



## CORROSION DETECTION ON STEEL STRUCTURES USING IMAGE PROCESSING

<sup>1</sup> Vinothini K, <sup>2</sup> Abishek K, <sup>3</sup> Ezhil Selvan P, <sup>4</sup> Aswajith P

<sup>1</sup>Assistant Professor, <sup>2</sup>Student, <sup>3</sup>Student, <sup>4</sup>Student

<sup>1</sup>Department of Electronics and Communication Engineering,  
<sup>1</sup>Sri Ramakrishna Engineering College, Coimbatore, India

**Abstract:** This paper aims at the implementation of corrosion detection utilising machine algorithms. The system detects corroded image structure from objects. The basis of hue of the images has a significant impact on whether or not an object has rusted. The detection of corroded steel structures is captured through an RPI camera and the images are processed using the proposed Convolutional Neural Network algorithm to detect corroded images automatically through an open CV platform and processed using the Raspberry Pi Microcontroller. The processed image is then compared with the given data set, which is analyzed by class value. Here, by classifying the corroded images by numerical class values according to the extent of it affected by rust. The CNN algorithm is used to differentiate and train data according to a given input image. The experimental result shows a proposed system that achieves greater accuracy with minimum training loss when compared with the existing algorithms.

**Index Terms -** Raspberry Pi, Open CV, ReLu, CNN Algorithm.

### I. INTRODUCTION

The Latin verb corroded, which meaning "to wear away gradually," is the source of the term corrosion. The electrochemical response between an environment and a metal or alloy is described as corrosion in science. Along with mechanical loss via erosion, abrasion, or wear, it is one of the two primary factors in the corrosion of metal. Sometimes it can be stopped, but controlling it makes more sense from an economic standpoint. Consequently, engineers choose their materials assuming that the structure would be maintained during its useful life. The International Convention for the Safety of Life at Sea, or SOLAS, and specific class regulations govern ship maintenance in the shipping sector. To keep their class approval, ships must use a through-life survey system. The structure of the ship's requirement for maintenance is revealed through class surveys, intermediate surveys, yearly surveys, and bottom/docking studies of the hull. Surveys may be labor-intensive, costly, and subject to human error. The idea of automated inspections has gained traction in recent years. Drones will be able to inspect ship structures more effectively and for less money and time because of developments in artificial intelligence and drone technology. These procedures support the future of inspections as envisioned by classification societies and owners. The day of a drone flying inside a ship with a camera to find structural issues like rust and report them is now closer than ever. The goal of this study is to look for rust in pictures. A presentation of the datasets created to support the techniques utilized. Results are given alongside examples of the study's successes. The study's conclusion makes recommendations for more studies.

### II. LITERATURE SURVEY

In order to discover metallic corrosion, A texture evaluation technique has been used to find out metallic corrosion. This approach targeted the complex texture of the corroded place and utilized a preferred deviation clear to discover the easy texture as corroded. Both of the techniques, an AdaBoost-primarily based totally corrosion detector (ABCD) and a color-primarily based totally corrosion detector with vulnerable classifiers (WCCD), combine vulnerable classifiers to reap excessive performance. The first approach involves steps: the extraction of pixels with complex texture in the usage of the strength of the symmetric grey degree co-incidence matrix, and the categorization of extracted pixels in the usage of their coloration information. To discover corrosion, they evolved a coloration monitoring utility and retrained the bvlc-reference-cafenet version in the usage of switch studying. Using information from special sources, their packages have been examined, and it turned into proof that the deep studying technique produces higher outcomes. A computer imaginative and prescient method for the identity and quantitative assessment of corrosion on pipelines. The approach used various threshold values to come across corrosion primarily based totally on the saturation of its color and the k-way clustering set of rules to pinpoint the maximum critically broken regions. A Gaussian aggregate version changed into skilled the usage of photos of used components attaining an 85 % accuracy at the take a look at the set. When evaluating the overall performance of counselled and mounted ImageNet designs, it changed into found that the VGG16 structure with RGB or YCbCr enters photos accomplished the great. They used CNNs, one to decide whether or not the photo changed into broken or now no longer and the alternative to decide what sort of harm changed into there. They have a look at produced outstanding results, with an accuracy charge of 86.7% for every one of the 7 instructions of damages used to teach

their version. The CNN changed into taught to differentiate among 4 instructions. As a result, after 37 days, the version's accuracy rose from its preliminary sixty-six percent to ninety-three percent. Both computer's imaginative and prescient and deep getting-to-know strategies could be placed to take look at in this investigation. An exceptional texture-primarily based totally approach treats the corroded regions proven on photos of vessel cubicles as gadgets that want to be observed and located.

