Journal of Achievements in Materials and Manufacturing Engineering

International Scientific Journal published monthly by the World Academy of Materials and Manufacturing Engineering

Feature extraction and template matching algorithm classification for PCB fiducial verification

C.L.S.C. Fonseka *, J.A.K.S. Jayasinghe
Department of Electronics & Telecommunication, University of Moratuwa,
Moratuwa 10400, Sri Lanka
* Corresponding e-mail address: ee05sameera@gmail.com

ABSTRACT

Purpose: Automatic Optical Inspection (AOI) systems that are extensively used in the industry of Electronics Manufacturing Services (EMS), performs the inspection of Surface Mount Devices (SMD). One of the main tasks of such an AOI system is to align a given PCB to the parameters of the corresponding PCB positioning system by a process called fiducial alignment. However, no detailed analysis has been carried out so far on the methodologies that can be used to have a very precise identification of PCB fiducial points. In our research, we have implemented an AOI system for the inspection of soldering defects of Through Hole Technology (THT) solder joints, which can be integrated to a desktop soldering robotic platform. Such platforms are used in environments where no specific lighting conditions can be provided within a surrounded atmosphere. Therefore, an AOI system that is capable of performing fiducial alignment of any given PCB under varying lighting condition is highly preferred. In this paper, we have presented a detailed analysis on feature extraction and template matching algorithms that can be used to implement a very precise fiducial verification process under normal lighting condition.

Design/methodology/approach: A detailed analysis and performance evaluation have been carried out in this paper on prominent image comparison algorithms that are extensively used in the field of image processing.

Findings: According to the analysis carried out in this paper, it could be observed that the combination of feature extraction and template matching algorithms gives the best performance in PCB fiducial verification process.

Research limitations/implications: This paper only presents the implementation of the front end of our proposed AOI system. The implemented methodologies for the automatic identification of soldering defects will be discussed in separate research papers.

Practical implications: The methodologies presented in this paper can be effectively used to implement a very precise and robust PCB fiducial verification process that can be efficiently integrated to a desktop soldering robotic system.

Originality/value: This research proposes a very accurate fiducial verification process that can be used under varying lighting conditions on a wide range of different PCB fiducial points.

Keywords: Automatic Optical Inspection, Feature extraction, Template matching, SIFT, SURF, FAST, Box filtering, FLANN

Reference to this paper should be given in the following way:
C.L.S.C. Fonseka, J.A.K.S. Jayasinghe, Feature extraction and template matching algorithm classification for PCB fiducial verification, Journal of Achievements in Materials and Manufacturing Engineering 86/1 (2018) 14-32.

ANALYSIS AND MODELLING

1. Introduction

In modern electronics manufacturing industry, automated production process is preferred due to increasing cost for labour, skill dependency of human operators, less availability of manpower and etc. Soldering robotic systems [1] are one of the versatile machines, which have been developed to fulfil the requirement of a human operator to perform soldering process automatically. These systems are useful when soldering the components, which cannot be soldered through wave soldering and selective soldering machines [2], due to low temperature, withstand capability. However, none of the commercially available soldering robotic systems is equipped with real time quality inspection capability. The soldering robotic system with an integrated automatic optical inspection (AOI), will provide real time quality inspection capability over solder joints while improving the overall efficiency of the system. We have presented detailed analysis that covers the first stage of the AOI platform, where the solder pad area is precisely identified from the PCB surface in [3]. There the colour models that render optimum outcome for different solder pads and PCB colours have been evaluated based on their accuracy in transforming foreground and background colour vectors with much larger vector length difference. It could be proved that the behaviour of the same colour model varies with the varying PCB colour and the solder pad type. In that paper, methodologies for eliminating the effect of uneven light distribution over the solder pad area and offsets between the PCB surface and the solder pad area were informatively presented.

Even though the positioned solder pads can be accurately identified by the vision system as presented in our paper [3], it still cannot differentiate between solder pads on different PCB types. Our proposed AOI system has been designed to produce a fully automated desktop soldering robotic system. Therefore, the vision system must be capable of precisely identifying whether the PCB under solder (PUS) is the correct one that should be soldered by the robotic platform or not. This task can be accomplished by comparing user defined master points, which are also known as fiducial points, extracted from model images taken from the golden sample (GS) PCB, and input images taken from PUS. This process is referred to as fiducial verification. In this paper, we present a detailed analysis of extensively used feature extraction and image comparison algorithms for the implementation of fiducial verification process. The fiducial verification process is not only useful for differentiating between different PCB types, but also for aligning PUS to the robotic system platform. This procedure is also known as fiducial alignment, which is the process of estimating relative distances to all the points on a PCB with respect to the calculated distances from the system origin to the defined fiducials on the PCB. Minimum of two fiducial points must be defined from the computer aided design (CAD) file relevant to that particular PCB in order to have an accurate calculation of relative distances to rest of the points on the PCB.

In this paper, Section 2 presents a detailed overview on commonly used feature extraction algorithms in various automatic vision applications. Section 3 presents the performance evaluation of feature extraction algorithms described in section 2, along with commonly used distance matching methods for feature matching between model and input images for the implementation of fiducial verification process. Section 4 describes the analysis and performance evaluation of template matching algorithms for the accurate localization of model image (I_M) area within the input image, once the existence of model image inside the input image is verified by the selected feature extraction algorithm described in section 3. We have presented the conclusions made on the analysis carried out in section 2 to 4 under section 5. This section also presents a brief overview on the implementation of quality analysis stage for the automatic quality assurance of a solder joint.

2. Algorithms for fiducial verification

Computer vision can be mainly carried out by two methods.

- Template matching,
- Feature extraction.

In template matching method [4-8], the occurrence of a model image inside the input image is computed by convolution between model image and input image. Various algorithms have been proposed for template matching like CCOEFT, normalized CCOEFT, CCORR, normalized CCORR, SQDIFF and normalized SQDIFF etc. [9,10]. The resulted image from the above algorithms contain a global optima which corresponds to the best matching point as illustrated in Figure 1. Therefore, the application of template matching algorithm does not provide precise outcome, when the existance of the model image is not exactly known prior to the use of these algorithms. Therefore, it should be clear that fiducial verification process cannot be done effectively using above template matching algorithms according to the images provided in Figure 1a.

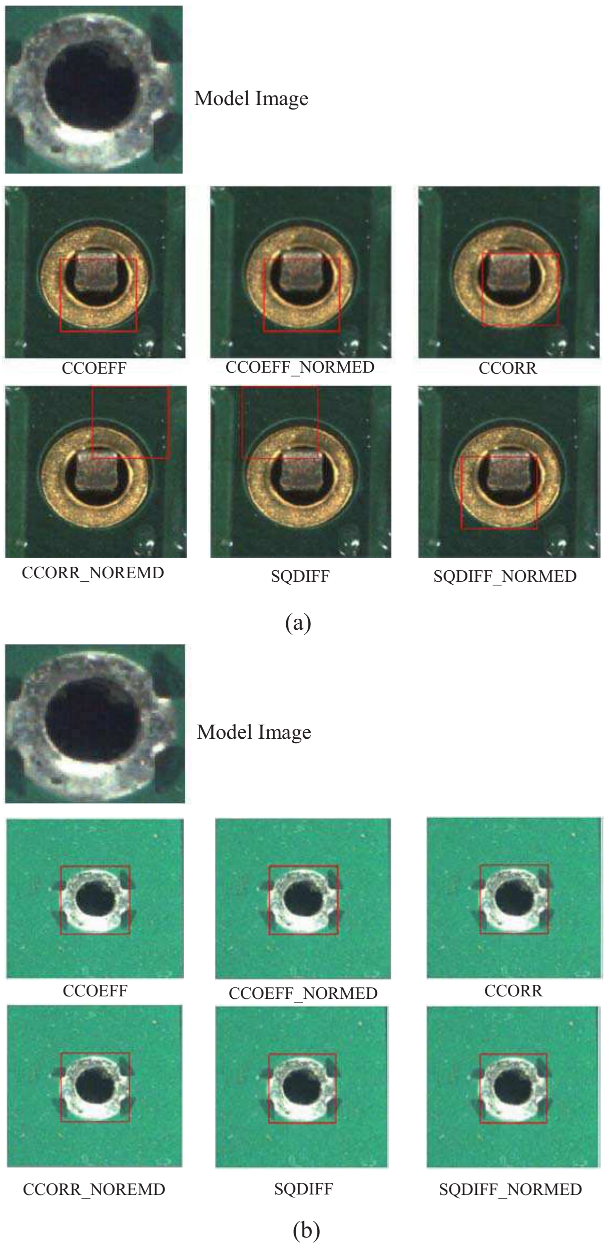


Fig. 1. Best matching points detected by various template matching algorithms. (a) Matching of model image with an input image from a different PCB type. (b) Matching of model image with an input image from same PCB type

In feature extraction methods, features are identified from a model image and their subsequent matching in the input image are found. There is no clear definition for 'feature', but there are some properties that can be evaluated at a given point, to decide whether that particular point/area is a good feature or not [11].

A feature should be:

- perceptually meaningful,
- analytically special (Ex. minimum or maximum point in a given data set),
- easily distinguishable in different images,
- invariant to scale, orientation, illumination and etc.,
- insensitive to noise.

In order to compare images based on their features, many research works have been carried out. They mainly rely on extracting features/key points in a given model image and find their corresponding matching in input image at different scales and orientations [12]. Various algorithms have been proposed for the extraction of features of a given image, like kadir and brady algorithm [11,13], SIFT [8,10,14], SURF [15-17], FAST [18], Principle component analysis (PCA) [19], Independent component analysis (ICA) [20] and etc. In this study, SIFT, SURF and FAST algorithms have been taken into consideration because of their proven robustness, accuracy and fast detection of key points in an image. Figure 2 illustrates feature matched images using above stated algorithms.

