Assignment 2:

Vegard Aaberge
Industrial Automation
Mechanical Engineering
20/09/2013

Introduction

In this assignment the task is to create a program that can check a PCB if it is ready for shipment or not. The idea is to make the process automated.

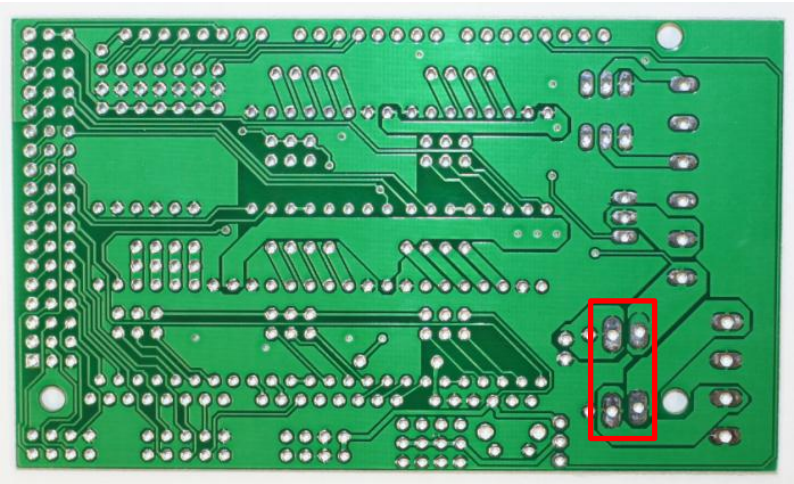
His program uses Matlab, and it starts off by importing the reference images of the PCB board, and finds the reference values for the drilled holes, conductive pathways and the area for the fuse pads.

After the program is finished with the reference image, then all of the other images are loaded and the program compares if it is similar to the reference values. If it is similar, then it will output that it is correct. If it is incorrect, then it will output it is incorrect and how many are incorrect.

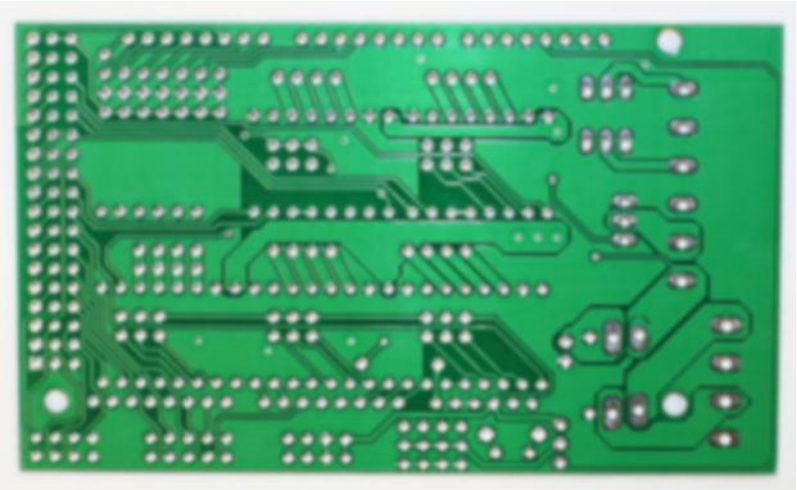
Processed Image

GIMP was used to create a processed image to easier calculate the area of the fuse pads. This was done, because the capabilities of GIMP in image processing are much stronger than matlab, and gave a very accurate solution.

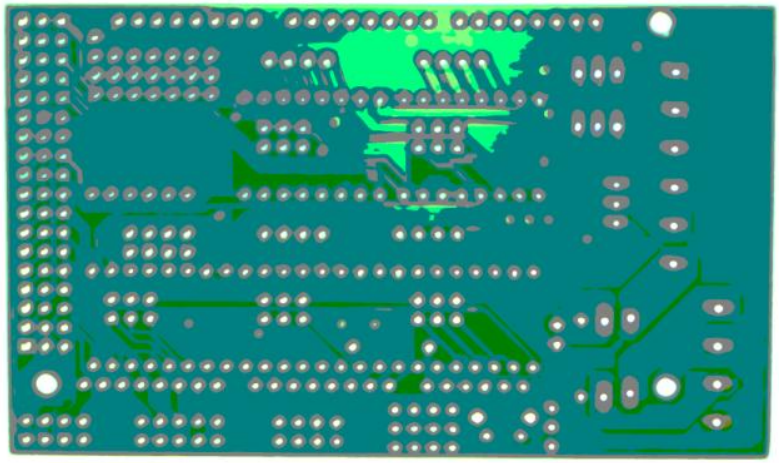
Input Image:



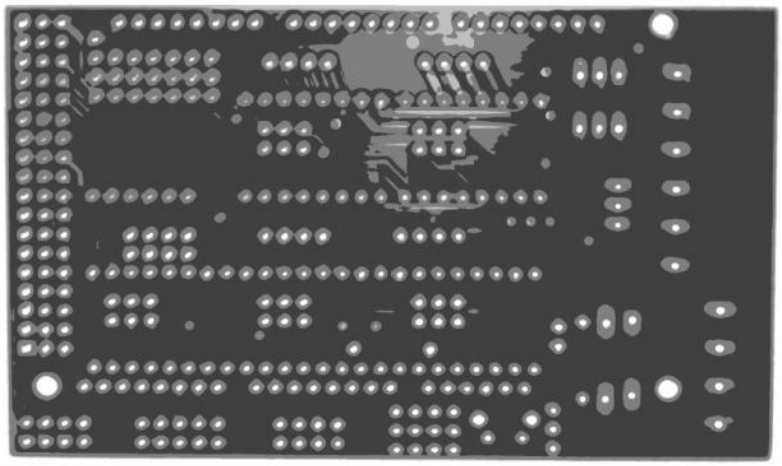
A Gaussian Blurr (kernel: 20 by 20) was done to remove noise from the image.



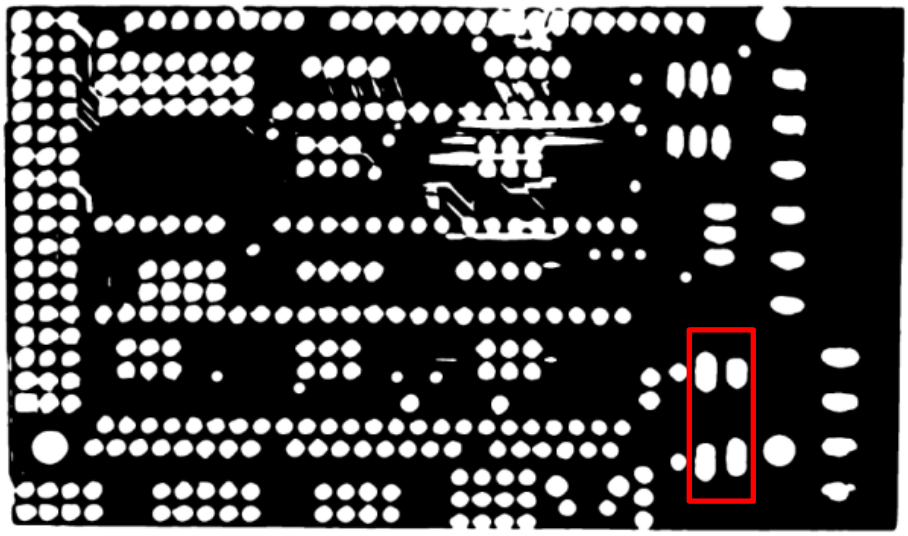
Then the image was Posterized to three colours. Three colours were able to detect the difference between the pads, the wires and the green background.



Light was desaturated, to remove colours from the image.



Finaly the image was posterized with two colours, making the whole pad area white.



The image can now be imported into matlab, and we can use regionprops to detect the white circles. The number of pixels in each object can be added together to find the area of each pad.

This process needs to be done for all of the images that are going to be processed.

Finding the tolerance of centre locations:

From the reference image the length of the PCB was found to be 2461 pixels and the height was 1454 pixels. The dimensions of the PCB board are 104.14 x 61.468 mm. So for the length there is 23.63 pixels per mm and 23.65 pixels per mm for the height.

The tolerance is +-1mm. To be safe, the holes need to be within a tolerance of 23 pixels.

It was chosen to use the reference image for the tolerance, and not measure the distance from the edges, because the edge in the image is not a sharp border, and it is not defined where on the border it is measured from.