### III. DEVELOPED METHODOLOGY

This chapter deals with algorithms of CNN for the above algorithm. The CNN algorithms for the aforementioned algorithm are covered in this chapter. On the underlying Raspbian Jessie OS with Anaconda Navigator, a high-level scripting language called OpenCV Python is utilized to categories the rust in metals. Using a dataset of 2895 photos, the algorithm was trained. The model had a 96 percent accuracy rate. The model's sensitivity and specificity were both about 65%, which increased prediction accuracy.

#### WEAK-CLASSIFIER COLOR-BASED CORROSION DETECTOR (WCCD) (EXISTING ALGORITHM)

WCCD is a supervised classifier that has two weak classifier-like stages. It was built utilizing a cascade strategy. In order to develop a global classifier that performs noticeably better overall, it is necessary to connect numerous rapid classifiers with poor performance together. Each weak classifier uses a different property of the objects to categories, reducing the incidence of false positive detections at each level. The classifiers must display a false negative proportion that is quite near to zero for a decent overall performance.

$$E = \sum_{i=0}^{31} \sum_{j=0}^{31} p(i, j)^2$$

where  $p(i, j)$  is the probability of the occurrence of grey levels  $i$  and  $j$  at distance  $d$  and orientations  $\alpha$  or  $\alpha + \pi$ . Patches with an energy lower than a given threshold  $\tau E$ . The pixels of the patches that have made it beyond the roughness stage are filtered in the second step. At this point, the colour information that can be seen from corroded regions is utilized. The classifier functions over the Hue-Saturation-Value (HSV) space more specifically after realizing that the HSV-values that may be visible in corroded regions are confined in a narrow subspace of the HS plane. The V component has not been seen to be relevant or essential to characterize the color of corrosion, but it is utilized to prevent the well-known instability in the computation of hue and saturation when color is close to white or black.

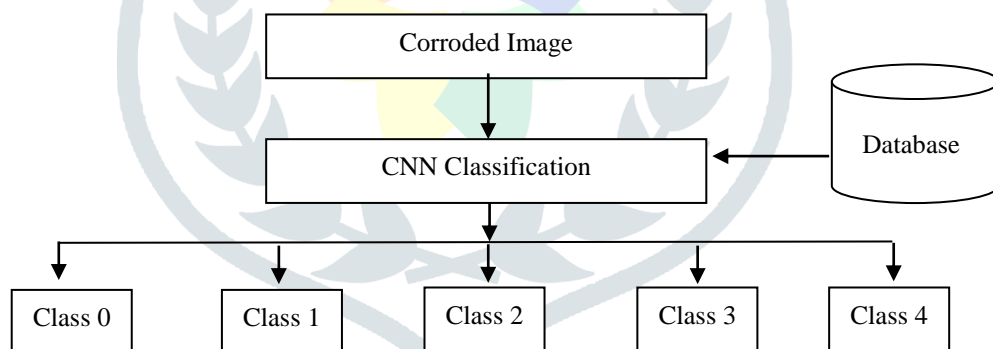


Fig 1. Block Diagram

Fig 1 represents the block diagram of the convolutional neural networks (CNN). The model is trained with the dataset. The model uses flatten layer, fully connected layer and sigmoid function to classify the images. K-Means clustering algorithm is used to find the area affected.

### IV. ALGORITHM IMPLEMENTATION

The steps involved in CNN algorithm are as shown below,

#### STEPS USED IN CNN

- Provide input image into convolution layer
- Choose parameters, follow filters with strides, padding if requires. Perform convolution at the picture and follow ReLU activation to the matrix.
- Add as many convolutional layers as satisfied
- Flatten the output and feed it into a completely related layer (FC Layer)
- Output the magnificence of the usage of an activation function (Logistic Regression with value functions) and classifies images.

