Journal of Imaging Science and Technology® 54(5): 050306–050306-15, 2010.

Alignment of Individually Adapted Print Patterns for Ink Jet Printed Electronics

Tapio Manninen, Ville Pekkanen, Kalle Rutanen, Pekka Ruusuvuori, Risto Rönkkä[†] and Heikki Huttunen
Tampere University of Technology, Tampere, Finland
E-mail: heikki.huttunen@tut.fi

Abstract. *Printed electronics is a new technology for manufacturing electronic structures using standard printing processes. One emerging application area is in microelectronics packaging, where printing the interconnections allows manufacturing modules in miniature size. The basic principle is that the integrated circuits are molded into a background material such that the connection pads are left visible on top. After the background substrate has hardened, the wiring is printed on top of the module by using conductive ink. Typically, however, there are unidealities in the manufacturing process that result in the components and their connection pads being slightly displaced. This article describes an alignment and adaptation system that adapts the wiring bitmap to match the displaced components on a per-module basis. Moreover, the adjusted bitmaps can be combined together to enable printing of several modules at the same time. © 2010 Society for Imaging Science and Technology.*
[DOI: 10.2352/J.ImagingSci.Technol.2010.54.5.050306]

INTRODUCTION

The current trend in the electronics manufacturing is the increasing awareness of environmental implications of production and logistics. At the same time consumers are more demanding and looking for products to serve their needs. Further, the electronics industry is investing significantly on technologies that would allow miniaturization of electronics assemblies. The key point in satisfying these needs is to have a manufacturing technology that is environmentally friendly and allows miniaturization of electronics assemblies with a high level of flexibility in demand supply networks. One of the most prominent new technologies satisfying these needs is printed electronics.

Printed electronics is an additive process and a relatively new area of research, which uses traditional printing devices for interconnecting or manufacturing components. One of the fundamental technologies is ink jet printing, which relies on the principle of the drop-on-demand ink jet, but utilizing electrically functional liquids. An example of an application of this technology uses conductive nanoparticle and dielectric inks to create interconnection circuits between the connection pads of the integrated circuits (ICs) and the discrete components that have been molded into the background surface.[1] In this article we call this concept a printed module. One such module is shown in Figure 1 (left), where there are four ICs molded into the background. The next step in the processing would be to print the wiring—shown on the right—on top of the module.

One of the most significant challenges in printed modules is the accuracy of the manufacturing process. Inaccuracies in the component placement process, molding process related movement, and molding material shrinkage and bending all create errors in component locations. The latter category is typically the most significant one, and also the most unpredictable.[2]

The unpredictability of the deformation of the module is the main reason why the traditional inspection approaches[3] for printed circuit board (PCB) manufacturing are unsuitable solutions for this misalignment problem. In PCB manufacturing the background materials are rigid, which allows only translation and rotation. In our case, the substrate may shrink or expand causing the components to move with respect to each other. In addition to the complexity of the transformation, most existing approaches concentrate on quality control instead of dynamic correction. In our case the interconnection structure resides as an image in computer memory, which makes it easy to correct prior to printing.

This article addresses the problem of aligning individually modified print patterns by describing an adaptation system, which uses computer vision for detecting the true locations of the connection pads and modifies the bitmap to be printed accordingly. The material is partially based on three earlier conference articles. First, we proposed a system that acts as a *front end* for the complete system, i.e., detects the locations of the connection pads of the ICs from a high

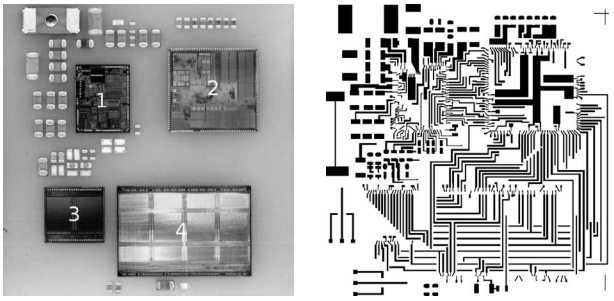


Figure 1. A sample four-IC module (left) ready for printing the connections on the right.

[†]Also at: DropAim Oy, Tampere, Finland.

Received Dec. 23, 2009; accepted for publication Apr. 30, 2010; published online Aug. 19, 2010.
1062-3701/2010/54(5)/050306/15/$20.00.

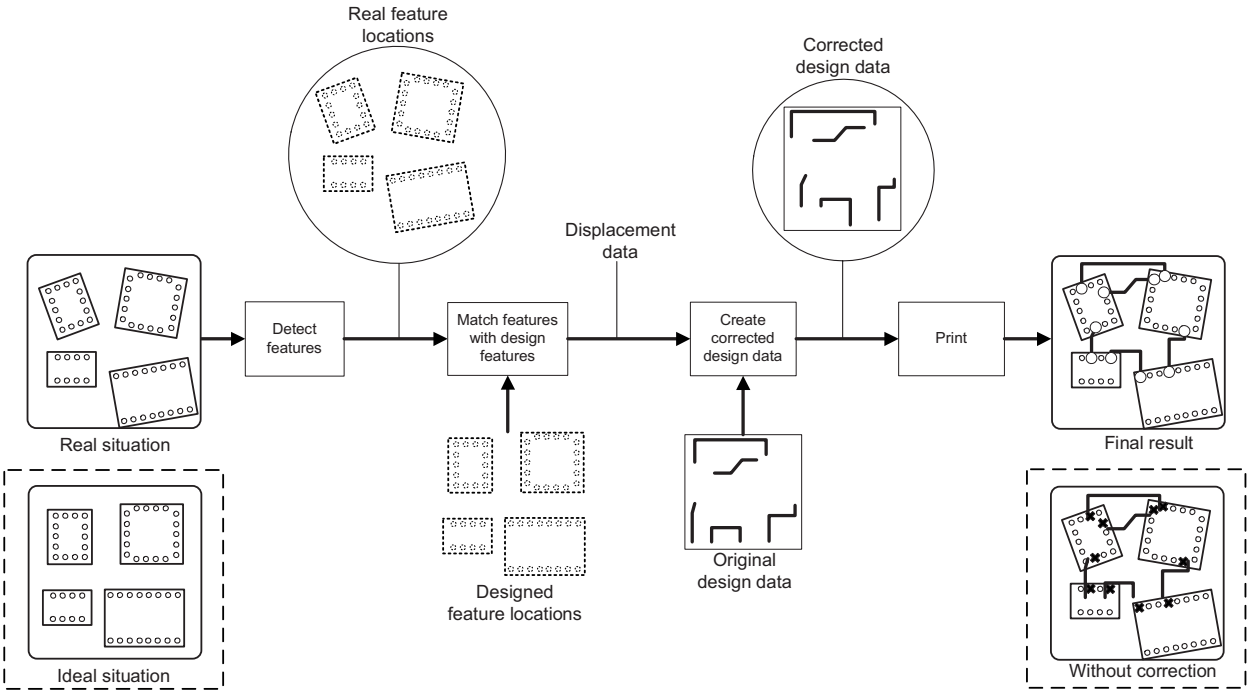


Figure 2. Diagram of the entire correction system.

resolution color image.[4] Next, we studied the design of the *back end* for the system that couples the intended positions of the ICs and their connection pads with the measured locations.[5] Finally, an algorithm for adjusting the bitmap automatically was proposed.[6] In addition to the unified and more detailed description of the correction system, this article extends the study to printing multiple modules simultaneously. By photographing the individual modules separately, the alignment algorithm can be used to create a print file that covers an arbitrary number of modules, enabling the printing of a panel of electronics modules at the same time.

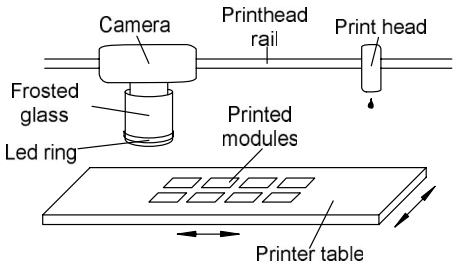


Figure 3. The hardware setup used in the experiments.

SYSTEM DESCRIPTION

Figure 2 illustrates the operation of the proposed alignment system. The processing begins with the acquisition of the image of the real situation, where the components may have drifted away from their intended locations. This image is considered as a transformed version of the original design data. In order to find the required correction, a set of relevant features (in our case the connection pads of the ICs) and their locations are detected from the image (see the section Front End). The true coordinates of the objects are passed to the next block (see the section Back End), which searches for a transformation between the two coordinate sets (i.e., combines the knowledge where the objects are with the knowledge where they should be). As a side product, we get full statistics of the displacements, and can monitor the manufacturing process.