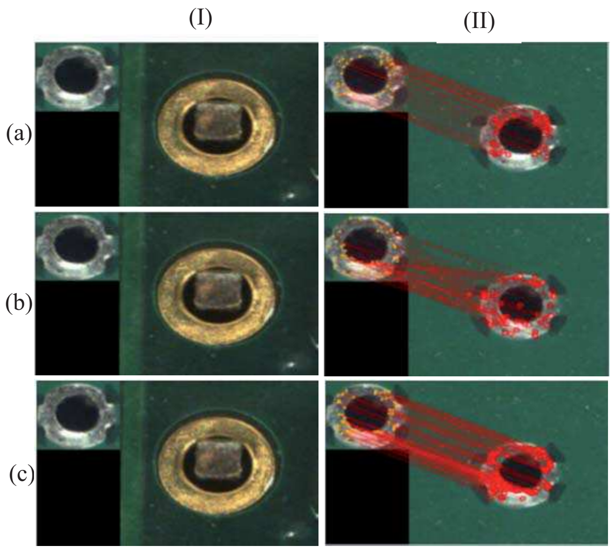


Fig. 2. Feature matched images using SIFT (row a), SURF (row b) and FAST (row c) for wrong PCB (column I) and correct PCB (column II). Red lines depicts matched features

According to Figure 2, it can be seen that feature matching algorithms render better outcome compared to template matching algorithms (Fig. 1) when assuring the existance of model image inside a located image. Next section presents a brief explanation on SIFT, SURF and FAST algorithms respectively.

2.1. SIFT

SIFT, Scale Invariant Feature Transform, is a method of extracting of distinctive invariant features from a model image that can be efficiently used to perform accurate matching between different views of that model image occurred inside a given input image [21]. The detected features are proven to be invariant to scale, rotation and provide robust matching across a substantial range of affine distortion, noise, and illumination variation [21]. This algorithm can be described as a combination of following sub stages, where each stage adds a weighted outcome to the next stage.

- Constructing a scale-space: *implement a scale invariant platform by computing the difference of blurred images using a gaussian kernel to blur the original image and a set of scaled down images.*
- Key point localization and selection of descriptive key points: *Localization of key points extracted from generated Difference of Gaussian (DoG) images from scale space and refinement of the detected keypoints based on their stability and repeatability of detection.*
- Orientation assignment: *One or more orientations are assigned to each key point locations based on local image gradient directions in order to have a rotational invariance.*
- Generation of SIFT features: *Descriptor to each located key point is generated that makes them invariant to local shape distortion and illumination variation.*

Constructing a scale-space

An inherent property of real-world objects is that they only exist as meaningful entities over certain ranges of scale. A simple example is the concept of a branch of a tree, which makes only sense in a range of few centimetres to few meters. It is meaningless to discuss this example at nanometre or kilometre level. At those levels, it is more convenient to discuss about the molecules in a tree leave and the forest where the tree grows respectively. The construction of scale space attempts to replicate this concept to digital images. The first step of key point detection is to identify locations and scales that can be repeatedly assigned under different views of the same object. In this process, the points must be localized in a way that they do not change their way of existence in the original image at different scales. These stable features can be estimated with the use of a continuous function of scale, referred as scale space [21].

SIFT uses Gaussian function [22] to generate its scale space structure by progressively blurred out images resulted from the original image and a set of scaled down images by a factor of 2 [8,14,21,23]. The generation of a Gaussian blurred image is illustrated in Eq. (1).

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \qquad (1)$$

where,
$L(x,y,\sigma)$: Resulted image from the convolution between the input image, $I(x,y)$ and gaussian kernel, $G(x,y,\sigma)$.

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

Here x, y are the spatial coordinates in 2-D space and σ is the standard deviation of the gaussian kernel. Fig. 3 illustrates the first two layers that contain progressively blurred out Gaussian images in SIFT scale space structure.

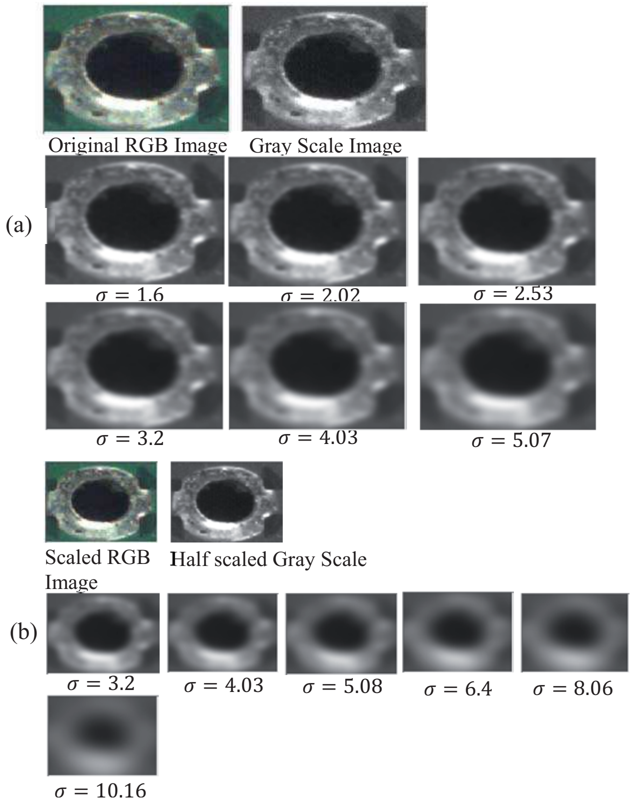


Fig. 3. Few samples of scale space. (a) Resulted images after convolving the original image with Gaussian kernel at five different σ values where σ = 1.6, 2.02, … , 5.07; (b) Resulted images after convolving the scaled down image by factor of 2 with gaussian kernel at five different σ values where σ = 3.2, 4.03, … , 10.16

Once the Gaussian blurred images are computed, scale-space is generated as illustrated in Figure 4.

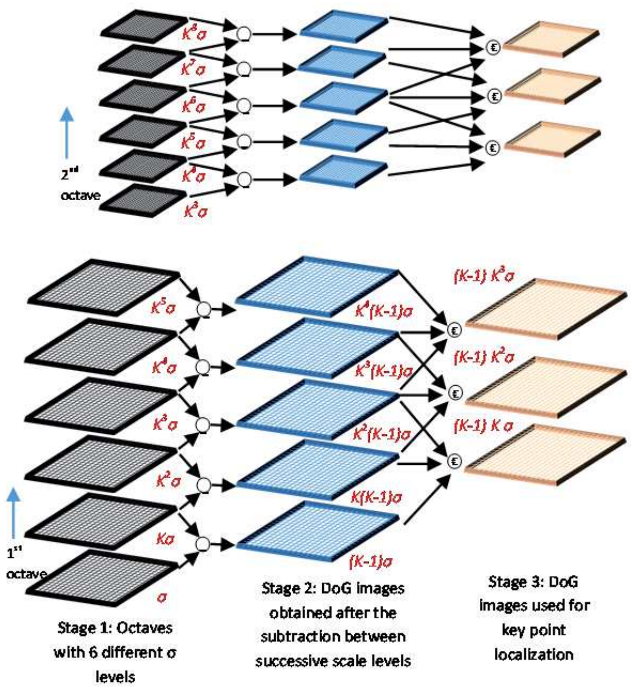


Fig. 4. For each octave (set of Gaussian blurred images obtained from Gaussian function as illustrated in Eq. (1) at different σ in scale space, initial image is convolved with Gaussian kernel at different σ and the adjacent gaussian images are subtracted to obtain DoG images. Each black square illustrates a Gaussian blurred image with $k^n\sigma$, where *n=0,1,2.....* Each blue square illustrates computed DoG image by subtracting two adjacent Gaussian blurred images ($k^{n+1}\sigma$ and $k^n\sigma$). Each brown square illustrates a resulted image after neighbour pixel comparison method

In SIFT, difference of Gaussian images (DoG) are computed to detect stable feature locations in scale-space. Eq. (2) illustrates how this approximation is computed on a given set of Gaussian smoothed images [21].

$$\begin{aligned} D(x,y,\sigma) &= (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \\ &= L(x,y,k\sigma) - L(x,y,\sigma) \end{aligned} \tag{2}$$

where,

$D(x,y,\sigma)$	: Resulted image from DoG,
$G(x,y,k\sigma)$	: Gaussian kernel at kσ,
$G(x,y,\sigma)$	: Gaussian kernel at σ,
$L(x,y,k\sigma)$	: Gaussian blurred image at kσ,
$L(x,y,\sigma)$	: Gaussian blurred image at σ,
$I(x,y)$	: Input image.

Here, $k = 10^{\log 2/n}$. n is the number of Gaussian blurred images per octave. This process is illustrated in stage 2 of Figure 4.

Key point localization and selection of descriptive key points

In order to estimate, whether a particular point is a key point, it must be differentiable from its neighbours. This is accomplished in SIFT in a way that, a particular pixel in a DoG image is compared with reference to its eight neighbours and the corresponding nine neighbours possessed by the above and below DoG images to determine whether that particular point is a local extrema or not [14,21]. Particular sample point will be decided as a local extrema, only if it is the maximum or minimum compared to its 26 neighbouring points [21]. Stage 3 of Figure 4 illustrates how this process is accomplished.

Once the algorithm localizes the key point with the proposed neighbour comparison method over successive DoG images, it is required to find a detailed relationship to nearby data for location, scale and ratio of principle curvatures [21]. This information is used by the algorithm to eliminate key points with unstable properties. SIFT uses following procedure to identify the stability of the detected key points.

