**Detection of Damage on Steel Manufacturing**

*Report submitted to the SASTRA Deemed to be University as the requirement for the course*

**ICT300 - MINI PROJECT**

*Submitted By*

**Iraianban S**

**(Reg.No:124014017, B.Tech-ICT)**

**Sanjai Praveen S**

**(Reg.No:124014043, B.Tech-ICT)**

**Swetha S**

**(Reg.No:124014055, B.Tech-ICT)**



**SCHOOL OF COMPUTING**

**THANJAVUR , TAMIL NADU, INDIA -613001**

i



**SCHOOL OF COMPUTING**

**THANJAVUR – 613 401**

**Bonafide Certificate**

This is to certify that the report titled “**Detection of Damage on Steel Manufacturing**” submitted as a requirement for the course, ICT300: **MINI PROJECT** for B.Tech. is a bonafide record of the work done by **Iraianban S (Reg.No:124014017, B.Tech-ICT) , Sanjai Praveen S (Reg.No:124014043, B.Tech-ICT) , Swetha S (Reg.No:124014055, B.Tech-ICT)** during the academic year 2021-22, in the School of Computing, under my supervision.

**Signature of Project Supervisor :**

**Name with Affiliation :** N. Senthil Anand , Assistant Professor –II , Dept of mca/soc

**Date**

**:**

Mini Project *Viva voc*e held on

**Examiner 1** **Examiner 2**

ii

**Acknowledgements**

We would like to thank our Honourable Chancellor **Prof. R. Sethuraman** for providing us with an opportunity and the necessary infrastructure for carrying out this project as a part of our

curriculum.

We would like to thank our Honourable Vice-Chancellor **Dr. S. Vaidhyasubramaniam** and **Dr. S. Swaminathan,** Dean**,** Planning & Development, for the encouragement and strategic supportat every step of our college life.

We extend our sincere thanks to **Dr. R. Chandramouli,** Registrar, SASTRA Deemed to be University for providing the opportunity to pursue this project.

We extend our heartfelt thanks to **Dr. A. Umamakeswari**, Dean, School of Computing, **Dr. B.Santhi**, Associate Dean, Department of Computer Application, **Dr. R. Muthaiah**, AssociateDean, Department of Information Technology and Information & Communication Technology and **Dr. V. S. Shankar Sriram**, Associate Dean, Department of Computer Science and Engineering fortheir motivation and support offered in materializing this project.

Our guide **N. Senthil Anand , Assistant Professor –II** , School of Computing was the driving force behind this whole idea from the start. His deep insight in the field and invaluable suggestions helped us in making progress throughout our project work. We also thank the project review panel members for their valuable comments and insights which made this project better.

We would like to extend our gratitude to all the teaching and non-teaching faculties of the School of Computing who have either directly or indirectly helped us in the completion of the project.

We gratefully acknowledge all the contributions and encouragement from my family and friends resulting in the successful completion of this project. We thank you all for providing me an opportunity to showcase my skills through project.

iii

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No** |
| 1 | Input Images | 4 |
| 2 | Visual Inspection by manually | 13 |
| 3 | Visual Inspection By Matlab Code | 13 |
| 4 | Resized Image | 14 |
| 5 | Image after Roughness Analysis | 14 |
| 6 | Image after Color Analysis | 15 |
| 7 | Image after Combination of Roughness Analysis and Color Analysis | 15 |

iv

**Abstract**

Visual Inspection is a method that used for corrosion detection which take more time for large areas. way to overcome the time constraints by developing the corrosion detection algorithm that can detect the corrosion in series of images. there are numerous structure for shade of corrosion on steel structures . A dataset of corroded images including rust , non-uniform illumination, etc. corrosion detection algorithm based on the roughness analysis and color analysis, and evaluation metrics. This algorithm works with the combination of two visual aspects roughness analysis and color analysis in order to detect the corroded area. roughness analysis calculated by graylevel co occurrence matrix . color analysis calculated by HSV color space . Dataset consists of larhe number of corroded images which implemented in the algorithm .when the combination of roughness and color analysis the algorithm detect efficiently.

