PACIFIC EARTHQUAKE ENGINEERING
RESEARCH CENTER

Structural Evaluation of Overhead Box Beam Highway Sign Structures

Khalid Mosalam[1]
Wenhao Ruan[1]
Jiawei Chen[2]
Selim Günay[2]
Amarnath Kasalanati[2]

[1]Department of Civil and Environmental Engineering, University of California, Berkeley

[2]Pacific Earthquake Engineering Research Center

PEER Report 2025/02
Pacific Earthquake Engineering Research Center
Headquarters at the University of California, Berkeley
September 2025

PEER 2025/02
September 2025

STATE OF CALIFORNIA – DEPARTMENT OF TRANSPORTATION

TECHNICAL REPORT DOCUMENTATION PAGE

TR-0003 (REV 04/2024)

1. REPORT NUMBER CA26-3673	**2. GOVERNMENT ASSOCIATION NUMBER**	**3. RECIPIENT'S CATALOG NUMBER**
4. TITLE AND SUBTITLE Structural Evaluation of Overhead Box Beam Highway Sign Structures		**5. REPORT DATE** September 2025
7. AUTHOR(S) Khalid M. Mosalam, Wenhao Ruan, Jiawei Chen, Selim Günay, and Amarnath Kasalanati		**6. PERFORMING ORGANIZATION CODE**
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Civil and Environmental Engineering University of California, Berkeley Berkeley, CA 94720 Pacific Earthquake Engineering Research Center University of California, Berkeley		**8. PERFORMING ORGANIZATION REPORT NO.** PEER Report No. 2025/02
12. SPONSORING AGENCY AND ADDRESS California Department of Transportation Division of Research, Innovation and System Information P.O. Box 942873 Sacramento, CA 94273-0001		**10. WORK UNIT NUMBER** **11. CONTRACT OR GRANT NUMBER** Agreement No. 65A0774, Task Order No. 08
13. TYPE OF REPORT AND PERIOD COVERED Final report. *TODO: confirm reporting period if Caltrans requires it.*		**14. SPONSORING AGENCY CODE**
15. SUPPLEMENTARY NOTES Prepared under the PEER–Bridge Master Agreement, Agreement No. 65A0774, Task Order No. 08.		
16. ABSTRACT This project quantifies the relationship between observed corrosion and corresponding degradation in the structural strength of Box-Beam Overhead Sign Structures (BBOSS) managed by the California Department of Transportation. The study integrates high-fidelity finite element modeling and model updating, full-scale laboratory testing, vibration-based field testing, and artificial-intelligence-driven vision methods for automated corrosion detection. The work focuses on real-world California BBOSS installations, including a decommissioned Seacliff structure that was transported to the Pacific Earthquake Engineering Research Center Structural Laboratory for comprehensive testing. Comparative testing of the corroded Seacliff structure and an identical corrosion-free counterpart, supported by updated simulations, was used to derive quantitative correlations between corrosion severity and structural capacity loss.		
17. KEY WORDS Overhead sign structures; box-beam overhead sign structures; corrosion; structural health monitoring; finite element modeling; model updating; genetic algorithm; computer vision; artificial intelligence; full-scale testing; Caltrans; PEER.		**18. DISTRIBUTION STATEMENT** No restrictions.
19. SECURITY CLASSIFICATION (of this report) Unclassified	**20. NUMBER OF PAGES** 161	**21. COST OF REPORT CHARGED**
Reproduction of completed page authorized.		

DISCLAIMER STATEMENT

This document is disseminated in the interest of information exchange. The contents of this report reflect the views of the authors who are responsible for the facts and accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California or the Federal Highway Administration. This publication does not constitute a standard, specification or regulation. This report does not constitute an endorsement by the Department of any product described herein.

For individuals with sensory disabilities, this document is available in alternate formats. For information, call (916) 654-8899, TTY 711, or write to California Department of Transportation, Division of Research, Innovation and System Information, MS-83, P.O. Box 942873, Sacramento, CA 94273-0001.

Structural Evaluation of Overhead Box Beam Highway Sign Structures

Khalid M. Mosalam[1]

Wenhao Ruan[1]

Jiawei Chen[2]

Selim Günay[2]

Amarnath Kasalanati[2]

[1]Department of Civil and Environmental Engineering
University of California, Berkeley

[2]Pacific Earthquake Engineering Research Center

PEER Report No. 2025/02
Pacific Earthquake Engineering Research Center
Headquarters, University of California, Berkeley
September 2025

ABSTRACT

The California Department of Transportation (Caltrans) manages hundreds of Box-Beam Over-head Sign Structures (BBOSS) installed across its highway network. These steel structures are particularly vulnerable to corrosion damage, especially at the connections involving ribbed sheet steel vertical diaphragms. Corrosion prevention and repair constitute a significant portion of main-tenance costs for this infrastructure. Structural integrity is currently assessed primarily through visual inspections, a process that is inherently subjective and prone to inconsistencies.

This project addresses the limitation of visual inspection by quantifying the relationship between observed corrosion and the corresponding degradation in structural strength. The overarching goal is to establish a robust and reliable mapping between corrosion indicators and capacity loss. To this end, the research integrates four complementary approaches: (1) high-fidelity Finite Element (FE) modeling with model updating through optimization techniques, (2) full-scale experimental testing of actual BBOSS, (3) field testing involving vibration-based system identification of BBOSS, and (4) Artificial Intelligence (AI)-driven, vision-based methods for automated corrosion detection using image data.

The study focuses on real-world BBOSS installations in California, including one near Davis (Northern California) and another near Seacliff in Ventura County (Southern California). The Seacliff structure, decommissioned due to aging, was transported to the Pacific Earthquake Engineering Research (PEER) Center's Structural Laboratory for comprehensive testing. In the final phase, comparative testing between the corroded Seacliff structure and an identical, corrosion-free counterpart, supported by updated simulations, was conducted to derive quantitative correlations between corrosion severity and structural capacity loss.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the California Department of Transportation (Caltrans) for providing financial support under Agreement No. 65A0774, as part of the PEER–Bridge Master Agreement (Task Order No. 08).

The authors also extend their sincere appreciation to the staff of the Pacific Earthquake Engineering Research (PEER) Center at the University of California, Berkeley, for their continued guidance and support. The facilities and resources provided by the PEER laboratories were essential to the successful execution of both the laboratory and field experiments.

This work builds upon the course project entitled "Automated Corrosion Detection using Deep Learning" by Mr. Guoqing Zhang, conducted under the supervision of the first author, which served as the foundation for the computer vision component of this study. The authors further acknowledge the valuable technical guidance and support of Dr. EuiHyun Choi, Mr. Gaofeng Su, and Mr. Ziyi Wang.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

1.1 MOTIVATION

Caltrans Structures Maintenance & Investigations (SM&I) Division regularly performs field inspections on approximately 18,000–19,000 state-owned overhead sign structures (e.g., Figure 1.1), with a full inspection cycle every eight years under Caltrans' Overhead Sign Structure (OHSS) program (California Department of Transportation (Caltrans), 2024a,b). The majority of these structures were installed more than 50 years ago (in the late 1960's and early 1970's) and have a history of problems with corrosion damage to connections of the ribbed sheet steel vertical diaphragms. Corrosion damage in overhead sign structures poses a significant risk to public safety and results in costly repairs. The limitations of current inspection methods, particularly manual inspection constrained by time, resources, equipments and traffic conditions, prevent a detailed and comprehensive assessment of the corrosion damage in sign structures.

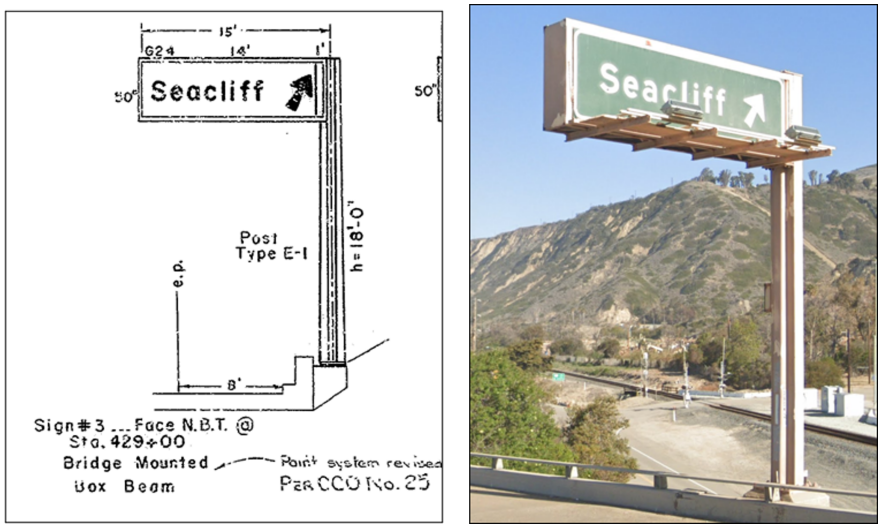


Figure 1.1: Geometry and location of the Seacliff sign structure, an example of a heavily corroded sign structure, currently demolished and used extensively in this study.

1.2 FRAMEWORK AND OBJECTIVES

To address these issues and maximize the efficiency and effectiveness of inspections, this study proposes a comprehensive framework for assessing the structural integrity of corroded overhead sign structures. By integrating field and laboratory testing, advanced numerical simulations, computer vision for detecting corrosion, and optimization algorithms for modal updating, the framework enables a more accurate and efficient approach to assessment of such infrastructure. Compared to traditional manual inspection, this integrated methodology enhances both precision and consistency in evaluating corrosion-induced damage.

Corrosion of steel structures poses a substantial and serious threat to the structural integrity and safety of civil engineering infrastructure. In recent years, the integration of advanced technologies such as the Finite Element Method (FEM), Computer Vision (CV), and various optimization algorithms, has revolutionized the approach and methods used in structural health monitoring and assessment. Using these developments, this study introduces a comprehensive framework and example for the health monitoring and assessment of sign structures. Figure 1.2 shows the proposed framework, which integrates field testing, data acquisition, AI-driven computer methods, FEM simulations, laboratory testing to simulate and analyze the effects of corrosion on structural strength and establish a corresponding mapping between corrosion index and strength. This will enhance the efficiency and effectiveness of asset management, and facilitate important decision making, such as prioritizing the structures to be replaced, repaired, or retrofitted.

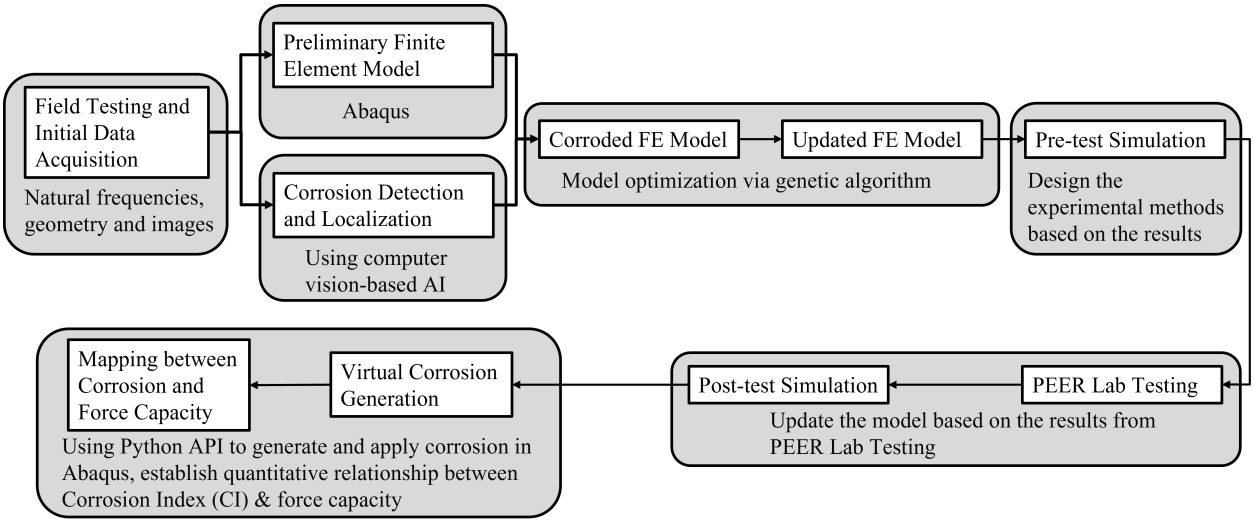


Figure 1.2: A framework for corrosion modeling and structural assessment.

The framework starts with field investigation and initial data acquisition, including geometry measurements, high-resolution imaging, and dynamic testing for natural frequency identification. These data serve as a foundation for developing the preliminary FE model using Abaqus FEA[1]. At this stage, the preliminary model only reflects the basic geometry and boundary conditions of the structure, without accounting for any corrosion or deterioration. During the modeling

[1] Abaqus FEA is a software suite for FE Analysis and Computer-Aided Engineering (CAD).

process, the initial model was built in the Abaqus User Interface and the corresponding python script (introduced in Appendix A) was recorded and organized from the journal file (*.jnl). The Abaqus Python API was subsequently employed to perform further tasks with the python script, such as optimizing the model, assigning corrosion pattern, and repeatedly executing it to establish the corrosion index (CI) - force capacity relationship. In other words, Abaqus Python API allows for automated application of corrosion effects and supports model optimization, ensuring higher efficiency and accuracy in simulating the corrosion and its impact on the structure.

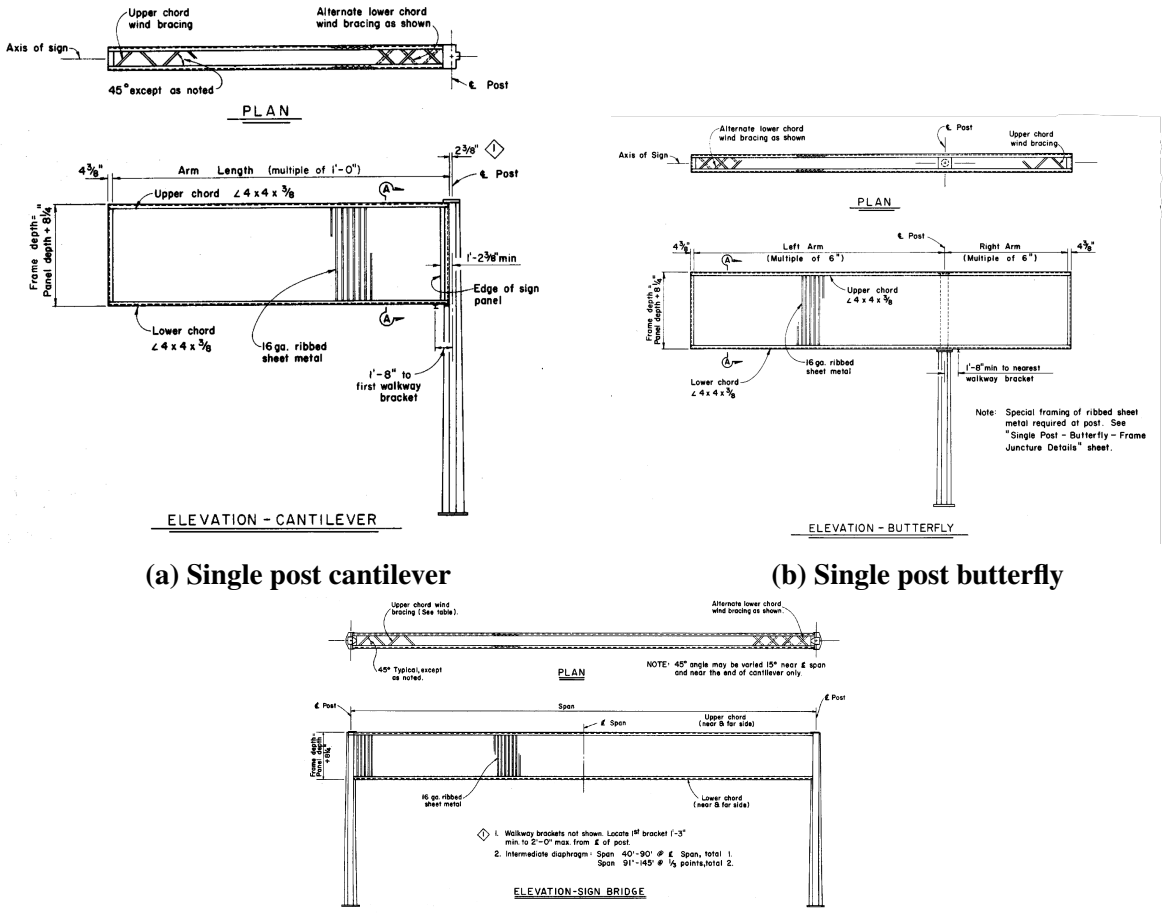
The next step involves utilizing a computer vision-based AI algorithm to detect and localize the corrosion areas on the structures using the high-resolution images collected earlier. This approach leverages advanced image processing algorithms to accurately identify the range and severity of the corrosion. The identified corrosion areas are then mapped onto the FE model using Abaqus Python API, enabling the generation of a corroded model that reflects the structural degradation. It is noted that the surface covered by the "Seacliff" sign plate remains essentially corrosion-free due to the shielding by the sign plate. Therefore, the present study on corrosion focuses on the exposed side of the box beam.

After establishing the corroded FE model, a genetic algorithm is implemented to iteratively update the model. This involves repeatedly and iteratively adjusting the structural parameters to minimize the discrepancies between the simulated and observed natural frequencies of the structure. The genetic algorithm continuously refines the model, ensuring that the FE model more accurately represents the current conditions of the corroded structure.

In order to properly set up the laboratory environment and conduct physical tests, pre-test simulations are carried out using the updated FEM. These simulations help guide the design of experimental methods, ensuring that the tests are effectively tailored to validate the model's predictions.

The framework culminated in physical testing at the Pacific Earthquake Engineering Research Center (PEER) Laboratory, where a corroded sign structure was subjected to controlled loading conditions. Because of the availability of a single-post-cantilever type sign structure (Figure 1.3a), the experimental phase focused on this type, which is also regarded as the most vulnerable configuration. Similar sign structures have been investigated in the literature. For example, Choi (2023) applied a frequency-based optimization algorithm to both single-post-cantilever and single-post-butterfly (Figure 1.3b) configurations, and further discussed its extensions to two-post configuration (Figure 1.3c). In this study, the developed framework and methodology for the single-post-cantilever type are applicable to other common configurations. To enable comparative evaluation, a similar and undamaged structure served as a reference for comparative evaluation of the single-post-cantilever type. The results from these tests were then used to further update the FEM, leading to a post-test simulation. This final step ensures that the model accurately reflects the structure's behavior under realistic loading conditions, providing a robust tool for structural assessment and informed decision-making.

To analyze how corrosion affects structural capacity, a Python script (details in Chapter 7) is used to generate synthetic corrosion patterns and apply them to the FE model. This allows the creation of a virtual dataset linking corrosion features with reaction force responses (strength capacity). A stress-weighted corrosion index (CI) is introduced, and its relationship with force



(a) Single post cantilever **(b) Single post butterfly**

(c) Two post frame

Figure 1.3: Three different types of highway sign structure configurations from (Choi, 2023).

capacity is quantified through regression, forming the basis for predictive strength evaluation under different corrosion scenarios.

1.3 ORGANIZATION OF THE REPORT

This report is organized into eight chapters and eight appendices, each addressing a key aspect of the research process and findings:

- **Chapter 2: Background** Provides the theoretical foundation necessary for understanding the methodologies used in this study. This includes discussions on AI-driven corrosion detection strategies, the FEM, genetic algorithms, and relevant experimental methods and procedures.
- **Chapter 3: Finite Element Model Development and Optimization** Describes the process of developing the FE model of the sign structure, integrating computer vision techniques

for corrosion detection, and updating the model using genetic algorithms. The chapter also covers the preliminary field testing and initial data acquisition used to create this pre-test model.

- **Chapter 4: 3D Model Idealized Loading and Pre-test Simulation** Outlines the various loading cases and pre-test simulations conducted on the preliminary FE model, including gravity load, point load, and pushover analyses, to prepare for the laboratory testing phase.

- **Chapter 5: Test Setup, Loading Protocol, and Instrumentation** Details the experimental setup and procedures performed in the Pacific Earthquake Engineering Research (PEER) Center Laboratory, where the physical testing of the sign structure took place. It includes descriptions of the instrumentation, welding procedures, and sensor attachment strategies used to measure structural responses under simulated conditions.

- **Chapter 6: Specimen Testing, Data Analysis, and Comparison** Presents the experimental results from tests of corroded and uncorroded sign structures. Presented results include strains measured by strain gauges, displacements obtained from 3D triangulation of wire-pot measurements, force-displacement relationships, energy dissipation evaluations, and comparisons of test results with numerical simulations.

- **Chapter 7: Automated Abaqus Simulation of Corrosion Effects and Structural Strength Loss** Develops a data-driven simulation framework to explore how varying synthetic corrosion scenarios affect structural strength. A stress-weighted corrosion index (CI) is proposed and correlated with force capacity, forming a basis for predictive degradation modeling.

- **Chapter 8: Conclusions and Future Work** Synthesizes the key findings of the study and highlights their implications for corrosion monitoring and maintenance planning. Future directions include multivariate modeling, Deep Reinforcement Learning (DRL)-based FE updating, and stochastic corrosion evolution.

- **Appendix A: Abaqus API** Introduces essential Abaqus operators and syntax used for creation and modification of the model.

- **Appendix B: Genetic Algorithm** Provides an overview of the genetic algorithm and its integration with the Abaqus interface for FE model updating.

- **Appendix C: Computer Vision** Presents a method for automatically extracting the corrosion distribution and corresponding coordinates from the sign structure image.

- **Appendix D: Welding Design** Illustrates the welding design implemented at the surfaces cut during the demolishing of the studied sign structure to make sure that the welded regions are stronger than the connected parts.

- **Appendix E: 3D Triangulation Algorithm** Presents the algorithm used for calculation of displacements with 3D triangulation, along with the corresponding MATLAB code optimized for parallel computation.

- **Appendix F: Python Implementation for Virtual Corrosion Generation and Corrosion-Index calculation** Presents a Python-based method to automatically generate virtual corrosion pattern and calculate corresponding Corrosion Index.
- **Appendix G: Python Implementation for Extracting Reaction Force values at Node Set** Describes a Python-based method through Abaqus Python API for extracting reaction force values from all nodes within a defined set, offering a more efficient approach to analyzing model results.
- **Appendix H: Field Deployment Guide for Corrosion Assessment** Provides a step-to-step guide to assess the residual capacity of corroded overhead sign structures using the developed "Location-Weighted Corrosion Index methodology."
- **Appendix I: UMAT Formulation using Clough Model for Shell Element** Provides the theoretical background and implementation of Clough Model for shell elements in a UMAT subroutine.

2 Background

2.1 HAMMER TEST AND STRUCTURAL DYNAMICS

The hammer test is a widely used experimental method to evaluate the dynamic properties of structures due to its effectiveness and simplicity. It is particularly useful in structural health monitoring (SHM) in identifying natural frequencies and mode shapes. This test method is applicable in various engineering contexts and circumstances. For example, Oregui et al. (2015) utilized field h ammer t est m easurements t o i dentify t he n atural f requencies a nd m onitor d amage i n railway tracks. Similarly, Liu et al. (2022) analyzed the vertical vibrations in railway ballast through applying sensors buried in railway ballast in conjunction with hammer tests.

In a hammer test, an impact force is manually applied to the structure and induces vibration. The exact force magnitude, $F(t)$, is generally unknown, while the resulting structural response, measured in terms of acceleration, can be captured using sensors and analyzed using the equation of motion as a free vibration response. The dynamic behavior of the structure can be described using the equation of motion below for a single-degree-of-freedom (SDOF) system:

$$m\ddot{u}(t) + c\dot{u}(t) + ku(t) = F(t) \tag{2.1}$$

where m is the mass, c is the viscous damping coefficient, k is the stiffness, $u(t)$ is the displacement response, $\dot{u}(t)$ is the velocity response, $\ddot{u}(t)$ is the acceleration response, and $F(t)$ is the applied impact force.

By recording the acceleration response $\ddot{u}(t)$ (e.g., Figure 3.3b) using accelerometers and applying techniques such as the Fast Fourier Transform (FFT) to the free vibration response of the structure, valuable dynamic characteristics, such as natural frequencies and mode shapes, can be extracted without knowing the exact force $F(t)$. In summary, hammer test and the sensors employed to record the resulting vibration data provide an efficient and practical approach for identifying the dynamic properties of a structure.

2.2 COMPUTER VISION AND CORROSION DETECTION

Computer Vision (CV) and Artificial Intelligence (AI) techniques have been widely used as efficient tools for structural health monitoring and assessment (Mosalam and Gao, 2024; Spencer

et al., 2019; Bao et al., 2019). Generally, computer vision techniques are capable of extracting two primary structural features: texture, which refers to surface details and patterns, such as corrosion or cracks, and displacement, which indicates the movement or deformation of structures within the scene. Additionally, these techniques have been frequently utilized in structural components detection (Narazaki et al., 2020; Cheng et al., 2024), disaster reconnaissance (Tsai et al., 2020), and damage segmentation and localization (Cha et al., 2017). Such applications not only enhance the accuracy of structural assessments but also contribute to more efficient and automated monitoring process.

Traditionally, corrosion detection has relied heavily on manual inspection by engineers. However, this approach is not only dangerous, time-consuming and labor-intensive but also subject to inconsistencies due to subjective judgment. To address these challenges, CV algorithms have increasingly been employed in corrosion detection (Khayatazad et al., 2020). In some previous work such that a python-based deep learning model by (Petricca et al., 2016) and the application of the YOLO algorithm by (Casas et al., 2024), have proposed framework and methodologies for automated steel corrosion detection and segmentation. However, these approaches often face limitations, particularly in achieving precise localization of corroded areas.

To overcome these challenges, Das et al. (2024) investigated the performance of various deep learning architectures for corrosion detection and concluded that ResNet34 and U-Net achieved the best results for semantic segmentation and image classification. Srivastava et al. (2021) successfully utilized an 8-layer U-Net to achieve corrosion detection and classification with limited dataset.

In our project, we integrate the strengths of ResNet and U-Net models. ResNet is utilized for its exceptional performance in image classification tasks, ensuring accurate detection of corrosion presence. U-Net, known for its effectiveness in image segmentation tasks and its adaptability to limited training datasets, is employed for the precise localization of corrosion. This dual-model approach allows us to accurately detect and delineate corroded regions even with a relatively small annotated dataset, as illustrated in (Siddique et al., 2021; Srivastava et al., 2021). This methodology represents a significant advancement over traditional techniques and earlier CV-based approaches, providing more reliable and consistent results in the context of steel corrosion detection. The following paragraphs introduce the basic structure and concept of ResNet and U-Net architecture.

2.2.1 Residual Neural Network

A novel approach known as Deep Residual Learning for Image Recognition (Figure 2.1) was introduced in (He et al., 2016), proposing a new formulation for the relationship between the input and output of a layer as:

$$y = F(x, \{W_i\}) + x \tag{2.2}$$

where x and y are the input and output vectors of the layer, and the function $F(x, \{W_i\})$ represents the residual mapping to be learned. In this formula, the training and optimization focus on the residual part, which helps prevent the gradient from vanishing as the number of layers increases during back-propagation and avoid the degradation problem. In his article, ResNet demonstrated efficient and accurate feature learning on the ImageNet training dataset. The 34-layer ResNet

exhibited significantly lower training error compared to both the 18-layer ResNet and the 34-layer plain network, proving that the ResNet architecture offers significant advantages in the field of image recognition and classification, particularly for deeper architectures.

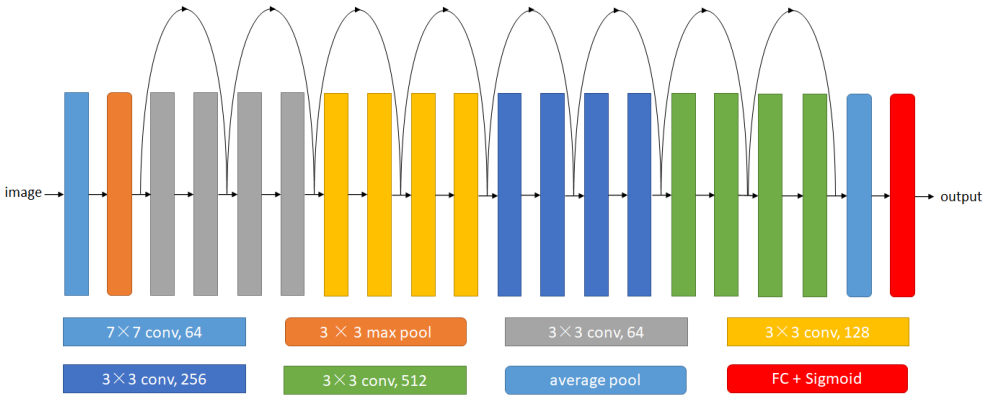


Figure 2.1: A simple 18-layer ResNet Architecture

2.2.2 U-Net

U-Net introduced by (Ronneberger et al., 2015), is a fully convolutional neural network (FCN) named for its U-shaped architecture (see Figure 2.2). The original FCN may suffer from resolution degradation between successive layers because of the convolution and pooling, which have a detrimental impact on the resolution of the final output. As shown by the gray arrows in the diagram, skip connections are used to copy information from before each pooling operation to the upsampled results. This enables the decoding layers to better reconstruct the original information, thereby compensating for potential information loss during the upsampling process. In this scenario, capturing more effective information allows the model to perform better with limited samples. This makes it particularly suitable for corrosion detection, where samples are relatively scarce. Therefore, we utilize The U-Net architecture for the corrosion identification task.

2.3 FINITE ELEMENT METHOD

2.3.1 Basic Introduction

The FEM is a powerful numerical technique used to solve complex engineering problems, particularly those involving continuous and discontinuous systems (Mosalam, 2024). FEM is widely applied across multiple fields and recognized as a powerful tool for addressing various engineering problems and challenges especially for complex geometries and varying material properties. In broader field, FEM is extensively used in solving fluid dynamics and heat transfer problems in aerospace engineering (Marcum and Weatherill, 1995), and plays a critical role in the design, analysis and optimization of biomedical devices (Mollica and Ambrosio, 2009).

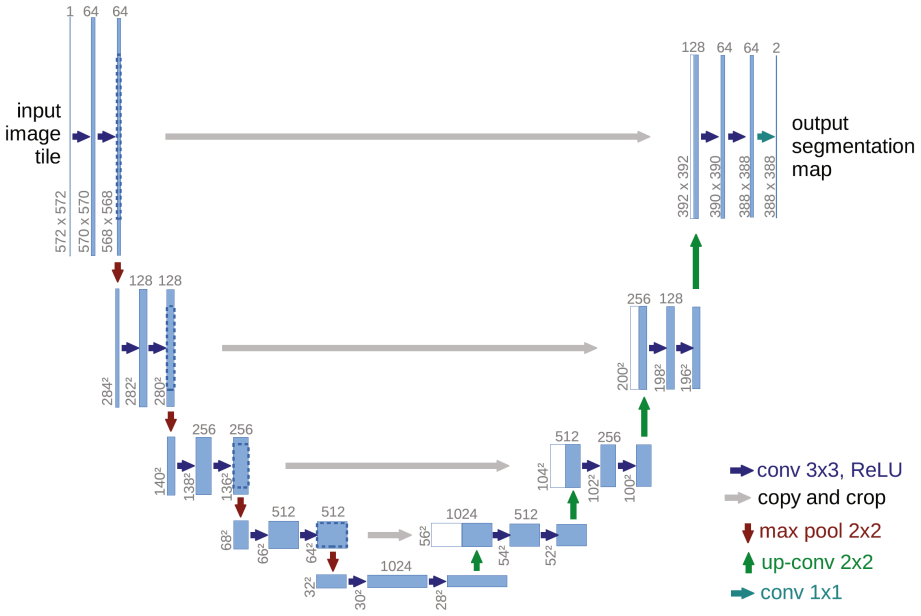


Figure 2.2: The U-Net architecture

The core principle of FEM is discretizing a "domain" into a finite number of smaller, simpler elements that can be individually analyzed. This process converts a complex problem into a set of linear or nonlinear algebraic equations, which are easier to solve. By applying boundary conditions, such as loads and constraints, FEM calculates the responses of the system, typically in terms of stress, strain, or deformation (displacement). The FEM simplifies complex models by dividing them into numerous smaller, manageable elements. This "elemental approach" makes it feasible to analyze intricate shapes, such as human organs or complex mechanical devices, which would otherwise be challenging to study in terms of mechanics.

The "domain" can encompass a wide range of materials or systems, from steel and concrete structures to fluids (Chung, 1978), and even biological tissues such as human organs (Lizee et al., 1998; Kreuzer et al., 2021). This versatility makes FEM an invaluable tool for addressing diverse challenges across modern engineering disciplines.

2.3.2 Finite Element Analysis Software

As previously mentioned, Abaqus is utilized in this research to develop the FE model of the box sign structure, providing a comprehensive platform for accurate model creation and enabling detailed mechanical and dynamic simulations. With its diverse element options, including beams, shells, and solids, Abaqus enables precise representation of the structure's geometry and material properties. Furthermore, the software facilitates the application of detailed boundary conditions to accurately reflect real-world constraints and conditions, such as fixed supports, applied loads or even thermal impacts. For modal analysis, Abaqus proves especially valuable, as it helps

determine natural frequencies and mode shapes effectively, which are crucial for evaluating the structural health of the sign structure and identifying potential vulnerabilities under dynamic loads or material degradation. Moreover, Abaqus offers strong support for various simulation methods and solvers, and through the Newton–Raphson method, it is capable of performing sophisticated nonlinear analysis and simulation. By leveraging Abaqus's capabilities, this study can perform in-depth simulation and analysis.

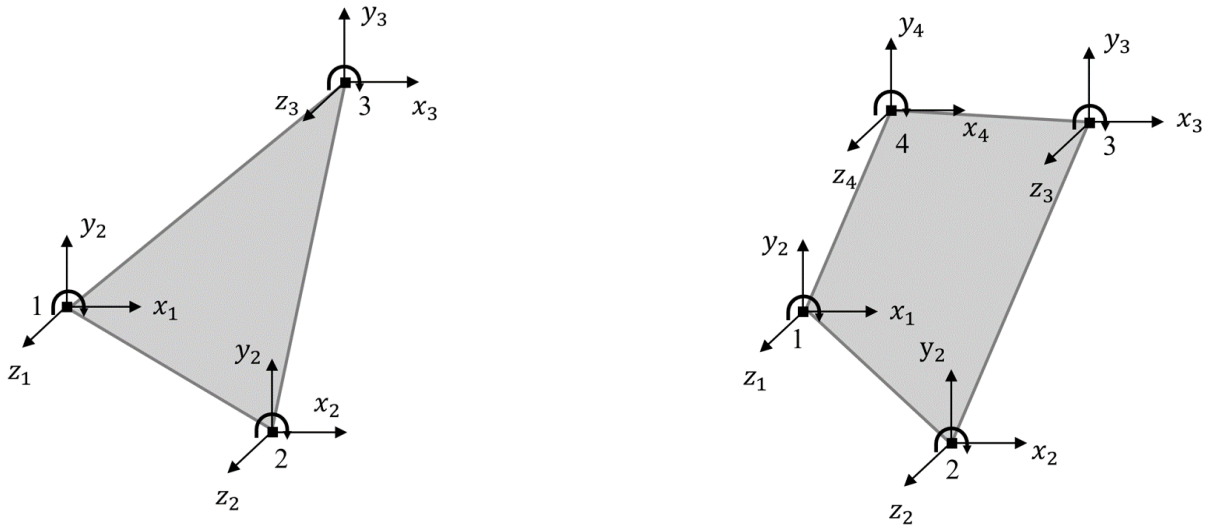
2.3.3 1D Beam Element

Traditionally, 1D elements are employed to model structures or physical phenomena that exhibit one-dimensional characteristics. In this research, the 1D beam elements are specifically used to model **slender structural members** that primarily experience bending, shear, torsion, and axial forces. As depicted in Figure 2.3, each element is defined by two nodes located at the opposing ends of the element. Each node of beam element has six degrees of freedom (DOFs), comprising three translational DOFs and three rotational DOFs (Dassault Systèmes, 2006). This configuration allows the beam element to perform complicated deformations, including bending, torsion, and axial extension. The beam element is defined and oriented along a single direction, with its cross-sectional properties such as shape, area, and moment of inertia are defined separately and has no relationship with the element's length. Beam elements are particularly effective and efficient for the applications where the structural component's length is much greater than its thickness.

Figure 2.3: 1D beam elements

2.3.4 2D Shell Element

In Abaqus, shell elements are essential for the modeling of thin-walled structures such as plate, shells, and sheet. Shell elements in Abaqus are typically represented by Triangular (Tri) and Quadrilateral (Quad) elements (Dassault Systèmes, 2006), which are comprised by 3 nodes and 4 nodes respectively. Similar to the 1D beam elements, each node in 2D element also have three translational DOFs and three rotational DOFs (refer to Figure 2.4). The versatility of shell elements lies in their ability to model both curved and flat surfaces with high computational efficiency, making them particularly useful during the analysis of thin-walled structures and other components where the thickness is negligible compared to the other dimensions like aircraft fuselages, car bodies, and steel plates. Similar to the setting of cross-sectional properties in Beam element, the thickness of the Shell Element is defined separately.



(a) Triangular element **(b) Quadrilateral element**

Figure 2.4: 2D triangular and quadrilateral elements

2.4 GENETIC ALGORITHM

The genetic algorithm (GA) is a population-based optimization method inspired by the principles of natural selection and evolutionary biology. It is particularly effective for solving nonlinear and high-dimensional problems where gradient-based methods are less effective. GA iteratively evolves solutions using operations such as selection, crossover, and mutation.

In civil engineering, GA has been applied to tasks such as FE model updating (Luo and Zhao, 2011) and structural optimization (Tang and Gu, 2002). In this study, GA is used to calibrate uncertain model parameters so that the computed modal frequencies match field-tested values. A detailed discussion of its application to the present problem is provided in Section 3.4, with the implementation code presented in Appendix B.

3 FE Model Development and Optimization

3.1 FIELD TESTING AND PRELIMINARY DATA ACQUISITION

PEER team conducted two sets of field hammer tests on in-service cantilever s igns. The first one took place on a single post cantilever structure at Russell Blvd exit in Davis, CA on February 9, 2023. The second testing was conducted on a single post cantilever structure at the Seacliff exit in Ventura, CA, on September 21, 2023. The following sections describe the testing on the Seacliff sign structure prior to its removal.

As a heavily corroded sign structure, the single post sign structure at the Seacliff exit in Ventura, California, is selected as the prototype structure. Due to severe corrosion and corresponding safety concerns, this sign structure was decided to be removed by Caltrans and replaced with a new sign structure. Before removal of this sign structure, members of the research team conducted a site visit to perform forced and free vibrations on this sign structure and to collect corresponding acceleration data.

During the field test, the research team used PEER-CENTS[1] to monitor the vibration of the sign structure when subjected to impact hammer tests, refer to Figure 3.1. The field test results, in terms of identified natural periods and corresponding natural frequencies in the X, Y, and Z directions, are presented in Table 3.1.

As Figure 3.2 shows, three PEER-CENTS were attached to the structure in different positions. Figure 3.3 shows the natural frequencies of the X-Z plane, same as in-plane (IP) direction. Figure 3.3b shows the acceleration data in the X direction collected from sensor 1 and the Figure 3.3a shows the corresponding Fourier amplitude spectrum. These results in the time and frequency domains (also supported by the coupled accelerations in the Z-direction, not shown for here brevity) consistently identify the natural period and corresponding frequency in the IP direction as $0.30\ s$ and 3.31 Hz, respectively. Similarly, the natural period and frequency in the "Out-of-Plane" (OOP), i.e., Y direction, are identified as $0.44\ s$ and 2.27 Hz. These two frequencies served as "target" frequencies to achieve for the subsequent 3D FE model development and optimization.

[1] PEER-CENTS is PEER-Cost Effective Next Technology Sensor, which is an in-house microcontroller-based 6 degrees of freedom sensor: 3 translational accelerometers for acceleration & 3 gyroscopes for angular velocity.

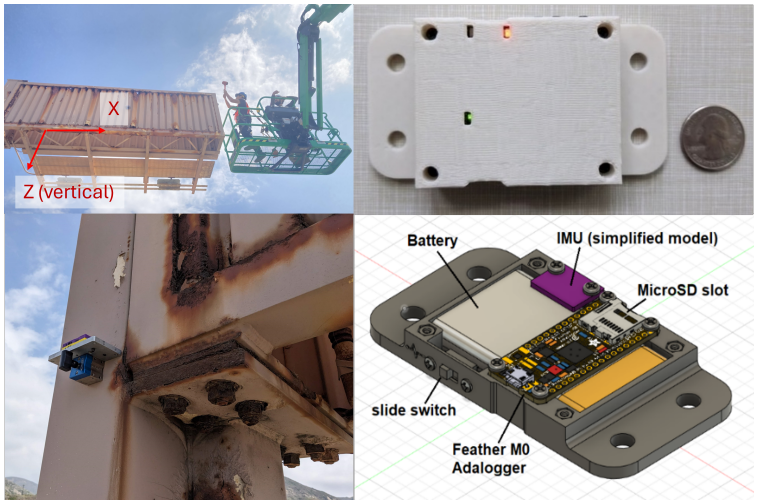


Figure 3.1: Field tests of the Seacliff sign structure using PEER-CENTS.

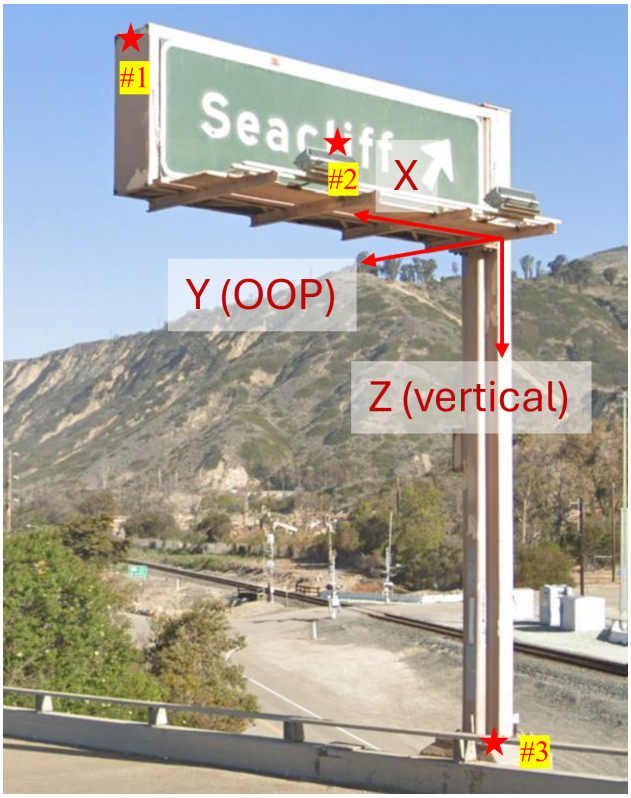
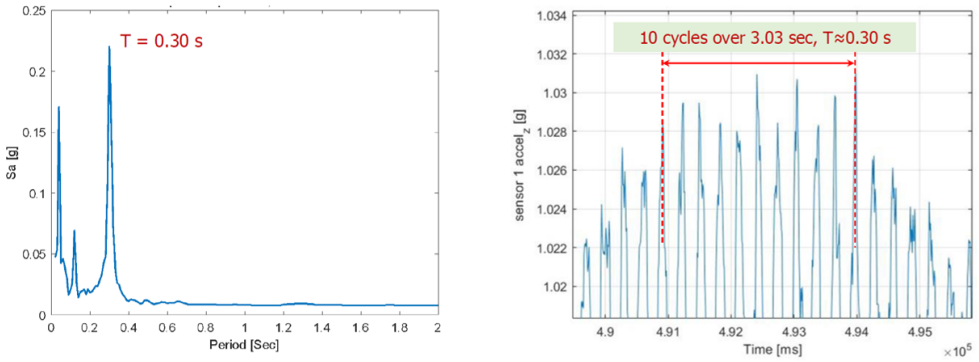


Figure 3.2: The locations of PEER-CENTS used in the field tests.