Once the displacements of all objects have been tracked, the next task is to adjust the print files of the following layer to match accurately with the true component placement. The constraints for the adjustment are twofold: all IC contacts must match the intended point in the print file, while the contacts for the following layers should stay as they are. The latter condition is required in order not to propagate the errors further to the following layers. This simplifies subsequent processing and may also speed it up, since the following layers can be aligned exactly to a low resolution printing grid, thus, enabling the use of a lower resolution. The correction procedure is described in the section Correction.

The system can be easily extended into printing of multiple modules at the same time. Since the alignment system corrects the displacements at the IC level, it is equivalently able to manage displacements of multiple modules at the same time. The extension to panel printing is described in the section Printing Multiple Modules Simultaneously.

The hardware setup is illustrated in Figure 3. Our solution uses Canon EOS 5D digital camera with a Canon MP-E 65 mm macrolens. The camera has been mounted on the printer rack of iTi (Imaging Technology International) XY MDS2.0 ink jet printer. The rack also includes the print head, a UV-light source (for curing of the printed material), and other printing related equipment. A vertically adjustable camera mounting bracket along with the zoom of the optics

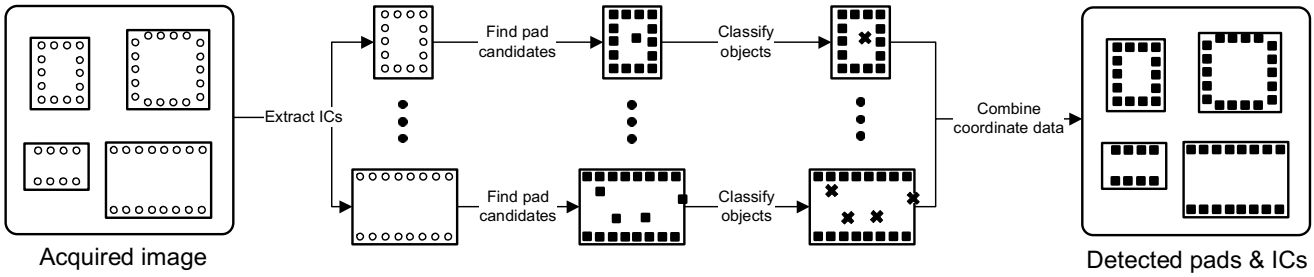


Figure 4. The block diagram of the front end.

are used for controlling the focus and magnification of the images. A translucent glass with a led ring has been installed around the space between the objective and the target in order to illuminate the target area and to avoid distracting reflections from the surrounding environment. The camera is connected to a PC, which operates the camera and runs the correction software. The printer table below the rack is mobile and uses vacuum to hold the printed modules in place.

FRONT END

The *front end* part of the system aims at detecting the ICs and the connection pads on them. The locations of the detected connection pads are sent further to the *back end* that couples the data with the corresponding design data and finds the transformations that each of the ICs has experienced.

The block diagram of the front end is shown in Figure 4. In the whole system diagram (Fig. 2) front end covers the *Detect features* block. There are three major steps in front end. The first step, *IC extraction*, separates the ICs from the background. The next step, *connection pad finding*, aims at finding the connection pads from the detected IC areas. This is done in two stages: the first step is a preliminary search step to extract the connection pad candidates. The first step uses a low-level logical operation, which is computationally very fast. This step is necessary because the following step is too complex to do for all image pixels. The final step, *classification*, looks into the candidate areas more thoroughly to select only candidates that are actual connection pads. In the following, we will describe these three phases in detail.

IC Extraction

Each IC has individually drifted away from its intended position. The first thing to do is to separate the ICs from the module background. The reason for the detection of the IC locations is that it allows restricting the further connection pad candidate search only onto the IC areas. This, in turn, speeds up the processing and also removes possible false matches in the background area. If the computational complexity is not a critical issue, this step can also be omitted.

Single modules are aligned on larger module panels, which ensures that every time an image is acquired, the orientation of the target module can be assumed to be approximately the same as that of the previously printed module. The method doing the IC extraction is based on this fact: a template of the IC layout is manually drawn on a single module image and stored for comparison with latter modules. The data include a cropped RGB (red, green, blue) image of the IC layout with location information of each IC. This template can be used to detect the ICs of the subsequent modules printed after the first one: The image of the first module is used as a template and matched with that of the latter modules. The manually drawn locations of the ICs in the first module now serve as approximate locations in the subsequent images.

In order to speed up the IC detection, both the template image and the camera image can be heavily downsampled. In our experience, downsampling ratios as high as 10–15 (with respect to the 12.8 Mpix camera images) still give acceptable accuracy. After downsampling, the resized module template is matched to the resized image. The matching is made such that the square error of the corresponding pixels in the template and the image is minimized.[7] Knowing the IC locations in the template image and the location of the best match in the target image, we can determine the coordinates of the ICs.

The complete IC extraction procedure is described in Algorithm 1. The algorithm takes in the module image I and returns the coordinates of the ICs in it. The algorithm also uses a template image T of each IC. The template images have been acquired beforehand in a system calibration phase.

Algorithm 1 Extracting the ICs

*Calculate score image S(x0,y0) proportional to negative of the square sum of errors $I(x0+x,\ y0+y)-T(x,y)$, where x and y span the dimensions of T

*Find IC coordinates (x0,y0) that maximize S(x0,y0)

An example of the template module is shown in Figure 5(a). Fig. 5(b) shows the result after matching the template with an image taken from another module. The IC borders can be determined from the corresponding ones in the template image after the template is positioned to the point with the maximum match. Due to different IC misalignment characteristics between the calibration module and the module under processing, the extraction result may still have some error. However, this is not an issue as the exact IC borders need not to be known and the errors in the IC locations occur in significantly smaller scale compared to the

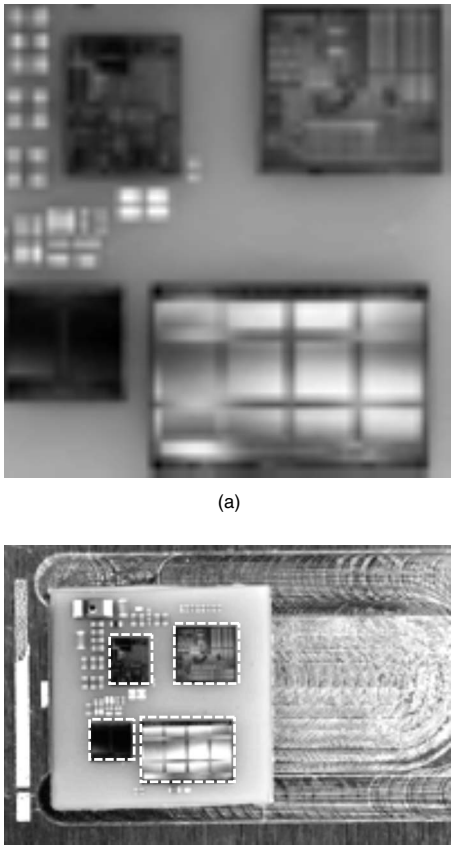


Figure 5. A module layout template (a) with known IC coordinates is matched against the downsampled camera image (b) in order to detect the ICs. The module locations inferred from the manually annotated coordinate data are shown in (b) with dashed rectangles.

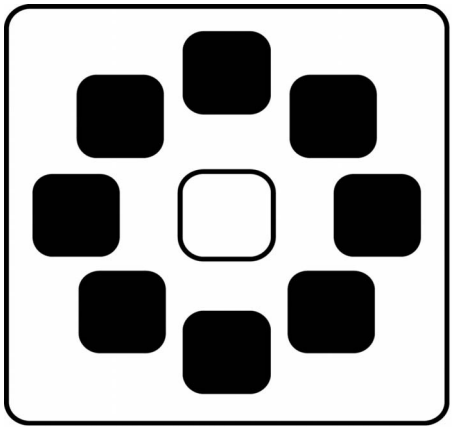


Figure 6. Mask used in the modified logical level method.

module dimensions. The most important thing is that all the connection pads should remain inside the detected IC areas. This can be ensured by adding a small margin outside each detected IC border.

Finding Connection Pad Candidates within the IC Areas

Connection pad finding aims at segmenting feasible candidates for IC connection pad locations. The segmentation step should return preferably too many candidates because the following step uses a more accurate classification algorithm that removes the false matches.

Connection pad extraction is based on the visual appearance of the connection pads. Because of their metallic material, the connection pads are very reflective and thus appear brighter than their surroundings in the image. The appearance is made further distinguishable by circular lighting around the camera objective. This way the reflective connection pads look rather donutlike and are hard to be confused with any other objects on the ICs.

