Automated Via Detection for PCB Reverse Engineering

Ulbert J. Botero, David Koblah, Daniel E. Capecci, Fatemeh Ganji, Navid Asadizanjani, Damon L. Woodard, Domenic Forte

Florida Institute for Cybersecurity Research, University of Florida, Gainesville, Florida, United States
{Jbot2016, dkoblah, dcapecci, fganji}@ufl.edu, {nasadi, dwoodard, dforte}@ece.ufl.edu

Abstract

Reverse engineering (RE) is the only foolproof method of establishing trust and assurance in hardware. This is especially important in today's climate, where new threats are arising daily. A Printed Circuit Board (PCB) serves at the heart of virtually all electronic systems and, for that reason, a precious target amongst attackers. Therefore, it is increasingly necessary to validate and verify these hardware boards both accurately and efficiently. When discussing PCBs, the current state-of-the-art is non-destructive RE through X-ray Computed Tomography (CT); however, it remains a predominantly manual process. Our work in this paper aims at paving the way for future developments in the automation of PCB RE by presenting automatic detection of vias, a key component to every PCB design. We provide a via detection framework that utilizes the Hough circle transform for the initial detection, and is followed by an iterative false removal process developed specifically for detecting vias. We discuss the challenges of detecting vias, our proposed solution, and lastly, evaluate our methodology not only from an accuracy perspective but the insights gained through iteratively removing false-positive circles as well. We also compare our proposed methodology to an off-the-shelf implementation with minimal adjustments of Mask R-CNN; a fast object detection algorithm that, although is not optimized for our application, is a reasonable deep learning model to measure our work against. The Mask R-CNN we utilize is a network pretrained on MS COCO followed by fine tuning/training on prepared PCB via images. Finally, we evaluate our results on two datasets, one PCB designed in house and another commercial PCB, and achieve peak results of 0.886, 0.936, 0.973, for intersection over union (IoU), Dice Coefficient, and Structural Similarity Index. These results vastly outperform our tuned implementation of Mask R-CNN.

I. Introduction

Two terms that dominate discussion of modern technology are automation and cyber-security, albeit for different reasons. In combination with advances in machine learning, computer vision, and pattern recognition, the computational abilities of our electronics today paint a picture of a future, where a computer can instead perform various burdensome tasks. The pros for automation are numerous, notably benefits that include increased efficiency, improved throughput, reliability, and reduced manual labor. Much of the modern-day discussion around cyber-security is of a different tone. Constant preparedness for the next possible vulnerability and the necessity of designing with security in mind are cyber-security objectives aiming to provide trust and assurance in all facets of technology. This is especially true when discussing hardware security, where a system is only as secure as the hardware embodying it.

The security of a printed circuit board (PCB) is one such example that is often overlooked, but if compromised can have drastic ramifications. Since PCBs serve as the heart of electrical systems, if it is compromised by an attack such as a hardware Trojan or low-quality counterfeit, the entire system could be compromised. In fact, as recently as October 2018, a report entitled “The Big Hack” [1] was released claiming that unauthorized computing components had been found inside several specialized servers of high profile entities such as Apple, Amazon, and the US Government. As reported in [2], these hardware Trojans are especially difficult to detect since they appeared as simple signal conditioning components whose impact would not be detected using runtime Trojan detection techniques. Security experts suspected that the assembly facility owned by Supermicro might have implanted the chip, which could serve as a backdoor for spying information exchanged over networks equipped with the altered PCBs of servers. The article also alluded to the possibility of embedding such components in internal layers of the PCB. This highlights the necessity of performing security assurance on physical systems to avoid the ramifications of such an attack.

One of the commonly practiced methods for assuring the security of hardware, especially from attacks such as this, is to perform full reverse engineering (RE) of the hardware in question. However, this is not practical with the current state of RE. With regards to PCB RE, the current state-of-the-art using X-ray Computed Tomography (CT) has addressed the biggest shortcoming of legacy PCB RE techniques – its destructive nature. Previously, the RE process consisted of delayering and imaging a PCB layer-by-layer, typically using a digital camera or optical microscope, until the sample no longer works [3]. In contrast, PCB RE via X-ray CT systems non-destructively reproduces a 3D point cluster of the PCB sample followed by reconstruction to produce a 3D volumetric model, consisting of 2D image “slices” of the scanned PCB [4]. An X-ray CT slice is a cross-sectional image of a specific region in the 3D volume that allows observing within the object.

Automation of the entire PCB RE process can be broken down into 5 major steps: image acquisition, image preprocessing,

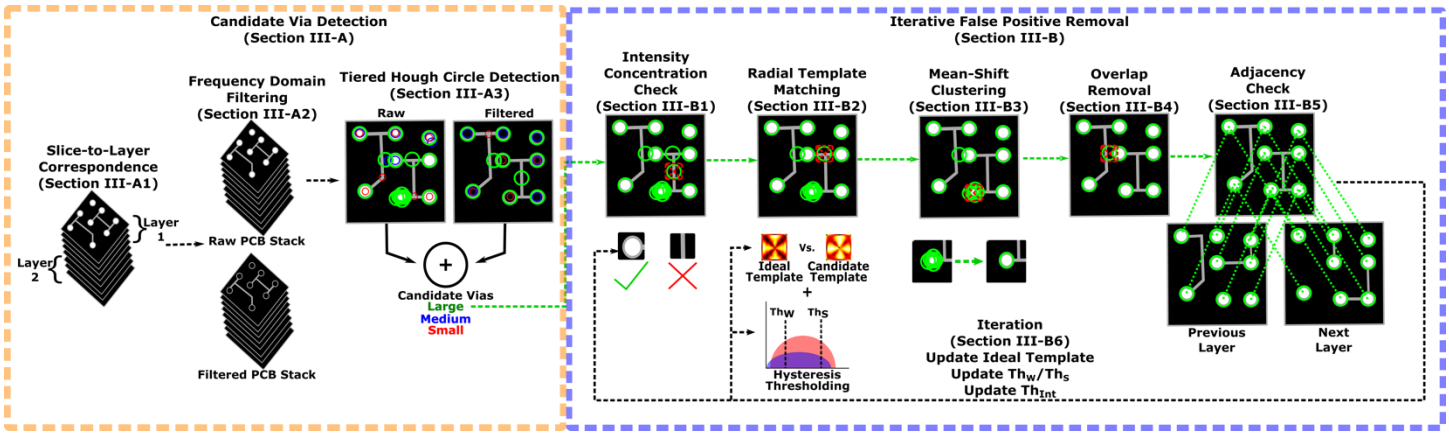


Figure 1: Automated Via Detection Framework: Candidate Via Detection & Iterative False-positive Removal

feature extraction, feature analysis, and evaluation [5]. At each stage, numerous challenges make automation of RE at the PCB level difficult. Leveraging advances in computer vision, machine learning, image analysis, and pattern recognition, we can facilitate the automation of each of these stages to make the PCB RE process faster, more accurate, and less reliant upon SMEs.

Earlier work by [6] done at the pre-processing stage of the framework addressing alignment and slice-to-layer correspondence facilitates our current contribution to the automated PCB RE framework, namely, automated via detection at the feature extraction stage. Initially, via detection may seem like a straightforward task using the Hough circle detection algorithm from image processing [7]. However, the main difficulty in via detection is the necessity to detect and localize a high quantity of densely packed vias accurately and in a manner that requires minimal human interference or algorithmic parameter tuning. The variability of vias across their sizes, board samples, and imaging conditions makes the development of a generalizable detection methodology especially difficult. Not to mention, the Hough circle transform, which is well studied in the literature [8]–[11], suffers from noise and false-positive detections. There have been multiple variants, and improved schemes proposed, but none are readily available or easily adaptable as the Hough circle transform [12]–[17]. To address the above challenges, we provide an unsupervised via detection algorithm that utilizes domain-specific knowledge of X-ray CT and PCB design to iteratively remove false-positive circles, initially detected by the Hough circle detection. In this paper, we elaborate on experiments conducted on a real-world 6-layer PCB and commercially available Xilinx Spartan 3 PCB, which demonstrates the effectiveness of our method.

The remainder of the paper is organized as follows. Section II will provide a brief background on the different algorithms implemented in this process. Section III discusses our methodology in-depth, whereas Section IV provides a qualitative and quantitative analysis of the results. Lastly, Section V concludes the paper.

II. Background Information

A. Hough Circle Transform

The generalized application of the Hough Transform is to estimate the presence of a specific object using a voting system. The entire process occurs in the parametric space, which uses values to describe a specific mathematical model. The first application of the transform was to identify lines, but specializations now allow circles, among other objects classes, to be recognized. The Hough Circle Transform selects viable circle candidates using the aforementioned voting method and selecting local maxima in an accumulator matrix. A circle with radius R and center (a, b) can be described with the parametric equations: $x = a + R\ cos\theta$ and $y = b + R\ Sin\theta$.
With θ ranging from 0 to 2π, a circle with the radius of R can be generated. If R is known, every "edge" pixel produces a circle with specific a and b values. Accumulator cells compile "votes" for each registered point. The highest-valued cell indicates the center position. If R is not known, a 3D parameter space is produced for all three parameters. Each possible R creates a 2D parameter space [7].

B. Hysteresis Thresholding

Contrary to Standard Thresholding, this method uses double thresholds to provide three separate classes of objects. The strong and weak groups are used as classified, but members of the "maybe" group are reassigned depending on existing conditions.