3.2 PRELIMINARY FINITE ELEMENT MODEL SETUP

The development of a FE model is essential for accurately simulating the structural behavior of overhead box beam sign structures. The model must incorporate realistic conditions to ensure



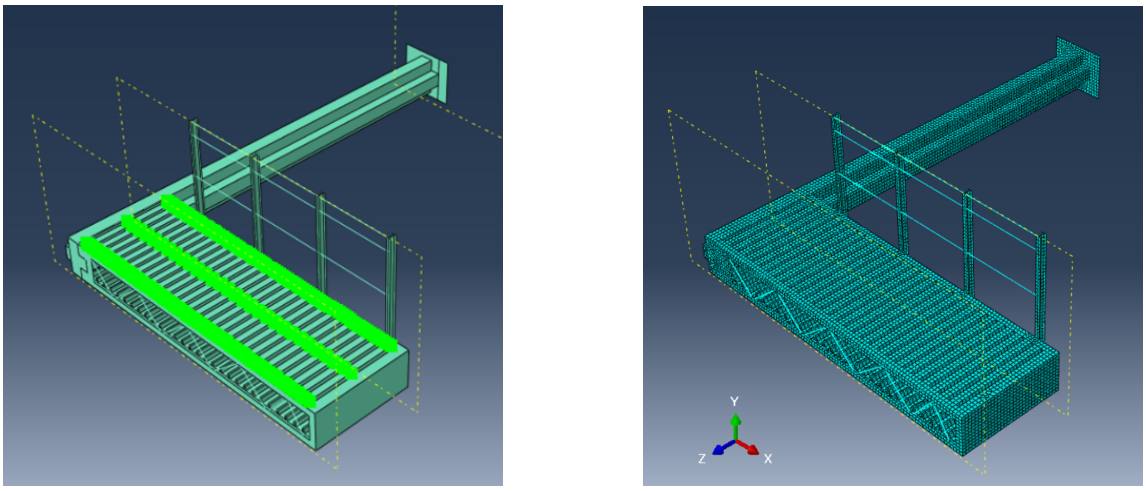
(a) Response spectrum **(b) Number of cycle counts**

Figure 3.3: Response spectrum and vibration cycles in IP direction

Table 3.1: Natural periods and frequencies of the Seacliff sign structure in the field tests.

Mode	Period [s]	Frequency [Hz]
Y direction (OOP)	0.44	2.27
$X-Z$ plane (IP)	0.30	3.31

accurate results. This section outlines the key steps involved in developing and optimizing the FE model, including geometry definition, mesh generation, material assignment, and boundary condition implementation. A preliminary Abaqus FE model was developed without accounting for corroded areas, as shown in Figure 3.4.



(a) Mass Assignment **(b) Mesh Generation**

Figure 3.4: FE model of the Seacliff sign structure, oriented horizontally

3.2.1 Geometry Definition

The sign structure was modeled as a 3D model that represents the main load-bearing components, including the overhead box beam, the post and the base plate. The dimensions and configurations were based on actual structural specifications and measurements. However, accurately measuring the thickness of the posts proved to be challenging. As a result, an initial estimate for the column thickness was used during the preliminary modeling, which was then updated using the optimization algorithm, the details of which are described in Section 3.4.

3.2.2 Mesh Generation

A structured mesh was employed in the FE model to balance computational efficiency and accuracy. During the mesh generation, a combination of shell and beam elements was adopted. For the primary structural components —such as the post, ribbed sheets, and base plates— use of triangular (S3R) and quadrilateral (S4R) shell elements proved to be both efficient and effective. This choice is particularly suitable given that the box sign structure primarily consists of thin steel sheets, and it significantly enhances the computational efficiency compared to 3D solid elements. For the brace components, the length of the wind brace is much larger than the cross-sectional dimensions. Accordingly, the beam elements (B31) in Abaqus were used for these braces, with cross-sectional geometry of the beam elements specifically defined as L-shaped (depicted in Figure 3.5). This approach of using appropriate elements for different components not only ensures accurate representation of the structural behavior but also optimizes the overall computational performance and efficiency.

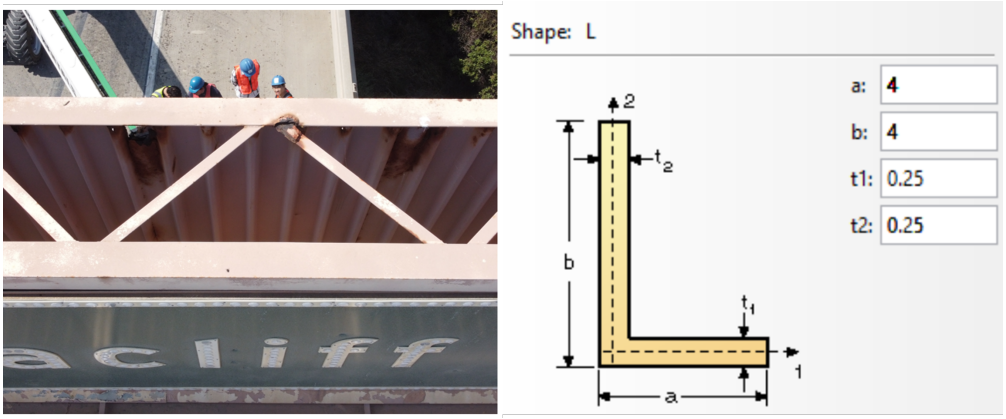


Figure 3.5: Detailed wind braces images and Abaqus beam element setting

3.2.3 Material and Mass

A nonlinear material model was used to analyze the structural behavior in Pushover and cyclic loading. Values of the material properties used in this model are presented in Table 3.2. A kinematic hardening model was adopted here, where the two pairs of yield stress and plastic strain

values correspond to the hardening behavior definition in Abaqus. Moreover, "Gravity" command in the "Load" module was used to simulate self-weight of the structure. The "Seacliff" sign itself

Table 3.2: Material properties used in the Seacliff sign structure FE model.

Material Properties		
Property	**Value**	
Mass Density		
Units [lb·s^2/in^4]	0.0007487	
Elasticity		
Elastic Modulus [ksi]	27,500	
Poisson's Ratio	0.26	
Plasticity		
Yield Stress [psi]	36,300	75,000
Plastic Strain	0	0.2287

was also included in the model (see Figure 1.1). The mass of the sign plate has a significant impact on the natural frequencies and the dynamic responses of the structure. In Abaqus, the presence of the sign plate is modeled by assigning an inertial mass to the corresponding surface of the structure (refer to the green regions in Figure 3.4a).

3.2.4 Step and Solver

In this study, "Static, Linear perturbation" in "Step" module, is adopted to extract the natural frequencies of the structure. On the other hand, "Static, General" is utilized for simulations under various loading conditions such as pushover analysis or cyclic loading.

3.2.5 Boundary Conditions

All nodes on the base plate are selected, and constraints were applied to fix all six degrees of freedom (DOFs) at each node. (Figure 3.6 represents a demonstration of this process, where the node fixation in the real model is more densely distributed), effectively anchoring the base plate to the ground. The fixed boundary condition was verified by negligible deformations measured by two displacement transducers, one horizontal and one vertical, placed between the base plate and the reaction wall.

3.2.6 Summary

Overall, the preliminary FE model has been established using all externally measurable geometric data. However, there were some unknown parameters, such as the thickness of columns, the material properties, and the corrosion impact on material strength. The following sections focus on assessing the degree of corrosion using CV techniques, refining the model with corroded locations

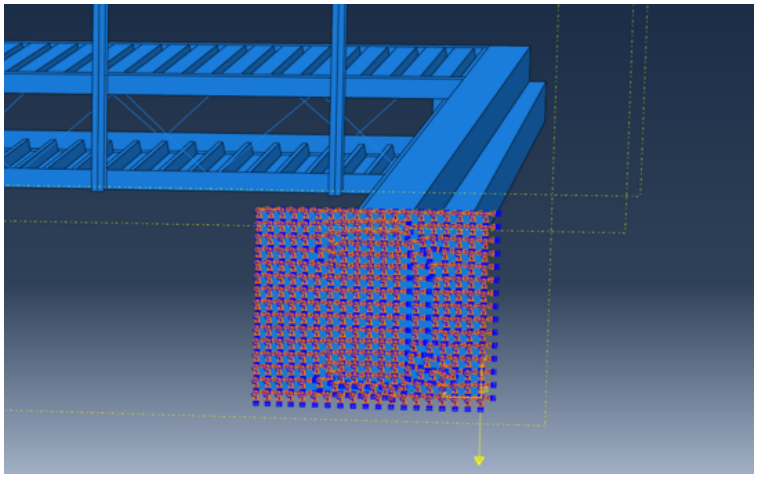


Figure 3.6: Boundary Condition in Base Plate

and optimizing the unknown parameters to match the target frequencies shown in Table 3.1 using genetic algorithm. This preliminary model served as a foundation for the continued development of the FE model, ultimately leading to a more robust and reliable assessment of the Sign Structure.

3.3 COMPUTER VISION BASED CORROSION DETECTION

The project team developed a computer vision-based algorithm for corrosion detection, capable of identifying and localizing corroded areas within raw images. Figure 3.7 illustrates the basic pipeline of identifying and localizing the corrosion. The Res-Net algorithm is employed first to determine whether corrosion is present in the images. If corrosion is detected, the U-Net model is then utilized to accurately identify and detect the corroded areas, as shown in Figure 3.8.

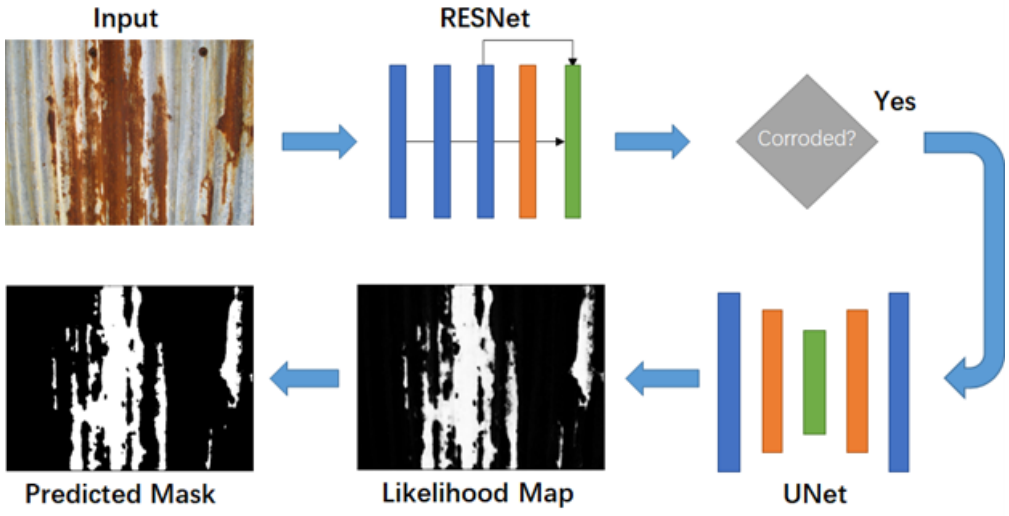


Figure 3.7: Computer Vision-Based Corrosion Detection Workflow Using RESNet and UNet

The utilization of U-Net model is highly effective in image segmentation tasks to accurately delineate the corroded regions. U-Net's architecture, with its symmetric encoder-decoder structure, enables precise localization by capturing both the contextual information from the image and the detailed features at various scales. This dual-step approach ensures both accurate detection and precise localization of corroded areas, as illustrated in Figure 3.8.

In developing this algorithm, several preprocessing steps, such as image normalization and augmentation, are also incorporated to enhance the model's robustness against variations in lighting and perspective. The models were trained and validated on a comprehensive dataset of corrosion images, achieving a high degree of accuracy and reliability. It is noted that this algorithm has potential applications in various fields, including industrial inspection and infrastructure maintenance, where early detection of corrosion can significantly reduce repair costs, prevent structural failures and increase efficiency.

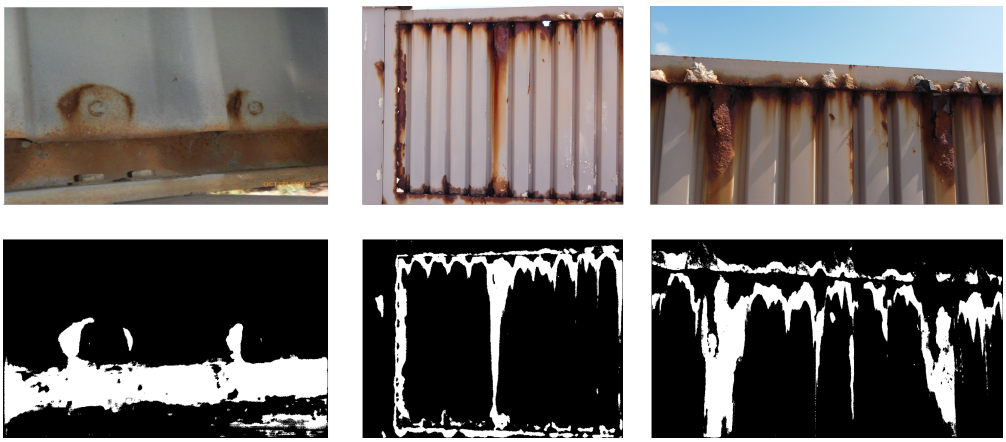


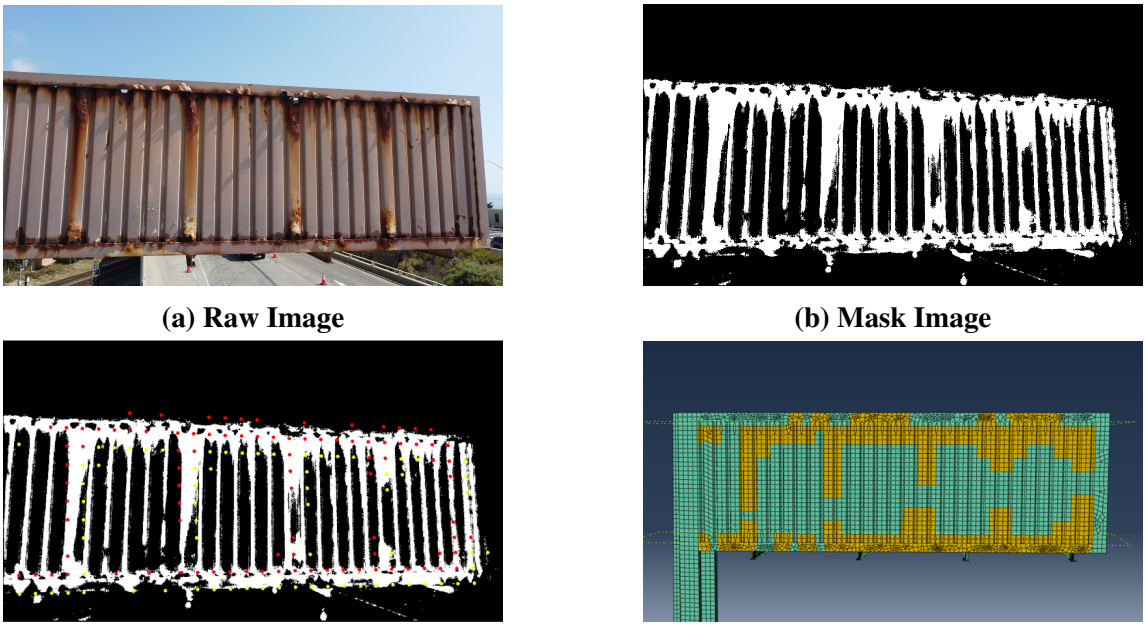
Figure 3.8: Raw and Mask Images[2]

In order to incorporate the computer vision results into Abaqus FE simulation, a python script was developed to facilitate the process of quantifying corroded areas in structural images and their incorporation into the Abaqus model. The workflow involves three main steps: image loading, manual point selection, and corroded area calculation. First, the code loads the raw image and a corresponding mask image, ensuring that both are correctly processed and their dimensions recorded. Following this, the user is prompted to manually select four points on the image that outline a region of interest (e.g., the boundary of the beam structure). This selection process is interactive, allowing the user to visually confirm the points chosen on the image. Once the points are selected, the code organizes them to form a valid quadrilateral, ensuring the region is properly defined. The pixel width and height of this quadrilateral are calculated and then projected and mapped into the real dimension of the beam. Finally, this rectangle is subdivided into a 30×10 grid of smaller rectangles, which provides element sizes consistent with the FE mesh ensuring that the corrosion pattern can be properly mapped into the FE model.

Since the corrosion patterns are inherently irregular, we adopted a pixel-based projection approach to map the detected corrosion onto the predefined 300-cell grids. For each grid cell, the

[2] A mask image consists of only two pixel values, 0 and 1 for black and white, respectively.

proportion of the white pixels (representing corroded areas in Figure 3.9b) was calculated. If the corroded (white) area exceeded 50% of the cell, the entire grid cell was classified as corroded; otherwise, it was classified as uncorroded. A python script (refer to Appendix A.2) was implemented to map the grid-based corrosion information into the FE model, ensuring the proper alignment with the location and geometry of the beam. The identified corroded regions were subsequently assigned modified material properties and section definitions as discussed in Section 7.1. This approach enables accurate simulation of the structural behavior under corrosion effects, and allows for a flexible and interactive method of analyzing specific areas within images. This method is particularly beneficial in cases where automated detection may be challenging due to the irregular shapes of corrosion. Figure 3.9 illustrates the overall workflow of detecting corrosion from the raw images and mapping the corrosion pattern into the Abaqus model.



(a) Raw Image

(b) Mask Image

(c) Mask Image with Annotated Cells

(d) Corrosion Reflection in Abaqus

Figure 3.9: Steps of incorporating the areas of corrosion into the FE model.

3.4 MODAL UPDATING USING GENETIC ALGORITHM

3.4.1 Unknown Parameters and Target Frequencies Setting

An optimization algorithm was used to update the unknown model parameters (refer to the examples shown in Figure 3.10) by matching the natural frequencies from the FE model and the field tests. The results are summarized in Table 3.3.

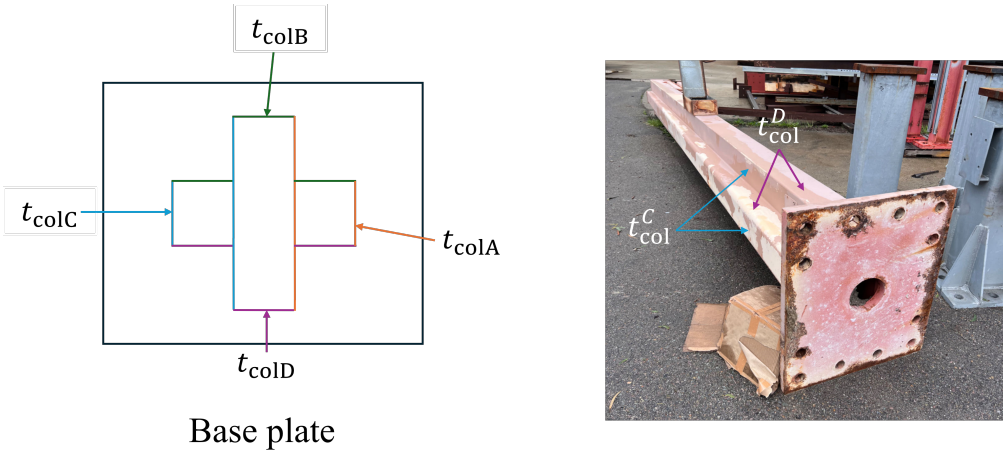


Figure 3.10: Example parameters of the sign structure to be optimized.

3.4.2 Genetic Algorithm and Population Initialization

As previously explained in Section 2.4, the model was updated using a genetic algorithm for optimization. To maximize the search space and avoid local optima, it is crucial to maintain the randomness of the initial population. Therefore the following equation was used to generate the population.

$$\mathbf{G}_{\text{init}} = \mathbf{p}_{\text{lb}} + \mathbf{r}\,(\mathbf{p}_{\text{ub}} - \mathbf{p}_{\text{lb}}) \tag{3.1}$$

where $\mathbf{G}_{\text{init}}$ represents the initial generation's parameter vector, $\mathbf{p}_{\text{lb}}$ and $\mathbf{p}_{\text{ub}}$ denote the lower and upper bounds of the parameter vector respectively, with each element corresponding to the bounds of a specific modal parameter. And $\mathbf{r}$ is a random vector with the same dimension as the parameter vector, where each element is a value between 0 and 1.

3.4.3 Adaptation Evaluation

The optimization algorithm tracks and minimizes the Root Squared Error (RSE) to determine a more suitable set of parameters. The RSE is expressed as follows:

$$RSE = \sqrt{\left[(f_{\text{model,IP}} - f_{\text{test,IP}})^2 + (f_{\text{model,OOP}} - f_{\text{test,OOP}})^2\right]} \tag{3.2}$$

The results of the RSE before and after optimization of parameters are listed in Table 3.4, where f_{test} is the measured frequency from field tests, treated as the target frequency of the optimization algorithm. On the other hand, f^{i}_{model} and f^{o}_{model} are respectively the initial and optimized frequency values before and after application of the optimization algorithm with respective values of RSE^{i}_{model} and RSE^{o}_{model}, indicating significant reduction of the RSE and improved accuracy of the FE model, upon updating the values of the unknown model parameters with the optimization algorithm.

3.4.4 Crossover and Mutation

Crossover and mutation are critical operations in the genetic algorithm that generate new solutions, allowing the algorithm to explore the parameter space. In the context of this study, these operations are applied to a population of candidate solutions, where each individual represents a complete set of structural parameters, including variables such as the column wall thickness and the elasticity modulus of corroded material.

The crossover operation combines two parent individuals to create offspring, enabling exploration of new regions within the solution space. Let $\mathbf{G}_1$ and $\mathbf{G}_2$ denote two parent individuals, each containing a vector of model parameters. An offspring generated through crossover is computed as:

$$\text{Offsprings} = \mathbf{G_1} + \mathbf{m}(\mathbf{G_2} - \mathbf{G_1}) \tag{3.3}$$

where $\mathbf{m}$ is a vector of mixing factors that determines the contribution of each parent to the offspring's genetic material. The mixing factors $\mathbf{m}$ are controlled by the crossover rate R_{CO}, which determines the proportion of genes that are blended between the two parents. This mechanism allows certain parameters, such as material stiffness or geometry dimensions, to blend between two candidate solutions, potentially leading to improved frequency alignment with the field data.

To further maintain diversity within the population and avoid premature convergence, the mutation operation is applied to individuals with a certain probability, known as the mutation rate R_{RM}. During mutation, a small random perturbation is added to the individual's genetic material, enabling the algorithm to explore the solution space more thoroughly. The mutation operation can be described as:

$$\mathbf{G}'_k = \mathbf{G}_k + \epsilon \mathbf{G}_k \tag{3.4}$$

where ϵ is a small random perturbation bounded by ϵ_{bound}. This perturbation allows the algorithm to escape local optima and search for a more optimal solution globally. Through the combination of crossover and mutation, the genetic algorithm can effectively balance exploration and exploitation, enhancing the likelihood of finding the global optimum in complex optimization problems.

By iteratively applying crossover and mutation across generations, the algorithm refines the parameter vectors to minimize the frequency mismatch between the model simulations and field testing results. The genetic algorithm is fully integrated with the Abaqus Python API, where each individual parameter is used to update the FE model, run the simulation, and extract modal frequencies for evaluation. Details of the implementation and code are provided in Appendix B.

3.4.5 Results and Summary

The results of the genetic algorithm applied to the modal updating problem are summarized in Tables 3.3 and 3.4. The algorithm successfully optimized the FE model parameters to closely align the predicted modal frequencies with the experimental data.

Table 3.3: Summary of values of original and optimized parameters.

Symbol	Parameter	Initial Value	Bounds	Optimized Value
t_{colA}	t_{colA} [in]	0.800	[0.64, 0.96]	0.704
t_{colB}	t_{colB} [in]	0.200	[0.16, 0.24]	0.201
t_{colC}	t_{colC} [in]	0.800	[0.64, 0.96]	0.824
t_{colD}	t_{colD} [in]	0.200	[0.16, 0.24]	0.160
$E_{\text{uncorroded}}$	$E_{\text{uncorroded}}$ [ksi]	27,500	[22,000, 33,000]	22,237
$\nu_{\text{uncorroded}}$	Poisson ratio of steel	0.26	[0.208, 0.312]	0.24
η_{E}	Ratio of elastic moduli	0.40	[0.2, 0.6]	0.35

As shown in Table 3.3, the optimization process led to significant adjustments in the model parameters. For example, the thickness of column surface A (t_{colA}) was reduced from its initial value of 0.800 in to an optimized value of 0.704 in. Similarly, the elastic modulus of steel ($E_{\text{uncorroded}}$) was adjusted from 27,500 ksi to 22,237 ksi. In addition, the ratio of elastic moduli, defined as $\eta_{\text{E}} = E_{\text{corroded}}/E_{\text{uncorroded}}$, was optimized from 0.40 to 0.35, reflecting the algorithm's ability to reconcile the model predictions with the experimental data.

Table 3.4 presents the root mean squared error (RSE) between the test frequencies and both the initial and optimized model frequencies for the out-of-plane (OOP) and in-plane (IP) modes. The initial model showed an RSE of 0.58, with the out-of-plane frequency ($f^i_{\text{model}} = 2.69$ Hz) and in-plane frequency ($f^i_{\text{model}} = 3.71$ Hz) both deviating notably from the experimental values. After optimization, the RSE was significantly reduced to 0.0072, with the optimized model frequencies ($f^o_{\text{model}} = 2.28$ Hz for OOP and $f^o_{\text{model}} = 3.31$ Hz for IP) closely matching the experimental data.

Table 3.4: Root Mean Squared Error (RSE) for initial and optimized model frequencies.

Mode	f_{test} [Hz]	f^i_{model} [Hz]	RSE^i_{model}	f^o_{model} [Hz]	RSE^o_{model}
OOP	2.27	2.69	0.58	2.28	0.0072
IP	3.31	3.71		3.31	

The results demonstrate the effectiveness of the genetic algorithm in improving the fidelity of the FE model. The updated parameters not only brought the model's modal frequencies into closer agreement with the experimental measurements but also significantly reduced the overall error. These findings validate the application of genetic algorithms in the modal updating process, particularly for complex structures where traditional methods may struggle to achieve similar levels of accuracy.

4 3D Model Idealized Loading and Pre-test Simulation

Several analyses were conducted for pretest simulations and are described below. Cases I-IV are linear elastic analyses with different loading configurations conducted to obtain the corresponding stiffness values. Some of these stiffness values were used to convert the force amplitudes initially provided by Caltrans to displacement amplitudes for a more reliable control of the test program. On the other hand, cases V and VI are nonlinear pushover analyses in the IP vertical and OOP directions, respectively, to characterize the force-displacement relationships of the structure in the corresponding directions and to estimate the yield points and force capacities for selecting the proper actuators and the instrumentation to use in testing. This preliminary simulation was conducted solely for the design of the laboratory testing, while the actual experimental results and subsequent analysis are presented in Chapters 6 and 7.

4.1 CASE I: GRAVITY LOAD

In this case, the analysis was conducted using the dead load of the structure. It was used as a basic analysis case for checking several key parameters prior to finalizing the test program, e.g., checking the values of the weight of the structure and its corresponding displacement. Beam weight in Table 4.1 was calculated as the vertical reaction at the base of the column. The specified displacement in Table 4.1 is the vertical one at the tip of the beam. For a cantilever beam of span length of 15 ft (180 in), refer to Figure 4.1, the deflection due to self-weight to span ratio is $0.60/180 \approx 1/300$, which is a reasonable value that is expected under the self-weight of steel cantilever beams[1]. The corresponding IP vertical stiffness due to *distributed* load in the *vertical* direction was also computed and reported in Table 4.1 as $k_{IP}^{dv} = 7,452$ lb/in. This value is computed in this manner to be used in Section 5.2 to define the loading protocol in terms of displacement amplitudes.

[1] The deflection limit for a steel cantilever beam in terms of its span length L is usually $L/180$ for live load and $L/90$ for combined dead and live load. These limits are further reduced according to current codes widely adopted in the U.S. if there are nonstructural components or brittle materials (such as walkways or parapets) attached to the structural element. It is to be noted that according to the *Standard Specifications for Structural Supports for Highway Signs, Luminaires, and Traffic Signals*, Sixth Edition, 2013 (LTS-6), Section 10.4.1, this limit is $L/150$.

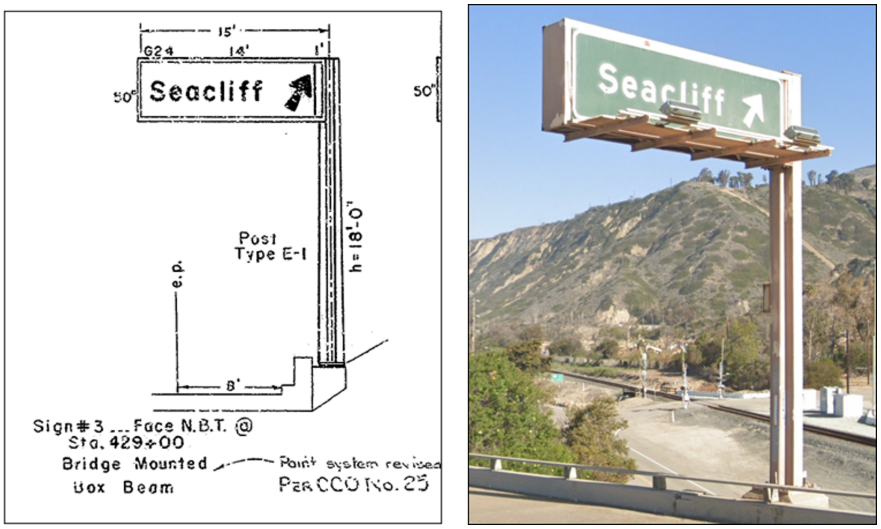


Figure 4.1: Seacliff sign structure geometry and location.

Table 4.1: Results for case I simulations including the IP vertical stiffness.

Beam weight [lb]	Displacement [in]	IP stiffness [lb/in]
4,455	0.60	7,452

4.2 CASE II: POINT LOAD IN VERTICAL DIRECTION

The laboratory test was initially planned to use a vertical actuator at the tip of the beam. To compute the stiffness in this configuration, a vertical point force was similarly applied in the Abaqus model and the corresponding displacement was calculated. The results are presented in Table 4.2, where the IP stiffness due to *point* load in the *vertical* direction is $k_{IP}^{pv} = 2,799.3$ lb/in.

Table 4.2: Results for case II simulations including the IP vertical stiffness.

Force [lb]	Displacement [in]	IP stiffness [lb/in]
1,000	0.357	2,801
3,000	1.07	2,804
5,000	1.79	2,793

4.3 CASE III: POINT LOAD IN OOP DIRECTION

The wind forces act on the sign structure in the OOP direction as pressure, however it is possible to apply equivalent displacements at the tip of the beam in the OOP direction that causes similar strains at critical locations due to the distributed pressure. This displacement can be applied using a single actuator placed at the tip of the beam along the OOP direction and this configuration can

overcome several challenges that can occur in the case of applying pressure. Case III considers this configuration of a point load applied at the tip of the beam in the OOP direction. In this loading case, two stiffness values were computed using the OOP displacement at either the tip or the middle of the beam to give the results listed in Tables 4.3, respectively. From these results, the respective point load OOP stiffness values are $k_{OOP}^{pt} = 534.5$ lb/in.

Table 4.3: Results of case III simulations including the OOP stiffness (tip displacement).

Force [lb]	Displacement [in]	OOP stiffness [lb/in]
1,000	1.88	532
2,000	3.70	541
3,000	5.65	531

4.4 CASE IV: WIND LOAD IN OOP DIRECTION

This case considers the application of wind effect in the OOP direction as pressure. Stiffness was computed as pressure multiplied by surface area, $A = 9756.47$ in^2, divided by either the tip (Table 4.4) or middle point displacement. From these results, the respective distributed load OOP stiffness value is $k_{OOP}^{dm} = 1,656.5$ lb/in.

Table 4.4: Results of case IV simulations including the OOP stiffness (tip displacement).

Pressure [psi]	Displacement [in]	OOP stiffness [lb/in]
0.5	2.94	1,659
1.0	5.90	1,654

4.5 CASE V: PUSHOVER ANALYSIS IN VERTICAL DIRECTION

Pushover analysis was conducted in the vertical direction by increasing the vertical tip displacement in increments and determining the corresponding force at the same location. Figure 4.2 shows the resulting pushover curve. The yield point is computed as the point where the tangent stiffness reduces by 5% of the initial. With this definition, yielding occurred around 1.3 in (blue x in Figure 4.2), i.e., $(5.5/12)\,/[18-(50/12)/2] = 2.88\%$ drift, which is significantly large[2]. However, using the most stringent requirement of 2.5% of H, i.e., $H/40$ according to the *Standard Specifications for Structural Supports for Highway Signs, Luminaires, and Traffic Signals*, Sixth Edition, 2013 (LTS-6), Section 10.4.2 and conservatively taking $H = 18$ ft, Figure 4.1, the allowable displacement is 5.4 in. Assuming yielding is about 1.0 times the allowable displacement (considering the severely damage status of the Seacliff sign structure), the "expected yield" displacement is about 5.4 in, which is practically the same as the value determined by the FE computations. This relatively large value of 5.5 in can be attributed to the low stiffness of the sign structure, where the yield force of 10 kip occurs at a relatively large displacement. The current Abaqus model includes nonlinear material and geometric nonlinearity and includes reduced steel modulus of elasticity for the visually identified corroded locations. However, at this stage, it does not consider any reduction of the corroded bolt strength at the connections or other connection effects. If there is any damage due to corrosion in the connection, yield can occur at a smaller load level and this is explored later in the laboratory tests. The peak force is observed to be around 15 kip, which guides the selection of the actuator to be used in the test setup. The vertical displacement that needs to be applied is expected to be large, i.e., at least 4 times the yield displacement, i.e., > 22 in, indicating the need for a long-stroke actuator.

4.6 CASE VI: PUSHOVER ANALYSIS IN OUT-OF-PLANE DIRECTION

Figure 4.3 shows the pushover curve and the yield point in the OOP direction. As expected, the stiffness and force capacity in the OOP direction are smaller than those in the vertical direction. The stiffness in the OOP direction is almost 1/3 of that in the vertical direction (625 lb/in vs. 1,818.2 lb/in). Because of this lower stiffness, the yield displacement is significantly large in the OOP direction, around 6.8 in (3.6% drift). Even very rare wind loading conditions (i.e., 3,000 year wind) may not cause major damage (corresponding displacement is only about 5 in, i.e., less than

[2] The ASCE 7-16 standard does not suggest an allowable drift limit for wind design as it does with a seismic design but, according to the non-mandatory Appendix CC (Serviceability Considerations) of ASCE 7-16, common usage for *building design* is on the order of 1/600 to 1/400 of the building or story height without more details. Typical wind drift limits in common usage vary from H/100 to H/600 for total building drift and h/200 to h/600 for interstory drift, depending on building type and the type of cladding or partition materials used. The most widely used values are H for building height (or h for story height)/400 to H (or h)/500 (ASCE Task Committee on Drift Control of Steel Building Structures, 1988). An absolute limit on interstory drift is sometimes imposed by designers in light of evidence that damage to nonstructural partitions, cladding, and glazing may occur if the interstory drift exceeds about 0.4 in (10 mm). It is to be noted that the above discussion is only for comparison because it predominantly pertains to deflection limits intended for buildings. These may not be applicable to the sign structures. This is because most building structures of similar height to the sign structures are considered rigid while the design of sign structures typically assumes flexible structure.

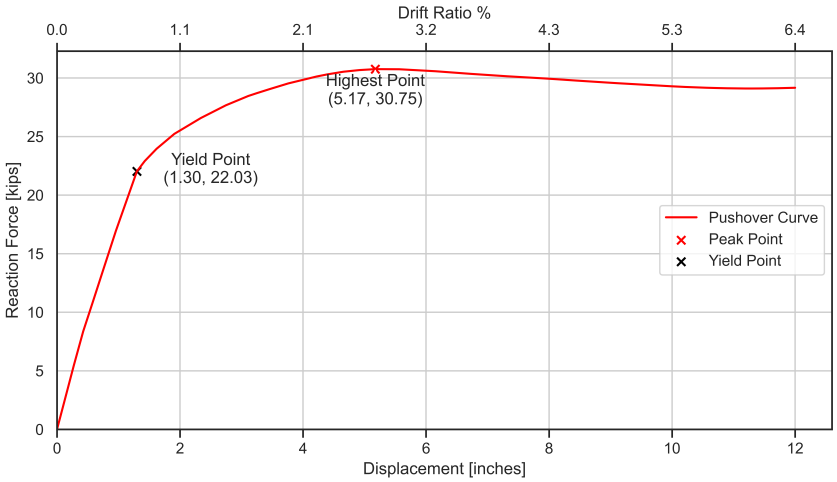


Figure 4.2: Pushover analysis results in the vertical (IP) direction.

the yield displacement in this direction) to this sign structure. It is noted that this observation was based on the pre-test FE model results, where the OOP loading in the tests demonstrated similar behavior.

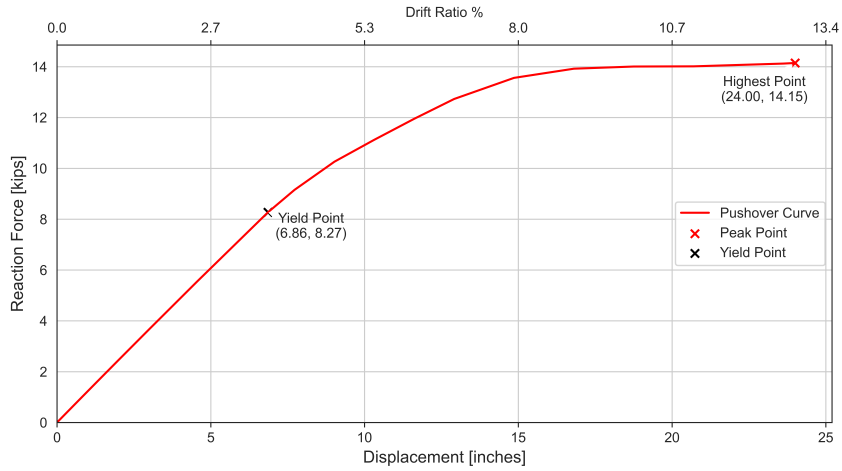


Figure 4.3: Pushover analysis results in the horizontal (OOP) direction.

4.7 TIME HISTORY ANALYSIS

Because of the L-shaped geometry of the sign structure, the beam displaces in the vertical direction due to possible IP earthquake ground motion, even if the vertical component is not applied. For demonstration of this seismic response, a structural analysis model of the sign structure was developed in OpenSees using elastic line elements and was subjected to the Tarzana ground motion

of the 1994 Northridge earthquake as shown in Figure 4.4, where both IP lateral and vertical components were applied at the base of the sign structure. This model used the correct geometry and some parameters were tuned to match the IP natural frequency identified from the field tests, i.e., 3.31 Hz (Table 3.1). Rayleigh damping was used with coefficients calculated with 2% damping ratio in the first two modes. The time history of vertical displacement at the tip of the beam is shown in Figure 4.5 with a peak displacement close to 5.5 in, which demonstrates that this type of earthquake loading can be viewed as a major cause of the vertical response of the beam of this sign structure. The objective of this preliminary analysis with the elastic model was to explore the level and order of magnitude of the vertical displacement at the tip of the beam due to ground motions. Higher levels of displacement are expected in an inelastic analysis and nonlinear time-history simulations will be conducted in future studies using the higher fidelity Abaqus model.

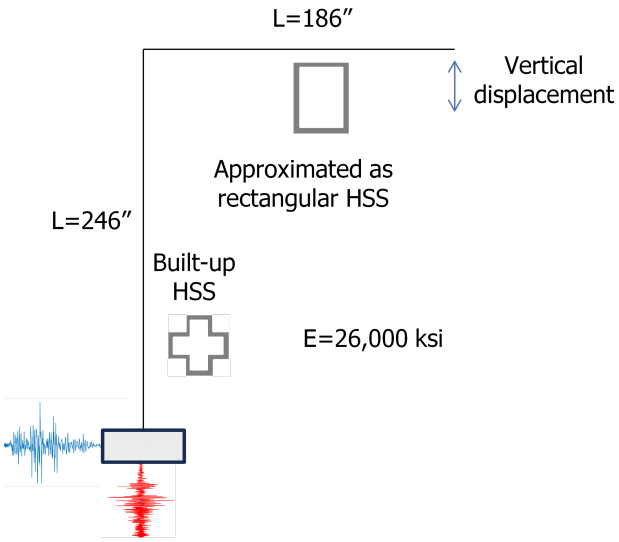


Figure 4.4: OpenSees model used for demonstrating the seismic response of the sign structure.

4.8 SUMMARY AND LOADING PROTOCOL SELECTION

By comparing Tables 4.2 and 4.3, we observe that the stiffness in the in-plane (IP) direction is greater than that in the out-of-plane (OOP) direction. Further comparison of the two modes under pushover analysis (Figures 4.2 and 4.3) reveals that, although the IP direction exhibits significantly higher stiffness than the OOP direction, the yield point occurs much earlier. This indicates that the IP direction is more prone to yielding and inelastic response compared to the OOP direction.

The loading protocol and force amplitudes recommended by Caltrans are presented in Table 4.5. As observed, certain force amplitudes in the vertical direction exceed the force capacity determined by the corresponding pushover curve (Figure 4.2). Based on the results of the preliminary FE model and the interest to explore the response of the sign structure in vertical direction,

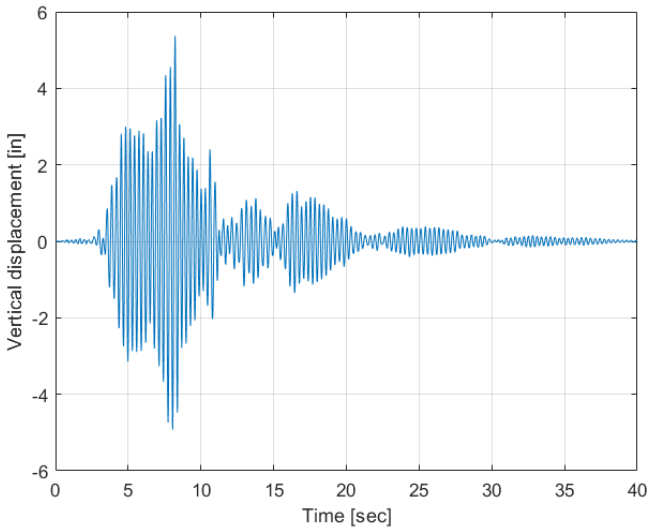


Figure 4.5: Time history of beam tip vertical displacement due to applying IP and vertical components of the Northridge Tarzana ground motion.

where reasonable levels of displacements can occur due to earthquake ground motion, a cyclic loading protocol was developed in the vertical direction, with OOP forces increased at a few cycles, which is discussed in detail in the next chapter.