Because of the appearance of the connection pads, the same template matching method to highlight the connection pad candidates can be used as in the IC extraction stage. After matching the template, the connection pads appear in the image as bright, round objects against a darker background (assuming that we denote better match with higher intensity). Thus, an intuitive solution for pad finding is a *blob detector* that highlights the objects and extracts the foreground by thresholding. Usual approaches for highlighting the objects include the Laplacian of Gaussian or some morphological operation (such as the *tophat transformation*).[7] However, a method called *logical level thresholding*[8] essentially performs both highlighting and thresholding in a computationally efficient manner, and seems to produce the best result in our case.

Logical level thresholding was originally proposed for extraction of text from document images. Therefore, it is aimed at extracting character strokes or long curves with similar thickness and brightness. Since our application is slightly different, we apply a modified version of the original method. The basic idea is to look for valleys at all eight possible directions (45° apart) around a single image pixel. The directions can be interpreted as masks, so in our case it is adequate to modify those masks. While the original method uses masks that resemble lines, our single mask simply resembles of the letter "O," see Figure 6. The squares represent neighborhoods which, in fact, overlap each other. The requirement for an object candidate is that all neighborhoods marked with a black square have a mean intensity, which is at least a given number of units lower compared to the mean intensity of the neighborhood in the center.

After the previous step plenty of objects still remain in addition to the connection pads in the binary image. The next step is to use the morphological properties of the found features to eliminate obvious false connection pads. This is done by fitting the shapes with ellipses and filtering them out based on the length of the minor and major axis of the ellipses thus removing all elongated shapes. With this way we can control the area and roundness of the objects which are to be approved as connection pads.

The final stage of finding the connection pad candidates is to generate the coordinates of the found objects. The point representing the location of the found pad candidate is taken

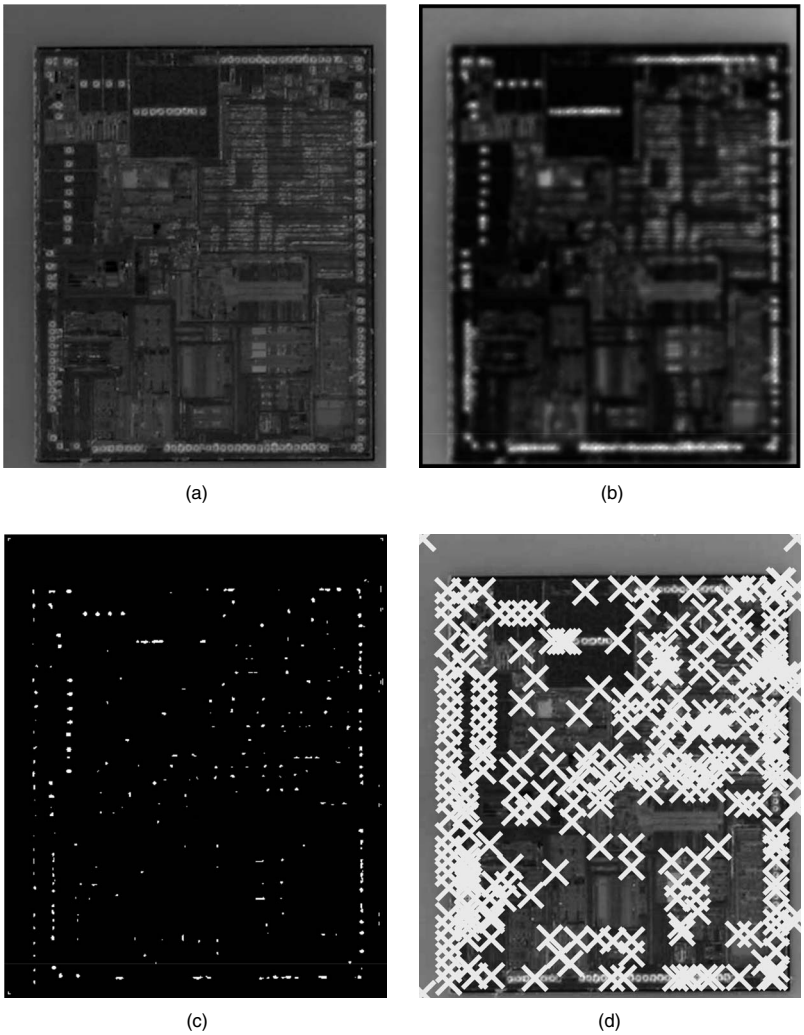


Figure 7. Finding connection pads candidates in IC number one. (a) shows the original IC image, in (b) the image has been matched with a connection pad template, and in (c) there is the result after logical level thresholding and shape filtering. All the candidates are illustrated in (d) with crosses.

to be the corresponding local optimum point in the template matched image. Each object in the thresholded image produces a single coordinate pair with this manner. If the set of coordinates includes two or more objects that are located closer each other than the minimum known distance of two connection pads, only the one having the largest value in the template matched image is taken into account.

There is a step by step algorithm description listed in Algorithm 2. The algorithm takes in the module image I and returns a set of coordinate pairs indicating the locations of the connection pad candidates in the image. A predefined template image T is used like in IC extraction. Parameters needed in logical level thresholding are the stroke width (size of the pencil in the original context) and a relative threshold value telling how much brighter/darker the processed local area has to be on average when compared to the neighboring areas. These two parameters highly affect the number of connection pad candidates that the algorithm produces. The only parameter in shape filtering is the limit for the size of the fitted ellipses. In our experiments we used stroke width equal to 1.4 times the diameter of a connection pad, a threshold value, which is approximately 4% of the range of the values in the template matched image (with 8-bit images this is equal to ten), and allowed ellipse minor/major axis length of [1.0,1.75] measured in terms of connection pad diameters. Using parameter values dependent of the size of the connection pads prevents us from having to tune the parameters each time a new type of connection pad is presented.

Algorithm 2 Finding connection pad candidates

*Calculate score image S(x0,y0) proportional to negative of the square sum of errors $I(x0+x, y0+y)-T(x, y)$, where x and y span the dimensions of T
* Apply modified logical level thresholding for S
* Filter out unrealistic shapes from the resulting binary image
* Inside each remaining object, find coordinates of the corresponding local optimum in S
* Discard duplicate (and near duplicate) coordinate pairs

An example result of the connection pad candidate finding step is illustrated in Figure 7. This module is a dif-

ficult case for the detection because it is dirty from earlier printing tests (after which the ink was removed). Fig. 7(a) shows the original image of one IC, and Fig. 7(b) is the corresponding component after template matching (the template was a connection pad captured from another module). Fig. 7(c) is the result after logical level thresholding and shape filtering. It can be seen that a high number of connection pads remain undetected on areas where the points are densely packed. The reason for this is that after template matching [Fig. 7(b)] the connection pads get bulged (due to small pitch and dirt) and finally get thrown away in the thresholding step [Fig. 7(c)]. Traces of old ink can also been seen in the middle of the IC. Enough of this ink gets detected as connection pad candidates, as the color of the ink is similar to the color of the connection pads. The final detection results are shown in Fig. 7(d)

Classification

As seen in Fig. 7, the connection pad candidate finding step may return more false matches than there are true connection pads. However, the amount of data is now significantly lower compared to where we started. This makes it possible to use a high-end binary (two-class) classifier for classifying each connection pad candidate into a true connection pad or a spurious object. For this purpose we use artificial neural networks (ANNs).[9]

ANNs were selected as the classifier because of their flexibility. They can be trained by showing examples of connection pads and typical candidate objects that are not connection pads. The training process is always run when a new module with new connection pad appearance is introduced (like we did in the experiments *Experiment 2: Printing Ten Multi-IC Modules* and *Experiment 1: Connectivity Test*, for example). Experiments have shown that a single module with a couple hundred connection pads and approximately the same number of spurious objects is adequate to complete the training.

The ANN used in the classification is a fully connected feedforward network with 243–25–1 structure. In other words, the network has 243 inputs, one hidden layer with 25 neurons, and a single output neuron telling whether the input pattern is a connection pad or not. Size for the input layer is chosen such that the ANN can take in size 9×9 RGB images of connection pad candidates ($9\times9\times3=243$). Image size 9×9 is chosen as it is rather small but can still represent the connection pad with an adequate resolution. A proper size for the hidden layer giving the best performance together with the ability for the classifier to generalize was selected by training the ANN multiple times with different amount of neurons on the hidden layer and then measuring its performance, following the well-known approach called *network pruning*.[9]

A general pattern recognition ANN has one an output neuron for each class. The values of the output neurons give confidence levels for the input object belonging to a particular class. In the case of a binary classifier it is possible to use a single output neuron instead of two. We use this single neuron solution together with hyperbolic tangent as the output neuron's activation function. This makes the ANN output to be a single scalar on the interval $(-1,1)$, where positive outputs indicate connection pads and negative outputs indicate spurious objects.

