Defects Detection in PCB Images by Clustering, Rotation and Distributed Cumulative Histogram

[1]ROMAN MELNYK, [2]ROMAN KVIT

[1]Software Department
Lviv Polytechnic National University
12 Stepan Bandera St., Lviv, 79013
UKRAINE

[2]Department of Mathematics
Lviv Polytechnic National University
12 Stepan Bandera St., Lviv, 79013
UKRAINE

Abstract: The K-means clustering of pixels intensity to segment the PCB image to a binary form, hierarchical clustering and separation of elements on the PCB image, flood-filling of chains for their comparison with the reference ones, three methods of determination of the turn angle for alignment of the board, subtraction formulas, algorithms to rotate the image to its normal position, algorithms to build the image of the distributed cumulative histogram are considered in this paper.

Key Words: printed circuit boards, chains, defects, clustering, flood-filling, pixel histogram, distributed cumulative histogram, image rotation, alignment.

Received: January 2, 2023. Revised: August 11, 2023. Accepted: September 19, 2023. Published: October 8, 2023.

1 Introduction

Many publications are devoted to problems of defect detection in the printed circuit boards. In the last time, new works describe an application of artificial networks for inspection of inaccuracies of PCB production. Among them, neural networks are used in articles [1-3]. Many publications contain subtraction algorithms. Examples are in works [4-6]. As a rule, defects are classified to facilitate their correction. The earlier article [7] uses a table of connectivity wires based on contacts from the reference image. Inaccuracies of contacts and traces such as shift, extra metal, and others are not considered.

The surveys [8-10] contain many publications describing approaches and methods of the detection of defects in printed circuit boards as well as their classification. The main ideas of the following works [11-13] are image subtraction, defects extraction, thinning, and increasing the visibility of elements on the board. Different techniques to elaborate a plane are applied. The subtraction operations between PCB images are very sensible to a deviation of the sizes of contacts and traces from the reference ones and give extra or lacking pixels that do not influence the correct work of the circuit.

Short, open circuits and depression changes are discussed and researched in the works [14,15] for machine vision and learning inspection systems. All mentioned approaches differ between themselves by complexity, application objects, and efforts for their implementation. Some use the SURF method to determine the image features [16,17] for matching and alignment. In common, most of the mentioned approaches are very time-consuming to realize. In this paper, some simple approaches are presented. They are the following: K-means clustering of the pixel intensity to segment the PCB image to a binary form, hierarchical clustering and separation of elements in the PCB image, flood-filling of chains for their comparison with the reference ones, three methods of determination of the turning angle for alignment of the board, subtraction formulas, algorithms to rotate the image to its normal position, algorithms to build the image of the distributed cumulative histogram, measurement of the chain position and its defects.

2 Clustering of PCB images

Two clustering algorithms have been developed for the analysis of the printed circuit board image. The former algorithm is assigned for preprocessing and preparation of the printed circuit board image for basic image processing and defect detection. The second algorithm is applied to find and separate analyzed objects such as contacts, circuits, and electronic devices. Two algorithms considered are K-means clustering and agglomerative hierarchical clustering.

A. K-means clustering algorithm

The K-means clustering algorithm divides all pixels of the PCB image into K segments of pixels and assigns equal intensity for them.

Iterations of the algorithm repeat to distribute pixels into clusters to minimize all distances between the centroids and their pixels. The module criterion function is as follows:

$$S = \sum_{k=0}^{K} \sum_{i=1}^{m(k)} w_{ki} \, | I_{ki} - \bar{I}_k | ,$$

where w_{ik}= 1 if a pixel with intensity I_{ir} enters into the cluster k, otherwise, w_{ik}= 0; $\bar{I}_k$ is the intensity of the k -th centroid.

The process of assignment of pixels to different clusters is shown in Fig. 1. The input data are pixels of different intensity. The first iteration is a random assignment of K centroids and searching for closest pixels to them. The mean intensity of centroids is recalculated. Then the iterative process continues until replacement of pixels is not fixed.

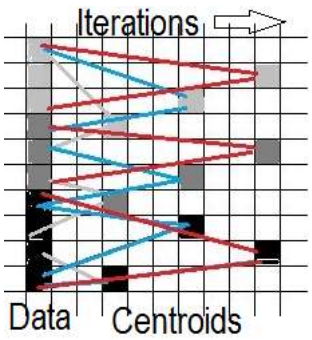
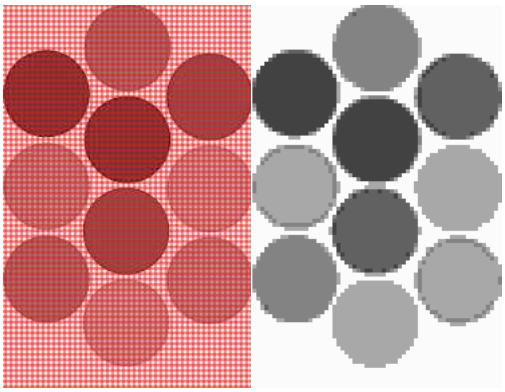


Fig. 1. Iterations of pixels' assignment in the K -means algorithm

The process is less time-consuming when the clustering algorithm works with rectangles created after covering the image with a grid. The input image is covered with a grid of 3×3 rectangles in Fig. 2a. In this case, in Fig. 2b, the edges of the clustered image are blurred. After the end of the algorithm steps all pixels in clusters are redrawn with the mean intensity value or other.



a b

Fig. 2. An image covered by a grid (a) and its clustered segments (b)

The clustering algorithm includes the following steps:

S0. For all points of the input set $x_j \in X$.

S1. Assignment of K clusters centroids $x_i^0 \, (i = 1, ..., K)$ having an interval of the image intensity S

$$x_i^0 = (s / (K+1)) * i, \; i = 1, ..., K ,$$

S2. Searching for leaf pairs by the similarity function

$$\forall (x_i^0, x_j) \text{ count } F(x_i^0, x_j) .$$

S3. Calculating and searching for pairs with the smallest distance value

$$F^*(x_i^0, x_j) = \min F(x_i^0, x_j), \; i = 1, ..., K, \; j \in I ,$$

and adding the member x_j to a cluster with the centroid x_i^0 .