.

**KEY WORDS:** Visual Inspection , Roughness Analysis, Color Analysis

v

**Table of Contents**

**Title Page No.**

Bonafide Certificate ii

Acknowledgements iii

List of Figures iv

Abstract v

1.Summary 1

2. Merits and Demerits of the base paper 3

3. Source Code 4

4. Snapshots 13

5.Conclusion and Future Plans 16

6. References 17

7.Appendix – Base Paper 18

vi

**CHAPTER 1**

**SUMMARY OF THE BASE PAPER**

**Base Paper Details:**

Title : Detection of corrosion using automated image processing

Journal : Development in the Built Environment

Author : M. Khayatazad, L. De Pue, W. De Waele

Publisher : Received 15 April 2020; Received in revised form 15 July 2020; Accepted 16 July 2020

Available online 23 July 2020, 2666-1659/© 2020 The Author(s). Published by Elsevier Ltd.

This is an open access article under the CC BY-NC-ND license

**Summary:**

**Introduction:**

Corosion is a degradation or destruction of metal due to chemical reaction with some external sources like temperature , humidity etc. first step to intiate is Visual Inspection (VI). VI approach is used treat defects in surface and give rough description as result. In corrosion detection we have two important aspects such as texture analysis and color analysis. When we combine texture analysis and color analysis the result will be more efficient.

**Corrosion Detection Algorithm:**

In the view of visual inspection corroded area is rougher than non corroded areas and shade of corroded region is hue between red and brown. Algorithm divided into 2 main parts such as roughness analysis and color analysis. Roughness analysis define area that is based on uniformity metric. Second step is color analysis investigate the candiate area as corroded or not.

**Roughness Analysis:**

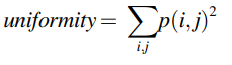
There were the mix of corroded and non corroded area . we differentiate that by uniform color distribution or not. Corroded areas have non uniform color distribution and non corroded areas have uniforn color distribution. We segmented the images to patches to measure uniformity. The value of uniformity is 0 and 1.

* **Non corroded patches:** uniformity value is equal to 1 refered as uniform color

distribution

* **Corroded patches :** uniformity value is equal to 0 refered as non uniform

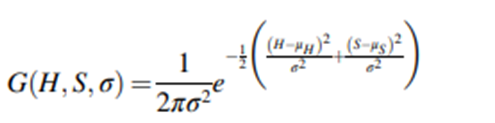
Color distribution



1

**Color Analysis:**

In color analysis, we use HSV color space. Corrosion at atmospere are in the shade of red , yellow and red-brown. Comparing the corrosion color with reference color to determine the corroded pixels.the parameters of color analysis used is gaussian filter and threshold of color spectrum.



**Performance Metrics:**

* For perfect corrosion detection, precision=1 and recall =1.
* When the algorithm said to be ideal precision and recall near 1.

**Dataset:**

Dataset Name:

Corrosion Image Data Set for Automating Scientific Assessment of Material

Description :

Dataset consist of 600 images which is the mixture of different scale of ratings. These are obtained from labrotary over 10 years.there where note only corrosion is present and also damages are also present

2

**CHAPTER 2**

**MERITS AND DEMERITS OF THE BASE PAPER**

**Merits:**

1. Corrosion Detection using Automated image processing provides an objective and accurate way, because it eliminates human bias and errors in corrosion detection.

2. It saves time compared to manual visual inspection and analyze large number of image data in short time.

3. It is not necessary to any physical contact with the surface of the steel structure.

4. It is easy to handle, and it doesn’t need any high level of technical experts.

**Demerits:**

1. It is expensive to maintain.

2. It only detect corrosion in outer surface.

3.Poor quality of images may affect accuracy of the result.

