Project Update : AI & ML in AM

Abhishek Njayakkatt Salilkumar (MM19B017)
Sathiya Murthi Sankaran

Guide - Dr. Murugaiyan Amirthalingam

Date - 10 / 10 / 2022

Title : Defect Prediction in Additively Manufactured components using AI-ML image analysis

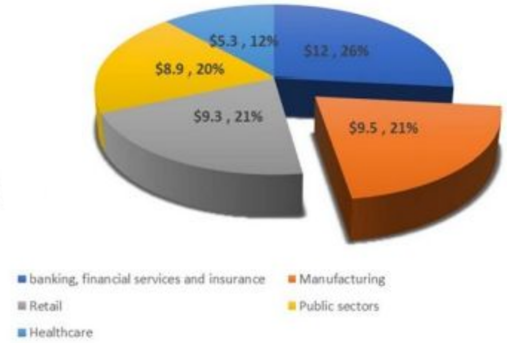
Objective : Detecting & Classifying defects in Additively Manufactured components by studying cross section images using Machine Learning Algorithms & correlating process parameters that contribute to these defects

Tasks -

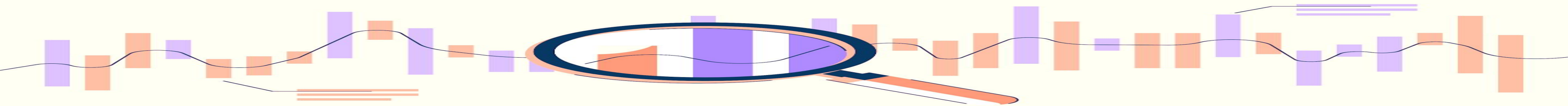
- Using **image analysis** tools to classify the defects from Optical and SEM images of Laser Powder Bed Fusion process
- **Front end creation** where the user could input a folder which would then return the defect percentage and the number of defects in the folder
- **Training a model** to predict the defect and classify it into Blow holes, Lack of Fusion and Grain Boundary Cracking
- Correlating with the **processing parameters** and the defect percentage and location

Overview

- Lots of opportunity of using ML and AI in AM, but as of now use is scarce
- AI could find use in the areas of process optimization, design correlation, design improvement, defect detection and microstructural design, but as the data now is not perfect nor reliable, it possess as a hurdle and its availability is also a cause for concern
- Reliable data collection, storage and sharing is paramount in development of ML algorithms for AM.



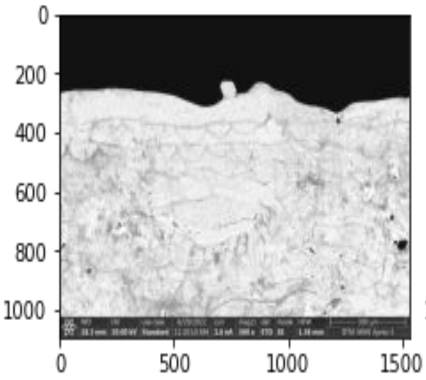
- Defects detection, the ability to create a 3D image of the build during the build or ability to monitor the microstructure and grain orientations are some examples that are still not fully explored.
- The project aims at incorporating ML algorithms to automatically compute the defects in a given image and correlate it with different parameters that contribute to these defects.
- For this, the samples images as well as the processing conditions was taken into consideration to develop the output.

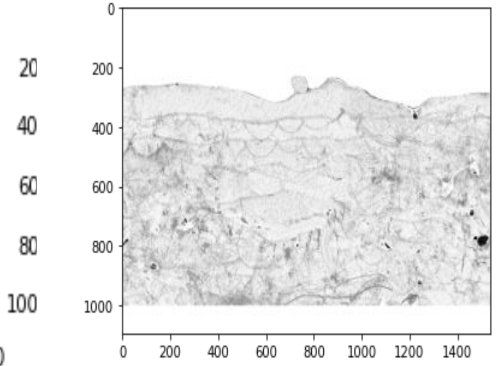


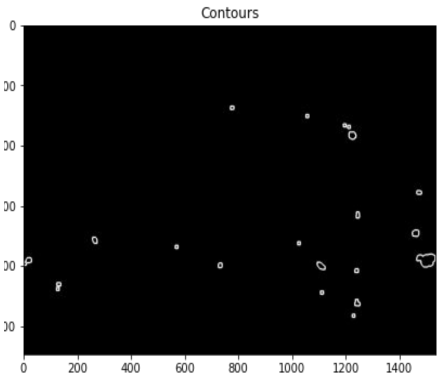
Outline of Project

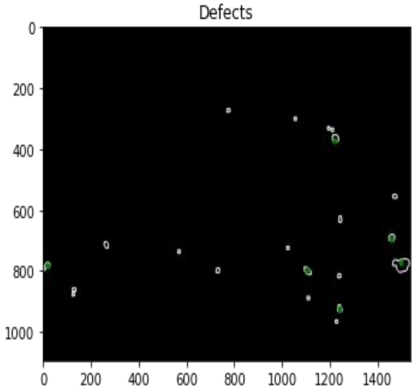
Data Collection → Image Processing → Contour Detection → Area Estimation → Plotting Defects → Cumulative Results

