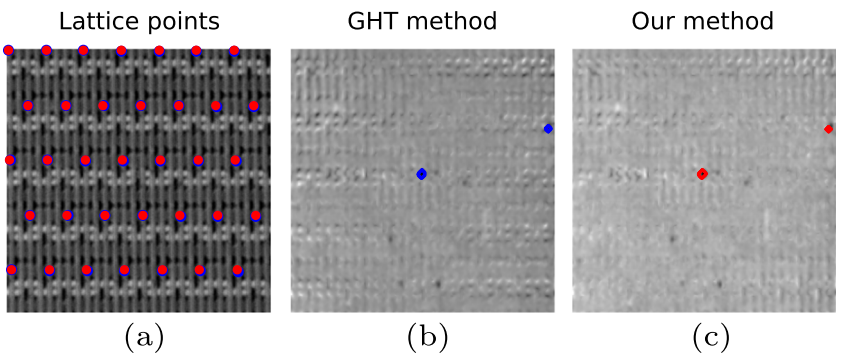


Fig. 15 Lattice points and results of GHT method and our method

The main idea of the generalized Hough transform (GHT) method in [32] of extracting lattice vectors is to build a parallelogram grid with each pair of linearly independent vectors and score how close the peak points are to the grid (see Algorithm 3). The usage of $1/\|v_i\|$ leads to a high score in L for v_i with small length. However, this method only uses the vectors $v_{\hat{i}}$ and $v_{\hat{j}}$ to determine the lattice vectors and does not use other vectors that are constant multiples of them.

We modified the method slightly to use more linearly dependent vectors to obtain the lattice vectors. We consider a straight line passing through the center of domain. As mentioned in Sect. 3.1, a straight line passing through three peak points is judged to have pattern information. If the line passes through more peak points, the pattern is better represented. To find the lattice vectors, we take two straight lines containing the largest number of peak points. If the straight lines have the same number of peak points, we take the line with the smaller distance between the points. The proposed lattice vector extraction algorithm is described in Algorithm 4. The lattice points generated from the lattice vectors extracted by our proposed algorithm are more accurate because the error is reduced by using more linearly dependent vectors. The amount of computation of our proposed extraction algorithm is also less than that of the GHT method.

Figure 15a shows the lattice points for two methods in input image. Blue dots and red dots represent the lattice points generated by the GHT method and our method, respectively. A flattened image obtained with the blue lattice points is shown in Fig. 15b. Near the boundary of the image, traces of structures remain. Figure 15c shows the flattened image obtained by our method. Compared to the GHT method, our method produces more accurate lattice points.

Author Contributions All authors contributed to the study conception and design. Data collection and analysis were performed by Jinkyu Yu and Songhee Han. The first draft of the manuscript was written by Jinkyu Yu and Chang-Ock Lee. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding This work was supported by Samsung Electronics Co., Ltd. (IO201216-08216-01).

Declarations

Competing interests The authors declare no competing interests.