Before feeding to the ANN, each object image is resized into the required 9×9 size (by using bicubic resampling) and blurred. Blurring suppresses the noise in the images, thus, giving the ANN a better performance and capability to generalize. It also makes the classifier more invariant to small shifts in the input image. The blurring is done by convolving the images with a Gaussian convolution kernel.

After the network has been constructed and trained (by using a training algorithm like *scaled conjugate gradient back-propagation*[10]), the classification step still needs a suitable decision boundary, i.e., a threshold value of the connection pad confidence level that can tell whether an object is a connection pad or not. The most obvious solution is to a threshold value equal to zero. However, we can exploit the knowledge about the number of real connection pads on the module/IC by picking up only the most likely connection pads such that the number of positive classifications is at most a fraction of the true number of connection pads. The higher the fraction, the more false positives (spurious objects classified as connection pads), and the lower the fraction, the more false negatives (connection pads classified as spurious objects) will be included.

Algorithm 3 Classifying connection pad candidates

```
* For each subimage J around points (X,Y) in I
  - Resize J for ANN input
  - Blur J
  - Calculate and store ANN output for J
* Sort ANN outputs
* Classify at most M objects with largest nonnegative ANN outputs as connection pads
* Classify remaining objects as spurious objects
```

A good compromise is to restrict the amount of positive classifications to be at most 80% of the amount of true connection pads. Justifications for the particular percentage has been presented in our earlier experiments.[4] Notice that the reason for using this kind of restriction is that we need to avoid getting too much false positives. Later in the *back end* part we will see that the false positives are more harmful than false negatives.

As a conclusion, an algorithm description of the classification step is listed in Algorithm 3. The algorithm takes in the module image I and the coordinates (X,Y) of the detected connection pad candidates. It uses a pretrained ANN that contains information on two parameters needed: the input image size and the convolution kernel used in blurring the input images. The algorithm also uses a parameter M, which determines the maximum number of positive classifications allowed. In our experiments these three parameter values correspond to 9×9 RGB image, Gaussian with standard deviation of 0.1 pixels, and 80% of the total number of

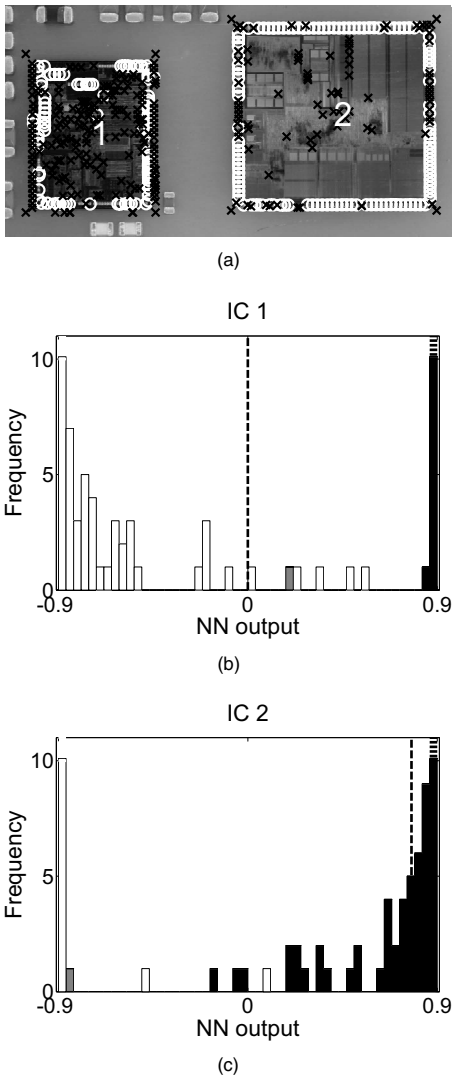


Figure 8. An example of the connection pad classification. In (a), white circles denote pad candidates classified as connection pads and black crosses denote those classified as spurious objects. (b) and (c) show the histograms of the ANN outputs for all connection pad candidates. The shade of the bars indicate the *true* class of the points (black=true connection pad, white=true spurious object). Overlapping parts of the black and white bars are indicated with gray. The highest bars have been truncated as they have approximately 80% of all the hits. Black dashed lines indicate the decision boundaries used in generating (a). These are calculated as described in Algorithm 3.

connection pads on the particular module, respectively. Notice that in the training stage of the ANN, exactly the same preprocessing steps are done to the input images as when actually using the ANN.

Figure 8 shows an example of the classification result when the 80% rule has been applied on each IC. Classification results of two different ICs are illustrated in Fig. 8(a). Figs. 8(b) and 8(c) show the histograms of the ANN outputs together with the decision boundaries. The connection pad candidates that produce network outputs above the decision boundary are considered as connection pads and those that produce outputs below the boundary as spurious objects.

In the case of the IC number one in Fig. 8(a), many of the connection pads have been completely left undetected in the earlier stage. Hence, all the candidates producing positive output are classified as connection pads. Like seen on the histogram in Fig. 8(b), this produces seven false positives (spurious objects classified as connection pads). All the positive classifications in the case of the IC number two are true positives like those that can be seen on the histogram in Fig. 8(c). In this case, only the most obvious candidates are classified as connection pads such that the amount of positive classifications does not exceed 80% of the true amount of connection pads in that IC. There are plenty of false negatives (connection pads classified as spurious objects). However, this will not become a problem as will be seen in the next section.

BACK END

The purpose of the *back end* part of the process is to cover the next two blocks in the system diagram in Fig. 2. These blocks are *match features with design features* and *create corrected design data*. The process is described in the following three subsections: Pairing the Point Sets, Determining the IC Transformations, and Correction.

The general idea of back end is the following: First, we seek for a one-to-one correspondence between the detected and the designed connection pads. Second, a transformation model is formulated for each IC that transforms the displaced connection pads to their correct locations. Finally, the actual print pattern is adjusted in order for it to match with the true component placement. The constraints for the adjustment are twofold: all IC connection pads must match the intended point in the print file, while the contacts for the following layers should stay as they are. The latter condition is required in order not to propagate the errors further to the following layers.

Pairing the Point Sets

Determining the displacements of the connection pads starts by finding a correspondence between the connection pads in the design and the ones detected from the module image. This is done for each IC separately. Since the front end result may have missing connection pads as well as false ones, the goal is to find well-matching subsets of both input sets.

The point pattern matching algorithm of Chang et al.[11] is used for finding the correspondence between the detected and designed connection pads, which are now considered as two two-dimensional point sets. Although not the most efficient one in terms of time complexity, the method is robust against point sets with both missing and extra points as well as against point sets having local similarities in point locations. These are key advantages in our case when comparing to several other algorithms.

Generally, the point pattern matching algorithm finds a transformation between any two point sets. The resulting transformation is a *similarity transformation*, which transforms a point $(x_1, y_1) \in \mathbb{R}^2$ in the first set into a point

$(x_2, y_2) \in \mathbb{R}^2$, which is as close to the corresponding point in the second set as possible,

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = s \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}, \quad (1)$$

where $\theta \in [0, 2\pi]$ is the rotation angle, $s \in \mathbb{R}_+$ is the scale, and $t_x, t_y \in \mathbb{R}$ are the translation terms.

At this point, we are not interested in the transformation parameters provided by the point pattern matching algorithm because the IC transformations are *rigid* (Rigid transformation is like similarity transformation but the scaling term being equal to one. In other words, rigid transformation only consists of rotation and translation.), not similarity transformations. Instead, the optimal transformations are used for inducing the point pairs separately by transforming the design data of each IC. After this the two point sets are paired using the Euclidean distance as the criterion. A rule of thumb is that every connection pad in the design is paired to the closest object detected in the connection pad candidate finding step in front end as long as it is not more than 5 μm away. This way also the connection pads that the front end classifies as a poor match are paired.

As an output, the coupling of the pads returns information whether the design connection pads were linked to some true connection pad or not. Missing pairs usually indicate that some of the connection pads could not be found from the module image, for example due to dust or other occluding objects.

Determining the IC Transformations

After the correspondence between the detected connection pads and the designed connection pads has been established, the next thing to do is to determine the IC transformations and the errors that these transformations bring into the locations of the connection pads. This is done in two steps. First, a global alignment transformation is determined. Second, the optimal IC transformations are determined.