Data Collection	Image Processing	Contour Detection	Area Estimation	Plotting Defects	Cumulative Results
Optical and SEM images	Using OpenCV Python	Using Pixel Intensities	For Thresholding	Visual representation of total defects	Output for all the images in a given folder









Methodology Followed

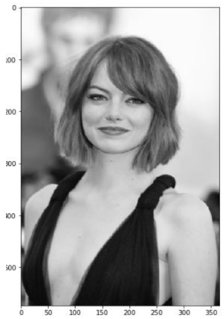
1. Thresholding & Edge Detection
2. Algorithm to detect whether the contour would describe a circle or if it was irregular.
3. Calculating Area of Contours
4. Input Folder and Plotting Results

Image Processing

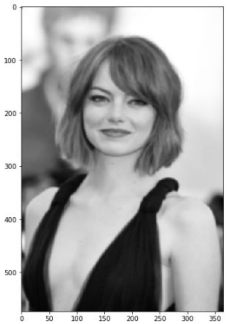
- Converting to B/W
- Gaussian Blur - to reduce noise
- Binary Thresholding
- Canny Edge detection :
    - Noise reduction;
    - Gradient calculation;
    - Non-maximum suppression;
    - Double threshold;
    - Edge Tracking by Hysteresis
- Image Dilation

```python
def process(img, show = False) -> dict:
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    img = cv2.GaussianBlur(img, (3,3), 0)
    ret,thresh = cv2.threshold(img,70,255,cv2.THRESH_BINARY)
    edges = cv2.Canny(image=thresh, threshold1=100, threshold2=200) # Canny Edge Detection
    img_dilation = cv2.dilate(edges, None, iterations=4)
    contours, hierarchy = cv2.findContours(img_dilation, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    contour_img = np.zeros(np.shape(img))
```