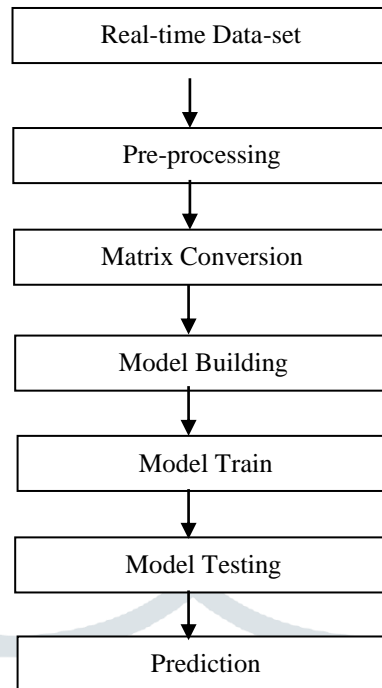


Fig. 2 Flow Diagram

**Proposed Methodology**

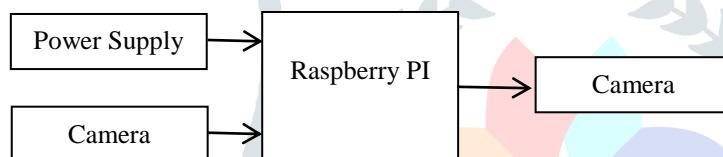


Fig. 3 Block diagram of proposed system

A dedicated camera input connection on the Raspberry Pi enables users to take real-time pictures. Based on the model, we can then estimate to which class the image belongs. The model in this project predicts the class to which the output belongs and provides the class value number. Through a PC, the processed image with the class value of the corroded and uncorroded image is shown.

**IV. RESULTS AND DISCUSSION**

In this proposed system Raspberry pi 4 operating system is connected to the monitor using USB cable. The program for Conventional Neural Network algorithm was written in Python IDE 3, stimulated in Anaconda Navigator.

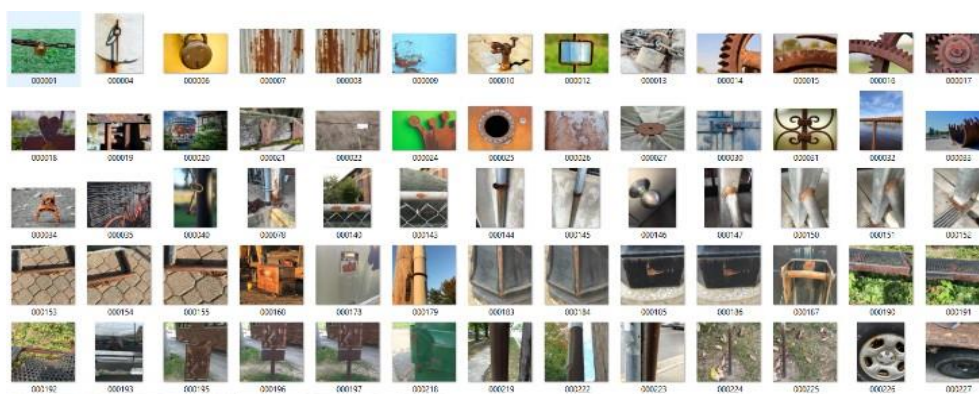


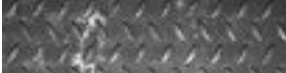





Fig. 4 Software Implementation Output



Fig. 5 Software Implementation Output

Table 1. Training and Validation Loss

S. No.	Image	Estimation Time (ms/Step)	Training Loss	Validation Loss
1		42s 170ms/step	0.1091	7.1856
2		42s 168ms/step	0.0755	7.1336
3		44s 177ms/step	0.0882	5.3075
4		51s 203ms/step	0.1212	5.0422
5		86s 342ms/step	0.0818	5.0003
6		92s 370ms/step	0.0996	3.7589

The Table 1 represents the output of the CNN algorithm implemented in the Anaconda Navigator using Tensor-flow. It shows the input test image of leaf and the output terminal parameters. This terminal shows the parameters like estimation time, training loss, validation loss.