Do RefTask

This is the task first executed in the program. It will load the reference image, and figure out what the relevant dimensions are. It will find the number of pathways in the image, the centre locations, and the average area of the 4 reference fuse pads.

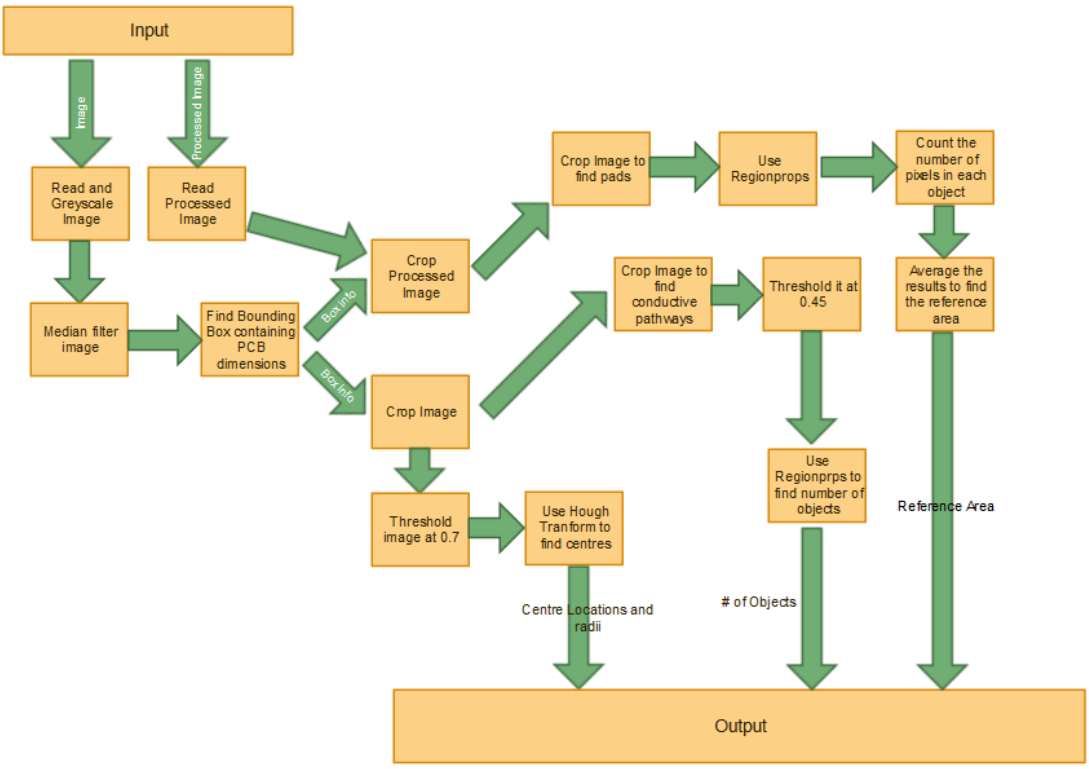
DoRefTask starts up by importing the images, greyscaling them and cropping away the whitespace before it checks for locations.

Regionprops measures properties in an image. Regionprops was used three times in this code. The first time it was used, was with the regionprops bounding box. That was used to find the boundaries of the PCB board, so the whitespace can be cropped away. It is very important to get rid of the whitespace to get correct position measurements.

The second time it was used was to find the area of the fuse pads. First the area of interest was cropped out. After regionprops was used, then the number of pixels in each object was counted. The result for the 4 fuse pads was averaged and becomes an output of the RefTask function.

The third time it was used was to find the conductive pathways. The area of Interest was cropped out, it was thresholded to highlight the conductive pathways, and then regiongroups was used to determine how many objects there are.

The centre locations were found by thresholding the image to get defined areas, and then using Hough transform. Finaly the results for centre locations, number of objects, and the reference area is outputted to the main program.



