Noise removal in polygonal maps using a geometric Descriptor Framework

Shmuel Rippa | IMVC 2015

24 March 2015

About Us


orbotech.
Virtually Every Electronic Device in the World is Produced Using Orbotech Systems
Leading supplier of digital production and process solutions

All of the world's major electronic brands rely on Orbotech equipment in the manufacturing process

Global Presence

- 50 offices worldwide close to customer sites
- 2200 employees
- Headquarters and main R&D in Israel
- 500 scientists and engineers
- Revenues in 2014 - $583M

Our Automatic Optical Inspection (AOI) Systems

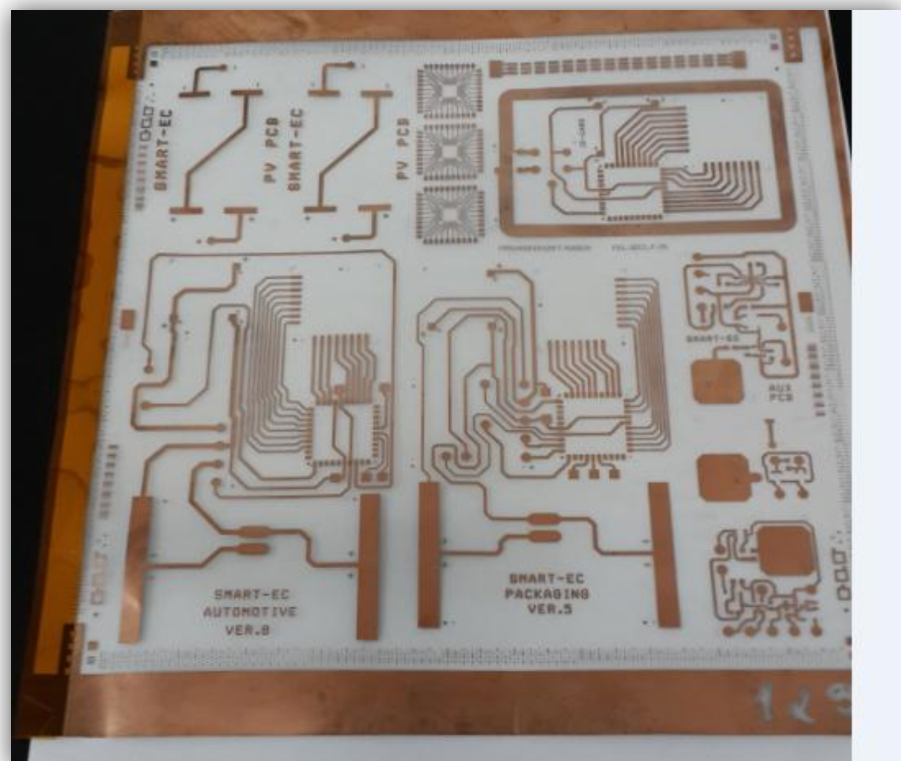
Find defects in Printed Circuit Boards (PCBs) and Flat Panel Displays (FPD) in high speed (< min)

Problem complexity ~ Find a grain of rice in central park … in 30 seconds with (nearly) full detection and very few false positives