- The algorithm finds the sub pixel accuracy of detected keypoints using Taylor expansion [21];
- Then the algorithm performs contrast comparison of detected keypoints against a predetermined threshold value [21]. If the magnitude of the function computed from Taylor expansion at the keypoint location is below the defined threshold, it is discarded [21];
- Finally the algorithm discards keypoints that lie on the edges of the image. This process is carried out because DoG images contains a strong edge response, even if the location along an edge is poorly determined [14,21]. Therefore, they are more unstable to small amount of noise. The task of neglecting key points along an edge is determined by computing Principle Curvatures (PC) of resulted DoG image. A detailed explanation on this process and how this is accomplished using hessian matrix [24-26] is provided in [21].

At the end of this refinement process of detected key points, it is said the remaining key points are invariant to scale of the image [14,21]. The next step is to compute the orientation of remaining set of key points, in order to make them rotational invariant.

Orientation assignment

A good feature extraction algorithm must be capable enough to detect same features inside the rotated versions of the same image which the key points were originally detected. Therefore, it is required to make a detected key point invariant to rotation. In SIFT, rotational invariance is achieved by assigning orientation to each key point based on local image properties. The estimation of the orientation of a given key point (θ_{ref}) is performed by calculating the gradient magnitudes and directions around a particular key point within a defined neighbourhood inside the respective Gaussian blurred image ($L(x, y, k^n\sigma)$). The selection of $L(x, y, k^n\sigma)$ is clearly explained in [21]. The orientation assignment process is performed by calculating the gradient magnitude, $m(x, y)$ and the orientation, $\theta(x, y)$ based on the pixel differences over each pixel that falls within a defined circular neighbourhood of the key point. The computation of $m(x, y)$ and $\theta(x, y)$ is illustrated in Eq. (3) and Eq. (4) respectively [14,21].

$$m(x,y) = \sqrt{\left(L(x+1,y) - L(x-1,y)\right)^2 + (L(x,y+1) - L(x,y-1)^2} \quad (3)$$

$$\theta(x,y) = \tan^{-1}\left((L(x,y+1) - L(x,y-1))/ /\left(L(x+1,y) - L(x-1,y)\right)\right) \quad (4)$$

The radius of the circular neighbourhood depends on the scale of the key point and it is estimated as $1.5 \times scale$ [21], where $scale$ can be defined as $k^n\sigma$ where $n = 1,2, \dots. p$. Once this process is completed, an orientation histogram is calculated that has 36 bins, covering 360° range of orientation, where the starting value of a bin can be expressed in general as $\theta_b = 2\pi(b-1)/36$, where $b = 1,2 \dots 36$. A detailed explanation of this process is provided in [21].

Generation of SIFT features

Algorithm has generated key points, which have locations, scale, and orientation that will make them invariant to scale and rotation, up to this stage of processing. In real time images, there will be lots of illumination variation and 3-D object transformations. Therefore, there must be a way of making a particular key point independent to these changes. At the final step of SIFT algorithm, an unique identifier is generated based on the local image region, that will make a scale and rotational invariant key point to be distinctive over above stated variations. In order to create SIFT descriptor for a key point (x_k, y_k), a square neighborhood with a width of $12k^n\sigma$ is centered at the keypoint, in the direction of θ_{ref} inside its respective gaussian smoothed image. Then this region is sub divided into 4×4 sub regions, which has a width of $3k^p\sigma$ in each dimension. Then local gradients are computed inside each sub region. Then an 8-bin orientation histogram is computed for each sub region. In this case, the particular key point will have a descriptor that contains 128 ($4 \times 4 \times 8$) feature vectors. A detailed explanation on how this process is accomplished is provided in [21]. The detected key points on several model images by SIFT algorithm are illustrated in Figure 5. According to Figure 5, it can be clearly seen that over 95% of the detected key points lie over the prominent object locations inside the input image. The performance variation of SIFT under different conditions will be evaluated in section 3.

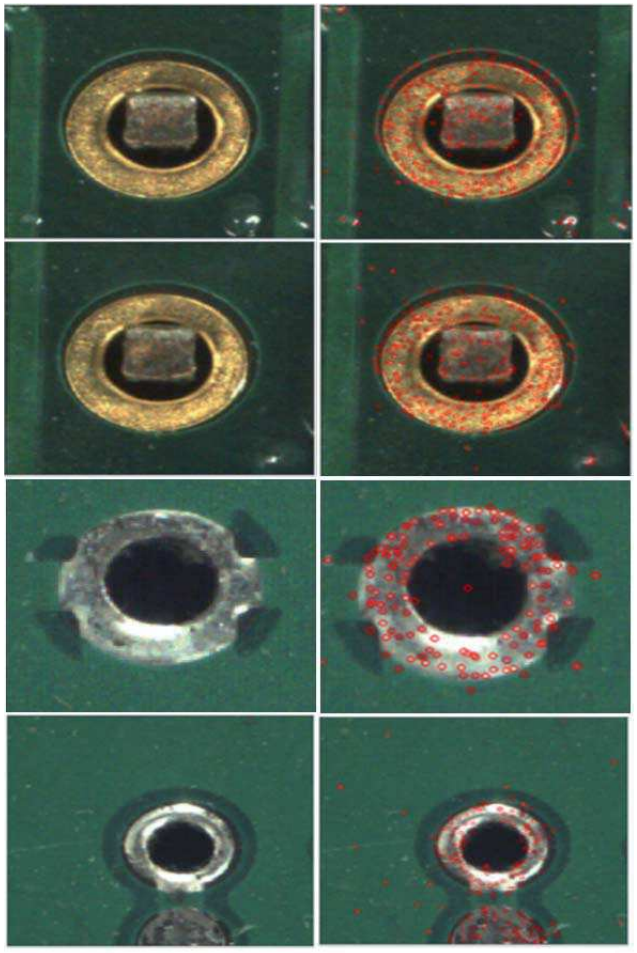


Fig. 5. Detected key points on solder pads, vias and test pads on different PCBs at number of scales per sample = 3, contrast threshold = 0.03, edge threshold = 10 and starting scale (σ) =1.6 using SIFT feature detector

2.2. SURF

SURF, Speed Up Robust Features, is a fast and robust algorithm developed by H. Bay, for the detection of local invariant features and finds their subsequent matching in a provided scene [15-17]. SURF has been developed with the purpose of providing a fast computation on a given image, which makes it an ideal algorithm for real time applications like tracking and object recognition [16]. Similar to SIFT, SURF also relies on scale space representation for the detection of features that are scale and rotational invariant. This algorithm has been proved to render better results with higher speed and accuracy compared to SIFT [16]. The implementation of SURF will be briefly discussed in four steps in this paper.

- Generation of integral images and box filters: *This contains the generation of integral images and box filters to approximate gaussian second order partial derivatives.*
- Constructing a scale-space: *implement a scale invariant platform with the use of box filters together with integral images.*
- Key point Localization: *This contains the localization of stable key points with the use of hessian matrix in box filter space and computation the orientation of detected features.*
- Generation of SURF feature description: *This contains the generation of a unique identifier for each detected key point as computed in SIFT.*

Generation of integral images and box filters

In SIFT, scale-space is generated from the convolution between a given input image and the Gaussian kernel at different scale levels. It is not efficient in real time applications due to the computational complexity of Gaussian kernel [15-17]. As a way of speeding up this process, SURF algorithm came up with an alternative method. It takes the advantage of using integral images [27] together with rectangular shape uniform kernels, referred to as “box-filters” [17,28], in a way that their combination approximates the role of Gaussian kernels in a much efficient way [15,17]. Eq. (5) illustrates how the pixel value of an integral image U at a point (x, y), is computed from the corresponding input image *I*.

$$U(x,y) = \sum_{0\le i\le x} \sum_{0\le j\le y} I(i,j) \qquad (5)$$

where,
$U(x,y)$: Value of integral image at (x,y)
$I(i,j)$: Value of input image *I* at (i,j)

According to Eq. (5), it is quite clear that value of an integral image at point (x,y) means the summation of pixels above that point in both *x* and *y* directions inside *I*. It can be proved that it takes only three additions and four memory lookups to calculate the sum of pixel intensities contained by any upright rectangular area formed inside I, once the integral image is computed. SURF takes this advantage and approximates LoG with box filters in generating the scale space. The implementation of box filters is a complex process. A detailed explanation on how the box filters are computed is presented in [17].

In SIFT, the scale space is generated by down scaling the original image at each octave while smoothing that down scaled image at different σ levels. But with the use of box filters together with integral images, there is no requirement in SURF to iteratively apply the same filter to the output of previously filtered layer [15]. SURF applies box filters of any size at exactly the same speed directly on the original image and even in parallel [15]. Therefore, the scale space (also known as box space) is generated in SURF, by up scaling the filter size rather than iteratively reducing the image size [15]. This procedure outperforms SIFT in terms of decreasing computational complexity in generating scale space, while preserving high frequency components that can be lost in zoomed-in variants of the original image. As in SIFT, SURF also creates different octaves (Fig. 4 stage 1) where each octave represents a series of filter response maps obtained by convolving same input image with a filter of increasing size as described above. Figure 6 illustrates how the box filter space structure in SURF looks like.

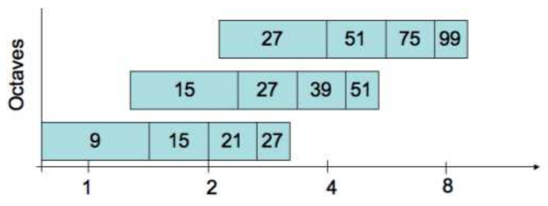


Fig. 6. Graphical representation of how the size of the box filter varies within three octaves

We have so far presented how the box space generation is accomplished in SURF. Next step is to understand how the stable key points are detected from the generated box space in SURF.

