IMAGE PROCESSING, ANALYSIS, AND MANAGEMENT TOOLS FOR GUSSET PLATE CONNECTIONS IN STEEL TRUSS BRIDGES

Final Report

PROJECT TPF-5(259)

Oregon Department of Transportation

IMAGE PROCESSING, ANALYSIS, AND MANAGEMENT TOOLS FOR GUSSET PLATE CONNECTIONS IN STEEL TRUSS BRIDGES

Final Report

PROJECT TPF-5(259)

by
Christopher Higgins, Ph.D., P.E.
Daniel Gillins, Ph.D., P.L.S.
Michael Scott, Ph.D.
Sinisa Todorovic, Ph.D.
Farid Javadnejad
Shravya Varakantham

for

Oregon Department of Transportation
Research Section
555 13th Street NE, Suite 1
Salem OR 97301

and

Federal Highway Administration
400 Seventh Street, SW
Washington, DC 20590-0003

and
The Departments of Transportation of California, Idaho, New York, North Carolina, Texas and Wisconsin

October 2016

Technical Report Documentation Page

1. Report No. FHWA-OR-RD-17-01	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Image Processing, Analysis, and Management Tools for Gusset Plate Connections in Steel Truss Bridges		5. Report Date October 2016 6. Performing Organization Code
7. Author(s) Christopher Higgins, Daniel Gillins, Michael Scott, Sinisa Todorovic, Farid Javadnejad, Shravya Varakantham		8. Performing Organization Report No. TPF-5(259)
9. Performing Organization Name and Address Oregon State University School of Civil and Construction Engineering 101 Kearney Hall Corvallis, OR 97331-3212		10. Work Unit No. (TRAIS) 11. Contract or Grant No.
12. Sponsoring Agency Name and Address Transportation Pooled Fund Program: Lead State: Oregon Dept. of Transportation Research Section and Federal Highway Admin. 555 13th Street NE, Suite 1 400 Seventh Street, SW Salem, OR 97301 Washington, DC 20590-0003 and: The Departments of Transportation of California, Idaho, New York, North Carolina, Texas and Wisconsin		13. Type of Report and Period Covered Final Report 14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This report details the research undertaken and software tools that were developed that enable digital images of gusset plates to be converted into orthophotos, establish physical dimensions, collect geometric information from them, and conduct specification-based and nonlinear finite element analyses of the connections. A software package, GussetImageProcessing.jar, was written that allows the user to process the images, remove the perspective, establish scale, identify the gusset plate boundaries, identify and group fasteners to truss members, and collect the geometric information to conduct AASHTO-LRFR connection ratings on the processed digital images. A separate software package, zGusset.tcl, uses the open-source program OpenSees as the analysis engine and is used to perform non-linear finite element analysis of a gusset plate model. To organize and query the collection of large numbers of high-fidelity quantitative images, a new approach was developed using Geographic Information System (GIS) to link images spatially across various geometric scales, temporally across inspection intervals, and to associate relevant metadata to the image view.

17. Key Words Gusset Plate, Steel Truss Bridge, Image Processing, finite element analysis, Geographic Information System	18. Distribution Statement Copies available from NTIS, and online at http://www.oregon.gov/ODOT/TD/TP_RES/

19. Security Classification (of this report) Unclassified	20. Security Classification (of this page) Unclassified	21. No. of Pages 119	22. Price

SI* (MODERN METRIC) CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
		AREA		
in^2	square inches	645.2	millimeters squared	mm^2
ft^2	square feet	0.093	meters squared	m^2
yd^2	square yards	0.836	meters squared	m^2
ac	acres	0.405	hectares	ha
mi^2	square miles	2.59	kilometers squared	km^2
		VOLUME		
fl oz	fluid ounces	29.57	milliliters	ml
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.028	meters cubed	m^3
yd^3	cubic yards	0.765	meters cubed	m^3
NOTE: Volumes greater than 1000 L shall be shown in m^3.				
		MASS		
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
		TEMPERATURE (exact)		
°F	Fahrenheit	(F-32)/1.8	Celsius	°C