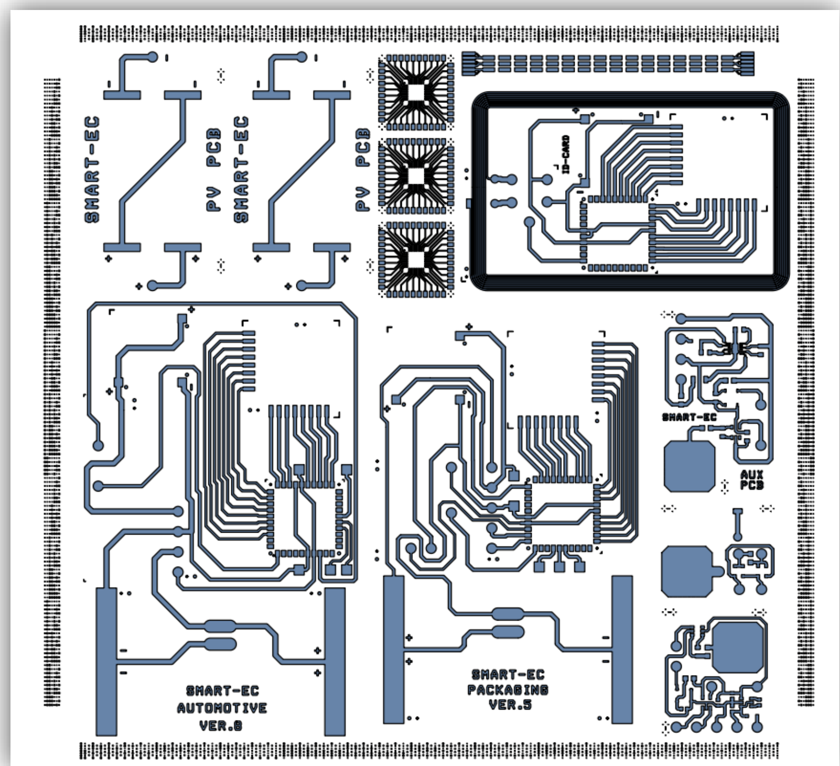
Background and Problem Description

PCB Inspection – A Plausible Approach

Detect and report discrepancies between the design and the inspected panel

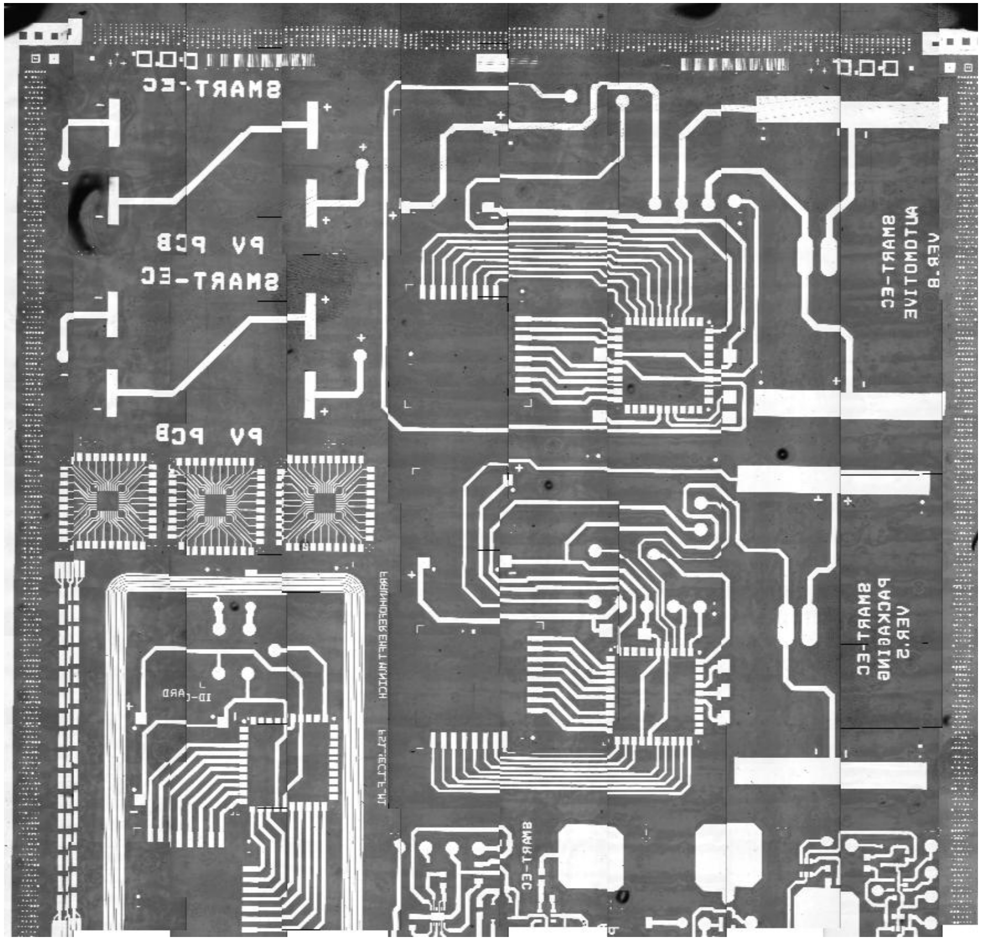


Panel to be inspected for defects



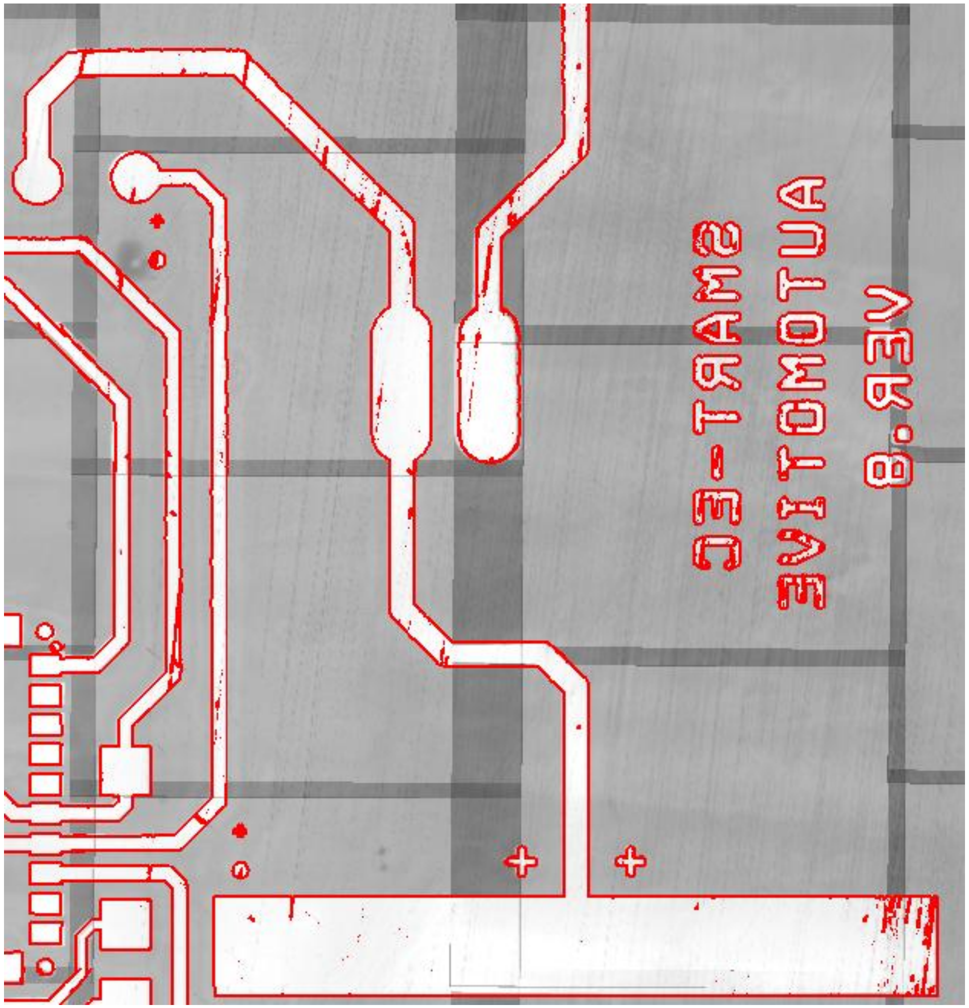
Design drawing of panel – CAD file

Step 1 – Grab an image of the board