Key point localization

Once the box space is generated, the next step is to localize stable key points from the box space. The procedure for detecting the key points can be explained in three major steps in SURF [17]:

- Feature filtering: *this is achieved with the use of scale-normalized determinant of hessian.*
- Feature selection: *this is achieved with the combination of non-maxima suppression and thresholding.*

- Scale space location refinement: *this is achieved with the use of second order interpolation.*

Each of these steps contains complex computations. Therefore, a detailed explanation of these methods does not add a weighted outcome to our paper. A detailed explanations on the implementation of each of these steps can be found in [17].

Generation of SURF feature descriptor

The descriptor for a detected key point implemented in SURF, describes the intensity content within the key point neighbourhood. This procedure is accomplished in a similar way followed in SIFT and its variants for the extraction of gradient information on a particular key point. The process of defining an unique identifier/descriptor in SURF is based on two major steps.

- Orientation assignment around the key point to achieve rotational invariance,
- Generating SURF descriptor based on the orientation details to give a distinctive identity to each key point.

As described earlier in the key point localization, the orientation assignment and generation of SURF descriptor contain complex mathematical computations. we do not go into more details of the implementation of these methods, since providing a detailed explanation of the implementation of these methods does not fall in the scope of this paper. A complete description of the implementation of these methods is available in [17]. The SURF descriptor is also a 16×4 vector that consists of mean and absolute mean values of gradients extracted from a spatial grid (R) divided into 4×4 sub regions [17] as described in SIFT. The detected key points on several model images by SURF feature detector are illustrated in Figure 7.

According to Figure 7, it can be clearly seen that over 95% of detected key points lie on the object regions and distinctive areas in a given image. The performance variation of SURF under different conditions and the accuracy of identifying similarities between model image and input image will be evaluated later in this paper. We have so far described very complex algorithms that have been developed keeping a close approximation to human vision in terms of having rotational and scale invariance of detected features at a considerable amount of computation cost. We will learn about a fast feature detection algorithm that have been developed especially targeting for platforms with less computational power, but require high speed detection of features.

2.3. FAST

FAST, Features from Accelerated Segment Test [18,29,30], is a fast corner detection algorithm developed by Edward Rosten and Tom Drummond. The main idea behind the development of FAST algorithm is to have a fast feature detector for real time applications, which are running on platforms with limited computational power [29]. In our application, the speed of detecting key points does not play a major role, since the fiducials of a given PCB are verified only once for that particular PCB. We mainly focused on robustness and accuracy of this algorithm to locate most stable features from a given input image, which can be differentiable from other points on the same PCB and other PCB types.

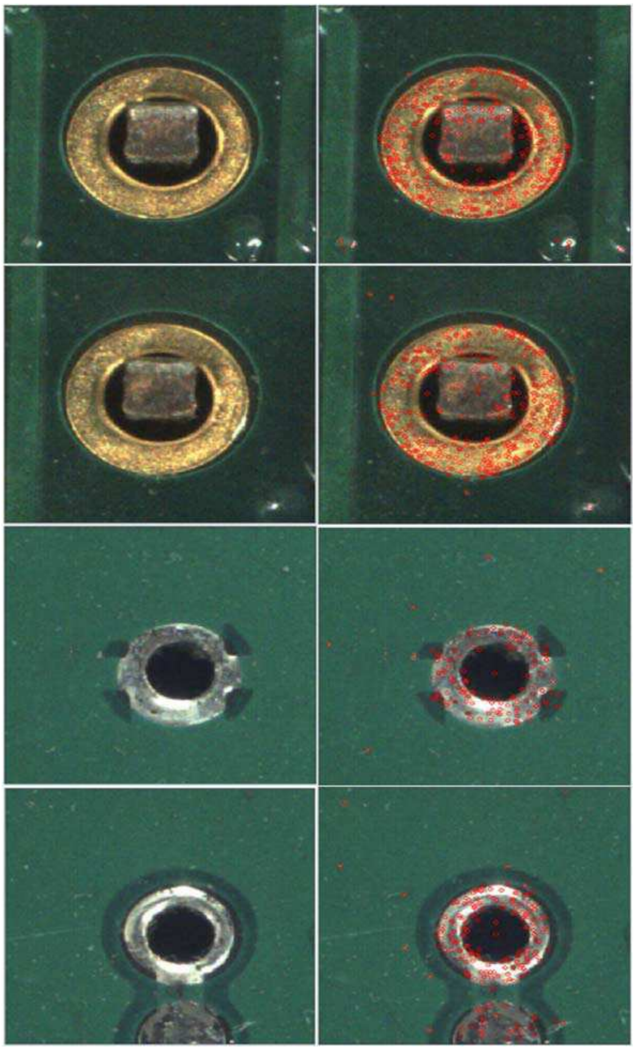


Fig. 7. Detected key points on solder pads, vias and test pads on different colour PCBs at no. of octaves = 4, threshold = 500, starting box filter size $= 9 \times 9$ using SURF detector

In FAST, estimation of a desired pixel, p is whether a corner or not is accomplished based on its intensity level, I_p and its relationship to the 16 pixels, numbered in clockwise direction that lie inside a circular region that has the radius of 3 pixels [18,29]. Figure 8 illustrates how this neighbourhood is implemented over a given candidate pixel.

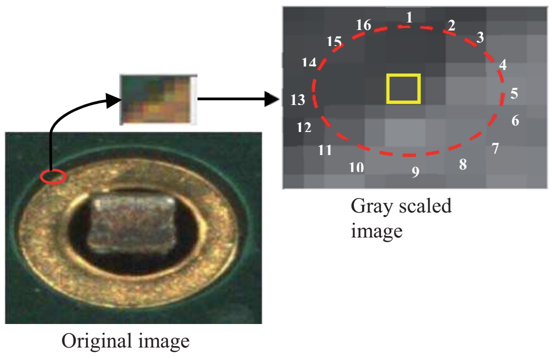


Fig. 8. Selection of neighbouring pixels lie inside the defined circle cantered at candidate pixel P (Yellow Square)

A detailed explanation on how this neighbour comparison process is carried out is presented in [18]. Once the stable key points are detected as explained in [18], the algorithm must compute an unique descriptor to each detected key point as SIFT and SURF to make a detected key point to be distinct. However, the authors of FAST do not provide their own descriptor for the detected features. Therefore, we used one of the well-known descriptor algorithms, Binary Robust Independent Elementary Features (BRIEF) [31], which has been developed for generating a descriptor for detected key points from a feature detector.

We know that SIFT and SURF use descriptors with the sizes of 128 and 64 dimensional vectors respectively, which lead to have memory constraints for the applications which have limited memory capacity available. BRIEF comes into picture at such situations with providing binary strings for detected key points rather than computing individual descriptor to each of them [31]. A detailed explanation on how the BRIEF descriptor works can be found in [31]. The detected key points on several model images by FAST algorithm are illustrated in Figure 9.

As stated earlier, FAST is not a scale and rotational invariant feature detector as SIFT or SURF. However, in our application, these two properties do not add any considerable impact on the performance of the algorithm. Because the PCB is placed on a positioning bed and the distance between the camera and the PCB is precisely controlled by the robotic system ensuring that no scale distortion or a slight displacement occurs.

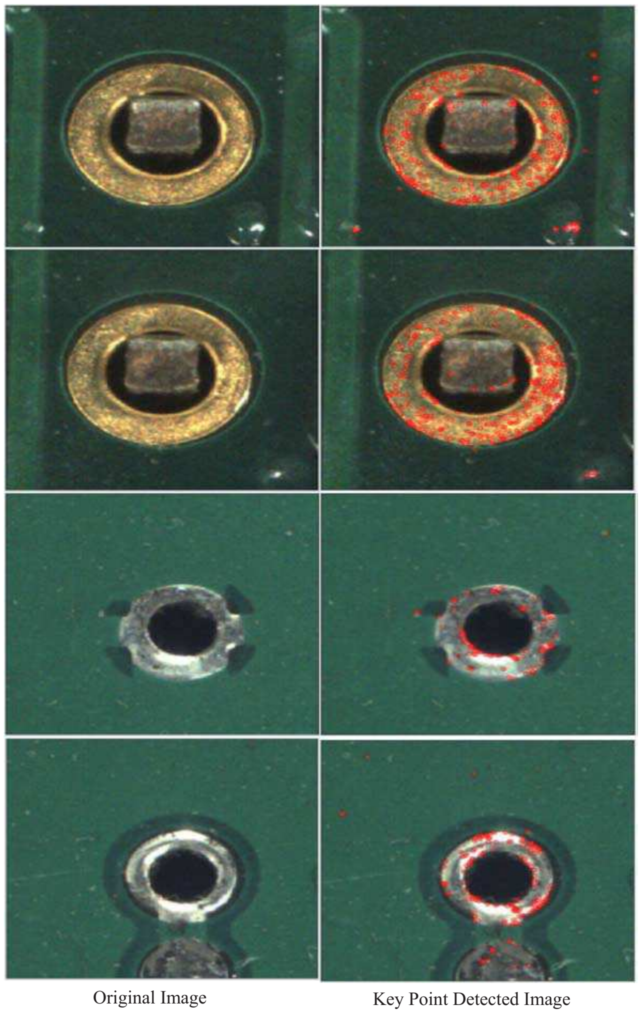


Fig. 9. Detected key points on solder pads, vias and test pads on different colour PCBs at $I_t = 30$ with non-maximum suppression using FAST corner detector. Here I_t is a user defined threshold value

3. Performance evaluation of feature extraction algorithms

In section 2, we illustrate how SIFT, SURF and FAST feature detectors localize distinguishable features from a given image. In order to evaluate performance of feature extraction algorithms analysed in section 2, three main evaluation criteria have been taken into consideration in our application:

- locations of detected key points,
- accuracy of the algorithm in various illumination conditions with respect to the repeatability of detected key points,
- number of correct and wrong matches made at different illumination conditions with different distance matching algorithms.

