A Framework for PCB Design File Reconstruction from X-ray CT Annotations

Carson Sobolewski
Florida Institute for National Security
University of Florida
Gainesville FL, USA
csobolewski@ufl.edu

David S. Koblah
Florida Institute for National Security
University of Florida
Gainesville FL, USA
dkoblah@ufl.edu

Domenic Forte
Florida Institute for National Security
University of Florida
Gainesville FL, USA
dforte@ece.ufl.edu

***Abstract*—Reverse engineering (RE) is often used in security-critical applications to determine the structure and functionality of various systems, including printed circuit boards (PCBs). Although it has both beneficial and malicious uses, it is particularly vital within the realm of hardware trust and assurance. PCB RE enhances legacy electronic system replacement, intellectual property (IP) protection, and supply chain integrity. To contribute to the requirements of effective PCB RE, extensive research has been conducted on the analysis of PCBs using X-ray computed tomography (CT) scans, including image segmentation focusing on via and trace annotation. Applying extracted annotations, this work outlines a Python-based framework, coupled with the open-source KiCaD software, for the automated reconstruction of PCB design files. Given the via, pad and trace annotations, in addition to board dimensions, the algorithm automatically recognizes board shape, trace size, and connections to reconstruct the bare PCB accurately. This technique was tested on three distinct layers of a sample multilayer PCB with great success. Its feasibility holds great promise for future extensions to complete the entire PCB RE framework. The project source code is available for reference on GitHub.** [1]

***Index Terms*—printed circuit board, reverse engineering, via, trace, annotation**

I. Introduction

Access to electronic hardware designs presents adversaries with the opportunity to not only modify existing designs but also to replicate them, leading to the infiltration of counterfeit products across the supply chain. This poses a significant threat to both general and sensitive applications. The country's military supply is especially vulnerable to counterfeit products, which affect the safety of service members [1]. As outlined in the nearly decade-old report submitted to Congress by the Department of Defense (DoD), the United States is confronted with challenges in the global PCB market, where Asia dominates with 90% of manufacturing, leaving the U.S. with a mere 5% share [2]. This heavy reliance on foreign-produced PCBs for military needs raises profound security concerns.

In the pursuit of hardware trust and assurance, RE plays various crucial roles. For legacy electronic systems, it addresses the issue of obsolescence due to the unavailability of original design files and schematics. Although they remain useable for decades, finding compatible equipment replicas on the market may be laborious. On the other hand, settling for upgrades instead may raise compatibility issues. With a robust RE process that successfully reproduces the structure and functionality, obsolescence may be avoidable for these legacy systems [3]. RE is also invaluable in the identification of malicious alterations to an existing hardware design. Within the design cycle, untrusted parties can maliciously alter original design requirements. With the prevalence of attacks, such as hardware trojans [4], a seamless detection method enhances the security of the electronic supply chain.

Traditionally, RE methods have involved destructive techniques such as physically delayering a PCB for detailed imaging, which can be labor-intensive and time-consuming. However, recent advancements have introduced non-destructive options, such as X-ray CT. X-ray CT enables the creation of a 3D volumetric model of the PCB sample without causing damage. In particular, Asadizanjani et al. [5] introduced a non-destructive framework combining image-stitching, reconstruction, filtering, and segmentation to produce a comprehensive 3D model of the PCB. This approach facilitates examination along all three-dimensional axes, revealing vital information about vias and traces, which are essential elements that enable the routing of electrical signals across different layers and throughout the board. However, the effectiveness of this method can be hindered by noise, impacting the identification of smaller vias and thinner traces. To address these challenges, Botero et al. proposed a via-focused detection approach, leveraging techniques such as Hough Circle Transform [6] and frequency-based filtering to enhance detection accuracy and automate the process [7]. While this approach showed promising results, it required prior knowledge of via radii range and sizes, which could limit its generalizable to different PCB designs. In response, a recent paper proposed a novel deep-learning approach capable of identifying vias across various PCB designs without the need for predefined parameters. This method demonstrated significantly faster runtime with quality of results comparable to or better than the unsupervised iterative Hough-based method [8]. The contributions included faster feature-based detection and adaptive post-processing methods.

In the work outlined in this paper, we attempt to round off the PCB RE framework by utilizing annotations of vias and traces to regenerate a viable layout. Once the layout recon-

[1] https://github.com/koblahdavid/xray_ct_reconstruction

Fig. 1: An overview of the RE process performed on PCBs

struction is complete, we can extract the full structure of the PCB and conduct vital tests. By combining this reconstructed layout with a complete bill of materials, we can generate a near-perfect and highly accurate replica of the device-under-test (DUT). The main contributions outlined in this paper are:

- A Python-based framework that exploits vectorized segmentation results to reproduce useable PCB layout files.
- Seamless interaction with open-source KiCAD PCB design software to provide real-world results.
- Modular system with the ability to be integrated into an existing PCB RE toolkit.