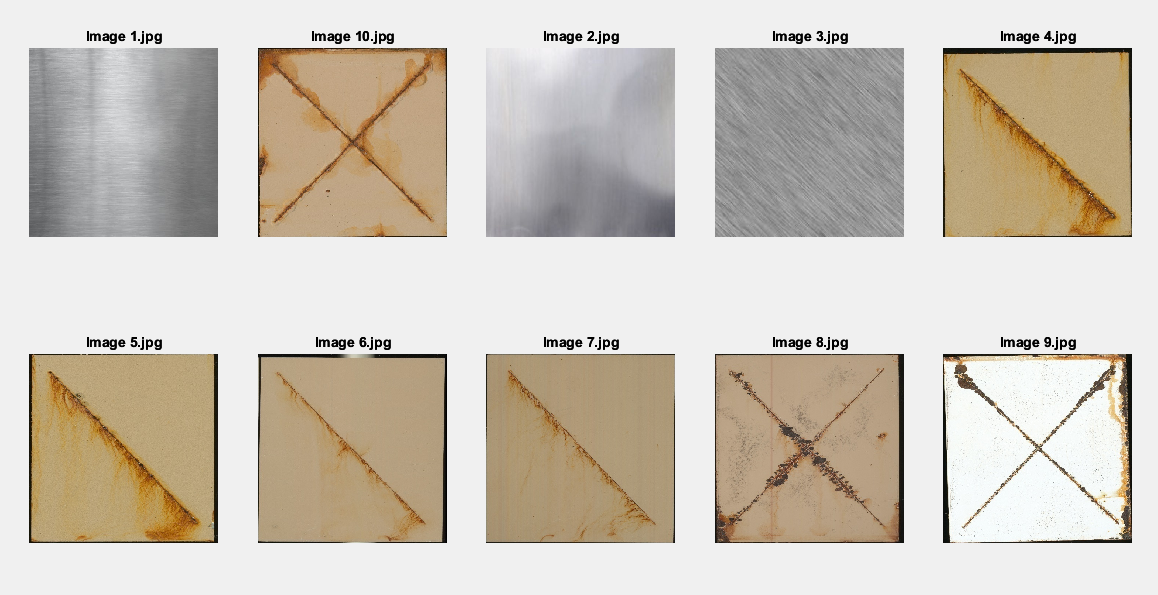
4.It gets effected to external conditions such as temperature, humidity etc.

3

**CHAPTER 3**

**SOURCE CODE**

**Input Images:**



**Visual inspection by manual:**

<! DOCTYPE html>

<html>

<head>

<title>Visual Inspection</title>

</head>

<body>

<h1>Visual Inspection Process for Corrosion Detection</h1>

<p>The first step towards the maintenance of structures is Visual Inspection (VI). This approach only treats surface defects and delivers a

rough description of the condition of the structure and its deterioration.

Nowadays this is mainly done by humans to collect qualitative data.</p>

<p>The images are classified here based on two categories Corroded and Non-Corroded</p>

<table>

<tr>

<td>

<h2>Corroded</h2>

<img src="./Corroded/I137\_steel\_mill-finish\_spray-zinc-phosphate-wo-chrome-seal\_MIL-DTL-53022\_MIL-DTL-53039.jpg" height="100" weight="100">

<img src="./Corroded/I275\_steel\_abrasive-blasted\_Abrasive-blasted\_MIL-DTL-53022\_MIL-DTL-53039.jpg" height="100" weight="100">

4

<img src="./Corroded/I317\_steel-(high-hard)\_abrasive-blasted\_spray-zinc-phosphate-wo-chrome-seal\_MIL-DTL-53022\_MIL-DTL-53039.jpg"height="100" weight="100">

<img src="./Corroded/I426\_Steel---HHA\_abrasive-blasted\_Zinc-Phosphate-(nonchrome-seal)\_MIL-DTL-53022\_MIL-DTL-64159.jpg" height="100" weight="100">

<img src="./Corroded/I449\_Steel---LCS\_mill-finish\_silane-pretreatment\_MIL-PRF-23377\_MIL-PRF-85285.jpg" height="100" weight="100">

<img src="./Corroded/I526\_Steel---LCS\_mill-finish\_trivalent-chromium-pretreatment\_MIL-DTL-53022\_MIL-DTL-64159.jpg" height="100" weight="100">

<img src="./Corroded/I96\_steel-(high-hard)\_abrasive-blasted\_chromated-wash-primer\_MIL-DTL-53022\_MIL-DTL-53039.jpg" height="100" weight="100">