According to Figures 5, 7 and 9, it could be seen that majority of the detected key points lie inside the object regions in a given input image. Table 1 illustrates the results obtained based on the localized region of key points over 100 different images (this will be the size of the sample which will be taken into consideration for rest of the analysis carried out later in this paper) using these three algorithms. There are two main regions have been considered in this analysis process named as foreground (region where the main object lies in the given input image – solder pad, via, fiducial point etc.) and background (region that does not contain the main object region – PCB surface). The percentage of key points that are detected in foreground and background regions is calculated with respect to the total number of detected key points in a given input image at normal illumination level. The normal illumination level is taken in between 250-750 lux in our application.

Table 1.
Average percentage of key points detected in the regions of objects in SIFT, SURF and FAST feature detectors

Algorithm	Localized in foreground region	Localized in background region
SIFT	95%	5%
SURF	98%	2%
FAST	96%	4%

According to Table 1, it can be seen that all of these feature detectors are capable of detecting features inside foreground regions in a given image, even though SURF slightly outperforms other two algorithms in terms of accuracy in detecting key points.

As stated beginning of this paper, our proposed vision system has been developed to integrate with a desktop soldering robotic system. One of the main goals of the proposed AOI system is its ability to operate in different environments where no specific lighting is applicable. Therefore, we have analysed the performance of these algorithms at different illumination conditions as illustrated in Table 2. We compared the repeatability of detected key points inside the same image during consecutive trials (50 trials) at highly illuminated (lux level in between 1000 to 1500) and poorly illuminated (lux level in between 75 to 200) conditions against the key points detected inside the same image at normal illumination level.

Table 2.
Average repeatability percentage of key points detected inside given images using SIFT, SURF and FAST feature detectors at different illumination levels

Algorithm	(1000-1500) lux	Lux level < 200 lux
SIFT	78.5%	86%
SURF	82%	85%
FAST	72%	76%

According to Table 2, it can be clearly seen that SIFT and SURF algorithms outperform FAST at different illumination levels in terms of key point detection repeatability. Even though SURF outperforms SIFT at highly illuminated environments, SIFT takes the advantage in less illuminated environments. Once the key points are detected on a sample image (In our case model image), their robustness to be precisely detected in a given input image (In our case located image) must be fairly evaluated. This task is accomplished with the use of a feature matching algorithm that is capable of matching descriptors of the detected features of a model image to the descriptors of detected features of a located image (I_L). In our application, two main feature matching algorithms have been evaluated for matching feature descriptors that are generated for both I_M and I_L.

- Brute Force Matcher [10],
- Fast Library for Approximate Nearest Neighbour (FLANN) [10].

In our application, three main distance calculation methods, L1-norm distance calculation, L2-norm distance calculation and hamming distance calculation [10] have been used for brute-force matcher.

We have so far described several feature matching and distance calculation algorithms that can be used to match features between a model image (I_M) and located image (I_L). Now it is required to evaluate the performance of these algorithms based on their stability and accuracy of matching features under varying illumination conditions. In our application, number of matched points and the accuracy of matching between I_M and I_L have been analyzed in two ways as listed below.

- Percentage of erroneous matching between given I_M and I_L,
- Percentage of correct matching between given I_M and I_L.

Here, the erroneous detection percentage means the number of falsely matched key points out of the detected key points in I_L to the detected key points in I_M. The successful detection percentage is the opposite of erroneous detection percentage. Both of these points are analysed under three different illuminations conditions as presented in Table 2. Tables 3-5 illustrate the erroneous detection rate of these three feature detectors together with feature descriptor matching algorithms at illumination levels of (250-750) lux, (75-200) lux and (1000-1500) lux respectively. Figures 10-12 illustrates several examples of the application of these feature detectors together with descriptor matching algorithms.

Table 3.
Percentage of erroneous matching between given model images and located images at (250-750) lux

Algorithm Name	L1-Norm	L2-Norm	FLANN
SIFT	6.93%	5.72%	0.13%
SURF	0%	0.09%	0.56%

Algorithm Name	L1-Norm	L2-Norm	Hamming	FLANN
FAST	12.66%	5.23%	7.33%	0.02%

Table 4.
Percentage of erroneous matching between given model images and located images at (75-200) lux

Algorithm Name	L1-Norm	L2-Norm	FLANN
SIFT	10.05%	9.02%	2.1%
SURF	1.9%	1.3%	1.1%

Algorithm Name	L1-Norm	L2-Norm	Hamming	FLANN
FAST	19.5%	11.2%	18%	1.8.%

Table 5.
Percentage of erroneous matching between given model images and located images at (1000-1500) lux

Algorithm Name	L1-Norm	L2-Norm	FLANN
SIFT	11.5%	10%	2.8%
SURF	1.2%	1.8%	1.3%

Algorithm Name	L1-Norm	L2-Norm	Hamming	FLANN
FAST	17.3%	16.8%	14.5%	4.8%

According to the results obtained in Tables 3-5, it can be seen that SURF is still able to hold the erroneous detection rate below 2% for each feature matching algorithm, even though FAST holds the lowest figure with FLANN in Tables 3 and 4. Another interesting point that can be revealed with the obtained results is that FLANN feature matching library provides the best accuracy for all three feature detectors. Tables 6-8 illustrates the successful average detection percentage obtained by these algorithms.

Table 6.
Percentage of successful matching between given model images and located images at (250-750) lux

Algorithm Name	L1-Norm	L2-Norm	FLANN
SIFT	4.7%	4.4%	1.5%
SURF	18.5%	17.7%	41.1%

Algorithm Name	L1-Norm	L2-Norm	Hamming	FLANN
FAST	17.3%	8.4%	14.9%	8.2%

Table 7.
Percentage of successful matching between given model images and located images at (75-200) lux

Algorithm Name	L1-Norm	L2-Norm	FLANN
SIFT	3.3%	3%	0.6%
SURF	15.2%	14.3%	36.5%

Algorithm Name	L1-Norm	L2-Norm	Hamming	FLANN
FAST	14.3%	5.6%	10.1%	6.5%

Table 8.
Percentage of successful matching between given model images and located images at (1000-1500) lux

Algorithm Name	L1-Norm	L2-Norm	FLANN
SIFT	3.15%	2.9%	0.45%
SURF	14.5%	14.1%	35.3%

Algorithm Name	L1-Norm	L2-Norm	Hamming	FLANN
FAST	13.1%	4.8%	9.3%	6.2%

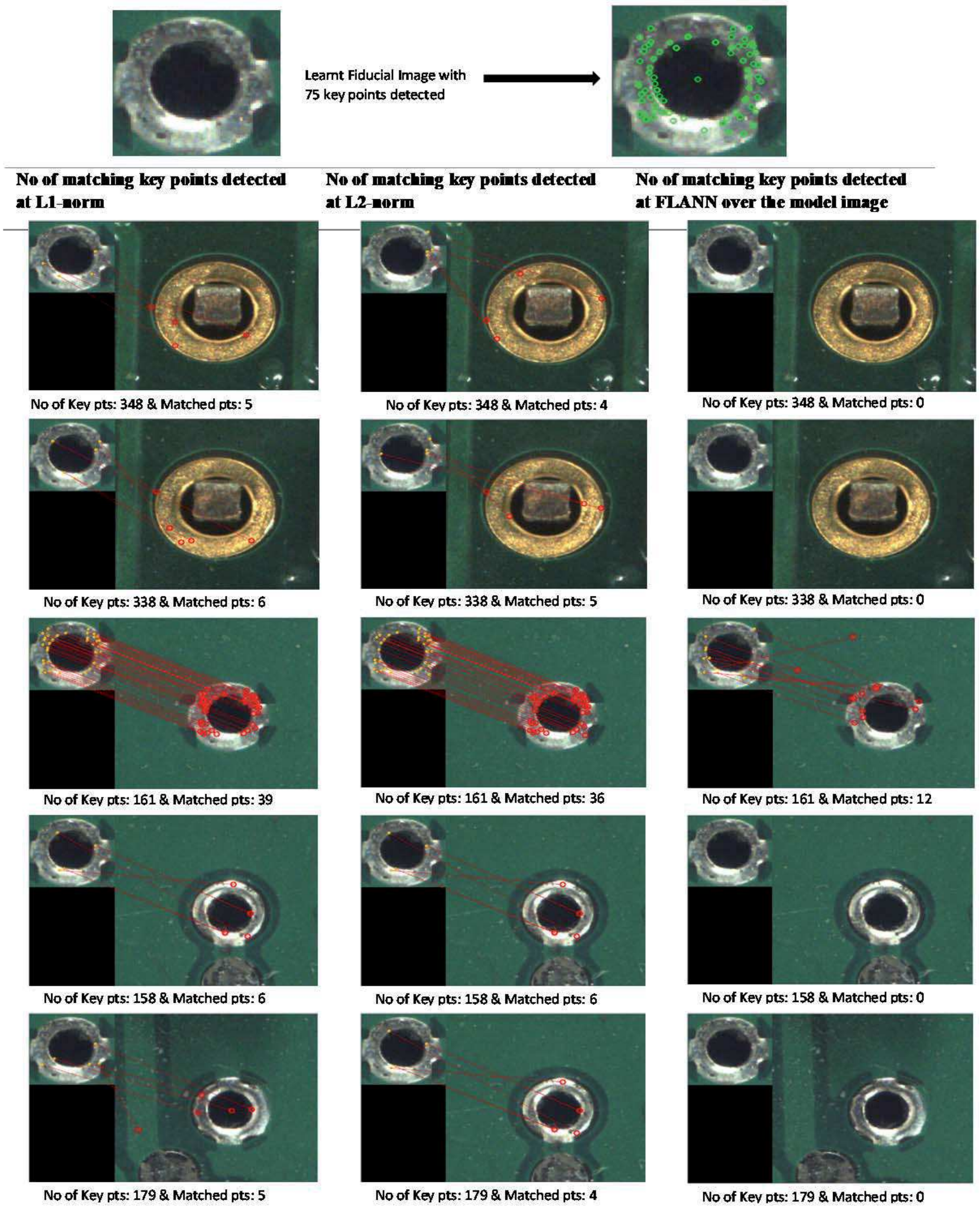

Learnt Fiducial Image with 75 key points detected
No of matching key points detected at L1-norm
No of matching key points detected at L2-norm
No of matching key points detected at FLANN over the model image
No of Key pts: 348 & Matched pts: 5
No of Key pts: 348 & Matched pts: 4
No of Key pts: 348 & Matched pts: 0
No of Key pts: 338 & Matched pts: 6
No of Key pts: 338 & Matched pts: 5
No of Key pts: 338 & Matched pts: 0
No of Key pts: 161 & Matched pts: 39
No of Key pts: 161 & Matched pts: 36
No of Key pts: 161 & Matched pts: 12
No of Key pts: 158 & Matched pts: 6
No of Key pts: 158 & Matched pts: 6
No of Key pts: 158 & Matched pts: 0
No of Key pts: 179 & Matched pts: 5
No of Key pts: 179 & Matched pts: 4
No of Key pts: 179 & Matched pts: 0