C. Structural Similarity Index

In order to measure the decrease in the quality of an image during processing, the Structural Similarity Index (SSIM) was created. Two visually-identical images from the same shot are compared using this metric. It deduces perceptual differences between images[1]. For two images of a and b with common size $N \times N$ the SSIM score is computed as follows:

[1] In this paper, a manually annotated PCB via image is compared to an automatically annotated PCB via data using SSIM.

$$SSIM(a,b) = \frac{(2\mu_a\mu_b + c_1)(2\,\sigma_{ab} + c_2)}{(\mu_a^2 + \mu_b^2 + c_1)(\sigma_a^2 + \sigma_b^2 + c_2)}$$

where μ_a and μ_b are the means of a and b, respectively. Both σ_a^2 and σ_b^2 represent their individual variances, while σ_{ab} indicates their covariance [18].

D. Mean-Shift Clustering

This clustering technique does not require foreknowledge of the number of clusters and does not restrict the shape of the clusters. For n given data points x_i, $1 \leq i \leq n$ on a d-dimensional space R^d the multivariate kernel density estimate obtained with kernel $K(x)$ and window radius h is

$$f(x) = \frac{1}{nh^d}\sum_{i=1}^{n} k\frac{(x - x_i)}{h}$$

The term

$$m_{h,g}(x) = \frac{\sum_{i=1}^{n} x_i g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)} - x$$

is the mean shift. The mean shift procedure, obtained by

- Computation of the mean shift vector $m_h(x^t)$, and
- Translation of the window $x^{t+1} = x^t + m_h(x^t)$

converge to a point, where the gradient density function is zero. The modes of the density function with points converging towards them define the basins of attraction and indicate distinct clusters [19].

E. Expectation Maximization

The Expectation Maximization (EM) algorithm is an iterative statistical algorithm that determines the maximum likelihood or maximum a posteriori (MAP) estimates of statistical parameters for a model or set of models. It involves an expectation step where the log-likelihood function is created and evaluated using a current estimate of model parameters, followed by a maximization step that computes the parameters that maximize the expectation of the log-likelihood function. These two steps are alternated in a repeating fashion until reaching convergence, where the values no longer change [20].

III. Methodology

Reverse engineering the location of vias in an X-ray CT imaged PCB is a task heavily reliant upon image analysis. Since, in a design, vias are typically presented in the shape of a circle, the detection and localization of these features can be achieved via the Hough transform circle detector. However, while the Hough circle detector is sufficient in most circle detection tasks, its known shortcomings are the computational complexity and propensity to produce false-positive detections. A high number of false-positives are especially hindering for reverse engineering since the goal of the process is to reproduce a design of the highest fidelity and accuracy to ensure performance or reliability when physically reproduced. For these reasons, we propose leveraging Hough circle detection for initial via detection, followed by a series of stages incorporating domain knowledge to facilitate the unsupervised iterative removal of false-positive detections.

Figure 1 provides an illustrative example of our proposed methodology, beginning with a candidate via detection. Candidate via detection algorithm first cross-matches slices and layers in a board (slice-to-layer correspondence). Afterward, frequency-domain filtering is applied to the slices to remove noise for the optimal candidate via by utilizing the Hough circle detection algorithm for various radii ranges and all the slices belonging to the discrete layers. Next iterative false-positive removal for each tier focuses on removing candidate vias whose pixel intensities along its circumference falls below a learned threshold unique to x-rayed copper rings of vias. Followed by screening remaining candidate vias based on their similarity to the iteratively learned radial-gradient behavior expected of an x-ray imaged via unique to the board, size, and data type(raw/filtered). The remaining stages leverage PCB design constraints such as consolidating a cluster of candidate vias to their most likely center, removing any candidates that violate spacing design constraints, and verifying the minimum requirement of adjacent via connectivity. This series of false-positive evaluations is used to better inform the evaluation process in subsequent iterations until convergence, where no more candidates are detected nor removed. Lastly, our proposed methodology concludes with evaluating the concentricity of detected vias across tiers following iterative false-positive removal to ensure the strongest via is returned as the final detection. While this serves as a high-level description of our proposed methodology, the below sections provide a more in-depth explanation.

A. Candidate Via Detection

The first step for via detection is the detection of circles in general. Typically, the popular Hough circle detector algorithm would be implemented with little modification; however, there are specific challenges associated with this task that require a more nuanced approach, as described in the sections below.

1) Slice-to-Layer Correspondence: The most important features in a PCB are the vias and traces, especially vias providing the electrical connections between layers in a PCB. As shown in Fig. 2, 3 types of vias used in a PCB [21]:

1) Through-hole vias connect the exterior layers of the PCB to one another.
2) Blind vias connect an exterior layer to an internal layer.
3) Buried vias connect two internal layers together.

To properly reverse engineer a design, especially automated, it is important to be aware of and detect these types of vias. Vias on the exterior layers are visible when analyzing a multilayer PCB visually or optically, but knowing which category they belong to is impossible without analyzing the internal layers. With the aid of X-ray CT, we are able to analyze these internal layers non-destructively but not without introducing challenges due to the imaging modality.

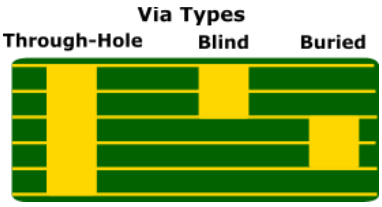


Figure 2: Types of Via in a multi-layer PCB: 1) Through-Hole: Surface to surface 2) Blind: Surface to internal layer 3) Buried: Internal layer to internal layer

In particular, the data produced from X-ray CT is a 3-dimensional stack of 2-dimensional images, referred to as slices, used to generate the 3D model of the PCB. Each slice of the PCB stack contains partial information, and the fusion of slices provides a reproduced image of a physical layer. To know what layer a via corresponds to, it is important to know what slices correspond to a respective layer. The study in [6] provides an unsupervised algorithm that achieves accurate slice-to-layer correspondence containing the respective slices per layer. Furthermore, the slices grouped in a particular layer can be fused to form the typical layer image seen in CAD software or when inspecting a PCB physically/optically. These slice-to-layer groupings and combined layer images are utilized throughout to facilitate accurate physical localization of detection results.

2) Pre-Processing: Another challenge introduced by X-ray CT is the noise artifacts. Most noise sources are handled at the image acquisition stage of RE, but two especially impactful to the quality of RE that can be controlled (to some extent) through image processing are blur artifacts and noise from high-impedance materials [5]. Blur artifacts are the result of a slight tilt of the PCB sample's orientation inside the X-ray imaging chamber. This slight tilt is done to maximize the amount of information received and reconstructed during the imaging process. Still, it can also sometimes introduce these blurred image regions as an unintended byproduct. Noise from high-impedance materials constitutes artifacts caused by the inability of X-ray particles to fully-pass through the object being imaged. In the case of PCB RE, solder and electrical components act as high-impedance materials and can drastically affect the quality of imaging as well as fidelity of features like vias or traces.

We utilize a frequency-based filter to address these two noise sources, as seen in Fig. 3, before implementing circle detection and subsequent false-positive removal. We utilize a 34th order 2D filter with magnitude responses of [0, 0, 5, 10] at each of the following normalized frequencies [0, 0.075, 0.3, 1], where 1 is the Nyquist frequency. This filter sharpens the image to remove the effects of blurring and accentuates the edges throughout the image to reduce smearing impact, due to high-impedance materials. While this approach does not remove the noise sources completely, it dramatically reduces their effect on the images and maintains the fidelity of vias for initial circle detection and subsequent false-positive removal.

For these reasons, we apply this filter to every slice in the original 3-D PCB stack and subsequent fused layer images. This creates two sets of the same X-ray CT imaged PCB stack: (1) a filtered version of the data with minimized noise artifacts while maintaining high fidelity features, and (2) the original non-filtered slices and fused-layer images, also referred to as the raw dataset. Later stages leverage both sets of data in a symbiotic manner.

3) Tiered Hough Circle Detection: With the knowledge of slice-to-layer correspondence, and the noise being suppressed, the next step figures out the collection of candidate vias using the Hough circle detection algorithm. Due to the shape of vias, primarily being circular, the Hough circle detection algorithm is well-suited for initial RE of vias from X-ray CT images due to its ability to detect and localize. However, there are some critical insights needed before blindly applying this algorithm for this application.

As mentioned in Section II-A, two key parameters of the Hough detection algorithm are the radius of circles to be searched for and the accumulator threshold. The accumulator threshold is responsible for the quantity and quality of candidate circles the algorithm returns. The higher the accumulator value, the more likely the candidate circle is indeed a genuine circle and not a false-positive. However, a nuance to this particular imaging modality and application is the quality of circles that are returned are not always genuine vias, as highlighted in Fig. 4. A typical PCB design has 100s of vias per layer, and PCB design itself can result in many unintended circular pixel patterns due to the sparseness of the boards aside from vias and traces. Due to the voting nature of the accumulator and the Hough detectors -known to be susceptibility to noise, the strongest circles detected may not, in fact, be the correct vias we look for. For this reason, instead, it is important to maximize our likelihood of detecting genuine vias and screen out the false-positives in later stages using the knowledge we have of the problem space.