</td>

</tr>

<table>

<tr>

<td>

<h2>Non-Corroded</h2>

<img src="./Non-corroded/2(resize).jpg" height="100" width="100">

<img src="./Non-corroded/3(resize).jpg" height="100" weight="100">

<img src="./Non-corroded/4(resize).jpg" height="100" width="100">

</td>

</tr>

</table>

</body>

</html>

**Visual inspection by matlab:**

% Define the directory

img\_dir = 'C:\Users\Kokila Saravanan\OneDrive\Documents\MATLAB\Visual Inspection';

% Define the directory to save the corroded images

corroded\_dir = 'D:\Corroded';

% Define the directory to save the non-corroded images

non\_corroded\_dir = 'D:\Non-corroded';

% Create the directories if they don't exist

if ~exist(corroded\_dir, 'dir')

mkdir(corroded\_dir);

end

if ~exist(non\_corroded\_dir, 'dir')

mkdir(non\_corroded\_dir);

end

5

% Get a list of all the image files

img\_files = dir(fullfile(img\_dir, '\*.jpg'));

% Initialize a cell array

image\_table = cell(length(img\_files), 1);

% Loop through each image file

for i = 1:length(img\_files)

% Load the image

img = imread(fullfile(img\_dir, img\_files(i).name));

% Convert the image to grayscale

gray\_img = rgb2gray(img);

% Enhance the contrast of the image

enhanced\_img = imadjust(gray\_img, [0.2 0.8], []);

% Perform thresholding

binary\_img = imbinarize(enhanced\_img, 'adaptive');

% Fill the gaps in the corroded regions

filled\_img = imfill(binary\_img, 'holes');

% Remove small objects from the image

cleaned\_img = bwareaopen(filled\_img, 50);

% Find the connected regions in the image

cc = bwconncomp(cleaned\_img);

% Get the properties of the regions

stats = regionprops(cc, 'Area', 'BoundingBox');

% Filter out regions

valid\_regions = stats([stats.Area] > 1000 & [stats.Area] < 50000);

% Draw the bounding boxes on the original image

figure;

imshow(img);

hold on;

for j = 1:length(valid\_regions)

rectangle('Position', valid\_regions(j).BoundingBox, 'LineWidth', 2, 'EdgeColor', 'r');

end

hold off;

6

% Save the corroded and non-corroded images separately

if isempty(valid\_regions)

imwrite(img, fullfile(non\_corroded\_dir, img\_files(i).name));

else

imwrite(img, fullfile(corroded\_dir, img\_files(i).name));

end

% Store the image in the cell array

image\_table{i} = img;

end

% Display the images in a table

figure;

tiledlayout('flow', 'TileSpacing', 'Compact');

for i = 1:length(image\_table)

nexttile;

imshow(image\_table{i});

title(img\_files(i).name, 'Interpreter', 'none');

end

**Resize the image:**

% Load image

img = imread("Images\I152\_steel-(high-hard)\_abrasive-blasted\_flash-rust-inhibitor\_MIL-DTL-53022\_MIL-DTL-53039.jpg");

% Get the current size of the image

currentSize = size(img);

% Define the new width of the image

newWidth = 500;

% Compute the new height of the image

newHeight = round(currentSize(1) \* newWidth / currentSize(2));

% Resize the image

resizedImg = imresize(img, [newHeight, newWidth]);

7

% Display the original and resized images side by side

figure;

subplot(1,2,1);

imshow(img);

title('Original Image');

subplot(1,2,2);

imshow(resizedImg);

title('Resized Image');

**Roughness Analysis:**

% Set the directory path

imageDir = 'C:\Users\Kokila Saravanan\OneDrive\Documents\MATLAB\Visual Inspection';

% Get the list of all image files

imageFiles = dir(fullfile(imageDir, '\*.jpg'));

% Define the patch size and number of patches

patchSize = 32;

numPatches = 64;