Grab an image

Step 2 – Extract edges

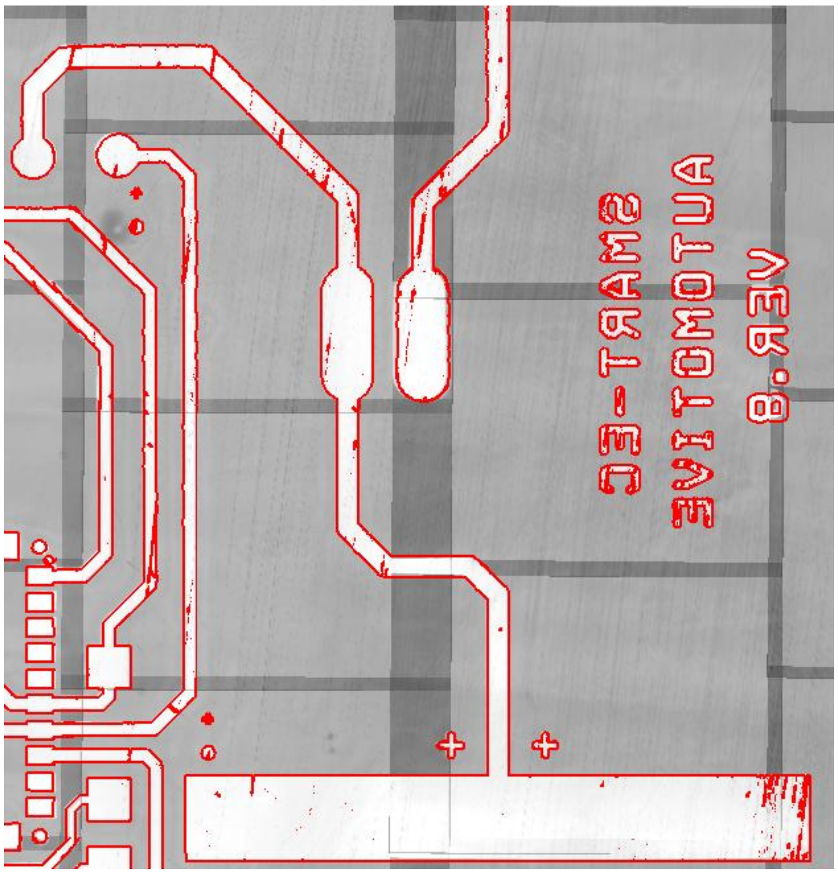


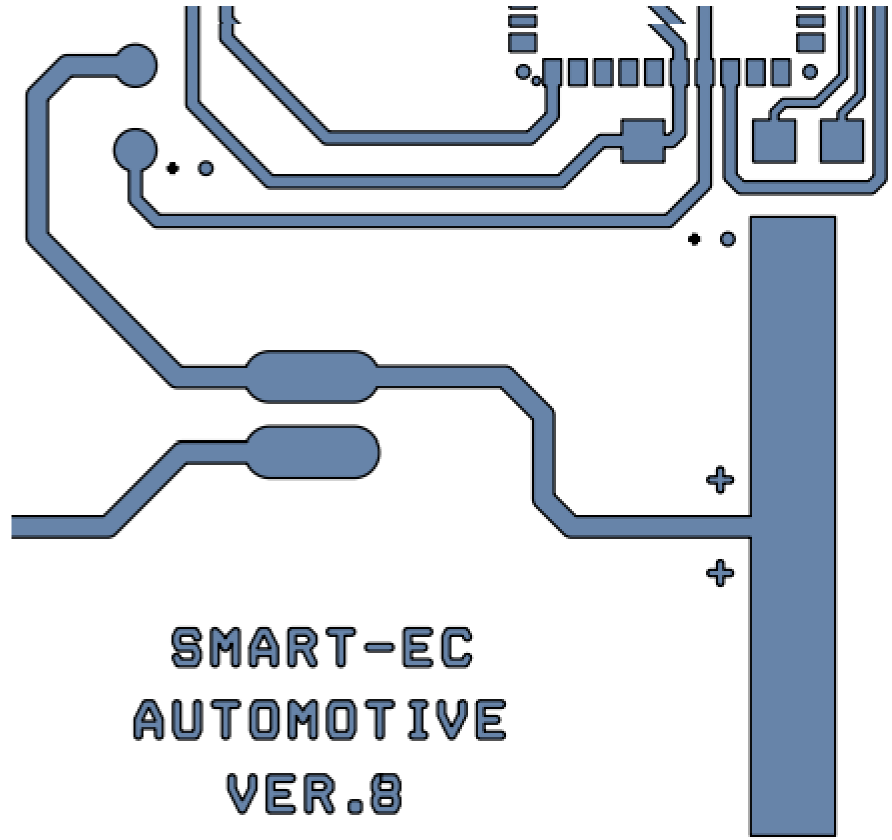
Grab an image

↓

Extract edges

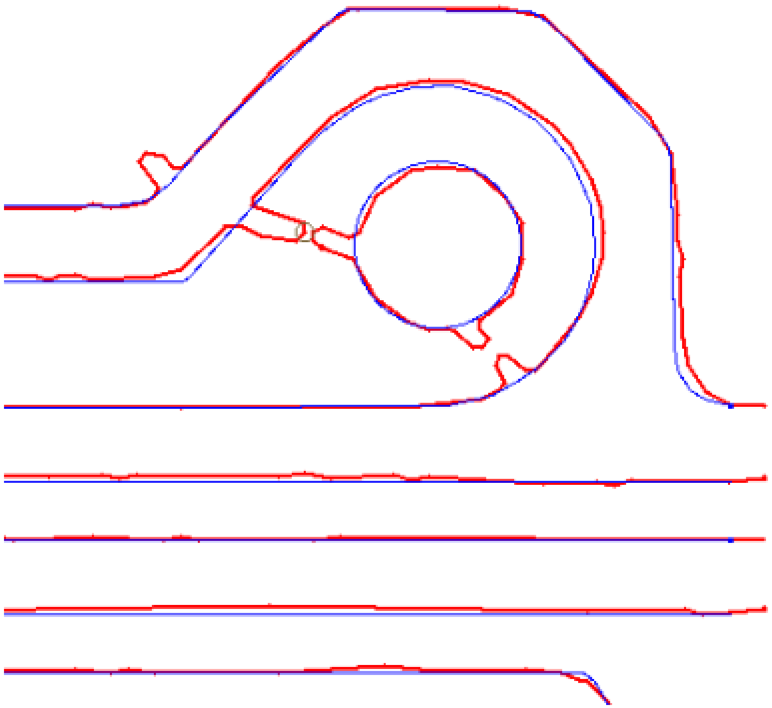
Step 3 – Align edges to the design

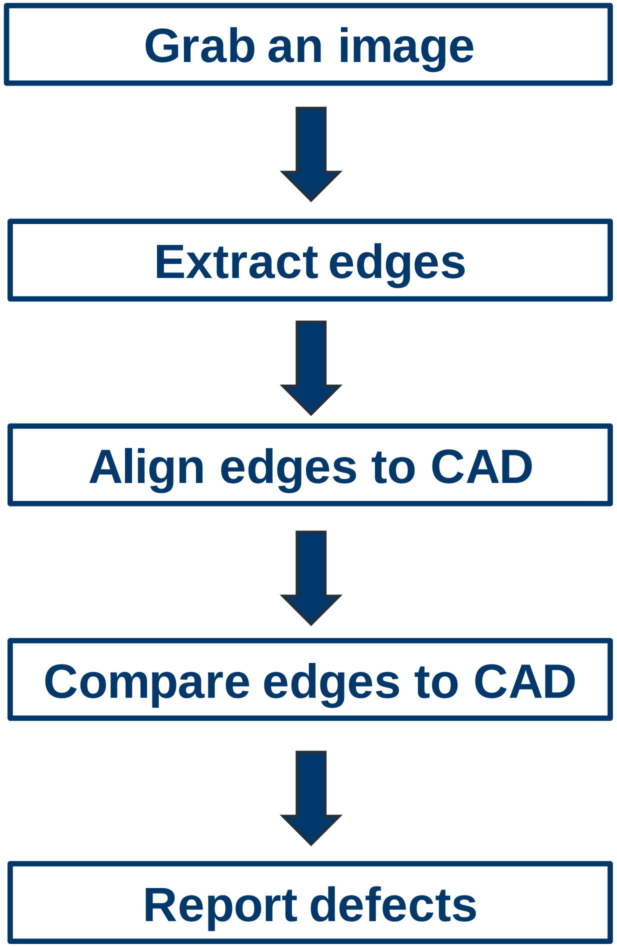




Design drawing of panel

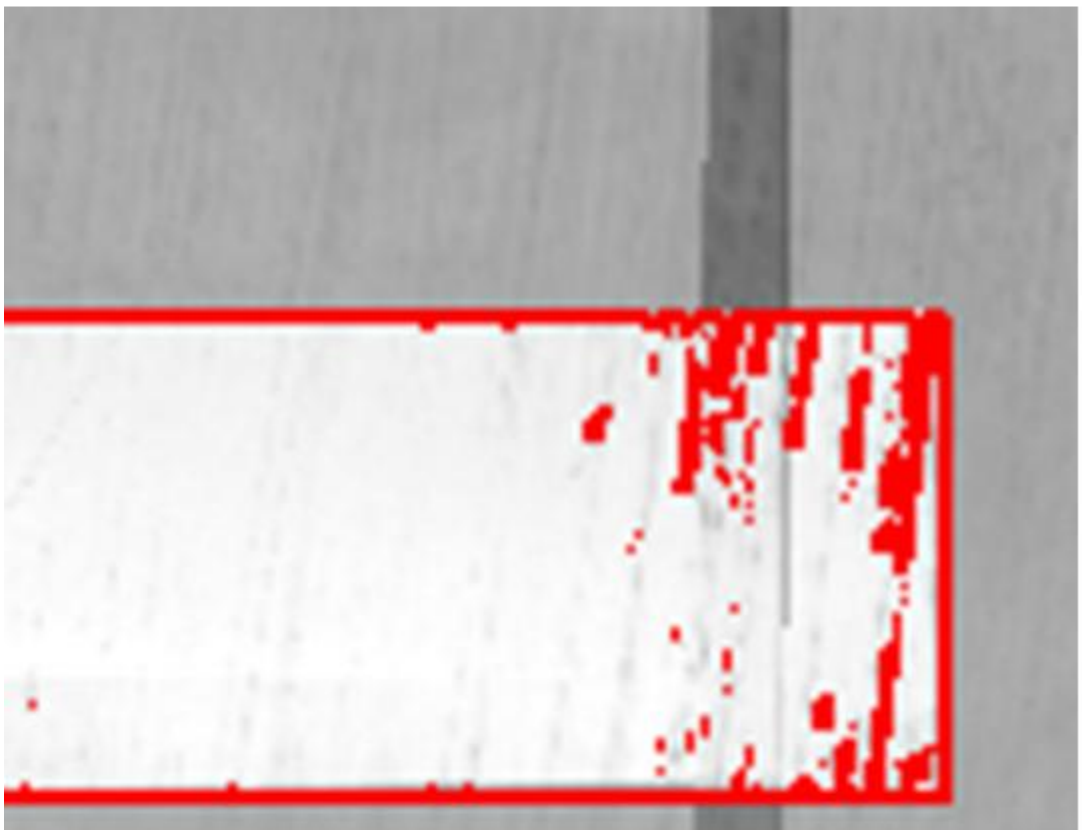
Step 4 – Compare edges to design & report defects