Table 2. Training and Validation Accuracy

S. No.	Image	Estimation Time	Training Loss	Validation Loss
1	Image 1	0.9534	0.4000	1
2	Image 2	0.9725	0.4000	1
3	Image 3	0.9658	0.5000	2
4	Image 4	0.9478	0.5000	3
5	Image 5	0.9667	0.3333	3
6	Image 6	0.9600	0.6667	4

The Table (2) shows the type of disease classified and parameters like estimation time, training accuracy, test accuracy. As the training accuracy increases the sensitivity and specificity of the prediction increase.

Table 3. Analysis

Parameters	Class Value 3	Class Value 4	Total
Test Positive	True Positive = 5	True Positive = 1	Positive = 6
Test Negative	False Negative = 2	False Negative = 4	Negative = 6
Total	No of Class Value 3 Detected = 7	No of Class Value 4 Detected = 5	Grand Total = 12

From the table 3 it can be inferred that both sensitivity and specificity are more than 65%. Also, various parameters like Sensitivity, Specificity, Pre-Test Predictivity and Likelihood Ratios can be inferred.

### Summary of the Model

```

Model: "sequential"
Layer (type)                Output Shape              Param #
-----
conv2d (Conv2D)              (None, 118, 118, 32)     896
max_pooling2d (MaxPooling2D) (None, 59, 59, 32)       0
conv2d_1 (Conv2D)            (None, 57, 57, 64)       18496
max_pooling2d_1 (MaxPooling2D) (None, 28, 28, 64)       0
conv2d_2 (Conv2D)            (None, 26, 26, 64)       36928
max_pooling2d_2 (MaxPooling2D) (None, 13, 13, 64)       0
flatten (Flatten)            (None, 256)              0
dense (Dense)                (None, 128)              32896
dropout (Dropout)            (None, 128)              0
...
Total params: 139,780
Trainable params: 139,780
Non-trainable params: 0
  
```

Fig 6. Summary of the Model

Figure 6 represents the summary of the model trained. It shows image parameters available at each stage of layer used. The number of parameters trained is 1,39,780.

### Execution Results



Fig 7. Execution results 1



Fig 8. Execution results 2

The Figure 7 and Figure 8 shows the experimental results of test image of metal.

### V. CONCLUSION

The objective of this project is to build algorithms for corrosion detection. The system's ability to determine whether an object is corroded or not is significantly influenced by the color of the image. The experimental results show that the built algorithm is a tool for automated corrosion identification by digital image analysis that delivers improved accuracy and precision with minimal mean square when compared to existing approaches.

## REFERENCES

- [1] J.A. I. Diaz, M. I. Ligeralde, J. A. C. Jose and A. A. Bandala, "Rust detection using image processing via Matlab," TENCON 2017 - 2017 IEEE Region 10Conference, 2017, pp. 1327-1331, doi: 10.1109/TENCON.2017.8228063.
- [2] Agarwala, Vinod S., Perry L. Reed, and Siraj Ahmad. "Corrosion detection and monitoring-A review." CORROSION 2000. NACE International, 2000.
- [3] Alkanhal, T.A., 2014. "Image Processing Techniques Applied for Pitting Corrosion Analysis", pp. 385–391 no. June 2014.
- [4] Baraldi, A., Parmiggiani, F., 1995. "An investigation of the textural characteristics associated with Gray Level Co-occurrence Matrix statistical parameters. Geosci. Remote Sensing", IEEE Trans. 33, 293–304.
- [5] A. Sharma, and T. Tejinder, "Techniques for Detection of Rusting of Metals using Image Processing: A Survey," International Journal of Emerging Science and Engineering, vol. 1, 2013, pp 60-62.
- [6] Z. Chen, L. Liu, L. Li, and H. Li, "A two-stage model for project optimization in transportation infrastructure management system," *Mathematical Problems in Engineering*, vol. 2014, Article ID 914515, 8 pages, 2014.
- [7] F. Bonnin-Pascual and A. Ortiz, "Corrosion detection for automated visual inspection, developments in corrosion protection," in *Developments in Corrosion Protection*, pp. 620–632, IntechOpen, London, UK, 2014.