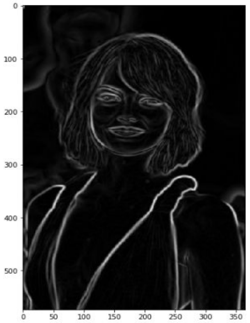
Steps in Canny Edge Detection



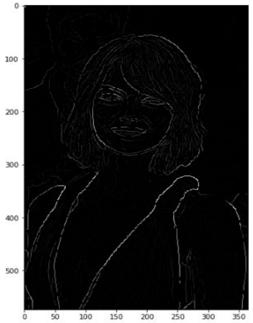
Given Image



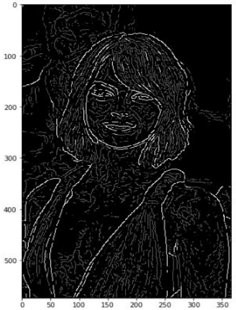
Noise Reduction



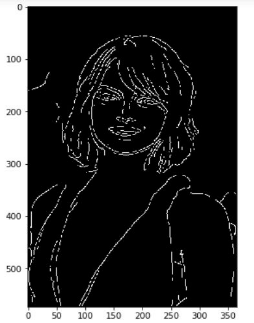
Gradient Calculation



Non-Max Suppression

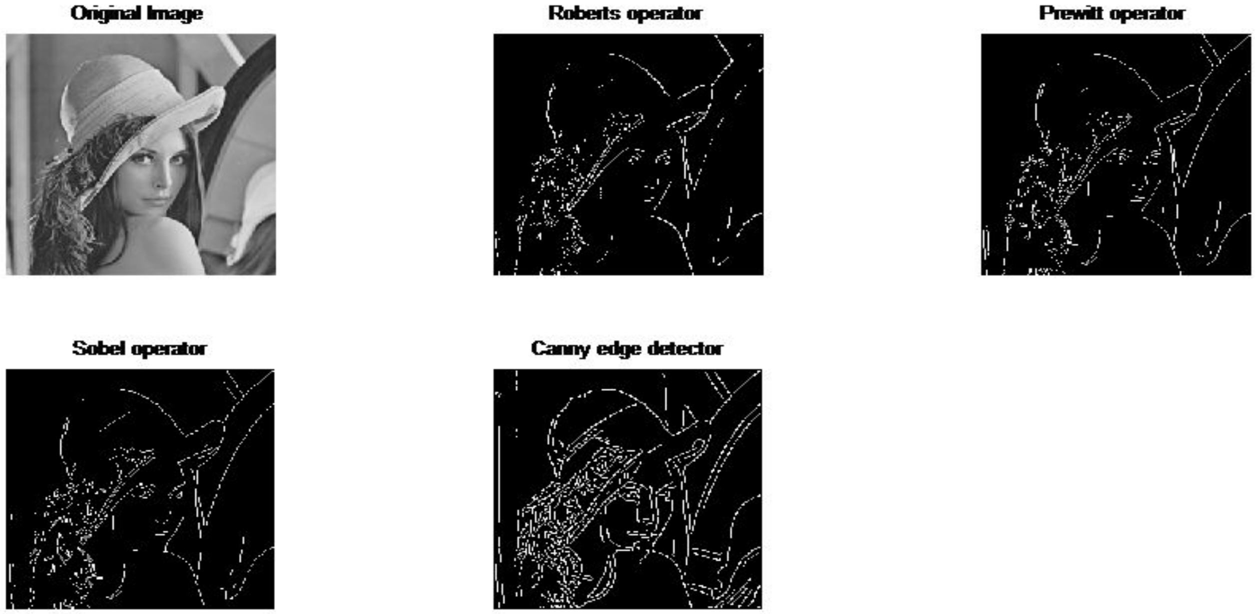


Double Thresholding



Hysteresis

Why Canny Edge Detection?



Circle Detection

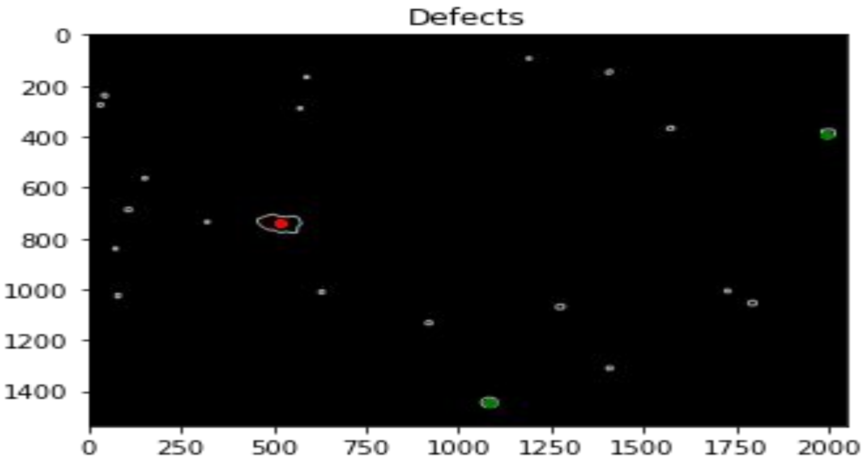
- Setting an area threshold
- Finding the median of the individual contours
- Calculating the standard deviation
- Applying a threshold on the standard deviation
- Checking if the deviation from median to edges is within the limit

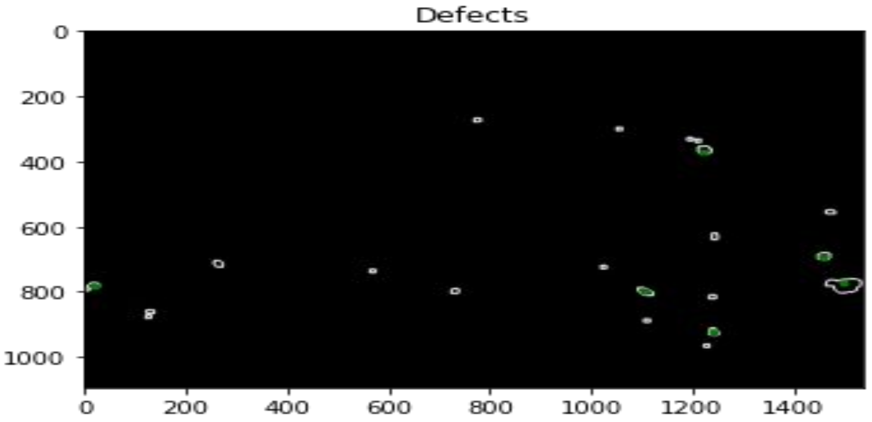
```
if (area > area_thresh):
    m = ((np.sum(contour, axis=0))/len(contour)).squeeze()
    median.append(m)
    contour_thresh.append(contour)
    cnt = contour.squeeze()
    d = []
    for a in cnt:
        d.append(dist(m, a))
    mean_d.append(np.mean(d))
    std_d.append(np.std(d))
```

Note - Rejected Hough Transform Circle Estimation, due to time complexity and edge points from non-target-shape objects reduce the detection quality.