Table 4.5: Amplitudes of force suggested by Caltrans & corresponding calculated displacements.

Group	Set	Definition	IP Vertical [kip or in]			OOP [kip or in]		
			Force	Displ.	Actuator	Force	Displ.	Actuator
Low level cycles	L0	Self weight	3.9	1.20	0.00	0.0	0.00	0.00
	L1	Self weight+500 lb LL[a]+nominal wind	4.4	1.36	0.15	0.3	0.31	0.31
Working stress cycles	W0	Self weight	3.9	1.20	0.00	0.0	0.00	0.00
	W1	Self weight+10 year wind	3.9	1.20	0.00	1.9	1.78	1.78
	W2	Self weight+25 year wind	3.9	1.20	0.00	2.4	2.19	2.19
	W3	Self weight+50 year wind	3.9	1.20	0.00	2.7	2.53	2.53
Strength cycles	S0	Self weight	3.9	1.20	0.00	0.0	0.00	0.00
	S1	Self weight$\times 2.2 + 100$ year wind	8.6	2.65	1.44	3.1	2.89	2.89
	S2	Self weight$\times 4.4 + 300$ year wind	17.2	5.29	4.09	3.8	3.52	3.52
	S3	Self weight$\times 6.2 + 700$ year wind	24.2	7.46	6.26	4.4	4.04	4.04
	S4	Self weight$\times 8.2 + 1,700$ year wind	32.0	9.87	8.66	5.0	4.63	4.63
	S5	Self weight$\times 9.5 + 3,000$ year wind	37.1	11.43	10.23	5.4	5.03	5.03

[a] LL = Live load.

5 Test Setup, Loading Protocol, and Instrumentation

5.1 TEST SETUP AND LOADING PROTOCOL

5.1.1 Assembling the Sign Structure

As discussed earlier, due to severe corrosion and corresponding safety concerns, Caltrans decided to remove the Seacliff sign structure from its location and replace it with a new sign structure. The removed sign structure was then transported to the PEER laboratory for the experimental program. The geometry of the sign structure did not allow it to be transported as a single unit. Therefore, it was decided to disconnect the box beam from the post at the bolted connection. However, it was not possible to remove the bolts in the field conditions and it was needed to cut the box beam close to the connection. Figure 5.1 shows the cut box beam and the post as two separate units. To be able to conduct the tests, these two pieces needed to be welded together. As shown in Figure 5.1, there were three main components that needed to be welded, including four perimeter angles (shown in red box labeled 1 in the figure), two diagonal angles (shown in red box 2), and two ribbed sheet metal pieces (shown in red box 3).

Figure 5.1: The box beam and post separated during removal of the Seacliff sign structure from its location

The weld length and thickness needed to be designed such that the welded sections did not form a weak link and the welded sections were stronger than the connected pieces. Appendix D provides detailed calculations for characterizing the minimum welding thickness required to ensure such weld strength. As a result of these calculations, Table 5.1 summarizes the minimum welding thickness values for different components. Figure 5.2 shows various images of the welded sign structure.

Table 5.1: Weld thickness required for different components

Set	Thickness [in]
Perimeter Angle	0.230
Diagonal Angle	0.140
Ribbed Sheet	0.125

Figure 5.2: Sign Structure after welding

5.1.2 Test Setup

For lab safety and convenience, it was decided to test the sign structure in the horizontal configuration. A drawing was prepared for the design and setup of the experiments in this configuration (Figure 5.3). As illustrated in this figure, the sign structure was mounted on the reaction wall horizontally with an adapter plate. The IP loading was applied with an MTS actuator with load and displacement capacities of $\pm$150 kips and $\pm$20 inches, respectively (Figure 5.4). The actuator applied the load to the third I-shaped beam (counting from the post toward the tip) of the frame using a loading clamp, ensuring that localized damage at the frame's tip is avoided.

To apply cyclic loading to the structure, a loading clamp was installed on the frame (Figure 5.5). The clamp consists of three loading plates held together with four threaded rods. The rods were slightly prestressed such that the loading plates were snugged to the top surface of the frame and the top and the bottom surfaces of the I-beam. Two spacers were inserted between the loading plate in the middle and the top surface of the I-beam to avoid loading the diagonal angle trusses directly.

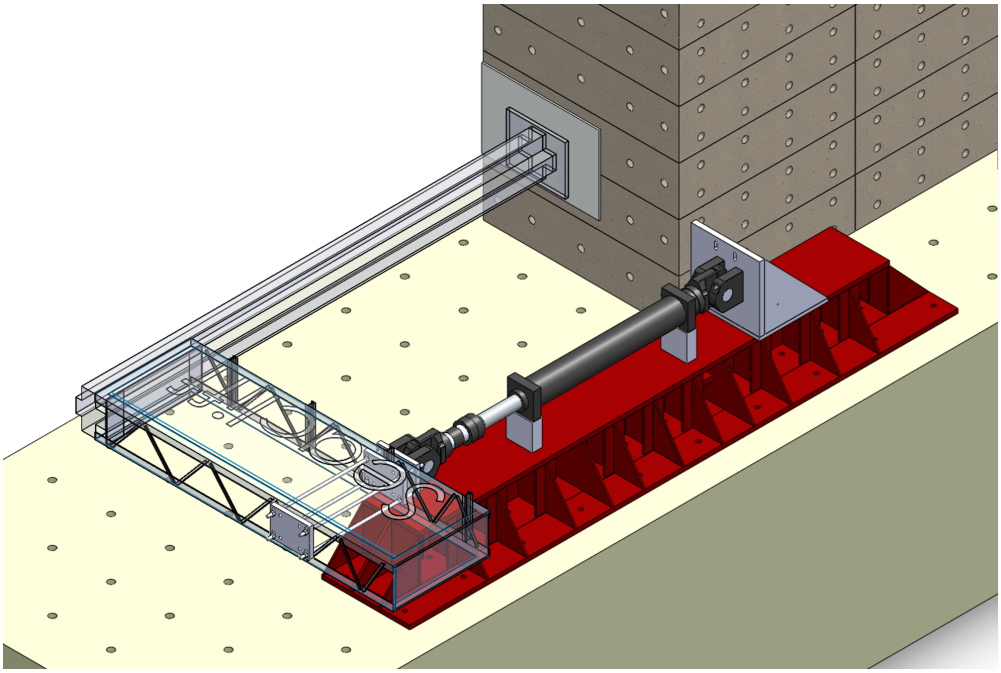
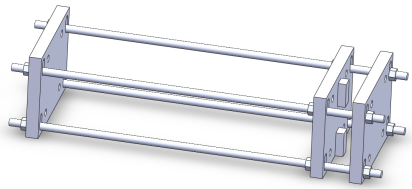


Figure 5.3: Test Setup

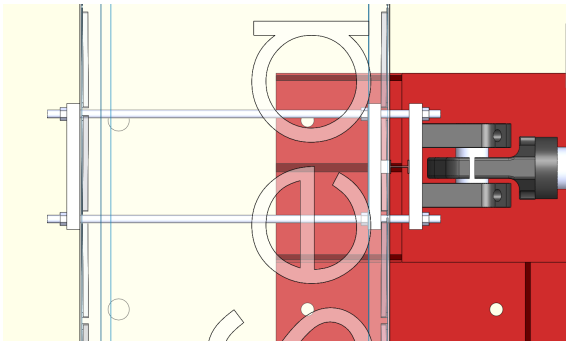
Figure 5.4: The actuator in PEER lab

5.2 LOADING PROTOCOL

As discussed in Section 4.7, significant vertical displacements of the box beam occur due to horizontal and vertical ground shaking. To represent these displacements, a cyclic displacement protocol is developed according to FEMA-461 (Federal Emergency Management Agency, 2007), which



(a) Perspective View



(b) Installation of Loading Clamp on the Frame

Figure 5.5: Loading Clamp

provides guidance on test protocols for seismic response of structural and nonstructural components. It is noted that no additional gravity or wind loads exist in the vertical direction. The displacement protocol consists of two cycles at each amplitude (see Figure 5.6), with the initial two cycles having a displacement amplitude of 0.2 inches. Subsequent cycles' amplitudes were scaled by a factor α relative to the previous cycle. When the amplitude is less than 4 inches, which is the estimated yield displacement, the scaling factor α is set to 1.4, while for amplitudes equal to or greater than 4 inches, α is reduced to 1.2 to capture more cycles and major events in the inelastic range. The first two low-level cycles are used to verify that the test setup, controller, data acquisition system, and instrumentation function as intended, before proceeding to higher levels. A loading rate of 0.04 in/s was used.

To assess whether larger displacements need to be included in the protocol, a pushover analysis was conducted using the developed pre-test FE model by applying displacements at the location of the actuator. This analysis aims to evaluate the adequacy of the existing protocol under larger displacement conditions, reflecting the demands of realistic earthquake loading scenarios.

Wind loading was applied as distributed loading in the OOP direction. The total wind force in an event with a 3,000-year return period in California was calculated as 5.4 kips. The total weight of the box beam (including the walkway), sign, loading clamp, and actuator weight were 5.6 kips (Table 5.3), already higher than the planned OOP load, therefore additional loads (e.g., lead weights) were not applied in the OOP direction. These measured weights were also used in Abaqus for accurate application of the forces in the FE model.

5.3 SENSORS OVERVIEW AND INSTALLATION

5.3.1 Strain Gauges

A strain gauge is a sensor to measure strain on the surface of an object. When the object undergoes slight deformation due to an external force or pressure, the strain gauge stretches or compresses along its direction, causing a change in its electrical resistance and influence the current. By measuring the change of the current, the strain in that location can be recorded. For such measurement,

Table 5.2: Displacement protocol with amplitude and number of cycles listed at each level

Number	Amplitude	# of Cycles
1	0.200	2
2	0.280	2
3	0.392	2
4	0.549	2
5	0.768	2
6	1.076	2
7	1.506	2
8	2.108	2
9	2.952	2
10	4.132	2
11	4.959	2
12	5.950	2
13	7.140	2
14	8.569	2
15	10.282	2
16	12.339	2

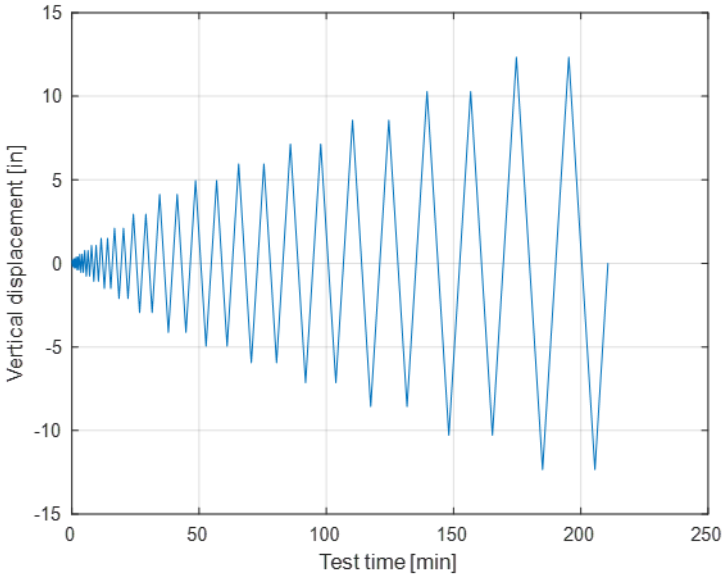


Figure 5.6: Utilized displacement protocol

the strain gauge needs to be perfectly bonded to the structure. On a steel structure, the installation location must first be thoroughly polished to ensure a flat and clean surface, where the strain gauge can be firmly attached.

The strain gauges were installed on the sign structure mainly in five regions: the post

Table 5.3: Weights of sign structure components and the applied OOP load.

Components	Weight [kg]	Weight [kips]
Box Beam including the walkway	890	1.96
Loading clamp	270	0.60
Sign	50	0.11
Actuator	1,340	2.95
Total OOP load	2,550	5.62
Post	1,485	3.27

bottom and top, perimeter angles, diagonal angles, and the ribbed sheet metal. Figure 5.7 illustrates the locations of all 17 strain gauges placed on the corroded sign structure, which are used to capture key strain responses during vertical cyclic loading. Two types of strain gauges are used: linear

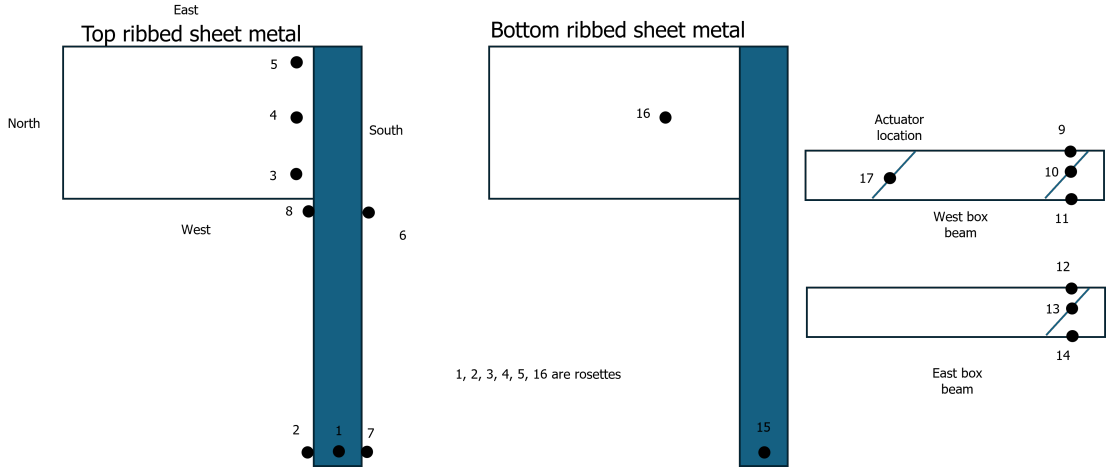
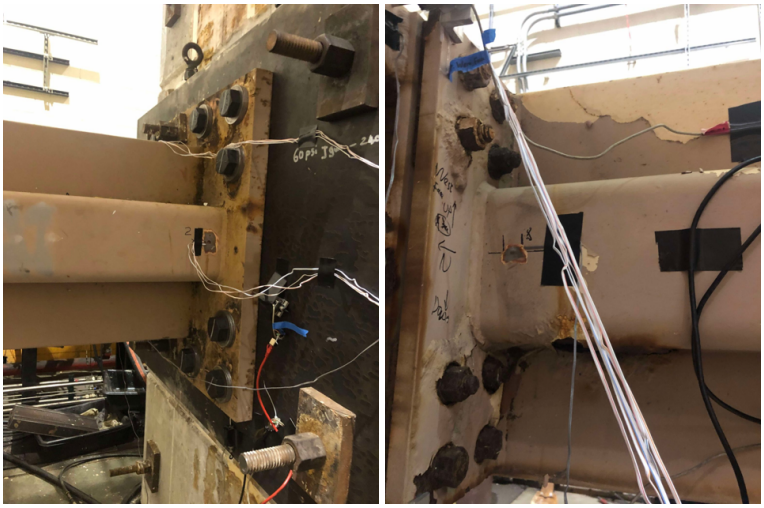


Figure 5.7: Strain gauge distribution overview of corroded specimen

strain gauges (e.g., 6, 8, and 15) and strain rosettes (1, 2, 3, 4, 5 and 16). Linear strain gauges were placed at locations where strain in a certain direction (e.g., axial strain) is expected to be the major strain component, such as those at the perimeter and diagonal angles (e.g., 9, 10, and 11). Rectangular strain rosettes were installed on the ribbed sheet metal and bottom of the post (1, 2 and 3) to measure multidirectional strains at $0°$, $45°$, and $90°$, enabling the determination of principal and shear strains. Figure 5.8 shows the strain gauges placed on the post, and Figure 5.9 illustrates the rectangular strain rosettes attached to the ribbed sheet metal.

A strain rosette is a tool designed to measure strain in multiple directions on a surface. It typically consists of three or more linear strain gauges arranged at specific angles, to capture strain data from different orientations. The principal strains (and stresses in the linear elastic range of response) can be calculated as follows (Harris and Sabnis, 1999), where the strain gauges are



(a) Post Bottom **(b) Post Top**

Figure 5.8: Post strain gauges.

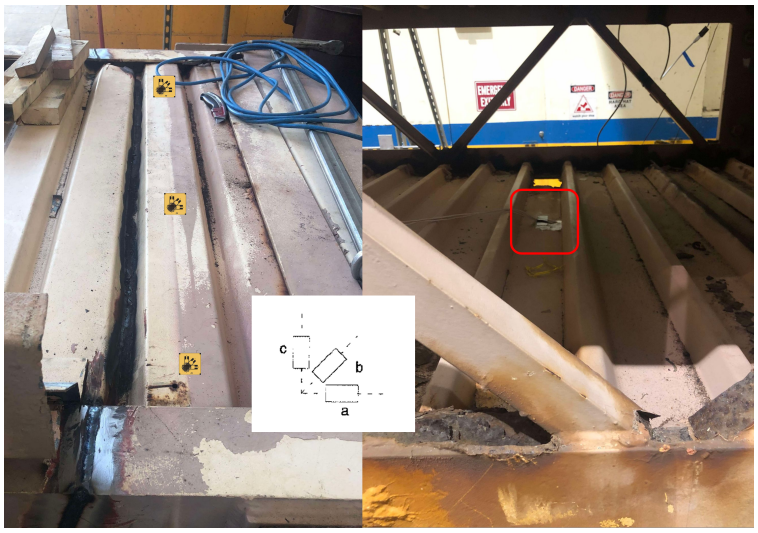


Figure 5.9: Ribbed sheet metal strain Rosettes

oriented at $0°$, $45°$, and $90°$:

$$\sigma_{max} = \frac{E}{1-\nu}\left(\frac{\epsilon_a + \epsilon_c}{2}\right) + \frac{E}{1+\nu}\sqrt{\left(\frac{\epsilon_a - \epsilon_c}{2}\right)^2 + \left(\frac{2\epsilon_b - \epsilon_a - \epsilon_c}{2}\right)^2} \tag{5.1}$$

$$\tau_{max} = \frac{E}{1+\nu}\sqrt{\left(\frac{\epsilon_a - \epsilon_c}{2}\right)^2 + \left(\frac{2\epsilon_b - \epsilon_a - \epsilon_c}{2}\right)^2} \tag{5.2}$$

and the Principal Strains and maximum shear strain from rectangular strain rosette are:

$$\epsilon_1, \epsilon_2 = \frac{\epsilon_a + \epsilon_c}{2} \pm \sqrt{\left(\frac{\epsilon_a - \epsilon_c}{2}\right)^2 + \left(\epsilon_b - \frac{\epsilon_a + \epsilon_c}{2}\right)^2} \tag{5.3}$$

$$\gamma_{\max} = 2\sqrt{\left(\frac{\epsilon_a - \epsilon_c}{2}\right)^2 + \left(\epsilon_b - \frac{\epsilon_a + \epsilon_c}{2}\right)^2} \tag{5.4}$$

where ϵ_a, ϵ_b and ϵ_c are the strains measured from three strain gauges, E and ν represent the Elastic Modulus and Poisson's Ratio respectively.

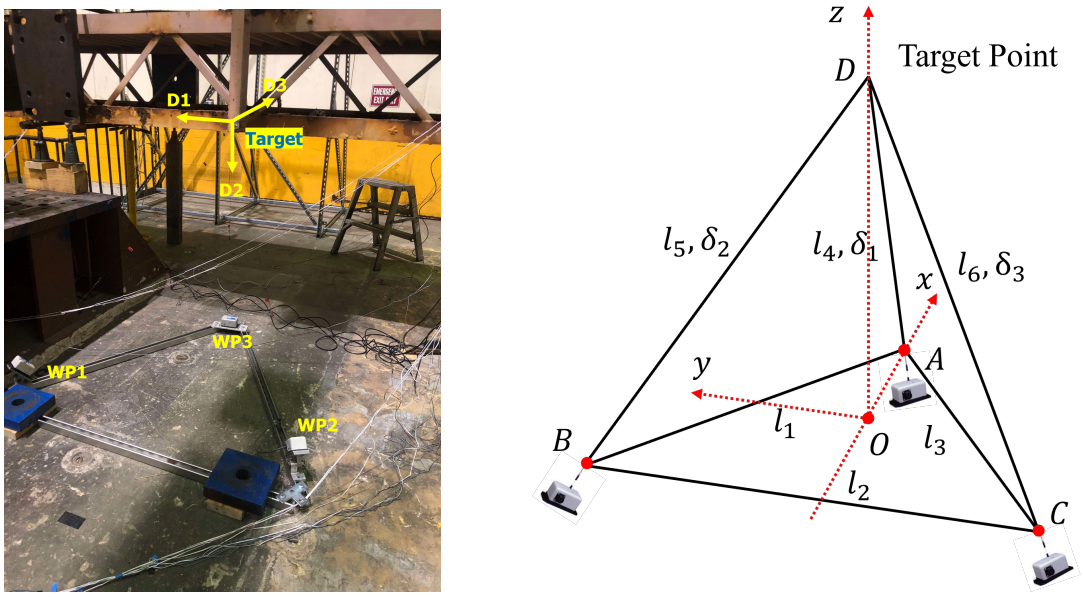
5.3.2 Wire Potentiometers

Wire potentiometer (Wirepot) is an essential and common tool for measuring linear position or displacement. A potentiometer operates similarly to a sliding rheostat, where displacement changes the current flow, enabling precise measurement of the displacement. For instance, Moustafa and Mosalam (2015) utilized wirepots to measure global displacements of the concrete box-girder bridges and determine the necessary geometric transformations for the horizontal actuators forces and displacements.

In a three-dimensional Euclidean space, accurately describing the displacement of a point requires three unknowns (X, Y, and Z, as depicted in Figure 5.10b), and three independent equations are required to solve for these unknowns. This implies that three wirepots are required to measure the displacements independently along the three directions of one single point. Figure 5.10 provides an example of how these three displacements at a point are measured using three independent wirepots. Appendix E explains the process of establishing a local coordinate system, calculating the displacements of the target point within the system, and then mapping the results back to the global coordinate system. As illustrated in Figure 5.11, displacements at eight target points on the box beam were measured to capture the complex three dimensional deformations caused by bending, shear and twisting due to the application of IP and OOP forces. Each target point requires three wirepots to accurately determine the corresponding displacement in three directions, so there are 24 wirepots needed in total.

5.3.3 Novotechnik LVDT

The LVDT (Linear Variable Differential Transformer) is a sensor designed for precise displacement measurements, commonly used to detect structural deformation and slip. For instance, Lee et al. (2015) utilized LVDTs in combination with strain gauges to measure the curvature of posts. Figure 5.12 illustrates the installation and application of four Novotechnik LVDTs on sign structures. As shown, two LVDTs were mounted at the top of the beam, while the other two were mounted at the bottom. All four sensors were located at the connection between the beam and the post. The primary objective of the setup is in two parts: first, to evaluate the slip behavior of the bolt at



(a) Lab setup **(b) Schematic diagram**

Figure 5.10: Experimental setup and schematic representation of 3D wirepot triangulation

the connection, and second, to assess the torsional deformation of the beam along its length. The beam's curvature can be determined as:

$$\kappa = \frac{\epsilon_1 - \epsilon_2}{L} \tag{5.5}$$

where ϵ_1 and ϵ_2 are the strains converted from the displacement readings of the two LVDTs installed in the opposite sides of the post and the L is the distance between two LVDTs.

5.4 NEW (UNCORRODED) STRUCTURE PREPARATION

To evaluate the impact of corrosion on the strength of the sign structure, a new corrosion-free beam with the same geometry and material characteristics was manufactured and the tests were conducted using this structure. It is noted that the post was not replaced.

5.4.1 Beam Reassembling

We constructed the new beam specimen with all the new material. The ribbed sheet metal was assembled, forming the base structure of the beam. The perimeter and diagonal angles were welded to complete the box beam's framework. The new specimen replicates the existing specimen's design, including all its components, such as the walkway, ensuring consistency for comparative analysis. The corrosion-free specimen consists of the newly fabricated box beam and the existing post (see Figure 5.13).

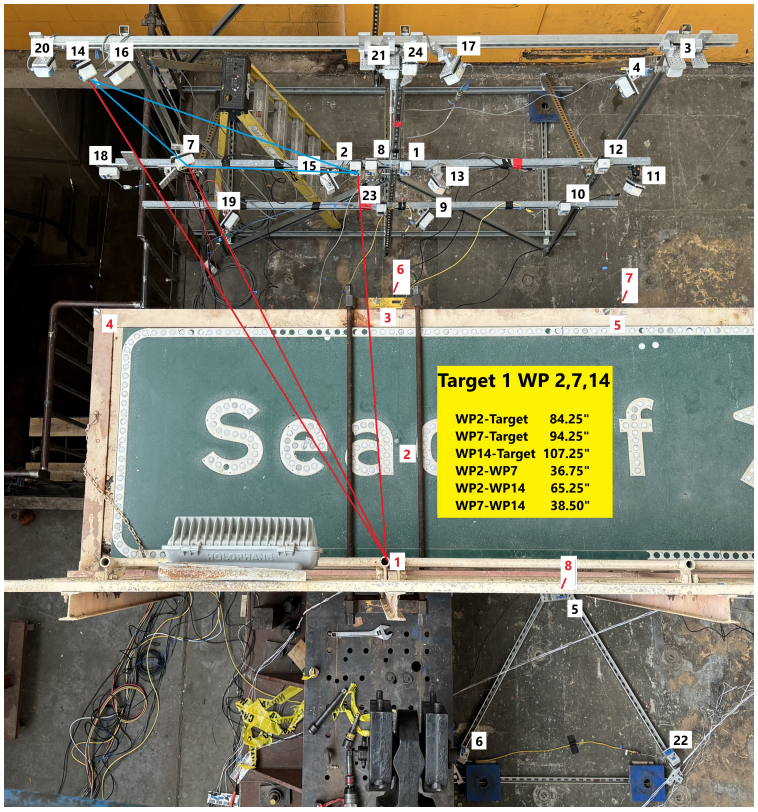


Figure 5.11: Locations and numbering of targets and wirepots for measuring the box beam displacements

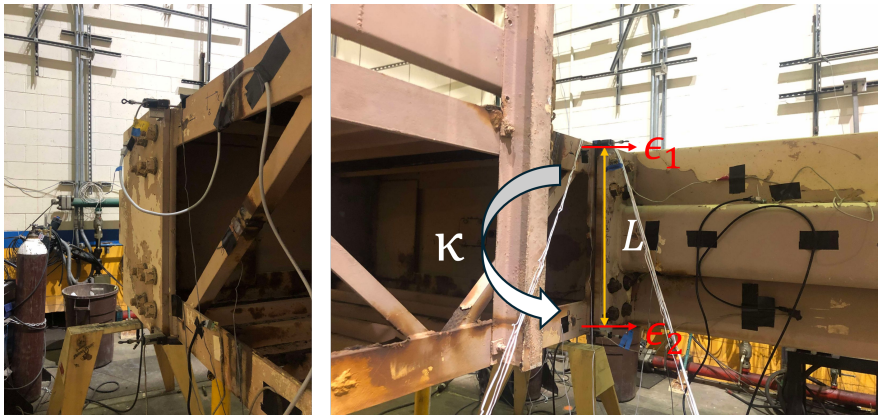


Figure 5.12: Novotechnik LVDT installation

5.4.2 Sensor Re-installation

The re-installation of sensors involve reusing the wirepots and Novotechnik. During the removal of the existing specimen, these sensors were temporarily disconnected from their cables and recon-

Figure 5.13: Uncorroded Beam Reassembling

nected afterwards. New strain gauges were installed on the box beam to facilitate strain measurements. Additionally, wirepot targets were repositioned and secured to ensure proper alignment and functionality in the new setup. Figure 5.14 shows the locations of new strain gauges. Compared with the original locations (Figure 5.7), additional strain gauges were placed on the ribbed sheet metal (e.g., 18, 19, 20, 22, and 23) and at the top of the post near the connection (e.g., 21 and 24), where damage to the corroded sign structure occurred.

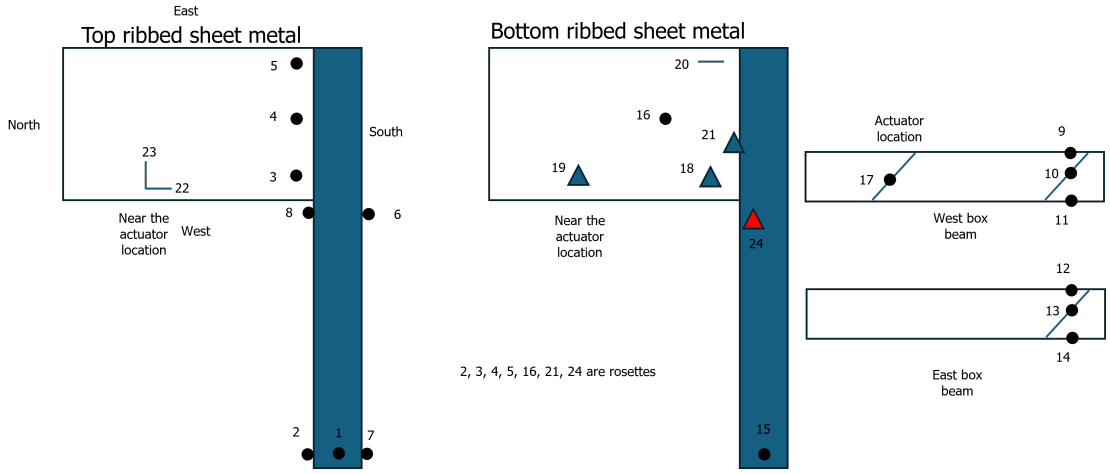


Figure 5.14: Strain gauge distribution overview of corrosion-free specimen

6 Specimen Testing, Data Analysis and Comparison

Conducted tests include: (i) hammer tests to identify the dynamic characteristics of the sign structure, (ii) low level tests to make sure that the controller, data acquisition system, all sensors, and lab equipment function properly, (iii) the actual cyclic test to characterize the response and strength of the sign structure under various loads. Each of these tests were performed for the corroded and corrosion-free sign structures and are discussed in the following sections.

6.1 HAMMER TESTING

To compare the natural frequencies of the sign structure in the laboratory and at the field, hammer tests were performed. Table 6.1 lists the identified natural frequencies, along with those from the FEM simulations.

Table 6.1: Natural periods of different sign structures, refer to Figure 6.1 for directions.

Structure Type	Natural Period [s]		
	X-Direction	Y-Direction	Z-Direction
Field	0.44	0.31	0.30
FEM Simulation	0.44	0.30	0.30
Lab (Corroded)	0.35	0.33	0.33
Lab (Corrosion-Free)	0.34	0.31	0.31

This table demonstrates that the natural periods in the Y and Z directions (i.e., In-Plane, IP) closely align with the results from both field testing and the FEM simulation, however the natural periods in the X direction (OOP) exhibit variations. Potential reasons for this difference are (i) the effect of the bridge that the sign structure is located in the field, (ii) differences in the mass excited in the OOP direction during the hammer tests in the lab and at the field. Since the corroded and corrosion-free structures have almost the same mass distribution, it is also observed from this table that corrosion in the bottom ribbed sheet metal did not have a major effect on the IP and OOP stiffness of the beam.

Figure 6.1: Directions used for natural period characterization and presentation of displacement results.

6.2 MATERIAL TESTING

A series of material tests were performed to quantify the mechanical properties of both corroded and corrosion-free steel components after completing the full-scale quasi-static testing on two specimens. These material properties were later incorporated into the FE model for further analysis, and used for interpretation of test results as discussed in the following sections. A total of four samples were tested: one sample extracted from the corroded post, two from the new angle section, and one from the new ribbed sheet. All specimens were machined into standardized dog-bone coupons using a water-jet cutter in accordance with the ASTM E8/E8M-24 (ASTM International, 2024) guidelines for metallic tensile testing.

Material testing comprised tensile tests, that were conducted to characterize the stress-strain behavior of the material samples, as illustrated in Figure 6.2. A 2-inch extensometer (in Figure 6.2a) was mounted on the gauge length of each specimen to accurately capture the axial tensile strain throughout the loading process. The applied tensile force was indirectly estimated by

monitoring the oil pressure of the test machine. All specimens fractured within the mid-length of the gauge section (Figure 6.2b), which confirms the validity of the strain measurements.

(a) Before testing **(b) After rupture**

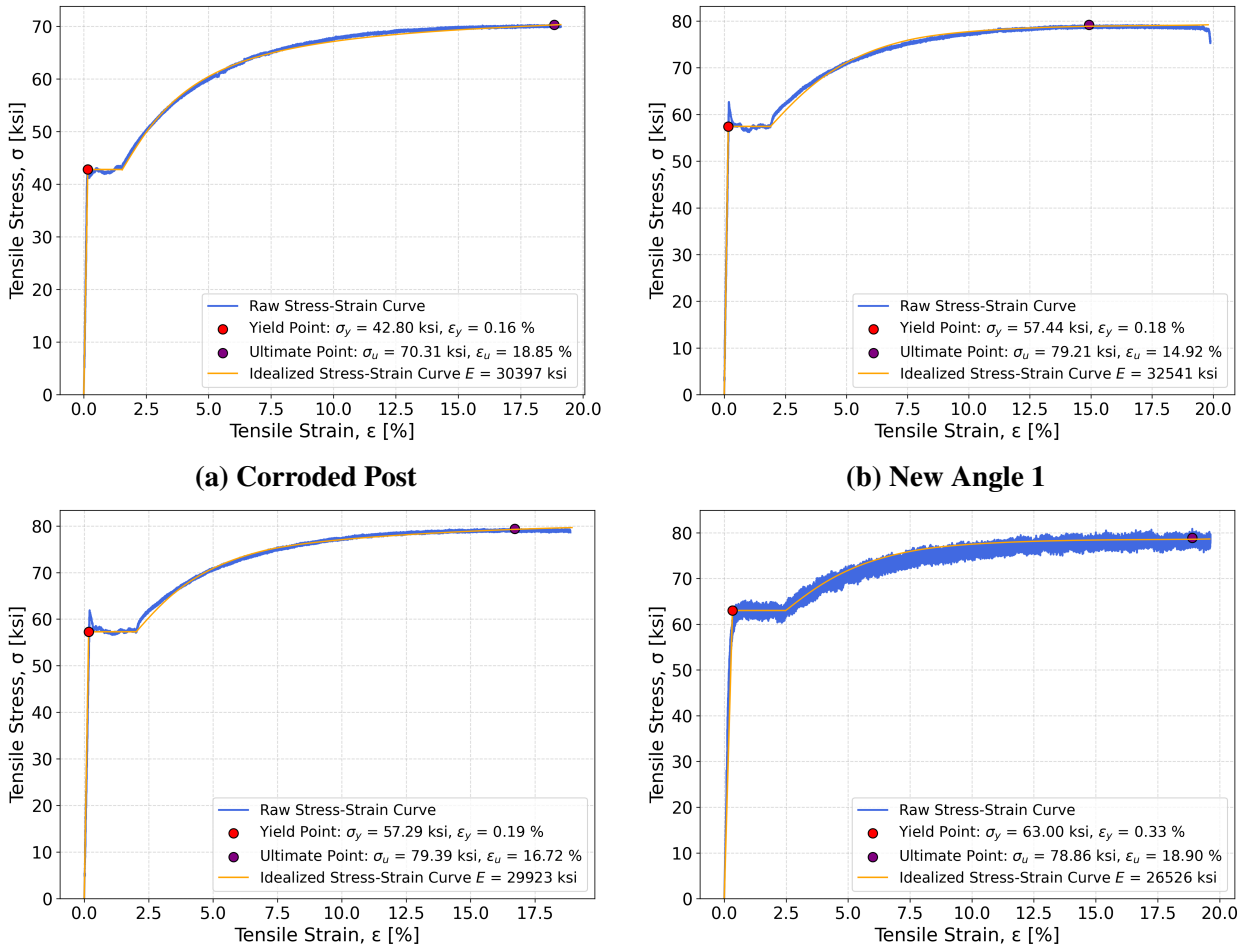
Figure 6.2: Experimental setup for Material Testing.

Figure 6.3 presents the engineering stress-strain curves of the four samples, with yield and ultimate points annotated. An idealized approximation line was also plotted for each curve to facilitate the estimation of the material properties. Table 6.2 summarizes the extracted elastic modulus, yield stress, yield strain, ultimate tensile strength and strain for all tested materials.

Table 6.2: Mechanical properties extracted from tensile stress–strain curves

Specimen Label	E [ksi]	σ_y [ksi]	ε_y [%]	σ_u [ksi]	ε_u [%]
Corroded Post Specimen	**30397**	**42.80**	**0.16**	**70.31**	**18.85**
New Angle Specimen 1	32541	57.44	0.18	79.21	14.92
New Angle Specimen 2	29923	57.29	0.19	79.39	16.72
New Angle Average	**31232**	**57.37**	**0.185**	**79.30**	**15.82**
New Ribbed Sheet Specimen	26562	63.00	0.33	78.86	18.90

Since the ribbed sheet sample was relatively thin (thickness = 1/16 inch), it was difficult for the testing machine to grab the specimen firmly. As a result, slippage occurred during the test, as evidenced by visible slip marks on the specimen after testing. This compromised the accuracy of the strain measurements for the ribbed sheet compared to the angle specimens. Therefore, in the subsequent analysis, the material properties of the **Corroded Post Specimen** and the **Average Angle Specimen** were selected to represent the corroded and uncorroded steel materials, respectively.



(a) Corroded Post **(b) New Angle 1**

(c) New Angle 2 **(d) New Ribbed Sheet**

Figure 6.3: Tensile stress–strain curves of all tested specimens.

From Table 6.2, the elastic modulus of the corroded post (30,397 ksi) is comparable to that of the new angle specimens (average 31,232 ksi), suggesting that corrosion did not substantially degrade the material's initial stiffness. However, the yield stress of the corroded specimen is significantly lower than that of the uncorroded ones, approximately 25% lower. This reduction highlights the detrimental effect of corrosion on steel's yield point. The ultimate strength of the corroded specimen (70.31 ksi) is 11% lower than that of the new angle specimens (79.30 ksi average), suggesting that while corrosion reduces yield capacity, the material still retains considerable ultimate tensile strength.

6.3 OBSERVATION-BASED STRUCTURAL RESPONSE ANALYSIS

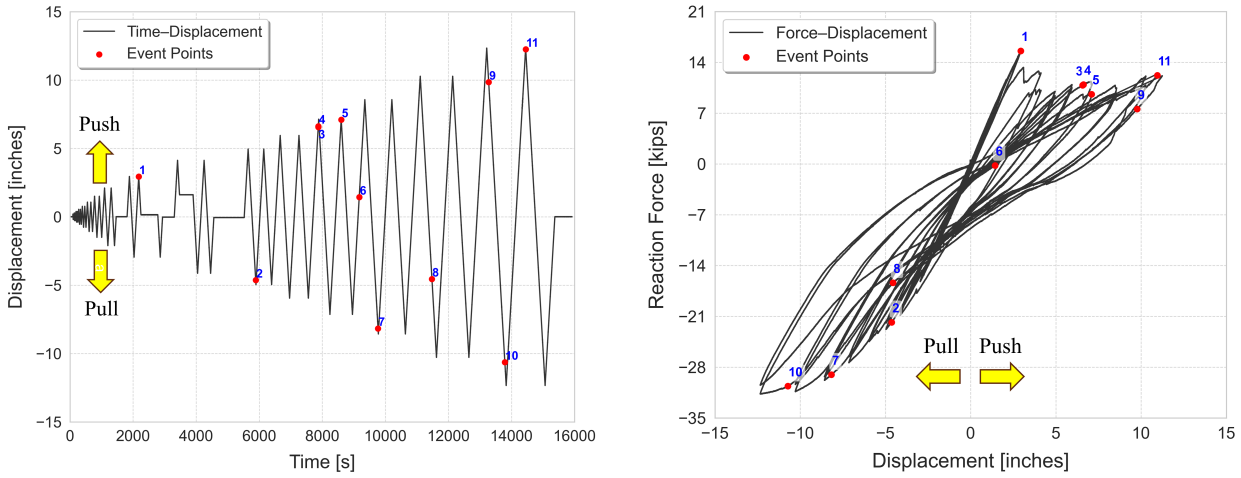
This section provides a detailed and comprehensive assessment of the structural behavior under cyclic loading for both corroded and corrosion-free specimens. Observations are derived from experimental video frames, photos, and strain gauge measurements. These observations are used

to identify the damage progression and damage locations, deformed shapes, yield initiation, and failure modes. The observations also provide insights into how corrosion affects these phenomena and changes the patterns. The layout and distribution of these strain gauges can be found in Figure 5.7 (corroded) and Figure 5.14 (Corrosion-Free).

6.3.1 Corroded Specimen: Yielding and Damage Progression

This section illustrates the key observations and events during the testing of the corroded specimen. Table 6.3 summarizes the observed structural responses of different components. Each event is described with its corresponding component and category, and has been annotated in the displacement history (Figure 6.4a) and the force-displacement relationship (Figure 6.4b). The yield-strain is determined as $\varepsilon_y = \pm 1600$ μstrain from the corroded post material test results in Figure 6.3a and Table 6.2

As described in Table 6.3, the beam region exhibited the earliest yielding behavior, followed by damage propagation into the post elements. Inelastic buckling and significant permanent deformation was observed in the ribbed sheet, especially near the bottom connection with the angle, where visible detachment and cracking occurred. These observations indicate that the connection between the ribbed sheet and the angle is a structural weak point, where detachment and cracking under cyclic loading could substantially reduce the overall structural force capacity and energy dissipation capacity.