Specifically, we apply circle detection with a low enough accumulator threshold, which ensures that all valid vias are

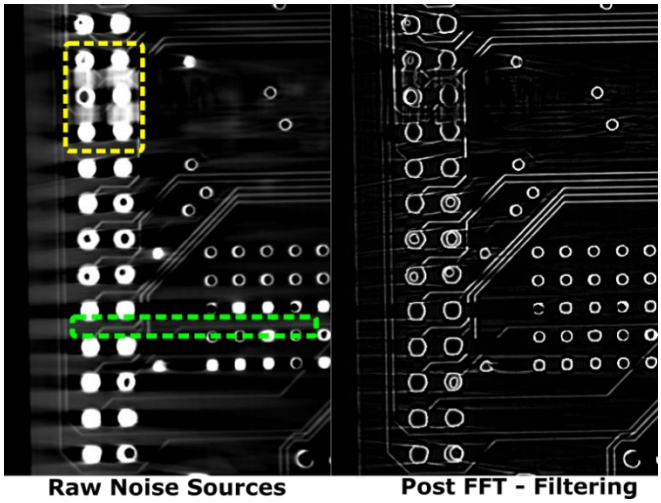


Figure 3: Noise in X-Ray CT PCB:(green square) Blur Artifacts (yellow square) High Impedance Material Artifacts, and post filtering result

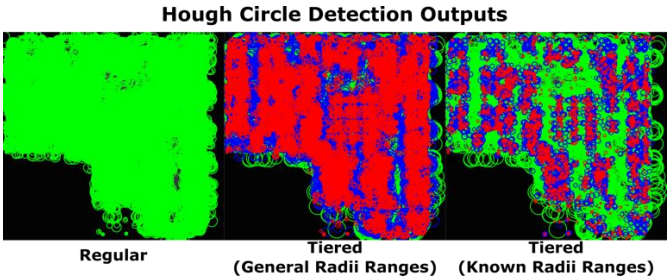


Figure 4: Comparison of outputs/candidate vias for different implementations of hough circle detection on Layer 4 of Test Board (R-Small Tier, B-Medium Tier, G-Large Tier)

detected, albeit with varying false-positive levels for the sets of filtered and raw slices screened at later stages. The idea behind this is that a via can be detected in one set of raw or filtered data due to the fidelity of via features, although it varies at each slice with the amount of noise or imaging quality. All images presented are based upon experiments with an accumulator threshold of 15.

Additionally, we employ a tiered sizing for circle detection based on the radius range. In RE, one example of the varying amounts of a priori knowledge one may have is the range of radii to expect in a design. Therefore, we explore two scenarios in this paper as well: (1) the reverse engineer practitioner knows specific radii range or (2) simply an upper and lower bound of sizes. For our experiments, we employed a three-tier model (small, medium, and large) and a four-tier model (small, medium, medium 2, and large) to highlight how the operator can specify the number of tiers.

For the assumption that the reverse engineer does not have specific range information per sizes, we assume a lower bound of 2 pixels since that is the smallest reasonable size a via may be in a given image, however very unlikely. While an operator may not have the range for each size of via it is not unreasonable or difficult to determine an upper bound on via size either from quick image analysis or physical inspection. However, suppose the reverse engineer gets this information from physical measurements. In that case, it is important to remember to translate to the pixel level resolution since the radii are measured in terms of pixels in the algorithm. The range of each size is simply the range between maximum and minimum radii divided by the total number of tiered detectors in use. Furthermore, where one size range ends, the next begins one pixel more.

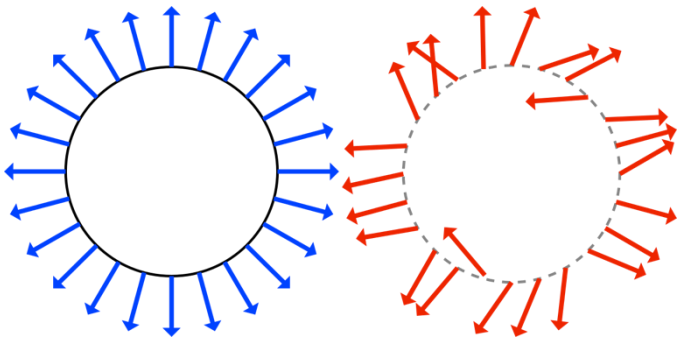
Note that even knowing the radii range and setting a high accumulator threshold may not help provide the correct vias without taking subsequent processing steps. These steps are necessary because of the noise, feature fidelity, and the nature of the imaging modality as X-ray CT imaged PCBs can contain many unintended circular pixel patterns, when being analyzed without context. Lastly, for each tiered detector size and each slice in a layer, the circles detected in both the raw and filtered versions are concatenated to form one large list of candidate vias. That list is concatenated with other candidate vias of slices from the same layer grouping (Section III-A1). To wrap up, each tiered circle detector could have produced a list of candidate vias for every layer to be screened for false-positives afterward.

B. Iterative False-positive Removal

The next step in the via detection process is to iteratively remove false-positive circle detections by leveraging various properties unique to vias imaged through X-ray CT. This process is designed to be able to learn unsupervised and generalize to each board uniquely since there is a large amount of variety from not only board to board but via size to via size in a single design. Therefore, this iterative process learns the unique properties of a given dataset while applying properties consistent across PCB designs based on known design constraints. The following is applied individually to each tiered circle detector's set of candidate vias per layer.

1) Intensity Concentration Check: As noted earlier, X-ray CT imaging results in a grayscale image. With vias being constructed from a high-impedance material, albeit not enough to impose noise artifacts, this results in a white ring in the image, where the copper conducting ring is located in the physical design. Furthermore, if the via has had a component depopulated, the via will likely contain even more high pixel intensities due to the residual high-impedance solder. Hence, a likely-detected via (i.e., true positive) should have a moderate to a high concentration of high grayscale pixel intensities in its frame of interest along with the copper conducting ring. Vias detected falsely (i.e., False-positive) should often have far lower pixel intensities in the ring of the candidate frame. Ergo, these false-positives can be removed by computing the concentration of high-intensity pixels along the circumference in a candidate frame and removing those that fall below a given threshold.

The threshold mentioned above, Th_{Int}, is computed by leveraging the output of the tiered circle detector (Section III-A3) and the raw/filtered fused layer images (Sections III-A1/III-A2). The output of the circle detection stage is the radius and center (x, y) of every candidate via for each of the tiered sizes (small, medium, and large). We use that circle information to localize the candidate via in its raw and filtered layer image and create a frame of the candidate via from one another. The candidate via frame has a length and width equal to the diameter of the candidate via and encapsulates only the via in question from said layer images as tightly as possible. For every candidate via in a layer, we take the intersection (i.e., pixel by pixel logical AND operation) of the region of interest with an ideal ring of white pixels the same radius as the candidate circle, followed by computing the percentage of non-zero pixels. Next, we compute the mean non-zero pixel intensity concentration for all candidate frames for both raw and filtered versions of the candidate frames to produce two separate thresholds. These thresholds are unique to each layer and bounded by the minimum possible intensity concentration for the largest possible radius of the tiered radii range. If a candidate via fails both the tests with the raw and filtered intensity concentration thresholds, it is likely a false-positive and does not make it to the next stage of screening.



(a) Ideal Radial Gradient Vectors (b) Image Gradient Vectors
Figure 5: Heat maps (shown in Figure 6) are generated by taking the dot product of the image gradient directions and an ideal radial gradient image.

2) Radial Template Matching: The next step in the false-positive removal process is utilizing the prototypical gradient radial behavior of a via in an X-ray CT imaged PCB. A typical circle in an image exhibits an outwards radial behavior when analyzing the gradient direction of the intensities in the image. A via also exhibits this behavior but in varying degrees depending on the amount of noise in the image and whether the via is filled with solder or empty.

We compute the radial behavior of remaining candidate vias after the intensity concentration check stage (Section III-B1) by computing the image gradient of each candidate via and normalize the gradient direction to unit vectors. We then compute the ideal radial behavior of a circle by computing the unit vector of each pixel location in a frame the same size as the candidate via's frame.

$$M_{x,y} = \left| \left[\frac{\nabla(\text{ideal})}{|\nabla(ideal)|} \right]_{x,y} \cdot \left[\frac{\nabla(detected)}{|\nabla(detected)|} \right]_{x,y} \right|$$

As shown in the above equation, for each pixel location, we take the dot product between the ideal radial behavior (Fig. 5a) and the computed radial behavior of the candidate via (Fig. 5b), giving a measure of "radialness" bounded by $[-1, 1]$, where "1" represents two vectors in the same direction, and -1 represents vectors in opposite directions. We then take the absolute value of this dot product (therefore $M_{x,y} \in [0, 1]$), because we are only interested in measuring whether the gradients are in the same radial orientation. The result of this dot product is best represented as a heat map, where a circle in an image exhibiting perfect radial behavior would be a circular image of the highest intensity.

When analyzing 100's of candidate vias radial-behavior and comparing that with an ideal case, a heat map image with a clear cross pattern in the image can be seen, as in the base template of Fig. 6. Figure 6 also highlights the varying degrees of radial behavior, whether the via in question is of high quality, low quality, or a noisy mix of the two. While computing the gradient direction is dependent on the image quality and type, the resultant radial behavior heat map can be viewed as a transform agnostic across imaging specific properties.