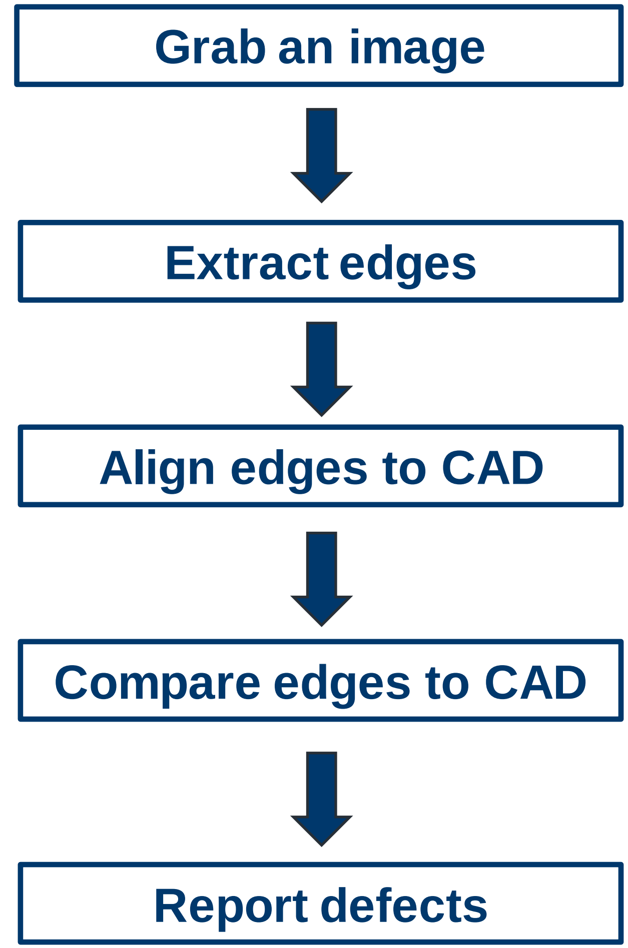




The problem

Noise on edges can result in failure of the alignment procedure, killing the whole detection pipeline



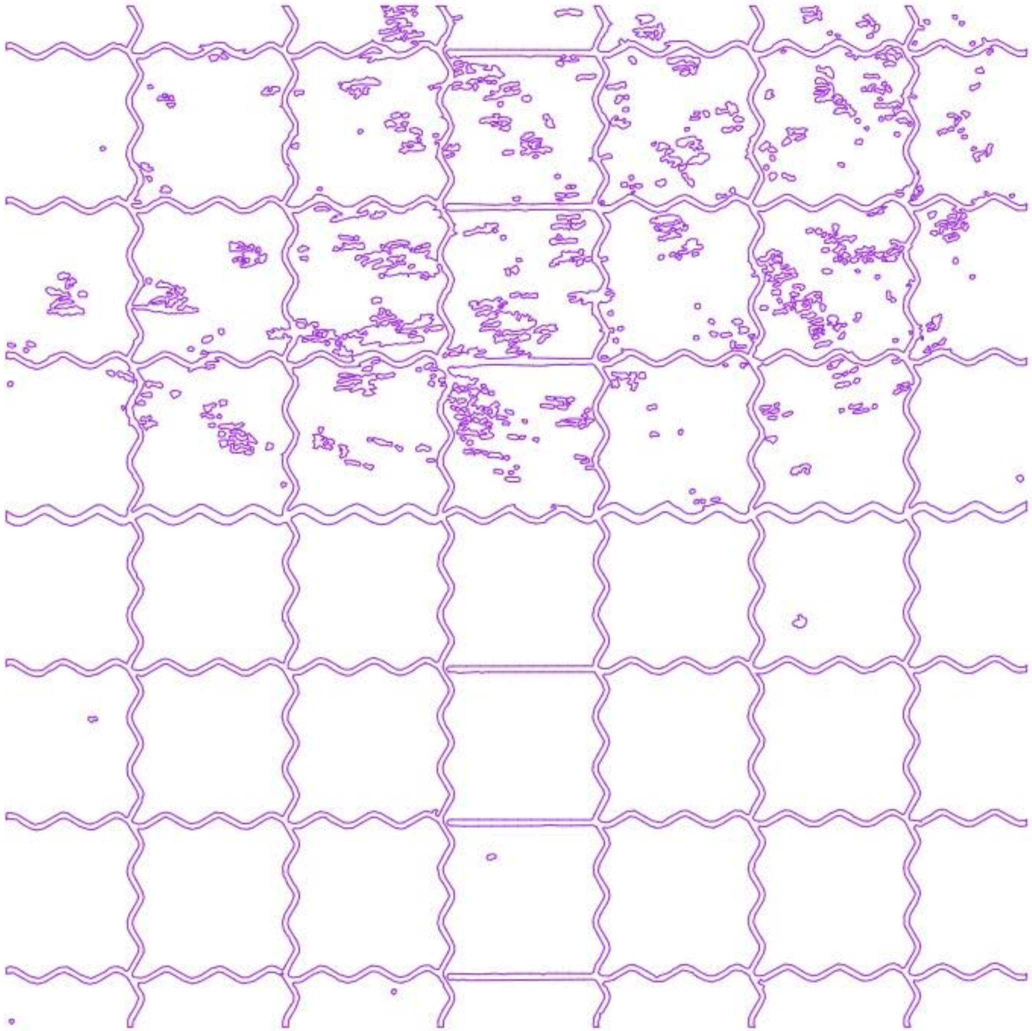


A Solution

When noise looks like noise …

It is easy for humans to filter out the noisy edges even without any reference

Because, well, it just looks like noise



So why not extract some features and deploy our favorite anomaly detection procedure?

Extracting features from PCB polygonal maps

Natural approach is to extract features from the original contours


Image taken from http://www.danleach.net/

o limited FOV of the

ontours of equal
riptors for each of them

Tuning parameter: The selected arc-length

Need to be learned. We took arc-length==10

Geometric descriptors for patches of contours

This lecture focused on a class of linearity measures defined on open contours.

Definition:

Linearity measure of an open contour C is a function f(C) that:

1. **Always assumes a value in [0,1]**
2. **Attains its maximal value iff the contour C is a straight line**
3. **Is invariance under translation, rotation and uniform scaling**

Why does it make sense: Contours of a PCB board are very smooth, approaching linearity if viewed at the "right" scale for that board.

Note: The framework works with any type of descriptors defined on open contours

Linearity measures

The straightness index [1]:

$$0 \le \mathrm{si} = \frac{\|p_0 - p_1\|_2}{l(\mathcal{C})} \le 1$$

where p_0, p_1 are the two end-points of $\mathcal{C}$ and $l(\mathcal{C})$ is its arc-length.