The rest of the paper is organized as follows: Section II introduces the key concepts in this paper. Section III explains the methodology, including how each key component is identified. Section IV provides qualitative results, with corresponding discussions. Section Vconcludes the paper and provides directions for future work.

II. Background & Related Work

Since this framework adds to the existing PCB RE process, capturing the core concepts and foundational work that underpin the proposed approach is essential. The following subsections provide the necessary context to fully understand the contributions presented in this paper.

A. PCB Reverse Engineering

The typical RE process comprises of delayering, imaging, annotation, and extraction. While the integrated circuits (ICs) and PCBs vary in their specific needs, key aspects of the current steps cut across both device classifications (see Fig. 1). Both versions have also been generally characterized as tedious, laborious, and costly for stakeholders at all levels. However, ongoing research has forged towards automating various stages of the process. Ultimately, a swift, efficient, and automated method greatly benefits hardware trust and assurance needs, particularly in fulfilling requirements such as hardware trojan detection [9]. The depopulation step entails carefully detaching all attached components from the populated board to expose underlying traces, vias, and the board structure itself. It allows for unobstructed access to the board's architecture, ensuring a clear view for subsequent imaging and analysis. Once depopulation is complete, imaging is conducted using a suitable modality, such as optical, X-ray, or scanning electron microscopy, depending on the level of detail required. Each imaging technique offers distinct advantages: optical imaging provides an overview of the board layout, X-ray imaging reveals internal structures and hidden layers, and scanning electron microscopy (SEM) allows for high-resolution examination of fine details. In this paper, we focus on non-destructive X-ray computed tomography (CT), highlighting its crucial role and emphasizing its feature resolution capabilities. Annotation is crucial for creating an accurate representation of the board's components and layout, which serves as the foundation for subsequent stages in the analysis pipeline. Layout extraction is the final, integrative phase that brings together all extracted and annotated components to reconstruct the entire architecture. This enables a holistic understanding of the system's design and operational pathways.

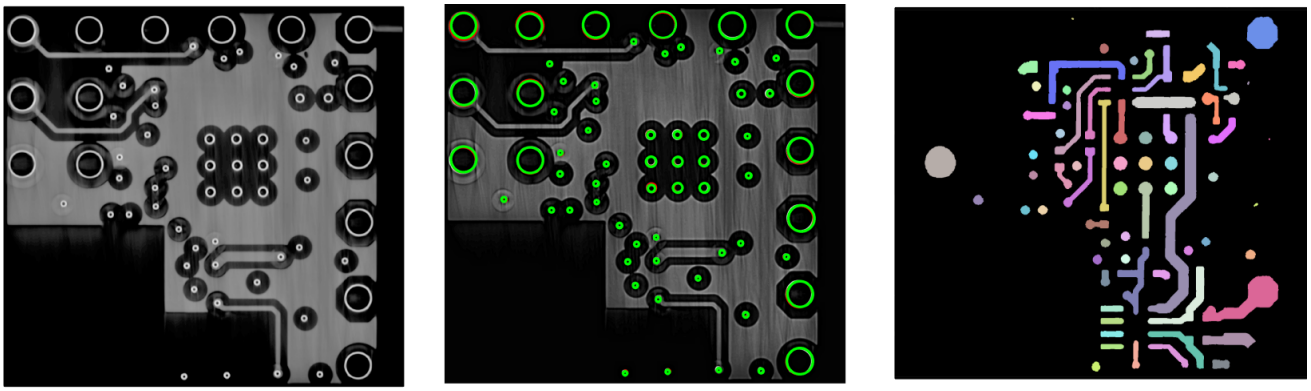
1) X-ray Computed Tomography: X-ray CT, widely employed in electronics and structural assessments, generates a three-dimensional representation by compiling multiple two-dimensional projections, providing a comprehensive view of the object's internal composition [10]. While imaging technologies like scanning electron microscopy (SEM) [11] and focused ion beam (FIB) [12] tend to be more destructive, they offer higher resolution capabilities beneficial for discerning much smaller features, particularly those found in ICs. The X-ray CT modality is particularly valuable for the modern-day multi-layered PCB [5] . The three main stages of the X-ray CT framework applied to multi-layered PCBs are:

1) **Scan**: A set of x-ray projections are taken over a rotation of the imaging axis of 180 or 360 degrees.
2) **Reconstruction**: The projections are processed by the filtered Feldkamp cone-beam method to create the stack of cross-section slices.
3) **Analysis and visualization**: The reconstructed cross-section slices are processed into three-dimensional models for visual inspection.

The typical tomography process requires 500-1500 projections and several hours between acquisition and reconstruction

2) Annotation: The role of annotations in the PCB RE process cannot be understated. They help highlight and understand the structure and positions of different elements in a device. The annotation module comprises of three submodules: denoising, segmentation, and vectorization.