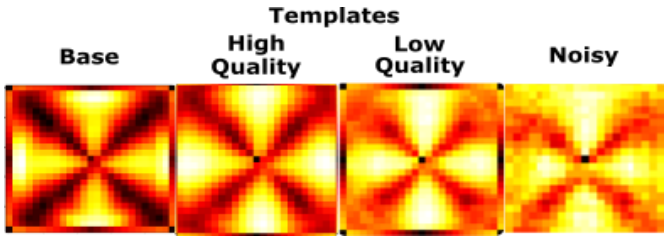


Figure 6: Radial Behavior Templates composed from high quality, low quality, and noisy detected vias.

Although this is the general shape to be expected, closer analysis shows certain via sizes exhibit radial behavior unique to size and board. Therefore, by using a general template and computing the similarity of the candidate via's radial behavior to that template, false-positive vias can be greatly reduced. This uniqueness of radial behavior can be extended further to vias from raw data versus filtered data as well. Both maintain the cross-pattern behavior but to varying degrees, similar to sizes. Thus, false-positive vias can be screened by using a template unique to the size and data type, filtered vs. raw, of the candidate via. To effectively screen out false-positive vias, we employ a hysteresis threshold-based approach.

a) Hysteresis Thresholding: We begin by computing the SSIM score of each candidate via's radial behavior against the template radial behavior of that respective size and data type. SSIM is used since we are interested in those candidate vias whose radial behavior is most similar to the template radial behavior structurally, as opposed to similarity at a pixel level resolution. Afterwards, we now employ hysteresis thresholding in combination with the Expectation Maximization (EM) algorithm (Section II-E [20]) where we calculate 2 thresholds, referred to as Th_w and Th_s, to create 3 classes/distributions (weak, maybe, and strong). These 3 distributions represent our confidence whether the candidate via is genuine or a false-positive.

First, we separate the candidate vias into two distributions, weak and strong, based on computed SSIM scores. Empirically, it was determined that a good baseline threshold for this task is a value of 0.41 for each size and data type. SSIM scores above this threshold are placed into the strong distribution and those below in the weak. The means of these distributions are used to initialize the EM algorithm, which iteratively determines the maximum likelihood of each candidate via's SSIM score belonging to either the weak or strong distribution. Upon convergence of the EM algorithm, the SSIM scores are then classified to each distribution, and the statistics of each distribution are used to compute Th_w and Th_s. The means of each serve as the starting point for the thresholds and are uniquely scaled between 1 and 2 standard deviations according to the separability of each layer's distributions, as measured by the d' value [22]. A minimum value of 0.3 for each threshold is in place to prevent instances where the thresholds may vanish, and all candidates pass hysteresis thresholding. Thus, the thresholds are computed as follows:

$$d' = \frac{\mu_s - \mu_w}{\sqrt{\frac{1}{2}(\sigma_s^2 + \sigma_w^2)}}$$

$$Th_w = \mu_w + \left(1 + \frac{d'}{10}\right)\sigma_w, Th_w \geq 0.3$$

$$Th_s = \mu_s - \left(2 - \frac{d'}{10}\right)\sigma_s, Th_s \geq 0.3$$

If the SSIM score assigned to a candidate via is above Th_s we insert it into the strong class, and if above Th_w but below Th_s, it is inserted into the maybe distribution. Otherwise, if it is below Th_w, we do not consider that a via that has the chance of being genuine and insert it into the weak distribution. Hysteresis thresholding for edge detection utilizes double thresholds in a similar fashion, where edge pixels that pass the strong threshold are considered strong, those that do not pass the weak threshold are ignored, and those above weak, but below strong, are considered weak edges. For hysteresis-based edge detection, weak edges connected to a strong edge are moved to the strong category, and all others remain weak and suppressed. We utilize a similar approach for our hysteresis-based thresholding for via false-positive screening.

Specifically, we apply this double threshold approach to both raw (r) and filtered (f) versions of data resulting in 4 thresholds (Th_{w-r}, Th_{s-r}, Th_{w-f}, Th_{s-f}) and 6 distributions (raw weak, raw maybe, raw strong, filtered weak, filtered maybe, and filtered strong). Any candidate vias, whose radial SSIM scores fall below $Th_{w-r/f}$ are placed into the weak distribution for both filtered and raw data, respectively, and ignored. Additionally, any candidate vias that fall into the maybe distribution for both raw and filtered data are also considered likely false-positives and thus also moved to the respective weak distributions. Strong vias are those that are above Th_s for at least raw or filtered data and above Th_w for both filtered and raw data. Vias that pass this stage are considered likely to be true-positives according to their gradient behavior and are moved to the filtered and raw strong distributions. The candidate vias that finish in the strong distribution are those that have passed radial template matching with hysteresis thresholding.

However, the success of this screening procedure is not guaranteed, and thus, later stages serve to leverage domain-specific properties of vias to effectively remove any false-positives that may have inadvertently passed radial behavior screening. It should be noted that the original baseline threshold of 0.41 is only used in the first iteration to initialize thresholds. Subsequent iterations, discussed more later in Section III-B6, follow this same process but with updated thresholds used to separate scores into weak and strong distributions whose means initialize the EM algorithm for computing the next iteration's thresholds.

3) Mean-Shift Clustering: Now we can employ mean-shift clustering [19], discussed earlier in Section II-D, to condense the high-density groupings of remaining candidate vias that share similar centers and radii. Where each resultant cluster center is a proposed candidate via's true center and radius. The earlier discussed slice-to-layer correspondence property could be further utilized since we know that while slices contain varying degrees of information, their fusion forms a consistent layer image. Thus, while different slices of a layer may produce different detected circles, the produced centers and radii should be relatively consistent for true positive vias throughout all slices. The only hyper-parameter necessary for mean-shift clustering is the kernel bandwidth. The kernel bandwidth parameter can be optimized depending on the aspect ratio of the imaging scan in relation to the size of the board and the density of features. If the kernel bandwidth is too large, it is easy for the clustering to merge adjacent candidate vias incorrectly, especially if they are very densely packed together. If the kernel bandwidth is too low, then all of the candidate vias will serve as their own clusters. Based on empirical evidence, we have determined a kernel bandwidth of 10, sufficient for PCBs with sparse and densely populated with vias.

4) Overlap Removal: With the dense groupings of proposed candidate vias now condensed to singular candidate vias, we leverage PCB design rules to remove any candidate vias that overlap with one another. Specifically, it is common practice in PCB design that no via should be designed within 10mil(254μm) of one another to minimize interference [23]. Therefore, for any pair of candidate vias violating this design rule, one is likely a false-positive and the other genuine. However, this minimum spacing is physical measurement and needs to be related in terms of pixels. With prior knowledge of the physical dimensions of the board, the distance in terms of pixels can be computed in the following manner:

$$minDist_{pixels} = \frac{designRule.\sqrt{H_I^2 + W_I^2}}{\sqrt{H_P^2 + W_P^2}}$$

where H is the height, and W is the width of the respective image (I) or physical board (P). Figure 7 provides a toy example of these violations. Now we localize and evaluate each candidate via's circumference in relation to all other candidate

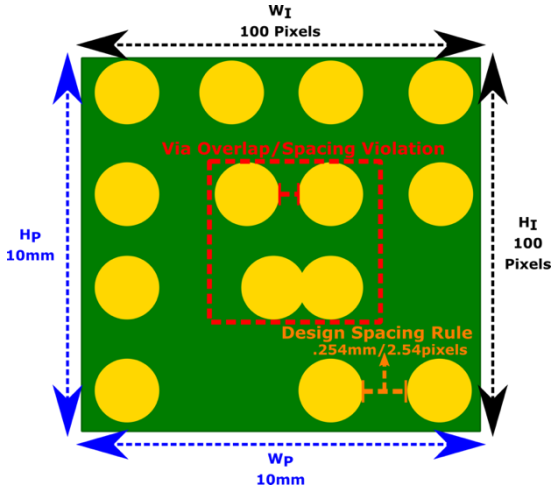


Figure 7: Overlap/Spacing Rule Example highlighting conversion of physical and pixel domain dimensions and PCB design violations

vias of the same layer to confirm that no vias are within the computed minimum distance. If candidate vias overlap or are within that minimum distance, the candidate via whose raw radial SSIM score, as computed in Section III-B2, is higher remains as the valid candidate via.

5) Adjacency Check: Due to the possibility of vias being buried or blind and the likelihood that in practice, the reverse engineer not knowing that information a priori, it is important to be able to detect as accurately as possible at a layer level resolution. Thus, once all earlier false-positive screening steps have been completed for each layer of a tiered size, we begin the analysis of each layer in relation to one another. While blind or buried vias may not be present at every layer in a board, they are at least present on adjacent layers in order to maintain connectivity in the design. Therefore, to facilitate accurate detection without making too many assumptions, we assume that each layer is at most connected to either the layer above or below itself (buried via behavior).