Fig. 10. Feature matching between images using SIFT together with distance measurement algorithms

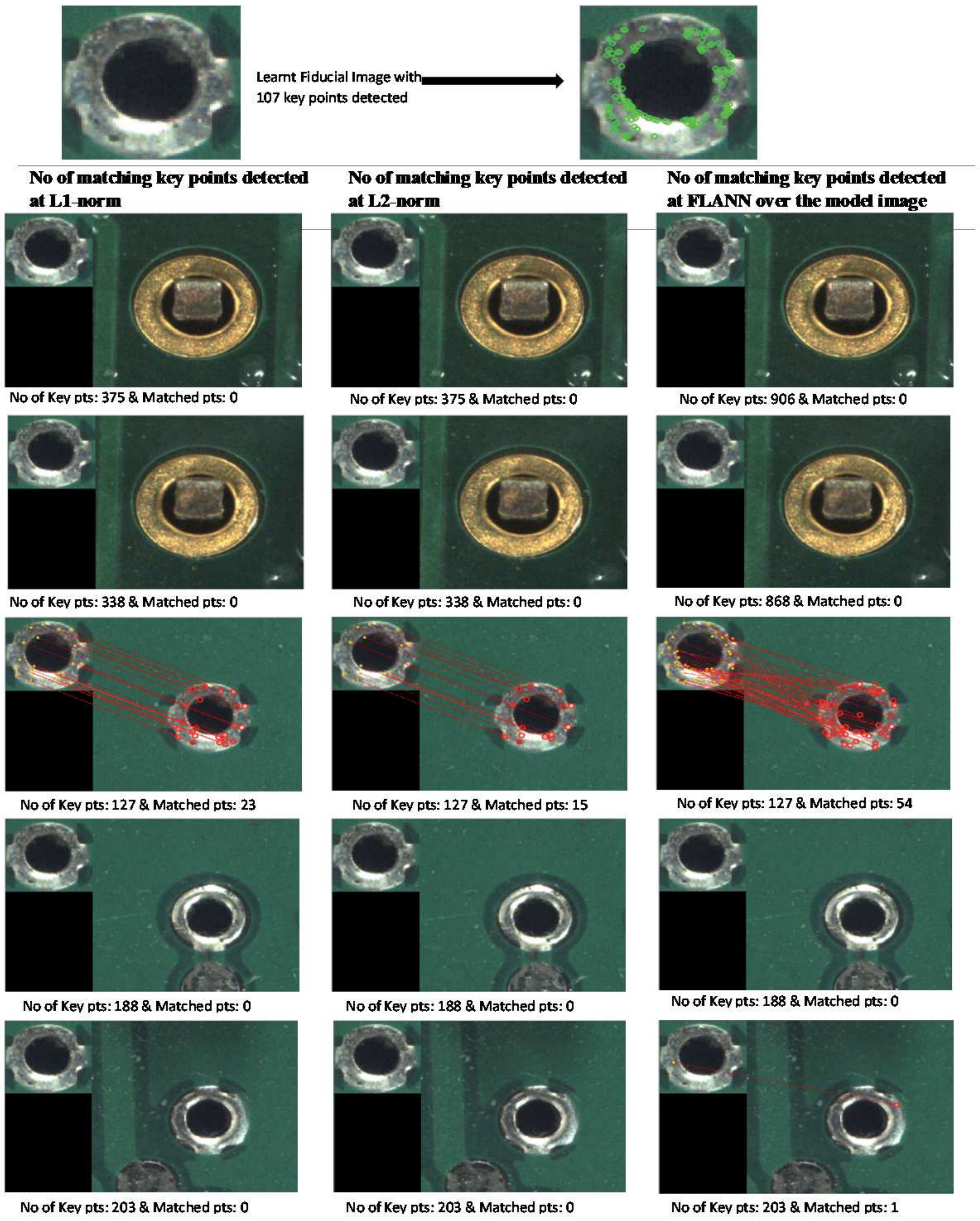

Learnt Fiducial Image with
107 key points detected
No of matching key points detected at L1-norm
No of matching key points detected at L2-norm
No of matching key points detected at FLANN over the model image
No of Key pts: 375 & Matched pts: 0
No of Key pts: 375 & Matched pts: 0
No of Key pts: 906 & Matched pts: 0
No of Key pts: 338 & Matched pts: 0
No of Key pts: 338 & Matched pts: 0
No of Key pts: 868 & Matched pts: 0
No of Key pts: 127 & Matched pts: 23
No of Key pts: 127 & Matched pts: 15
No of Key pts: 127 & Matched pts: 54
No of Key pts: 188 & Matched pts: 0
No of Key pts: 188 & Matched pts: 0
No of Key pts: 188 & Matched pts: 0
No of Key pts: 203 & Matched pts: 0
No of Key pts: 203 & Matched pts: 0
No of Key pts: 203 & Matched pts: 1

Fig. 11. Feature matching between images using SURF together with distance measurement algorithms

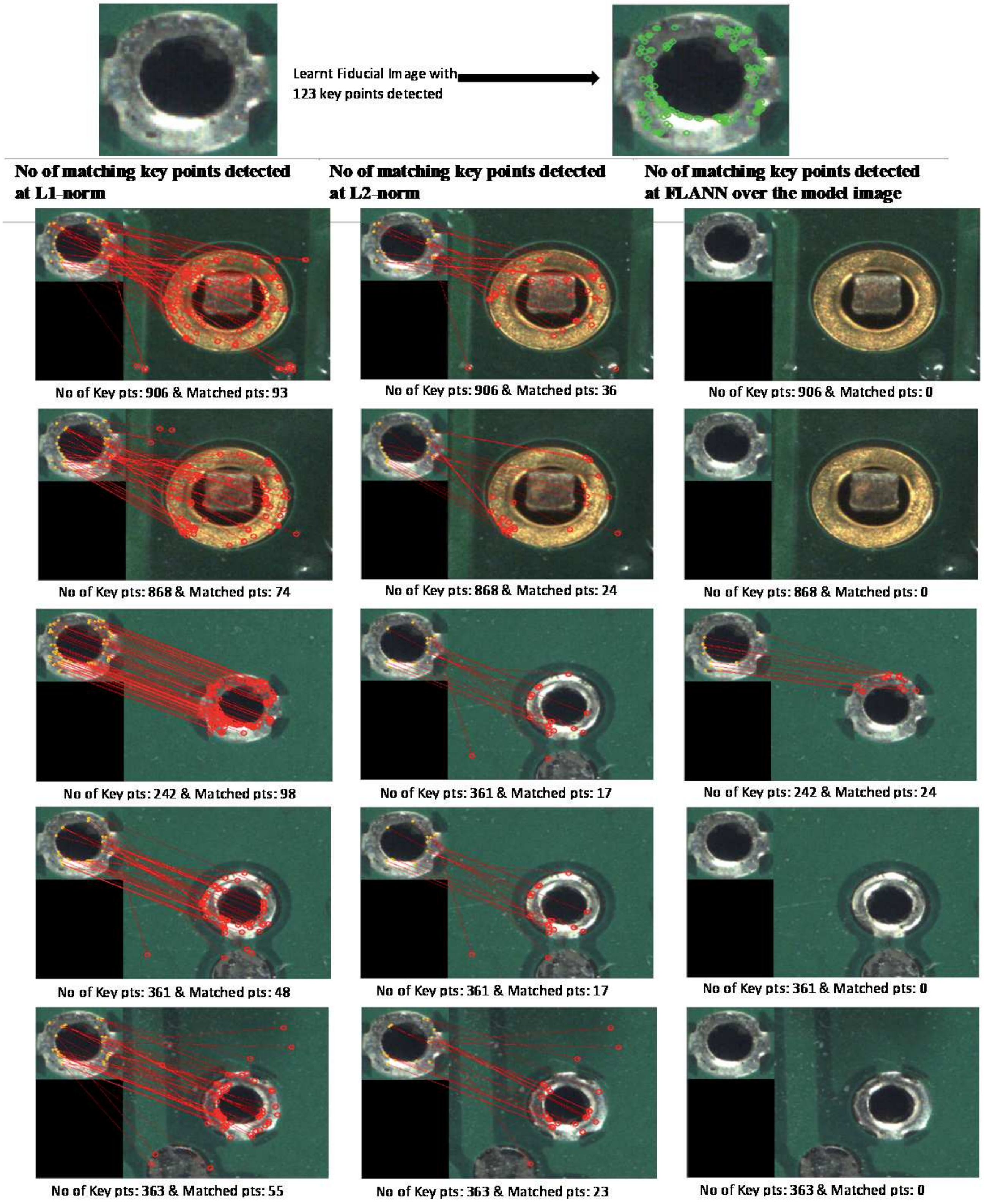

Learnt Fiducial Image with 123 key points detected
No of matching key points detected at L1-norm
No of matching key points detected at L2-norm
No of matching key points detected at FLANN over the model image
No of Key pts: 906 & Matched pts: 93
No of Key pts: 906 & Matched pts: 36
No of Key pts: 906 & Matched pts: 0
No of Key pts: 868 & Matched pts: 74
No of Key pts: 868 & Matched pts: 24
No of Key pts: 868 & Matched pts: 0
No of Key pts: 242 & Matched pts: 98
No of Key pts: 361 & Matched pts: 17
No of Key pts: 242 & Matched pts: 24
No of Key pts: 361 & Matched pts: 48
No of Key pts: 361 & Matched pts: 17
No of Key pts: 361 & Matched pts: 0
No of Key pts: 363 & Matched pts: 55
No of Key pts: 363 & Matched pts: 23
No of Key pts: 363 & Matched pts: 0