References

1. Aiger D, Talbot H (2010) The phase only transform for unsupervised surface defect detection. In: 2010 IEEE computer society conference on computer vision and pattern recognition, pp 295–302. https://doi.org/10.1109/CVPR.2010.5540198
2. Alrawashdeh MJ, Sabri SRM, Ismail MT (2014) Evaluation the initial estimators using deterministic minimum covariance determinant algorithm. AIP Conference Proceedings 1605(1):894–899. https://doi.org/10.1063/1.4887708
3. Baranwal AK, Pillai S, Nguyen T et al (2021) An SEM-based deep defect classification system for VSB mask writer that works with die-to-die and die-to-database inspection methods using multiple digital twins built with the state-of-the-art neural networks. In: Renwick SP (ed) photomask technology 2021, international society for optics and photonics, vol 11855. SPIE, https://doi.org/10.1117/12.2601004, 118550D
4. Bret T, Hofmann T, Edinger K (2014) Industrial perspective on focused electron beam-induced processes. Applied Physics A 117(4):1607–1614. https://doi.org/10.1007/s00339-014-8601-2
5. Butler RW, Davies PL, Jhun M (1993) Asymptotics for the minimum covariance determinant estimator. The Annals of Statistics 21(3):1385–1400. https://doi.org/10.1214/aos/1176349264
6. Chen SH, Perng DB (2011) Directional textures auto-inspection using principal component analysis. Int J Adv Manuf Technol 55(9–12):1099–1110. https://doi.org/10.1007/s00170-010-3141-1
7. Criminisi A, Perez P, Toyama K (2004) Region filling and object removal by exemplar-based image inpainting. IEEE Trans Image Process 13(9):1200–1212. https://doi.org/10.1109/TIP.2004.833105
8. Crum W, Camara O, Hill D (2006) Generalized overlap measures for evaluation and validation in medical image analysis. IEEE Trans Med Imaging 25(11):1451–1461. https://doi.org/10.1109/TMI.2006.880587
9. Davy A, Ehret T, Morel JM et al (2018) Reducing anomaly detection in images to detection in noise. In: 2018 25th IEEE international conference on image rrocessing (ICIP), IEEE, pp 1058–1062, https://doi.org/10.1109/ICIP.2018.8451059
10. Dong X, Taylor CJ, Cootes TF (2020) Defect detection and classification by training a generic convolutional neural network encoder. IEEE Trans Signal Process 68:6055–6069. https://doi.org/10.1109/TSP.2020.3031188
11. D'Urzo L, Bayana H, Vandereyken J et al (2017) Continuous improvements of defectivity rates in immersion photolithography via functionalized membranes in point-of-use photochemical filtration. In: Hohle CK (ed) Advances in patterning materials and processes XXXIV, international society for optics and photonics, vol 10146. SPIE, pp 453–461, https://doi.org/10.1117/12.2258582
12. Ehret T, Davy A, Morel JM et al (2019) Image anomalies: a review and synthesis of detection methods. J Math Imaging Vis 61(5):710–743. https://doi.org/10.1007/s10851-019-00885-0
13. Elmanhawy W, Kwan J (2018) Layout schema generation: improving yield ramp during technology development. Solid State Technology 61(6):18–23
14. Fuglede B, Topsoe F (2004) Jensen-Shannon divergence and Hilbert space embedding. In: International symposium on information theory, pp 31–36. https://doi.org/10.1109/ISIT.2004.1365067
15. Ghosh B, Bhuyan MK, Sasmal P et al (2018) Defect classification of printed circuit boards based on transfer learning. In: 2018 IEEE applied signal processing conference (ASPCON), pp 245–248, https://doi.org/10.1109/ASPCON.2018.8748670
16. Goldberg K, Benk MP, Wojdyla A et al (2015) EUV actinic bright-field mask microscopy for predicting printed defect images. In: Photomask technology 2015, vol 9635. SPIE, pp 271–277, https://doi.org/10.1117/12.2196966
17. Grosjean B, Moisan L (2009) A-contrario detectability of spots in textured backgrounds. J Math Imaging Vis 33(3):313–337. https://doi.org/10.1007/s10851-008-0111-4
18. Hagi H, Iwahori Y, Fukui S et al (2014) Defect classification of electronic circuit board using SVM based on random sampling. Procedia Computer Science 35:1210–1218. https://doi.org/10.1016/j.procs.2014.08.218