Assuming adjacent layer connectivity, we concatenate the remaining candidate vias of the layer under test with those directly above and below itself. Once again, we apply mean-shift clustering to these candidate vias to consolidate the groupings of vias that share similar centers and radii to singular candidate vias. Next, we utilize radial template matching and hysteresis thresholding once again. Any radial SSIM scores of genuine candidate vias present on the currently evaluated layer and adjacent layers will be much greater than any remaining false-positives due to the via in question likely not being present in the adjacent layer images. Thus, we compute an adjacency score (AS) by taking the average radial SSIM scores across the adjacent layers and compare to the average hysteresis thresholds ($Th_{w/s-a}$), calculated earlier in stage (Section III-B2), across the same layers. These thresholds and score are calculated, for both raw and filtered data, as follows:

$$AS(V,L) = \frac{\sum_{l=L-1}^{L+1} SSIM_{radial}(V,l)}{l_a}$$

$$Th_{w-a}(L) = \frac{\sum_{l=L-1}^{L+1} Th_w(l)}{l_a}$$

$$Th_{s-a}(L) = \frac{\sum_{l=L-1}^{L+1} Th_s(l)}{l_a}$$

where L is the current layer under evaluation, V is the current candidate via, and l_a is the number of adjacent layers, including the layer under evaluation. Only candidate vias with AS above Th_{s-a} for at least raw or filtered data and above Th_{w-a} for both filtered and raw data are considered valid. Lastly, remaining candidate vias undergo overlap removal in the same manner as Section III-B4. The remaining vias are the predicted genuine vias for each layer and tiered size detector.

6) Iteration: Each previous stage aims to leverage properties of X-ray CT imaging and PCB design to screen out as many false-positive vias as possible from the original set of detected circles. The first stage of false-positive removal, intensity concentration checking (Section III-B1), serves to remove the most blatant false-positives. After a false-positive removal iteration, the average intensity concentration for the predicted genuine vias can be computed and used as a more informed threshold for the intensity concentration check of the next iteration, so long as the average intensity is not more significant than the ideal intensity of the tiered detector's largest radius.

Next, radial template matching (Section III-B2) is responsible for screening out the vast majority of candidate circles based on a candidate via's gradient radial behavior compared to the template of ideal radial behavior of an X-ray CT imaged via. However, while the radial template is the ideal behavior of an X-ray CT imaged via the actual behavior, it can depend on the size, amount of noise, and imaging quality of the vias/board sample being evaluated. Thus, during initial radial template matching, there may have been many false-negative screenings that never made it to subsequent screening stages and evaluation based on the initial template.

Consequently, we propose an iterative approach to better learn the unique radial template behavior of a via an unsupervised fashion. After the false-positive screening, the new template for a tiered size is the average of all gradient radial behavior of the remaining predicted genuine vias for all layers of that tiered size. This is done for raw and filtered gradient radial behaviors separately to produce two new templates that are both used in the next iteration of the false-positive removal pipeline. Most notably, in radial template matching, overlap removal, and adjacency checking, SSIM comparison between computed radial behavior and template radial behavior depends on template quality.

Furthermore, hysteresis thresholding is affected by iteration as well. As the template improves over iterations, SSIM scores of previous false negative candidate vias should not only improve and pass subsequent screening, but false-positive scores should decrease as well. This is due to the updated radial templates providing a better characterization of vias unique to tiered size, data type, and board. Thus, changing the composition/statistics of weak, maybe, and strong distributions used in hysteresis thresholding and EM each iteration. These updated thresholds are additionally used during adjacency checking each iteration as well.

It is important to remember that each iteration of false-positive removal is performed on the same set of candidate vias initially returned from the tiered circle detectors. Hence, no new circles are detected per iteration, but any increase in the number of vias after the false-positive removal is due to changes in the learned template from iteration to iteration. A round of iterative false-positive removal performed by the detector is considered completed once the number of genuine vias detected matches that of the previous iteration. Once there is no change in the detected vias from iteration to iteration, the template will not change, meaning that no changes will be seen at any other stage of false-positive removal, thus signifying convergence.

C. Concentricity Check

Upon completing the iterative false-positive removal process for every candidate via, they are passed to the final screening step, concentricity checking. Concentricity is defined as two

circles that share the same center. If a via was detected by multiple tiers, it would have the same center with different radii. This stage simply consists of combining the vias from every tier and performing overlap removal similar to previous steps. The only caveat for this stage is the radial template used to evaluate correctness is the raw template respective of the detected via's tiered detector. Afterward, this assures that the most correct via is detected if two tiers detected vias at the same center location but different radii size.

Before translating these pixel-level results into physically manufacturable designs, the final step is taken by the reverse engineer operator to validate and remove any clear remaining false-positives. Although the operator still performs a manual evaluation, this framework still provides significant savings in time and manpower for the reverse engineering of design, especially a design densely populated with vias.

IV. Results & Discussion

A. Experimental Setup

We performed our experiments on two sets of data. One was a 6-layer PCB, referred to as Test Board, with physical dimensions of 15mm by 15mm and 3-D stack dimensions of 1357 x 1286 x 157 slices. The other dataset is a commercially available 6-layer Xilinx Spartan 3 PCB with estimated dimensions 280mm by 210mm and 3-D stack dimensions of 1871 x 1858 x 58 slices. Additionally, due to manual depopulation resulting in large amounts of noise artifacts, only 3 layers of the Spartan board have features of high enough fidelity for circle detection and subsequent via RE.

Important hyper-parameters of our experiments were utilizing 3 tiers of circle detectors for our Test Board. Additionally, to highlight the flexibility of our approach, we present our results employing 4 tiers for detection on the Spartan 3 PCB but experiments with 3 Tiers had results comparable within 1%. For the experiments of our Test Board, we experimented across accumulator thresholds of 10, 15, 20, and 25. For the Spartan board, we experimented with an accumulator threshold of 15 only since anything higher did not provide valid detected circles where genuine vias are located, and anything lower became unnecessarily computationally intensive due to the density of vias on the board and the volume of the 3-D stack of slices.

Additionally, two separate radius sizing scenarios have been considered: (1) the reverse engineer operator only has a minimum and maximum radius of the entire design, and (2) a known range for via radii, for each tiered size, is known by the RE operator to choose from. For the assumption that the reverse engineer has previous radii sizing information, we use ranges of 2-10 pixels for the small detector (Test/Spartan), 11- 18 pixels for the medium detector (Test/Spartan), 19-22 pixels for the medium 2 tier detector of the Spartan data only, and 30-40/44-55 pixels for the large detector of the Spartan and Test Boards respectively. In the case that no prior information is known besides a maximum and minimum possible via size, we utilize a minimum of 2 pixels for both and a maximum of 40/55 pixels for the Spartan and Test Boards, respectively. These sizes are specific to the Test Board and Spartan dataset and experiments. The minimum distance between adjacent vias is 10mil (254μm) for overlap removal.

Lastly, we use a kernel of bandwidth 10 for all mean-shift clustering. Moreover, for both datasets, we employ the same initial radial template for both raw and filtered data evaluation in the first iteration, which has been computed by averaging over hundreds of gradient radial behavior heat-maps for vias of all sizes and data types (Base Template Fig. 5).

Deep Learning Based Via Detection: Masked-RCNN: A suitable means of measuring the potency of the proposed method described is to compare it to the current state of the art for deep learning-based object detection. Since our goal is not only the accurate detection, but also localization of a via, the Mask R-CNN [24] is selected. It detects objects in an image while simultaneously generating a high-quality localized segmentation mask for each instance. The object mask output requires extracting the finer spatial layout of an object. The finer details of the Mask R-CNN involve a Region Proposal Network (RPN), which proposes candidate object bounding boxes and the feature extraction stage, which uses Region of Interest Pooling (RoIPool) from each candidate box and performs classification and bounding-box regression. A mask encodes an input object's spatial layout. Thus, unlike class labels or box offsets that are inevitably collapsed into short output vectors by fully-connected layers, extracting the spatial structure of masks can be addressed naturally by the pixel-to-pixel correspondence provided by convolutions.

The off-the-shelf implementation of Mask R-CNN used provides weights pre-trained on the MS COCO Dataset, which consists of 80 different classes, including suitcase, cat, dog and baseball bat. The code available on Github is unaltered, and the pre-trained weights are fine-tuned to locate and identify vias on two sets of PCB tomographies. Our intention is to make the vias an additional class. The dataset for fine-tuning consists of 192x192 sized images with a single via of no specific size contained in each image. The VGG Image Annotator (VIA) is the annotation tool used to define the vias in each training image. For a circle, the parameters provided are the center coordinates and radius, thus for each image, an empirically determined minimum of 10 circle masks are used. This is to provide more annotation points for training, especially since circle annotation points are just their centers. Only 100 via images are used for the final training/tuning process.

To supplement the existing Masked R-CNN infrastructure, we extend the Config class to modify the training parameters, and the Dataset class to access training images, test images, and annotations for the vias. The annotations used on the COCO dataset are polygons, but we use circles; hence we modify the dataset class to retrieve the circle features provided. We change the minimum confidence value from 0.9 to 0.8 to allow possibly-detected vias with lower confidence values. Although the maximum number of instances per image value remains at 100 during training, we increase it to 1000 for the Spartan Board during the inference step. This is to accommodate the via-count of the board that is above 800. The minimum and maximum

Table I: Segmentation Mask Evaluation Results: Known Vs. General Radii Vs. Mask R-CNN (T.B. = Test Board, S3 = Spartan 3)

Segmentation Mask Evaluation									
Layer	IoU			DICE			SSIM		
	Known	General	Mask R-CNN	Known	General	Mask R-CNN	Known	General	Mask R-CNN
T.B.-0	.8623	.8521	.4872	.9209	.9140	.4935	.9732	.9716	0
T.B.-1	.8860	.8722	.5816	.9362	.9274	.6533	.9734	.9715	.9567
T.B.-2	.8832	.8407	.5831	.9345	.9065	.6550	.9726	.9691	.9548
T.B.-3	.8800	.8495	.6591	.9325	.9124	.7484	.9724	.9698	.9597
T.B.-4	.8863	.8579	.5835	.9364	.9182	.6561	.9735	.9695	.9569
T.B.-5	.8620	.8523	.4875	.9206	.8955	.4937	.9728	.9674	0
S3-0	.7190	.6947	.5062	.8108	.7891	.5865	.9477	.9410	.8694
S3-1	.7361	.7052	.5352	.8261	.7992	.5753	.9487	.9403	.8912
S3-2	.7342	.6976	.4903	.8244	.7921	.538	.9498	.9411	.7815

image dimensions are also changed from 800 and 1024 to 192 and 192 to suit our prepared dataset. Parameters such as the number of epochs and batch size remain at 30 and 8, respectively. We change the original bounding box to a circle. This is to allow seamless comparison with our method.