S4. Recalculating the cluster's centroids $x_i^0 \, (i = 1, ..., K)$.

S5. The end of the procedure (for all $x_j \in X$).

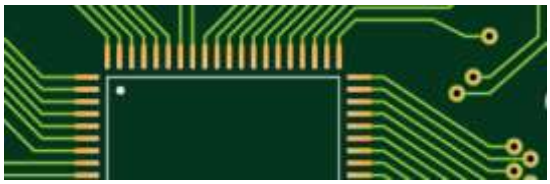
Steps S0-S4 are run until the centroids do not change their positions.

If a clustering algorithm is applied to the input image to obtain two segments ($K = 2$) the advantages of the approach are as follows:

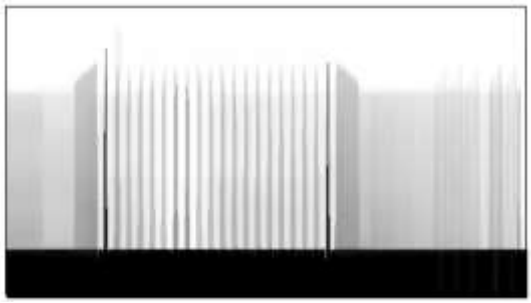
1. No one pixel is deleted;
2. Pixels are grouped naturally near their centroids;
3. Segment colors are uniform and their values do not affect the subsequent fill.

The input image, for example, is shown in Fig. 3a.

The DCH image (the algorithm is described below) in Fig. 3b distinctly illustrates how pixels are distributed in the input grey image.



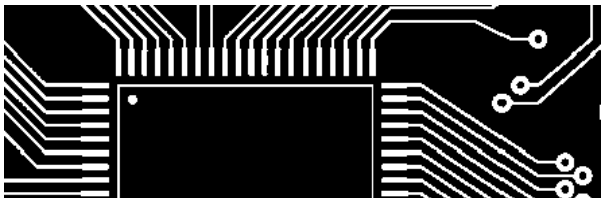
a



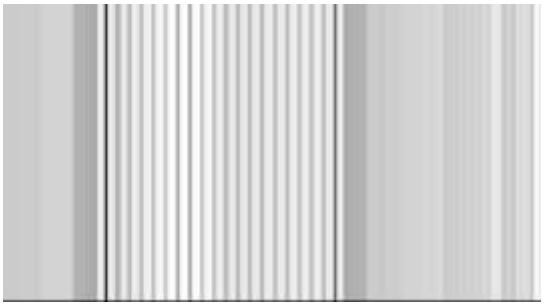
b

Fig. 3. The PCB image (a) and its DCH image (b)

Fig. 4a shows two groups of pixels in the clustered image. The first one contains contacts with traces, and the second group contains background.



a



b

Fig. 4. The PCB image with two clusters (a) and its DCH image (b)

The DCH image in Fig. 4b distinctly illustrates how pixels are distributed in the clustered image. It confirms that only one intensity of objects in every segment of a 2-clustered image is available.

For better illustration, the image histogram is calculated as follows:

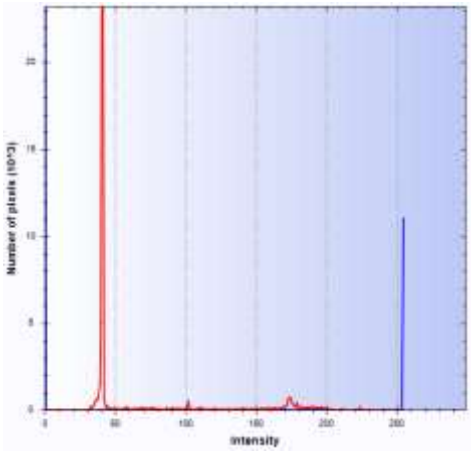
$$h(i) = card\{(u,v) \mid I(u,v) = i\},$$
$$i = 0,1, ..., 255,$$

and the cumulative histogram is calculated:

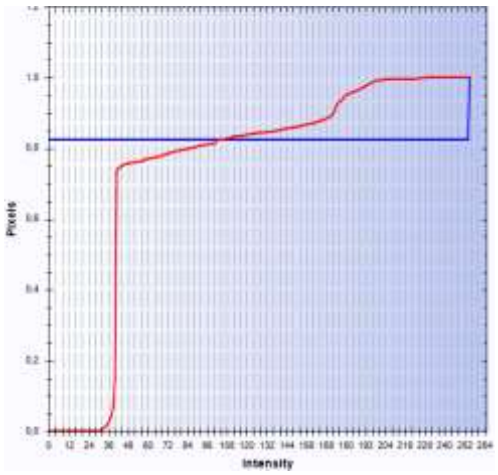
$$H(i) = \sum_{j=1}^{i} h(j),$$

where $I(u,v)$ is the pixel intensity, $h(i)$ is the intensity frequency, $H(i)$ is the accumulating frequency for the given intensity.

These four histograms for the input and clustered images are shown in Fig. 5.



a



b

Fig. 5. Histograms of the input (red) and clustered (blue) images: ordinary (a), cumulative (b)

Two red histograms characterize the input grey image; two blue histograms are properties of the clustered image. They illustrate that all objects on the circuit board after clustering are uniform in color and ready for further processing.

B. Hierarchical clustering of the binary image

The hierarchical clustering algorithm is a more complex procedure because it solves the decomposition problem. All connected areas are formed and described by the hierarchical algorithm.

An example to illustrate a transformation process to build one area from its components is shown in Fig. 6. The number of rectangles corresponds to the number of nodes in the hierarchical tree.