Let us first consider the global alignment of the printed pattern. The purpose of the alignment transformation is to rotate and translate the design into the same coordinate system with the module in such a way that the print pattern approximately matches the module. The alignment transformation is thus a rigid transformation. As there are not any alignment marks, three different strategies for the alignment are proposed:

(a) Do the alignment such that the costs of the final correction of the design data will be as low as possible. In the sequel, we call this alternative the *mean distance approach*.
(b) Do like previously but favor small-pitch areas such that the correction is made easier on areas with closely located connection pads.
(c) Minimize the distances for all connection pads on a single IC (so-called calibration IC). Because the connection pad distances within an IC are exact, the match for the calibration IC will be very accurate. We will call this alternative the *single IC approach*.

Algorithm 4 Finding IC transformations

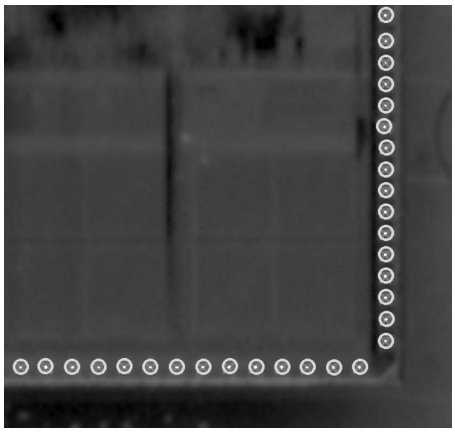
* Assign a weight for each connection pad according to the chosen alignment scheme
* Find the parameters of the optimal alignment transformation (XD, YD) → (XM, YM)
* Apply inverse alignment transformation for (XM,YM) to obtain (XA,YA)-coordinates of the detected connection pads aligned with the design
* For each IC:
  - Find the parameters of the optimal IC transformation (XD[i], YD[i]) → (XA[i], YA[i]), where index i refers to the coordinates of the connection pads on the IC

All three approaches can be represented in a single framework: weighted least-squares. Setting the weights equal gives the mean distance approach. Giving more weight on connection pads with close neighboring connection pads results in the second of the proposed strategies. Setting the weights to unity for the calibration IC and zero for others gives the single IC approach.

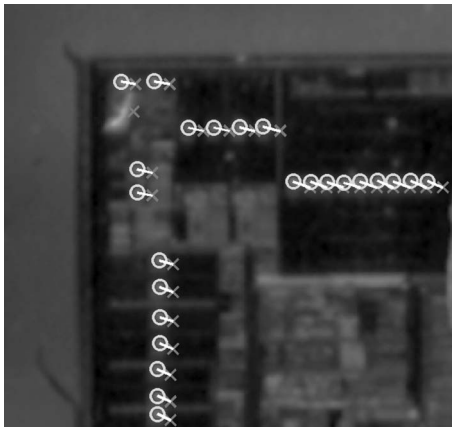
Next, we need to determine the parameters of the rigid transformation that carries out the alignment in an optimal way. Unfortunately, the least-squares solution for the rigid transformation—especially in the weighted case—is nontrivial. Such a solution has been proposed by Schaefer et al.[12] and Müller et al.[13] However, their solution is slightly invalid, which can in some cases result in a rotation matrix that performs reflection instead of rotation. This can be avoided by using the determinant condition introduced earlier by Umeyama[14] for the unweighted case.

After the alignment has been done, the IC transformations are determined, again, by using the solution for the optimal rigid transformation (without weighting). The whole procedure is described in Algorithm 4. The algorithm takes in a set of design connection pad coordinates (XD,YD) and a set of detected connection pad coordinates (XM,YM). These two sets have equal amount of points and they have been sorted in the pairing step such that the corresponding points form a pair. The algorithm returns the parameters (rotation and translation) of each IC transformation. Also the parameters of the alignment transformation are returned because these will be useful in the actual printing phase (see the section Printing Multiple Modules Simultaneously).

The pairing of the connection pads after applying the alignment transformation is illustrated in Figure 9. In this example, we used the single IC approach as the alignment method. The calibration IC with nonzero weights is the IC shown in Fig. 9(a). Because there is no distortion in the IC dimensions, the match is perfect. However, the mismatch in the IC in Fig. 9(b) is clearly visible because of the compression of the background substance, which becomes even more severe for ICs residing further away from the calibration IC. In this case, the two blocks are roughly 9 mm away (for reference, the block width is about 2 mm). Note that one connection pad does not have a pair due to the dust



(a)



(b)

Figure 9. Pairing the design data with the detection result. The figures show the match with the true locations of the connection pads. (a) shows a part of the calibration IC, and (b) is a part of another IC.

particle in Fig. 9(b). This, however, is not a problem as the determined overall IC transformation is sufficient for conducting the correction of the interconnections.

Correction

After the IC transformations have been discovered, the remaining step is to adjust the wiring diagram to match the individual module. First, displacement vectors $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \ldots, \boldsymbol{\delta}_N \in \mathbb{R}^2$ are calculated as the difference between the designed connection pad locations and those mapped with the IC transformations. Here N denotes the total number of connection pads on the module. Each displacement vector defines the required displacement for a corresponding wire end in the wiring diagram to match the right connection pad.

The correction method we have proposed[6] processes the data on the bitmap level and does not employ any information about the structure of the design data; for example, the wiring and the dielectric layers are treated equally. The method is based on a displacement function $\mathbf{d}: \mathbb{R}^2 \to \mathbb{R}^2$, which determines the amount of correction needed in horizontal and vertical directions for each point in the original wiring diagram.

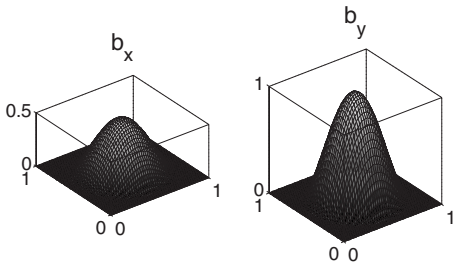


Figure 10. An example of smooth bump created with piecewise biquadratic polynomial with $(x_i, y_i)=(0.5, 0.5)$, $\boldsymbol{\delta}_i=[0.51]^T$, and $a=0.4$ (making the correction radius R_i equal to 0.45).

The derivation of the displacement function follows the idea of blobs.[15] Each wire end or design connection pad should create a smooth blob or bump onto the displacement function. Certain requirements are set for each bump. The most obvious requirement is that the ends of the wires should be translated onto the true connection pads, i.e., the bump heights are equal to the displacement vectors $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \ldots, \boldsymbol{\delta}_N$. A secondary criterion is *smoothness*, in the sense that the wiring should not have sharp corners or discontinuities that might harm the functionality of the module. Finally, the bumps *should have a finite support*, i.e., the correction for each connection pad should take place inside a limited area.

An example of a function satisfying the requirements set for the bump model is a particularly chosen biquadratic polynomial. In this case the definition of the N bump functions $\mathbf{b}_i$ $(i=1, \ldots, N)$ evaluated at point $(x,y) \in \mathbb{R}^2$ can be formulated as

$$\mathbf{b}_i(r) = \begin{cases} \left(\dfrac{r^4}{R_i^4} - \dfrac{2r^2}{R_i^2} + 1 \right) \boldsymbol{\delta}_i, & \text{when } r < R_i, \\ \mathbf{0}, & \text{otherwise}, \end{cases} \tag{2}$$

where $r=\sqrt{(x-x_i)^2+(y-y_i)^2}$ is the distance of point (x,y) from the ith design connection pad (x_i,y_i). Parameter R_i is the correction radius allowed for the wire end connected to the ith connection pad.

Since the amount of required correction varies in different areas of the module, it is reasonable to make the correction radii R_i adaptive and dependent on the correction distance. This can be achieved, for example, simply by setting the radius proportional to the connection pad displacement: $R_i=a\|\boldsymbol{\delta}_i\|$, where the constant $a \in \mathbb{R}^+$ is so called *smoothness parameter*. The larger the parameter a, the larger the correction area and the smoother the correction. An example bump is shown in Figure 10.

The second step is to create the actual displacement function $\mathbf{d}$ by adding all the individual displacements together. Simple summation is not adequate since we want to preserve the bump properties (height, smoothness, and finite support) in the displacement function. This can be achieved by introducing a weighting function w_i for each bump $\mathbf{b}_i$,

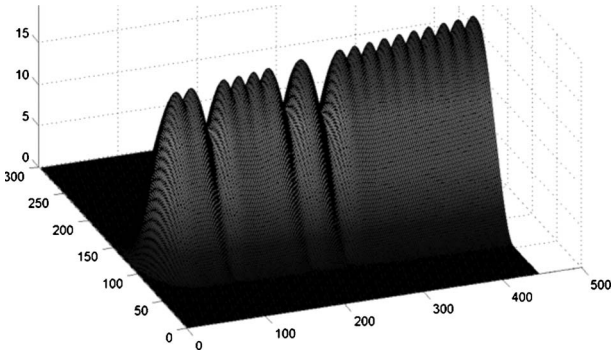


Figure 11. A part of the displacement map used in Fig. 13 (only horizontal displacement shown; the vertical component of the displacement map is similar).