Due to varied imaging conditions during the X-ray process, denoising applies noise suppression through various techniques. After acquiring the image data, the nature of noise is assessed to determine the most appropriate denoising technique [7]. Some denoising algorithms include Gaussian smoothing [13], median filtering [14], non-local mean denoising [15], wavelet transform [16], and deep learning-based



a) b) c)

Fig. 2: (a) Sample PCB X-ray CT layer with (b) via and (c) trace segmentation results

denoising [17]. Parameter tuning is often necessary to optimize the denoising process.

Segmentation involves separating structures within the image based on grayscale pixel intensity or other qualifying factors. For a depopulated PCB, the main objects of interest are vias, traces, and pins. For vias, a two-stage object detection system uses deep learning to extract single and layer-by-layer positions. It can retrieve vias at varying scales [8].

The trace detection framework facilitates the automated and unsupervised extraction of both traces and copper conducting planes from PCB designs. It uses cycles in graph space to separate incorrectly connected components, thereby yielding high-fidelity results in image space [18]. This approach, coupled with the automated via detection system, unifies the Annotation stage of RE and helps create a comprehensive workflow crucial for designing trustworthy PCBs.

Finally, the vectorization stage converts the image into useable polygon shapes. This allows the recovery of design files as close to the original PCB layout as possible.

B. KiCad

KiCad is a popular open-source software suite within the realm of electronic design automation (EDA), primarily used for making schematic diagrams and PCB layouts [19]. Its capabilities encompass the complete design process, including the generation of manufacturing files. It is available on Windows, macOS, and Linux, and allows collaborative engagement and ongoing development within the community, ensuring continual refinement and augmentation of its capabilities. As a research tool, KiCad is cost-effective for most PCB design needs.

III. Methodology

Due to the unique challenges presented by each core component of the proposed framework, this section emphasizes how the algorithm specifically addresses each one. Hence, the critical elements, *pads*, *vias*, and *traces*, are detailed individually. A high-level outline of our algorithm can be seen in Algorithm 1.

A. *Image Preprocessing*

Although noiseless images are ideal for any processing tasks, X-ray CT images are often distorted, either by rotation, extended areas, or other deviations. Before analyzing the pads, vias, and traces, it is crucial to ensure the image is properly aligned. This is because our annotation coordinates are provided as pixel coordinates that need to be translated to fit the new board location. The alignment process involves rotating and cropping the input image before performing further analysis. Both procedures are described below:

a) ***Rotation****:* To properly align the board to a 90-degree angle, we first perform line detection to detect traces and other major line segments on the board image. We use a Hough transform [6], specifically paying attention to lines within 5 degrees of verticality and horizontality. With the results, we find other vertical or horizontal lines present. Using the median angle of the largest group of selected lines, we rotate the board to align with the nearest 90-degree angle increment. The angle of rotation is then used to adjust the pad, via, and trace coordinates.

b) ***Cropping****:* After board alignment via rotation, cropping is required to prepare the image. Using Canny edge detection [20], we find the outer edges on the board. After this, the image is cropped to the found edges of the board. Like the rotation process, the coordinate change is stored for later use in calculating pad, trace, and via coordinates.

B. *Annotation Analysis*

After aligning and cropping the board, we begin annotation analysis. The S3A tool [21] processes given annotations, but cannot identify each annotation type (pad, via, or trace). Hence, we manually add the types to assist our framework. The next subsections detail these distinct methods.

1) ***Challenges of Retrieving Annotations****:* Each component of interest requires different operations to achieve optimal results. The challenges of each one show the difficulties of this undertaking.

a) Pads: For pads, our annotations, labeled as perimeter points, are sufficient for drawing in KiCad's `pcbnew` PCB

editor. As a result, we simply add the list of points for each pad to the `padList` list.

b) Vias: To draw vias in `pcbnew`, we require diameter and center points. However, our data is provided as labels along the circumference of the circular vias. We use the `OpenCV moments()` function to find the center point and then calculate the diameter by subtracting the minimum X coordinate among the circle's points from the maximum X coordinate. The center point is added to the `centerList` list, the diameter is added to the `diameterList` list, and the via circumference points are added to the `viaList` list.

c) Traces: Like pad and via annotations, the trace annotations are provided as a set of perimeter points. However, `pcbnew` a center line and a width for each segment of the trace. To obtain the center line, we first convert the perimeter points into a `Shapely` polygon. After this, we use the `centerline` package to compute an estimated center line for the trace using a Voronoi diagram [22]. However, the generated center lines are mostly uneven. Similarly, the endpoints of the traces usually have extraneous center line segments that extend into corners, as seen in Fig. 3. To remove the extraneous line segments, we first create a list with tuple entries for each segment and the corresponding length. We then use this list to create a weighted `NetworkX` graph to calculate the shortest paths. We find the short pairwise paths with maximum lengths between points in the graph using Dijkstra's shortest path algorithm [23], keeping those within 15% of the maximum shortest path distance. The percentage value was empirically determined during our experimentation process. From these paths, we determine the least angular fluctuation by summing the change in angle between each segment in the center line. The true center line should be one of the longest in the graph while having less than the maximum angle change (the significant angle change at the end with an extraneous segment is eliminated). After determining which set of points corresponds to the true center line, we cut down the line segment count and smooth out the irregularities in the line by creating a new line that has points only at significant angular changes (empirically determined to be greater than 20 degrees). This eliminates most of the smaller segments. Lastly, we round up the angular values of each segment in the smoothed line to the nearest 45-degree angle. This is to enforce proper PCB design rules since most production circuit boards have their traces locked to 45-degree increments. It helps our algorithm to produce realistic and feasible KiCad design files. The final updated path is added to the `updatedLinesList` list.