The Problem: Measure is too crude – any closed contour will get a score of zero

A better linearity measure [2]:

$$0 \le \mathrm{dc} = \frac{\|p_0 - p_c\|_2 + \|p_1 - p_c\|_2}{l(\mathcal{C})} \le 1$$

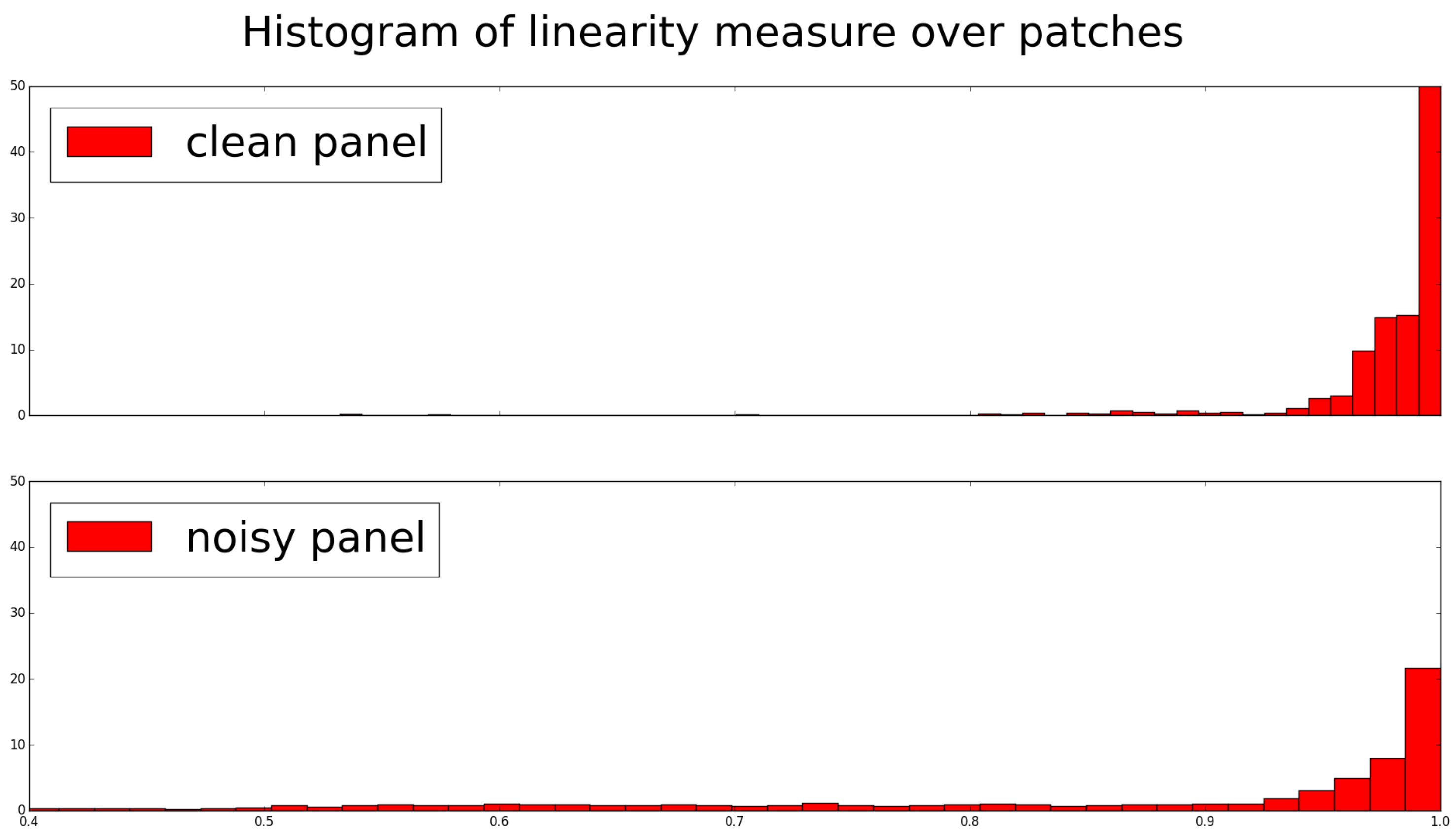
where p_c is the centroid of the points, namely the weighted (by length) average of the points of the contour.

(Image taken from [2])

Fig. 1 Five displayed curves (*solid lines*) have different linearities measured by $\mathcal{L}(\mathcal{C})$. The straightness index has the same value for all five curves

[1] "How to reliably estimate the tortuosity of an animal's path: straightness, sinuosity, or fractal dimension?" ,S. Benhamou (2004)|
[2] "Measuring Linearity of Curves" , J. Zunic, J. Pantovic and P.L. Rosin (2014)

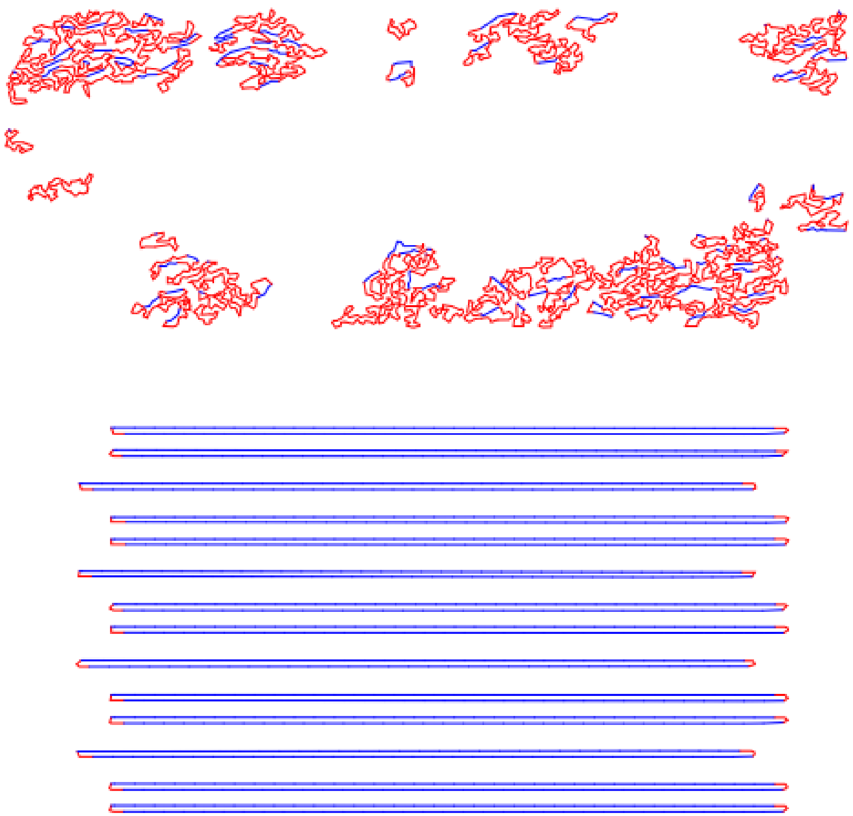
Computing the threshold for labeling patches as smooth or noisy