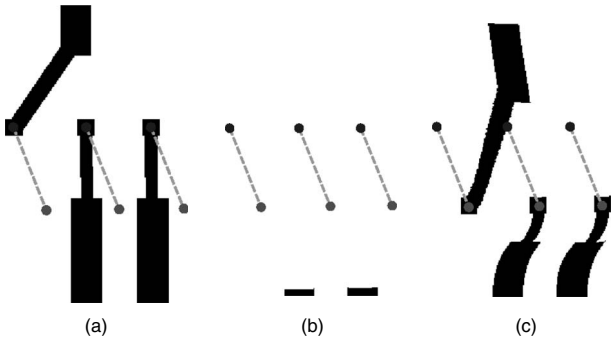


Figure 12. Illustration of the correction process. (a) shows the connection pads and wiring before correction. In (b), the wire ends have been erased and in (c) the wire ends have been redrawn using the proposed method.

$$w_i(r) = \begin{cases} \dfrac{1}{2r^2R_i^2 - r^4} - \dfrac{1}{R_i^4}, & \text{when } r < R_i, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The weight is zero outside the correction radius R_i and goes smoothly toward infinity when approaching the bump center.

By using the weighting functions and the individual bumps, the actual displacement function is defined as

$$\mathbf{d}(x,y) = \frac{\sum_i w_i(x,y)\mathbf{b}_i(x,y)}{\sum_i w_i(x,y)}. \tag{4}$$

The definition is simply a weighted sum of all the bumps $\mathbf{b}_i$. For simplicity, we have omitted two special cases when $\mathbf{d}$ is undefined: when (x,y) is one of the connection pads (x_i,y_i), we should define $\mathbf{d}(x,y)=\boldsymbol{\delta}_i$, and when (x,y) is outside the radius R_i of any correction point (x_i,y_i), we should define $\mathbf{d}(x,y)=0$. An example of a combined displacement function is shown in Figure 11.

After a displacement function has been established, the actual correction can be done. Figure 12 shows the stages of the correction. In Fig. 12(a) there is the original wiring bitmap. The connected dots indicate the wire end points without correction and the desired target locations. The first stage of the correction—illustrated in Fig. 12(b)—is to erase the falsely oriented wire ends. The erasing is limited to only those pixels that have nonzero displacement according to the displacement function. After the wire ends have been erased, each erased wire pixel is remapped according to the displacement function. Figure 12(c) shows the corrected wiring in the example case.

The algorithm description of the complete correction procedure of the wiring bitmap is given in Algorithm 5. The algorithm takes in the original wiring bitmap, the coordinates of the connection pads, and the displacement vectors. It returns the corrected wiring bitmap. The smoothness parameter is needed in the first step when calculating the correction radii around each connection pad.

Algorithm 5 Correcting the wiring bitmap

* Calculate correction radii
* Evaluate displacement function inside each correction radii
* Construct corrected wiring bitmap by remapping the wire pixels according to the displacement function

An example result of applying the correction method is shown in Fig. 13. Fig. 13(a) shows a part of the module ready for printing. However, due to component drifting, the connection pads have translated, and printing the design bitmap directly will result in failed connections, see Fig. 13(b). Note that this displacement cannot be corrected by translation because that would increase the errors on other components of the module (this is actually the mean distance fit minimizing the square errors of all the wire ends in every IC). The correction results are shown on Fig. 13(c). The displacement function used in the example is the particular one seen earlier in Fig. 11. As expected, the connection pad pattern is clearly visible.

PRINTING MULTIPLE MODULES SIMULTANEOUSLY

A major advantage in using an online print pattern generation system is that it enables the printing of multiple modules with a single printing run. By detecting the locations of the modules and the module components, it is possible to automatically apply the alignment rotations of each print pattern and then to catenate the patterns into a single large panel.

One of the advantages of ink jet printed electronics as an electronics manufacturing method is the scalability of the printable area. This valuable asset becomes more efficiently utilized when printing in panels. After minimizing the total printing time as well as the amount of operators' manual labor, and after taking better advantage of the available printing area that the printer can cover on a single run, the advantages gained when being able to print multiple modules at same time seem to be indisputable. The improvements in performance lead to a better printing quality together with savings in time, energy, and material.

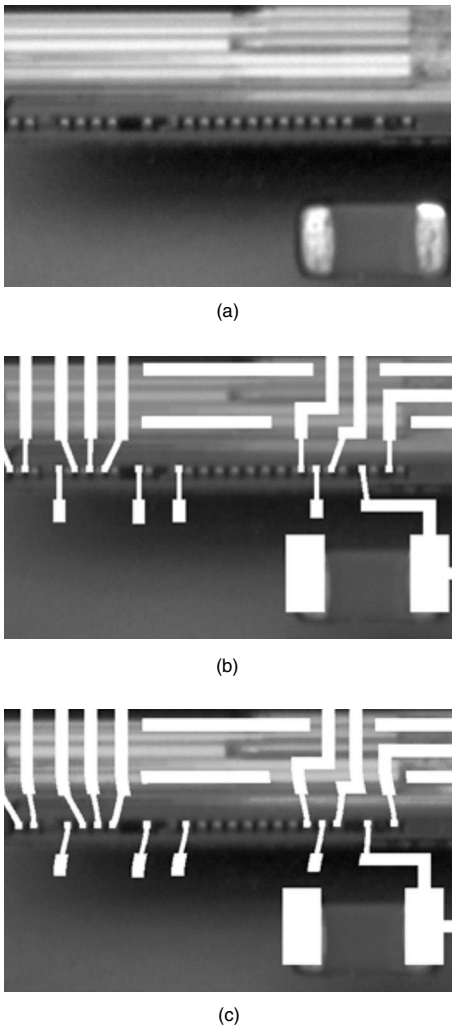


Figure 13. Original module ready for printing is shown in (a), and the simulated printing result without correction in (b). The simulated result after correction is shown in (c).

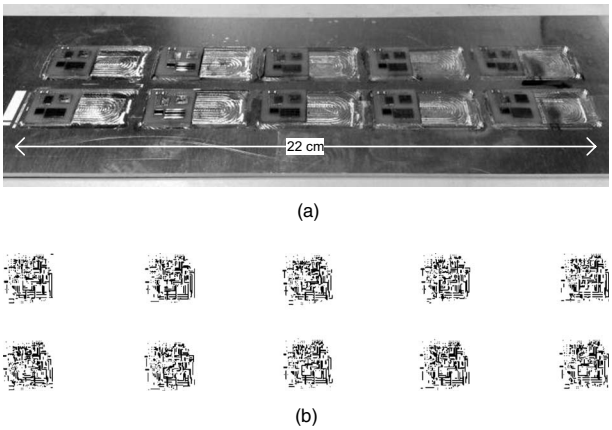


Figure 14. (a) shows a 2×5 panel of modules. (b) shows the wiring that is going to be printed on top.

In case of multiple modules, the print pattern correction is run separately for each module. First, the modules are individually photographed. After this the images are processed for detecting the modules and, finally, corrected print patterns are generated for each module. If the lateral shift between the module images is precisely known, a single large print pattern can be put together. This print pattern covers all the modules at once and it can be used similarly to conventional patterns with only one module on them. There is a photograph of multiple modules on a slotted plate used for approximate positioning of the modules in Figure 14(a). The corresponding print pattern—which has been created by individually running the correction system on each module and then catenating the resulting print patterns—is shown in Fig. 14(b).

The first issue to be noted when compiling a large print pattern is the rotational error, which becomes more and more significant as the dimensions of the panel increase. In our case, the width of the panel in Fig. 14(a) in the head movement direction is about 22 cm. Thus, a rotational error of 1°, for example, would cause a disparity of $22\ \text{cm}\times\sin(1°)\approx3.8$ mm at the right end of the panel. The disparity scales almost linearly near 0°. Thus, a negligible looking rotational error of 0.01° would still result in 38 μm error, which is in the order of the pitch of the connection pads.

Conventionally, the rotational error in the print pattern is manually compensated by rotating the printer table and, thus, errors larger than 0.01°—like in the previously worked out example—are easily created. This suggests that it would be nearly impossible to print the panel in Fig. 14 with the conventional method. However, by using the print pattern generation system, the relative locations and orientations of the modules on the panel are automatically determined by using the alignment transformation estimated in the section Determining the IC Transformations. Further on, an accurate rotation function of the printer table is made completely redundant.