(a) Time vs. Displacement **(b) Displacement vs. Force**

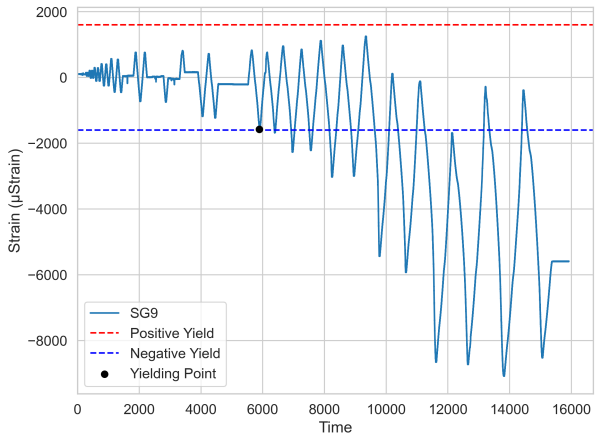
Figure 6.4: Key events annotated on time–displacement and force–displacement curves of corroded specimen

Figure 6.10 shows the ultimate failure pattern and observed heavy damage in the corroded specimen. As shown in the Figure 6.10a, the entire specimen has undergone severe deformation, with the beam section twisting downward to the ground, and the bottom ribbed sheet detaching from the frame as a result of buckling. The separation of ribbed sheet from the frame increased the stiffness difference between the box beam top and bottom planes and resulted in the observed

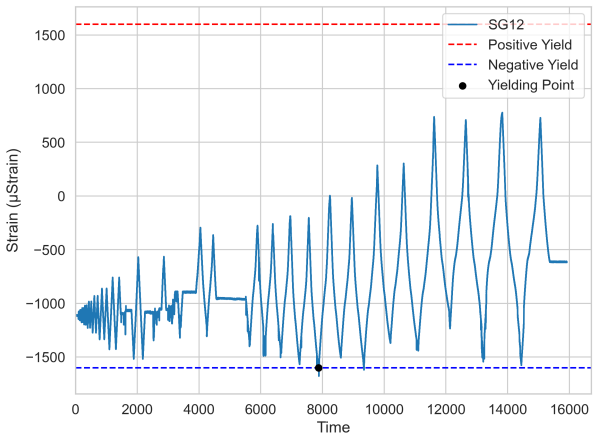
Table 6.3: Observation Events and Corresponding Figures with Structural Categories

No.	Observation Description	Figure	Component	Category
1	Corroded (lower) ribbed sheet metal buckling, reaching the force capacity in the push direction, with lower force peaks in all subsequent cycles	Figure 6.4b	Box Beam	Buckling & Peak Load
2	SG9 was the first to exceed the yield strain (± 1600 μstrain), indicating the start of plastic deformation at the west upper beam.	Figure 6.5a	Box Beam	Yielding
3 & 4	SG12 (east upper beam) and SG14 (east bottom beam) yielded almost at the same time during an upward loading cycle, suggesting that top and bottom flanges on the east side entered plasticity together.	Figures 6.5b, 6.6a	Box Beam	Yielding
5	SG11, on the West Bottom beam, exhibited yielding one cycle after SG14.	Figure 6.6b	Box Beam	Yielding
6	The lower ribbed sheet began to detach from the angle, accompanied by a distinct cracking sound. In the video, visible longitudinal movement occurred as the actuator moved.	Figure 6.7a	Ribbed Sheet	Connection Failure
7	SG6 on the south post (away from the beam) yielded in the later phase of loading, showing delayed post engagement.	Figure 6.7b	Post	Yielding
8	During the push direction, the ribbed sheet near the actuator experienced significant permanent deformation. Nonlinear curvature remained after unloading, indicating local loss of stiffness.	Figure 6.8a	Ribbed Sheet	Permanent Deformation
9	During the pull direction, very obvious localized permanent deformation appeared at the circled area on the ribbed sheet. This deformation persisted post-unloading.	Figure 6.8b	Ribbed Sheet	Permanent Deformation
10	SG8 on the north post (near the beam) yielded late in the test, indicating deformation had extended to the post area.	Figure 6.9a	Post	Yielding
11	Under push direction loading, more permanent deformation was observed in the ribbed sheet away from the actuator zone.	Figure 6.9b	Ribbed Sheet	Permanent Deformation

twisting about the shown X axis. Figure 6.10b further reveals significant local buckling in the post at the connection region, and fracture at the top of the post.

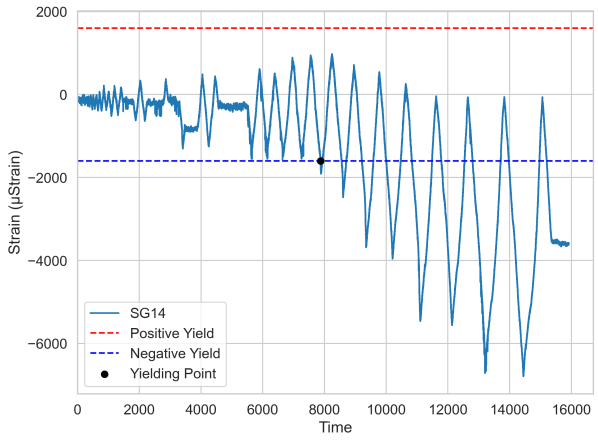


(a) SG9 Yield

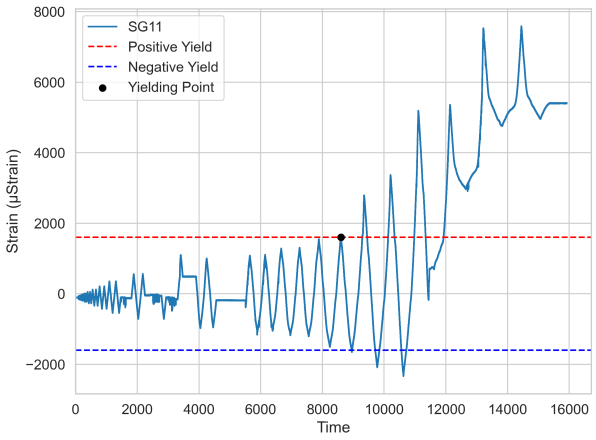


(b) SG12 Yield

Figure 6.5: Yield observations of SG9 and SG12

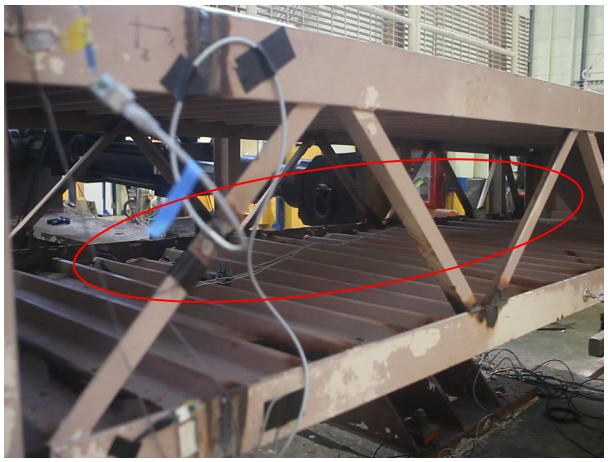


(a) SG14 Yield

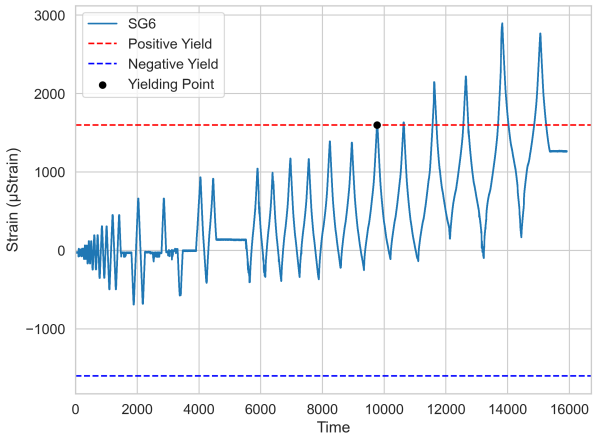


(b) SG11 Yield

Figure 6.6: Yield observations of SG14 and SG11



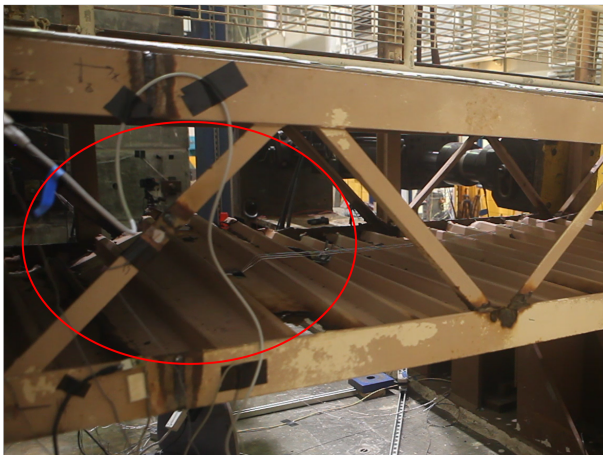
(a) Lower ribbed sheet detachment



(b) SG6 Yield

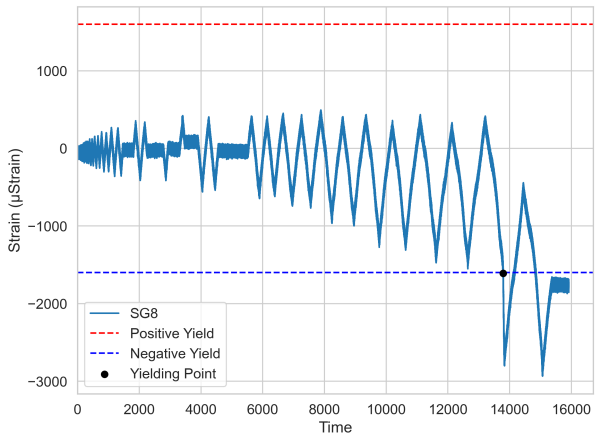
Figure 6.7: Detachment observation and SG6 Yield

(a) Permanent deformation near actuator (Push direction)

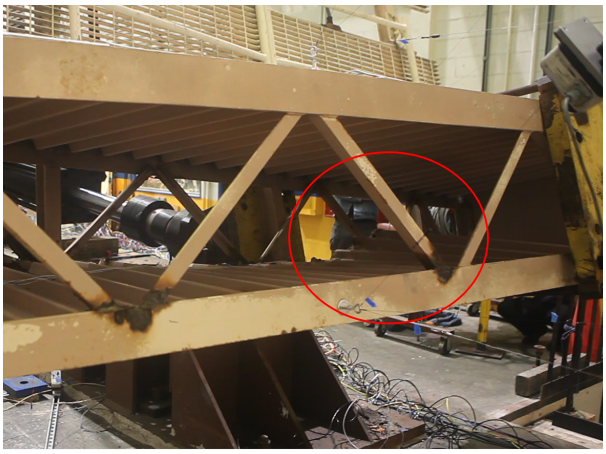


(b) Permanent deformation near actuator (Pull direction)

Figure 6.8: Deformation observations during push and pull directions

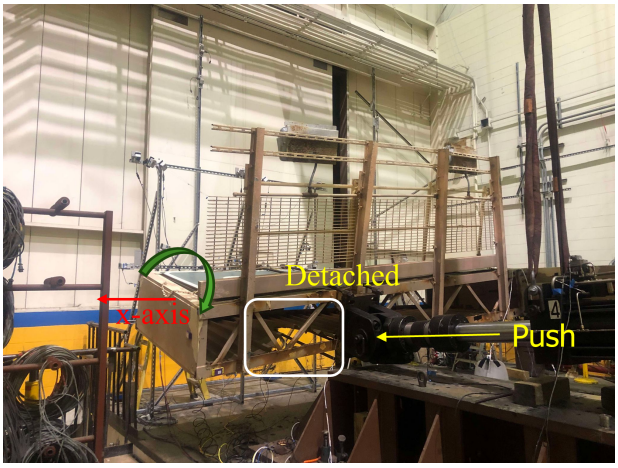


(a) SG8 Yield after loading

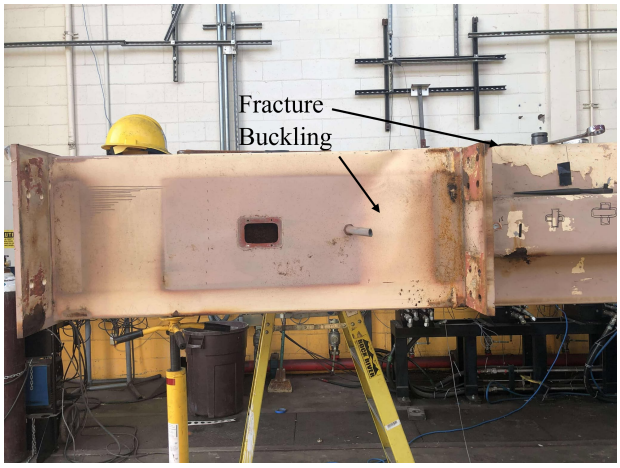


(b) Permanent deformation in Push direction

Figure 6.9: Post-loading SG8 yield and permanent deformation



(a) Detachment of Ribbed Sheet



(b) Post Buckling and Fracture

Figure 6.10: Corroded specimen experiencing major displacements and damage.

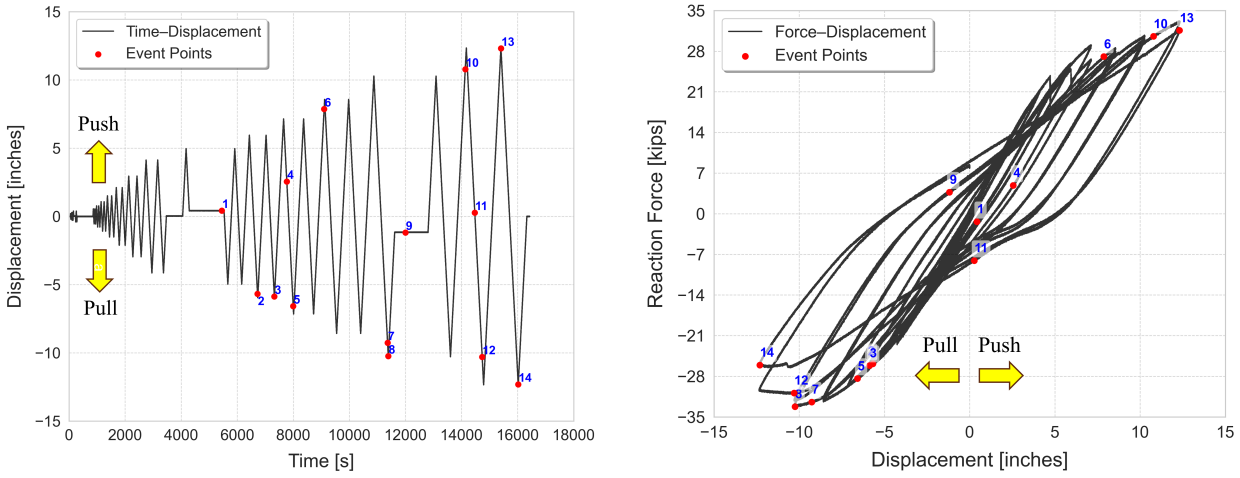
6.3.2 Corrosion Free Specimen: Yielding and Damage Progression

This section presents the key observations from the corrosion-free specimen during cyclic loading (Table 6.4). All events are annotated in the time–displacement curve (Figure 6.11a) and the force–displacement response (Figure 6.11b).

To identify the yielding points, we used the average yield strain of $\varepsilon_y = \pm 1850$ μstrain for the beam (based on the new angle material in Table 6.2), and $\varepsilon_y = \pm 1600$ μstrain for the post (based on the corroded post material). Compared to the corroded specimen, the corrosion-free specimen exhibited a different failure pattern. The post region showed earlier and more localized deformation, while some strain gauges on the beam yield later and even some remained within the elastic range throughout the test.

Strain gauges SG6, SG8, SG21, and SG24, installed on the post surfaces, recorded significant nonlinear strain, reaching over 10 to 20 times the yield level. While such magnitudes are common in post-yield behavior, the consistent concentration of deformation in the post suggests that cyclic demands were primarily absorbed by the vertical elements. This indicates that, in the absence of corrosion, the ribbed sheet–angle connections remained intact, enabling the posts to sustain the majority of inelastic deformation.

The comparison between the two specimens highlights how corrosion alters the damage mechanisms and overall failure pattern. In the corroded specimen, the weakened ribbed sheet-angle connection led to earlier detachment and component separation, shifting the inelastic demand from post to beam. These findings underscore the critical impact of corrosion on load path, failure sequence, and energy dissipation behavior.

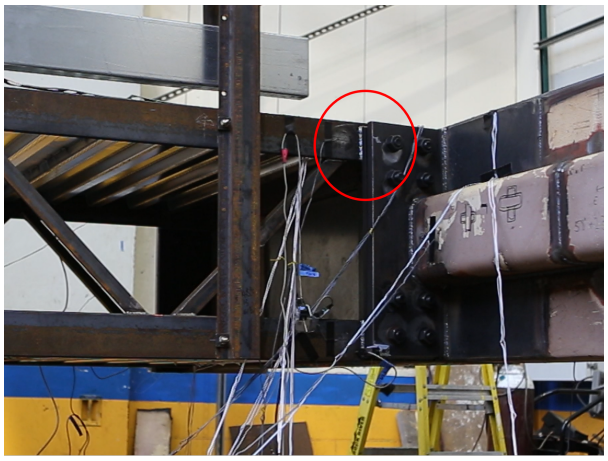


(a) Time vs. Displacement **(b) Displacement vs. Force**

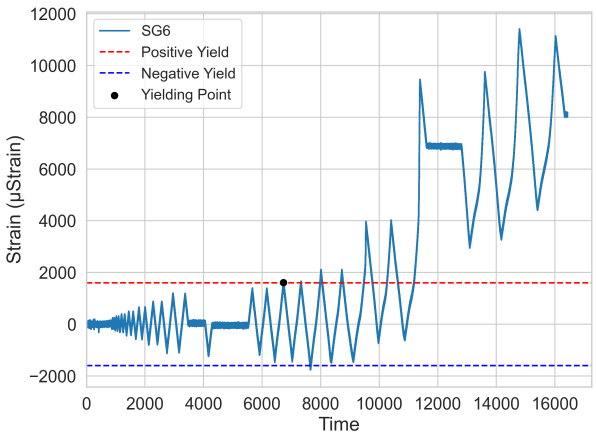
Figure 6.11: Key events annotated on time–displacement and force–displacement curves of corrosion-free specimen

Table 6.4: Observation Events and Corresponding Figures for the Corrosion-Free Specimen

No.	Observation Description	Figure	Component	Category
1	Visible opening and closing was observed at the joint connection under cyclic loading, suggesting possible slippage at the connection.	Figure 6.12a	Connection	Damage
2	SG6 (south post, far from beam) yielded early in the test, indicating yielding of the post.	Figure 6.12b	Post	Yielding
3	SG21, located at the beam-to-post junction, exceeded the yield strain.	Figure 6.13a	Post	Yielding
4	A loud cracking sound occurred, and the post–beam connection showed visible signs of failure.	Figure 6.13b	Connection	Damage
5	SG24 (upper post surface) yielded during the same loading cycle.	Figure 6.14a	Post	Yielding
6	SG9 (west upper beam flange) yielded later in the test.	Figure 6.14b	Box Beam	Yielding
7	SG11 (west bottom beam flange) yielded after SG9, completing beam flexural engagement.	Figure 6.15a	Box Beam	Yielding
8	After yielding, SG6 continued to deform plastically, with strain increasing to approximately 6 times the yield level.	Figure 6.12b	Post	Large Deformation
9	Progressive failure occurred at the connection joint. The beam detached from the post and a lateral crack extended nearly 2/5 of the post height.	Figures 6.15b, 6.16a	Connection	Damage
10	SG8 (north post near beam) yielded in the later stage of loading.	Figure 6.16b	Post	Yielding
11	The post wall (red box area) showed severe twisting and deformation. The crack reached near the bottom of the joint.	Figures 6.17a, 6.17b	Post	Global Deformation
12	SG21 exhibited continued nonlinear response after yielding, reaching approximately 20 times the yield strain.	Figure 6.13a	Post	Plastic Deformation
13	SG8, after initial yielding, developed stable plastic strain exceeding 10 times the yield level.	Figure 6.16b	Post	Plastic Deformation
14	SG24 also entered post-yield response with strain surpassing 10 times the yield strain.	Figure 6.14a	Post	Plastic Deformation

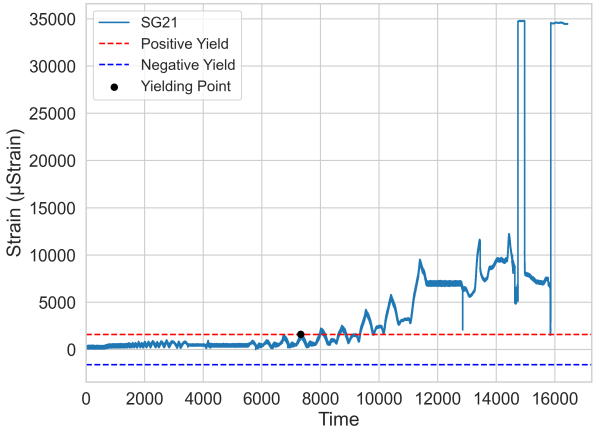


(a) Joint Opening



(b) SG6 Yield

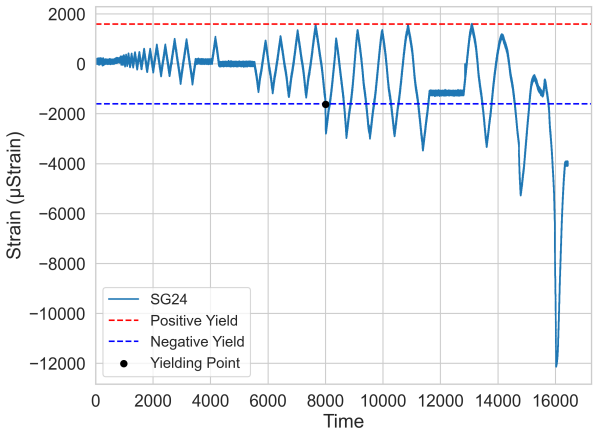
Figure 6.12: Joint loosening and SG6 Yield



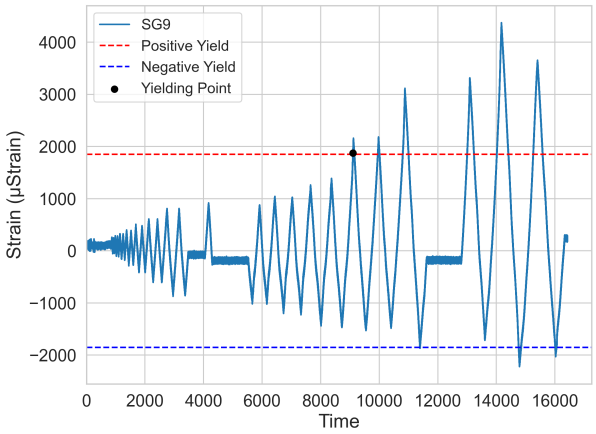
(a) SG21 Yield

(b) Initial Connection Failure

Figure 6.13: SG21 Yield and Initial Connection Failure

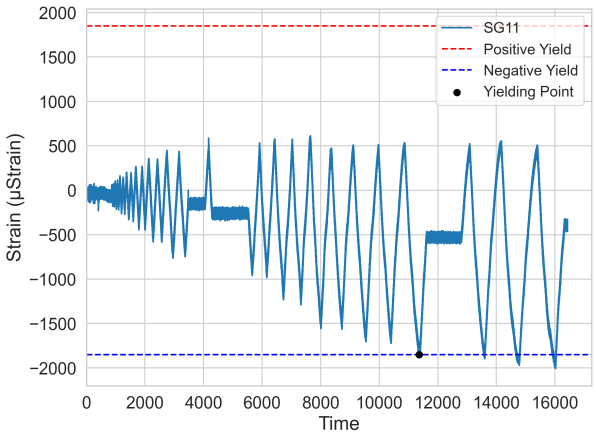


(a) SG24 Yield

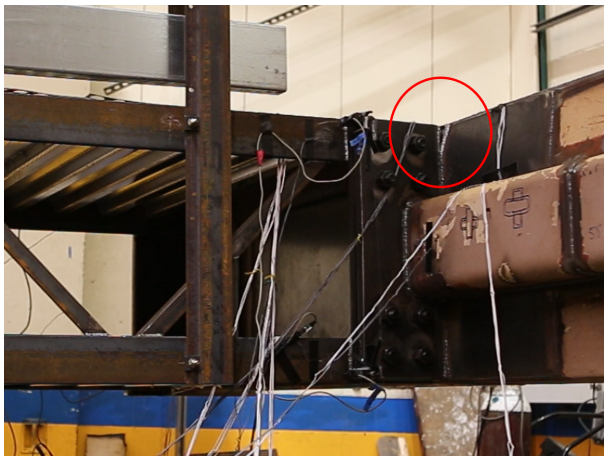


(b) SG9 Yield

Figure 6.14: Yielding observations from SG24 and SG9

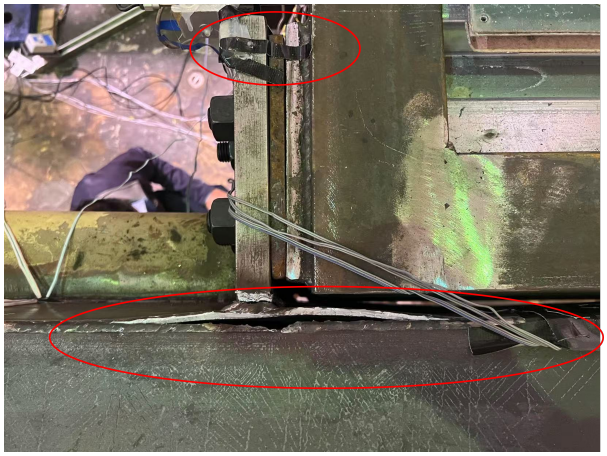


(a) SG11 Yield

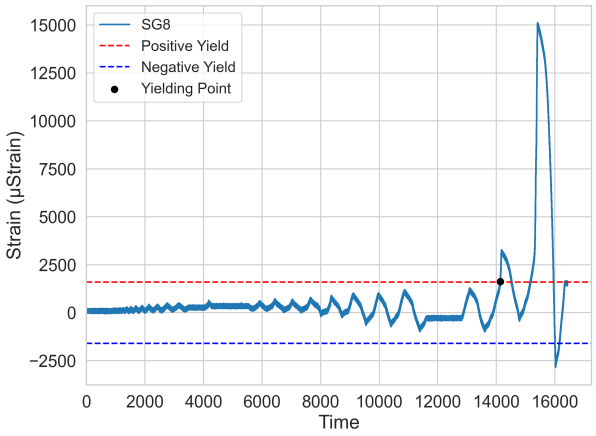


(b) Progressive Connection Failure

Figure 6.15: SG11 Yield and Progressive Connection Failures

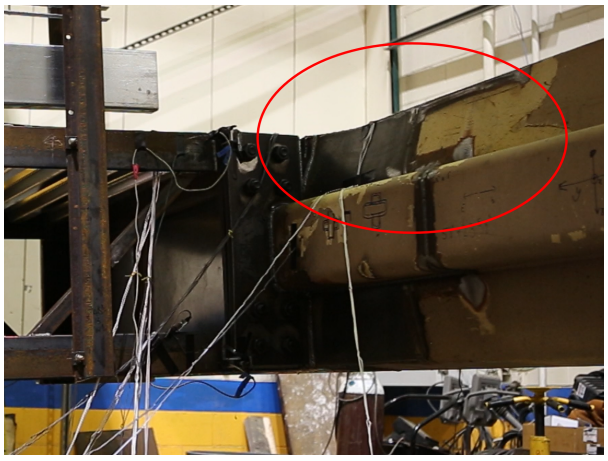


(a) Column Beam Detachment

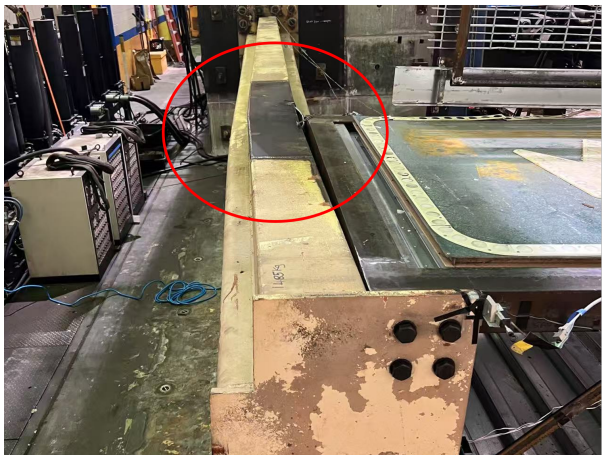


(b) SG8 Yield

Figure 6.16: Column Beam Detachment and SG8 Yield



(a) Post Deformation View 1



(b) Post Deformation View 2

Figure 6.17: Severe Damage and Deformation at Post

6.4 STRAIN GAUGE RESULTS

6.4.1 Degraded (Corroded) Specimen Results

Figure 6.18 shows the strain measurements of Strain Gauge #2 & #7, located at the base of the post (see the strain gauge locations in Figure 5.7). Yielding is observed at SG #2 & #7 (using the yield strain of $1600\mu\varepsilon$ from material testing), and residual strain remains after unloading and in following cycles. The yield onset at the post base occurs when the actuator displacement reaches 10.282 inches. This is expected due to the nature of steel stress-strain relationship, with constant stress at the yield plateau before reaching the strain hardening region.

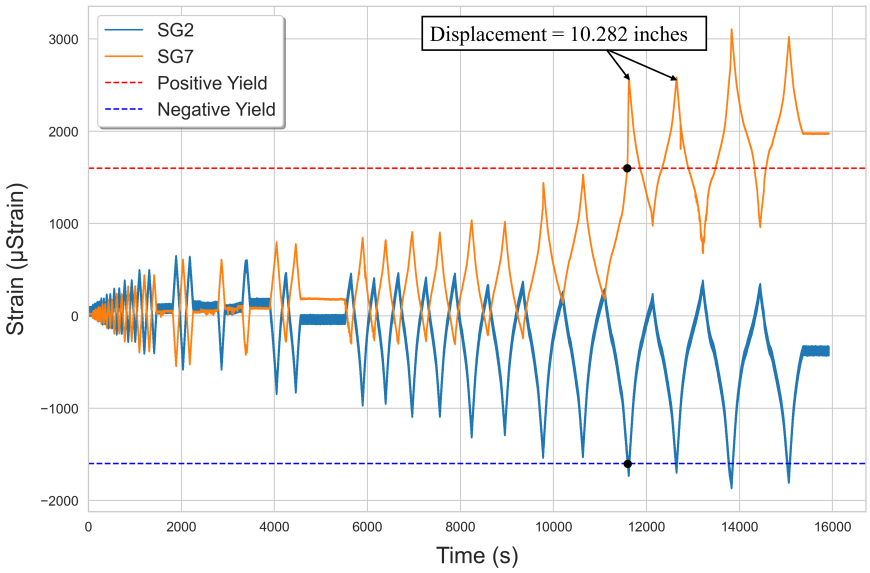


Figure 6.18: Strain Gauge #2 & #7 results at the corroded structure post.

Figure 6.19 presents the strain data for SG #9, #10, #11, all placed on the box beam angles near the connection, which is the location of highest stress under the applied IP cyclic and the OOP static loading. SG #9 and #11 are located at the perimeter angles, while SG #10 is placed on the diagonal angle as shown in Figure 5.7. This figure also plots the strain data from SG #17, which was placed on the diagonal angle near the actuator location to monitor the presence of any local stress concentrations near the point of loading application. Throughout the test, SG #10 and SG #17 remained below the yield strain ($|\varepsilon_{SG}| < 1600\ \mu\varepsilon$), indicating no pronounced local stress concentration at the loading point and limited participation of the braced-truss members. By contrast, the angle yielded: SG #9 reached yield at an actuator displacement of 5.95 inches, and SG #11 yielded at 7.14 inches. This shows that the diagonal member contributes much less to resisting the in-plane loading, consistent with the main use of these members as wind braces in the OOP direction.

In the linear elastic range, the signs of SG #9 and #11 are the same as expected because the bending moment due to the in-plane moment is expected to cause nearly uniform tension and compression across the depth of the box beam where SG #9 and #11 are located. As discussed

earlier, twisting occurred as the test progressed due to the stiffness difference between the top and bottom ribbed sheet metal. As twisting occurs, SG #9 and #11 strains start to deviate from each other. With twisting amplifying the effect of OOP loading, residual tensile and compressive strains occur at SG #11 and SG #9, respectively at the top and bottom angles.

Figure 6.19

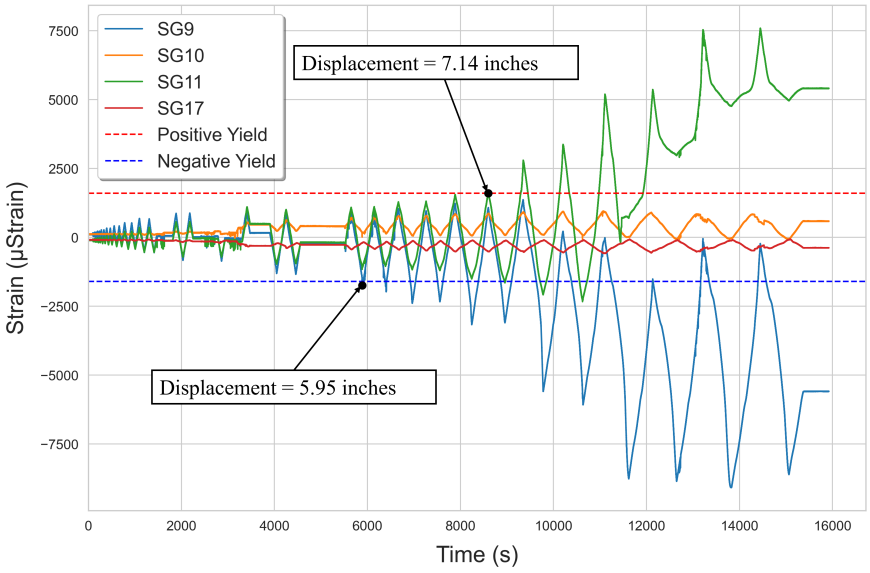
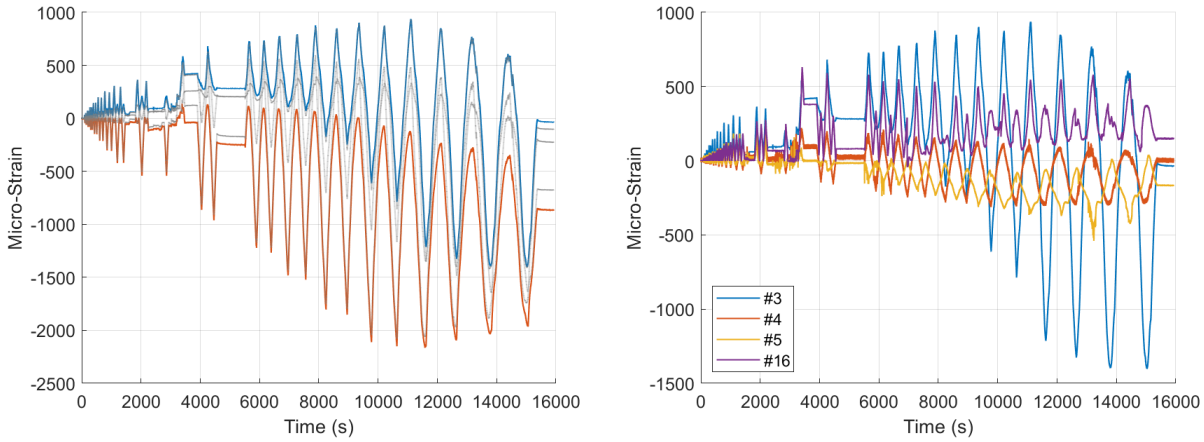


Figure 6.19: Measured strains at the box beam angles of the corroded specimen.

Figure 6.20 presents the strain measurements in the ribbed sheet metal. Figure 6.20a contains the details about the strain rosette in three directions (gray lines) and the corresponding principal strains (blue line is the maximum while orange is the minimum). Figure 6.20b illustrates all the principal strains listed on the ribbed sheet, from which, we observe that SG #3, #4 and #5 exhibit a gradually decreasing trend, indicating that the ribbed sheet above is undergoing progressive compression. This observation aligns well with the bottom frame in Figure 6.19, which is also experiencing compression. However, in terms of magnitude, the compressive strain in the ribbed sheet is significantly lower than that in the frame, approximately -2000 micro-strain compared to -9000 micro-strain. Additionally, it is evident that the strain at SG 3 is significantly greater than that at SG 4 and SG 5. This suggests that during cyclic loading, the connection between the ribbed sheet and the column experiences higher stress, making it more susceptible to damage. This observation also aligns with the actual experimental results.



(a) Strain rosette #3 **(b) Principal strains on the ribbed sheet metal**

Figure 6.20: Strain gauge data on the ribbed sheet metal.

6.4.2 New (Corrosion-free) Structure Results

As discussed earlier, SG#1-17 are placed at the same locations as the corrosion-free structure, allowing for comparison. Similar to Figure 6.18 for the corroded structure, Figure 6.21 presents the strain data from SG #2 & #7 in the corrosion-free structure.

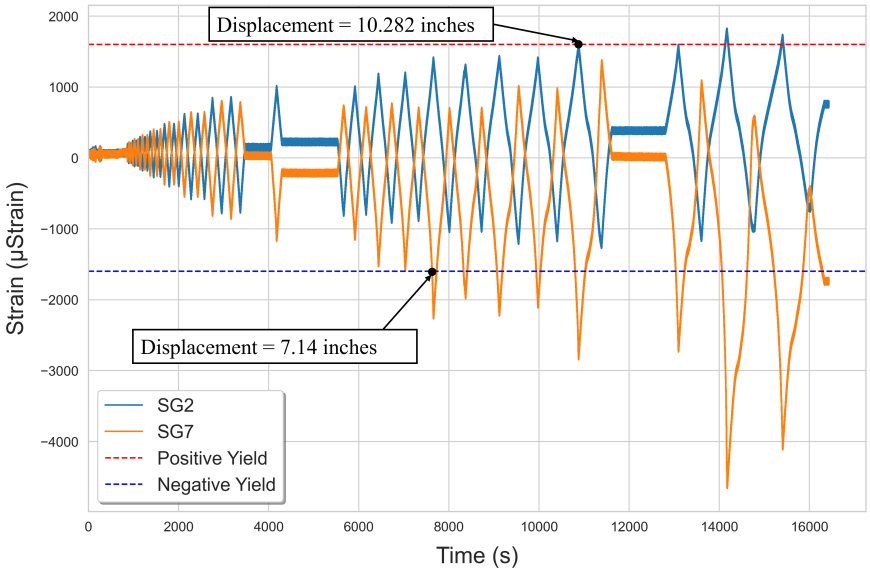


Figure 6.21: Strain Gauge #2 & #7 results at the corrosion-free structure post.

It is observed from this figure that, similar to the corroded structure, yielding occurs at the location of SG#2 & #7. It is observed that for the corrosion-free specimen, SG #2 still exhibits yielding at a displacement of 10.282 inches, similar to the corroded structure. However, a key difference is that SG #7 yields earlier, at 7.14 inches. As previously discussed in Section 6.3,

this shift indicates that corrosion alters the failure pattern, delaying the yielding of the post in the corroded specimen.

Figure 6.22 analyzes the strain gauges #9 & #10 placed on the box beam angles, and strain gauge #11 mounted on the L-shaped braced truss. It is observed that yielding occurs at both SG #9 & #11 in the angle. Compared to the corroded specimen, however, yielding in the corrosion-free specimen occurs later, at a displacement of 8.569 inches.

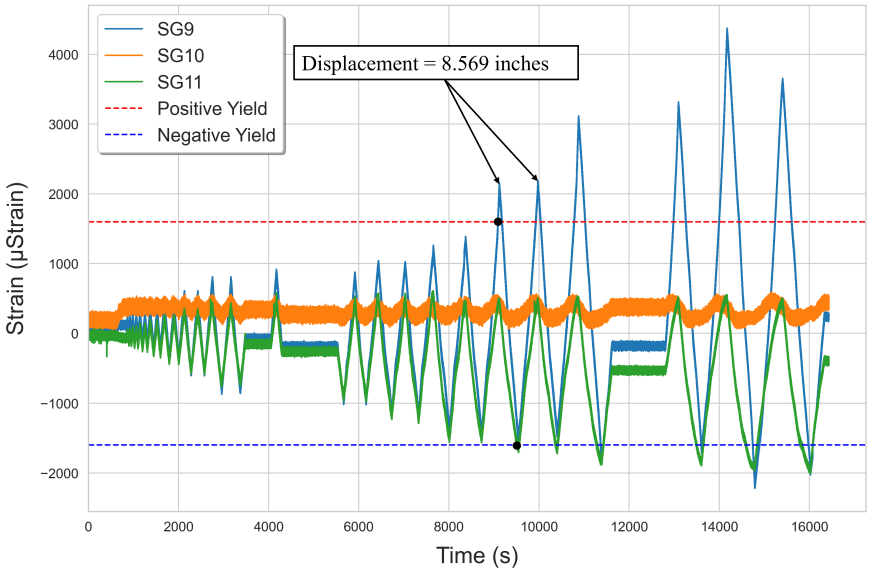


Figure 6.22: Corrosion-free SG data in angle and L-beam.

6.4.3 Summary

The analysis of linear strain gauges placed on the post and angle reveals that the corrosion-free specimen reaches yield earlier at the post. However, because the ribbed sheet–angle connection in the corrosion-free specimen remains intact, yielding in the angle occurs later compared to both the post and the corresponding location in the corroded specimen. These findings suggest that relying on local strain measurements may underestimate the structural degradation in corroded specimens. Therefore, a comprehensive assessment incorporating both strain data and global displacement monitoring is essential for accurately evaluating the impact of the cyclic loads and corrosion.

6.5 WIREPOT MEASUREMENTS

As shown in Fig. 5.11, eight targets were monitored to measure displacement using wirepots. Fig. 6.23 illustrates the 3D graphical representation of Target #1 and its corresponding three wirepots. This point is located near the connection between the actuator and the structure. In this figure, Points A, B, C and D represent the wirepots #7, #2, #14 and target #1, respectively. By employing the triangulation method detailed in Appendix E, a local coordinate system was first established with reference to the geometry in Figure 5.10b, and then transformed into the global coordinate system (in Figure 6.1) through rotation and translation.

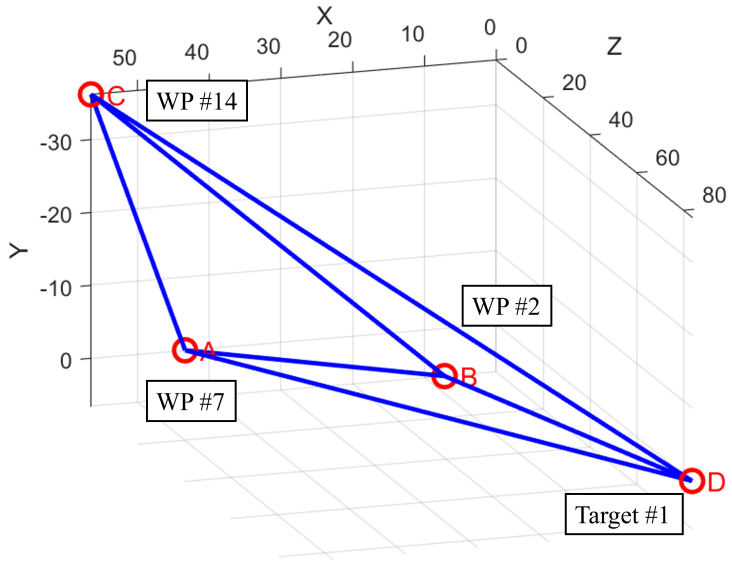
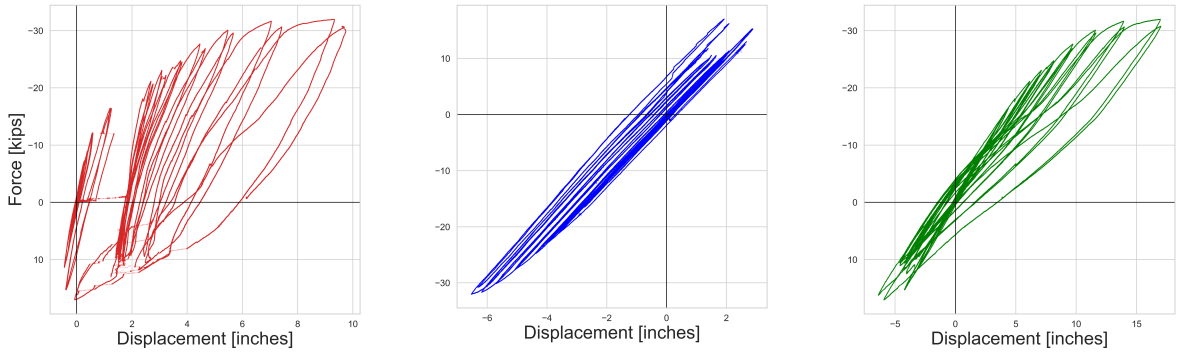


Figure 6.23: 3D graphical representation of target #1.