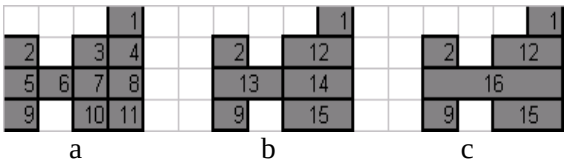


Fig. 6. Formation of rectangles: input 1-11 (a), four new 12-15 (b), five new 12-16 (c)

In Fig. 6a eleven elementary rectangles are input data for the algorithm. After the first step of the algorithm in Fig. 6b new rectangles are formed and shown: $(3,4=12)$, $(5,6=13)$, $(7,8=14)$, $(10,11)=15$. After the second step in Fig. 6c only one new rectangle is formed $(13,14=16)$. The reason is that no more rectangles are to join with others.

For this small example, the hierarchical tree is shown in Fig. 7.

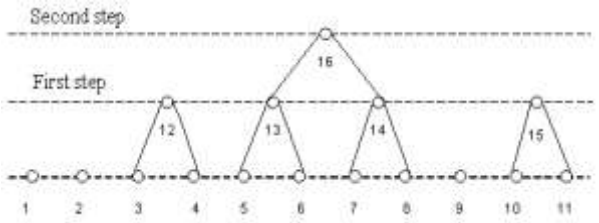


Fig. 7. Hierarchical process of formation of rectangles

Such a process is organized by applying the criterion and constraint functions to rectangles-candidates planned to be united. For them, the functions are calculated, and the best pair is ready to join:

$$F^* = \min(F_{ki}),\ k, j \in I,$$

where I is a set of possible pairs of initial rectangles and their combinations. As the criterion, the length of the common border for two rectangles is accepted and calculated. The criterion is calculated only for two input rectangles forming a new one of a rectangular form.

The sequence of steps in the hierarchical algorithm is as follows.

1. Space dividing. Imposing a grid with a step from a set of values $1\times1, 2\times1, ..., 3\times3$ on an input image to form the set of micro-rectangles (MR).
2. For each MR their neighbors are searched around the rectangle.
3. If the new planned rectangle R is of rectangular form and satisfies the brightness constraint it is added to the list L of the candidates for merging.
4. If some pairs of MR have common elements they are removed from the list L. In the (a,b), (b,c), (a,d), (d,f) two pairs (a,d), (b,c) are removed.
5. New rectangles are formed, and previous steps are repeated with them.

These five steps of the clustering algorithm for a grid with a step 1×1 (one pixel) over a black image are fulfilled. There are no extra or lacking pixels in the resulting image. This is illustrated in Fig. 8 by two examples with the input and clustered images. Fig. 8b contains rectangles with different colors.

Fig. 8. Input (a) and clustered (b) images

The hierarchical clustering algorithm has nonlinear complexity $O(N^2)$. Thus, for large images, a process is time-consuming if additional operations are not realized.

The first step to improve time characteristics is covering of the image by a grid with a step of 2×2. In this case, initial rectangles are fully and partially filled by black pixels as this is shown in Fig. 9.

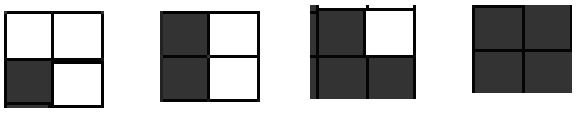


Fig. 9. Different types of input rectangles for clustering

In a process of clustering new rectangles are built. Some of them have empty positions as this is shown by examples in Fig. 10. Positions are empty because the input image and initial rectangles have no black pixels with these coordinates.

To control the process of rectangle formation the criterion function is formulated. Let the number of black pixels in an area be S_o, and in the area of the clustered rectangle S_{oc}. If clustering is accurate a condition $S_{oc} = S_o$ is satisfied.

To control the relation between S_{oc} and S_o a brightness of clusters as a control parameter is accepted in the merging algorithm. The parameter indicates percentage relations between black and white pixels entering rectangles, for example, $1/6$, $3/8$, $3/9$, $4/10$ as this is shown in Fig. 10.

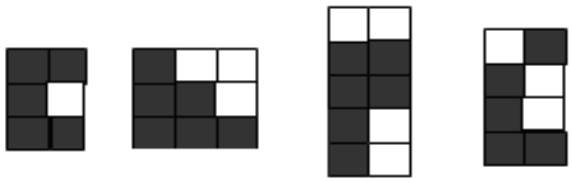
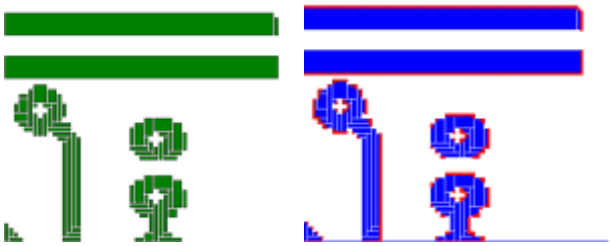


Fig. 10. Different filling of intermediate rectangles

An example of clustering for a grid with a step of a 2×2 cell is shown in Fig. 11a. In Fig. 11b a difference between the clustered image and error pixels is shown. Error pixels are marked with red.



a b

Fig. 11. Clustered image with a 2×2 cell (a) and its difference between error pixels (b)

New rectangles in the image are of larger sizes than in the previous 1×1 pattern. They contain extra pixels because final rectangles are filled with black without checking positions of rectangles what they contain. Then positions of error pixels are checked, marked, and isolated.

When clustering by rectangles 1×2, 2×1, 2×2 and higher steps the original clustering image I_{oc} is easily obtained by subtraction of the clustered and error images:

$$I_{oc} = I_c(2\times2) - I_e(2\times2),$$

where $I_c(2\times2)$ is the clustered image, $I_e(2\times2)$ is the image with error pixels. The subtraction operation is special because in the first input image it fills 0 in positions where the second input image has a value of 1 and does not change an input pixel when the second pixel has a value of 0:

$$0=0\cdot\theta\cdot0,\ 1=1\cdot\theta\cdot0,\ 0=1\cdot\theta\cdot1,\ 0=0\cdot\theta\cdot1.$$

The computing complexity of such subtraction is $O(N)$.