B. Accuracy

The most critical evaluation metric, especially for reverse engineering, is how accurate were predicted genuine vias compared to the ground truth. Excessive false positive (incorrectly detected) and false negative (incorrectly missed detections) via detections could have drastic impacts on a physically reproduced PCB therefore our iterative false positive removal stages are critical for ensuring accuracy. For our evaluation, we utilized image annotation software [25], [26] to hand label ground truth vias for every layer of the Spartan and Test Boards, resulting in a list of ground truth via centers and radii for each layer in the board designs.

Figure 8 is provided to show the scale at which false positives/negatives are initially returned from tiered hough circle detection (Section III-A3) on average for all layers of the Test Board utilizing known radii ranges and the degree to which subsequent false positive removal stages are successful. We can see that a semi-log plot is used to display the logarithmic decrease in the quantity of false positive/negative detections across subsequent stages. Most notably, between the intensity concentration check (Section III-B1) and radial template matching (Section III-B2) stages. The intensity concentration check serves as the initial screening stage to lighten the computational burden by removing the simplest and most obvious false positives. Followed by radial template matching's more intricate screening nature. Additionally, highlighting how radial template matching's stricter screening of false positives comes at the cost of more false negatives. However, subsequent stages serve to fine tune any missed false positive or false negative detections based on known PCB design constraints such as removing false positive overlapping vias (Section III-B4) and detecting missed false negatives based on via connectivity of adjacent layers (Section III-B5). Lastly, what can be learned from the high degree of overlap occurring across iterations will be discussed later in Section IV-C.

Furthermore, during the evaluation, we are not only concerned about classification but localization accuracy as well. Therefore, not only a via was detected when it should be, but also its size and location were determined correctly. Thus, a good evaluation method that takes both into account is to treat this entire problem as a segmentation challenge where a correct result will produce a segmented output for each layer as close to the segmented ground truth as possible. This serves as a global evaluation of the final predicted vias in relation to all the other vias detected in the same layer.

To effectively and accurately compare segmentation results, we take the centers and radii of the predicted genuine and ground truth vias and create a binary mask of only the vias for each. We then perform a logical AND operating between each of the respective binary masks and the raw layer images to produce segmented grayscale images containing only via information. These images are then evaluated based on common segmentation, and image quality assessment metrics: mean intersection-over-union (IoU) [27], mean DICE coefficient [28], and SSIM [18] between the ground truth segmented result and predicted segmented results. It is important to note that we do not use the popular evaluation metric pixel accuracy due to the misleading nature of the results in the context of this problem space. Specifically, pixel accuracy will always be high since the foreground features we are searching for are sparse compared to the vast background pixels that will dominate the metric calculation.

IoU measures the overlap between the ground truth mask and our predicted mask for each layer divided by the union of the two images. The formula for IoU is

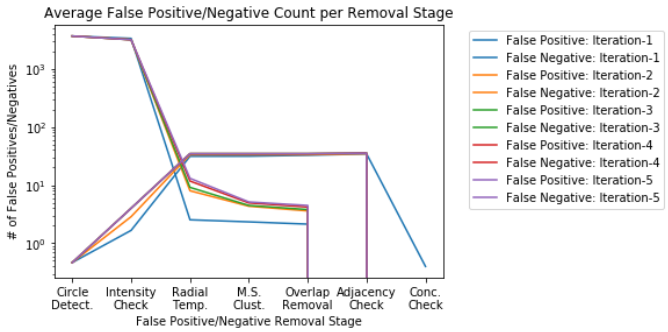


Figure 8: Semi-log plot displaying the average quantity of false positive and false negative detections present for each layer and size tier after each stage of the iterative false positive removal pipeline for each iteration until convergence.

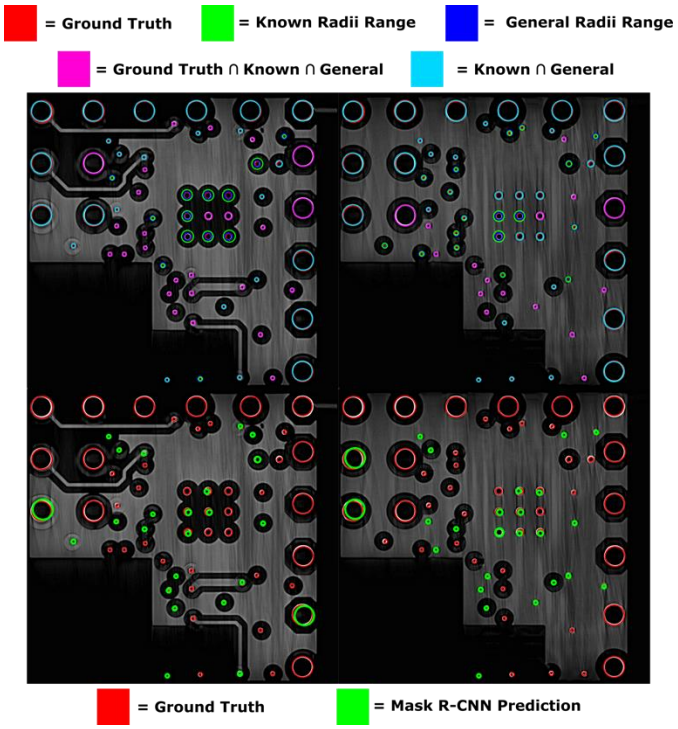


Figure 9: Via Detection Performance Comparison: Proposed Methodology (Top) Vs. Mask R-CNN(Bottom) on Test Board Layers 2(Left) and 4(Right).

$$IoU = \frac{M_p \cap M_{gt}}{M_p \cup M_{gt}}$$

where M is the produced mask of the predicted (with subscript p) or ground truth (with subscript gt) set of vias per layer. Since this is a binary segmentation task, the IoU is computed for both the foreground and background separately and then averaged to provide a mean IoU score for evaluation. Since the background is the dominant class of our resultant masks, this provides a better indication of performance by weighting both foreground (vias) and background (everything else) segmentation equally. The DICE coefficient, also referred to as F1-Score, is computed similarly to IoU, and we once again prefer the mean DICE score result due to class imbalance between foreground and background classes in the segmentation masks. DICE is computed as follows

$$\text{Dice} = \frac{2*\text{M}_\text{p} \cap \text{M}_\text{gt}}{\text{Total \# of Pixels}}$$

DICE coefficient behaves similarly to IoU, and both are positively correlated with one another. Meaning, they are often in accordance with one another when evaluating model performance. However, an important difference is that IoU penalizes instances of misclassified pixels harsher than DICE. Lastly, we utilize SSIM once again but now for assessing the quality of our predicted mask versus the ground truth mask.

Our results utilizing these metrics for evaluation are listed in Table I comparing performance when utilizing known radii in the initial circle detection versus general radii ranges from a known max and minimum radius as well as against the results

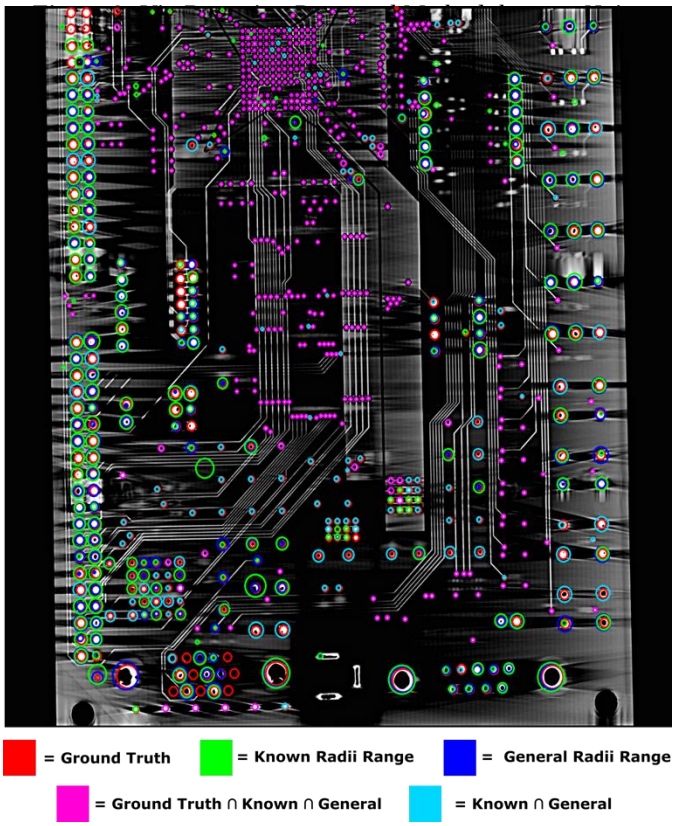


Figure 10: Via Detection Proposed Methodology performance on Layer 1 of Spartan 3 riddled with blurring and high-impedance material noise artifacts.

provided from training and implementing Mask R-CNN for via detection. The Test Board (T.B.) results are the average scores for each layer across the accumulator thresholds 10, 15, 20, and 25. These results highlight the consistency across varying quantities of initial false-positive circles to screen. Spartan 3 (S3) results are for experiments on an accumulator threshold of 15 for initial tiered circle detection only. As expected, the known radii outperformed the general across all metrics, albeit minutely. However, analyzing these results solely quantitatively can be misleading. Figures 9 and 10 highlight this for 2 layers of our Test Board and a single layer of the Spartan 3 board, highly affected by noise. All purple circles show instances where the predicted results utilizing general radii ranges or known radii ranges are both within 1 pixel difference of the ground truth for a via's center and radius. Light blue circles are where the predicted vias of both general and known radii match within 1 pixel of each other but differ by more with the ground truth. Lastly, the green circles are the general radii detected vias, blue ones are the known radii detected vias, and the red circles are the remaining ground truth.