6.5.1 Corroded-Specimen

Figure 6.24 presents the force-displacement relationships in the X, Y, and Z directions of target #4 (the point at the top of the box beam in the corner as shown in Figure 5.11). This figure provides critical insights into the deformation behavior of the specimen under loading. As discussed earlier, the applied force is in the negative Z-direction, with the X-axis oriented vertically upward and the Y-axis aligned with the beam longitudinal axis.

- **X-direction:** The force-displacement curve in the X-direction is illustrated in Figure 6.24a, represents the direction perpendicular to the specimen. It can be observed clearly that as the force magnitude increased, the displacement of target #4 in the X-direction shifted progressively toward the positive direction, eventually reaching a maximum displacement of approximately 10 inches. This behavior indicates that the entire beam has undergone significant plastic and permanent deformation. Compared with the Y directions, the hysteresis



(a) X-direction **(b) Y-direction** **(c) Z-direction**

Figure 6.24: Force–displacement relationships of Target #4 for the corroded specimen in the X, Y, and Z directions.

loops observed in the X-direction are significantly larger. It is noted that the applied force is in the Z-direction, therefore the concept of Z force - X displacement relationship is not fully meaningful in terms of energy dissipation, therefore quantified values are not provided and the results are discussed qualitatively.

- **Y-direction:** Although the applied force acts along the Z-axis, the displacement in the Y-direction (refer to Figure 6.24b) arises as a secondary response due to bending and lateral deformation of the specimen. The Y-direction displacement line demonstrates fewer obvious variations within the specimen plane. The force-displacement relationship in this direction appears approximately linear, and the estimated longitudinal stiffness of the specimen at target #4 in the Y-direction is roughly $30/6 = 5$ kips/in. Displacement in this direction is primarily due to the bending and lateral deformation of the post, which remained approximately linear elastic until the end of the test when fracture occurred at the top of the post.

- **Z-direction:** The force-displacement curve in the Z-direction (Figure 6.24c), aligned directly with the applied force, clearly represents the global structural response. As previously discussed, the Z-direction corresponds precisely to the actuator's pulling direction. The maximum displacement under pulling is significantly greater than under pushing, indicating that the corroded specimen exhibits notably reduced strength in the push direction compared to the pull direction. Additionally, the stiffness of Z-direction is roughly $30/15 = 2$ kips/in, which is numerically smaller than the slope estimated in Y-direction. Furthermore, as the force level increases, the hysteresis loops gradually change, accompanied by a reduction in slope. This trend is likely indicative of material nonlinearity, stiffness degradation, or progressive component failure within the specimen.

To investigate the displacement patterns of the eight target points, their deformed coordinates were recorded under the maximum imposed cyclic displacement of 12.3 inches in the Z-direction. Table 6.5 summarizes the maximum displacements of all target points, highlighting significant differences from each other and from the applied cyclic displacement. This data

provides valuable insights into the structural response, aiding in the assessment of deformation patterns and the evaluation of displacement distribution across the specimen. Particularly, this data supports the visually observed twisting of the box beam and large vertical displacements in the X-direction.

Table 6.5: Maximum displacement response of the corroded specimen under cyclic push-pull loading

Target	Push Direction			Pull Direction		
	X [in]	Y [in]	Z [in]	X [in]	Y [in]	Z [in]
1	−10.69	−1.59	−4.00	−2.90	4.33	10.56
2	−0.33	5.32	−4.28	6.79	2.48	10.54
3	2.49	1.65	−3.81	–*	–*	–*
4	0.01	2.06	−6.35	9.33	−6.54	16.96
5	0.44	−2.11	−2.66	−3.66	8.98	6.64
6	3.24	0.16	−11.52	−9.01	−1.26	14.97
7	1.25	2.43	−5.58	−1.74	−8.57	9.17
8	−6.12	−1.19	−8.28	−2.03	2.07	7.14

*Due to WP damage, values are not reported.

At the same time, Figures 6.25 and 6.26 illustrate the two-dimensional displacement patterns of five target points subjected to push loading. The undeformed configuration is depicted by dashed black links connecting the initial positions of these targets, while the colored solid lines represent their deformed positions after displacement under different load levels. These displacement patterns clearly depict how the beam progressively deforms with increasing load. Notably, targets #1 and #8 (show in Figure 6.26), which are located nearest to the side of the actuator, shows that there exists large downward vertical deformation. In contrast, targets #4, #6, and #5 (show in Figure 6.25), located farther from the actuator, displace different behavior. The points away from the post (targets #4 and #6) exhibit upward deformation while the points close to the post (target #5) still moves around the initial position.

6.5.2 Corrosion-free specimen

Figure 6.27 represents the Force-Displacement relationships of target #4 for the corrosion-free specimen in the X, Y, and Z directions. As previously discussed, the applied force is in the negative Z-direction, with the X-axis oriented vertically upward and the Y-axis aligned along another direction of the specimen. Compared to the corroded specimen, the corrosion-free specimen exhibits notable differences in stiffness, displacement magnitude, and hysteresis behavior due to the absence of corrosion-induced degradation and the change of failure pattern.

- **X-direction:** In the X-direction, the corrosion-free specimen exhibited relatively small displacement variations. Figure 6.27a clearly shows that the displacement was limited to approximately 5 inches. This illustrates that the specimen may not have large rotation along Y- or Z-direction like what happened in corroded specimen. Consequently, this behavior

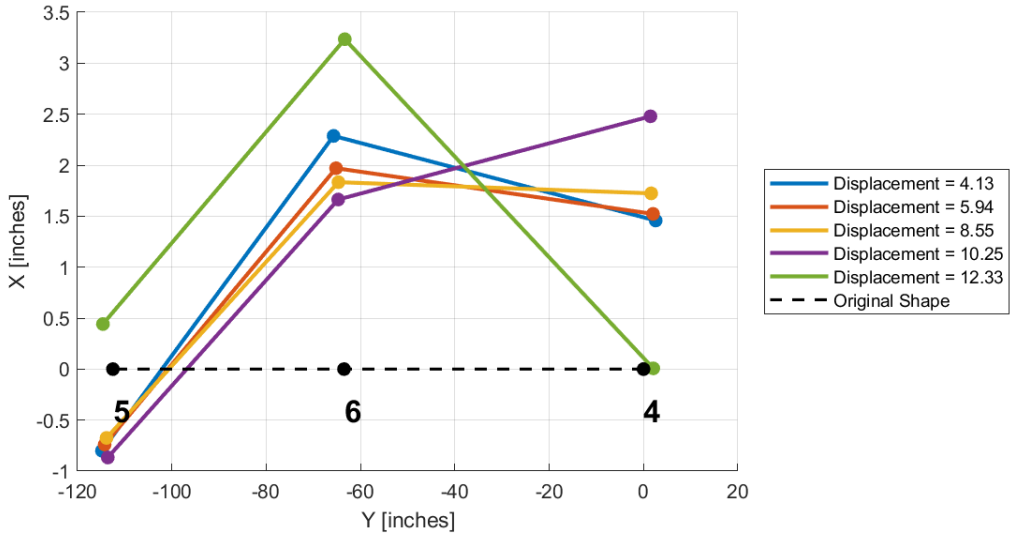


Figure 6.25: Two-dimensional deflection patterns of Targets #4–#6 under push loading (corroded specimen).

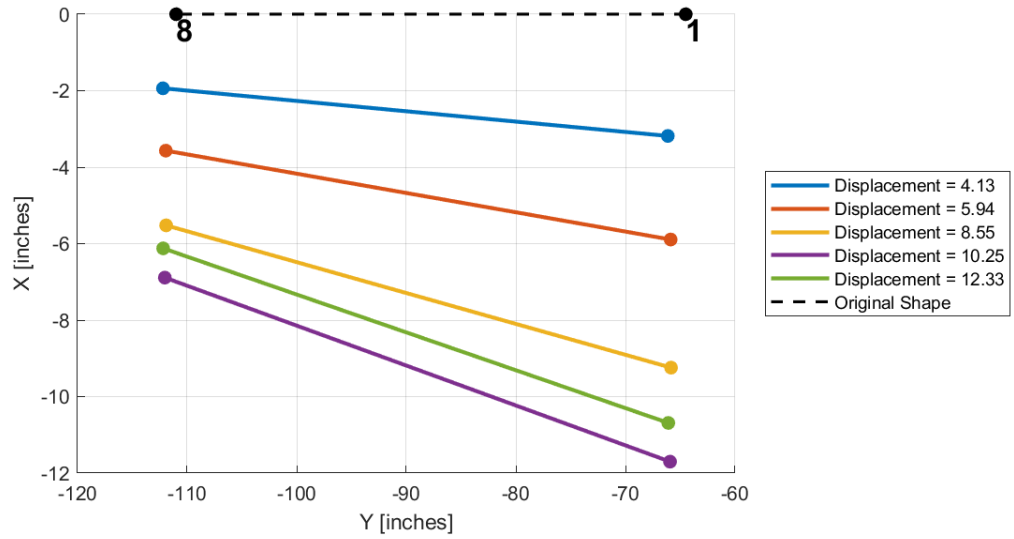
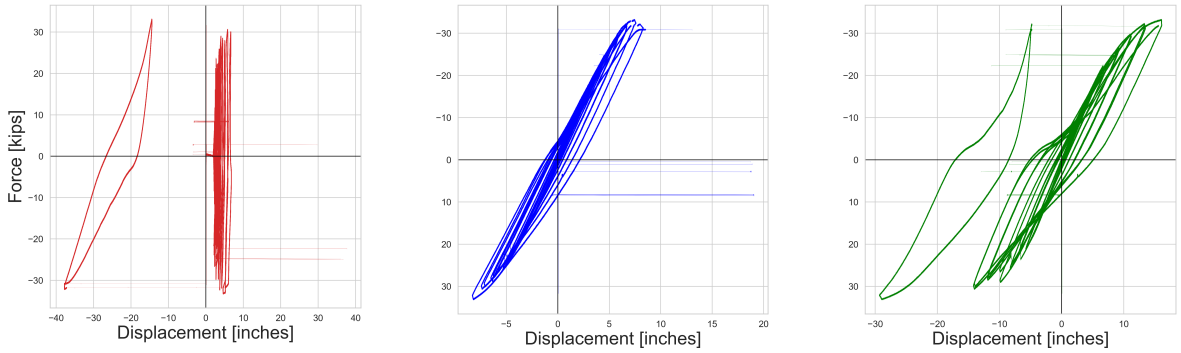


Figure 6.26: Two-dimensional deflection patterns of Targets #1 and #8 under push loading (corroded specimen).

indicates that corrosion significantly impacts the stiffness and strength of the specimen in the X-direction.In addition, when the force is in push direction, the maximum corresponding reaction force can reach 30 kips, which is around 3 times as the reaction force in corroded specimen.

- **Y-direction:** The force-displacement curve in the Y-direction (Figure 6.27b) is approximately linear, with an estimated longitudinal stiffness at target #4 of approximately $30/7 = 4.3$ kips/in, which is slightly smaller than the stiffness of corroded specimen. Note that this



(a) X-direction **(b) Y-direction** **(c) Z-direction**

Figure 6.27: Force–displacement relationships of Target #4 for the corrosion-free specimen in the X, Y, and Z directions.

value represents a lateral, coupling-induced response and is not directly comparable to the global axial stiffness in the Z-direction. However, a notable difference is observed: under loading in the same direction, the corrosion-free specimen exhibits displacement at target # 4 in Y-direction opposite to that in corroded specimen.

- **Z-direction:** Figure 6.27c depicted that the Z-direction force-displacement curve of corrosion-free specimen show more nonlinear configuration compared with corroded specimen. Notably, the hysteresis loops are significantly larger, indicating greater deformation capacity in the axial (global) response

The comparison between the corrosion-free and corroded specimens reveals significant differences in structural response due to material degradation. The corroded specimen exhibits more pronounced lateral deformation in the X-direction, indicating a reduction in lateral stiffness. In the Y-direction, both specimens show approximately linear behavior, but the apparent longitudinal stiffness of the corrosion-free specimen is slightly lower than that of the corroded specimen. In the Z-direction, the corrosion-free specimen exhibits an approximately axially symmetric response, with comparable stiffness and maximum displacement in both push and pull loading. This contrasts with the corroded specimen, which shows a pronounced reduction in maximum displacement and asymmetric behavior between two loading directions. In addition, the force–displacement curve of the corrosion-free specimen develops a noticeable pinching effect near the center, which is likely attributed to bolt loosening under cyclic loading. Such findings emphasize the need for corrosion-resistant materials and proactive maintenance strategies to ensure the long-term durability and reliability of structural components.

To further investigate the effect of corrosion on the structural response, the maximum displacement values under push and pull loading for both the corrosion-free and corroded specimens are compared. The results presented in Table 6.6(corrosion-free) and Table 6.5 (corroded) demonstrate significant differences in displacement magnitudes and deformation patterns.

The comparison between Tables 6.6 (corrosion-free specimen) and 6.5 (corroded speci-

Table 6.6: Maximum displacement values for corrosion-free specimen under push and pull loading.

Target	Push Direction			Pull Direction		
	X [in]	Y [in]	Z [in]	X [in]	Y [in]	Z [in]
1	–*	–*	–*	–*	–*	–*
2	-4.44	–*	–*	6.80	–*	–*
3	–*	–*	–*	–*	–*	–*
4	–*	–*	–*	–*	–*	–*
5	–*	–*	-16.73^{+}	–*	–*	8.89^{+}
6	8.95^{+}	-7.58^{+}	-9.87^{+}	8.90^{+}	-2.33^{+}	13.08^{+}
7	8.70	6.84	-5.63	-2.46	-9.05	8.63
8	-4.12	0.34	-7.92	-2.67	-3.64	7.41

*Due to WP damage, values are not reported.

+Values correspond to the maximum displacement recorded prior to WP damage observed during the final 1–2 loading cycles.

men) highlights distinct differences in structural response under push (negative Z) and pull (positive Z) loading. It should be noted that, due to extensive wirepot failures caused by large deformations during the final loading cycles, only a subset of displacement values is considered reliable for comparison. In particular, the measurements at Targets #6, #7, and #8 are the most representative and are therefore emphasized in the following discussion.

- **Push Direction (negative Z):** For the corrosion-free specimen (Table 6.6), measurable displacements are available mainly for Targets 2 and 6–8. Target points such as #6 (-9.87 inches) and #8 (-7.92 inches) show axial deformation. Some location (Target #5) exhibit unusually large value (-16.73 inches), which may because of large nonlinearity of structure or WP damage.

- **Pull Direction (positive Z):**Under pull loading, the corrosion-free specimen again exhibits larger displacement values compared to the corroded specimen. The Z-direction displacements under pull are positive and of similar magnitude to push at the same targets, indicating the approximately axially symmetric response between push and pull direction. By contrast, the corroded specimen shows a lower effective axial stiffness and a more pronounced push–pull asymmetry in Z direction, showing fewer energy dissipation because of corrosion.

The comparison between the two specimens highlights the detrimental effects of corrosion on structural deformation capacity. The corroded specimen exhibits significantly lower displacements, which may be attributed to localized corrosion-induced cracks, material loss, and stiffness increase due to rust accumulation. Additionally, the asymmetry in the displacement values in the corroded specimen suggests that corrosion-induced degradation is not uniform, leading to differential structural behavior under loading. In general, the corrosion-free specimen demonstrates a higher deformation capacity, suggesting a more ductile response under loading, whereas the corroded specimen exhibits reduced displacement magnitudes, indicating stiffness increase and potential brittle failure mechanisms. The loss of uniformity in deformation in the corroded specimen

further underscores the importance of corrosion monitoring and mitigation strategies to maintain structural integrity and prevent premature failure.

Similar to the deflected shape observed previously in the corroded specimen, Figures 6.28 and 6.29 illustrate the two-dimensional displacement of five target points on the corrosion-free specimen subjected to the same push deformation. Compared to the corroded specimen, the corrosion-free beam exhibits notably different deformation patterns. Targets #1 and #8 (shown in Figure 6.29), located closest to the actuator, consistently display downward vertical deformation, though with significantly smaller magnitudes than those observed in the corroded case. Meanwhile, targets #4, #5, and #6 (shown in Figure 6.28) exhibit obvious different behavior; all three targets move noticeably in the negative vertical direction. Particularly notable is target #6, which transitions from positive displacement in the corroded specimen to negative displacement in the corrosion-free specimen, highlighting a significant bending curvature occurring between the midspan and the beam tip. These comparative observations highlight how corrosion affects structural rigidity, resulting in larger and more asymmetric deformations than those seen in the corrosion-free specimen.

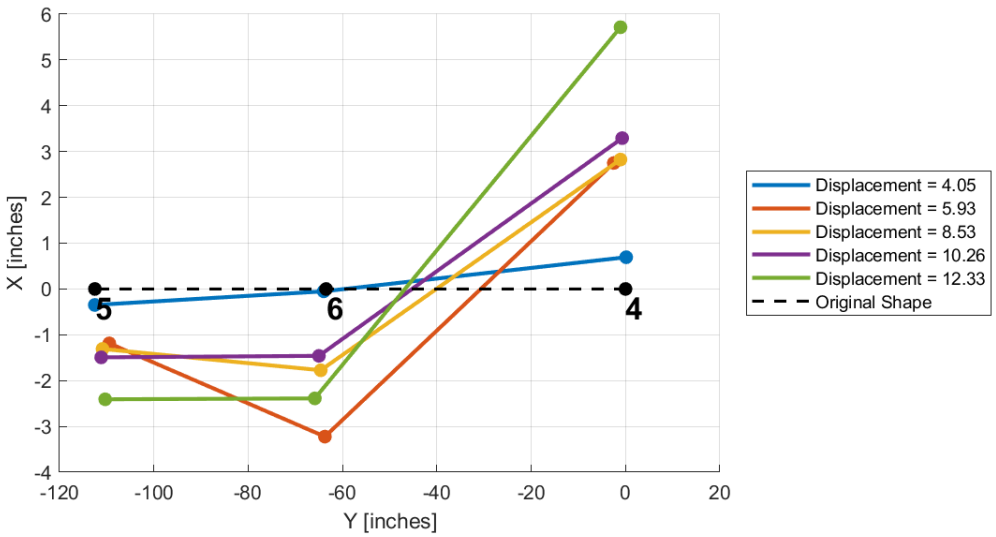


Figure 6.28: Deflection shape of targets #4, #5 & #6 of corrosion-free specimen.

6.6 LVDT RESULTS

In this experiment, a pair of Novotechniks Linear Variable Differential Transformers (LVDTs) were installed in the connection of beam and column on the tested structure to measure the curvature at the critical cross-section, as illustrated in Figure 5.12. The LVDTs were arranged vertically with a known separation distance $L = 23$ inches, capturing the differential displacement between the top and bottom surfaces of the beam section during cyclic loading. The result is shown in Figure 6.30.

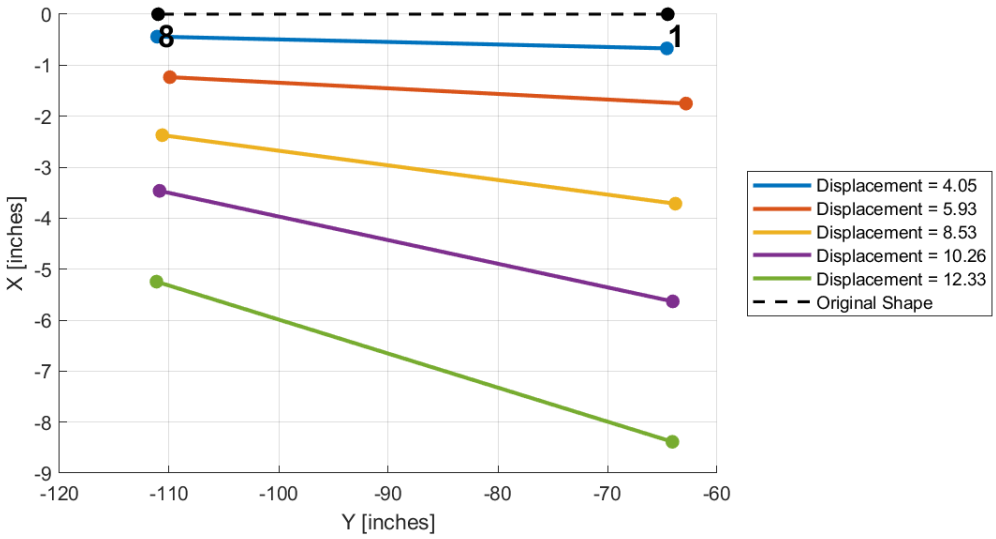
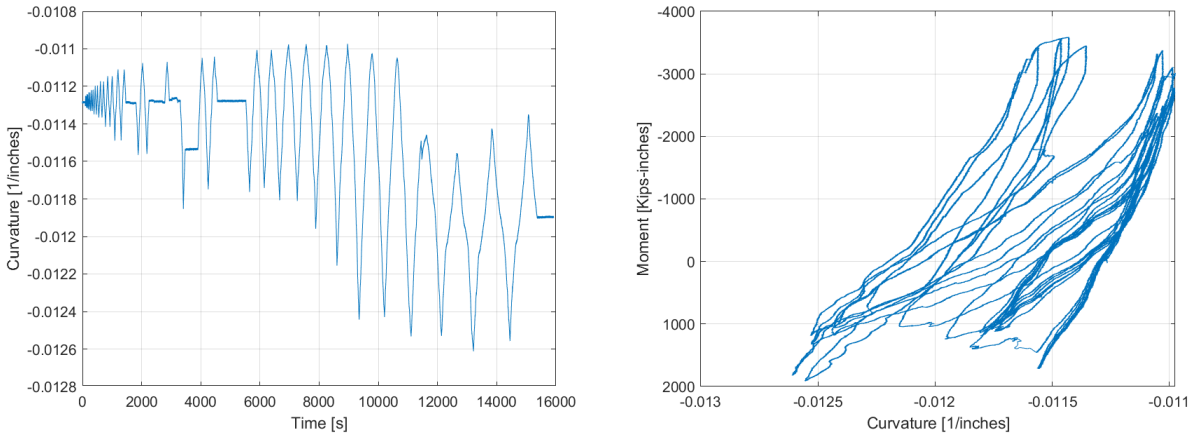


Figure 6.29: Deflection shape of targets #1 & #8 of corrosion-free specimen.



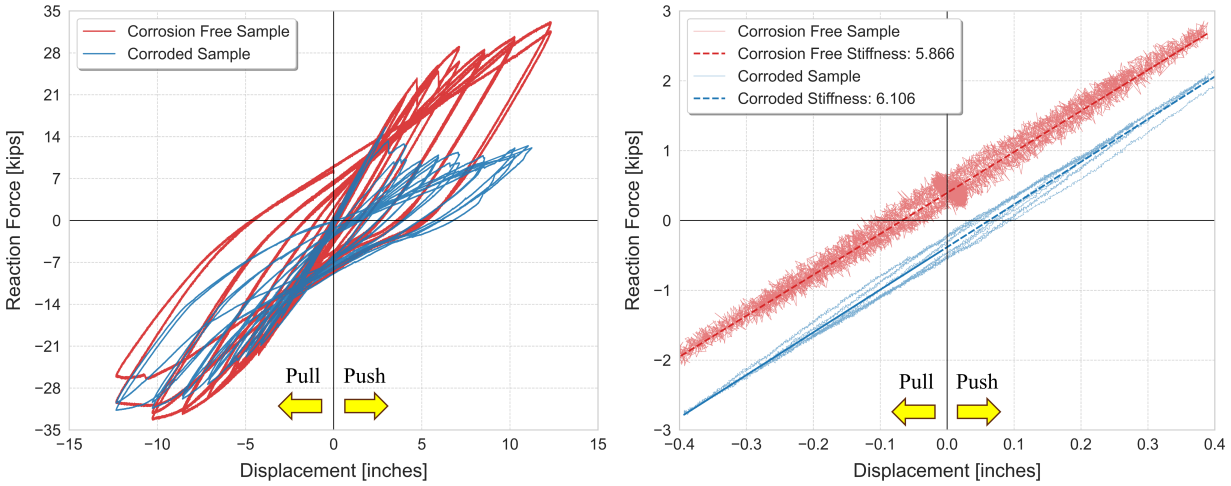
(a) Curvature-time relationship **(b) Moment-curvature relationship**

Figure 6.30: LVDT results of corroded specimen.

The resulting curvature data was synchronized with internal load cell measurements to produce moment–curvature hysteresis loops, providing insight into the local nonlinear flexure behavior of the cap beam. As shown in the curvature-moment plot (Figure 6.30b), the beam exhibits significant stiffness degradation and energy dissipation, indicative of cyclic inelastic behavior. Additionally, the curvature-time history (Figure 6.30a) demonstrates the progression of curvature under increasing lateral displacement cycles, with a noticeable increase in peak curvature amplitudes at higher drift levels, reflecting structural softening and accumulation of damage. For corrosion-free specimen, since the displacement is too large, the LVDTs are heavily damaged so the results have no value to be shown here (refer to Figure 6.27).

6.7 FORCE-DISPLACEMENT AND ENERGY DISSIPATION RESULTS

Figure 6.31a presents the force-displacement relationship obtained from the actuator of the corroded and corrosion-free specimens (positive displacements denote push, negative denote pull). Notably, in the pull direction, both specimens reached peak reactions of approximately 30 kips. In the push direction, however, the corroded specimen experiences a significant reduction in reaction force, with the peak force limited to about 15 kips, whereas the corrosion-free specimen can still reach 30 kips, demonstrating symmetric push-pull strength. This indicates that corrosion has a substantial impact on the push capacity, reducing it by at least 50%, while also affecting the overall failure behavior. In contrast, By the initial stiffness appears comparable for the two specimens (Figure 6.31b), suggesting negligible corrosion effects on initial stiffness.



(a) Force-displacement relationship comparison **(b) Initial stiffness comparison**

Figure 6.31: Force-displacement relationship before and after loading.

Combining the findings from Figures 6.31a and 6.31b, it is evident that the failure of structure corresponds to large displacements, which is further validated by the more detailed force–time relationships. Figure 6.32 compares the force–time relationships of the corroded and corrosion-free specimens. This representation provides direct identification of the corresponding reaction forces in each loading cycle. Similar to the results observed in Figure 6.31, the reaction forces in the pull direction are nearly the same for both specimens throughout the cycles. In contrast, as shown in Figure 6.32, when the displacement reaches 2.95 inches, both specimens are still able to sustain an identical reaction force of 15 kips. However, at a displacement of 4.132 inches (indicated by the black arrows in the figure), the maximum load-carrying capacity of the corroded specimen (blue curve) drops markedly, from the initial 15 kips to approximately 10 kips, which is only about half of that of the corrosion-free specimen at the same displacement. In the subsequent cycles, the reaction force of the corroded specimen remains within the range of approximately 9–15 kips, which corresponds to only about one-half to one-third of the peak loads observed in the corrosion-free specimen during the same cycles. It is suspected that the ribbed sheet–angle detachment occurred during the cycle corresponding to the 4.132 inches displacement. As the cyclic loading progressed,

this damage likely became more serious, yet the whole structure was still able to sustain a baseline load-carrying capacity of approximately 10 kips.

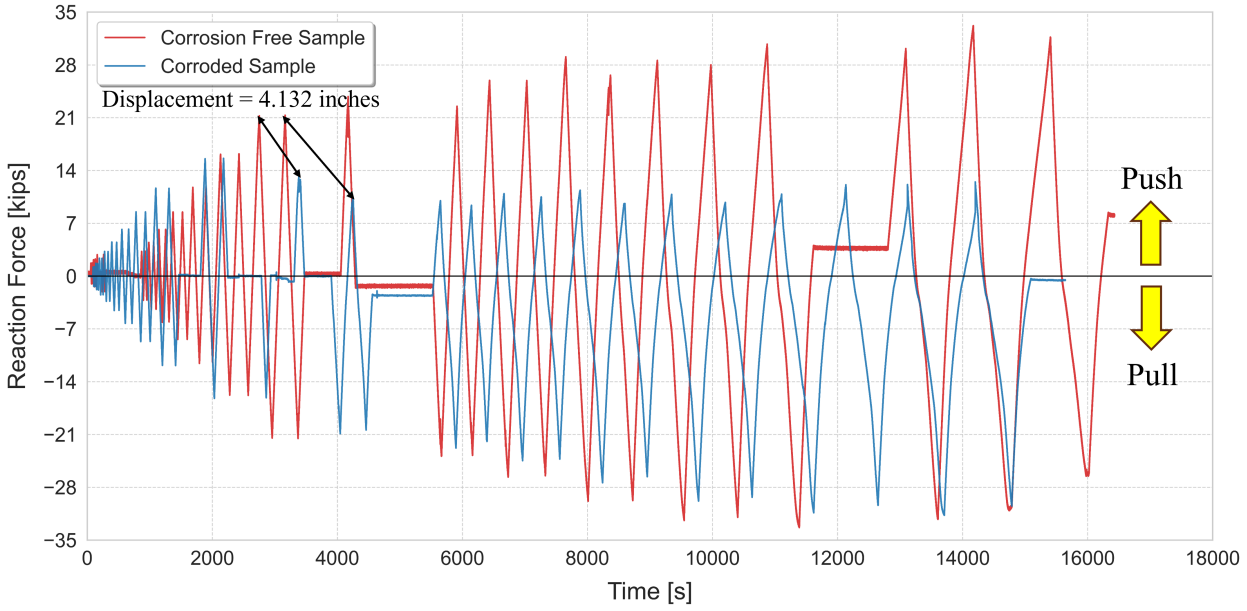


Figure 6.32: Force-time relationship comparison

Finally, we evaluate the performance of the two specimens using energy dissipation. The energy dissipation is a critical indicator of structural resilience, reflects the capacity of a material to absorb and dissipate input energy, mitigating failure risks under repeated loading conditions. The cumulative energy was computed directly from the measured force–displacement records using a discrete work summation. The incremental energy at each step i is calculated as:

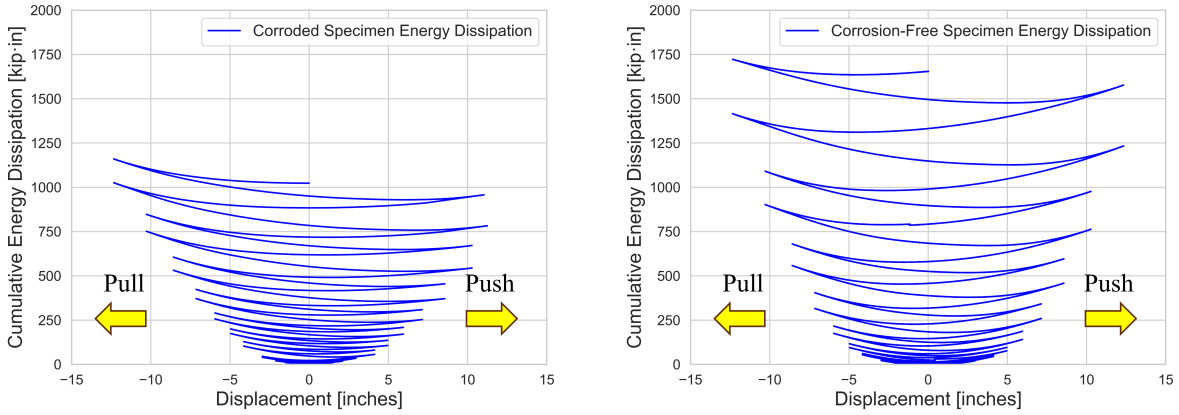
$$\Delta E_i = F_i \Delta u_i = F_i(u_i - u_{i-1}) \tag{6.1}$$

and the cumulative energy over the entire time series is

$$E_{\text{cum}}(n) = \Sigma_i^n \Delta E_i \tag{6.2}$$

Figure 6.33 illustrates the cumulative energy dissipation of corroded and corrosion-free specimens throughout cyclic loading. The plots indicate that the corrosion-free specimen exhibits significantly greater energy dissipation, with values reaching approximately 1,750 kip·in at maximum displacement, compared to the corroded specimen, which only attains around 1,150 kip·in. This discrepancy further underscores the detrimental impact of corrosion on structural performance, as the reduced energy absorption capacity compromises the specimen's ability to withstand cyclic loading. Moreover, in the push direction, the corroded specimen exhibits a markedly lower rate of cumulative energy growth than both its pull-direction response and the push response of the corrosion-free specimen. This indicates that corrosion and the detachment at ribbed sheet-angle connection limits the hysteresis-loop area and reduces energy dissipation under push loading. By contrast, the corrosion-free specimen shows sustained growth of cumulative energy with increasing displacement amplitude. These observations are consistent with the force–displacement asymmetry in

Figure 6.31, i.e., reduced push capacity in the corroded specimen translates into diminished energy accumulation. In summary, the results suggest greater vulnerability of the corroded specimen under push-dominated demands and motivate strengthening or retrofit of the ribbed sheet–angle connection. To better interpret and extend the experimental observations, a calibrated FE model was developed using Abaqus simulation, as discussed in the following chapter.



(a) Corroded specimen **(b) Corrosion-free specimen**

Figure 6.33: Cumulative energy dissipation.

7 Automated Abaqus Simulation of Corrosion Effects and Structural Strength Loss

As discussed in the previous chapters, computer-vision (CV) techniques were employed to identify corrosion on overhead highway sign structures, and a script based on Abaqus–Python API was developed to automate FE model generation. Building on these developments, this chapter (i) calibrates the FEM with the testing results, (ii) characterizes large numbers of statistically diverse corrosion scenarios, (iii) propagates these scenarios through an updated FE model simulation to obtain structural response, and (iv) establishes a refined quantitative relationship between the Corrosion Index (CI) and the resulting in-plane strength of the sign structure in the vertical direction. All FE simulations were conducted using Abaqus 2020.

7.1 FINITE ELEMENT MODEL CALIBRATION AND SIMULATION

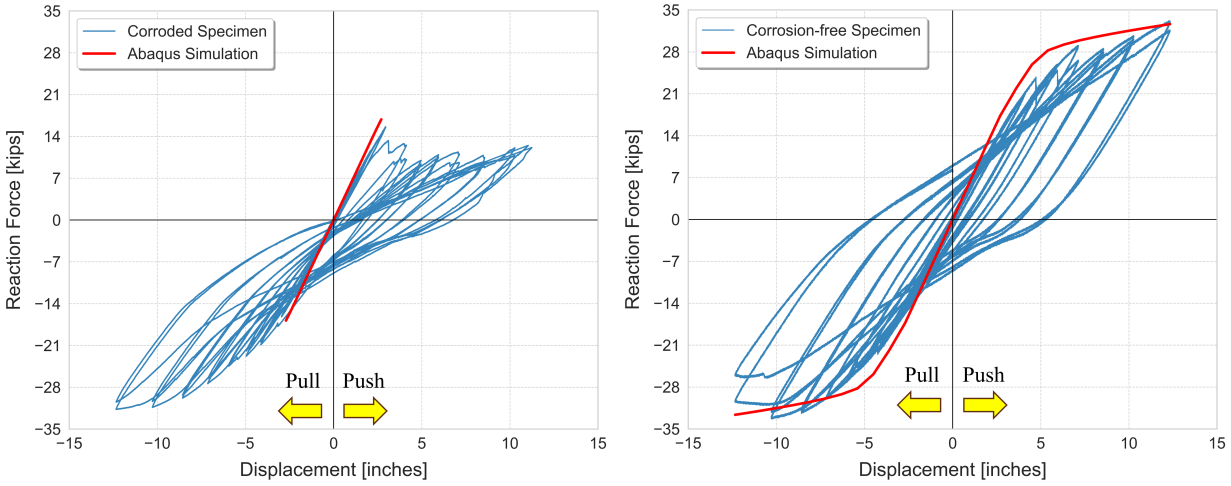
Based on the experimental results presented in Chapter 6, particularly the force–displacement relationship shown in Figure 6.31, the Abaqus material definitions were calibrated to better match the observed structural behavior. A set of calibrated parameters was modified based on the testing results and material tests from sign structures constructed in a similar era. Table 7.1 summarizes these parameters used in the FE model. The connection region, defined as the interface between the post and the box beam, required further refinement of material properties to better match the observed fracture behavior. This adjustment was made because Abaqus cannot accurately represent the pinching effect at the connection under cyclic loading (Mousavi et al., 2014; Wan et al., 2001).

Based on the calibrated parameters, both corroded and corrosion-free specimens were constructed and analyzed under monotonic pushover analysis. A static solver with displacement-controlled boundary condition was utilized to simulate pushover analysis. Figure 7.1 presents a direct comparison between the Abaqus pushover simulation (red curve) and the experimental cyclic loading results (blue curve). The calculation of the vertical reaction force and force capacity is explained in Section 7.4.

In the case of the corroded specimen (refer to Figure 7.1a), the Abaqus simulation closely matches the experimental envelope up to approximately 3 inches of displacement. At this displacement level, the Abaqus simulation terminated prematurely due to convergence failure (numerical instability), as explicitly indicated by abrupt increases in residual forces and large displacement

Table 7.1: Calibrated Material Properties used in Abaqus Simulation

Material	Property	Unit	Value	Plastic Strain
Original Steel from Post	Elastic Modulus	ksi	30,397	–
	Yield Stress	ksi	42.80	0.00
	Ultimate Stress	ksi	70.13	0.19
New Steel from Angle	Elastic Modulus	ksi	31,232	–
	Yield Stress	ksi	57.37	0.00
	Ultimate Stress	ksi	79.30	0.16
Connection Region	Elastic Modulus	ksi	18,238	–
	Yield Stress	ksi	20.68	0.00
	Ultimate Stress	ksi	58.08	0.16
Corroded Ribbed Sheet	Elastic Modulus	ksi	6,991	–
	Yield Stress	ksi	42.80	0.00
	Ultimate Stress	ksi	70.13	0.19
	Thickness	inch	0.014	–



(a) Corroded specimen **(b) Corrosion-free specimen**

Figure 7.1: Comparison of experimental and simulated force–displacement responses.

corrections recorded in the Abaqus output message file (*.msg). This numerical instability is consistent with the severe material degradation observed at the end of the second 3” amplitude cycle, capturing the actual corrosion-induced buckling of the corroded ribbed sheet metal observed experimentally. This highlights the fidelity of the calibrated Abaqus model, which fails not only at nearly the same displacement and peak reaction force as observed in the testing, but also captures the key damage event of the corroded specimen, which initiates at the ribbed sheet metal. This event is critical in the damage progression as evidenced by the smaller push direction force peaks of all subsequent cycles. It is noted that no attempt has been made to overcome this numerical instability as it is out of the scope of this project. Conversely, for the corrosion-free specimen (see Figure 7.1b), the Abaqus simulation exhibits an excellent match with the experimental results.

The simulated force–displacement curve aligns well with the experimentally measured hysteresis loops, successfully capturing both the initial stiffness and the peak reaction force capacity of the uncorroded specimen.

7.2 CORROSION INDEX DEFINITION

To quantify the impact of corrosion on the overall structural strength of different ribbed sheets, we introduce the concept of Corrosion Index (CI). As a starting point, we simulate the purely elastic, corrosion-free specimen under gravitational loading and obtain a 2D axial stress map in (S_{22} in Abaqus) over the surface of the sign beam. We adopted the purely elastic stress pattern to establish a conservative lower-bound solution, which can be applied to different nonlinear deformation patterns. The axial stress is oriented along the beam's longitudinal direction and serves as the baseline stress field for subsequent corrosion-related analysis.

Considering the mesh generated by Abaqus FE is too dense, which may make the subsequent process of selecting elements and assigning corrosion values significantly complicated. A Python script (see Appendix F) partitions the ribbed-sheet region into a 30×10 uniform grid ($30\,\text{columns} \times 10\,\text{rows}$). In each grid cell, the centroid $(x_i^c,\ z_i^c)$ is used to query pre-computed axial stress σ we have before; resulting Figure 7.2. For generating the axial stress map from the Abaqus model, a relatively larger element size was used in the meshing step to align with the 30×10 grid of the beam structure. This choice ensures that the elements in each grid cell can better represent the average local stress.

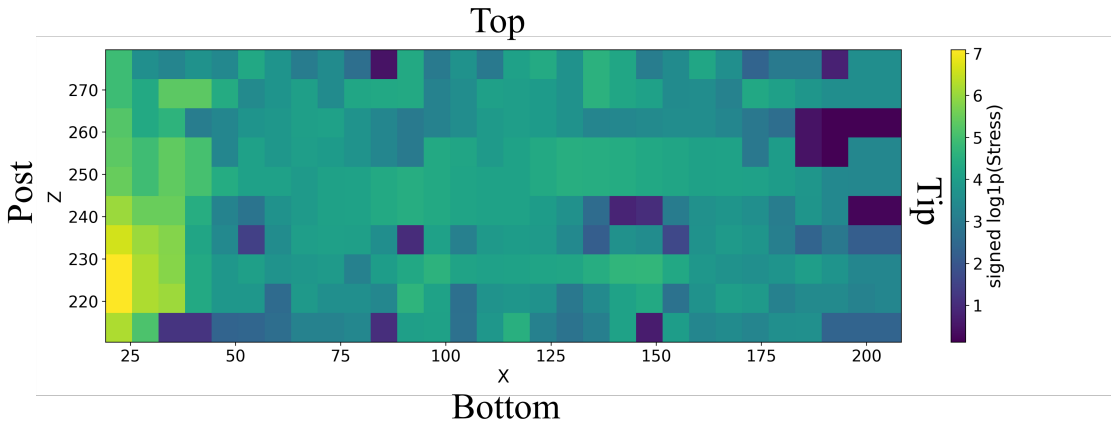


Figure 7.2: Axial stress map: Element-wise stress distribution from Abaqus simulations used for CI computation

In Figure 7.2, the left side corresponds to the post, while the right end represents the free tip of the cantilever beam. It can be observed that the stress is significantly higher near the bottom-left corner (at the bottom of the beam near the post), indicating that this region bears a greater share of the structural strength. In other words, if corrosion occurs in this area, its impact on the overall strength would be much more severe compared to the corrosion occurring near the tip of

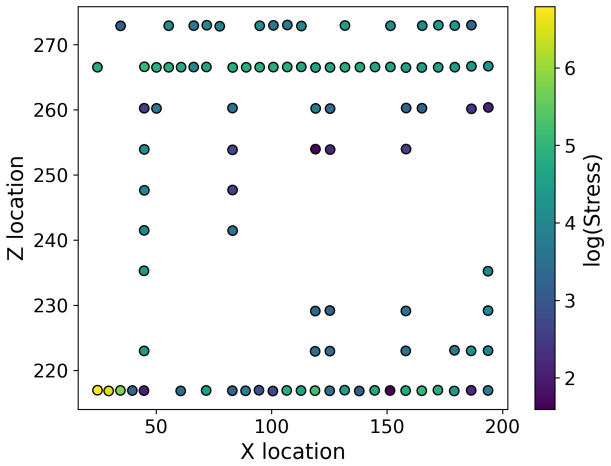


Figure 7.3: Element-wise stress distribution for corroded cells (corrosion distribution refers to Figure 3.9)

the beam. Figure 7.3 visualizes the stress distribution restricted to corroded cells from the corroded specimen, derived from the grid-wise corrosion pattern in Figure 3.9.