19. He K, Zhang X, Ren S et al (2016) Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), pp 770–778, https://doi.org/10.1109/CVPR.2016.90
20. Huang SH, Pan YC (2015) Automated visual inspection in the semiconductor industry: a survey. Computers in Industry 66:1–10. https://doi.org/10.1016/j.compind.2014.10.006
21. Huang W, Wei P (2019) A PCB dataset for defects detection and classification. ArXiv:1901.08204
22. Hubert M, Debruyne M (2010) Minimum covariance determinant. WIREs. Computational Statistics 2(1):36–43. https://doi.org/10.1002/wics.61
23. Hubert M, Van Driessen K (2004) Fast and robust discriminant analysis. Computational Statistics & Data Analysis 45(2):301–320. https://doi.org/10.1016/S0167-9473(02)00299-2
24. Impedovo D, Pirlo G, Sarcinella L et al (2012) Analysis of stability in static signatures using cosine similarity. In: 2012 international conference on frontiers in handwriting recognition, pp 231–235, https://doi.org/10.1109/ICFHR.2012.180
25. Iwahori Y, Futamura K, Adachi Y (2011) Discrimination of true defect and indefinite defect with visual inspection using SVM. In: International conference on knowledge-based and intelligent information and engineering systems, Springer, pp 117–125, https://doi.org/10.1007/978-3-642-23866-6_13
26. Iwahori Y, Kumar D, Nakagawa T et al (2012) Improved defect classification of printed circuit board using SVM. In: Intelligent decision technologies. Springer, pp 355–363, https://doi.org/10.1007/978-3-642-29920-9_36
27. Iwahori Y, Takada Y, Shiina T et al (2018) Defect classification of electronic board using dense SIFT and CNN. Procedia Computer Science 126:1673–1682. https://doi.org/10.1016/j.procs.2018.08.110
28. Kang H (2021) SK Hynix presents various solutions to difficult challenges related to EUV lithography process. https://english.etnews.com/20210210200001
29. Kervadec H, Bouchtiba J, Desrosiers C et al (2019) Boundary loss for highly unbalanced segmentation. In: Proceedings of the 2nd international conference on medical imaging with deep learning, proceedings of machine learning research, vol 102. PMLR, pp 285–296
30. Kim J, Nam Y, Kang MC et al (2021) Adversarial defect detection in semiconductor manufacturing process. IEEE Trans Semicond Manuf 34(3):365–371. https://doi.org/10.1109/TSM.2021.3089869
31. Kondo T, Ban N, Ebizuka Y et al (2021) Massive metrology and inspection solution for EUV by area inspection SEM with machine learning technology. In: Metrology, inspection, and process control for semiconductor manufacturing XXXV, international society for optics and photonics, vol 11611. SPIE, https://doi.org/10.1117/12.2583691, 1161111
32. Lin HC, Wang LL, Yang SN (1997) Extracting periodicity of a regular texture based on autocorrelation functions. Pattern Recognition Letters 18(5):433–443. https://doi.org/10.1016/S0167-8655(97)00030-5
33. Lin HD, Ho DC (2007) Detection of pinhole defects on chips and wafers using DCT enhancement in computer vision systems. The International Journal of Advanced Manufacturing Technology 34(5–6):567–583. https://doi.org/10.1007/s00170-006-0614-3
34. Lu CJ, Tsai DM (2008) Independent component analysis-based defect detection in patterned liquid crystal display surfaces. Image and Vision Computing 26(7):955–970. https://doi.org/10.1016/j.imavis.2007.10.007
35. Matsui M, Yano T, Odaka T et al (2012) Quantitative measurement of voltage contrast in scanning electron microscope images for in-line resistance inspection of incomplete contact. Journal of Micro/Nanolithography, MEMS, and MOEMS 11(2). https://doi.org/10.1117/1.JMM.11.2.023008, 023008
36. Mochi I, Fernandez S, Nebling R et al (2019) Absorber and phase defect inspection on EUV reticles using RESCAN. In: Extreme ultraviolet (EUV) lithography X, international society for optics and photonics, vol 10957. SPIE, https://doi.org/10.1117/12.2515160, 109570W
37. Nakagaki R, Honda T, Nakamae K (2009) Automatic recognition of defect areas on a semiconductor wafer using multiple scanning electron microscope images. Measurement Science and Technology 20. https://doi.org/10.1088/0957-0233/20/7/075503
38. Nakagawa T, Iwahori Y, Bhuyan M (2013) Defect classification of electronic board using multiple classifiers and grid search of SVM parameters. In: Computer and information science. Springer, pp 115–127, https://doi.org/10.1007/978-3-319-00804-2_9
39. Nam Y, Joo S, Kwak N et al (2022) Precise pattern alignment for die-to-database inspection based on the generative adversarial network. IEEE Trans Semicond Manuf 35(3):532–539. https://doi.org/10.1109/TSM.2022.3171788
40. Nguyen HV, Bai L (2011) Cosine similarity metric learning for face verification. In: Computer vision-ACCV 2010. Springer, Berlin, Heidelberg, pp 709–720, https://doi.org/10.1007/978-3-642-19309-5_55
41. Oberai A, Yuan JS (2017) Smart E-beam for defect identification & analysis in the nanoscale technology nodes: technical perspectives. Electronics 6. https://doi.org/10.3390/electronics6040087
42. Ouchi M, Ishikawa M, Shinoda S et al (2020) A trainable die-to-database for fast e-beam inspection: learning normal images to detect defects. In: Metrology, inspection, and process control for microlithography XXXIV, vol 11325. SPIE, https://doi.org/10.1117/12.2551456, 113252F
43. Paszke A, Gross S, Massa F et al (2019) PyTorch: an imperative style, high-performance deep learning library. In: Advances in neural information processing systems, pp 8024–8035. https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf
44. Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. IEEE Trans Pattern Anal Mach Intell 12(7):629–639. https://doi.org/10.1109/34.56205
45. Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In: Navab N, Hornegger J, Wells WM et al (eds) Medical image computing and computer-assisted intervention – MICCAI 2015. Springer International Publishing, Cham, pp 234–241, https://doi.org/10.1007/978-3-319-24574-4_28
46. Rousseeuw PJ (1984) Least median of squares regression. J Am Stat Assoc 79(388):871–880. https://doi.org/10.1080/01621459.1984.10477105
47. Rousseeuw PJ, Driessen KV (1999) A fast algorithm for the minimum covariance determinant estimator. Technometrics 41(3):212–223. https://doi.org/10.1080/00401706.1999.10485670
48. Sato Y, Huang SC, Maruyama K et al (2019) Edge placement error measurement in lithography process with die to database algorithm. In: Metrology, inspection, and process control for microlithography XXXIII, international society for optics and photonics, vol 10959. SPIE, https://doi.org/10.1117/12.2515143, 109590D
49. Shenzhen Eagle Eye Online Electronic Technology (2023) Eagle eye technology: AOI technical expert. http://en.eagle-eye-online.com/solution.aspx?TypeId=62&fid=t25:62:25
50. Shiina T, Iwahori Y, Takada Y et al (2017) Reducing misclassification of true defects in defect classification of electronic board. In: International conference on computer and information science, Springer, pp 77–92, https://doi.org/10.1007/978-3-319-60170-0_6
51. Shiina T, Iwahori Y, Kijsirikul B (2018) Defect classification of electronic circuit board using multi-input convolutional neural net-