The last thing to do before the layout of the print pattern covering the whole panel is fully determined is to compensate the camera rotation. Obviously, as the connection pad locations have been measured from the camera image, the rotation of the camera affects the estimated alignment rotations of the modules. One method for finding the camera rotation is to take multiple images from a static control point such that a small amount of translation to a given direction (i.e., the direction of the printer head movement) is applied between each image, see Figure 15. The trajectory of the control point in the images forms a straight line. The direction of this line in the camera view can be compared against the direction of the applied translation. The difference between these two directions gives the camera rotation. This kind of camera calibration step obviously needs to be carried out only once, i.e., after mounting the camera.

EXPERIMENTAL RESULTS

In our earlier articles, we have considered the compensation of error on a single module level.[4–6] In this section we concentrate on printing multiple modules in one printing run,

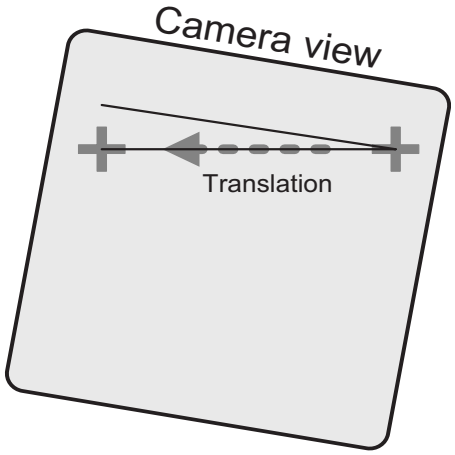


Figure 15. The angle of camera rotation can be determined by translating a static target in the X-Y-stage horizontally. The camera angle can be detected from the horizontal and vertical displacements in the camera view.

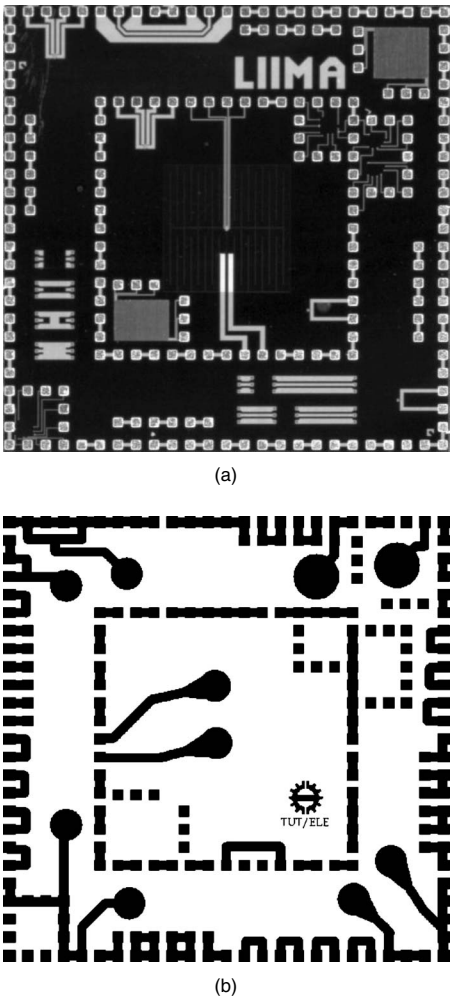
while compensating possible intramodule errors simultaneously. The experiments consist of two parts: First we consider printing five modules that are used for testing the connectivity of the printed patterns. The second experiment studies printing the interconnections of ten multi-IC modules simultaneously.

Experiment 1: Connectivity Test

The first experiment considers printing of five modules together with the verification of the electric functionality of the result. The wiring is printed on a special *daisy chain* module that is designed especially for connectivity testing. The connection pads are chained together and should create an integral connection between test points.

The modules used and the corresponding wiring image are shown in Figures 16(a) and 16(b), respectively. The connection pads differ from the ones in Experiment 2 (and in earlier discussion) by their size, shape, and color. They are also located all around the IC, not only on the edges. The diameter of the connection pads is about 220 μm, which makes them a significantly larger than the ones in the later experiment (40 μm).

As the test module consists of only one component, there occurs no component drifting that would make the print pattern not to match the module layout. In this experiment, component misalignment is artificially created by not aligning the print pattern optimally on the module but by using a fixed point on the background on the printer table to align each print pattern. The displacement function based correction is run separately for each module and a single print pattern is finally put together and printed.



(a)

(b)

Figure 16. (a) shows the type of module used in the first experiment. (b) shows the wiring image, which is going to be printed on the module.

Figure 17 shows the printing result, where multiple photographs have been catenated as a panorama image showing all the printed modules. The dashed line squares indicate the places where the print patterns have been aligned to. The error created by the intentionally careless module placement is corrected by detecting the true connec-

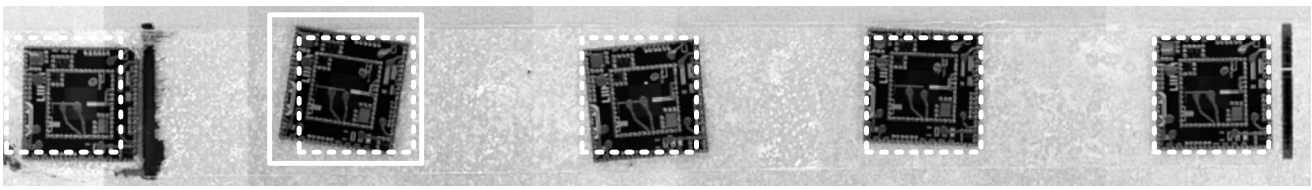


Figure 17. Printing result of the first experiment. Dashed line squares show where the modules are expected to be and where the print pattern is initially aligned. The module inside the solid line square can be seen in detail in Fig. 18.

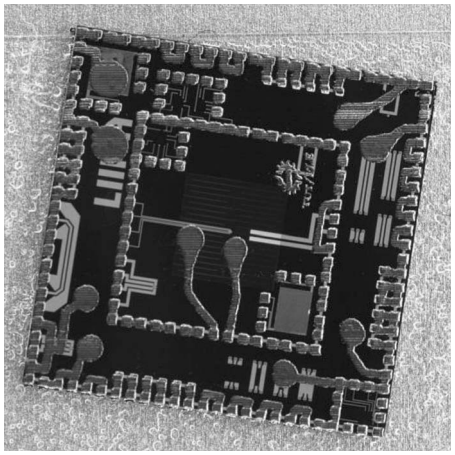


Figure 18. Zoomed printing result of the second module on the left in Fig. 17.

tion pad locations and by using the displacement function correction as described. The area surrounded by the solid line square in Fig. 17 is shown in larger scale in Figure 18. All the printed modules were verified to be electrically functional, as desired.

Experiment 2: Printing Ten Multi-ic Modules

In the second experiment, the wiring is printed on multi-IC modules such that ten modules are printed simultaneously. The module design is the one illustrated back in Fig. 1. The module is square shaped with each side 17 mm long. The diameter of the connection pads is 44 μm, and their pitch is typically 40–70 μm. The modules are printed in a panel in a similar manner as in the first experiment. The panel layout is the same as was already seen in Fig. 14. In this experiment, the detection of the connection pads is more challenging than in the first experiment. This is because of two reasons. First, the connection pads are now smaller in size. Second, the same modules have already been used in previous test printings and there are distracting ink traces on their surface.

Table I summarizes the connection pad detection results for all ten modules. The second column from the left shows the number of connection pad candidates resulting from the step described in the section Finding Connection Pad Candidates Within the IC Areas. In all there are 409 true connection pads in each module and typically half of the pad candidates are false. The third column shows the number of true connection pads that are missed already in the candidate finding phase. The obtained candidates are classified either as connection pads or spurious objects by using an ANN classifier. The fourth and fifth columns show the number of positive and negative classifications that were correct, respectively. By using this information, the performance of the binary classifier can be determined by examining the sensitivity and specificity of the classification result, which are the two last columns in Table I.

Consider the results shown in Table I. Sensitivity characterizes the classifier's ability to find true positives, while specificity characterizes its ability to rule out false matches. Sensitivity is calculated as the ratio of the number of true positive classifications and the total number of actual connection pads among all the candidate objects. The number of true positive classifications in our experiment sums up to 3137. There are altogether $10\times409=4090$ actual connection pads but 301 of them are missed in the candidate finding step so only $(4090-301)=3789$ of the actual connection pads gets to the classifier. Thus, the total sensitivity of the classifier is $3137/3789\approx82.79\%$. Specificity, in turn, is calculated as the ratio of the number of true negative classifications and the number of actual spurious objects among the candidate objects. The total number of true negative classifications in our case is 6870 and the total number of spurious objects is $10,738-3789=6949$ (all candidate objects minus the ones that are connection pads). Thus, the overall classifier specificity is $6879/6949\approx98.86\%$.