As introduced before, the use of a purely elastic, corrosion-free model and the axial stress distribution as the weighting factor allows the resulting stress map to be directly applied to similar structures with different corrosion patterns, without the need to rerun FE simulations. For this reason, the axial stress distribution under self-weight is adopted as an effective weighting factor in the definition of the CI, which is given as:

$$\mathrm{CI} \;=\; \sqrt{\left(\frac{\displaystyle\int_{\Omega_{\mathrm{corr}}} \tfrac{1}{2}\,\boldsymbol{\sigma} : \boldsymbol{\varepsilon}\, dV}{\displaystyle\int_{\Omega_{\mathrm{all}}} \tfrac{1}{2}\,\boldsymbol{\sigma} : \boldsymbol{\varepsilon}\, dV}\right)} \;\approx\; \sqrt{\left(\frac{\displaystyle\sum_{i}^{\mathrm{corr}} \frac{\sigma_i^2}{E_i}\, t_i\, A_i}{\displaystyle\sum_{j}^{\mathrm{all}} \frac{\sigma_j^2}{E_j}\, t_j\, A_j}\right)} \tag{7.1}$$

where

- Numerator Ω_{corr} and denominator Ω_{all} are the corroded subdomain and the entire structural domain, respectively.
- σ_k denote the axial normal stress (oriented along the beam axis) at cell k under linear elastic solution.
- E_i, t_i and A_i are elastic modulus, thickness, and Area at each cell respectively, where the three parameters are for the corrosion-free structure.
- Generally, this formulation corresponds to the integration of strain-energy density over the volume. The strain energy over the volume of a cell represents the weight (importance) of the cell and the CI can be interpreted as the sum of the weights of all corroded cells divided by

the sum of the weights of all cells. Use of such weights in the equation allows distinguishing locations that contribute more to the load carrying capacity, where the effects of corrosion would be more consequential, from those locations that contribute less to the load carrying capacity, where the effects of corrosion would be less pronounced.

- Summarizing, in the above expression, the quantity $\frac{1}{2}\boldsymbol{\sigma} : \boldsymbol{\varepsilon}$ represents the strain energy density, defined as one half of the double contraction[1] of the stress tensor $\boldsymbol{\sigma}$ with the strain tensor $\boldsymbol{\varepsilon}$. The numerator integrates this strain energy density over the *corroded domain* Ω_{corr}, while the denominator integrates it over the *entire domain* Ω_{all}. Thus, the ratio inside the square root measures the fraction of the total strain energy associated with corrected regions. The square root is introduced to rescale the strain-energy ratio into a stress-like indicator, making the values more evenly distributed between 0.0 and 1.0, where $\text{CI} = 0.0$ corresponds to the intact (corrosion-free) state and $\text{CI} = 1.0$ corresponds to the fully corroded state. Intermediate values reflect the proportion of strain-energy–weighted loss of material, with higher values indicating more severe corrosion in structurally significant regions.

7.3 PROBABILISTIC CORROSION PATTERN GENERATION USING TARGET MAPS AND TRUNCATED GAUSSIAN SAMPLING

After the CI formulation was established, a Python script was developed to automatically generate synthetic corrosion patterns based on the predefined 30×10 grid layout covering the structure's ribbed sheet surface. Each cell represents a rectangular area with dimensions $\Delta x \times \Delta z$, ensuring spatial coverage and resolution similar to the actual stress map used in FEA.

To enhance physical realism, we construct the **target probability map** with a **truncated Gaussian** kernel (shown in Figure 7.4). The gaussian distribution is truncated beyond a radius of three grid cells (standard deviation $\sigma = 1$). This approach concentrates higher probabilities around observed corrosion locations (refer to Figure 3.9), which aligns with the real-world observation that corrosion tends to grow around existing damaged areas rather than appear randomly across

[1] The operator ":" denotes the *double contraction* between stress tensor $\boldsymbol{\sigma}$ and strain tensor $\boldsymbol{\varepsilon}$, defined in 3D as

$$\boldsymbol{\sigma} : \boldsymbol{\varepsilon} = \sum_{i=1}^{3}\sum_{j=1}^{3} \sigma_{ij}\,\varepsilon_{ij} \tag{7.2}$$

This operation is the tensor analogue of the dot product for vectors, and in mechanics it yields the scalar quantity representing the stress–strain work density. In *Voigt notation*, where the stress and strain tensors are expressed as 6-component vectors,

$$\boldsymbol{\sigma} = \begin{bmatrix}\sigma_{11} & \sigma_{22} & \sigma_{33} & \sigma_{23} & \sigma_{13} & \sigma_{12}\end{bmatrix}^T, \quad \boldsymbol{\varepsilon} = \begin{bmatrix}\varepsilon_{11} & \varepsilon_{22} & \varepsilon_{33} & 2\varepsilon_{23} & 2\varepsilon_{13} & 2\varepsilon_{12}\end{bmatrix}^T \tag{7.3}$$

The double contraction expands as

$$\boldsymbol{\sigma} : \boldsymbol{\varepsilon} = \sigma_{11}\varepsilon_{11} + \sigma_{22}\varepsilon_{22} + \sigma_{33}\varepsilon_{33} + 2\big(\sigma_{23}\varepsilon_{23} + \sigma_{13}\varepsilon_{13} + \sigma_{12}\varepsilon_{12}\big) \tag{7.4}$$

The factors 2 associated with the shear strain terms arise from the relationship between the engineering shear strain γ_{ij} and the tensorial strain ε_{ij}, i.e., $\gamma_{ij} = 2\varepsilon_{ij}$. In the present study, since the indicator is governed by the axial response, only the $\sigma_{22}\varepsilon_{22}$ term in Equation 7.4 is considered.

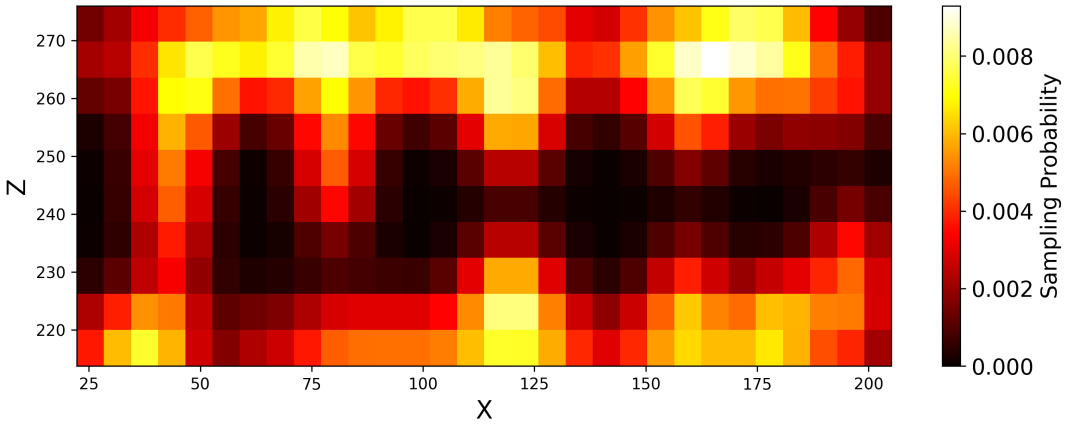
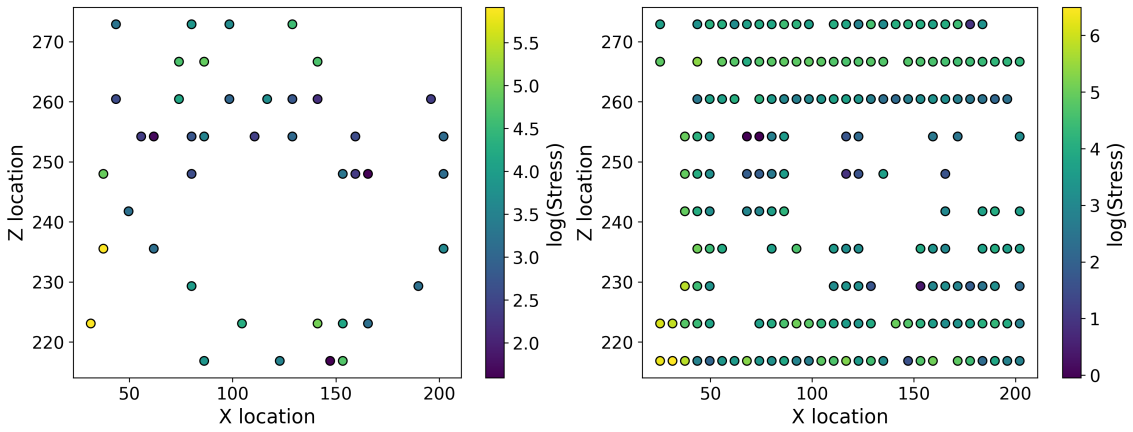


Figure 7.4: Target probability map constructed using a truncated Gaussian distribution centered at observed corrosion locations.

the surface. Truncation reduces probability leakage beyond the intended ribbed-sheet area, limits the influence of cells far from observed corrosion locations, and helps control variance in pattern generation. A small uniform baseline is then added so that every grid cell retains a non-zero probability, ensuring complete spatial coverage during sampling. This probability field is used directly to generate corrosion patches for each synthetic realization.

In each synthetic realization, a random number of corrosion patches is sampled, typically ranging from 10 to 300 regions. These grid cells are selected *without replacement* based on the target probability map described earlier, ensuring both spatial diversity and probabilistic fidelity to observed corrosion patterns. The output includes (i) individual CSV files (`corrosion_XX.csv`) listing the bounding coordinates of each corrosion patch, and (ii) a summary file (`CI_list.csv`) recording the CI value for each realization. Figure 7.5 illustrates two representative virtual corrosion patterns generated using this probabilistic sampling strategy.



(a) CI: 0.25 **(b) CI: 0.58**

Figure 7.5: Examples of synthetic virtual corrosion patterns with corresponding CI values.

7.4 BASE REACTION FORCE EXTRACTION AND FORCE CAPACITY CALCULATION

To evaluate the structural response under gravitational and lateral loading, the vertical reaction force (RF), defined as the global Z-axis in the Abaqus model (align with the pushover direction), is extracted from all nodes located at the fixed base of the specimen (see Figures 3.6 & 6.1). A Python script (see Appendix G) automates the retrieval of nodal reaction forces from the Abaqus output database (ODB) file.

Specifically, for each output frame of the nonlinear pushover analysis, the vertical reaction forces at all nodes in the predefined base nodes set are extracted from the ODB file and get summed. This total reaction force serves as a global strength indicator for the structure under a given corrosion pattern. The output is stored in a CSV file, with the first column representing the simulation time (or displacement step), and each subsequent column corresponding to the vertical reaction force at a specific node. This global reaction force measurement provides a more reliable assessment of structural capacity than single-node monitoring, especially under irregular corrosion distributions.

7.5 CORROSION-DRIVEN FEM AUTOMATION AND SIMULATION WORKFLOW

A unified python script was developed by integrating all previously established modules, including Abaqus mesh generation, corrosion pattern application, pushover analysis, as well as results extraction within the Abaqus ODB file. This automation enables efficient creation and simulation of the FEM under varying corrosion scenarios. With the synthetic corrosion datasets generated earlier, the script carries out the following tasks:

1. **Model creation and crossion application** – The script constructs the FE mesh and maps each corrosion patch from the corresponding `corrosion_XX.csv` file onto the shell elements in the specimen. Corroded regions are assigned a reduced thickness, based on experimental-simulation calibration, see Table 7.1.

2. **Nonlinear pushover analysis** – A displacement-controlled pushover analysis is performed in the in-plane (IP) direction with a target displacement amplitude of $12\,\mathrm{in}$.

3. **Reaction-force extraction** – At each analysis frame, the vertical reaction forces (RF) from all nodes in the predefined fixed base node set are extracted and summed to obtain the force capacity. This summation provides a more robust indicator of structural strength under varying corrosion scenarios.

4. **Data processing** – The calculated Corrosion Index (CI) and the corresponding force capacity are collected on each iteration for subsequent regression analysis and model calibration.

7.6 A DATA-DRIVEN QUANTITATIVE RELATIONSHIP BETWEEN CI AND NORMALIZED FORCE CAPACITY

This section investigates how corrosion severity, quantified by the Corrosion Index (CI), affects the structural load-bearing capacity, referred to here as the *normalized force capacity*. As introduced earlier, the force capacity is calculated by summing the vertical reaction forces across all nodes at the fixed base of the specimen. To improve the visualization ability, the force capacity values are normalized to a percentage scale, where 100% represents to the corrosion-free specimen. This not only makes the degradation trend easier to interpret and compare across different corrosion scenarios, but also enables the fit curve to be utilized in other sign structure with different sizes.

Figure 7.6 illustrates the relationship between CI and the normalized force capacity (C_f) obtained from FE simulations. Each blue dot corresponds to a synthetic corrosion realization, while red points represent experimental validation cases. A five parameters logistic (5PL) function is employed to model the nonlinear decay pattern as:

$$C_f(\mathrm{CI}) = L + \frac{U - L}{\Big(1 + \exp\big(k(\mathrm{CI} - \mathrm{CI}_{\mathrm{loc}})\big)\Big)^{\nu}} \tag{7.5}$$

where C_f denotes the normalized force capacity ($C_f = 100\%$ at the corrosion-free state), CI represents the corrosion index ($\mathrm{CI} = 0.0$ at the corrosion-free state), all parameters are defined and listed in Table 7.2.

Table 7.2: Fitted parameters for the five parameters logistic regression

Parameter	Meaning	Value
U	Maximum asymptote: normalized capacity at zero corrosion (%)	100
L	Minimum asymptote: residual capacity at severe corrosion (%)	51.8
k	Hill's slope or steepness of the curve	12.84
$\mathrm{CI}_{\mathrm{loc}}$	Location parameter controls the horizontal shift of the logistic curve	0.59
ν	Asymmetry (shape) factor: adjusts curve skewness	10

This logistic form effectively captures the initially stable force capacity at low corrosion levels, followed by a sharp decline and eventual stabilization as corrosion severity increases. The fitted curve achieves a coefficient of determination of $R^2 = 0.70$, indicating a relatively strong correlation between simulated data and the model. Importantly, the experimental points lie close to the fitted trend, further validating its predictive value across both synthetic and real-world data. The Mean Absolute Error (MAE = 8.9%) and Root Mean Squared Error (RMSE = 11.2%) appear moderate due to variability among different corrosion patterns, the model successfully captures the overall nonlinear degradation trend.

From this figure, it is observed that structures, with important load carrying locations severely corroded (e.g., $\mathrm{CI} = 0.5$), may have a strength loss of more than 45%. Furthermore, it is observed that the force capacity reduction saturates at approximately 48%, with several points near $\mathrm{CI} = 0.8$ experiencing a force capacity reduction of 40%. This saturation and no further

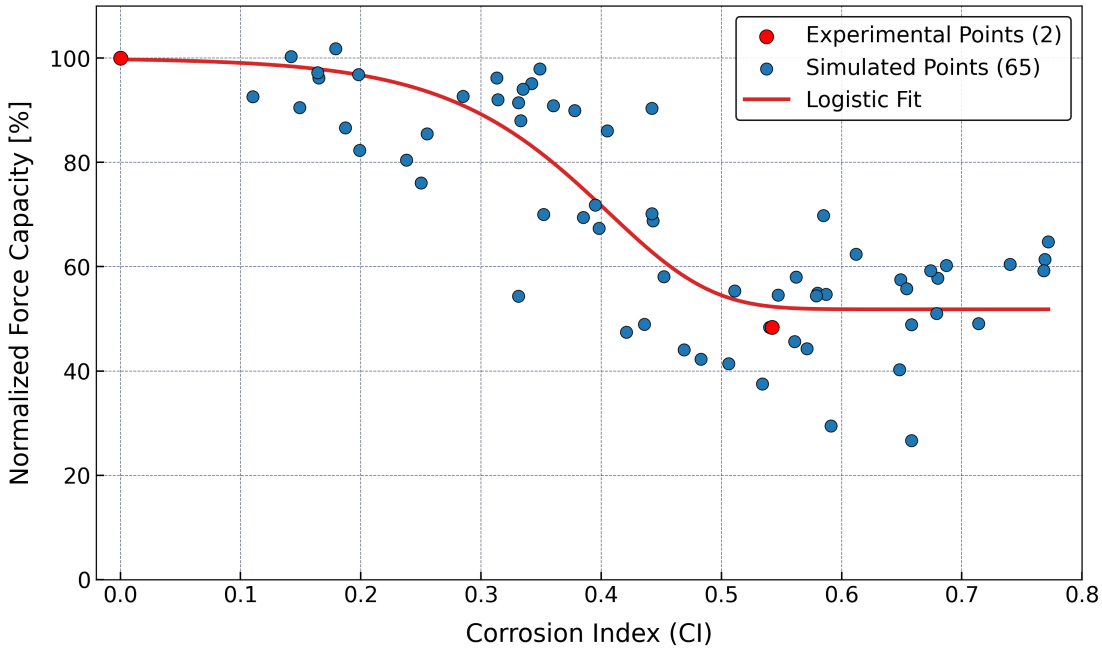


Figure 7.6: Logistic fit between Corrosion Index (CI) and Normalized Force Capacity.

reduction is due to a finite level of strength assumed to be available at the locations of corrosion, rather than losing the strength entirely at these locations.

Furthermore, the lower asymptote ($L = 51.8\%$) or the plateau at high CI has both physical and modeling origins. Physically, only one surface of the ribbed sheet is corroded; whereas the opposite face is shielded by the sign plate and kept corrosion-free. During the testing of corroded structure, the corroded ribbed sheet metal experienced major damage, while the uncorroded ribbed sheet surface remained undamaged and intact with the box beam, which continued to provide load resistance (As discussed earlier, this is the main reason for the major twisting observed during the corroded structure tests). Modeling-wise, an equal-displacement constraint was imposed on both faces in the Abaqus pushover setting, so the uncorroded face carries a non-negligible share of the load even when the other face's elastic modulus and thickness are degraded. Finally, based on the testing results, the corroded specimen with $\mathrm{CI} = 0.54$ was already unserviceable; therefore, for this configuration, $\mathrm{CI} \geq 0.54$ should be interpreted as severe corrosion.

7.7 AREA-BASED CORROSION INDEX CALCULATION

Although FEA combined with stress mapping provides a highly accurate evaluation of the residual capacity of corroded sign structures, it is computationally expensive and labor-intensive for routine field inspections. Field engineers require a rapid, accessible tool that relies purely on visual data (e.g., inspection photographs) without the necessity of building complex numerical models for every structure.

To address this important practical aspect, we first investigated a purely visual, geometry-

based metric: the Conventional Area Ratio Index (CI_{A}). This approach defines the corrosion index simply as the total corroded area normalized by the total area of the panel:

$$CI_{\mathrm{A}} = \frac{A_{\mathrm{corr}}}{A_{\mathrm{all}}} = \frac{\sum_i^{\mathrm{corr}} A_i}{\sum_j^{\mathrm{all}} A_j} \tag{7.6}$$

As shown in Figure 7.7, while this baseline visual model captures the general trend of capacity degradation, its predictive accuracy is limited, yielding a coefficient of determination of $R^2 = 0.61$. This dispersion occurs because the simple area ratio fails to account for the spatial distribution of the damage. In a cantilever structure, severe corrosion at the free end has limited impact on the overall structural integrity, whereas corrosion at the fixed support end can cause catastrophic failure.

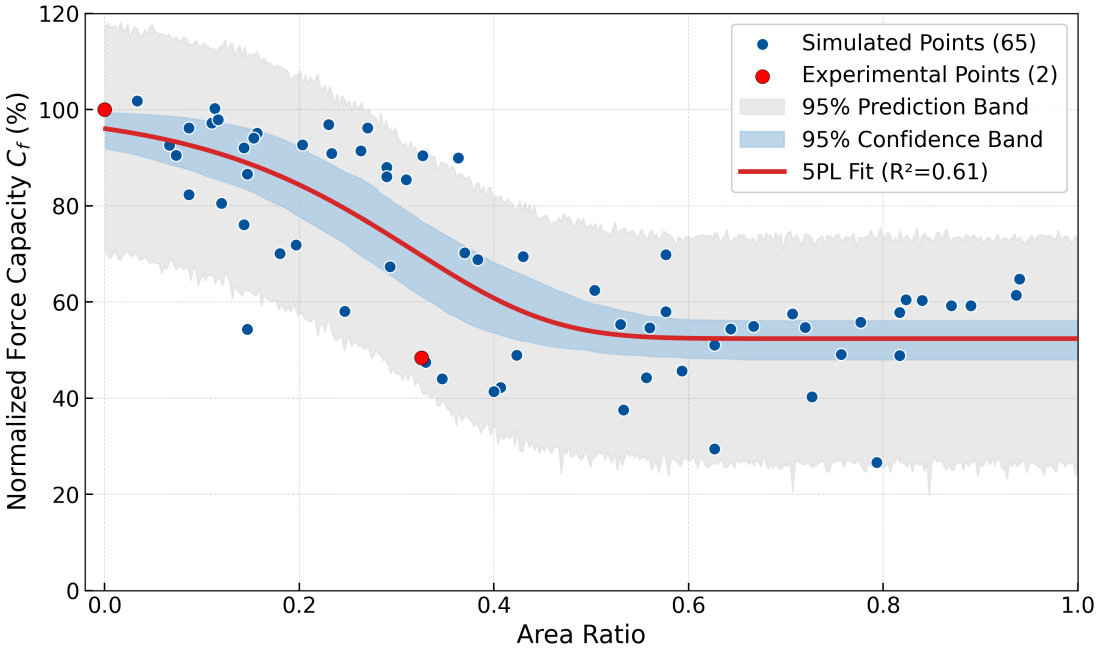


Figure 7.7: Conventional Area Ratio Model.

To improve the predictive performance while maintaining the simplicity of a purely visual assessment, we proposed a refined Location-Weighted Corrosion Model (CI_{L}). Based on the fundamental bending mechanics of cantilever plates, stress concentrations are predominantly localized at the fixed connection and along the top and bottom frames. Therefore, we divided the panel into a simplified evaluation grid with a $10\ \%$ edge-band weighting strategy (see Figure 7.8). The spatial weighting function $W(x, z)$ is decomposed into horizontal and vertical components $w_x(x)$ and $w_z(z)$, defined as piecewise constant:

$$\begin{aligned} w_x(x) &= \begin{cases} 3.0, & \text{if } x \le 0.1L \\ 0.5, & \text{otherwise} \end{cases} \\ w_z(z) &= \begin{cases} 3.0, & \text{if } z \le 0.1H \text{ or } z \ge 0.9H \\ 0.5, & \text{otherwise} \end{cases} \end{aligned} \tag{7.7}$$

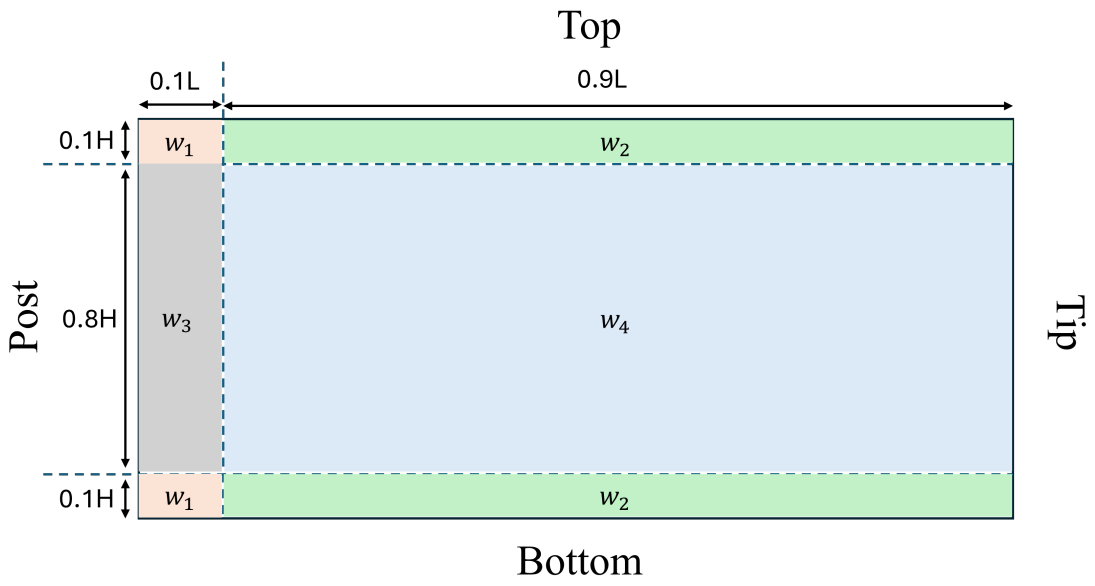


Figure 7.8: Location Weight Map.

where L and H represent the total width and height of the sign panel, respectively. The coordinate system is defined with its origin $(0, 0)$ at the bottom-left corner of the panel. As illustrated in the weighting map (Figure 7.8), this formulation generates a multi-level weight matrix across the panel, producing four distinct weight combinations (w_1 through w_4):

- **High-Stress Zones** $(w_1 = 3 \times 3 = 9)$: The intersections of the support edge and the top/bottom frames receive an extreme penalty to reflect their critical role in resisting the bending moment.
- **Transition Zones** $(w_2 = w_3 = 0.5 \times 3 = 1.5)$: Areas affected by either high longitudinal or vertical stress.
- **Low-Stress Zone** $(w_4 = 0.5 \times 0.5 = 0.25)$: Area in the middle of the panel.

To ensure the adjusted corrosion index is bounded within $[0, 1]$, normalization factors $\bar{W}_x$ and $\bar{W}_z$ are calculated based on the weighted average of the panel dimensions:

$$\begin{aligned} \bar{W}_x &= (0.1 \times 3.0) + (0.9 \times 0.5) = 0.75 \\ \bar{W}_z &= (0.2 \times 3.0) + (0.8 \times 0.5) = 1.0 \end{aligned} \tag{7.8}$$

The final location-weighted corrosion index CI_L is computed as the sum of the weighted corroded areas divided by the weighted total area:

$$CI_L = \frac{\sum_i^{\text{corr}} \left(A_i w_x(x_i) w_z(z_i)\right)}{A_{\text{total}} \left(\bar{W}_x \bar{W}_z\right)} = \frac{\sum_i^{\text{corr}} \left(A_i w_x(x_i) w_z(z_i)\right)}{0.75 A_{\text{total}}} \tag{7.9}$$

By introducing these location-based weights, the visual model effectively mimics the model using stress-weighted corrosion index. As shown in Figure 7.9, this adjustment tightens the data dispersion, increasing the R^2 to 0.71. This performance aligns with the stress-weighted model ($R^2 = 0.70$) in Figure 7.6. The improvement in R^2 arises not from a change in the logistic curve shape, but from a more physically meaningful redistribution of data points along the CI axis. Table 7.3 summarizes the fitted parameters and performance for the five-parameter logistic (5PL) regression curves associated with both models.

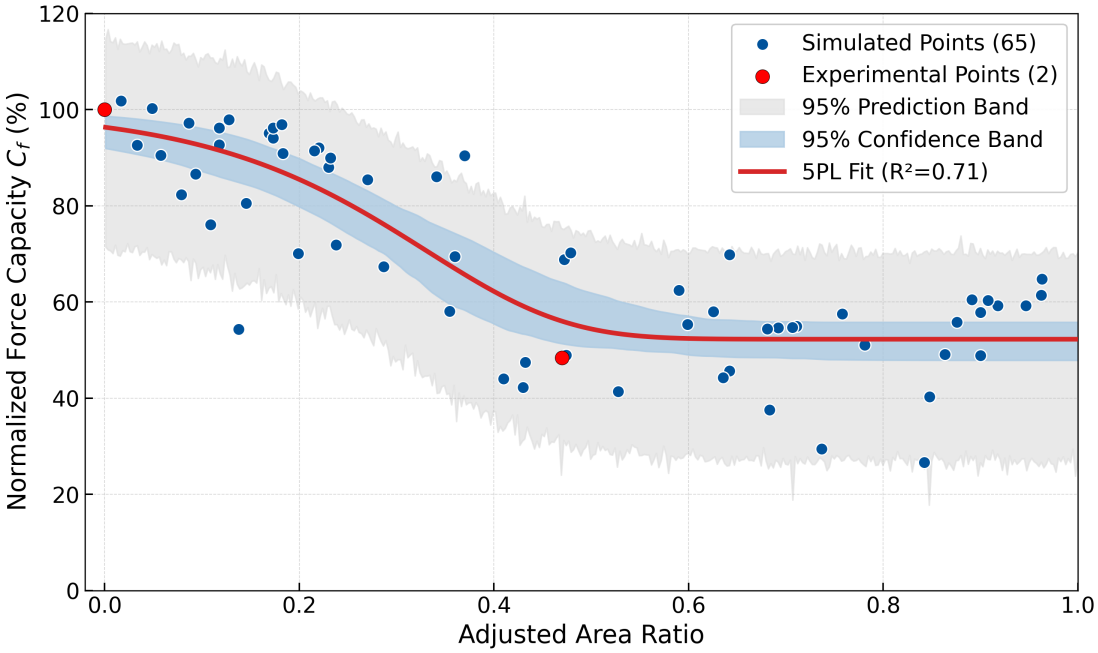


Figure 7.9: Location-Weighted Area Ratio Model.

Table 7.3: Fitted parameters for the five parameters logistic regression for two models

Parameter	Conventional Area Ratio	Location-Weighted
U	100	100
L	52.37	52.25
k	7.69	7.63
$\mathrm{CI}_{\mathrm{loc}}$	0.62	0.63
ν	10	10
R^2	0.61	0.71

7.8 DISCUSSION AND PRACTICAL DEPLOYMENT

The established relationship between the Corrosion Index (CI) and normalized force capacity provides a promising foundation for data-driven assessment of corrosion-induced strength loss in over-

head sign structures. The logistic model captures the observed degradation trend, the initial stable capacity at low CI levels, the rapid degradation phase, and a plateau at higher CI.

For Caltrans SM&I deployment, the traditional approach relies on an elastic, corrosion-free FE model to generate a stress map. While this provides a conservative lower-bound estimate of the stress distribution, requiring an FE simulation for every inspection is computationally expensive and impractical for routine field use. By introducing the Refined Location-Weighted Corrosion Model (CI_{L}), we have effectively bypassed the need for case-by-case numerical modeling. The newly proposed operational workflow for field engineers is streamlined as follows:

1. **Photo capture and CV-based corrosion detection**: Collect photos and detect corroded locations with the CV-based algorithm.
2. **Apply Spatial Weighting Matrix**: Overlay the standardized 10% edge-band penalty matrix onto the visual data to compute the location-weighted area of the damage, eliminating the need for an FE simulation.
3. **Estimate Strength Loss**: Using results from (1) & (2), calculate CI and estimate corresponding strength loss via the logistic relationship (Eq. 7.5) and Figure 7.9.

The complete operational workflow from field photograph acquisition through CV-based corrosion detection to CI_L calculation and capacity estimation is provided in Appendix H. From FEA-based stress-weighted model, through the basic area ratio model and ultimately to the proposed location-weighted model, the evolution highlights a critical finding in structural assessment: treating corrosion as a purely global scalar metric fundamentally ignores the mechanical reality of the structure. As previously demonstrated, two corrosion patterns with identical total areas can lead to drastically different capacity losses depending on their proximity to high-stress regions (e.g., the post-beam interface).

By simplifying those complex stress maps (Figure 7.4) into a standardized, grid-based penalty matrix (Figure 7.8), the CI_{L} model achieves the best of both worlds. It retains the mechanical rigor and high predictive accuracy of the Stress-Weighted approach while maintaining the operational simplicity of a purely visual, geometry-based assessment. Although some minor variance remains, this refined metric successfully bridges the gap between basic visual inspection and advanced structural mechanics, proving highly effective for field applications.

Looking ahead, the CI–capacity curve developed in this chapter may serve as a foundational tool for large-scale prioritization of structural inspections. Combined with UAV imaging and CV pipelines for automated CI estimation, it enables rapid and consistent screening of sign structures across the transportation network. Furthermore, by coupling the force capacity estimates with probabilistic deterioration models, Caltrans SM&I could adopt predictive maintenance strategies that are both cost-effective and risk-informed. Ultimately, this approach links visual condition assessment to a representative lower-bound estimate of structural performance, providing a simple yet conservative tool to support data-driven decision-making in sign structure management.

8 Conclusion and Future Extensions

This report presents a comprehensive framework for evaluating the structural integrity of overhead box beam sign structures, with a particular focus on the effects of corrosion. By integrating field testing, FE modeling, laboratory testing, and data-driven analysis, this study illustrates critical insights into the mechanical behavior of corroded and uncorroded structures under specific loading conditions, and provides a prediction curve for estimating the residual strength under different levels of corrosion.

8.1 CONCLUDING REMARKS

Key contributions of this research include:

1. **FE Model Development and Updating**

   A comprehensive FE model of the sign structure was developed to simulate structural responses under realistic loading and boundary conditions. The model was first iteratively optimized by genetic algorithms and then refined based on the analysis results from the laboratory testing. This modeling process ensured that the modal characteristics of the numerical model aligned with those obtained from field tests, improving the accuracy of subsequent assessment of structural response, and ensuring that the numerical simulations accurately represented the physical behavior of the structure.

2. **Image-Based Corrosion Detection and Model Integration**

   High-resolution images collected on site are fed to a hybrid *ResNet + U-Net* network for pixel-level corrosion segmentation, building upon recent advances in CV-based corrosion detection. The resulting masks are converted into geometric primitives that incorporated into the FE model through Abaqus Python API, enabling automated geometry updates reflecting actual corrosion patterns.

3. **Experimental Testing and Structural Performance Evaluation**

   Laboratory testing was conducted at the PEER Laboratory to assess the real-world behavior of both corroded and uncorroded sign structures. The tests involved force-displacement analysis, strain gauge measurements, and energy dissipation evaluations, allowing for a direct comparison of the effect of corrosion on different aspects of the structural response.

4. **Comparison of Corroded and Uncorroded Specimens**

   Experimental results highlighted the significant impact of corrosion on the structural response. Several specific observations are as follows:

   - Corrosion did not have a major effect on the IP and OOP stiffness and the natural frequencies.
   - For loading conditions W1–W3 and S0–S2 (Table 4.5), the corroded specimen was nearly elastic. This result is significant because S2 represents $4.4\times$ Self weight combined with 300-year wind.
   - Corrosion had a major effect (50%) on the strength in the push direction, while it had a minor effect on strength in pull direction.
   - Corrosion had a major effect on the force displacement relationship and the dissipated energy. The energy dissipated by the corrosion-free specimen is much larger than the energy dissipated by the corroded specimen.
   - Corrosion changed the damage and failure pattern. Damage in the corroded specimen was in the form of buckling of the corroded ribbed sheet metal and the corresponding weld pops with consequent twisting of the box beam due to the stiffness difference of the intact top ribbed sheet metal and the damaged bottom ribbed sheet metal. Fracture at the top of the post was observed after this damage occurred at the last two cycles.
   - Damage in the corrosion-free specimen migrated completely to the top of the post and the connection region, with both the top and bottom ribbed sheet metal remaining intact without minor damage throughout the entire testing.
   - Using terminology associated with buildings, the design of the existing single post sign structures from 1970s exhibit a strong-beam/weak-column behavior.

5. **Data-driven corrosion simulation and response analysis**.

   A unified, automated simulation framework was developed to systematically evaluate how diverse corrosion scenarios affect structural performance. Using a grid-based probabilistic sampling approach informed by field corrosion pattern, a dataset of synthetic corrosion patterns was generated and applied to the FE model. To quantify corrosion severity, a stress-weighted Corrosion Index (CI) was formulated, capturing both the extent and mechanical significance of damage. For each scenario, the total vertical reaction force across all fixed-base nodes was extracted as a global indicator of force capacity. A nonlinear regression model revealed a strong inverse relationship between CI and normalized force capacity ($R^2 = 0.70$), indicating that CI serves as an effective predictor of corrosion-induced strength degradation.

6. **Development of a Simplified, Field-Ready Capacity Assessment Tool.**

   Recognizing the computational barriers of structure-specific FEA for routine inspections, a Simplified Location-Weighted Corrosion Index (CI_{L}) was established. By distilling complex stress maps into a standardized 10% edge-band weighting grid, the model effectively translates structural mechanics into a practical visual assessment tool. The refined corrosion

index achieved a significant performance leap over the conventional purely area ratio (R^2 increased from 0.61 to 0.71), align with the performance using stress-weight map. This frame work empowers field engineers to estimate residual structural capacity directly from visual data without the need for numerical modeling, bridging the gap between rigorous mechanics and rapid field inspection.

8.2 LIMITATIONS AND FUTURE DIRECTIONS

8.2.1 Limitations of Abaqus workflow

Although Abaqus has robust nonlinear analysis capabilities, two major limitations were encountered in this study when modeling cyclic responses of corroded and corrosion-free specimens:

1. **No pinching effect in Abaqus built-in materials**: As discussed in this report, several articles (Mousavi et al., 2014; Wan et al., 2001) had shown that Abaqus does not provide any constitutive models to reproduce the pinching effect seen in testing results (Figure 6.31a). Consequently, a **User-Defined Material (UMAT)** subroutine in Abaqus was required to capture this effect.
2. **Cyclic analysis feasible only for corrosion-free model**: In pushover analysis, the corroded models aborted at a displacement about 3 inches (Figure 7.1a) due to severe damage and loss of convergence to match the testing results. Therefore, cyclic simulations presented in this report were limited to the corrosion-free specimen.

Based on the findings of this study, several areas offer promising directions for future research and development. These are summarized in the next sections.

8.2.2 Cyclic Loading Analysis Using a User-Defined Material

The experiments employed cyclic displacement loading, accordingly, a cyclic simulation is required to reproduce the hysteresis loops and energy dissipation. We implemented a user-defined hysteretic material based on a Clough-type model to capture the cyclic response, especially the pinching effect.

Clough Model and Abaqus Subroutine Implementation

The Clough model (Clough, 1966) is a bilinear hysteretic model with stiffness degradation during reloading, as illustrated in Figure 8.1. The model uses the rule that during reloading in a direction, the path follows either the maximum strain previously experienced in that direction, or the yield point if yielding has not occurred.

Abaqus UMAT is primarily written in Fortran, which is widely implemented in FEA for defining complex material behaviors. In this study, we adopted the user-defined hysteretic material

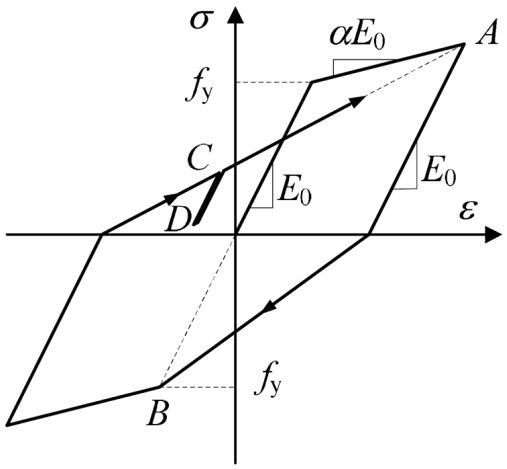


Figure 8.1: Clough Model Figure from (Qu and Pan, 2015)

subroutine provided as an open-source Fortran code on the PQ-Fiber website (Qu and Pan, 2015). This implementation, which was also employed in (Qu et al., 2017; Qu and Ye, 2011), is capable of reproducing the pinching effect and stiffness degradation observed in reinforced concrete members under cyclic loading. In our work, the subroutine was utilized to update the elastic modulus and tangent stiffness for the shell elements. This method has some limitations to be addressed in future studies. For completeness, the current implementation details are provided in the Appendix I.

Preliminary Simulation Results and Remarks

Through the UMAT setting in Abaqus, this model was incorporated into the material setting of the post shell elements used in our simulations. Figure 8.2 shows the experimental cyclic force-displacement response (blue) with Abaqus simulation using the UMAT subroutine (red). The simulated envelope tracked the force–displacement response reasonably well, matching the peak force and initial stiffness and capturing the overall backbone curve. Even this model cannot capture perfect pinching effect, the hysteresis loops exhibit a clear waist-narrowing near the origin, showing the stiffness deduction during reloading.

8.2.3 Out-of-Plane (OOP) Analysis and Scaled Wind Tunnel Testing

The experiments conducted in this study primarily focused on vertical load assessments, with wind loading applied in the Out-of-Plane (OOP) direction using pressures corresponding to peak wind speeds from standard code equations. Future research could include comprehensive simulations of wind-induced lateral forces and the corresponding structural response. To achieve this, scaled wind tunnel testing could be employed, allowing for controlled experimental conditions that replicate real-world wind effects on overhead sign structures. Moreover, coupling these wind-induced loading scenarios with corrosion progression models could enhance the predictive capability of the FE framework, providing a more holistic understanding of long-term structural degradation.

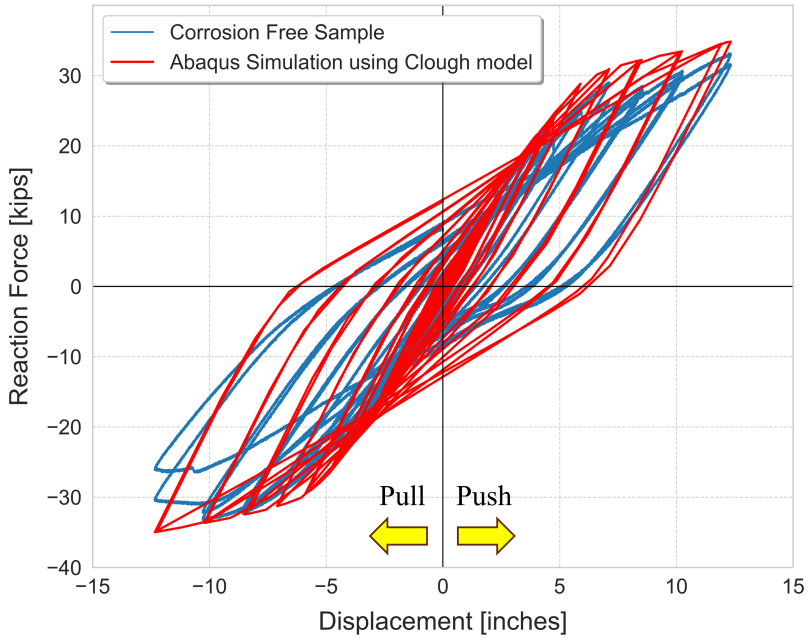


Figure 8.2: Cyclic Loading Simulation using UMAT

8.2.4 Deep Reinforcement Learning for Real-Time FE Updating

Although genetic algorithms were effective in this study, they are computationally intensive for larger models. Deep Reinforcement Learning (DRL) offers a more adaptive approach by treating the model updating as a learning problem. Unlike traditional optimization methods, DRL agents can learn optimal policies through interactions with the simulation environment, gradually improving predictions with each iteration. Methods such as A2C and A3C have been successfully applied to FE updating (Pang et al., 2023; Liu et al., 2024). In the future, integrating DRL with real-time sensor data can enable live updates to digital twins and support structural monitoring during service.

Future research could focus on integrating DRL into our FE model updating process, treating it as a continuous optimization problem. In this context, the DRL agent (A2C or A3C) interacts directly with the Abaqus simulation environment, learning optimal strategies for model calibration through iterative interactions. The policy network governs the adjustment of critical parameters within the FE model, not only limited to component thickness but also extending to material properties such as yield stress, strain hardening coefficients, and plastic deformation characteristics. For the reward function, beyond simply aligning natural frequencies with experimental data, the rewards cam ne extended to include key structural performance indicators observed during our lab testing, such as:

- **Force Capacity:** The maximum load capability of the structure under cyclic or static loading.
- **Initial Stiffness:** The slope of the initial linear portion of the force-displacement curve, reflecting the structural rigidity before yielding.

- **Energy Dissipation**: Representing the capacity of the structure to absorb and dissipate energy during cyclic loading.