In conclusion, no additional operations are required when a grid with a step 1×1 is applied. N additional simple operations are required for a grid with a step of a 2×2 cell. In the second case, the number of input rectangles is essentially smaller. For example, complexity $O(N^2)$ is changed by $O((N/4)^2+N)$ for 2×2 step and $O((N/9)^2+N)$ for 3×3 step. For large images, it is a significant gain in time.

C. Hierarchical clustering of the binary image. Clustering to connected areas

Preliminary previous clustering does not give the result because all connected areas are not considered as a single whole but as systems of filled rectangles. Thus, the second stage of the algorithm is developed to unite rectangles into connected areas. Intermediate objects are clusters.

The clustering procedure to unite rectangles in clusters of undefined shape and connected areas is based on the criterion function for two clusters. Two rectangles-clusters are merged if they have one or more common edges (Fig. 12).

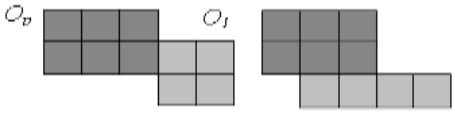


Fig. 12. Two cases of different common borders

To account for some pixel loss rectangles-clusters can participate in merging under conditions that they are located at a distance s (Fig. 13). The distance criterion includes a “diagonal” or “main” distance which is useful sometimes for ring structures.

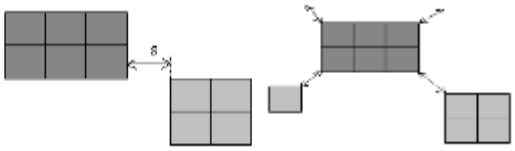
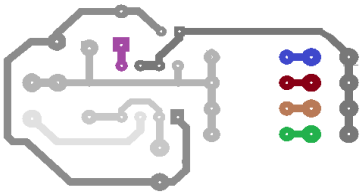


Fig. 13. Non-zero distances between clusters

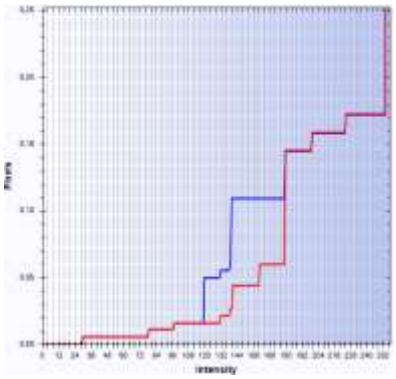
The process terminates when the merging of rectangles does not occur.

3. Clustering for separation of chains

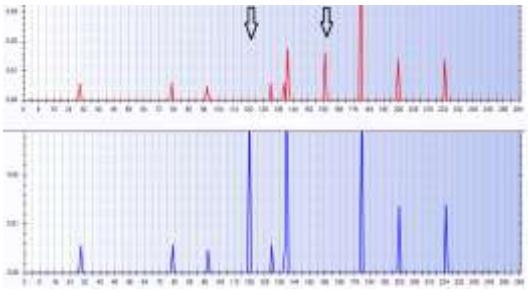
The results of the clustering algorithm are useful for the separation and comparison of connected circuits in the reference and manufactured PCB images. For example, two PCB images are clustered. The first one is the reference and the second is the manufactured one with short and open defects. The first is marked with different colors and shown in Fig. 14a. Four histograms for them are calculated and shown in Fig. 14b, c.



a



b



c

Fig. 14. PCB image (a), cumulative histograms of correct and defective PCBs (b), their histograms (c)

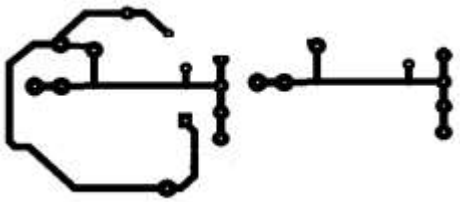
Due to similarity the defective PCB image is missed. Blue corresponds to the correct PCB and red is for the defective one in the two graphs of histograms. When one blue peak is absent in the new graph of a histogram it means that one chain as an independent object disappeared and by a short defect is joined to another chain. When one extra red peak appeared in the new graph it means that one independent fragment arose due to an open defect in some chain. The cumulative histogram also illustrates these two facts.

In conclusion, after obtaining such data the answer to the question about possible open or short defects is ready. Even if the numbers of components are the same in the correct and defective PCBs.

Obtained data also are suitable for the determination of coordinates, types, and other inaccuracies in the controlled PCB. Every chain has with own color (intensity). Thus, for every chain a histogram of pixels in the OX or OY axis is applied. For the given interval of intensity $I(\text{int})$ the numbers of pixels in columns or in rows are calculated as follows

$$h(c) = card\{(x,y)\,|\,I(x,y) \subset I(\text{int})\},$$
$$h(r) = card\{(x,y)\,|\,I(x,y) \subset I(\text{int})\},$$
$$(c = 0,1,...,w;\;\; r = 0,1,...,H).$$

It is not required to select and separate one chain from the resulting image because all of them have their own intensity marked in histograms in Fig. 14b, c. For illustration, only two chains are drawn with black and shown in Fig. 15 with a short defect (two chains are connected) and without it.



a b

Fig. 15. Short defect in two chains (a) and correct chain (b)

Two graphs of histograms in columns and rows are shown in Fig. 16. Differences are distinct. Additional functions determine the coordinates of defects. Black graphs are of defective chain, red is of the reference chain. They mark the coordinates of different positions. Pink marks a difference in the number of black pixels for every coordinate.