*2) **Design Creation**:* After analyzing the annotations to get the `padList`, `centerList`, `diameterList`, `viaList`, and `updatedLinesList` lists, we then call `pcbnew's` Python libraries to create the board and its components. However, we must first convert the pixel coordinates of our annotations to the millimeter format used by the KiCad application. We do this by extracting the dimensions of the board in millimeters and then calculating the ratio of millimeters per pixel in the XY direction. After calculating the ratio of millimeters per pixel, we create a rectangular board with the provided dimensions. Similar to Section III-B, we perform different operations for each type of object.

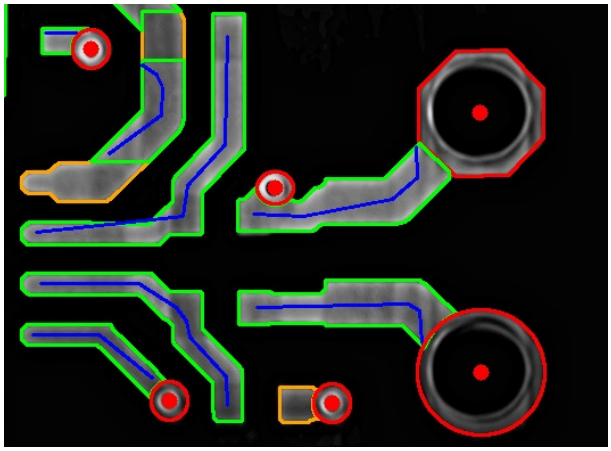


Fig. 3: A depiction of the extraneous segments on the generated center lines.

Algorithm 1 PCB Reconstruction Algorithm

```
Perform line detection using Hough transform
Rotate image based on number of vertical/horizontal traces
Load modified S3A annotations
for each annotation do
    if annotation is a pad then
        Add pad to padList
    else if annotation is a via then
        Detect center and diameter
        Add center and diameter to respective lists
        Add via points to viaList
    else if annotation is a trace then
        Turn edge points into polygon
        Find centerline of polygon
        Create graph and find shortest paths
        Simplify path and round angles
        Add updated path to updatedLinesList
    end if
end for
Use Canny edge detection to locate board edges
Crop image based on detected edges
Calculate mm per pixel ratio
Create PCB layout using pcbnew library
Add vias and pads to layout
Connect traces to pads or vias
Save board file
```

a) Pads: For pads, the required input is the `padList` for the footprint of the pad. However, we also need a location for its center. This is done by the average of each of the points in the pad from `padList` to provide the center position. After this, we create the pad, set its shape, extract a footprint, add the pad to it, and finally add the footprint to the board.

b) Vias: Here, we draw the vias using center points and diameters in `centerList` and `diameterList`. We use the