work. International Journal of Computer & Software Engineering 3. https://doi.org/10.15344/2456-4451/2018/137, 137
52. Sikka S, Sikka K, Bhuyan MK et al (2013) Pseudo vs. true defect classification in printed circuits boards using wavelet features. ArXiv:1310.6654
53. Stephant N, Rondeau B, Gauthier JP et al (2014) Investigation of hidden periodic structures on SEM images of opal-like materials using FFT and IFFT. Scanning 36(5):487–499. https://doi.org/10.1002/sca.21144
54. Sudre CH, Li W, Vercauteren T et al (2017) Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In: Deep learning in medical image analysis and multimodal learning for clinical decision support. Springer, pp 240–248, https://doi.org/10.1007/978-3-319-67558-9_28
55. Tabernik D, Šela S, Skvarč J et al (2020) Segmentation-based deep-learning approach for surface-defect detection. J Intell Manuf 31(3):759–776. https://doi.org/10.1007/s10845-019-01476-x
56. Takada Y, Shiina T, Usami H et al (2017) Defect detection and classification of electronic circuit boards using keypoint extraction and CNN features. In: The ninth international conferences on pervasive patterns and applications Defect, pp 113–116
57. Tao X, Zhang D, Ma W et al (2018) Automatic metallic surface defect detection and recognition with convolutional neural networks. Applied Sciences 8(9). https://doi.org/10.3390/app8091575, 1575
58. Tao Z, Liu H, Fu H et al (2017) Image cosegmentation via saliency-guided constrained clustering with cosine similarity. Proceedings of the AAAI conference on artificial intelligence 31(1). https://doi.org/10.1609/aaai.v31i1.11203
59. Waiblinger M, Bret T, Jonckheere R et al (2012) Ebeam based mask repair as door opener for defect free EUV masks. In: Photomask technology 2012, vol 8522. SPIE, https://doi.org/10.1117/12.966387, 85221M
60. Weimer D, Scholz-Reiter B, Shpitalni M (2016) Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. CIRP Annals 65(1):417–420. https://doi.org/10.1016/j.cirp.2016.04.072
61. Welsch RE, Zhou X (2007) Application of robust statistics to asset allocation models. REVSTAT-Statistical Journal 5(1):97–114. https://doi.org/10.57805/revstat.v5i1.44
62. Xia P, Zhang L, Li F (2015) Learning similarity with cosine similarity ensemble. Information Sciences 307:39–52. https://doi.org/10.1016/j.ins.2015.02.024
63. Xiao H, Ma E, Wang F et al (2009) Leakage study of 45nm SRAM devices with different layouts using advanced e-beam inspection systems. https://nccavs-usergroups.avs.org/wp-content/uploads/TFUG2009/2009_4xiao.pdf
64. Yeh CH, Wu FC, Ji WL et al (2010) A wavelet-based approach in detecting visual defects on semiconductor wafer dies. IEEE Trans Semicond Manuf 23(2):284–292. https://doi.org/10.1109/TSM.2010.2046108
65. Yin A, Fung A (2011) Effective analysis of optical inspection machines (IMPACT 2011). In: 2011 6th international microsystems, packaging, assembly and circuits technology conference (IMPACT), pp 408–410, https://doi.org/10.1109/IMPACT.2011.6117212
66. Yu J, Han S, Lee CO (2023) Dataset for defect inspection in semiconductor images. https://github.com/Jinkyu-Yu/Image-dataset.git
67. Zontak M, Cohen I (2009) Kernel-based detection of defects on semiconductor wafers. In: 2009 IEEE international workshop on machine learning for signal processing, pp 1–6, https://doi.org/10.1109/MLSP.2009.5306256