Since the classier specificity is of more interest to us than the sensitivity, we limited the number of positive classifier outputs to be at most 80% of the number of true connection pads like explained in the section Classification. As there are 409 connection pads on each module, at most 327 positive classifications are made in all ten cases (Notice the phrasing at most, as objects with negative classifier output are discarded even if they would fit inside the 80% margin.). From the specificity values of Table I it can be clearly seen that false positives are rare. The sensitivity values, on the other hand, are naturally determined by the 80% rule. The reason why the actual values are slightly above 80% is that the 80% is calculated with respect to the true number of connection pads (409) while in calculating the actual sensitivity the number of undetected connection pads is first subtracted.

The printing result of the second experiment is shown in Figure 19. The image has been created by catenating the individual module images after a rotation of $-0.19°$, which is the angle between the camera and the printer table determined during the calibration of the camera (see the section Printing Multiple Modules Simultaneously). Black color is used for filling the spaces such that the layout of the image equals the layout of the panel. All the modules have unique orientations. Even the whole jig holding the modules seems to be slightly rotated because the operator placed the jig manually on the printer table. All these unidealities are compensated by the correction system that creates the print image.

A close up of the printing result is shown in Figure 20. The image shows the result from the area of one IC. Like in all the other ICs, the wire ends meet the right connection pads. The thinnest parts of the wires consist of only one line of ink droplets. For this particular printer and droplet size, the scale of the used modules is close to the smallest possible. In the picture, small horizontal artifacts in the printing quality can be seen. These occur due to failures in individual printer nozzles.

Table I. Detection results for ten modules. The module marked with a star (⋆) was used for ANN training.

Module	*Candidates*	*Undetected*	*True pos.*	*True neg.*	*Sensitivity*	*Specificity*
1	1074	28	307	673	**80.58%**	**97.11%**
2	1012	32	322	634	**85.41%**	**99.84%**
3	1070	31	311	681	**82.28%**	**98.41%**
4	1147	22	320	753	**82.69%**	**99.08%**
5*	1048	37	303	661	**81.45%**	**97.78%**
6	1168	18	319	769	**81.59%**	**98.97%**
7	1188	20	322	794	**82.78%**	**99.37%**
8	1010	39	313	636	**84.59%**	**99.38%**
9	1012	48	300	647	**83.10%**	**99.39%**
10	1009	26	320	622	**83.55%**	**99.36%**
Tot.	10738	301	3137	6870	**82.79%**	**98.86%**

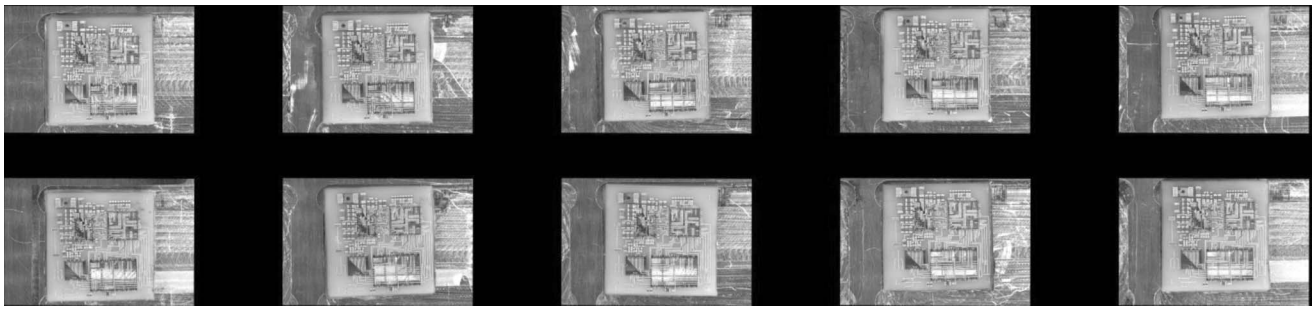


Figure 19. The printing result of the second experiment.

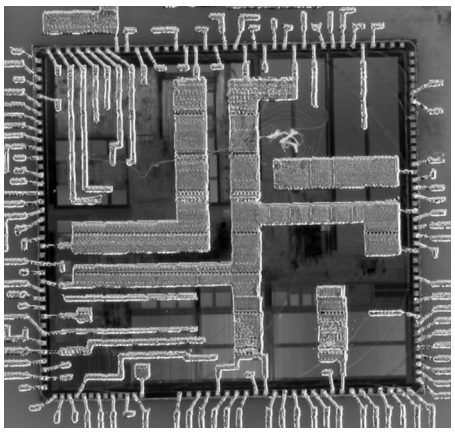


Figure 20. A close up from the printing result.

CONCLUSIONS

In this article, we have presented a method for dynamic adaptation of interconnections in printed electronics. In particular, we concentrate on the estimation of displacements in connection pad locations, and furthermore, correction of these displacements. The correction can be extended into simultaneous printing of multiple modules, which allows easier and faster printing of larger volumes of printed modules.

Experimental results indicate that the method can successfully adapt to unpredictable displacements of the components, as long as the displacements are within a reasonable range. The displacement allowed for a component before the correction fails depends on the module layout and the pitch of the connection points on the ICs. Although this may be considered as a limitation of the proposed method, practice has shown that serious faults in component placement or molding process of the modules need to take place before the correction fails.

As a conclusion, significant improvements in the manufacturing yield can be expected after integrating dynamic correction into the printing system. The fact that misalignment of the structures is a critical bottleneck preventing large scale application of printed electronics in miniature size makes the application area very significant for the future of electronics industry.

ACKNOWLEDGMENTS

The authors would like to acknowledge the Finnish Funding Agency for Technology and Innovation (TEKES) and its FinNano-program for the financial support.

REFERENCES

[1] J. Miettinen, V. Pekkanen, K. Kaija, P. Mansikkamäki, J. Mäntysalo, M. Mäntysalo, J. Niittynen, J. Pekkanen, T. Saviauk, and R. Rönkkä, "Ink jet printed System-in-Package design and manufacturing", Microelectron. J. **39**, 1740 (2008).

[2] K. Kaija, V. Pekkanen, M. Mäntysalo, and P. Mansikkamäki, "Controlling warpage of molded package for ink jet manufacturing", Microelectron. Eng. **85**, 518 (2008).
[3] M. Moganti, F. Ercal, C. Dagli, and S. Tsunekawa, "Automatic PCB inspection algorithms: A survey", Comput. Vis. Graph. Image Process. **63**, 287 (1996).
[4] H. Huttunen, P. Ruusuvuori, T. Manninen, K. Rutanen, R. Rönkkä, and A. Visa, "Object detection for dynamic adaptation of interconnections in ink jet printed electronics", *Proc. of IEEE ICIP'08* (IEEE, Piscataway, NJ, 2008), pp. 2364–2367.
[5] H. Huttunen, P. Ruusuvuori, T. Manninen, K. Rutanen, and R. Rönkkä, "Dynamic adaptation of interconnections in ink jet printed electronics", *Proc. of ICSES'08* (IEEE, Kraków, Poland, 2008), pp. 449–452.
[6] H. Huttunen, T. Manninen, K. Rutanen, P. Ruusuvuori, R. Mäkinen, and R. Rönkkä, "Dynamic correction of interconnections in printed electronics manufacturing", *Proc. IS&T's NIP25: International Conference on Digital Printing Technologies* (IS&T, Springfield, VA, 2009, pp. 589–592.
[7] R. Gonzalez and R. Woods, *Digital Image Processing*, 2nd ed., (Prentice-Hall, Upper Saddle River, NJ, 2002), pp. 205–208.
[8] M. Kamel and A. Zhao, "Extraction of binary character/graphics images from grayscale document images", CVGIP: Graph. Models Image Process. **55**, 203 (1993).
[9] S. Haykin, *Neural Networks: A Comprehensive Foundation* (Prentice-Hall, Upper Saddle River, NJ, 1999).
[10] M. Møller, "A scaled conjugate gradient algorithm for fast supervised learning", Neural Networks **6**, 525 (1993).
[11] S. Chang, F. Cheng, W. Hsu, and G. Wu, "Fast algorithm for point pattern matching: Invariant to translations, rotations and scale changes", Pattern Recognit. **30**, 311 (1997).
[12] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares", ACM Trans. Graphics **25**, 533 (2006).
[13] M. Müller, B. Heidelberger, M. Teschner, and M. Gross, "Meshless deformations based on shape matching", ACM Trans. Graphics **24**, 471 (2005).
[14] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns", IEEE Trans. Pattern Anal. Mach. Intell. **13**, 376 (1991).
[15] J. Blinn, "A generalization of algebraic surface drawing", ACM Trans. Graphics **1**, 235 (1982).