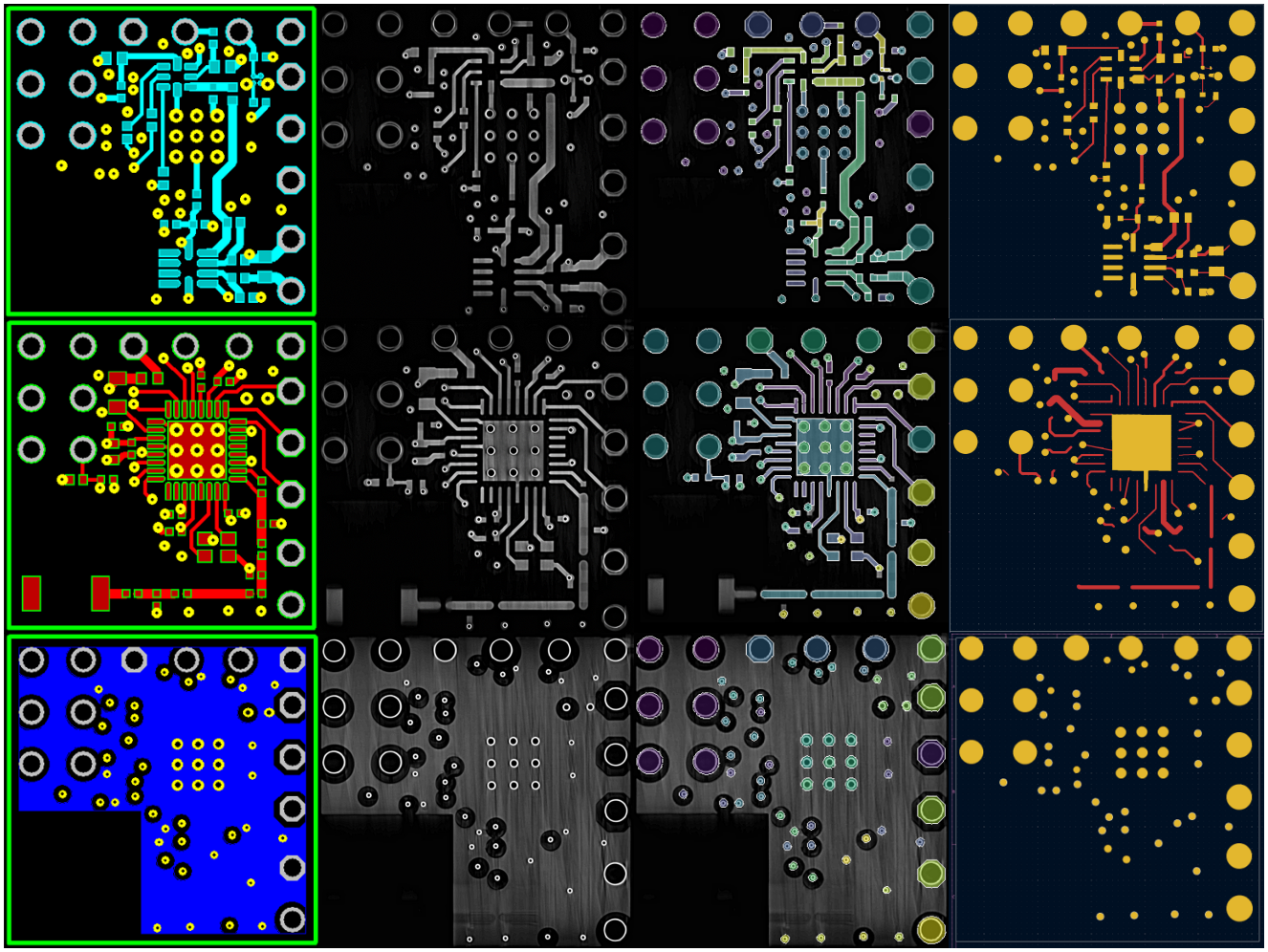


Fig. 4: Test board results for layers 1, 2, and 3 from top to bottom. In order from left to right, this figure shows the ground truth board, X-ray CT image, annotations, and design reconstruction.

`PCB_VIA()` class with a type of through. For the layers, we default to using `F.Cu` and `B.Cu`, which represent the front and back copper layers respectively. To supplement this, we set the drill diameter to 0.1 mm under the size of the via. The via is now added to the board for drawing.

c) Traces: We generate traces from the data in the `updatedLinesList`, carefully accounting for the surrounding layout to connect traces to nearby pads and vias, as the annotations do not include this information. To accomplish this, we iterate through each set of trace segments in `updatedLinesList`. For each trace, we identify the nearest pad or via and, if it is within a set proximity, we attempt to establish a connection. If the trace has fewer than three segments, making it difficult to extrapolate a connection, we directly connect to the geometric midpoint of the nearest pad or via. For longer traces, we first remove the last segment to avoid redundancy and then extrapolate the trace over a defined distance to check for intersections with a pad or via. If the extrapolated line intersects, we connect the trace at that point. Otherwise, we try connecting from the original trace endpoint, if feasible. Finally, we save the board file and generate the final design output.

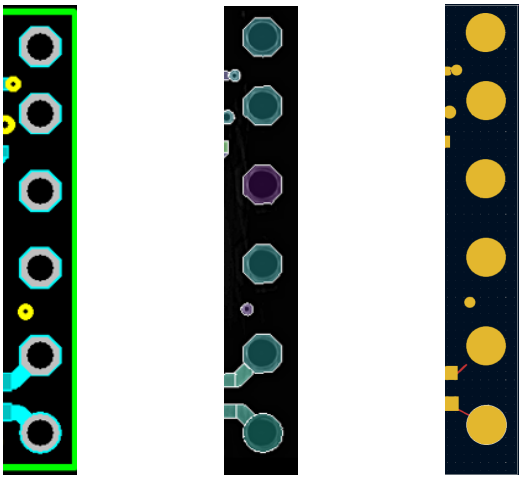
IV. Experiments & Results

For experimentation, we ran our framework on three layers of an in-house test PCB. The algorithm was primarily designed while testing on the first layer of the board and then applied without modification on the other two layers. Qualitative results are summarized in Fig. 4. The rest of this section provides further quantitative analysis of framework functionality:

To evaluate our framework, we conducted experiments on three layers of an in-house test PCB. Initially, the algorithm was designed and refined using the first layer, after which it was applied without further modifications to the remaining two layers. Qualitative outcomes are shown in Fig. 4, and the subsequent sections describe the analysis in detail.

A. Layer 1

This layer is distinguishable by its pads and contacts, as seen in Fig. 4. Due to these differences, we approach the extraction process distinctly preceding reconstruction.