% Initialize arrays to store the patch data, uniformity values, roughness values, and corroded/non-corroded labels

patchData = zeros(numPatches, patchSize^2);

uniformityValues = zeros(numPatches, 1);

roughnessValues = zeros(numPatches, 1);

labels = zeros(numPatches, 1);

% Loop over each image file

for i = 1:length(imageFiles)

% Read in the image and resize it

image = imread(fullfile(imageDir, imageFiles(i).name));

image = imresize(image, 0.5);

% Convert the image to grayscale

grayImage = rgb2gray(image);

% Select patches from the image

[rows, cols] = size(grayImage);

randRows = randi(rows-patchSize, numPatches, 1);

randCols = randi(cols-patchSize, numPatches, 1);

for j = 1:numPatches

patch = grayImage(randRows(j):randRows(j)+patchSize-1, randCols(j):randCols(j)+patchSize-1);

patchData(j, :) = reshape(patch, [1, patchSize^2]);

end

% Calculate the uniformity of the patches

uniformityValues = std(patchData, 0, 2)./mean(patchData, 2);

8

% Calculate the roughness of the patches

[Gmag, ~] = imgradient(patch);

roughnessValues = mean(Gmag(:));

% Set labels for corroded/non-corroded patches

threshold = 0.15;

labels(roughnessValues > 0 & uniformityValues > threshold) = 1;

% Separate corroded and non-corroded patches

corrodedPatches = patchData(labels == 1, :);

nonCorrodedPatches = patchData(labels == 0, :);

% Mark corroded areas

corrodedAreas = false(size(grayImage));

corrodedAreasCount = 0;

for j = 1:numPatches

if labels(j) == 1

patchRowStart = randRows(j);

patchColStart = randCols(j);

corrodedAreas(patchRowStart:patchRowStart+patchSize-1, patchColStart:patchColStart+patchSize-1) = true;

corrodedAreasCount = corrodedAreasCount + 1;

end

end

% Overlay the corroded areas on the image and display it

figure;

imshow(imoverlay(image, corrodedAreas, [1 0 0]));

title(sprintf('Image %d with %d Corroded Areas Marked', i, corrodedAreasCount));

end

**Color Analysis:**

% Define directory containing images

image\_dir = 'C:\Users\Kokila Saravanan\OneDrive\Documents\MATLAB\Visual Inspection';

% Load all images in directory

image\_files = dir(fullfile(image\_dir, '\*.jpg')); % Assumes images are in PNG format

num\_images = length(image\_files);

% Set filter parameters

sigma = 12;

hue\_threshold =0.89;

9

% Loop through all images

for i = 1:num\_images

% Load image

image\_filename = fullfile(image\_dir, image\_files(i).name);

image = imread(image\_filename);

% Convert image to HSV

image\_hsv = rgb2hsv(image);

% Apply Gaussian filter to hue channel

hue\_filtered = imgaussfilt(image\_hsv(:,:,1), sigma);

% Compute probability distribution of hue values

hue\_prob = histcounts(hue\_filtered(:), linspace(0, 1, 101), 'Normalization', 'probability');

% Identify corroded pixels

corroded\_pixels = hue\_prob(round(hue\_filtered\*100)+1) < hue\_threshold & (image\_hsv(:,:,2) > 0.5);

% Create mask to mark corroded pixels in red

red\_mask = uint8(cat(3, ones(size(image(:,:,1))), zeros(size(image(:,:,2))), zeros(size(image(:,:,3))))) .\* 255 .\* uint8(corroded\_pixels);

% Create color image to mark corroded pixels in red

marked\_image = image + red\_mask;

% Display corroded image if there are any corroded pixels

if any(corroded\_pixels(:))

figure;

imshow(marked\_image);

title(sprintf('Corroded Image %d', i));

end

end

**Combination of Roughness and Color Analysis:**

% Define image directory

img\_dir = 'C:\Users\Kokila Saravanan\OneDrive\Documents\MATLAB\Visual Inspection';

% Define roughness threshold

roughness\_threshold = 0.5;

% Get list of all image files in directory

img\_files = dir(fullfile(img\_dir, '\*.jpg'));