APPROXIMATE CONVERSIONS FROM SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
m	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi
		AREA		
mm^2	millimeters squared	0.0016	square inches	in^2
m^2	meters squared	10.764	square feet	ft^2
m^2	meters squared	1.196	square yards	yd^2
ha	hectares	2.47	acres	ac
km^2	kilometers squared	0.386	square miles	mi^2
		VOLUME		
ml	milliliters	0.034	fluid ounces	fl oz
L	liters	0.264	gallons	gal
m^3	meters cubed	35.315	cubic feet	ft^3
m^3	meters cubed	1.308	cubic yards	yd^3
		MASS		
g	grams	0.035	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams	1.102	short tons (2000 lb)	T
		TEMPERATURE (exact)		
°C	Celsius	1.8C+32	Fahrenheit	°F

*SI is the symbol for the International System of Measurement

ACKNOWLEDGEMENTS

This research was funded through a pooled fund study led by the Oregon Department of Transportation. Contributors to the research funding were Federal Highway Administration, Wisconsin DOT, Idaho Transportation Department, Caltrans, Texas DOT, North Carolina DOT, and New York DOT. This support is gratefully acknowledged.

DISCLAIMER

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

GENERAL INTRODUCTION AND ORGANIZATION OF REPORT

After the collapse of the I-35W Bridge in Minnesota, inspection and rating of steel truss bridge gusset plate connections has become necessary for bridge owners to establish their adequacy. When trying to perform gusset plate connection ratings, transportation agencies have found a number of challenges. These include missing design drawings for the connection details, uncertain as-built geometries, time-consuming extraction of the connection geometry from drawings, and the need for more sophisticated analysis tools to predict connection strength. Currently, the state of the practice for collecting dimensional information of connections is to acquire the data manually by visual inspection and conventional instruments (most commonly using tape measures). Such procedures require knowledgeable staff and the results are expected to be quite variable. Further, this approach is time consuming and requires post-processing of field notes into scaled drawings and methods to close errors. To overcome these limitations, close-range photogrammetry is used.

To address many of these needs, a pooled fund project was undertaken. There were five main objectives in the project:

- Develop digital photographic methods to capture, correct perspective distortion, and measure connection dimensions.
- Extract geometric features of the plate and fasteners from processed images for analysis tasks.
- Implement AASHTO-LRFD connection capacity calculations from the processed images.
- Develop open-source nonlinear finite element analysis of gusset plates as an alternative analysis approach.
- Develop of framework for managing quantitative inspection images.

The research outcomes from these objectives provide an effective means to collect geometric data and analysis inputs that hasten the time consuming task of steel truss bridge connection rating. A key feature of this study is a technique to effectively rectify digital images from non-stationary imaging positions using conventional cameras to produce orthographic digital images of steel truss bridge gusset plate connections so physical dimensions can be extracted for use in connection evaluation. Production of large amounts of high-fidelity quantitative inspection images requires a new way to effectively manage this data. To address this need, a geographic information system (GIS) framework was used. It enables linking image data to other metadata including inspection reports, design drawings, material certifications, etc. to enhance bridge

management. The open-source finite element software provides rapid nonlinear behavior predictions for gusset plates under user prescribed loading combinations including sway-buckling response, member interactions, and imperfection sensitivity. The combined features of these tools provide methods and means that will expedite gusset plate rating.

There are three distinct pieces to the research and this report is organized into three parts.

Part 1 – Image Processing for Gusset Plates

Part 2- Finite Element Modeling and Analysis of Gusset Plates from Images

Part 3- Interactive Bridge Image Management (iBIM)

PART 1: IMAGE PROCESSING FOR GUSSET PLATES

Shravya Varakantham
Sinisa Todorovic, Ph.D.
And
Christopher Higgins, PH.D., P.E.

1.0 INTRODUCTION AND BACKGROUND

Digital close-range photogrammetry offers an opportunity to rapidly and accurately collect geometric information on gusset plates. It has been used previously for historic architecture documentation and bridge inspections. Most of the approaches to date require multiple stations to obtain dimensions. Large numbers of control points are used to correlate the images. Methodologies for using single images have been used to assess both 2D and 3D geometry when applying constraints such as straight lines and vanishing points *(Heuvel 1998, Criminisi 2000)*.