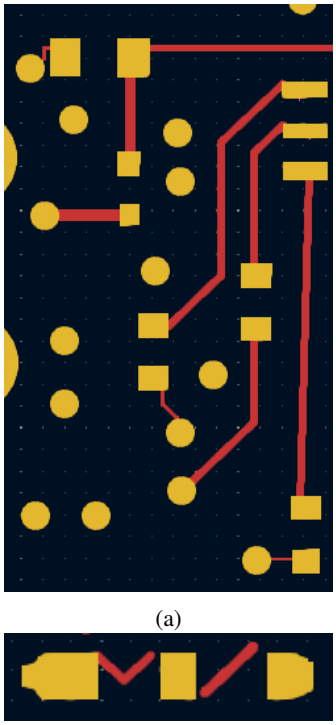
(a) (b) (c)

Fig. 5: A set of layer 1 vias in (a) ground truth, (b) annotations, and (c) reconstruction.

1) Pads: We carefully annotated the pads in the first layer using the S3A tool. Manual annotation allowed for clear differentiation of pads based on their darker contrast relative to the surrounding areas and their characteristic rectangular extensions from the main trace. Leveraging these detailed annotations, we reconstructed the pads with high fidelity, closely matching both the annotated data and the ground truth. However, minor deviations in pad shape were observed in the ground truth dataset.

2) Vias: Similar to the pads, we found the vias straightforward to reconstruct when provided with accurate input labels. By utilizing the center point and diameter calculations, we produced output vias that closely align with the ground truth and X-ray CT images. However, the reconstruction algorithm assumes all vias are circular, which results in non-circular vias being approximated as circles. This limitation is evident in Fig. 5, where the vias on the right side of layer 1 are correctly positioned and scaled but fail to retain the octagonal shape seen in the ground truth data, X-ray CT images, and annotations.

3) Traces: The algorithm's true performance is challenged in the reconstruction of traces and their connections to vias and pads. In layer 1, 38 traces are accurately reconstructed, as demonstrated in Fig. 6a. However, certain regions pose difficulties, particularly those with very short traces that lack sufficient length for effective extrapolation (as shown in Fig. 6b). This sub-optimal performance in small trace connections is likely due to insufficient data for accurate extrapolation—particularly when traces consist of fewer than three line segments. The algorithm's overall success in layer 1 can be attributed to the high density of pads and vias, which aids in aligning traces and eliminating extraneous segments when connections are properly established.



(a)

(b)

Fig. 6: (a) Good and (b) bad trace reconstructions for layer 1 of the test board.

B. Layer 2

In this layer, certain copper elements present challenges to the extraction process. We explain the results produced in the following sub-sections.

1) Pads: Layer 2 differs significantly from Layer 1, primarily due to the absence of labeled pads, except for the central rectangular copper block. This limitation makes a direct comparison of pad reconstruction challenging. The sole pad present matches the annotations precisely. However, discrepancies arise when comparing the annotations and X-ray CT images with the ground truth. The X-ray CT image shows

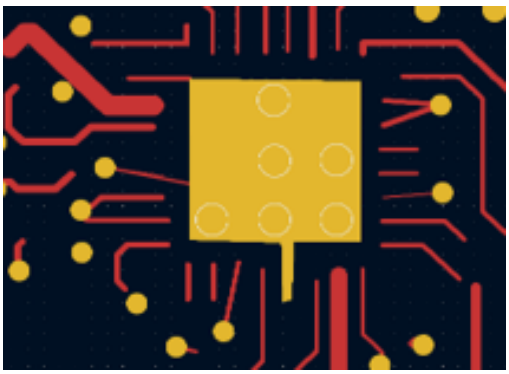


Fig. 7: Poor trace reconstructions surrounding the center copper block in layer 2 of the test board.

Fig. 8: Via reconstructions improperly shifted off of the board in layer 3 of the test board.

the pad merged with the main copper block, while the ground truth indicates some separation. This discrepancy likely results from the X-ray CT capturing a lower layer where a bridged trace connects the copper block to the pad.

2) Vias: The vias in Layer 2 are consistent with those in Layer 1, both in size and position. However, similar to Layer 1, the reconstructed files do not preserve the octagonal shape of the vias.

3) Traces: Trace reconstruction in Layer 2 poses significantly more challenges compared to Layer 1, particularly in areas surrounding the central copper block (Fig. 7). While the traces follow the correct general direction and location, their refinement is notably poorer. This degradation in trace accuracy can be attributed to the absence of pads, which play a crucial role in establishing structural integrity during the algorithm's reconstruction process. Inaccurate annotations exacerbate the issue, with missing traces (e.g., in the bottom left corner), incorrect pad labels, and faulty groupings, such as the bridged pad, contributing to the problem. Without the structural guidance provided by pad connections, traces often diverge in arbitrary directions, exacerbated by extraneous segments generated by the center-line algorithm. Consequently, the algorithm requires at least 80% of pads to produce accurate and realistic design outputs.