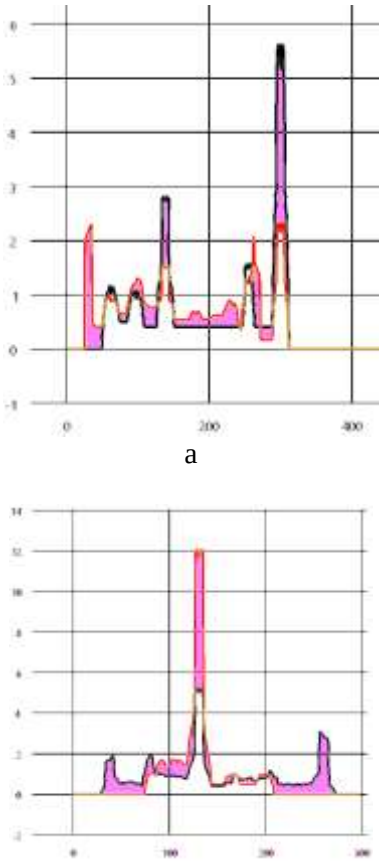


Fig. 16. Histogram from OX (a) and histogram from OY (b) of two chains

The second approach is designed for the determination of coordinates and visual inspection of defects. It uses images of separated chains and their distributed cumulative histograms.

Two images are compared according to the following comparison formulas:

$$I_r(x,y) = I_d(x,y) \quad \forall \mid I_e(x,y) - I_d(x,y) \mid < Tol ,$$

$$I_r(x,y) = RGB(\mid I_e(x,y) - I_d(x,y) \mid ,00),(red)$$
$$\forall \mid I_e(x,y) - I_d(x,y) \mid \geq Tol ,$$

and a sign is +,

$$I_r(x,y) = RGB(0,\mid I_e(x,y) - I_d(x,y) \mid ,0),(blue)$$
$$\forall I_d(x,y) - I_e(x,y) \geq Tol ,$$ and a sign is -,

where $I_r(x,y)$ is the pixel intensity of the resulting image, $I_e(x,y)$ is the pixel intensity of the first image, $I_d(x,y)$ is the pixel intensity of the second image, Tol is the tolerance value for controlling the difference between the pixel intensity of the reference and controlled samples.

The comparison formula is applied to two chains shown in Fig. 17. Differences between them and their DCH images are shown in Fig. 18.

Fig. 17. Open defect (a) and correct chain (b)

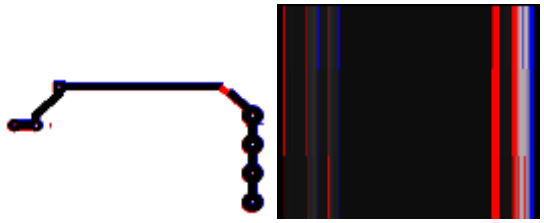


Fig. 18. Difference: between two chains (a) and their DCH images (b)

In Fig. 18a red marks inaccuracies of traces and the open defect. In Fig. 18b a red wide stripe corresponds to coordinates of the open defect in the OX axis. Other red strips mark inaccuracies.

In conclusion, there are many other approaches to determining the coordinates of defects by comparison of two separated chains from the reference and controlled images.

4. Rotating image and distributed cumulative histogram

A. Determination of angle by sizes of images

Sometimes a controlled PCB image is not correctly aligned with the reference PCB image for reasons of positions of the installation equipment or the camera angle.

In Fig. 19 two examples of the rotated PCB image and its clustered pattern are shown.

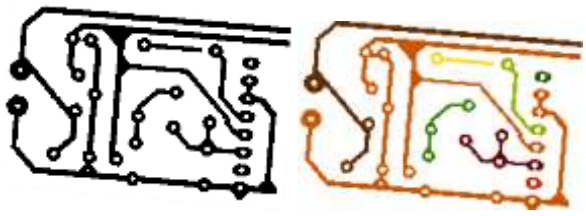
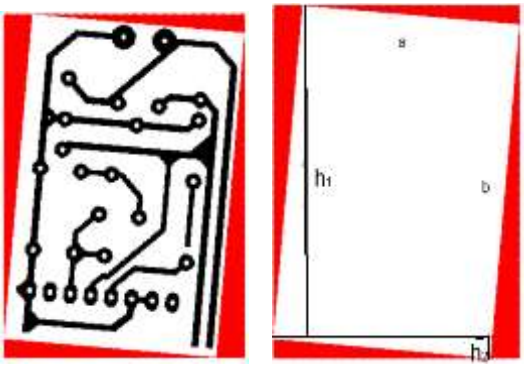


Fig. 19. Turned images: input (a), clustered (b)

Due to different angles, traces change their positions and sizes. Directly comparison with traces from the reference PCB image will give unpredictable results. In this case, two variants are possible: to rotate the reference image or the controlled image. Only the angle is required to be determined and software to realize rotation.

The unaligned PCB image as a rectangle is shown in Fig. 20a. An extra place connected with camera requirements and incorrect placement of the PCB image is marked with red. In Fig. 20b two triangles for the angle determination are shown.



a b

Fig. 20. Marked area on the incorrect placement of the PCB image (a) and heights of triangles (b)

The angle α is determined from the equation for two rectangles and a height h of the input image:

$$h = h_1 + h_2 = a\cos\alpha + b\sin\alpha ,$$

where values of h, a, b are known.

The equation is transformed into the quadratic equation with one unknown:

$$(a^2 + b^2)x^2 - 2hbx + (h^2 - a^2) = 0 .$$

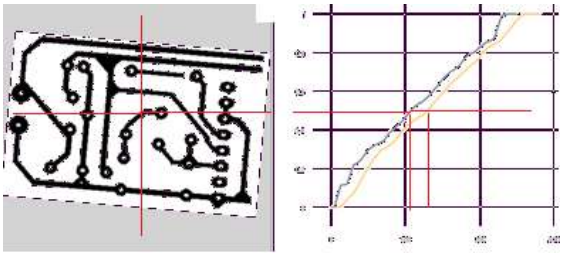
For the simple example, if $a = 1, b = 1, h = \sqrt{2}$, in the result $x = \cos\alpha = \sqrt{2}/2$ and $\alpha = 45^o$.

The determined angle is the input data for software to rotate and align the input image.

B. Determination of angle by centers of the image and pixels

The second approach is also simple and requires some mathematical calculations.