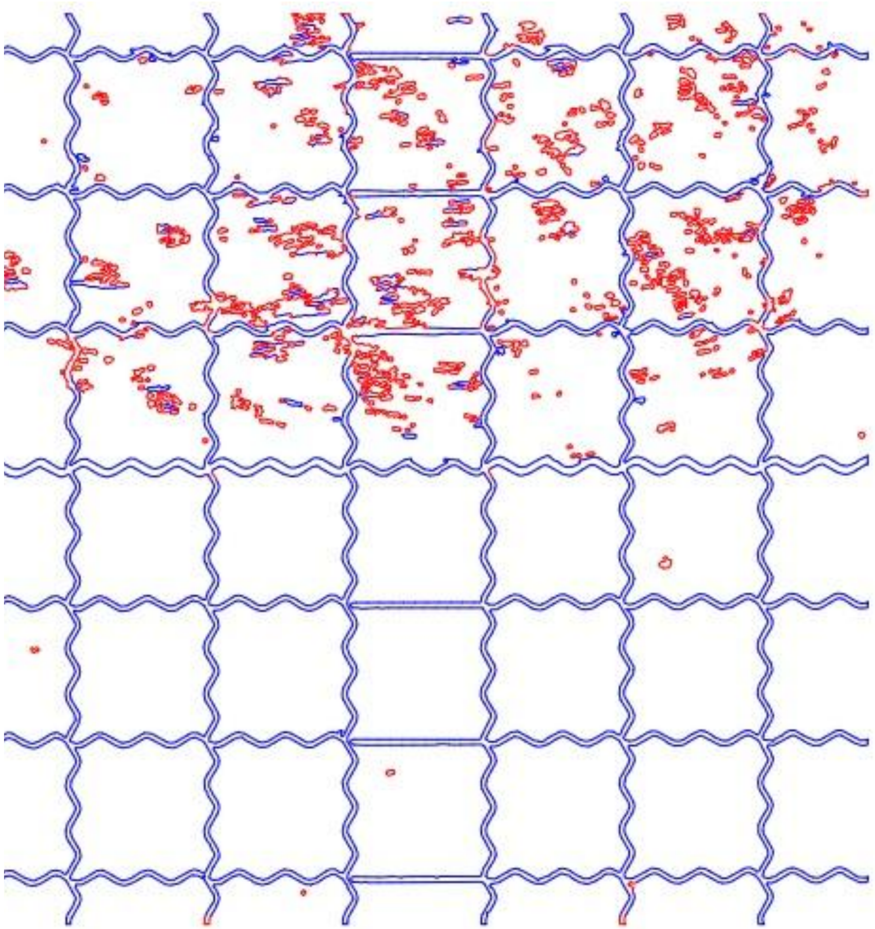


Tuning parameter: smooth/noise threshold

Need to be learned. We took threshold==0.9

And it works nicely …

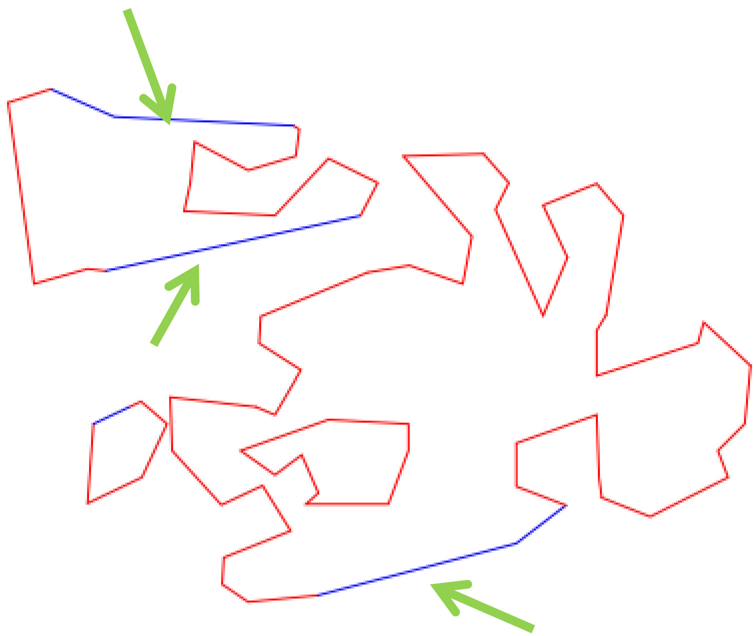


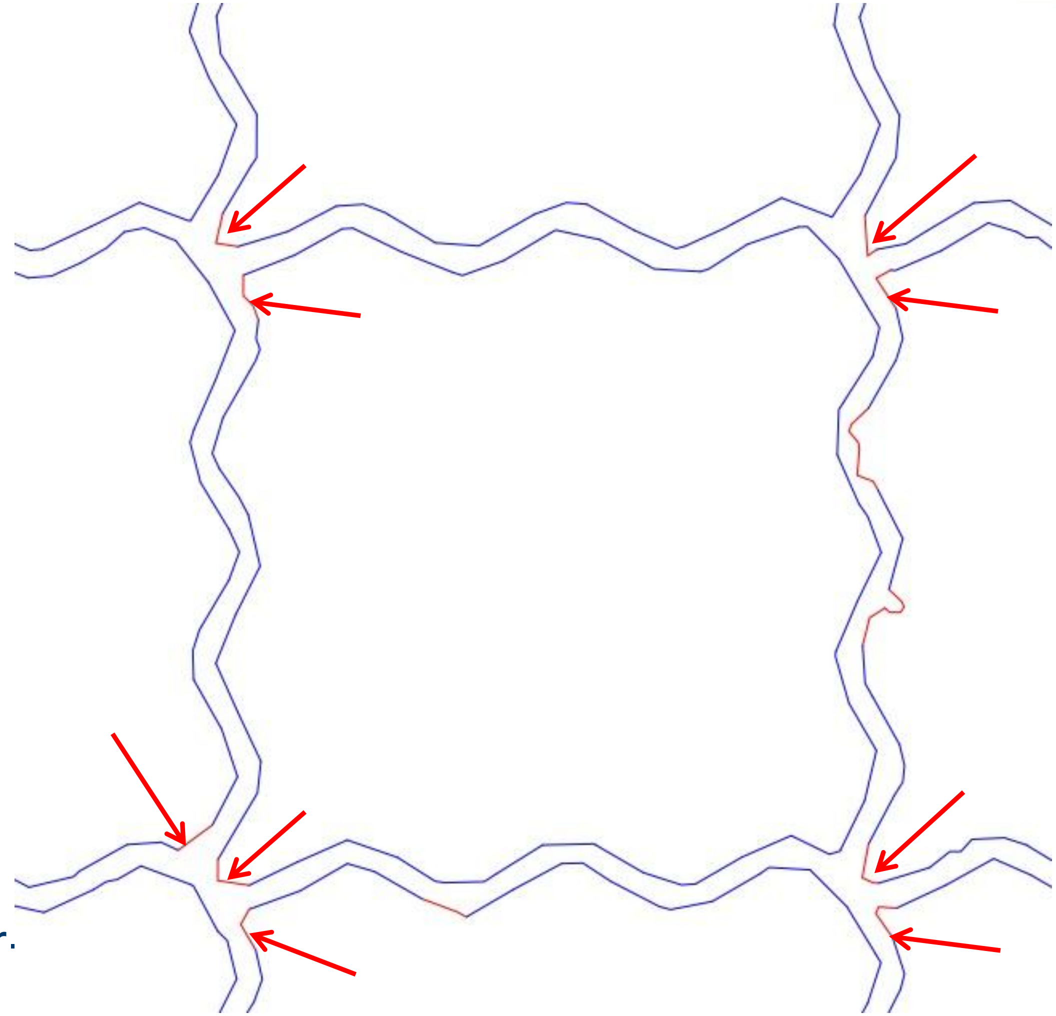


But does not solve the applicative problem

What seems to be the problem?