C. Layer 3

Layer 3 does not contain pads or traces, displaying only vias. While the vias are consistent in size and shape with those in Layers 1 and 2, many of the via reconstructions in Layer 3 are misaligned, with some shifted off the board, as shown in Fig. 8. This misalignment is likely caused by the presence of a large copper block covering most of the board. The Canny edge detection used for board cropping incorrectly identifies the copper block as the board's boundary, rather than the true board edges. Consequently, the improper cropping results in shifted coordinates, leading to vias being partially drawn off the board in the reconstruction.

D. Overlay Comparison of Layouts

We analyze the similarity between the reverse-engineered layout and the original layout using two key metrics: Intersection over Union (IoU) [24]and Hausdorff Distance [25]. These metrics provide a quantitative measure of the degree of overlap and the point-wise correspondence between the two layouts, respectively.

1) Intersection over Union (IoU): IoU is a commonly used metric for evaluating the accuracy of object detection and segmentation algorithms. It is defined as the ratio of the area of intersection to the area of union between two shapes. Mathematically, IoU is expressed as:

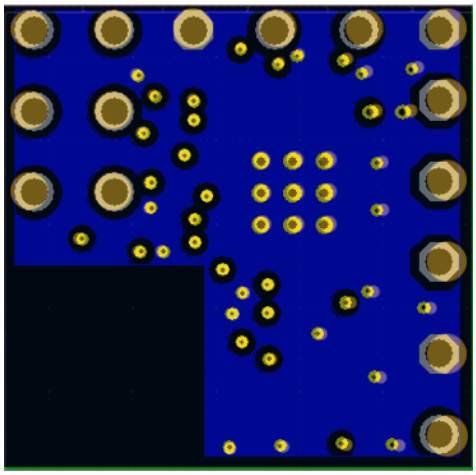


Fig. 9: A sample PCB layout layer with overlay result

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} \tag{1}$$

IoU values range from 0 to 1, where a value of 1 indicates a perfect overlap, and a value of 0 indicates no overlap at all. For our work, it provides insight into the extent to which the reverse-engineered layout spatially coincides with the original layout.

2) Hausdorff Distance: This measures the degree of mismatch between two sets by evaluating the maximum distance from a point in one set to the nearest point in the other set. Formally, the Hausdorff Distance $H(A, B)$ between two sets A and B is defined as:

$$H(A, B) = \max(h(A, B), h(B, A)) \tag{2}$$

where

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\| \tag{3}$$

and $\|\cdot\|$ denotes the distance metric, typically the Euclidean distance. A lower Hausdorff Distance indicates closer correspondence between the two layouts.

3) Overlay Results: For our comparison, we compute the IoU and Hausdorff Distance between the result and the original layout, as seen in Fig. 9. These are averages of the reported layers in the earlier sections. The results are as follows:

- **IoU**: 0.481
- **Hausdorff Distance**: 73.566

The IoU value suggests limited overlap between the reverse-engineered and original layouts, while the Hausdorff Distance of approximately 73.566 is relatively large, indicating significant discrepancies between certain points in the reverse-engineered layout and their nearest counterparts in the original layout. The observed metrics underscore the challenges of the current method for reverse-engineering layouts. Discrepancies arise from factors such as shape differences, where the process

produces simplified or approximated shapes that differ from the original layout, often resulting in polygonal segments becoming curved and fine details being lost or exaggerated. Additionally, slight positional variations, including shifts in position, scaling, or rotation during overlay, can significantly reduce the overlapping area, leading to lower IoU and higher Hausdorff Distance. To enhance accuracy in future efforts, several strategies can be employed: utilizing more precise alignment techniques, implementing advanced shape-matching algorithms to better capture the original layout's intricacies, and conducting iterative refinements through repeated comparisons and adjustments.

E. Design for Manufacturability (DFM)

The DFM report successfully describes certain aspects of the 6-layer PCB. It has dimensions of 5.670 inches by 5.670 inches and a board thickness of 0.0081 inches. It details 12 unique drill sizes, with a minimum hole size of 0.00394 inches, and does not feature any blind or buried vias. The typical trace width is 0.0243 inches, while the minimum trace width is 0.00334 inches.

V. Conclusion & Future Work

The primary limitations of the current implementation stem from minor challenges associated with imperfect annotations, which can marginally impact the alignment with ground truth data. Furthermore, ongoing efforts are focused on optimizing trace connections and refining design rules, recognizing the need for further improvements in these areas. Future work will prioritize automating schematic construction, incorporating advanced techniques such as neural networks to improve the accuracy of trace reconstruction and connections. This approach aims to address current limitations and significantly enhance the capabilities of PCB reverse engineering methodologies.