Two points are determined for the reference and inclined images. They are a center of the image and a center of black pixels. The first has coordinates $c_i = (W/2, H/2)$. The second center has the coordinates corresponding to the level of 0.5 in the cumulative histogram of black pixels, i.e., $c_p = (x(0.5ch), y(0.5ch))$. It is calculated for both images from the OX and OY axes. These coordinates mark a border dividing black pixels into equal parts. The lines dividing the image into parts are shown in Fig. 21a. One coordinate in the OY axis for the unaligned image is shown in Fig. 21b.



a b

Fig. 21. Divided PCB image with turning angle (a), and division of its cumulative graph (b)

The found coordinates for two images give two vectors:

$$A_u = [(0, W/2), (0, H/2)],$$
$$B_r = [0, x(0.5ch)), (0, y(0.5ch)],$$

where $x(0.5ch)$, $y(0.5ch)$ are the coordinates corresponding to the level of 0.5 in two cumulative histograms for the OX and OY axes of the input PCB image.

Examples of vectors are shown in Fig. 22 only for illustration not reflecting the considered PCB image. In practice, they are built separately because the sizes of the non-aligned and reference images are a little different (the first one is larger).

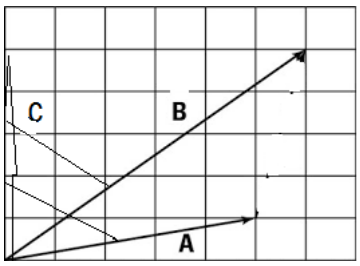


Fig. 22. Vectors to centers of black pixels

For two vectors two angles to vertical vector C are trivially calculated by the following formulas:

$$\alpha_a = \cos^{-Io}[(a^{oc}) / (|a| \cdot |c|],$$
$$\alpha_b = \cos^{-Io}[(b^{oc}) / (|b| \cdot |c|].$$

To align the controlled image a difference in found angles is required:

$$\alpha = \alpha_a - \alpha_a .$$

In the above-considered example with the turned PCB image relative values of direction vectors are $a = [(0.8), (0.13)]$, $b = [(0.8), (0.17)]$.

The angle of the turning is $\alpha = 63 - 58 = 5$.

This angle is the input data for the algorithm of rotation and building the distributed cumulative histograms.

C. Rotation of images

It is useful to construct DCH from different viewpoints to measure image properties. Depending on the viewing angle, the distributed histograms of the same image show a different distribution of pixels and their intensity.

The image intensity values are transformed into greyscale. The simple formula is used to average the pixel component values $R'G'B'$

$$R_{ij}^{'} = G_{ij}^{'} = B_{ij}^{'} = \frac{R_{ij} + G_{ij} + B_{ij}}{3},\ i = 1,...,B,\ i = 1,...,A,$$

where $R'_{ij}, G'_{ij}, B'_{ij}$ are values of new image pixels, B and A are sizes of the input image.

The final rotated image is of larger dimension due to an extra area surrounding it. Thus, to present them a new canvas is required. It is determined by the formulas:

$$d = \sqrt{A^2 + B^2},$$

$$\alpha = \begin{cases} \arcsin(A/d), & |\theta| > \pi/2, \\ \arcsin(B/d), & |\theta| \le \pi/2 \end{cases}$$

$$\beta = \begin{cases} \theta \bmod (\pi/2), & |\theta| > \pi/2, \\ \theta, & |\theta| \le \pi/2 \end{cases}$$

The sizes of the new image are from the following formulas:

$$A' = \sin(|\beta| + \frac{\pi}{2} - \alpha) * d,\ B' = \sin(\alpha + |\beta|) * d,$$

where A', B' are the width and height of the new canvas, α is the angle between the base and diagonal of the input image, θ is the turning angle, d is the diagonal.

Then every pixel of the input matrices image by the affine transformations is transferred to the new canvas:

$$\begin{pmatrix} 1 & -\frac{A'}{2} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\frac{B'}{2} & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \frac{B'}{2} & 1 \end{pmatrix} \begin{pmatrix} 1 & \frac{A'}{2} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

An example of the turned PCB image is shown in Fig. 23 where an extra area is marked with red.

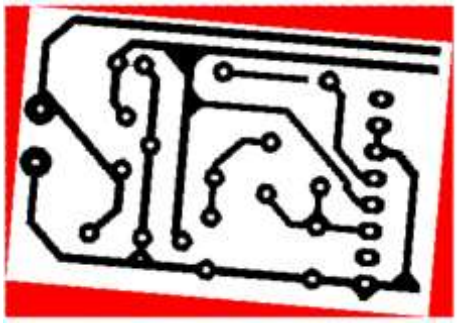


Fig. 23. Turned PCB image and extra area in canvas

In the rotated image all intensity, color properties, and center of the image are the same as in the input image. Geometrical properties measured horizontally and vertically are changed.

D. Building of image of distributed cumulative histogram

For all pixels' intensity values in rows and columns of the image matrix are stored (transparent positions are not accepted):

$$m_i[256] = \{0\},\ i = \overline{1,B},\ m_j[256] = \{0\},\ j = \overline{1,A},$$

$$m_i[R_{ij}] += \begin{cases} 1, & \alpha_{ij} \ne 0 \\ 0, & \alpha_{ij} = 0 \end{cases},\ i = \overline{1,B},\ j = \overline{1,A},$$

$$m_j[R_{ij}] += \begin{cases} 1, & \alpha_{ij} \ne 0 \\ 0, & \alpha_{ij} = 0 \end{cases},\ i = \overline{1,B},\ j = \overline{1,A},$$

where $m_i[256]$, $m_j[256]$ are histograms data in the rows and in the columns, R_{ij} is the color component, α_{ij} is the transparency value.