Individual Plots

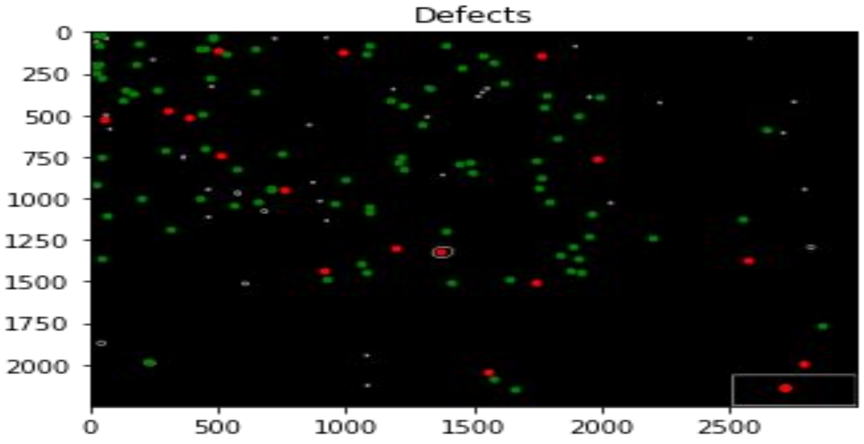
- Separate models for Optical and SEM images
- Plots visually represents the defects and the contours along with the calculated median
- Contours having area less than the threshold is removed

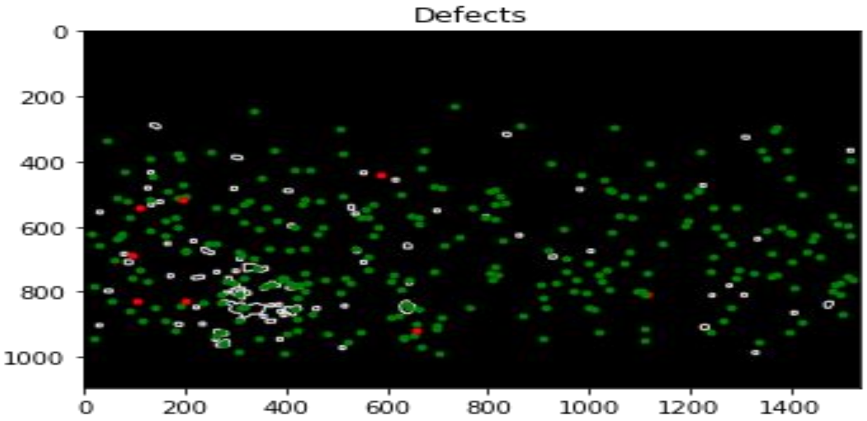




Cumulative Plots

- Consider Scale as constant throughout the folder
- algorithms returns a plot having all the set of defect locations
- Determines which is blow hole and which could possibly be a Lack of Fusion
- Gives a general localization of the defects





Individual Output

- Total defect count, Lack of fusion count as well its area percentage and Blow hole count along with its area percent is calculated
- For SEM images, primarily the defect observed was LOF so only that was considered.


```
{'A1': {'bottom.jpg': {'def_count': 5,
   'lof_count': 1,
   'lof_area_percent': 1.3209925925925925,
   'bh_count': 4,
   'bh_area_percent': 0.06774074074074074},
  'bottom_0002.tif': {'def_count': 0,
   'lof_count': 0,
   'lof_area_percent': 0,
   'bh_count': 0,
   'bh_area_percent': 0},
```

Cumulative data

- The total number of defects in all the images combined along with what percent of area do these contribute to was also calculated and returned as output to the csv file
- The area percent was the average of all the images.

condition	def_count	lof_count	bh_count	lof_area_percent	bh_area_percent	n_images
A1	108	21	87	0.162057	0.054086	50
condition	def_count	area_percent	n_images			
A	271	0.366162	31			

Going Ahead

- Using ML for defect classification by training a model
- Look into defect prediction of LOF and Grain Boundary cracks separately
- Correlating process parameters and the results obtained
- Incorporating manual addition of Scale

Thank you.