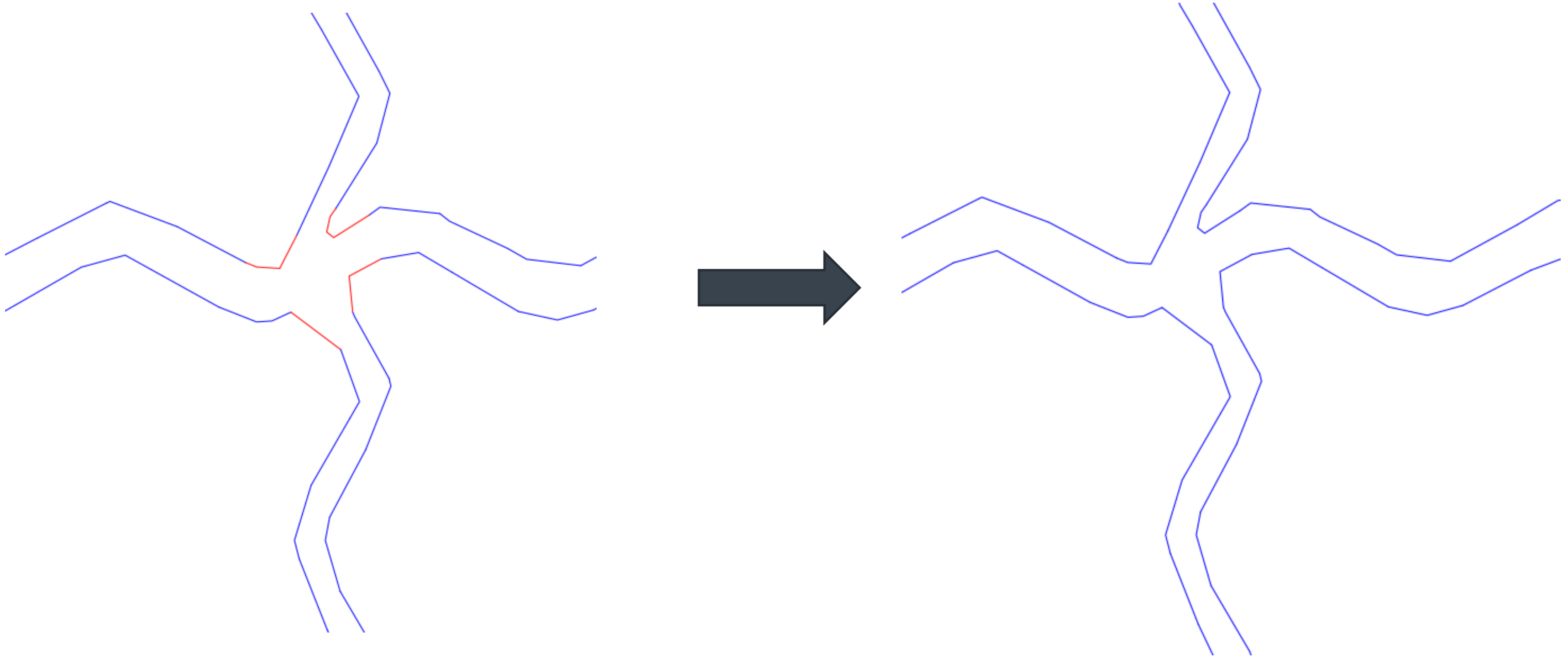
We label corners that are essential to the success of the registration as noise.





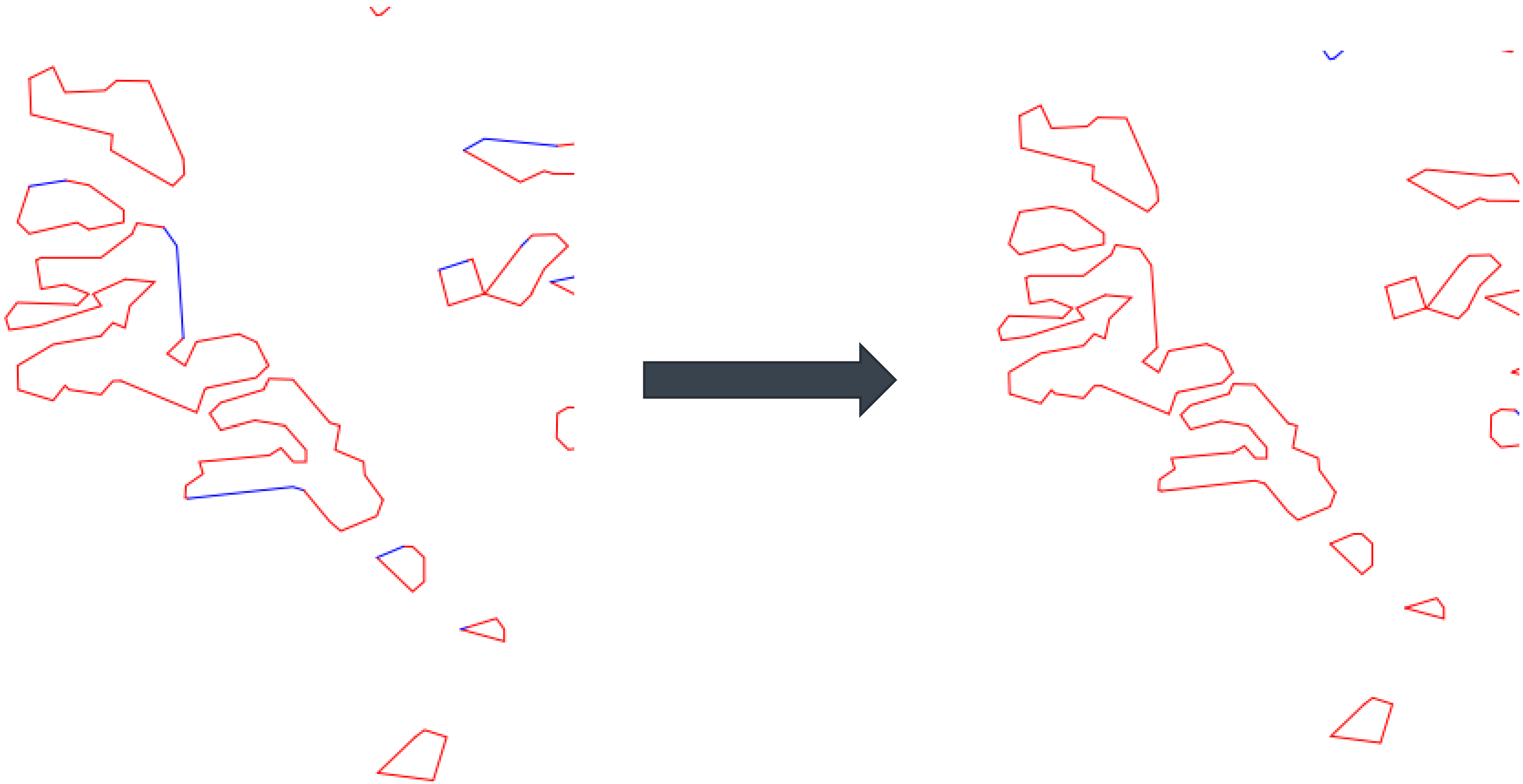
And we also label straight contour patches belonging to noisy components as smooth.

Adding semantic filtering: Re-label semantically smooth parts



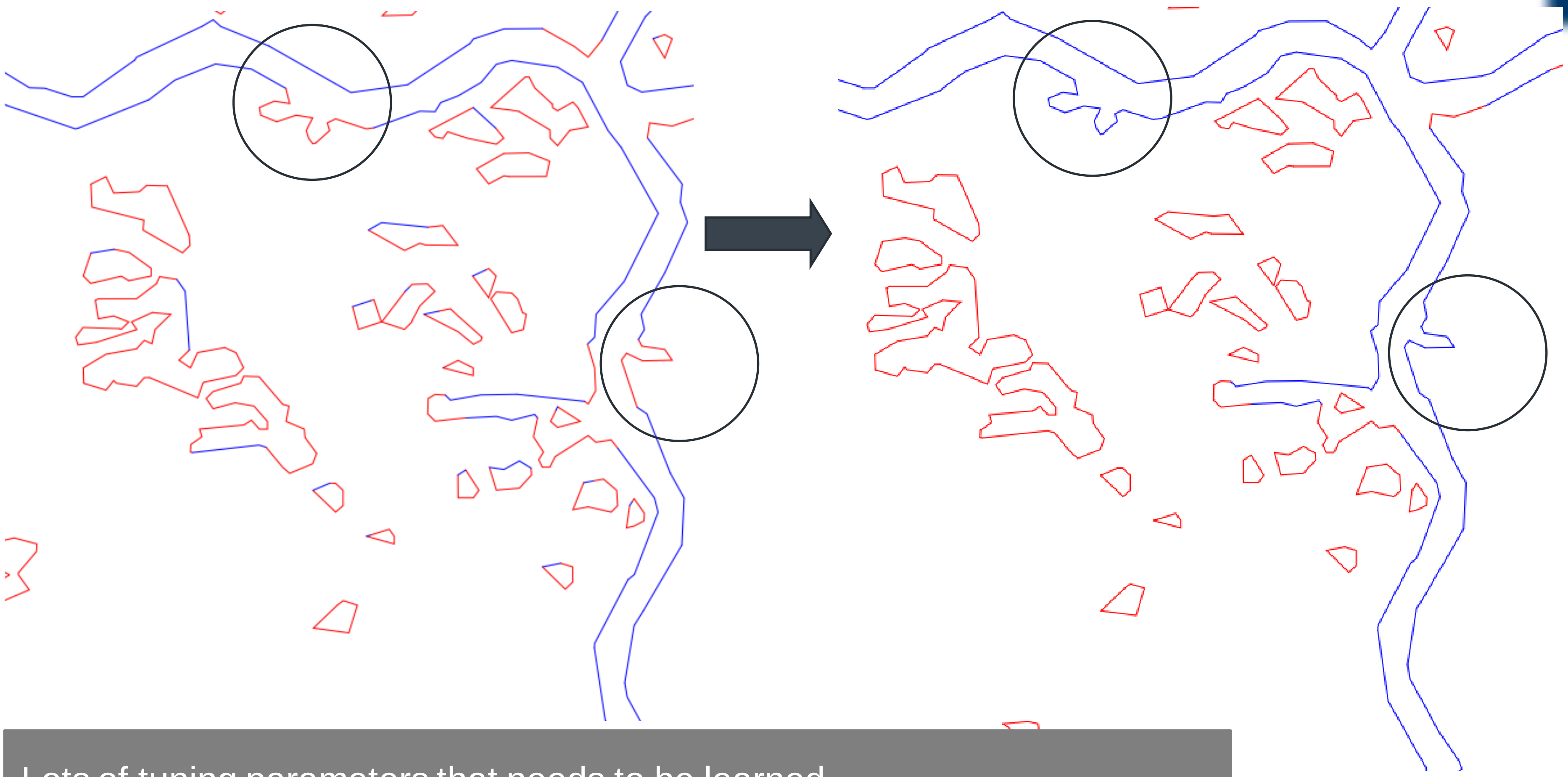
Re-label "noisy" parts that are semantically corners of a smooth path