Through trial-and-error learning, the agent can explore various parameter configurations, evaluating their impact on the reward indicators. This adaptive learning approach allows the DRL agent to efficiently converge toward parameter sets that best align the FE simulation outputs with experimental results.

8.2.5 Stochastic Corrosion Evolution and Predictive Modeling

To support long-term maintenance planning and predict the lifespan of corroded structures, future research should develop advanced methods for corrosion pattern simulation and corrosion prediction. In Chapter 7, a Gaussian-based approach was employed to generate realistic corrosion patterns, reflecting spatial variability and localized degradation. This methodology effectively captures random distribution characteristics and provides a robust foundation for FE model updating. However, further exploration of more sophisticated techniques is necessary to predict the evolution of corrosion under variable environmental conditions.

Stochastic methods such as Cellular Automata (CA) (Wolfram, 1983) and Markov Chains (Melchers, 2008) provide powerful methods by simulating random deterioration patterns over time. Future research could also incorporate machine learning-based prediction models to forecast corrosion expansion. Techniques such as Long Short-Term Memory (LSTM) networks (Jiang et al., 2022) have shown promise in prediction of corrosion. These methods can be trained on historical corrosion data and enable the prediction of when and where corrosion is likely to intensify. Integrating these models with the CI-Capacity function (Figure 7.6) would allow for predictive maintenance scheduling, estimating when the Corrosion Index (CI) is likely to reach critical thresholds that require inspection, repair, or replacement, providing early warning for maintenance needs.

8.2.6 Field Deployment and Generalization

The current framework, spanning CV-based detection, FE model updating, and performance estimation, can be applied to other steel structures beyond sign supports. Deploying this system across lighting poles, traffic gantries, or bridge trusses can validate its flexibility and help identify structure-specific improvements. Wider deployment across California could also support data-driven asset management at scale.

REFERENCES

American Institute of Steel Construction (2022). "Specification for structural steel buildings (ANSI/AISC 360-22)." *Report No. ANSI/AISC 360-22*, American Institute of Steel Construction, Chicago, Illinois, <https://www.aisc.org/>.

ASTM International (2024). "Standard test methods for tension testing of metallic materials." *Report No. E8/E8M-24*, ASTM International.

Bao, Y., Chen, Z., Wei, S., Xu, Y., Tang, Z., and Li, H. (2019). "The state of the art of data science and engineering in structural health monitoring." *Engineering*, 5(2), 234–242.

California Department of Transportation (Caltrans) (2024a). "Bridge and tunnel safety inspection resources." *Report No. Budget Change Proposal 7195*, California Department of Finance, <https://bcp.dof.ca.gov/2425/FY2425_ORG2660_BCP7195.pdf>.

California Department of Transportation (Caltrans) (2024b). *Structure Maintenance & Investigations* (SM&I) *Division: Overhead Sign Structures*, <https://dot.ca.gov/programs/maintenance/structure-maintenance-investigations>.

Casas, E., Ramos, L., Romero, C., and Rivas-Echeverría, F. (2024). "A comparative study of YOLO v5 and YOLO v8 for corrosion segmentation tasks in metal surfaces." *Array*, 22, 100351.

Cha, Y.-J., Choi, W., and Büyüköztürk, O. (2017). "Deep learning-based crack damage detection using convolutional neural networks." *Computer-Aided Civil and Infrastructure Engineering*, 32(5), 361–378.

Cheng, H., Chai, W., Hu, J., Ruan, W., Shi, M., Kim, H., Cao, Y., and Narazaki, Y. (2024). "Random bridge generator as a platform for developing computer vision-based structural inspection algorithms." *Journal of Infrastructure Intelligence and Resilience*, 3(2), 100098.

Choi, E. (2023). "Optimization algorithms for performance-based design and structural model updating." Ph.D. thesis, University of California, Berkeley, CA.

Chung, T. J. (1978). "Finite element analysis in fluid dynamics." *NASA STI/Recon Technical Report A*, 78, 44102.

Clough, R. W. (1966). "Effect of stiffness degradation on earthquake ductility requirements." *Report UCB/SESM-1966/16*, University of California, Berkeley, <https://escholarship.org/uc/item/21f175hg>.

Das, A., Dorafshan, S., and Kaabouch, N. (2024). "Autonomous image-based corrosion detection in steel structures using deep learning." *Sensors*, 24(11).

Dassault Systèmes (2006). *Abaqus Version 6.6 Documentation*. Section 3.5 Beam elements.

Dassault Systèmes (2006). *Abaqus Version 6.6 Documentation*. Section 3.6 Shell elements.

Federal Emergency Management Agency (2007). "Interim testing protocols for determining the seismic performance characteristics of structural and nonstructural components." *Report No. FEMA 461*, Federal Emergency Management Agency, Washington, D.C. (June).

Harris, H. G. and Sabnis, G. (1999). *Structural modeling and experimental techniques*. CRC press.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). "Deep residual learning for image recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jiang, X., Yan, Y., and Su, Y. (2022). "Data-driven pitting evolution prediction for corrosion-resistant alloys by time-series analysis." *npj Materials Degradation*, 6(1), 92.

Khayatazad, M., De Pue, L., and De Waele, W. (2020). "Detection of corrosion on steel structures using automated image processing." *Developments in the Built Environment*, 3, 100022.

Kreuzer, S. M., Briant, P. L., and Ochoa, J. A. (2021). "Establishing the biofidelity of a multi-physics finite element model of the human heart." *Cardiovascular Engineering and Technology*, 12, 387–397.

Lee, H., Günay, S., and Mosalam, K. M. (2015). "Comparison of the seismic response of reinforced auger pressure grout and concrete columns." *Engineering Structures*, 87, 139–152.

Liu, G., Cong, J., Wang, P., Du, S., Wang, L., and Chen, R. (2022). "Study on vertical vibration and transmission characteristics of railway ballast using impact hammer test." *Construction and Building Materials*, 316, 125898.

Liu, H., Yang, Z., Zhou, T., Wang, L., and Chen, Z. (2024). "Study on updating finite element model of steel truss structure based on knowledge-enhanced deep reinforcement learning." *Engineering Structures*, 316, 118576.

Lizee, E., Robin, S., Song, E., Bertholon, N., Coz, J.-Y. L., Besnault, B., and Lavaste, F. (1998). "Development of a 3D finite element model of the human body." *SAE Transactions*, 107, 2760–2782.

Luo, L. and Zhao, M. (2011). "The secondary development of ABAQUS by using Python and the application of the advanced GA (proceedings of the 2011 international conference on physics science and technology, ICPST 2011)." *Physics Procedia*, 22, 68–73.

Marcum, D. L. and Weatherill, N. P. (1995). "Aerospace applications of solution adaptive finite element analysis." *Computer Aided Geometric Design*, 12(7), 709–731.

Melchers, R. E. (2008). "Probabilistic modelling of steel corrosion in marine environments." *Structure and Infrastructure Engineering*, 4(4), 311–323.

Mollica, F. and Ambrosio, L. (2009). "The finite element method for the design of biomedical devices." *Biomaterials in Hand Surgery*, 31–45.

Mosalam, K. (2024). *CE222: Finite Element Methods*. University of California, Berkeley.

Mosalam, K. M. and Gao, Y. (2024). *Artificial intelligence in vision-based structural health monitoring*. Springer.

Mousavi, S. A., Zahrai, S. M., and Bahrami-Rad, A. (2014). "Quasi-static cyclic tests on super-lightweight eps concrete shear walls." *Engineering Structures*, 65, 62–75.

Moustafa, M. A. and Mosalam, K. M. (2015). "Seismic response of bent caps in as-built and retrofitted reinforced concrete box-girder bridges." *Engineering Structures*, 98, 59–73.

Narazaki, Y., Hoskere, V., Hoang, T. A., Fujino, Y., Sakurai, A., and Spencer Jr., B. F. (2020). "Vision-based automated bridge component recognition with high-level scene consistency." *Computer-Aided Civil and Infrastructure Engineering*, 35(5), 465–482.

Oregui, M., Li, Z., and Dollevoet, R. (2015). "Identification of characteristic frequencies of damaged railway tracks using field hammer test measurements." *Mechanical Systems and Signal Processing*, 54–55, 224–242.

Pang, I. K., Gao, Y., and Mosalam, K. M. (2023). "Deep reinforcement learning for structural model updating using transfer learning mechanism." *Computing in Civil Engineering 2023*, 364–371, <https://ascelibrary.org/doi/abs/10.1061/9780784485231.044>.

Petricca, L., Moss, T., Figueroa, G., Broen, S., and Horten, O. (2016). "Corrosion detection using A.I.: A comparison of standard computer vision techniques and deep learning model." *Proceedings of the Sixth International Conference on Computer Science, Engineering and Information Technology*, AIRCC Publishing Corporation, Chennai, India, 99.

Qu, Z. and Pan, P. (2015). *PQ-Fiber: Open-source material models for reinforced concrete and steel*. PQ-Fiber Project, <https://www.qu-zhe.net/pqfiber.htm>.

Qu, Z., Xie, J., Wang, T., and Kishiki, S. (2017). "Cyclic loading test of double K-braced reinforced concrete frame subassemblies with buckling restrained braces." *Engineering Structures*, 139, 1–14.

Qu, Z. and Ye, L. (2011). "Strength deterioration model based on effective hysteretic energy dissipation for rc members under cyclic loading." *Engineering Mechanics*, 28(6), 45–51 (in Chinese).

Ronneberger, O., Fischer, P., and Brox, T. (2015). "U-Net: Convolutional networks for biomedical image segmentation." *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds., Cham, Springer International Publishing, 234–241.

Siddique, N., Paheding, S., Elkin, C. P., and Devabhaktuni, V. (2021). "U-Net and its variants for medical image segmentation: A review of theory and applications." *IEEE Access*, 9, 82031–82057.

Spencer, B. F., Hoskere, V., and Narazaki, Y. (2019). "Advances in computer vision-based civil infrastructure inspection and monitoring." *Engineering*, 5(2), 199–222.

Srivastava, A., Ji, G., and Singh, R. K. (2021). "Application of deep-learning architecture for image analysis based corrosion detection." *2021 Smart Technologies, Communication and Robotics* (STCR), 1–5.

Tang, W. and Gu, Y. (2002). "Advances of genetic algorithms in structural optimization." *Advances in Mechanics*, 32, 26–40.

Tsai, A. Y., Günay, S., Hwang, M., Zhai, P., Li, C., El Ghaoui, L., and Mosalam, K. M. (2020). "Text analytics for resilience-enabled extreme events reconnaissance." *arXiv preprint arXiv:2011.13087*.

Wan, S., Loh, C.-H., and Peng, S.-Y. (2001). "Experimental and theoretical study on softening and pinching effects of bridge column." *Soil Dynamics and Earthquake Engineering*, 21(1), 75–81.

Wolfram, S. (1983). "Statistical mechanics of cellular automata." *Reviews of Modern Physics*, 55(3), 601–644.

Appendix A: Abaqus API

A.1 MODEL CREATION

Finite Element Modeling (FEM) is a fundamental tool in structural analysis and material simulation. This paper introduces a Python-based script for automated model generation in Abaqus. The script streamlines the creation of complex parametric models, including geometry definition, material assignment, meshing, and boundary condition setup. The workflow is implemented using Python scripting in Abaqus, enabling parametric control over the model's properties. Key components of the script are described below.

A.1.1 Parameter Definition

The model is parameterized to allow flexible adjustment of geometrical and material properties. The main parameters include:

- Thickness of structural elements.
- Elastic modulus and Poisson's ratio for uncorroded regions.
- Corrosion ratio to modify material properties in affected areas.

A.1.2 Geometry Creation

The geometry is defined using Abaqus' `ConstrainedSketch` and `BaseShellExtrude` methods. For example:

```
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
mdb.models['Model-1'].sketches['__profile__'].Line(point1=(13.75, 19.25),
    point2=(22.25, 26.75))
mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Part-1', type=
    DEFORMABLE_BODY)
mdb.models['Model-1'].parts['Part-1'].BaseShellExtrude(depth=276.0, sketch=mdb
    .models['Model-1'].sketches['__profile__'])
```

This ensures precise generation of structural elements, such as columns and beams.

A.1.3 Material Assignment

Materials are assigned to regions using the following commands:

```
mdb.models['Model-1'].Material(name='Material-1')
mdb.models['Model-1'].materials['Material-1'].Density(table=((0.0007487, ), ))
mdb.models['Model-1'].materials['Material-1'].Elastic(table=((210000, 0.3), ))
```

It is noted that regions are grouped into sets for easier management and sections are assigned using the method of `HomogeneousShellSection`.

A.1.4 Meshing

The mesh is generated with a uniform seed size:

```
mdb.models['Model-1'].parts['Part-1'].seedPart(size=5.0, deviationFactor=0.1,
    minSizeFactor=0.1)
```

This provides consistent mesh quality across the model.

A.1.5 Boundary Conditions and Loading

Boundary conditions are applied to simulate fixed-end supports, and a frequency step is created for modal analysis:

```
mdb.models['Model-1'].DisplacementBC(name='BC-1', createStepName='Step-1',
    region=mdb.models['Model-1'].rootAssembly.sets['Set-1'], u1=0.0, u2=0.0, u3
        =0.0)
mdb.models['Model-1'].FrequencyStep(name='Step-1', previous='Initial',
    numEigen=5)
```

A.2 CORROSION ASSIGNMENT

Listing A.1: Python code for Corrosion Assignment in Abaqus

```
def read_csv_to_numpy(filename):
with open(filename, mode='r') as file:
    csv_reader = csv.reader(file)
    data = list(csv_reader)
    numpy_array = np.array(data, dtype=float)
return numpy_array

df_coord = read_csv_to_numpy(csv_dir)
yMin = 3.25; yMax = 5.83

for row in range(df_coord.shape[0]):
    xMin = df_coord[row,0]
```

```
    zMin = df_coord[row,3]
    xMax = df_coord[row,2]
    zMax = df_coord[row,1]
    print(xMin, yMin, zMin, xMax, yMax, zMax)
    allElements = mdb.models['Model-1'].parts['Part-1'].elements
    selectedElements1 = allElements.getByBoundingBox(xMin-2, yMin, zMin-2, xMax
        +2, yMax, zMax+2)
    name_row = "CorrodedEle" + str(row)
    mdb.models['Model-1'].parts['Part-1'].Set(name=name_row, elements=
        selectedElements1)
    mdb.models['Model-1'].parts['Part-1'].SectionAssignment(offset=0.0,
       offsetField='', offsetType=MIDDLE_SURFACE, region=
       mdb.models['Model-1'].parts['Part-1'].sets[name_row], sectionName=
    'RibSheet_Corroded', thicknessAssignment=FROM_SECTION)
```

Appendix B: Genetic Algorithm

B.1 INTRODUCTION

This document explains the implementation of a genetic algorithm (GA) integrated with ABAQUS to optimize a set of parameters for minimizing the Mean Square Error (MSE) between computed and target frequencies. The code combines Python scripting for GA and ABAQUS's FE analysis capabilities.

B.2 GENETIC ALGORITHM SETUP

The GA is initialized with the following parameters:

```
Gen_num = 30    # Number of generations
popul = 7       # Population size
survive_rate = 0.4 # Survival rate
offspring_rate = 0.4 # Offspring rate
CO_rate = 0.7   # Crossover rate
RM_rate = 0.05  # Mutation rate
rand_bound = 0.005 # Random search bound
```

The objective function to minimize is defined as the square root of the sum of squared differences between the computed frequencies (`T1, T2`) and the target frequencies (`GT`):

```
def mse(T1, T2, GT):
    return ((T1 - GT[0])**2 + (T2 - GT[1])**2)**(0.5)
```

B.3 ABAQUS MODEL UPDATE AND ANALYSIS

The function `abaqus_run()` updates the FE model parameters and submits an ABAQUS job:

```
def abaqus_run(update_model_parameters):
    thickness_of_columnA = update_model_parameters[0]
    uncorroded_Elasticity = update_model_parameters[4]
    corroded_Elasticity = update_model_parameters[6] * uncorroded_Elasticity

```

```
    mdb.models['Model-1'].sections['ColA'].setValues(
        thickness=thickness_of_columnA)
    mdb.models['Model-1'].materials['Material-Corroded'].elastic.setValues(
        table=((corroded_Elasticity, uncorroded_Possion), ))
    mdb.Job(name='Job-1001', model='Model-1').submit()
    mdb.jobs['Job-1001'].waitForCompletion()
```

The computed frequencies are extracted from the ABAQUS report:

```
def rpt_read():
    with open('abaqus.rpt', 'r') as testfile:
        for i, line in enumerate(testfile):
            continue
        num_line = i
    with open('abaqus.rpt', 'r') as testfile:
        for j, line in enumerate(testfile):
            if j == num_line-8:
                T1 = float(line.split()[1])
            if j == num_line-7:
                T2 = float(line.split()[1])
    return T1, T2
```

B.4 CROSSOVER AND MUTATION

The function `CrossOver()` performs genetic operations to generate offspring:

```
def CrossOver(Gen, N, CO_rate, RM_rate, rand_bound, params_ub, params_lb):
    offsprings = []
    for i in range(N):
        parent1, parent2 = np.random.randint(0, len(Gen), 2)
        p = np.random.uniform(0, 1)
        if p <= RM_rate:
            candidate = Gen[parent1] + np.random.uniform(-rand_bound, rand_bound,
                len(Gen[parent1])) * Gen[parent1]
        else:
            multip = [random.uniform(0, 1) if i < len(Gen[parent1]) * CO_rate
                else random.randint(0, 1) for i in range(len(Gen[parent1]))]
            random.shuffle(multip)
            candidate = Gen[parent1] + np.multiply(multip, Gen[parent2] - Gen[
                parent1])
        offsprings.append(candidate)
    return offsprings
```

B.5 OPTIMIZATION WORKFLOW

The optimization iteratively evaluates the objective function, sorts the population, and generates new offspring:

```
for N in range(Gen_num-1):
   del Gen[Nsurvive:popul]
   offsprings = CrossOver(Gen, int(round(popul * offspring_rate)), CO_rate,
      RM_rate, rand_bound, params_ub, params_lb)
   for k in range(popul - Nsurvive):
      abaqus_run(Gen[k])
      T1, T2 = rpt_read()
      error = mse(T1, T2, GT)
      Gen[k] = np.append(Gen[k], error)
   Gen.sort(key=error_sort)
```

B.6 RESULTS

The best parameters and their corresponding frequencies are saved for further analysis:

```
params_save_overwrite = open("params_save.txt", 'w')
for p in Gen[0]:
   params_save_overwrite.writelines(str(p) + "\n")
params_save_overwrite.close()
```

Appendix C: CV Algorithm

```
class CorrodedCoordinateGeneration:
   def __init__(self, raw_img_path, mask_img_path,num_col,num_row):
      self.raw_img_path = raw_img_path
      self.mask_img_path = mask_img_path
      self.num_col = num_col
      self.num_row = num_row
      self.points = []
      self.raw_img = None
      self.mask_img = None
      self.selected_area = None

   def load_img(self):
      self.raw_img = cv2.imread(self.raw_img_path)
      self.mask_img = cv2.imread(self.mask_img_path, cv2.IMREAD_GRAYSCALE)
      if self.raw_img is None or self.mask_img is None:
         raise FileNotFoundError("One or both of the images could not be
            loaded.")
      self.yr_max = self.raw_img.shape[0]; self.xr_max = self.raw_img.shape[1]
      self.y_max = self.mask_img.shape[0]; self.x_max = self.mask_img.shape[1]

   def select_img_4points(self):
      '''
      Manually click four corners of the Sign Structure in order to calculate
      the area of the Sign
      '''
      self.raw_img = cv2.cvtColor(self.raw_img, cv2.COLOR_BGR2RGB) # cv2 uses
         BGR while matplotlib uses RGB
      fig, ax = plt.subplots()
      ax.imshow(self.raw_img)
      ax.set_title('Click to select 4 points')

      def onclick(event):
         if len(self.points) < 4:
            x, y = event.xdata, event.ydata
            self.points.append((x, y))
            ax.plot(x, y, 'ro')
            plt.draw()
            if len(self.points) == 4:
               fig.canvas.mpl_disconnect(cid)
               self.points = self.order_points(self.points)
               area = self.calculate_polygon_area(self.points)
```

```
                self.selected_area = area

    cid = fig.canvas.mpl_connect('button_press_event', onclick)
    plt.show()

def order_points(self, points):
    '''
    This function ensures the order of points is listed in Clockwise (CW)
        direction.
    '''
    points = np.array(points)
    rect = np.zeros((4, 2), dtype='float32')
    s = points.sum(axis=1)
    rect[0] = points[np.argmin(s)]
    rect[2] = points[np.argmax(s)]
    diff = np.diff(points, axis=1)
    rect[1] = points[np.argmin(diff)]
    rect[3] = points[np.argmax(diff)]
    return rect

def calculate_polygon_area(self, points):
    '''
    Calculate the Area of Polygon formed by the four nodes selected before.
    Using Shoelace Formula or Gauss's area formula.
    Area = 1/2*(sum(x_i*y_(i+1) - y_i*x_(i+1)) + x_n*y_1 - y_n*x_1)
    '''
    n = len(points)
    if n < 3:
        print("Error: At least 4 points should be clicked")
        return None
    area = 0
    for i in range(n):
        j = (i + 1) % n # j = 2,3,4,1 while i = 1,2,3,4
        area += points[i][0] * points[j][1] - points[j][0] * points[i][1]
    area = abs(area) / 2.0
    return area

def calculate_area_ratio_of_polygon_to_image(self):
    '''
    Calculate the ratio between the polygon area (The Sign Area) to the
        total image area
    '''
    if self.raw_img is not None and self.selected_area is not None:
        height, width, _ = self.raw_img.shape
        area_of_IMG = height * width
        area_of_sign = self.selected_area
        ratio_sign = area_of_sign / area_of_IMG
        print(f'Ratio Sign {ratio_sign}')
        return ratio_sign
    else:
        print("Error: Raw image or selected area not loaded")
        return None
```

```
91      def calculate_area_ratio_of_corroded_to_mask_img(self):
92          '''
93          Calculate the ratio between the corroded area (white pixels) to the
                total mask image area
94          '''
95          if self.mask_img is not None:
96              total_pixels = self.mask_img.shape[0] * self.mask_img.shape[1]
97              white_pixel_count = np.sum(self.mask_img == 255)
98              ratio_corroded = white_pixel_count / total_pixels
99              print(f'Ratio Corroded {ratio_corroded}')
100             return ratio_corroded
101         else:
102             print("Error: Mask image not loaded")
103             return None
104
105     def calculate_corroded_ratio(self):
106         '''
107         Calculate the ratio of corroded area to the sign area.
108         The ratio is used for generation of FE model later.
109         '''
110         sign_area_ratio = self.calculate_area_ratio_of_polygon_to_image()
111         corroded_area_ratio = self.calculate_area_ratio_of_corroded_to_mask_img
                ()
112         if sign_area_ratio is not None and corroded_area_ratio is not None:
113             corroded_to_sign_ratio = corroded_area_ratio / sign_area_ratio
114             print(f"Ratio of corroded area to sign area: {corroded_to_sign_ratio}
                    ")
115         else:
116             print("Error: Could not calculate the corroded to sign area ratio due
                    to missing data.")
117         self.corroded_to_sign_ratio = corroded_to_sign_ratio
118
119     def interpolate_points(self,p1,p2,num_points):
120         '''
121         Interpolate points between p1 and p2.
122
123         Parameters:
124         p1, p2 (array-like): Endpoints between which interpolation is to be done
                .
125         num_points (int): Number of points to interpolate.
126
127         Returns:
128         np.ndarray: Interpolated points.
129         '''
130         return np.linspace(p1,p2,num_points)
131
132     def generate_grid(self,points):
133         '''
134         Generate the location of grid points in Mask Images.
135
136         Parameters:
137         points (array-like): Four corner points of the selected area.
138
```

```
        Returns:
        np.ndarray: Grid points mapped to the mask image.
        '''
        top_edge = self.interpolate_points(points[0],points[1],self.num_col+1)
        bottom_edge = self.interpolate_points(points[3],points[2],self.num_col
            +1)
        grid_points = []
        for i in range(self.num_col + 1):
            left_col = self.interpolate_points(top_edge[i], bottom_edge[i], self.
                num_row + 1)
            grid_points.append(left_col)
        self.grid_points = np.array(grid_points) # Shape (31,11,2)

        # Map the grid from raw img to mask img
        self.points_mask = self.points.copy()
        self.points_mask[:,0] = self.points[:,0] * self.x_max / self.xr_max
        self.points_mask[:,1] = self.points[:,1] * self.y_max / self.yr_max

        self.grid_points_mask = self.grid_points.copy()
        self.grid_points_mask[:, :, 1] = self.grid_points[:, :, 1] * self.y_max
            / self.yr_max
        self.grid_points_mask[:, :, 0] = self.grid_points[:, :, 0] * self.x_max
            / self.xr_max
        return self.grid_points_mask

    def draw_grid_fig(self):
        '''
        Draw a figure with the grid overlaid on the mask image.
        '''
        mask_img_color = cv2.cvtColor(self.mask_img, cv2.COLOR_GRAY2BGR)
        polygon = plt.Polygon(self.points_mask, closed=True, fill=None,
            edgecolor='r')
        plt.gca().add_patch(polygon)

        # Draw the grid points on the mask image
        for i in range(self.num_col + 1):
            for j in range(self.num_row + 1):
                cv2.circle(mask_img_color, (int(self.grid_points_mask[i, j, 0]),
                                            int(self.grid_points_mask[i, j, 1])),
                                            radius=3, color=(0, 0, 255), thickness=-1)
        plt.imshow(cv2.cvtColor(mask_img_color, cv2.COLOR_BGR2RGB))
        plt.xlim(0, self.x_max)
        plt.ylim(self.y_max, 0)
        plt.gca().set_aspect('equal', adjustable='box')
        plt.show()

    def generate_box(self):
        '''
        Generate bounding boxes for the grid points in the mask image.

        Returns:
        np.ndarray: An array of bounding boxes with each box defined by [y1, y2,
             x1, x2].
```

```
186         '''
187         x_locs,y_locs,_ = self.grid_points_mask.shape
188         self.boxes = np.array([])
189         for yi in range(y_locs-1):
190             for xi in range(x_locs-1):
191                 point1 = self.grid_points_mask[xi,yi,:]
192                 point2 = self.grid_points_mask[xi+1,yi+1,:]
193                 box = np.array([point1[1],point2[1],point1[0],point2[0]])
194                 if yi == 0 and xi ==0:
195                     self.boxes = box
196                 else:
197                     self.boxes = np.vstack((self.boxes, box))
198         return self.boxes
199 
200     def detect_corrosion(self):
201         '''
202         Iterate every box in mask figure and detect the corrosion ratio.
203         If the ratio is larger than 50%, save the corresponding coordinates of
                points
204         into a cvs file for the Abaqus to excute.
205         '''
206         self.corroded_box = np.array([0,0,0,0])
207         self.corroded_ratio = []
208         self.corroded_point = np.array([0,0])
209         self.corroded_point2 = np.array([0,0])
210         mask_img_color = cv2.cvtColor(self.mask_img, cv2.COLOR_GRAY2BGR)
211 
212         for num_box in range(self.boxes.shape[0]):
213             box = self.boxes[num_box,:].astype(int)
214             test_box = self.mask_img[box[0]:box[1],box[2]:box[3]]
215             # [y_loc:y_loc + box_height, x_loc:x_loc + box_width]
216             white_pixel_count = np.sum(test_box == 255)
217             x,y = np.shape(test_box)
218             total_pixel_count = x * y
219             ratio = white_pixel_count/total_pixel_count
220 
221             if ratio >= 0.5:
222                 self.corroded_point = np.vstack((self.corroded_point,np.array([box
                        [2],box[0]])))
223                 self.corroded_point2 = np.vstack((self.corroded_point2,np.array([
                        box[3],box[1]])))
224                 cv2.circle(mask_img_color, (box[2], box[0]), radius=3, color=(0,
                        0, 255), thickness=-1)
225                 cv2.circle(mask_img_color, (box[3], box[1]), radius=3, color=(0,
                        255, 255), thickness=-1)
226             self.corroded_ratio.append(ratio)
227 
228         self.corroded_point = self.corroded_point[1:,:]
229         self.corroded_point2 = self.corroded_point2[1:,:]
230         df_point1 = pd.DataFrame(self.corroded_point, columns=['x1', 'y1'])
231         df_point2 = pd.DataFrame(self.corroded_point2, columns=['x2', 'y2'])
232         self.combined_df = pd.concat([df_point1, df_point2], axis=1)
233         # [x1, y1, x2, y2]
```

```
234         self.combined_df.to_csv("corroded_coordinates_mask.csv", index=False,
                header=False)
235 
236         plt.imshow(cv2.cvtColor(mask_img_color, cv2.COLOR_BGR2RGB))
237         plt.xlim(0, self.x_max)
238         plt.ylim(self.y_max, 0)
239         plt.gca().set_aspect('equal', adjustable='box')
240         plt.show()
241 
242     def abaqus_coord_transformation(self):
243         '''
244         Transform corroded points coordinates to Abaqus coordinate system.
245         '''
246         self.Abaqus_4pointCoord = np.array([
247             [22.25 , 276 ],
248             [205.125, 276 ],
249             [205.125, 213.75],
250             [22.25 , 213.75]
251         ], dtype=np.float32)
252         point1_abaqus_list = np.array([0,0])
253         point2_abaqus_list = np.array([0,0])
254         a = np.array(self.points_mask,dtype = np.float32)
255         M = cv2.getPerspectiveTransform(a, self.Abaqus_4pointCoord)
256 
257         for i in range(self.corroded_point.shape[0]):
258             point1 = np.array(self.corroded_point[i],dtype = np.float32)
259             point1_abaqus = cv2.perspectiveTransform(np.array([[point1]]), M).
                    reshape(-1)
260             point1_abaqus_list=np.vstack((point1_abaqus_list,point1_abaqus))
261 
262             point2 = np.array(self.corroded_point2[i],dtype = np.float32)
263             point2_abaqus = cv2.perspectiveTransform(np.array([[point2]]), M).
                    reshape(-1)
264             point2_abaqus_list=np.vstack((point2_abaqus_list,point2_abaqus))
265 
266         point1_abaqus_list = point1_abaqus_list[1:,:]
267         point2_abaqus_list = point2_abaqus_list[1:,:]
268 
269         df_point1_abaqus = pd.DataFrame(point1_abaqus_list)
270         df_point2_abaqus = pd.DataFrame(point2_abaqus_list)
271         self.combined_df_abaqus = pd.concat([df_point1_abaqus, df_point2_abaqus
                ], axis=1)
272         # [x1, y1, x2, y2]
273         self.combined_df_abaqus.to_csv("corroded_coordinates_abaqus.csv", index=
                False, header=None)
274 
275     def process(self):
276         self.load_img()
277         self.select_img_4points()
278         self.calculate_corroded_ratio()
279         self.generate_grid(self.points)
280         self.draw_grid_fig()
281         self.generate_box()
```

```
282         self.detect_corrosion()
283
284     def drawGridImage(self):
285         self.abaqus_coord_transformation()
286
287         top_edge = self.interpolate_points(self.Abaqus_4pointCoord[0,:],
288                                            self.Abaqus_4pointCoord[1,:], self.num_col+1)
289         bottom_edge = self.interpolate_points(self.Abaqus_4pointCoord[3,:],
290                                               self.Abaqus_4pointCoord[2,:], self.num_col
                                                  +1)
291
292         grid_points = []
293         for i in range(self.num_col + 1):
294             left_col = self.interpolate_points(top_edge[i], bottom_edge[i], self.
                    num_row + 1)
295             grid_points.append(left_col)
296         grid_points = np.array(grid_points) # Shape (31,11,2)
297
298         for coords in self.combined_df_abaqus.values:
299             x_min, y_max, x_max, y_min = np.round(coords)
300             x_coords = [x_min, x_max, x_max, x_min]
301             y_coords = [y_max, y_max, y_min, y_min]
302             plt.fill(x_coords, y_coords, 'k')
303         print(type(coords))
304
305         plt.xlabel('X Coordinate')
306         plt.ylabel('Y Coordinate')
307         plt.title('Grid Points with Corroded Areas')
308         plt.grid(True)
309         plt.show()
```

Appendix D: Welding Design

D.1 SET 1 PERIMETER ANGLES

The goal is to ensure that the weld strength exceeds the material's base strength. The cross-sectional area of the perimeter angle is calculated as:

$$A_1 = L_4 \times \frac{3}{8} \times 4 = 2.86 \text{ in}^2 \tag{D.1}$$

Given that the ultimate strength of the perimeter angle from Davis Structure is $\sigma_u = 65$ ksi, the ultimate shear force in the cross-section is computed as:

$$F_u = A_1 \times \sigma_u = 2.86 \times 65 = 186 \text{ kips} \tag{D.2}$$

The weld fracture force F_w must be greater than F_u. Considering the weld strength to be $F_{nw} = 70$ ksi, which is higher than the ultimate strength of the perimeter angle. According to AISC 36-22 (American Institute of Steel Construction, 2022), we require that:

$$F_w = 0.707wL \times F_{nw} \geq F_u \tag{D.3}$$

Here, $L = 16$ inches represents the total length of the perimeter angle. Solving Equation D.3 for the minimum weld thickness w_{min}, we obtain:

$$w_{\text{min}} = 0.23 \text{ inches} \tag{D.4}$$

Thus, the minimum required weld thickness to ensure the weld strength is sufficient is 0.23 inches.

D.2 SET 2 DIAGONAL ANGLES

The goal is to ensure that the weld strength exceeds the material's base strength. The cross-sectional area of the perimeter angle is calculated as:

$$A_2 = 0.68 \text{ in}^2 \tag{D.5}$$

Given that the ultimate strength of the perimeter angle from Davis Structure is $\sigma_u = 65$ ksi, the ultimate shear force in the cross-section is computed as:

$$F_u = A_1 \times \sigma_u = 0.68 \times 65 = 44 \text{ kips} \tag{D.6}$$

The weld fracture force F_w must be greater than F_u. Considering the weld strength to be $F_{nw} = 70\ \text{ksi}$, which is higher than the ultimate strength of the perimeter angle, we require that:

$$F_w = 0.707wL \times F_{nw} \geq F_u \tag{D.7}$$

Here, $L = 6\ \text{inches}$ represents the total length of the perimeter angle. Solving Equation D.7 for the minimum weld thickness w_{min}, we obtain:

$$w_{\text{min}} = 0.14\ \text{inches} \tag{D.8}$$

Thus, the minimum required weld thickness to ensure the weld strength is sufficient is $0.14\ \text{inches}$.

D.3 SET 3 RIBBED SHEET METAL

Since the thickness of the ribbed sheet is around $t_r \approx 0.0625\ \text{inches}$, according to AISC 360 - 22, Table J2.4, we could directily set the thickness of welding sheet to be:

$$w_{\text{min}} = 0.125\ \text{inches} \tag{D.9}$$

Appendix E: 3D Triangulation Algorithm

As shown in Figure 5.10b, points A, B, and C represent the positions of the Wirepots, while point D is the target point where displacement needs to be measured. Assuming that point D is projected onto the ABC plane as the origin O of the coordinate system, we define line OA as the x-axis and line OD as the z-axis to establish a Cartesian coordinate system. The coordinates of points A, B, C, and D can be expressed as:

$$\begin{aligned} A &= (x_A, 0, 0) \\ B &= (x_B, y_B, 0) \\ C &= (x_C, y_C, 0) \\ D &= (0, 0, z_D) \end{aligned} \tag{E.1}$$

There are six unknowns in this system, and we have exactly six known edge lengths to establish six equations, as listed below:

$$\begin{cases} (x_A - x_B)^2 + y_B^2 = l_1^2 \\ (x_B - x_C)^2 + (y_B - y_C)^2 = l_2^2 \\ (x_C - x_A)^2 + y_C^2 = l_3^2 \\ x_A^2 + z_D^2 = l_4^2 \\ x_B^2 + y_B^2 + z_D^2 = l_5^2 \\ x_C^2 + y_C^2 + z_D^2 = l_6^2 \end{cases} \tag{E.2}$$

After Equation E.2 has been solved, we have known all the locations of the points. When the loading begins, the locations of Point A, B and C will not change while the point D will have new coordinates which are:

$$D' = (x'_D, y'_D, z'_D) \tag{E.3}$$

The three Wirepots measure the extension or contraction during loading, represented as $\delta_1, \delta_2, \delta_3$, respectively. The corresponding new lengths between the target point and the Wirepots are:

$$\begin{aligned} l'_4 &= l_4 + \delta_1 \\ l'_5 &= l_5 + \delta_2 \\ l'_6 &= l_6 + \delta_3 \end{aligned} \tag{E.4}$$

Subsequently, we can set up a new set of equations as:

$$\begin{cases} (x_A - x'_D)^2 + (y_A - y'_D)^2 + (z'_D)^2 = (l'_4)^2 \\ (x_B - x'_D)^2 + (y_B - y'_D)^2 + (z'_D)^2 = (l'_5)^2 \\ (x_C - x'_D)^2 + (y_C - y'_D)^2 + (z'_D)^2 = (l'_6)^2 \end{cases} \tag{E.5}$$

In this way, we can know the coordinates of new D by solving Equation E.5, and compared with the original coordinates of D, we can know the displacement of the target point as:

$$\begin{aligned} \Delta_X &= x'_D \\ \Delta_Y &= y'_D \\ \Delta_Z &= x'_D - x_D \end{aligned} \tag{E.6}$$

The Matlab Script is listed as:

Listing E.1: MATLAB code for 3D Triangulation Algorithm

```
clear; close all; clc
l1 = 100; l2 = 100; l3 = 100;
l4 = 100; l5 = 100; l6 = 100;
delta1 = 0:0.001:1;
delta2 = 0:0.001:1;
delta3 = 0:0.001:1;

n = length(delta1);

disp('Starting computation...');
overall_timer = tic; % Start overall timer

%% Step 1: Precompute the static solution
% Calculate the corrdinates of the four points
syms xA xB yB xC yC zD xD_prime yD_prime zD_prime real
F1 = (xA - xB)^2 + yB^2 == l1^2;
F2 = (xB - xC)^2 + (yB - yC)^2 == l2^2;
F3 = (xC - xA)^2 + yC^2 == l3^2;
F4 = xA^2 + zD^2 == l4^2;
F5 = xB^2 + yB^2 + zD^2 == l5^2;
F6 = xC^2 + yC^2 + zD^2 == l6^2;

[sol_xA, sol_xB, sol_yB, sol_xC, sol_yC, sol_zD] = solve([F1, F2, F3, F4, F5,
    F6], [xA, xB, yB, xC, yC, zD], 'Real', true);
xA_sol = abs(double(sol_xA)); xB_sol = -abs(double(sol_xB));
yB_sol = abs(double(sol_yB)); xC_sol = -abs(double(sol_xC));
yC_sol = -abs(double(sol_yC));zD_sol = abs(double(sol_zD));

% Note: In different Target, the solution selection is different
xA = xA_sol(1); xB = xB_sol(1); yB = yB_sol(1);
xC = xC_sol(1); yC = yC_sol(1); zD = zD_sol(1);


%% Precompute deformed shape solver
% Calculate the coordinates of D function
```

```
syms L4_sym L5_sym L6_sym real

F7 = (xA - xD_prime)^2 + (yD_prime)^2 + zD_prime^2;
F8 = (xB - xD_prime)^2 + (yB - yD_prime)^2 + zD_prime^2;
F9 = (xC - xD_prime)^2 + (yC - yD_prime)^2 + zD_prime^2;

sol = solve([F7 == L4_sym^2, F8 == L5_sym^2, F9 == L6_sym^2], [xD_prime,
    yD_prime, zD_prime], 'Real', true,'ReturnConditions', true);
xD_func = matlabFunction(sol.xD_prime, 'Vars', [L4_sym L5_sym L6_sym]);
yD_func = matlabFunction(sol.yD_prime, 'Vars', [L4_sym L5_sym L6_sym]);
zD_func = matlabFunction(sol.zD_prime, 'Vars', [L4_sym L5_sym L6_sym]);

%% Step 2: Parallel computation
xD_disp = zeros(n, 1);
yD_disp = zeros(n, 1);
zD_disp = zeros(n, 1);
results = zeros(n, 6);

parfor i = 1:n
    % Update deformed lengths for this iteration
    l4_prime = l4 + delta1(i);
    l5_prime = l5 + delta2(i);
    l6_prime = l6 + delta3(i);

    % Use precomputed functions
    xD_vals = xD_func(l4_prime, l5_prime, l6_prime);
    yD_vals = yD_func(l4_prime, l5_prime, l6_prime);
    zD_vals = zD_func(l4_prime, l5_prime, l6_prime);

    % Find valid solution where zD > 0
    valid_idx = find(zD_vals > 0, 1);

    if ~isempty(valid_idx)
        xD_disp(i) = xD_vals(valid_idx);
        yD_disp(i) = yD_vals(valid_idx);
        zD_disp(i) = zD_vals(valid_idx) - zD;
    end

    results(i,:) = [delta1(i), delta2(i), delta3(i), xD_disp(i), yD_disp(i),
        zD_disp(i)];
end

overall_elapsed_time = toc(overall_timer); % Total elapsed time
disp('Computation completed.');

disp('Results for all delta combinations:');
disp('delta1 delta2 delta3 xD_disp yD_disp zD_disp');
disp(results);

fprintf('Total computation time: %.4f seconds.\n', overall_elapsed_time);
```

Appendix F: Python Implementation for Virtual Corrosion Generation and CI Calculation