For two or more rows (columns) of the image matrix the procedure is complicated because average values are taken into account:

$$m_i[R_{(i*E+k)j}]+=\begin{cases}1, A_{(i*E+k)j}\neq 0\\ 0, A_{(i*E+k)j}=0\end{cases}, \ i=\overline{1,\frac{B}{E}}, k=\overline{1,E}, j=\overline{1,A}$$

$$m_j[R_{i(j*E+k)}]+=\begin{cases}1, A_{i(j*E+k)j}\neq 0\\ 0, A_{i(j*E+k)j}=0\end{cases}, \ i=\overline{1,B}, k=\overline{1,E}, j=\overline{1,\frac{A}{E}}.$$

Then the cumulative histograms are calculated for all rows (columns):

$$M_{ik}=\sum_{j=1}^{k}m_{ij}, \ i=\overline{1,B}, k=\overline{0,255},$$
$$M_{kj}=\sum_{i=1}^{k}m_{ij}, \ j=\overline{1,A}, k=\overline{0,255},$$

where m_i, m_j represent the values of the intensity histograms, M_i, M_j are the values of cumulative histograms called distributed cumulative histograms.

The following step is to draw an image of the distributed cumulative histogram (DCH). The value of each pixel of this image corresponds to a value of the cumulative histogram. The new image will have a $B\times N$ pixel size (for rows) with the pixel values are calculated by the following formulas:

$$R'_{ij}=G'_{ij}=B'_{ij}=M_{ij}*\frac{255}{\max\limits_{1\le j\le N}M_i}, i=\overline{1,B}, \ j=\overline{1,N}.$$

For columns, the new image is of $N\times A$ size and the pixel values are calculated by the similar formulas:

$$R'_{ij}=G'_{ij}=B'_{ij}=M_{ij}*\frac{255}{\max\limits_{1\le i\le N}M_j}, i=\overline{1,N}, \ j=\overline{1,A},$$

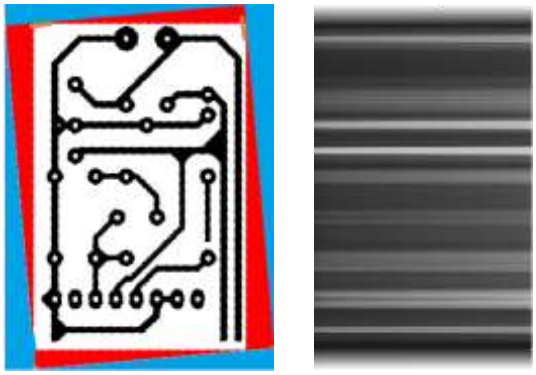
where $R'_{ij}, G'_{ij}, B'_{ij}$ are the pixel components in the DCH image, M_i is the cumulative histogram of i-row, M_{ij} is the value of j-element of the histogram, N is the number of elements in the cumulative histogram.

The 0 or 255 values of the DCH image pixels are replaced by transparent pixels ($\alpha=0$)

$$\alpha'_{ij}=\begin{cases}0, & R'_{ij}\in\{0,255\}\\ 255, & R'_{ij}\notin\{0,255\}\end{cases}, i=\overline{1,B}, \ j=\overline{1,A},$$

where A', B' are the DCH image sizes.

To illustrate the rotation results, the PCB image turned by some angle is taken for experiments. The angle is determined, and the image is rotated backward for this angle. During the rotation, the second extra area has arisen and is shown in Fig. 24a with blue.



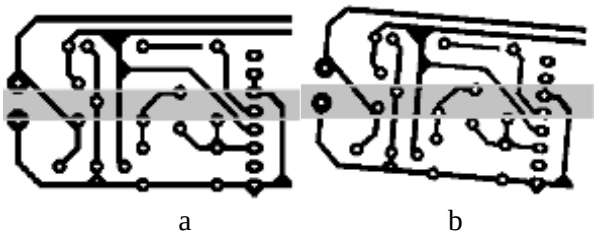
a b

Fig. 24. Input image rotated back (a) and its DCH image from OY (b)

The new blue area appears when the new canvas is constructed. The sizes of the real PCB image are the same as those for the input image. After cutting (cropping) or flood-filling of blue and red areas images are processed by defects detection algorithms. The resulting PCB image after rotating backward and its DCH image are shown in Fig. 24.

E. Alignment by DCH images

DCH images are useful for adjusting the position of an unaligned image. In Fig. 25 two images with marked parts are shown. These marked parts are cropped and their DCH images are built. Areas for cropping are chosen as $3-5$ rows of pixels in the middle of the image. A direction of cropping is chosen to intersect a larger number of elements. As a rule, along the larger side.



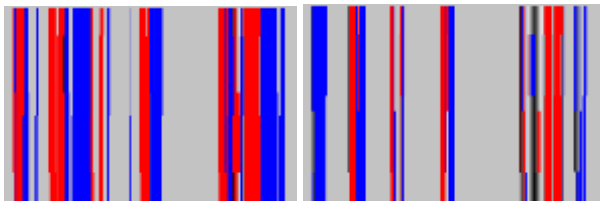
a b

Fig. 25. Central parts in input image (a) and turned image (b)

The DCH images are calculated and built from the OX axis for cropped parts. Comparison by mathematical subtraction of these images gives up

the result image with many red and blue areas meaning no coinciding areas in the input images.

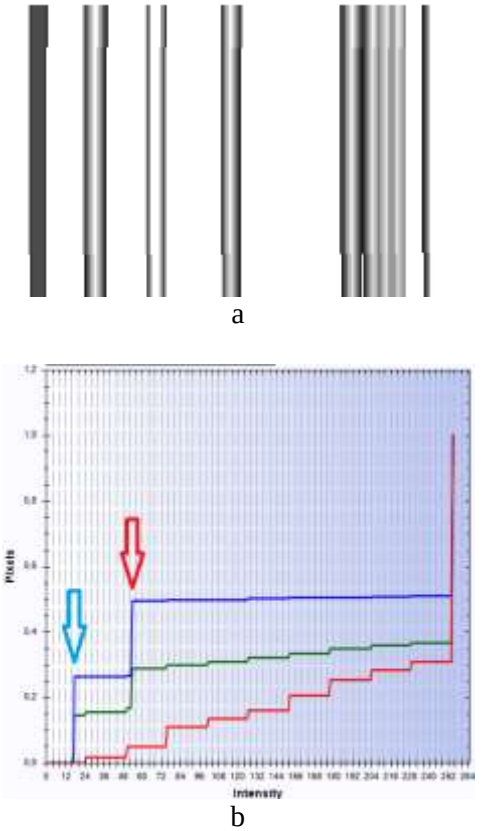
Two differences in DCH images built for the 5-degree angle and 1-degree angle of the turned PCB strips are shown in Fig. 26.