Do Task

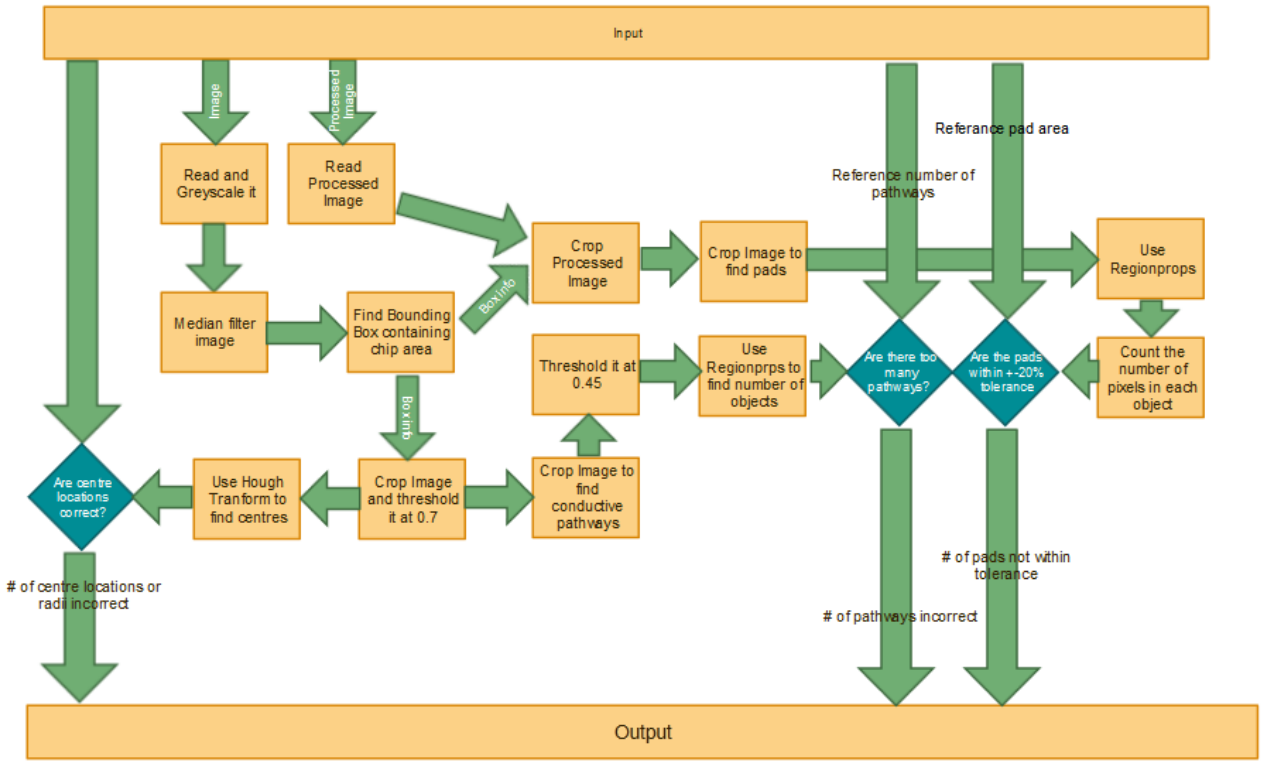
After the code has been executed by RefTask, then DoTask takes the outputs from RefTask, and checks for three things

1. Are the drill hole locations and radii within the 1mm tolerance?
2. Are any of the pathways broken?
3. Are the pads within the 20% area tolerance

The main difference between DoTask and DoRefTask is different inputs and outputs, DoTask has the three comparisons that do not exist in DoRefTask, and DoRefTask finds reference values while DoTask finds how many are broken.

DoTask uses the same method as DoRefTask to read the image, crop the image, to find how many conductive pathways, and to find centre locations. In the end it outputs

1. How many pathways are broken
2. How many holes are drilled incorrectly
3. How many pads are not within tolerance



Main Program

The main program can be seen below. The reference image and the processed image are inputted into DoRefTask. The outputs are

1. RefLenght – Number of conductive pathways
2. RefRadii and locations – The locations of the drilled holes
3. RefArea – The reference area for fuse pads