F.1 VIRTUAL CORROSION PATTERN GENERATION

The following script generates virtual corrosion patterns based on the real corrosion map and the axial stress field data. Both are included in simple text Comma-Separated Values (*.cvs) files, namely, `abaqus_real_corrosion.csv` and `abaqus_S22_Map.csv`, respectively. A truncated Gaussian distribution around the observed corrosion cells, combined with a small baseline weight (`baseline = 1e-3`), is used to guide the sampling importance at different locations informed by the real corrosion map. The spread of the Gaussian is controlled by `sigma_cells = 1.0`, and the influence radius is set to `radius = 3` to reduce overlap between sampled regions.

```python
import numpy as np
import pandas as pd
import os
import time
from scipy.spatial import cKDTree

# === CONFIGURATION ===
data_path = "./abaqus_real_corrosion.csv"
stress_csv_path = "./abaqus_S22_Map.csv"
output_folder = "./corrosion_patches"
num_csv = 10
baseline = 1e-3
sigma_cells = 1.0
radius = 3
ci_min_threshold = 0.01
ci_max_threshold = 0.9
max_attempts = 100
# ======================

# Load stress field
stress_df = pd.read_csv(stress_csv_path)
x_vals = stress_df["X"].values
z_vals = stress_df["Z"].values
stress_vals = np.abs(stress_df["S22 Stress"].values)
stress_points = np.column_stack((x_vals, z_vals))
```

```
tree = cKDTree(stress_points)

# Load real corrosion coordinates
df = pd.read_csv(data_path, header=None)
df.columns = ["xMin", "zMax", "xMax", "zMin"]
centers = np.column_stack(((df["xMin"] + df["xMax"]) / 2, (df["zMin"] + df["
    zMax"]) / 2))

# Define grid
num_col, num_row = 30, 10
x_min, x_max = 22.25, 205.125 # Same in Abaqus Location
z_min, z_max = 213.75, 276

dx = (x_max - x_min) / num_col
dz = (z_max - z_min) / num_row
area = dx * dz

all_boxes = []
grid_centers, grid_idx = [], []
for j in range(num_row):
    for i in range(num_col):
        x1 = x_min + i * dx
        x2 = x1 + dx
        z1 = z_min + j * dz
        z2 = z1 + dz
        all_boxes.append([x1, z2, x2, z1])
        grid_centers.append([(x1 + x2) / 2, (z1 + z2) / 2])
        grid_idx.append([i, j])
grid_centers = np.array(grid_centers)
grid_idx = np.array(grid_idx)

# Lookup stress for all grid centers
_, idx_all = tree.query(grid_centers)
stress_all = stress_vals[idx_all]
denominator = np.sum(stress_all)

# Importance weights: truncated Gaussian + baseline
importance = np.zeros(len(grid_centers))
for x_c, z_c in centers:
    i_c = int(np.clip((x_c - x_min) / dx, 0, num_col - 1))
    j_c = int(np.clip((z_c - z_min) / dz, 0, num_row - 1))
    di = np.abs(grid_idx[:,0] - i_c)
    dj = np.abs(grid_idx[:,1] - j_c)
    d = np.sqrt(di**2 + dj**2)
    mask = d <= radius
    importance[mask] += np.exp(-(d[mask]**2) / (2 * sigma_cells**2))

importance += baseline
importance /= importance.sum()

# Generate output
os.makedirs(output_folder, exist_ok=True)
summary_path = os.path.join(output_folder, "CI_list.csv")
```

```
if os.path.exists(summary_path):
    os.remove(summary_path)
used_sets = set()

for k in range(num_csv):
    attempt = 0
    success = False

    while not success and attempt < max_attempts:
        # Using Time Variation to avoid repeating
        seed_val = int(time.time_ns()) % (2**32) + attempt + k * 1000
        rng = np.random.default_rng(seed_val)

        num_select = rng.integers(10, 300)
        selected = rng.choice(len(all_boxes), size=num_select, replace=False, p=
            importance)

        selected_set = tuple(sorted(selected))
        if selected_set in used_sets:
            attempt += 1
            continue
        used_sets.add(selected_set)

        corroded_centers = np.array(grid_centers)[selected]
        _, idx_cor = tree.query(corroded_centers)
        stress_cor = stress_vals[idx_cor]
        numerator = np.sum(stress_cor)
        CI = numerator / denominator

        if ci_min_threshold <= CI <= ci_max_threshold:
            boxes_k = [all_boxes[i] for i in selected]
            fname = "corrosion_{:02d}.csv".format(k + 1)
            pd.DataFrame(boxes_k).to_csv(os.path.join(output_folder, fname),
                header=False, index=False)

            with open(summary_path, "a") as f:
                f.write("{}, {:.6f}\n".format(fname, CI))
            print("Saved:", fname, "| CI =", round(CI, 4))
            success = True

        attempt += 1

    if not success:
        print("Could not generate CI < {:.2f} after {} attempts for sample {}".
            format(
            ci_max_threshold, max_attempts, k + 1))
```

F.2 CORROSION INDEX CALCULATION USING AXIAL STRESS MAP

The following Python script provides the implementation used to calculate the Corrosion Index (CI) based on reaction force data and axial stress field outputs from Abaqus simulations.

```python
import pandas as pd
import numpy as np
from scipy.spatial import cKDTree

def compute_CI_from_csv(corrosion_path, stress_csv_path, E_0, E_cor, t_0,
    t_cor):
    # Area at each cell is the same
    # Load Corrosion Patch Data
    corrosion_patch_data = pd.read_csv(corrosion_path, header=None)
    x1 = corrosion_patch_data.iloc[:, 0]
    z2 = corrosion_patch_data.iloc[:, 1]
    x2 = corrosion_patch_data.iloc[:, 2]
    z1 = corrosion_patch_data.iloc[:, 3]

    # Load stress field
    stress_df = pd.read_csv(stress_csv_path)
    x_vals = stress_df["X"].values
    z_vals = stress_df["Z"].values
    stress_vals = np.abs(stress_df["Stress"].values)
    stress_points = np.column_stack((x_vals, z_vals))
    tree = cKDTree(stress_points)

    # Calculate grid centers
    grid_centers = np.column_stack(((x1 + x2) / 2, (z1 + z2) / 2))
    _, idx_all = tree.query(grid_centers)
    stress_corroded = stress_vals[idx_all]

    # Compute all CI values
    CI_energy = np.sum((stress_corroded**2/E_cor*t_cor) / np.sum(stress_vals
        **2/E_0*t_0))

    return CI_energy

# ------------------- User Settings -------------------
rf_csv_path = "./RF_Results.csv" # Reaction Force Results
stress_csv_path = "./abaqus_S22_Map_new.csv" # Abaqus Axial Stress Map
output_path = "CIRF_v5.csv"
E_0, E_cor, t_0, t_cor = 30397000, 6991000, 0.0625, 0.014
# -----------------------------------------------------

force_data = pd.read_csv(rf_csv_path)
rf3sum = np.trunc(force_data.iloc[:, 1] + force_data.iloc[:, 2]).astype(int)

results = []

for i in range(1, 11):
    corrosion_path = f"./Corrosion_Pattern_Datasets/corrosion_{i:02d}.csv"
    CI_energy = compute_CI_from_csv(corrosion_path, stress_csv_path,E_0, E_cor,
        t_0, t_cor)

    results.append({
        "Index": i,
        "rf3sum": rf3sum[i - 1],
```

```
        "CI_energy": np.round(CI_energy, 3),
    })

# Convert to .csv
df = pd.DataFrame(results)
df.to_csv(output_path, index=False)
print(f"Saved {output_path} with rf3sum inserted.")

# CI calculation of real corrosion pattern
corrosion_path = "abaqus_real_corrosion.csv"
compute_CI_from_csv(corrosion_path, stress_csv_path,E_0, E_cor, t_0, t_cor)
```

F.3 CORROSION INDEX CALCULATION USING LOCATION-WEIGHTED MAP

```

def compute_CI_A_from_csv(corrosion_path, stress_csv_path=None):
    # Load corrosion patch rectangles: x1, z2, x2, z1
    corrosion_patch_data = pd.read_csv(corrosion_path, header=None)
    x1 = corrosion_patch_data.iloc[:, 0].to_numpy(dtype=float)
    z2 = corrosion_patch_data.iloc[:, 1].to_numpy(dtype=float)
    x2 = corrosion_patch_data.iloc[:, 2].to_numpy(dtype=float)
    z1 = corrosion_patch_data.iloc[:, 3].to_numpy(dtype=float)

    # Load stress field
    stress_df = pd.read_csv(stress_csv_path)
    x_vals = stress_df["X"].values
    z_vals = stress_df["Z"].values
    stress_vals = np.abs(stress_df["Stress"].values)
    stress_points = np.column_stack((x_vals, z_vals))
    tree = cKDTree(stress_points)

    # Calculate grid centers for Stress
    grid_centers = np.column_stack(((x1 + x2) / 2, (z1 + z2) / 2))
    _, idx_all = tree.query(grid_centers)
    stress_corroded = stress_vals[idx_all]
    CI_energy_n = np.sqrt(np.sum((stress_corroded**2)) / np.sum(stress_vals**2)
        )

    # Total panel area dimensions based on coordinates
    x_min = 22.25
    x_max = 205.125
    z_min = 213.75
    z_max = 276.00

    total_width = x_max - x_min
    total_height = z_max - z_min
    total_area = total_width * total_height

    # Per-box corroded area and their center points
    w = np.abs(x2 - x1)
    h = np.abs(z2 - z1)
```

```
    box_areas = w * h

    x_centers = (x1 + x2) / 2.0
    z_centers = (z1 + z2) / 2.0

    # ------------------------------------------------------------
    # Method 1: Original Pure Area Ratio (Baseline)
    # ------------------------------------------------------------
    corroded_area = np.sum(box_areas)
    CI_A = float(np.clip(corroded_area / total_area, 0.0, 1.0))

    # ------------------------------------------------------------
    # Method 2: 10% Edge-Band Weighted Area Ratio (Precision Model)
    # ------------------------------------------------------------

    # Band Width (10%)
    edge_ratio_x = 0.10
    edge_ratio_z = 0.10

    # 1. Horizontal Direction
    w_x_band = total_width * edge_ratio_x
    cond_x = [
       x_centers <= x_min + w_x_band # Left 10% Dangerous
    ]
    choices_x = [3.0]
    weights_X = np.select(cond_x, choices_x, default=0.5)

    # 2. Vertical Direction (z)
    h_z_band = total_height * edge_ratio_z
    cond_z = [
       z_centers <= z_min + h_z_band, # Bottom 10%
       z_centers >= z_max - h_z_band # Top 10%
    ]
    # Up/Down Frame
    choices_z = [3.0, 3.0]
    weights_Z = np.select(cond_z, choices_z, default=0.5)
    final_weights = weights_X * weights_Z

    # 3. Weighted Corroded Area
    weighted_corroded_area = np.sum(box_areas * final_weights)

    # 4. Normalization
    # X direction
    avg_weight_x = (edge_ratio_x * 3.0) + ((1 - edge_ratio_x) * 0.5)
    # Z direction
    avg_weight_z = (2 * edge_ratio_z * 3.0) + ((1 - 2 * edge_ratio_z) * 0.5)

    weighted_total_area = total_area * (avg_weight_x * avg_weight_z)

    CI_A2 = float(np.clip(weighted_corroded_area / weighted_total_area, 0.0,
       1.0))

    return CI_energy_n, CI_A, CI_A2
```

```
89
90  # ------------------- User Settings -------------------
91  rf_csv_path = "./RF_Results_70.csv"
92  stress_csv_path = "./abaqus_S22_Map_new.csv"
93  output_path = "CIRF_simplified_v5.csv"
94
95  # -----------------------------------------------------
96
97  # Load rf3sum from force results
98  force_data = pd.read_csv(rf_csv_path)
99  rf3sum = np.trunc(force_data.iloc[:, 1] + force_data.iloc[:, 2]).astype(int)
100
101 results = []
102
103 for i in range(1, 81): # corrosion_01.csv ~ corrosion_80.csv
104     corrosion_path = f"./Corrosion_Pattern_Datasets_Simplified/corrosion_{i:02d
          }.csv"
105     if not os.path.exists(corrosion_path):
106         continue
107     CI_energy_n, CI_A, CI_A2 = compute_CI_A_from_csv(corrosion_path,
          stress_csv_path)
108
109     results.append({
110         "Index": i,
111         "rf3sum": rf3sum[i - 1],
112         "CI_energy_n": np.round(CI_energy_n, 3),
113         "CI_Area": np.round(CI_A, 4),
114         "CI_Area2": np.round(CI_A2, 4)
115     })
116
117 # First Experimental Point
118 results.append({
119     "Index": 81,
120     "rf3sum": 31000,
121     "CI_energy_n": 0.001,
122     "CI_Area": 0.001,
123     "CI_Area2": 0.001
124 })
125
126 # Second Experimental Point
127 corrosion_path = "abaqus_real_corrosion.csv"
128 CI_corr, CI_corrA,CI_corrA2 = compute_CI_A_from_csv(corrosion_path,
      stress_csv_path)
129 results.append({
130     "Index": 81,
131     "rf3sum": 15000,
132     "CI_energy_n": CI_corr,
133     "CI_Area": np.round(CI_corrA, 4),
134     "CI_Area2": np.round(CI_corrA2, 4)
135 })
136
137
138 # Save as CSV
```

```
df = pd.DataFrame(results)
df.to_csv(output_path, index=False)
print(f"Saved {output_path} with CI_A and rf3sum.")


data = pd.read_csv('./CIRF_simplified_v5.csv')

def logistic_5pl_UL(x, L, U, k, x0, nu):
   return L + (U - L) / (1.0 + np.exp(k * (x - x0)))**nu

CIA = data['CI_Area2'].values
force_capacity = data['rf3sum'].values / 1e3 # kips
U_ref = 31.0
force_capacity_norm = force_capacity / U_ref * 100
ydata = force_capacity_norm

p0 = [16/31*100, 100, 10.0, np.quantile(CIA, 0.4), 1.5]
bounds = ([0, 100, 0.0, 0.0, 0.3],
        [70, 101, 50.0, 1.0, 10])

popt_5A, _ = curve_fit(
   logistic_5pl_UL, CIA, ydata,
   p0=p0, bounds=bounds, maxfev=50000
)

y_fit = logistic_5pl_UL(CIA, *popt_5A)
r2_5A = r2_score(ydata, y_fit)
resid = ydata - y_fit

x_valsA = np.linspace(CIA.min(), 1.0, 400)
y_curve_5A = logistic_5pl_UL(x_valsA, *popt_5A)

# Bootstrap prediction band
n_boot = 1000
rng = np.random.default_rng(42)
n = len(CIA)

y_pred_boot = []
for _ in range(n_boot):
   idx = rng.integers(0, n, n)
   x_bs = CIA[idx]
   y_bs = ydata[idx]
   try:
      popt_bs, _ = curve_fit(
         logistic_5pl_UL, x_bs, y_bs,
         p0=popt_5A, bounds=bounds, maxfev=50000
      )
      y_bs_curve = logistic_5pl_UL(x_valsA, *popt_bs)
      eps = rng.choice(resid, size=len(x_valsA), replace=True)
      y_pred_boot.append(y_bs_curve + eps)
   except RuntimeError:
      continue

```

```
y_pred_boot = np.array(y_pred_boot)

# Prediction Range
pi_lower = np.percentile(y_pred_boot, 2.5, axis=0)
pi_upper = np.percentile(y_pred_boot, 97.5, axis=0)

y_mean_boot = []
for _ in range(300):
    idx = rng.integers(0, n, n)
    x_bs = CIA[idx]
    y_bs = ydata[idx]
    try:
        popt_bs, _ = curve_fit(
            logistic_5pl_UL, x_bs, y_bs,
            p0=popt_5A, bounds=bounds, maxfev=50000
        )
        y_mean_boot.append(logistic_5pl_UL(x_valsA, *popt_bs))
    except RuntimeError:
        continue

y_mean_boot = np.array(y_mean_boot)
ci_lower = np.percentile(y_mean_boot, 2.5, axis=0)
ci_upper = np.percentile(y_mean_boot, 97.5, axis=0)


fig, ax = plt.subplots(figsize=(10, 6), dpi=100)
ax.scatter(CIA, ydata, s=60, color="#00529B", edgecolor='w',
          linewidth=0.8, zorder=4, label=f'Simulated Points ({len(CIA)})')
ax.scatter([0, CI_corrA2], [100, 15/31*100], s=80, color='red', edgecolor='k',
          linewidth=0.3, marker='o', zorder=4, label='Experimental Points (2)')
ax.fill_between(x_valsA, pi_lower, pi_upper,
              color="#e0e0e0", alpha=0.7, zorder=1, label='95% Prediction Band')
ax.fill_between(x_valsA, ci_lower, ci_upper,
              color="#adcbe3", alpha=0.8, zorder=2, label='95% Confidence Band')

ax.plot(x_valsA, y_curve_5A, color="#D62728", lw=3,
        zorder=5, label=f'5PL Fit (R^2={r2_5A:.2f})')

ax.set_xlabel('Adjusted Area Ratio', fontsize=16)
ax.set_ylabel('Normalized Force Capacity [%]', fontsize=16)
ax.tick_params(axis='both', which='major', direction='in', length=5, width=1,
    labelsize=14)
ax.grid(True, ls='--', lw=0.5, alpha=0.5)
ax.legend(fontsize=14, framealpha=0.95)
ax.set_xlim(-0.02, 1)
ax.set_ylim(0, 120)

plt.tight_layout()
# plt.savefig("./CI_Simplified_AdjustedAreaRatio",dpi = 300)
plt.show()
```

Appendix G: Python Implementation to Extract Reaction Force Values at a Node Set

```python
# -- Python 2.7 --
# -*- coding: mbcs -*-
import os
import sys
import csv
from odbAccess import openOdb
from abaqus import *


# ------------- USER CONFIG -------------
JOB_NAME   = "Job-31"       # .odb base name (without extension)
STEP_NAME  = "Step-Pushover" # step containing RF output
NODE_SET_NAME = "BASENODESET" # node set to export
# NODE_SET_NAME = "ACT_NODE"
INSTANCE_NAME = 'PART-1-1'      # None = keep all instances; or set to "PART-1-1"
OUTPUT_VAR = "RF"           # field output variable
CSV_NAME   = "./BaseNodes_RF3.csv" # Output CSV name
# ---------------------------------------

odb = openOdb(path=JOB_NAME + '.odb')
def find_node_sets_label(NODE_SET_NAME):
    node_labels = []
    node_set = odb.rootAssembly.nodeSets[NODE_SET_NAME]
    for node in node_set.nodes[0]:
        node_labels.append(node.label)
    print("[INFO] Total nodes collected:", len(node_labels))
    return node_labels

def build_field_dict(field, instance_name):
    """{node_label: value}"""
    field_dict = {}
    for value in field.values:
        if (instance_name is None) or (value.instance.name == instance_name):
            field_dict[value.nodeLabel] = value
    return field_dict

```

```
37 def main(node_labels):
38     odb_file = JOB_NAME + ".odb"
39     if not os.path.exists(odb_file):
40         sys.exit("ODB file '" + odb_file + "' not found in the current directory
           .")
41 
42     odb = openOdb(path=odb_file, readOnly=True)
43 
44     if STEP_NAME not in odb.steps:
45         odb.close()
46         sys.exit("Step '" + STEP_NAME + "' not found. Available steps: " + str(
           odb.steps.keys()))
47 
48     step = odb.steps[STEP_NAME]
49 
50     with open(CSV_NAME, "w", newline='', encoding='utf-8') as f:
51         writer = csv.writer(f)
52         header = ["time"] + ["Node_{}_RF3".format(label) for label in
           node_labels]
53         writer.writerow(header)
54 
55         for frame in step.frames:
56             if OUTPUT_VAR not in frame.fieldOutputs.keys():
57                 odb.close()
58                 sys.exit("Field '" + OUTPUT_VAR + "' not found in frame.")
59 
60             field = frame.fieldOutputs[OUTPUT_VAR]
61 
62             #
63             field_dict = build_field_dict(field, INSTANCE_NAME)
64 
65             row = [frame.frameValue]
66 
67             for label in node_labels:
68                 if label not in field_dict:
69                     odb.close
70                     sys.exit("Node label " + str(label) + " not found in field '" +
                           OUTPUT_VAR + "' at frame time " + str(frame.frameValue))
71 
72                 val = field_dict[label]
73                 RF3 = val.data[2]
74                 row.append(RF3)
75 
76             writer.writerow(row)
77     odb.close()
78     print("[INFO] Export complete -> %s" % CSV_NAME)
79 
80 if __name__ == '__main__':
81     node_labels = find_node_sets_label(NODE_SET_NAME)
82     main(node_labels)
```

Appendix H: Field Deployment Guide for Corrosion Assessment

This appendix provides a step-by-step guide for Caltrans SM&I engineers to assess the residual capacity of corroded overhead sign structures using the Location-Weighted Corrosion Index methodology developed in Section 7.7. The entire workflow requires only field photographs and the provided Python scripts.

H.1 OVERVIEW

The assessment pipeline consists of three stages:

- **Stage 1. Corrosion Detection**: Run the trained U-Net segmentation model on a field photograph to generate a binary corrosion mask.
- **Stage 2. Coordinate Extraction**: Use the `corroded_shape_detection.py` tool to identify the sign panel boundary via interactive corner selection, discretize the mask into grid cells, and export the coordinates of corroded cells.
- **Stage 3. Index Calculation & Capacity Estimation**: Compute the location-weighted corrosion index CI_L and visualize the estimated residual capacity against the calibrated reference curve.

H.2 PREREQUISITES

H.2.1 Software Environment

The toolkit requires Python 3.8 or later with the packages listed in Table H.1:

Installation:

```
pip install numpy pandas opencv-python matplotlib torch torchvision pyyaml
    imageio
```

Table H.1: Required Python packages.

Package	Purpose
`numpy`	Numerical computation
`pandas`	CSV file I/O
`opencv-python` (`cv2`)	Image processing and perspective transformation
`matplotlib`	Visualization and interactive point selection
`torch` (PyTorch)	Running the trained segmentation model
`torchvision`	Pre-trained model architectures
`pyyaml`	Reading configuration files
`imageio`	Saving mask and probability images

H.2.2 Toolkit Directory Structure

The provided toolkit (`SIGN_CV_CODE/`), available upon request, contains the following key components:

- **README.md**: A markdown document describing how to execute the toolkit.
- **Corrosion segmentation (Stage 1)**:
  - `predict_unet.py`, `predictor.py`: U-Net inference scripts.
  - `unet/`: U-Net model architecture definition.
  - `UNET_CHECK_POINT_DIR/`: Pre-trained model weights included in the following file: `best_checkpoint.pytorch`.
  - `dataloader.py`, `dice_loss.py`, `utils.py`: Supporting modules.
- **Coordinate extraction (Stage 2)**:
  - `corroded_shape_detection.py`: Grid overlay, corrosion detection, and coordinate transformation.
- **Index calculation (Stage 3)**:
  - `ci_calculator.py`: CI_L computation and reference-curve plotting.
  - `CIRF_simulation.csv`: Simulation dataset for five-parameter logistic (5PL) curve fitting.
- **Configuration**:
  - `resources/test_unet_sign.yaml`: U-Net inference configuration.
- **Field inspection data**:
  - `Field_Inspection/raw/`: Place input field photographs here.
  - `Field_Inspection/mask/`: Binary masks (output from Stage 1).
  - `Field_Inspection/prob/`: Probability maps (output from Stage 1).

H.3 STAGE 1: CORROSION DETECTION VIA COMPUTER VISION

H.3.1 Photo Acquisition

Capture a clear, front-facing photograph of the sign panel. The following guidelines ensure good performance:

- Take the photograph of the entire panel with all four edges visible.
- Minimize perspective distortion by positioning the camera lens as close as possible to being parallel to the panel surface.
- Avoid strong light exposures or shadows.

Save and place the photograph as `./Field_Inspection/raw/raw.JPG`.

H.3.2 Running the Segmentation Model

The corrosion detection model is a U-Net segmentation network trained on overhead sign structure images. The developed inference is configured via `./resources/test_unet_sign.yaml`, which is a YAML file.

Configuration. The YAML file (see Listing H.1) specifies the model checkpoint path, inference threshold, and input/output directories, as follows:

Listing H.1: Configuration file (`test_unet_sign.yaml`).

```yaml
model_path: ./UNET_CHECK_POINT_DIR/best_checkpoint.pytorch
threshold: 0.5
output_dir: ./Field_Inspection
test_loader:
  file_path: ./Field_Inspection
  batch_size: 4
  phase: test
```

The `test_loader.file_path` should point to the directory containing a `raw/` subfolder with the input photograph(s).

To segment the corrosion part from the raw image, run:

```
python predict_unet.py --config ./resources/test_unet_sign.yaml
```

The model produces two outputs per input image:

- `./Field_Inspection/mask/<idx>_mask.png`: Binary corrosion mask (white $=$ 255 for corroded pixels, black $=$ 0 for intact surface). This is the primary input for Stage 2.

- `./Field_Inspection/prob/<idx>_prob.png`: Probability map shows the model's pixel-level confidence (useful for quality verification).

Visual inspection of the generated mask is recommended before proceeding. If the mask contains obvious false positives (e.g., shadows or stains misidentified as corrosion), the photograph should be retaken under improved lighting conditions.

H.4 STAGE 2: COORDINATE EXTRACTION

This stage uses the `corroded_shape_detection.py` script to establish the panel boundary, overlay an evaluation grid on the corrosion mask, and export the coordinates of the corroded cells.

H.4.1 Running the Coordinate Extraction

```
python corroded_shape_detection.py
```

H.4.2 Interactive Corner Selection

Upon execution, a matplotlib window displays the original field photograph. The user must **click four corner points** of the sign panel. The click order does not matter as the software automatically reorders the points into a consistent clockwise sequence. Points should be placed tightly along the sign panel edges for accurate boundary definition.

H.4.3 Processing Steps

After corner selection, the script performs the following operations automatically:

1. Overlays a grid of cells onto the binary mask image.
2. For each grid cell, computes the fraction of white (corroded) pixels.
3. Marks cells with corrosion fraction $\geq 50\%$ as corroded.
4. Applies a perspective transformation to map pixel coordinates from the mask image to the panel coordinate system.
5. Exports two CSV files to the project root directory:
    - `corroded_coordinates_mask.csv`: Corroded cell coordinates in the mask pixel space.
    - `corroded_coordinates_abaqus_new.csv`: Corroded cell coordinates in the panel coordinate system, formatted as `[x1, y1, x2, y2]` per row.

The file `corroded_coordinates_abaqus_new.csv` is the input for Stage 3.

H.5 STAGE 3: CORROSION INDEX CALCULATION AND CAPACITY ESTIMATION

H.5.1 Computing CI_L and Generating the Reference Plot

With the corroded cell coordinates from Stage 2, run:

```
python ci_calculator.py --csv corroded_coordinates_abaqus_new.csv \
   --save ./CI_plot.png
```

Table H.2 lists all available command-line options.

Table H.2: Command-line options for `ci_calculator.py`.

Flag	Default	Purpose
`--csv`	(required)	Path to corroded-cell coordinate CSV
`--xmin / --xmax`	22.25 / 205.125	Panel horizontal extent (panel coords)
`--zmin / --zmax`	213.75 / 276.00	Panel vertical extent (panel coords)
`--edge-ratio`	0.10	High-stress edge-band fraction
`--reference-csv`	`./CIRF_simulation.csv`	Simulation dataset for 5PL fitting
`--no-plot`	—	Skip figure generation
`--save`	—	Save figure to specified path

H.5.2 Interpreting the Output

The script produces two key outputs, as follows:

Console report. A text summary including the information of the data and CI_L (location-weighted corrosion index).

Reference curve plot. A figure (e.g., Figure H.1) overlaying the measured CI_L value on the 5PL reference curve fitted from the simulation dataset (`CIRF_simulation.csv`). The plot includes:

- A **green dashed vertical line** indicating the measured CI_L value.
- The fitted 5PL curve (This curve is from Figure 7.9) showing the relationship between CI_L and normalized force capacity C_f.
- **Horizontal reference lines** marking the 5PL fit value and the upper/lower bounds of the 95% prediction band at the measured CI_L.

The engineer reads the estimated strength-loss bracket directly from the plot and makes the final engineering judgment regarding the structural condition and any needed subsequent inspection.

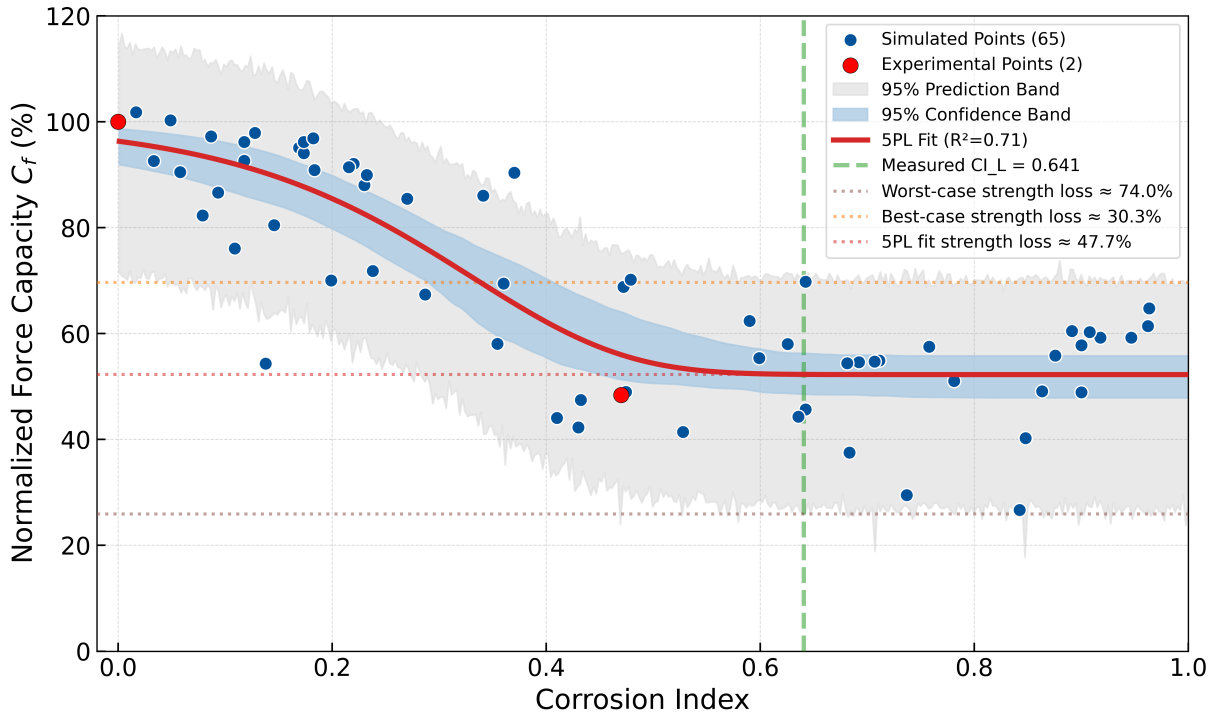


Figure H.1: Field assessment example: measured CI_L overlaid on the 5PL reference curve with 95% prediction band.

Appendix I: UMAT Formulation Using Clough Model for Shell Elements

I.1 SHELL MATRIX FORMULATION

UMAT Input Parameters.

$$\begin{aligned} E_0 &\quad \text{(initial tangent stiffness)},\\ \sigma_y &\quad \text{(initial yield stress)},\\ \eta &\quad \text{(post-yield hardening ratio)},\\ \beta &\quad \text{(reversal stiffness amplifier; here } \beta = 1.0),\\ \alpha &\quad \text{(Viscous Parameter } \alpha = 0 \text{ during static loading)} \end{aligned} \tag{I.1}$$

Plane-Stress UMAT Assembly. Directional stiffness values are taken as follows,

$$E_1 = E_t^{(1)}, \qquad E_2 = E_t^{(2)}, \qquad E_3 = E_0, \tag{I.2}$$

where $E_t^{(1)}$ and $E_t^{(2)}$ are tangents from the 1-D clough subroutines, and are explained in Section I.2. Poisson's ratios are

$$\nu_{12} = \nu_{13} = \nu_{23} = 0.25, \tag{I.3}$$

and shear moduli are

$$G_{12} = \frac{E_0}{2(1+\nu_{12})},\ G_{13} = \frac{E_0}{2(1+\nu_{13})},\ G_{23} = \frac{E_0}{2(1+\nu_{23})}. \tag{I.4}$$

However, the coupling (off-diagonal) terms were found to prevent the convergence of the Newton-Raphson iteration. Therefore, only diagonal terms are retained herein. The compliance matrix $\mathbf{S}$ and its inverse $\mathbf{C} = \mathbf{S}^{-1}$ are constructed as:

$$\mathbf{S} = \begin{bmatrix} \frac{1}{E_1} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{E_2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{E_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G_{12}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{G_{13}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G_{23}} \end{bmatrix}, \qquad \mathbf{C} = \mathbf{S}^{-1}. \tag{I.5}$$

The plane stress reduced stiffness follows from the Schur complement (Voigt order in Abaqus: 11, 22, 33, 12, 13, 23)

$$\mathbf{C}^{ps}_{pp} = \mathbf{C}_{pp} - \mathbf{C}_{p3}\, C^{-1}_{33}\, \mathbf{C}_{3p}, \qquad p = \{1, 2, 4\}. \tag{I.6}$$

$$\mathbf{C}^{ps} = \begin{bmatrix} C^{ps}_{11} & C^{ps}_{12} & 0 & C^{ps}_{13} & 0 & 0 \\ C^{ps}_{21} & C^{ps}_{22} & 0 & C^{ps}_{23} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ C^{ps}_{31} & C^{ps}_{32} & 0 & C^{ps}_{33} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{I.7}$$

Viscous Regularization. A Kelvin–Voigt type viscosity is added in directions 1 and 2:

$$c_v = \alpha \frac{E_0}{\Delta t}, \qquad C^{ps}_{11} \leftarrow C^{ps}_{11} + c_v, \quad C^{ps}_{22} \leftarrow C^{ps}_{22} + c_v. \tag{I.8}$$

The consistent tangent is then

$$\texttt{DDSDDE} = \mathbf{C}^{ps}. \tag{I.9}$$

Stress Update. Stress increments are computed as

$$\begin{bmatrix} \Delta\sigma_1 \\ \Delta\sigma_2 \\ \Delta\sigma_3 \\ \Delta\tau_{12} \\ \Delta\tau_{13} \\ \Delta\tau_{23} \end{bmatrix} = \mathbf{C}^{ps} \begin{bmatrix} \Delta\varepsilon_1 \\ \Delta\varepsilon_2 \\ \Delta\varepsilon_3 \\ \Delta\gamma_{12} \\ \Delta\gamma_{13} \\ \Delta\gamma_{23} \end{bmatrix}, \tag{I.10}$$

and the final stresses are expressed as follows,

$$\sigma_1^{\text{new}} = \sigma_1^u + \Delta\sigma_1, \quad \sigma_2^{\text{new}} = \sigma_2^u + \Delta\sigma_2, \quad \tau_{12}^{\text{new}} = \tau_{12}^u + \Delta\tau_{12}, \quad \sigma_3 = \tau_{13} = \tau_{23} = 0. \tag{I.11}$$

I.2 1-D CLOUGH MODEL SUBROUTINE

It is noted that the original code is from (Qu and Pan, 2015), it is summarized here for better understanding the Clough model we are using.

Subroutine Input & Output Parameters.

$$\begin{aligned} s &\quad \text{stress at the start of the step,} \\ e &\quad \text{committed strain at the start of the step,} \\ de &\quad \text{strain increment applied in this step,} \\ E_t &\quad \text{algorithmic (consistent) tangent modulus returned for this step.} \end{aligned} \tag{I.12}$$

It is to be noted that the *committed strain* refers to the accumulated strain state converged and stored at the end of the previous load step, which serves as the starting point for the current increment.

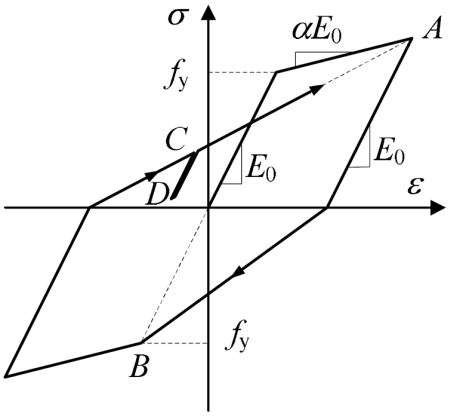


Figure I.1: Clough Model Diagram.

State variables (updated each step):

$$
\begin{aligned}
\varepsilon_{\max} &\quad \text{maximum strain attained so far (history variable),}\\
\varepsilon_{\min} &\quad \text{minimum strain attained so far (history variable),}\\
\varepsilon_{rt} &\quad \text{strain at the last load reversal toward tension,}\\
\sigma_{rt} &\quad \text{stress at the last load reversal toward tension,}\\
\varepsilon_{rc} &\quad \text{strain at the last load reversal toward compression,}\\
\sigma_{rc} &\quad \text{stress at the last load reversal toward compression,}\\
k_{\text{on}} \in \{0,1,2\} &\quad \text{loading-branch flag: } 0 \text{ (unset), } 1 \text{ (tension), } 2 \text{ (compression).}
\end{aligned}
\tag{I.13}
$$

Given the previous step stress–strain (σ, ε), the strain increment $\Delta\varepsilon$, and the new total strain $\varepsilon^+ = \varepsilon + \Delta\varepsilon$, the algorithm proceeds as follows.

Initialization and Load Reversal. If $k_{\text{on}} = 0$,

$$
\varepsilon_{\max} = \frac{\sigma_y}{E_0}, \qquad \varepsilon_{\min} = -\frac{\sigma_y}{E_0}, \qquad k_{\text{on}} = \begin{cases} 1, & \Delta\varepsilon \geq 0 \\ 2, & \Delta\varepsilon < 0 \end{cases}. \tag{I.14}
$$

Load reversal from tension direction to compression, update the stress/strain and record the max strain

$$
\text{If } (k_{\text{on}} = 1 \;\&\; \Delta\varepsilon < 0) \;:\; \begin{cases} k_{\text{on}} \leftarrow 2, \\ \text{if } \sigma > 0 :\; \varepsilon_{rc} \leftarrow \varepsilon,\; \sigma_{rc} \leftarrow \sigma, \\ \varepsilon_{\max} \leftarrow \max(\varepsilon_{\max},\, \varepsilon). \end{cases} \tag{I.15}
$$

Load reversal from compression direction to tension, update the stress/strain and record the min strain

$$
\text{If } (k_{\text{on}} = 2 \;\&\; \Delta\varepsilon > 0) \;:\; \begin{cases} k_{\text{on}} \leftarrow 1, \\ \text{if } \sigma < 0 :\; \varepsilon_{rt} \leftarrow \varepsilon,\; \sigma_{rt} \leftarrow \sigma, \\ \varepsilon_{\min} \leftarrow \min(\varepsilon_{\min},\, \varepsilon). \end{cases} \tag{I.16}
$$

Trial tangent and trial stress update.

$$E_{\text{trial}} = \begin{cases} \beta\, E_0, & (k_{\text{on}} = 2 \;\&\; \Delta\varepsilon > 0) \text{ or } (k_{\text{on}} = 1 \;\&\; \Delta\varepsilon < 0) \\ E_0, & \text{otherwise} \end{cases} \tag{I.17}$$

$$\sigma^{\text{trial}} = \sigma + E_{\text{trial}}\, \Delta\varepsilon, \qquad E_t \leftarrow E_{\text{trial}}. \tag{I.18}$$

Monotonic (post-yield) envelopes. For loading in tension ($\Delta\varepsilon \geq 0$),

$$\sigma^{+}_{\text{env}}(\varepsilon^{+}) = \sigma_y + \eta\, E_0\big(\varepsilon^{+} - \tfrac{\sigma_y}{E_0}\big), \qquad E^{+}_{\text{env}} = \eta\, E_0. \tag{I.19}$$

If $\sigma^{\text{trial}} \geq \sigma^{+}_{\text{env}}$ then set $\sigma \leftarrow \sigma^{+}_{\text{env}}$ and $E_t \leftarrow E^{+}_{\text{env}}$.

For loading in compression ($\Delta\varepsilon < 0$),

$$\sigma^{-}_{\text{env}}(\varepsilon^{+}) = -\sigma_y + \eta\, E_0\big(\varepsilon^{+} + \tfrac{\sigma_y}{E_0}\big), \qquad E^{-}_{\text{env}} = \eta\, E_0. \tag{I.20}$$

If $\sigma^{\text{trial}} \leq \sigma^{-}_{\text{env}}$ then set $\sigma \leftarrow \sigma^{-}_{\text{env}}$ and $E_t \leftarrow E^{-}_{\text{env}}$.

Clough pinching reloading rules (through-zero pinching). Let $\sigma_{\text{res}} = 0$ denote the reloading intercept (the origin). Define the "reversal-intercept" strains by

$$\varepsilon^{+}_{\text{res}} = \varepsilon_{rt} - \frac{\sigma_{rt} - \sigma_{\text{res}}}{E_0}, \qquad \varepsilon^{-}_{\text{res}} = \varepsilon_{rc} - \frac{\sigma_{rc} - \sigma_{\text{res}}}{E_0}. \tag{I.21}$$

Reloading toward tension ($\Delta\varepsilon \geq 0$). Let the final objective point as:

$$\sigma_{\max} = \max\Big(\sigma_y,\ \sigma_y + \eta\, E_0\big(\varepsilon_{\max} - \tfrac{\sigma_y}{E_0}\big)\Big). \tag{I.22}$$

If

$$\varepsilon^{+}_{\text{res}} \ \leq\ \varepsilon_{\max} - \frac{\sigma_{\max}}{E_0}, \tag{I.23}$$

define the reloading slope

$$\kappa^{+} = \min\left(\frac{\sigma_{\max} - \sigma_{\text{res}}}{\varepsilon_{\max} - \varepsilon^{+}_{\text{res}}},\ \beta E_0\right), \tag{I.24}$$

and the corresponding reloading line

$$\sigma^{+}_{\text{rel}}(\varepsilon^{+}) = \sigma_{\text{res}} + \kappa^{+}\,(\varepsilon^{+} - \varepsilon^{+}_{\text{res}}). \tag{I.25}$$

If $\sigma^{\text{trial}} > \sigma^{+}_{\text{rel}}$ then set $\sigma \leftarrow \sigma^{+}_{\text{rel}}$ and $E_t \leftarrow \kappa^{+}$.

Reloading toward compression ($\Delta\varepsilon < 0$). Let

$$\sigma_{\min} = \min\Big(-\sigma_y,\ -\sigma_y + \eta\, E_0\big(\varepsilon_{\min} + \tfrac{\sigma_y}{E_0}\big)\Big). \tag{I.26}$$

If

$$\varepsilon^{-}_{\text{res}} \ \geq\ \varepsilon_{\min} - \frac{\sigma_{\min}}{E_0}, \tag{I.27}$$

define

$$\kappa^{-} = \min\left(\frac{\sigma_{\min} - \sigma_{\text{res}}}{\varepsilon_{\min} - \varepsilon^{-}_{\text{res}}},\ \beta E_0\right), \quad \sigma^{-}_{\text{rel}}(\varepsilon^{+}) = \sigma_{\text{res}} + \kappa^{-}\,(\varepsilon^{+} - \varepsilon^{-}_{\text{res}}). \tag{I.28}$$

If $\sigma^{\text{trial}} < \sigma^{-}_{\text{rel}}$ then set $\sigma \leftarrow \sigma^{-}_{\text{rel}}$ and $E_t \leftarrow \kappa^{-}$.

State updates.

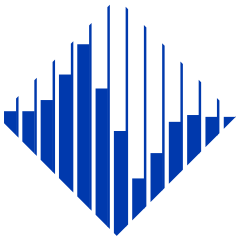
$$\begin{aligned} \varepsilon_{\max} &\leftarrow \max(\varepsilon_{\max},\, \varepsilon) \quad \text{(when in tension)}, \\ \varepsilon_{\min} &\leftarrow \min(\varepsilon_{\min},\, \varepsilon) \quad \text{(when in compression)}, \end{aligned} \tag{I.29}$$

and retain the most recent reversal points $(\varepsilon_{rt}, \sigma_{rt})$ and $(\varepsilon_{rc}, \sigma_{rc})$ as set above.

The Pacific Earthquake Engineering Research Center (PEER) is a multi-institutional research and education center with headquarters at the University of California, Berkeley. Investigators from over 20 universities, several consulting companies, and researchers at various state and federal government agencies contribute to research programs focused on performance-based earthquake engineering.

These research programs aim to identify and reduce the risks from major earthquakes to life safety and to the economy by including research in a wide variety of disciplines including structural and geotechnical engineering, geology/seismology, lifelines, transportation, architecture, economics, risk management, and public policy.

PEER is supported by federal, state, local, and regional agencies, together with industry partners.



PEER Core Institutions

University of California, Berkeley (Lead Institution)
California Institute of Technology
Oregon State University
Stanford University
University of California, Davis
University of California, Irvine
University of California, Los Angeles
University of California, San Diego
University of Nevada, Reno
University of Southern California
University of Washington

Pacific Earthquake Engineering Research Center
University of California, Berkeley
325 Davis Hall, Mail Code 1792
Berkeley, CA 94720-1792
Tel: 510-642-3437
Email: peer_center@berkeley.edu

ISSN 2770-8314
https://doi.org/10.55461/MDJO2682