Mills and Barber *(Mills and Barber 2004)* suggest that in close-range digital photogrammetry is suitable for structural surveying and conducted a case study Hastings Tower located in Birmingham, U.K. The case study was conducted using terrestrial surveying, laser scanning, digital camera for multi-station convergent photogrammetry, and metric camera for stereo photogrammetry. The study showed that the digital multi-station convergent photogrammetry produced results of higher precision and accuracy than conventional stereo photogrammetry. Proper planning and location selections were important to obtain accurate measurements for each of methodologies.

Chandler et al. *(Chandler et al. 2005)* suggested low-cost digital cameras have the potential for use in close-range digital photogrammetric measurements after lens calibrations are performed. Chandler et al. created an experimental test model out of medium density fiberboard. They then generated a reference digital elevation model (DEM) of the test model using total station, a digital camera, and commercial imaging software. Three low-cost digital cameras were calibrated using an external calibration method and the commercial software. Images taken from the low-cost digital cameras generated digital elevation models that were compared to the reference DEM. The results of the study show that low-cost digital cameras generated DEMs close to the true DEM, however lens calibration was important for accurate results.

Arias et al. *(Arias et al. 2004)* suggest close-range photogrammetry techniques provide a simple and cost effective approach to documentation of agro-industrial buildings. Arias et al. created a model of a traditional agro-industrial building taken from multi-station setup of 2 plumb lines, 24 camera positions and 27 control points. The data obtained a root-mean-square error of 9 mm for their model.

Jáuregui et al. *(Jáuregui et al. 2006)* arranged a multi-station setup of large grid of targets, scale bars, and cameras. The maximum error between photogrammetric values and hand measurements was reported as 1.43%.

Rodríguez et al. *(Rodríguez et al. 2008)* demonstrated the capability of taking 3 digital photographs and distance measurements from a mounted position of a flat element and obtaining the dimensions. This was done by finding the relative position of the camera to the flat element.

Tommaselli and Reiss *(Tommaselli and Reiss 2005)* determined the dimension of a billboard using a mathematical model. The linear horizontal and vertical lines on the billboard are used to

create a rotation matrix from an indirect method by applying constraints based on geometric relationships. The point coordinates then can be determined by inverse collinearity equations.

Lee and Chang *(Lee and Chang 2005)* used digital image processing for the assessment of rust on steel bridges. Liu et al. *(Liu et al. 2006)* utilized image processing methods to detect rivets for aircraft lap joints.

Although using digital image processing to detect a crack on a concrete surface is difficult due to voids, blemishes, shading, and shapes of cracks, it has attracted broad interest and been studied by several researches such as Ito et al. *(Ito et al. 2002)*, Dare et al. *(Dare et al. 2002)*, Fujita et al. *(Fujita et al. 2006)*, Yamaguchi and Hashimoto *(Yamaguchi and Hashimoto 2006)*, Yamaguchi et al. *(Fujita et al. 2008)*, and Yamaguchi and Hashimoto *(Yamaguchi and Hashimoto 2009)*.

Based on review of the literature, the present research differs from previous work because for evaluation of steel truss bridge gusset plate connections, images cannot easily be taken from a mounted stationary position. In the field, photographs of connections will likely be taken from a snooper with both the snooper and bridge in motion from wind and traffic or by climbing on the structure. Thus, it will not be feasible to obtain stationary positions to correlate stereo or multi-station images. To rapidly setup and collect the geometric data, control points are established using a reusable magnetic reference target placed on the gusset plate. An image rectification and metrification methodology was developed and implemented in a software program to allow a non-specialized user to process the captured images and collect the data necessary for rating steel gusset connections. This approach significantly reduces the amount of time needed in the field to acquire the dimensional data and the resulting fidelity is as good as or better than manually collected and recorded data. As a result of the proposed methodology, only the thickness of the gusset plates and a single dimension need to be directly field measured to enable collect the required gusset plate data. This significantly reduces the time required in the field for data collection.