Fig. 12. Feature matching between images using FAST together with distance measurement algorithms

According to the results obtained in Tables 6-8, it can be seen that the SURF algorithm is able to hold its successful detection percentage above 30% even at different illumination levels. This further means that SURF together with FLANN is capable of precisely matching nearly 30 key points out of 100 detected key points in I_L to the corresponding 30 keypoints in I_M. The successful PCB fiducial image identification of SURF can be made further clear according to Figures 10-12. The computation time of an algorithm was not taken into consideration during our analysis because the expected accuracy is more important than the processing time. In addition to that fiducial verification process is carried out only one time per a given PCB. Therefore, the processing time will not add a significant benefit to our application. According to the results obtained in Tables 1-8, it should be obvious that SURF together with FLANN enables to have more stable and accurate detection rate for fiducial verification in our application.

The next step is the precise detection of model image area inside the located image once the proposed feature detection algorithm assures the occurrence of the model image inside the located image. As we have stated earlier in this paper, once the fiducial image is identified and localized inside a located image acquired from the camera view, the system calculates the distance to the centre of the localized region from the robotic system origin. This process is utilized at least over two user defined fiducial images. Once these two distances are calculated, the vision system calculates relative distances to other points on the PUS reference to these distances. Therefore, it must be clear that the precise localization of exact area contained by the fiducial image is very important in our application. We have analysed the performance of two methods to accomplish this task at highest accuracy to estimate the model image area inside the located image as listed below:

- Using homography matrix,
- Using template matching techniques.

A homography is a perspective transformation of a plane that is a re-projection of a plane from one camera view into a different camera view subjective to change in the position and orientation of the camera. Homography matrix is a 3×3 transformation matrix that is used to perform this transformation between planes. It is required to have minimum of four points to find this transformation matrix. In our application, we use set of points that have been matched between I_M and I_L with our proposed feature detector and find the corresponding perspective transformation matrix. Once this 3×3 matrix is computed, it is used to find the corners of I_L to corresponding points in I_M. Figure 13 illustrates the results obtained with the use of a homography matrix generated from correctly matched points between several images.

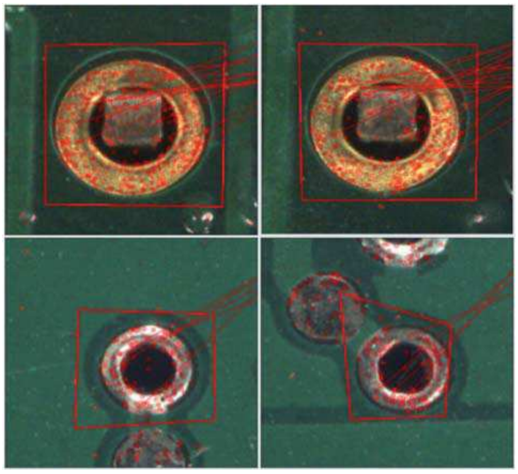


Fig. 13. Falsely detected areas of the model image inside the input image using homography matrix

According to Figure 13, it is obvious that the estimation of model image area inside the input image with homography matrix is far beyond the expected accuracy in our application. We have proved earlier in this paper that template matching would not be a good choice to localize a model image inside a given input image, when the existence of model image is unknown (Fig. 1). But at this stage of our application, we have confirmed the existence of I_M inside $I_L$with our proposed feature detector. Therefore, there might be a possibility to have a precise localization of model image area inside located image using template matching algorithms. Section 4 presents a performance evaluation of six different template matching algorithms over a sample of 100 images.

4. Performance evaluation of template matching algorithms (TM)

Template matching is a high level machine vision technique that localizes the best matched location of a model image inside a given input image [6]. Template matching techniques are flexible and relatively straightforward to use, which makes them one of the popular methods of object localization in the fields of surveillance, vehicle tracking, robotics, medical imaging, manufacturing and etc. [7]. The occurrence of similarities, resulted from comparison between I_M and I_L, are reflected in the resulted image (I_R). In this paper, six types of template matching techniques, squared difference (TM_SQDIFF) error, normalized squared difference (TM_SQDIFF_NORMED) error, concept of correlation [32] among I_M and I_L (TM_CCORR), concept of normalized version of correlation among I_M and I_L (TM_CCORR_NORMED), concept of correlation coefficient (TM_CCOEFF) and normalized version of correlation coefficient (TM_CCOEFF_NORMED) [9,10] have been discussed and analysed for accurately identifying the location of I_M inside I_L.

The basic concept of a template matching algorithm is to find objects in a given image, which have minimum error difference with a model image as explained with the six different template matching methods in [9,10]. The complexity of these methods varies starting from simple squared difference of error between I_M and I_L to more complex processes like computing correlation and correlation coefficient between I_M and I_L. The performance of each template matching algorithm is evaluated to find out the best suited algorithm for the localization of fiducial area inside the located image. Figure 14 illustrates the outcome of these algorithms over the feature matched images using SURF. Table 9 illustrates the acquired results over 100 different images.

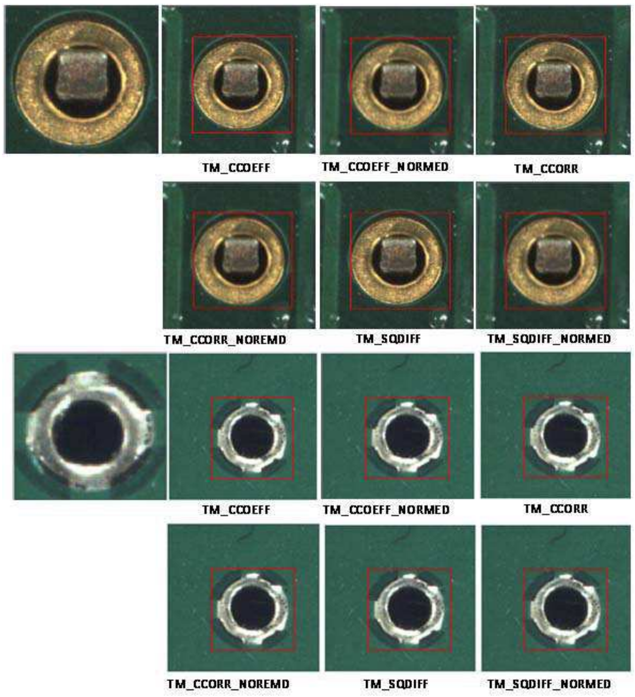


Fig. 14. Model image localization inside given input image using template matching algorithms

Table 9.
Successful detection rate of template matching algorithms over the feature matched located images

Template Matching Algorithm	Success Rate
TM_CCOEFF	100%
TM_CCOEFF_NORMED	100%
TM_CCORR	100%
TM_CCORR_NORMED	100%
TM_SQDIFF	100%
TM_SQDIFF_NORMED	100%

According to Table 9, it can be seen that almost every algorithm showed the best performance in precise localizing of the given model image. However, we have to keep in mind that these results have been obtained assuming an ideal condition of a given PCB. It means that it is assumed that the PCB has no manufacturing impurities like, the sizes of the objects on a PCB may not be unique over range of PCBs. The template matching algorithms are vulnerable to dimension changes of input images. Therefore, it is required to find the most robust algorithm out of these algorithms, which is less vulnerable to slight dimension changes of located images. In order to utilize this analysis process, we scaled down the located images by 2% from their actual size and verified with the template matching algorithms. Figure 15 illustrates the outcome of this process.

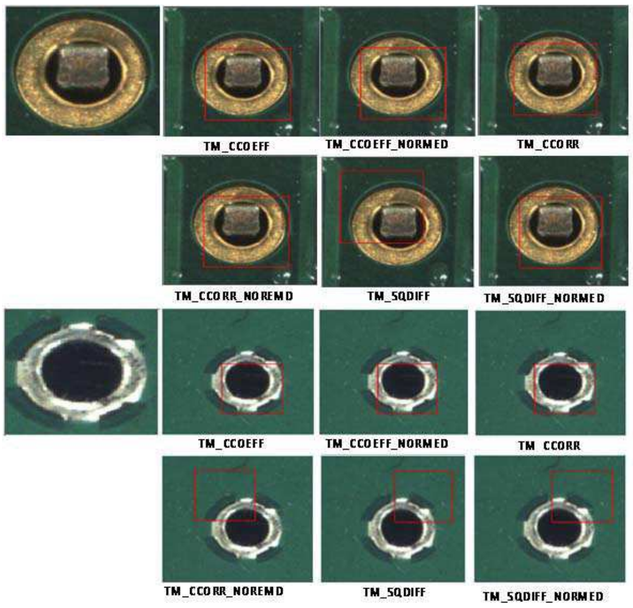


Fig. 15. Results obtained from template matching algorithm over 2% scaled down images

According to Figure 15, it can be seen that TM_CCORR is still capable of rendering most accurate results even with the slight variation of size of a located image. Table 10 contains the results obtained over 100 different scaled down images by a percentage of 2%.