a b

Fig. 26. Differences in DCH images: 5-degree angle (a) and 1-degree angle (b)

When the angle equals 0-degree the difference fully coincides with the input pattern. It is shown in Fig. 27a. The cumulative histogram is applied to measure this difference. There are three histograms in Fig. 27b: red for the difference of $0-0$, green for $0-1$, and blue for $0-5$ angles.



a

b

Fig. 27. Difference between DCH images of aligned and reference strips (a) and cumulative histograms from the *OY* axis (b) of DCH images

The green graph is close to the required one. Only blue and red pixels in the image form steps indicating the main contribution to the difference. The blue graph is far away indicating absolutely no coincidence with the required result. Histograms give the number of blue and red pixels. Thus, this characteristic is accepted to adjust alignment to the required position.

5. Conclusion

The approach is based on the K-mean and hierarchical clustering algorithms. The first one is applied to segment PCB images into a binary form. The second algorithm is used to separate all contacts, traces, and chains. Separated elements of two printed circuit boards are compared. Connectivity and intensity defects are detected and measured. They are visually presented by the mathematical comparison and in images of distributed cumulative histograms. Three approaches for the alignment of the PCB image based on the determination of angles are considered. The image rotation algorithm to the reference view is developed. Images of distributed cumulative histograms help to adjust angles of alignment and mark possible defects and inaccuracies.

References

[1] V. A. Adibhatla, H.-C. Chih, C.-C. Hsu, J. Cheng, M. F. Abbod, and J.-S. Shieh, Defect Detection in Printed Circuit Boards Using You-Only-Look-Once Convolutional Neural Networks, *Electronics*, Vol.9, No.9, 2020, pp. 1547.

[2] S. McClure. Extracting and Classifying Circuit Board Defects using Image Processing and Deep Learning, *towardsdatascience.com*, Feb. 4, 2020 [Online]. Available: https://towardsdatascience.com /building-an-end-to-end-deep-learning-defect-classifier-application-for-printed-circuit-board-pcb-6361b3a76232].

[3] Jungsuk Kim, Jungbeom Ko, Hojong Choi and Hyunchul Kim, Printed Circuit Board Defect Detection Using Deep Learning via A Skip-Connected Convolutional Autoencoder, *Sensors*, No.21(15), 2021, 4968.

[4] Zhu, A. Wu and X. Liu, Printed circuit board defect visual detection based on wavelet denoising, *IOP Conference Series: Materials Science and Engineering*, Vol. 392, 2018, pp. 062055.

[5] Y. Hanlin and W. Jun, Automatic Detection Method of Circuit Boards Defect Based on Partition Enhanced Matching, *Information Technology Journal*, Vol.12, No.11, 2013, pp. 2256-2260.

[6] Vikas Chaudhary, Ishan R. Dave and Kishor P. Upla, S. V., Visual Inspection of Printed Circuit Board for Defect Detection and Classification. *International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2017, pp. 732-737.

[7] M. H.Tatibana, R. and de A. Lotufo. Novel Automatic PCB Inspection Technique Based on Connectivity, *Proceedings X Brazilian Symposium on Computer Graphics and Image Processing*, 1997, pp. 187-194.

[8] K. P. Anoop, N.S. Sarath and V. V. Sasi Kumar, Review of PCB Defect Detection Using Image Processing, *International Journal of Engineering and Innovative technology (IJEIT)* Vol.4, Is.11, 2015, ISSN: 2277-3754.

[9] D.B. Anitha, and M. Rao, A survey on Defect Detection in Bare PCB and Assembled PCB using Image Processing Techniques, *International Conference on Wireless Communications, Signal Processing and Networking*, 2017, pp. 39-43.

[10] M. Moganti, F. Ercal, C. H .Dagli, and S. Tzumekawa, Automatic PCB inspection algorithms: A review, *Computer Vision and Image Understanding*, Vol.63, No2, 1996, pp. 287-313.

[11] F. B. Nadaf and V. S. Kolkure, Detection of Bare PCB Defects by using Morphology Technique, *Morphology Technique International Journal of Electronics and Communication Engineering*. Vol.9, No.1, 2016, pp. 63-76.

[12] Y. Hanlin, W. Jun, Automatic Detection Method of Circuit Boards Defect Based on Partition Enhanced Matching, *Information Technology Journal*, Vol.12(11), 2013, pp. 2256-2260.

[13] J. Nayaka, K. Anitha, B.D. Parameshachari, R. Banud and P. Rashmi, PCB Fault Detection Using Image Processing, *IOP Conference Series: Materials Science and Engineering*, Vol.225, 2017, pp. 1-5.

[14] J. P. R. Nayak, K. Anitha, B. D. Parameshachari, R. Banu, and P. Rashmi, PCB Fault Detection Using Image Processing, *IOP Conference Series: Materials Science and Engineering*, Vol. 225, 2017, pp. 012244.

[15] S. Guan, F. Guo, A New Image Enhancement Algorithm for for PCB Defect Detection, *International Conference Intelligence Science and Information Engineering (ISIE)*, 2011, pp. 454-456.

[16] Deepanshu Tyagi, Introduction to SURF (Speeded-Up Robust Features) Mar 20, 2019, https://medium.com/data-breach/introduction-to-surf-speeded-up-robust-features-c7396d6e7c4e

[17] Edouard Oyallon, Julien Rabin. An Analysis of the SURF Method, IPOL, Image Processing On Line on 2015–07–20. https://doi.org/10.5201/ipol.2015.69 pp.176-218.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors contributed in the ratio of 70 (Melnyk) to 30 (Kvit) in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.