2.0 METHODOLOGY

The methodology relies on digital photographs taken of a real world image (gusset plate) and placed on a two-dimensional image plane (camera sensor). Typically when an image is captured with a camera and lens, it contains perspective distortion (parallel lines converging at a finite point), as well as other distortions due to the lens characteristics (most commonly barrel or pin-cushion distortions). To remove the perspective, the image is rectified using a mathematical transformation which maps elements in the real world image to the photographic image plane. To remove barrel distortion or pin-cushion, either flat field lenses must be deployed or lens correction parameters can be used to post-process images. For the present case, barrel distortion and pin-cushion are minimized using flat field lenses that minimize these distortions.

When features of the real world image correspond generally to a single plane, as in the case of gusset plate connections, a two-dimensional correspondence can be used to rectify the image. This simplifies the transformation. One of the most common techniques for image rectification is the direct linear transformation (DLT) algorithm. To perform the transformation, point correspondences in the real world image are mapped to pixel locations on photographic image plane. When the geometrical features in the real world are established to provide a true dimensional scale, the resulting transformation can not only be rectified, it can be scaled to the real world dimensions. This permits measurements to be made from the rectified image. To implement the DLT transformation, the image plane and real world planes are assumed to be homogeneous. A transformation matrix H, is determined to satisfy the following equation for each control point, *i*:

$$H \bullet \begin{bmatrix} X_i' \\ Y_i' \\ 1 \end{bmatrix} = \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix} \qquad \textbf{(2.1)}$$

where X' and Y' are photograph control point coordinates, X and Y are real world control point coordinates, and H is a 3x3 matrix. A singular value decomposition (SVD) is used to solve for the H matrix given the correlated pixel and real-world coordinates. A minimum of 4 control points are needed (x and y coordinates each provide a degree-of-freedom (DOF) for a total of 8 DOFs and 1 more is provided by scale). If more points are available, error estimations can be made, as described subsequently. The method generates a two-dimensional transformation based solely on the control point coordinates. As such, camera parameters (calibration) are not needed. Such DLT algorithms are used in commercial photogrammetric software such as Orthorpro.

As stated previously, the DLT requires known control points be established in the real-world image that can be captured in the image plane. For the present approach, reference targets were developed that establish nine (9) real-world control points (over-constraining the transformation matrix). To rapidly deploy the control points on the gusset plate, reference targets were made to

be magnetic. Two target sizes were produced. One is relatively small (8 in.) and the other is relatively large (24 in.) as seen in Figure 2.1. The small reference target is used for smaller gusset plates or for splice plates between chord members and the large reference target is used for most typical large gusset plate connections. The targets are made from 1x0.25 in. aircraft aluminum bar stock. The material was precisely machined and round niobium disc magnets were pressed into the bars. These act as some of the nine (9) control points. The remaining control points are black machined circles, 0.5 in. diameter and are distributed across the target arms. Spacing of the control points is 2 in. and 6 in., for small and large targets, respectively. The arms of the target are orthogonal to one another to spread the control points across the image. The material thickness (0.25 in.) of the reference target allows it to be sufficiently close to the gusset plate surface so that offset correction is not required (a large arbitrary standoff distance can be prescribed ~100000 ft). The large target can be disassembled to allow for easier transport in the field.