For the proposed approach, when comparing the detection results of a layer utilizing known radii versus general radii, we can see that all vias are detected with high precision for the Test Board. Most notably in Spartan 3, the penalization observed in evaluation comes from the medium/medium 2 tier detector detecting vias but over- or underestimating their radius size. This incorrect estimation, along with the few false-positive or

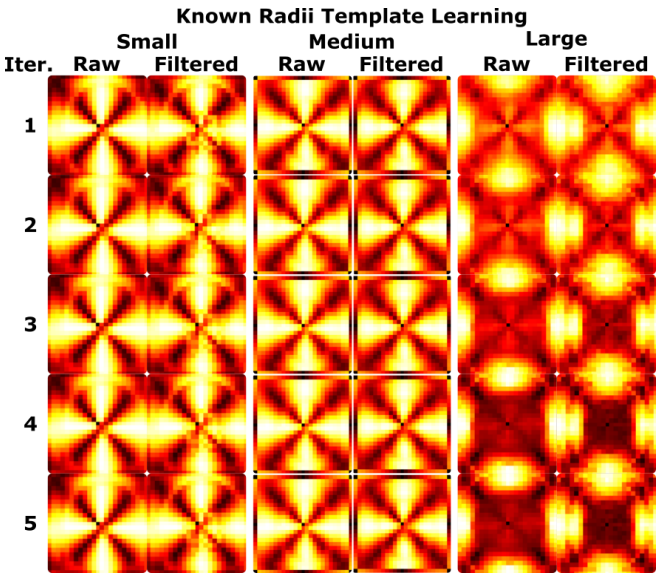


Figure 11: Known Radii Iterative Template Learning for Test Board highlighting how throughout iterations templates adapt and become sharper, most notably in large tier.

false-negative detections, can be attributed to the high degree of noise corruption from manually depopulating the board under test and its effect on the X-ray process for the image acquisition stage of PCB RE. However, we can see the robustness of the approach with highly accurate results in the presence of such noise. Therefore, while this is not a perfect detection of a via, it is not entirely inaccurate and easily remedied by an operator performing a final screening of the results. Regardless, this process provides a substantial savings of time and manpower resources by no longer requiring an RE operator to manually inspect and validate each via instead of a few blatant misses. This is most evident in a densely populated design like the Spartan 3 with over 800 genuine vias per layer.

Our utilization of Mask R-CNN can be considered an off-the-shelf implementation utilizing singular via images to fine-tune a pre-trained network. This implementation's significantly less-accurate results are outlined in Table I and highlighted in Fig. 9. For the Test Board, via classification is not accurate with a vast amount of clear missed detections. We find that sharpening our training images produces the best prediction results. The performance also improves when images are filtered and sharpened before prediction. We do not attempt any contrast enhancement techniques at any stage of use that contributes to its classification issues. Our relatively higher minimum confidence value of 0.8 reduces the number of detected vias, but this trade-off is necessary to reduce false-positive levels. Prediction results are relatively worse on the Spartan 3 Board with a significant amount more of false-positive detections. The via sizes are significantly larger on the Test Board, and this is a problem since we train on vias from the Test Board only. Polygon annotations provide more description points for objects in an image and are better suited for the Mask R-CNN implementation we utilize. Our circle annotations only record centers and radii. The Spartan 3 board also has a significantly higher amount of noise, and this makes identifying vias more difficult.

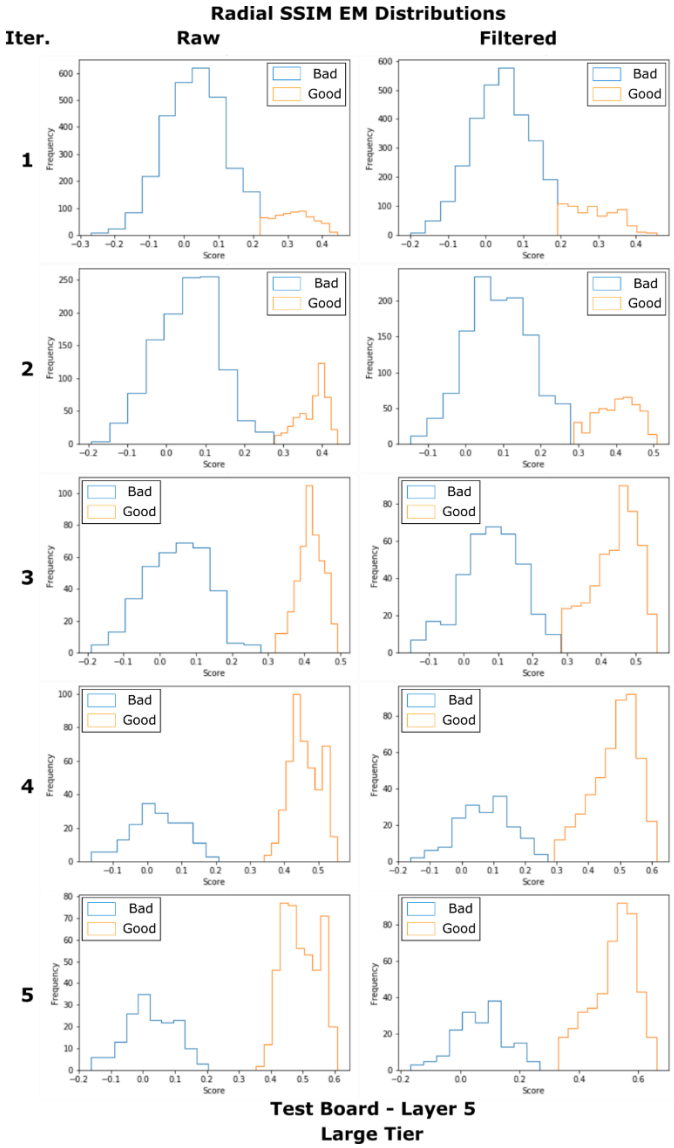


Figure 12: Radial SSIM score distributions post EM for layer 5 of Test Board utilizing large tiered detector per iteration. As algorithm iterates and template improves(Fig. 10) distributions become more separable.

A better-performing implementation of Mask R-CNN is possible by supplementing a much larger dataset of additional Via specific training images from a variety of PCB samples as opposed to the 100 used for fine tuning of the MS COCO pretrained network. Albeit, at the cost of time and manual labor for the annotation of such needed data. Furthermore, existing architectures supplemented with techniques presented here such as Hough circle detection or radial gradient behavior on a region of interest could likely improve performance.

C. Iteration

It is crucial to effectively screen and remove any vias detected falsely due to the output of the detector that can cover a wide range of candidate vias with varying quality (as high as 20,000 for T.B. and 130,000 for S3). However, it is equally important for the initial detection and false-positive removal process to generalize across various board designs. By iteratively applying false-positive removal, we can do so in a more effective

unsupervised way by learning the characterization of high-quality candidate vias unique to size, data type, and board. This is highlighted by analyzing the gradient radial templates of the filtered and raw tiered circle detectors over iterations and their impacts on SSIM scoring.

From Fig. 11, we can gain a large amount of insight into the false-positive removal process per iteration. First of all, for the Test Board we can see that there is a clear difference in templates across not only the size, but also data type. Small and medium templates are more similar to one another than their large counterparts due to the minimal difference in vias detected for those sizes on the Test Board. The large vias are of vastly different composition than the small and medium vias, resulting in a drastically different template radial behavior.

Analyzing these templates along with respective SSIM score distributions can provide insight into the quality of initial detection and convergence. The difference between the first templates of the small and medium tier compared to those in the next iterations is less significant than the large template-regardless of being filtered or raw. Thus, this signifies that the vast majority of vias detected in early iterations for the smaller tiers were screened as true positives, remaining in the later iterations. For the large tier, the template evolution over iterations shows a sparse initial detection of genuine vias, but as it iterates, it learns and characterizes better. Specifically, the initial template was not very representative of the large via behavior, unlike small and medium vias. This results in inaccurate thresholds, and consequently, few genuine vias passed screening, which is also shown by the lack of separability between distributions in Fig. 12. However, those vias passing the test were used to create the next iteration of the template that is more tuned to the characterizations of the large vias than the previous template. As being evident by the figures, the algorithm continued improving the templates, separability, and thus, the thresholds until it converged. At the final stage, we see vastly different final templates and distributions compared to the initial one.