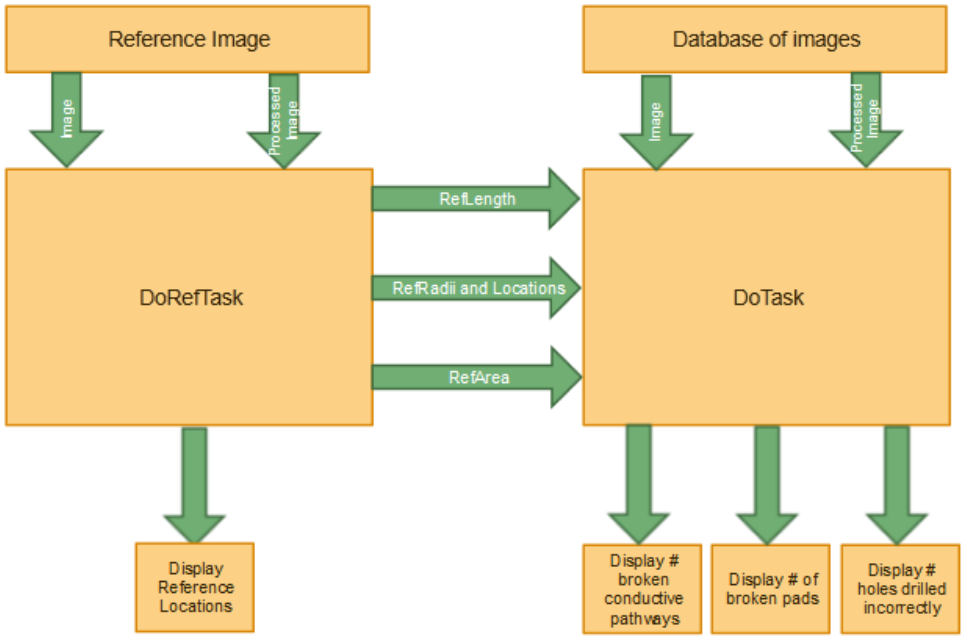
The Reference locations are shown to let the user aware of what drilled holes the program is talking about.

Database of images

Before inputting image and processed image data to DoTask, then all of the image data is saved in Image matrix to make management easier.

```
Image{1} = 'fuse pad damaged.jpg';
ProcessedImage{1} = 'fuse pad damaged Changed.jpg';

Image{2} = 'hole move 2.jpg';
ProcessedImage{2} = 'hole move 2 Changed.jpg';

Image{3} = 'hole move 1.jpg';
ProcessedImage{3} = 'hole move 1 Changed.jpg';

Image{4} = 'translated.jpg';
ProcessedImage{4} = 'translated Changed.jpg';

Image{5} = 'broken conductive path.jpg';
ProcessedImage{5} = 'broken conductive path Changed.jpg';

for i = 1:length(Image)
    [centers, radii, Exceed, Broken, offset] = DoTask(Image{i},ProcessedImage{i}, ReferenceArea, RefLength, RefCenters,RefRadii);
```

A for loop is used to execute the program till all of the images are evaluated. A further extension could be to allow it to collect data from a text file instead.



When DoTask is finished, then it outputs three results about the number of conductive pathways, the number of broken pads and the number of holees drilled incorrectly.

The results are outputted to the user. The output can be seen below.

Output

```
>> Assignment2code

The reference centres is at
location 1: x-cordinate: 2109 y-cordinate: 1221
location 2: x-cordinate: 137 y-cordinate: 1210
location 3: x-cordinate: 2093 y-cordinate: 62

For fuse pad damaged.jpg
1 pad is broken
All the circuits paths are working
All the holes are drilled correctly

For hole move 2.jpg
All the pads are working
All the circuits paths are working
2 holes are drilled incorrectly

For hole move 1.jpg
All the pads are working
All the circuits paths are working
1 hole is drilled incorrectly

For translated.jpg
All the pads are working
All the circuits paths are working
All the holes are drilled correctly

For broken conductive path.jpg
All the pads are working
1 circuit path is broken
1 hole is drilled incorrectly
```

Visual Noun

To differentiate PCBs from each other, I would recommend stamping each PCB with a black barcode. The barcode can contain information such as factory number, and date of production. To read the barcode, then the image needs to be greyscaled and thresholded. Then the data of how it changes colour can be read across a line and converted into the output.

The method is good, because it is not necessary to create a new program to understand the barcode. A normal barcode reader can do the job just fine. A disadvantage is that it may make the PCB look a little ugly. A potential way to solve that is by using a Qr barcode, which is a small square instead of black stripes. A potential disadvantage is that it will be harder to read.