Table 10.
Successful detection rate of template matching algorithms over the scaled down located images by a percentage of 2%

Template Matching Algorithm	Success Rate
TM_CCOEFF	92%
TM_CCOEFF_NORMED	92.5%
TM_CCORR	95%
TM_CCORR_NORMED	80%
TM_SQDIFF_NORMED	83%

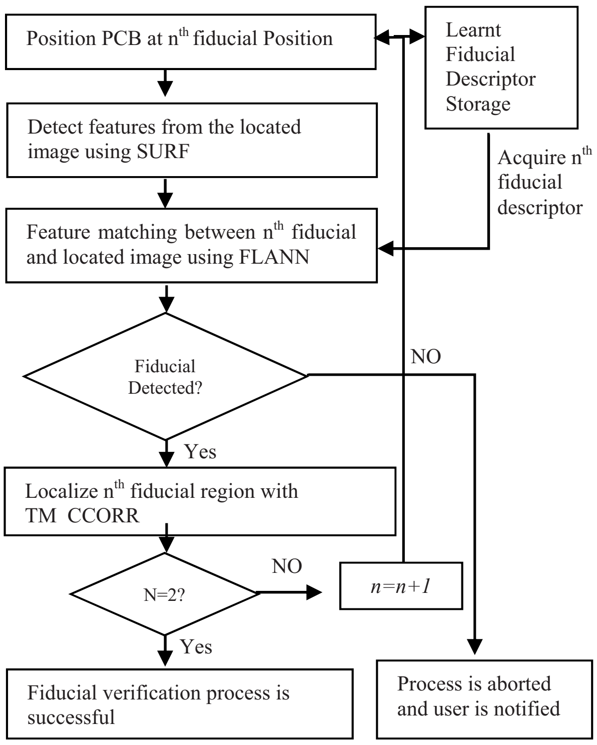
Even though TM_CCORR gives the highest successful detection rate, it is still not perfect. However, the effect of this 5% error can be made further minimized with the procedures described in our paper [3] that describes the precise identification of solder pad area from the PCB surface.

5. Conclusions

The main goal of this paper is to evaluate the performance of feature extraction and template matching algorithms in terms of their robustness and efficiency in precise identification and localization of fiducial image inside a given image. We were able to find out the most precise method to identify and localize the learnt PCB fiducials, that is independent from the operator‘s skill level and interaction using SURF together with FLANN and TM_CCORR. We have analysed the performance of these algorithms under different illumination conditions and scale variations of the located image. The identified methods could be proven to be less vulnerable to these practical imperfections according to the analysis carried out during the implementation of our proposed fiducial verification process (Tables 1-10). Figure 16 illustrates how this fiducial verification process is implemented in our application.

The accurate identification of fiducials adds significant benefits to the overall performance of the robotic system as listed below.

- Enables very precise positioning of the points to be soldered;
- Reduces the risk of soldering iron touching nearby components, since this process ensures the precise positioning of the system and it assures whether the placed PCB is the actual one to be soldered;
- Makes the system independent from the skill level of the operator.



Position PCB at nth fiducial Position
Learnt Fiducial Descriptor Storage
Detect features from the located image using SURF
Acquire nth fiducial descriptor
Feature matching between nth fiducial and located image using FLANN
Fiducial Detected?
NO
Yes
Localize nth fiducial region with TM_CCORR
N=2?
NO
n=n+1
Yes
Fiducial verification process is successful
Process is aborted and user is notified

Fig. 16. Process flow structure of the proposed fiducial verification process

Once the fiducials are precisely localized by the vision system, the next step is to calculate all the relative distances to the THT solder pads that should be soldered by the robotic platform. The procedure of detecting whether the particular solder pad is precisely positioned by the robotic system positioning mechanism has been informatively presented in the published paper by us [3]. The completion of the proposed AOI system requires another two main stages to be operational.

- Automatic detection of THT component lead and localization of its location after the soldering process is completed.
- Implementation of methods to perform automatic quality assurance of a THT solder joint.

Acknowledgements

This research was supported by Prof. J.A.K.S Jayasinghe, University of Moratuwa by providing a good guidance and Mr.Chandana Dissanayake (Managing Director Variosystems (Pvt) Ltd, Sri Lanka), by providing me the opportunity and all the required accessories to work on this research. I am immensely grateful to Prof. J.A.K.S Jayasinghe, University of Moratuwa for his comments on an earlier version of the manuscript. If any error exists, they are my own and should not tarnish the reputation of him.

References

[1] G. Goldberg, ROI of a Soldering Robot, Circuit Assembly Online Magazine, http://www.circuitsassembly.com/ca/magazine/26341-automation-1609.html.

[2] PCB Assembly Equipment for Through Hole Assembly and Soldering, DDM Novastar, automated production systems, https://www.ddmnovastar.com/.

[3] C.L.S.C Fonseka, J.A.K.S Jayasinghe, Color Model Analysis for Solder Pad Segmentation on Printed Circuit Boards, International Journal of Scientific and Research Publications 6/11 (2016) 212-225.

[4] R. Katukam, P. Sindhoora, Image Comparison Methods & Tools: A Review, Proceedings of the 1st International Conference "Emerging trends in Information Technology" ETIT, 2015, 35-42.

[5] H. Ney, B. Leibe, Matching Algorithms for Image Segmentation, January, 2010.

[6] N. Praveen, D. Kumar, I. Bardwaj, An Overview on Template Matching Methodologies and its Applications, International Journal of Research in Computer and Communication Technology 2/10 (2013) 988-995.

[7] A. Banharnsakun, S. Tanathong, Object Detection based on Template Matching through use of best so far ABC, Computational Intelligence and Neuroscience 2014 (2014) ID: 919406 1-8.

[8] OpenCV, Introduction to SIFT, http://docs.opencv.org/3.3.0/da/df5/tutorial_py_sift_intro.html.

[9] OpenCv 2.4.13.3 documentation, Template Matching, http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html.

[10] G. Bradski, A. Kaehler, Learning OpenCV, O'Reilly, 2008.

[11] A. Aichert, Feature extraction techniques, Proceedings of the Camp Medical Seminar WS0708, 2008/

[12] Ch. Wu, Advanced Feature Extraction Algorithms For Automatic Fingerprint Recognition System", PhD Thesis, State University Of New York, New York, 2007.

[13] T. Kadir, D. Boukerroui, M. Brady, An analysis of the Scale Saliency algorithm, Robotics Research Laboratory, University of Oxford Publishing House, Oxford, 2003.

[14] I.R. Otero, M. Delbarico, Anatomy of the SIFT Method, Image Processing On Line 4 (2014) 370-396.

[15] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-Up Robust Features (SURF), Proceedings of the European Conference on Computer Vision, 2006, 404-417.

[16] J.T. Pedersen, SURF: Feature detection & description, Q4 2001, http://cs.au.dk/~jtp/SURF/report.pdf.

[17] E. Oyallon, J. Rabin, An Analysis of the SURF Method, Image Processing On Line 5 (2015) 176-218.

[18] E. Rosten, T. Drummond, , Machine learning for high-speed corner detection, Proceedings of the European Conference on Computer Vision ECCV 2006: Computer Vision – ECCV 2006, 430-443.

[19] J. Bollen, H. Van de Sompel, A. Hagberg, R. Chute, A Principal Component Analysis of 39 Scientific Impact Measures, PLoS One 4/6 (2009) 1-11.

[20] V.D. Calhoun, V.K. Potluru, R. Phlypo, R.F. Silva, B.A. Pearlmutter, A. Caprihan, S.M. Plis, T. Adali, Independent Component Analysis for Brain fMRI Does Indeed Select for Maximal Independence, PLoS One 8/8 (2013).

[21] D.G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision 60/2 (2004) 91-110.

[22] M. Basu, Gaussian-Based Edge-Detection Methods-A Survey, IEEE Transaction on Systems, Man and Cybernetics, Part C: Applications and Reviews 32/3 (2002) 252-260.

[23] T. Lindeberg, Edge Detection and Ridge Detection with Automatic Scale Selection, Proceedings of the IEEE Computer Society Conference: Computer Vision and Pattern Recognition, 1996.

[24] X. Yin, B.W.H. Ng, J. He, Y. Zhang, D. Abbott, Accurate Image Analysis of the Retina Using Hessian Matrix and Binarisation of Thresholded Entropy with Application of Texture Mapping, PLoS One (2014), doi: 10.1371/journal.pone.0095943.

[25] B.S. Morse, Differential Geometry, Lecture 11, Brigham Young University, 1998-2000.

[26] D.E. Brown, The Hessian matrix: Eigenvalues, concavity, and curvature, BYU Idaho Department of Mathematics, April 2014.
[27] S. Ehsan, A.F. Clark, N. ur Rehman, K.D. McDonald-Maier, Integral Images: Efficient Algorithms for Their Computation and Storage in Resource-Constrained Embedded Vision Systems, Sensors (Basel) 15/7 (2015) 16804-30.
[28] C.R. Molenkamp, Accuracy of Finite-Difference Methods Applied to the Advection Equation, Journal of Applied Meteorology 28 (1989).
[29] D.G. Viswanathan, Features from Accelerated Segment Test (FAST), http://homepages.inf.ed.ac.uk /rbf/CVonline/LOCAL_COPIES/AV1011/AV1Feature fromAcceleratedSegmentTest.pdf.
[30] M. Muja, David G. Lowe, Fast Approximate Nearest Neighbors With Automatic Algorithm Configuration, Computer Science Department, University of British Columbia, http://citeseerx.ist.psu.edu/viewdoc /download?doi=10.1.1.160.1721&rep=rep1&type=pdf.
[31] M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: Binary Robust Independent Elementary Feature", CVLab, EPFL, Lausanne, Switzerland.
[32] Digital Image Correlation: Overview of Principles and Software, SEM 2009, University of South California.