Figure 8 reaffirms this analysis. The high degree of overlap occurring between false positive and false negative curves across iterations is due to the convergence of the algorithm for the different size tiers at different iterations. Specifically, the algorithm is modularly implemented so that when one size converges but others have not the converged size will stop iterating while the others continue. Therefore, the number of false positives and negatives for that size across the different stages remain consistent and is added to the respective false detections of the sizes still iterating. As mentioned earlier, the small and medium tiers converge quickly according to their templates in Fig. 11 and thus the only changes occurring across later iterations are seen at the Large tier and impact overall detections slightly as the final detections are fine tuned for optimal performance. While according to Fig. 8 it may seem that minimal improvement is occurring in later iterations, we can see from Fig. 12 that these later iterations are crucial for improving separability of distributions for the still iterating large tier and thus improving the confidence and quality of the final detections.

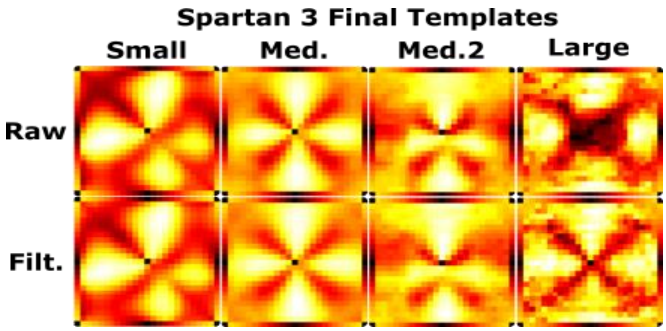


Figure 13: Final Spartan 3 templates of general radii for 4 tiers. The presence of noise and false-positives in the Spartan detections distorts and blurs templates affecting final detections to differing degrees depending on the amount of distortion.

Additionally, when using the general sizes, as opposed to knowing size ranges a priori, the templates iterate for much longer, and some learned template tiers are much less defined (See Fig. 13). This lack of definition signifies the blurring of information used to compute the templates. In other words, a lack of uniformity among the vias used to compute the template due to sizes mixing or false-positive sneaking in. When looking at the final results of the generally sized vias for Spartan 3, Fig. 9, we can see instances of detected medium or medium 2 vias encapsulating a bit more of the background image with the via than necessary. This is likely the source of the blurring of these templates.

Lastly, we can see the robustness and unsupervised learning ability of our approach across board datasets. Specifically, regardless of the differing imaging quality and feature density, the Test Board and Spartan 3 have accurate detection results due to the ability to learn their very different templates, see Figs. 11 and 13. The Spartan dataset is clearly very noisy compared to the Test Board, but still, we can see that the algorithm adapted and learned the templates appropriate to each tier due to our other false-positive removal steps in combination with learning a better-characterized template unique to the board, tier, and data type.

V. Conclusions

In conclusion, this paper achieves a vital step necessary in the automation of RE for PCBs. Not only are vias detected in a highly accurate automated fashion, but this framework is adaptable across board designs. The clear next steps are to extend our approach of combining domain knowledge with image analysis techniques to other essential stages in automated PCB RE. Most notably, trace identification, vectorization, and holistic evaluation.

Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1315138 and DGE-1842473.

References

[1] J. B. Robertson and M. B. B. Riley, “The Big Hack: Statements From Amazon, Apple, Supermicro, and the Chinese Government,” 2018. [Online]. Available: https://www.bloomberg.com/news/articles/2018-10-04/the-big-hack-amazon-apple-supermicro-and-beijing-respond

[2] S. K. I. S. Moore, “This Tech Would Have Spotted the Secret Chinese Chip in Seconds,” IEEE Spectrum, 2018. [Online]. Available: https://spectrum.ieee.org/riskfactor/computing/hardware/thistech-would-have-spotted-the-secret-chinese-chip-in-seconds

[3] J. Grand, “Printed Circuit Board Deconstruction Techniques,” in Proceedings of the 8th USENIX Workshop on Offensive Technologies - WOOT ’14, 2014. [Online]. Available: https://www.usenix.org/conference/woot14/workshopprogram/presentation/grand

[4] N. Asadizanjani, M. Tehranipoor, and D. Forte, “PCB Reverse Engineering Using Nondestructive X-ray Tomography and Advanced Image Processing,” IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 7, no. 2, pp. 292–299, 2017.

[5] U. J. Botero, R. Wilson, H. Lu, T. Rahman, M. A. Mallaiyan, F. Ganji, N. Asadizanjani, M. M. Tehranipoor, D. L. Woodard, and D. Forte, “Hardware Trust and Assurance through Reverse Engineering: A Survey and Outlook from Image Analysis and Machine Learning Perspectives,” Tech. Rep.

[6] U. J. Botero, N. Asadizanjani, D. L. Woodard, and D. Forte, “A Framework for Automated Alignment and Layer Identification of XRay Tomography Imaged PCBs,” in GOMACTech, San Francisco, 2020.

[7] H. Kalvi ¨ ainen, P. Hirvonen, L. Xu, and E. Oja, “Probabilistic and non- ¨ probabilistic Hough transforms: overview and comparisons,” Image and Vision Computing, 1995.

[8] R. O. Duda and P. E. Hart, “Use of the Hough Transformation to Detect Lines and Curves in Pictures,” Communications of the ACM, vol. 15, no. 1, pp. 11–15, 1 1972. [Online]. Available: http://portal.acm.org/citation.cfm?doid=361237.361242

[9] E. R. Davies, “A modified Hough scheme for general circle location,” Pattern Recognition Letters, vol. 7, no. 1, pp. 37–43, 1 1988.

[10] J. Illingworth and J. Kittler, “SURVEY A Survey of the Hough Transform,” Tech. Rep., 1988.

[11] P. Kierkegaard, “A method for detection of circular arcs based on the Hough transform,” Machine Vision and Applications, vol. 5, no. 4, pp. 249–263, 9 1992.

[12] K. L. Chung and Y. H. Huang, “Speed up the computation of randomized algorithms for detecting lines, circles, and ellipses using novel tuningand LUT-based voting platform,” Applied Mathematics and Computation, vol. 190, no. 1, pp. 132–149, 7 2007.

[13] G. M. Schuster and A. K. Katsaggelos, “Robust circle detection using a weighted MSB estimator,” in Proceedings - International Conference on Image Processing, ICIP, vol. 3, 2004, pp. 2111–2114.

[14] S. Das, S. Dasgupta, A. Biswas, and A. Abraham, “Automatic circle detection on images with annealed differential evolution,” in Proceedings - 8th International Conference on Hybrid Intelligent Systems, HIS 2008, 2008, pp. 684–689.

[15] V. Ayala-Ramirez, C. H. Garcia-Capulin, A. Perez-Garcia, and R. E. Sanchez-Yanez, “Circle detection on images using genetic algorithms,” Pattern Recognition Letters, vol. 27, no. 6, pp. 652–657, 4 2006.

[16] Y. Qiao and S. H. Ong, “Connectivity-based multiple-circle fitting,” Pattern Recognition, vol. 37, no. 4, pp. 755–765, 4 2004.

[17] C. Akinlar and C. Topal, “EDCircles: A real-time circle detector with a false detection control,” Pattern Recognition, vol. 46, no. 3, pp. 725–740, 3 2013.

[18] Z. Wang, E. P. Simoncelli, and A. C. Bovik, “Multi-scale structural similarity for image quality assessment,” in Conference Record of the Asilomar Conference on Signals, Systems and Computers, vol. 2, 2003, pp. 1398–1402.

[19] D. Comaniciu and P. Meer, “Mean Shift: A Robust Approach toward Feature Space Analysis,” Tech. Rep.

[20] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data Via the ¡i¿EM¡/i¿ Algorithm,” Journal of the Royal Statistical Society: Series B (Methodological), vol. 39, no. 1, pp. 1–22, 9 1977. [Online]. Available: http://doi.wiley.com/10.1111/j.2517-6161.1977.tb01600.x

[21] I. McLoughlin, “Secure embedded systems: The threat of reverse engineering,” in Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS, 2008, pp. 729–736.

[22] N. A. Macmillan and C. D. Creelman, Detection Theory: A User’s Guide: 2nd edition. Lawrence Erlbaum Associates, 8 2004.

[23] “Design Rules Check — Worthington Assembly Inc.” [Online]. Available: https://www.worthingtonassembly.com/design-rules-check

[24] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” Tech.Rep. [Online]. Available: https://github.com/

[25] A. Dutta and A. Zisserman, “The VIA Annotation Software for Images, Audio and Video,” in Proceedings of the 27th ACM International Conference on Multimedia. New York, NY, USA: ACM, 10 2019, pp. 2276–2279. [Online]. Available: http://dl.acm.org/doi/10.1145/3343031.3350535

[26] “Visual Geometry Group - University of Oxford.” [Online]. Available: http://www.robots.ox.ac.uk/vgg/software/via/

[27] D. M. W, “EVALUATION: FROM PRECISION, RECALL AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION,” Journal of Machine Learning Technologies, vol. 2, no. 1, pp. 37–63, 2011. [Online]. Available:

http://dspace.flinders.edu.au/dspace/http://www.bioinfo.in/contents.php?id=51
[28] F. Ge, S. Wang, and T. Liu, “New benchmark for image segmentation evaluation,” 2007.