References

[1] N. Asadizanjani, M. T. Rahman, and M. Tehranipoor, *Physical Inspection of Printed Circuit Boards*. Cham: Springer International Publishing, 2021, pp. 67–99. [Online]. Available: https://doi.org/10.1007/978-3-030-62609-9_4

[2] L. Greenemeier, "The pentagon's seek-and-destroy mission for counterfeit electronics," Apr 2017. [Online]. Available: https://www.scientificamerican.com/article/the-pentagon-rsquo-s-seek-and-destroy-mission-for-counterfeit-elect

[3] U. J. Botero, N. Asadizanjani, D. L. Woodard, and D. Forte, "A framework for automated alignment and layer identification of x-ray tomography imaged pcbs," in *Proceedings of the Annual GOMACTech Conference*, 2020.

[4] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor, "Hardware trojans: Lessons learned after one decade of research," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 22, no. 1, pp. 1–23, 2016.

[5] N. Asadizanjani, M. Tehranipoor, and D. Forte, "PCB Reverse Engineering Using Nondestructive X-ray Tomography and Advanced Image Processing," *IEEE Trans Compon Packaging Manuf Technol*, vol. 7, no. 2, pp. 292–299, 2017.

[6] J. Illingworth and J. Kittler, "A survey of the hough transform," *Computer vision, graphics, and image processing*, vol. 44, no. 1, pp. 87–116, 1988.

[7] U. J. Botero, F. Ganji, N. Asadizanjani, D. L. Woodard, and D. Forte, "Semi-supervised automated layer identification of x-ray tomography imaged pcbs," in *2020 IEEE Physical Assurance and Inspection of Electronics (PAINE)*. IEEE, 2020, pp. 1–6.

[8] D. S. Koblah, U. J. Botero, S. P. Costello, O. P. Dizon-Paradis, F. Ganji, D. L. Woodard, and D. Forte, "A fast object detection-based framework for via modeling on pcb x-ray ct images," *J. Emerg. Technol. Comput. Syst.*, vol. 19, no. 4, oct 2023. [Online]. Available: https://doi.org/10.1145/3606948

[9] C. Bao, D. Forte, and A. Srivastava, "On reverse engineering-based hardware trojan detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 49–57, 2016.

[10] P. J. Withers, C. Bouman, S. Carmignato, V. Cnudde, D. Grimaldi, C. K. Hagen, E. Maire, M. Manley, A. Du Plessis, and S. R. Stock, "X-ray computed tomography," *Nature Reviews Methods Primers*, vol. 1, no. 1, p. 18, 2021.

[11] A. Mohammed and A. Abdullah, "Scanning electron microscopy (sem): A review," in *Proceedings of the 2018 International Conference on Hydraulics and Pneumatics—HERVEX, Băile Govora, Romania*, vol. 2018, 2018, pp. 7–9.

[12] L. Holzer and M. Cantoni, "Review of fib-tomography," *Nanofabrication using focused ion and electron beams: Principles and applications*, vol. 559201222, pp. 410–435, 2012.

[13] A. M. Wink and J. B. Roerdink, "Denoising functional mr images: a comparison of wavelet denoising and gaussian smoothing," *IEEE transactions on medical imaging*, vol. 23, no. 3, pp. 374–387, 2004.

[14] B. Justusson, "Median filtering: Statistical properties," *Two-dimensional digital signal prcessing II: transforms and median filters*, pp. 161–196, 2006.

[15] A. Buades, B. Coll, and J.-M. Morel, "Non-local means denoising," *Image Processing On Line*, vol. 1, pp. 208–212, 2011.

[16] D. Zhang and D. Zhang, "Wavelet transform," *Fundamentals of image data mining: Analysis, Features, Classification and Retrieval*, pp. 35–44, 2019.

[17] S. Yu, J. Ma, and W. Wang, "Deep learning for denoising," *Geophysics*, vol. 84, no. 6, pp. V333–V350, 2019.

[18] U. J. Botero, F. Ganji, D. L. Woodard, and D. Forte, "Automated trace and copper plane extraction of x-ray tomography imaged PCBs," in *2021 IEEE Physical Assurance and Inspection of Electronics (PAINE)*. IEEE, Nov. 2021. [Online]. Available: https://doi.org/10.1109/paine54418.2021.9707715

[19] G. Kanagachidambaresan, "Introduction to kicad design for breakout and circuit designs," in *Role of Single Board Computers (SBCs) in rapid IoT Prototyping*. Springer, 2021, pp. 165–175.

[20] L. Ding and A. Goshtasby, "On the canny edge detector," *Pattern recognition*, vol. 34, no. 3, pp. 721–725, 2001.

[21] N. Jessurun, O. Paradis, A. Roberts, and N. Asadizanjani, "Component detection and evaluation framework (cdef): A semantic annotation tool," *Microscopy and Microanalysis*, vol. 26, no. S2, pp. 1470–1474, 2020.

[22] F. Aurenhammer and R. Klein, "Voronoi diagrams." *Handbook of computational geometry*, vol. 5, no. 10, pp. 201–290, 2000.

[23] J.-C. Chen, "Dijkstra's shortest path algorithm," *Journal of formalized mathematics*, vol. 15, no. 9, pp. 237–247, 2003.

[24] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 658–666.

[25] C. A. Rogers, *Hausdorff measures*. Cambridge University Press, 1998.