Re-label semantically noisy parts



Re-label “smooth” parts that are semantically corners of a noisy path

Still, far from complete→ lot's of place for improvements



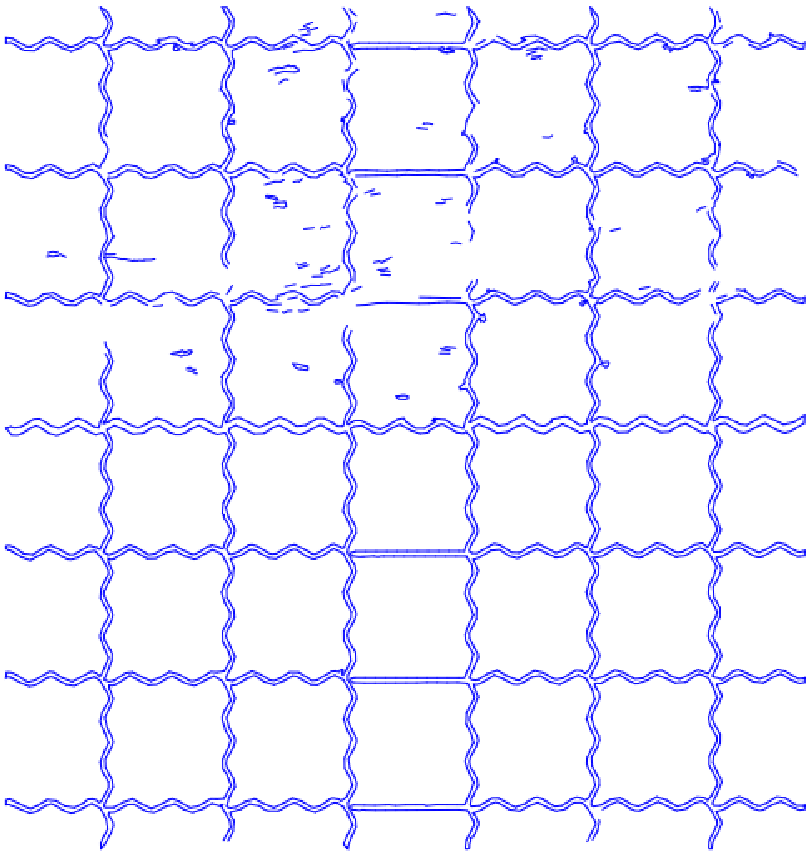
Lots of tuning parameters that needs to be learned

We took simple rule: If arc-length of patch-string smaller than two surrounding patch-strings → label reversed

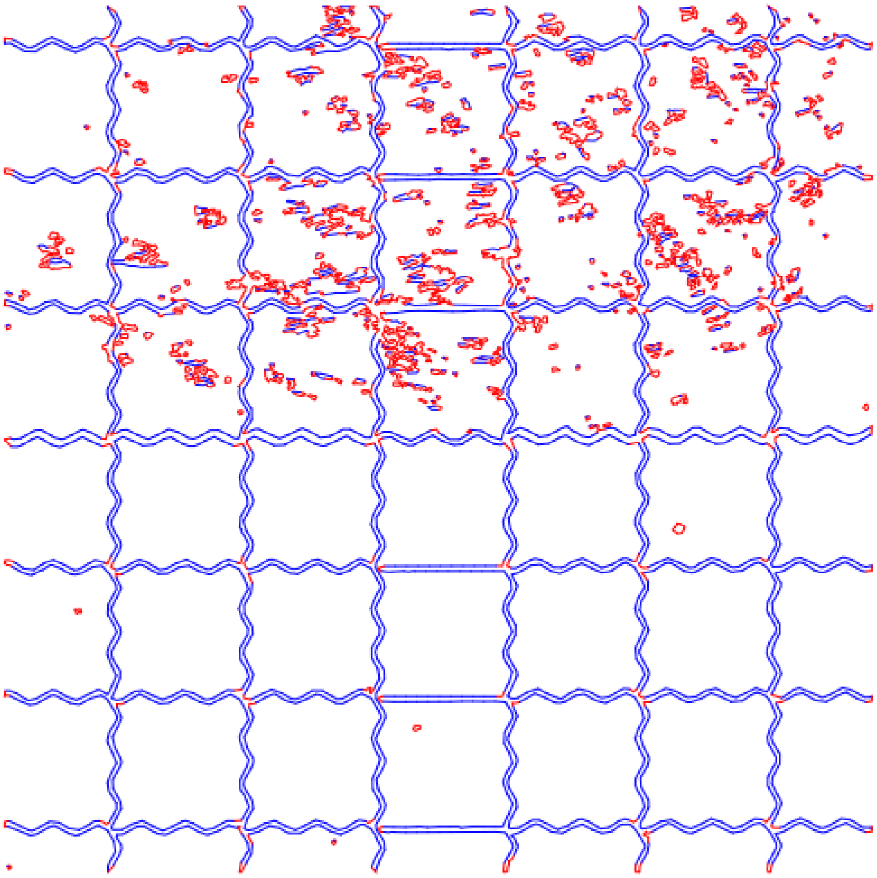
orbotech. The Language of Electronics

Final result

Map after semantic filtering



Labeled polygonal map



Summary: The GDF computational pipeline

Overlapping or non-overlapping patches	**Decompose polygonal map into patches of equal arc-length**
Many descriptors to choose from	**Compute one of more descriptors for each patch**
Can experiment with supervised/un-supervised methods	**Classify each patch as either smooth or noisy based on computed descriptors**
Can use domain specific knowledge to enhance robustness	**Use semantic filtering to re-label patch-strings**

Possible extensions

- Add other scoring functions
  - Ellipses, squares, circles,…
- Add more descriptors
- Use more sophisticated anomaly detectors
- Robust methodology for tuning the hyper-parameters
- Enhance framework to handle tasks other than anomaly detection

orbotech
The Language of Electronics

THANK YOU