10

% Calculate number of rows and columns needed for subplots

num\_images = numel(img\_files);

num\_cols = ceil(sqrt(num\_images));

num\_rows = ceil(num\_images / num\_cols);

% Create figure to display images in subplots

figure;

% Loop through all images in directory

for i = 1:num\_images

% Read image

img = imread(fullfile(img\_dir, img\_files(i).name));

% Convert image to HSV

img\_hsv = rgb2hsv(img);

% Calculate roughness using Sobel edge detector on V channel

img\_v = img\_hsv(:, :, 3);

Hx = [-1 0 1; -2 0 2; -1 0 1];

Hy = Hx';

Ix = conv2(img\_v, Hx, 'same');

Iy = conv2(img\_v, Hy, 'same');

Gmag = sqrt(Ix.^2 + Iy.^2);

roughness = sum(Gmag(:)) / numel(Gmag);

% Apply adaptive thresholding using Otsu's method on V channel

level = graythresh(img\_v);

img\_bw = imbinarize(img\_v, level);

% Create a binary mask to identify corroded pixels

corroded\_mask = (Gmag > roughness\_threshold) & img\_bw;

% Use morphological operations to fill in corroded areas

se = strel('disk', 5);

filled\_mask = imclose(corroded\_mask, se);

% Create a 3D array of the red color to mark corroded pixels

red\_color = [255, 0, 0];

red\_array = repmat(reshape(red\_color, [1, 1, 3]), [size(img, 1), size(img, 2), 1]);

% Replace corroded pixels with red pixels and non-corroded pixels with original image pixels

img\_replaced = img;

img\_replaced(filled\_mask) = red\_array(filled\_mask);

11

% Display image in subplot

subplot(num\_rows, num\_cols, i);

imshow(img\_replaced);

title(img\_files(i).name, 'Interpreter', 'none');

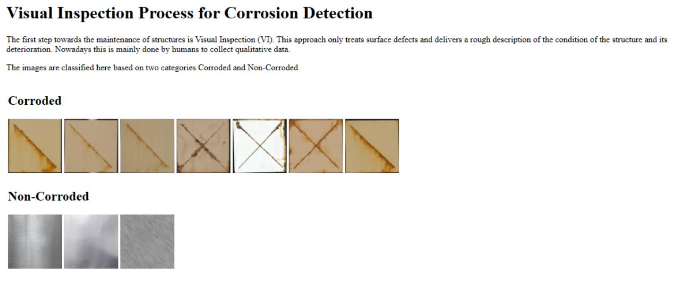
end

12

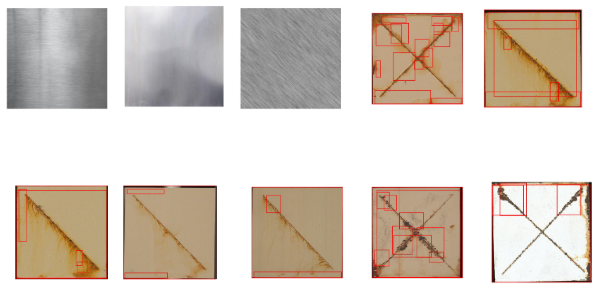
**CHAPTER 4**

**SNAPSHOTS**

**Visual Inspection by Manual:**

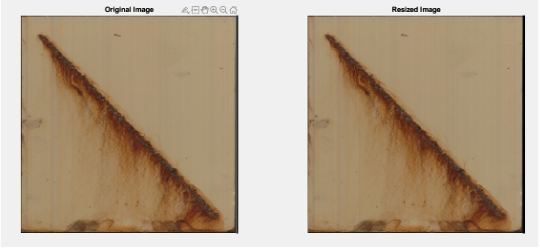


**Visual Inspection by Matlab code:**

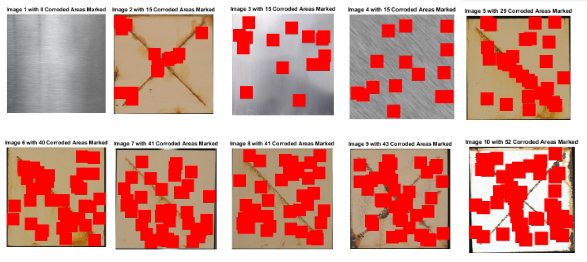


13

**Resize the Image:**

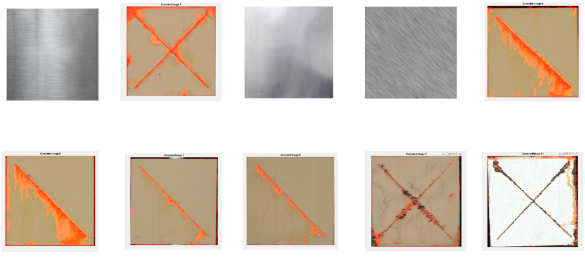


**Roughness Analysis:**

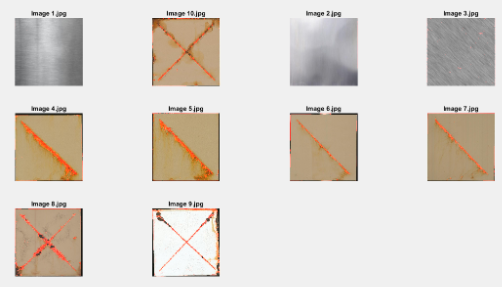


14

**Color Analysis:**



**Combination of roughness and color Analysis:**



15

**CHAPTER 5**

**CONCLUSION ANS FUTURE WORKS**

**Conclusion:**

Roughness analysis determines almost entire area as rough but not able to conclude all rough area as corrsion. The colour analysis appropriately determines the corroded surface and also the rust strains as corroded regions. Considering the above drawbacks the combination of roughness and colour analysis gives more efficient and accurate result to determine the corroded region.

**Future Works:**

1.Corrosion detection using sensing technologies such as ultrasonic, magnetic flux leakage, or eddy current to improve the accuracy.

2.Corrosion detection can be continuously monitoring by real time monitoring system to prevent damage in early stage.

3.Corrosion detection using CNN to improve accuracy.

4.Corrosion detection can be done in extreme condition like off-shore platform or pipelines.

16

**CHAPTER 6**

**REFRENCES**

1. AbdelRazig, Y.A., 1999. Construction Quality Assessment: a Hybrid Decision Support Model Using Image Processing and Neural Learning for Defect Recognition. Purdue University.
2. Agdas, D., Asce, M., Rice, J.A., Asce, M., Martinez, J.R., Lasa, I.R., 2016. Comparison of Visual Inspection and Structural-Health Monitoring as Bridge Condition Assessment Methods, vol. 30, pp. 1–10 no. 3.
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. Bonnin-Pascual, F., Ortiz, A., 2014. “Corrosion Detection for Automated Visual Inspection,” in Developments In Corrosion Protection. INTECH, pp. 619–632.
6. Chen, P.H., Chang, L.M., 2003. Artificial intelligence application to bridge painting assessment. Autom. ConStruct. 12 (4), 431–445.
7. Chen, P.-H., Chang, Y.-C., Chang, L.-M., Doerschuk, P.C., 2002. Application of multiresolution pattern classification to steel bridge coating assessment. J. Comput. Civ. Eng. 16 (4), 244–251.

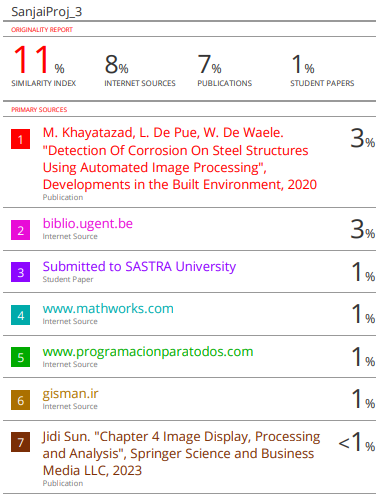
17

**CHAPTER 7**

**APPENDIX**

**Base Paper Link**[**:** https://doi.org/10.1016/j.dibe.2020.100022](file:///C:\Users\User\Downloads\:%20https:\doi.org\10.1016\j.dibe.2020.100022)

18



19



20