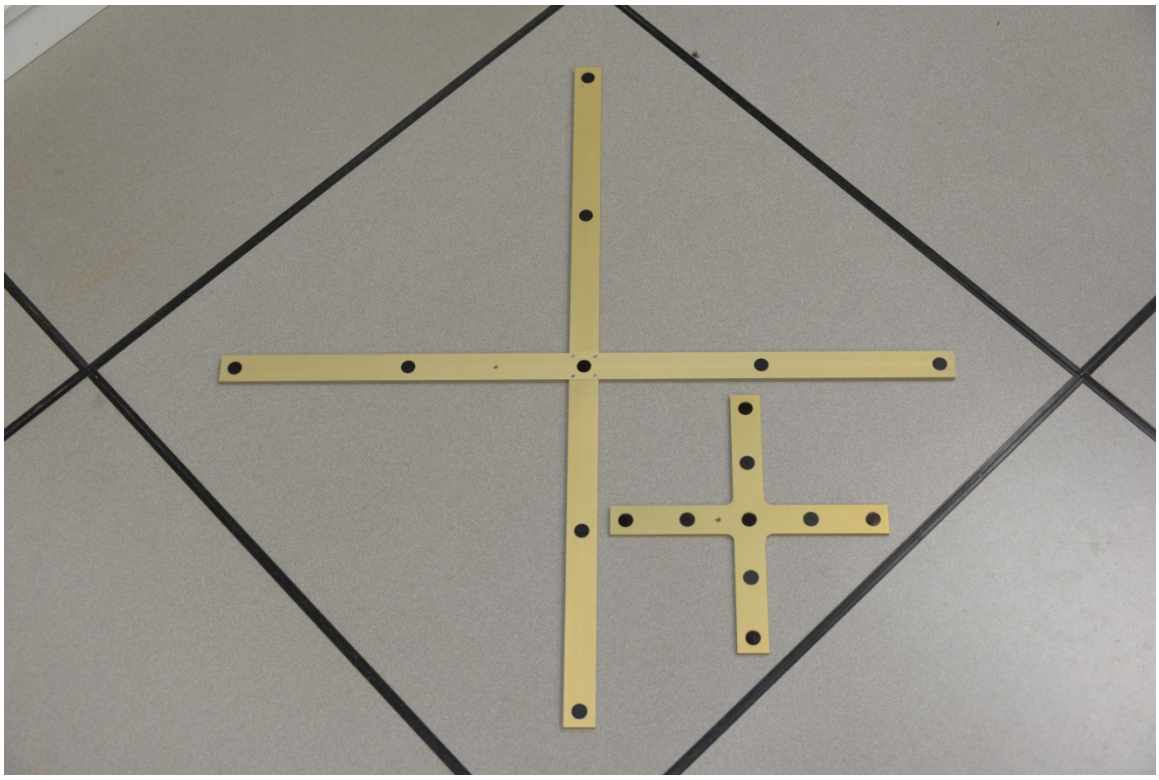


Figure 2.1: 24 in. and 8 in. reference targets (defined as “New” targets in software)

2.1 ERROR ESTIMATION OF MEASUREMENTS

Error is introduced by camera and reference target imperfections, movement as the photograph is taken, and during the control point selection. The nine (9) control points on the reference targets are used to determine the measurement error in the rectified image as the sample size for the error calculations. The root-mean-square error (RMSE) and standard deviation error (σ) are obtained from the difference in the predicted coordinates calculated from the H-matrix and their true coordinates as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\theta_i' - \theta_i)^2} \quad \textbf{(2.2)}$$

where θ_i' is the predicted value and θ_i is the true value of the ith control point, and n is the number of control points;

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(e_i - \bar{e})^2} \quad \textbf{(2.3)}$$

where e_i is the location error of the ith point, $\bar{e}$ is the average location error, and n is the number of control points. These are determined from the nine (9) known control points established using the reference target in the image.

The error is assumed to be normally distributed and a 99.7% confidence interval was selected which corresponds to three standard deviations. The pixel scale depends on the number of pixels per inch (real scale) on the original digital photograph and calculated based on the smallest number of pixles across the known length between the outermost points on the two target arms. The predicted location uncertainty with 99.7% confidence for a single point in the rectified orthographic image is estimated to be within the sum of the RMSE, three standard deviations, and one pixel size. The one pixel size accounts for user selection uncertainty. For measurement of a line length within the bounds of the reference target, the measurement error would be twice that from a single point location uncertainty and computed as:

$$E_{measurement} = 2 \times \left(RMSE + 3\sigma + pixel \right) \quad \textbf{(2.4)}$$

These results are provided in reference files for each of the images that are processed using the software tool developed in this research. The implementation of the rectification and other processing steps used in the software tool are described below.

3.0 IMPLEMENTATION

The image rectification and scaling is implemented as the first major step in a software tool that includes other processing features that use the images as the input for gusset plate analysis and rating tasks. The software tool and its image processing features are described in this chapter. Each section encapsulates the problem statement associated with each feature, the associated challenges, and the solution description. Additional relevant literature is distributed for each task, presented in corresponding sections and is cross-referenced. Appendix A presents a description of graphical user interface of the software tool created to process gusset plate images. The associated source code and video documentation are available on-line.

3.1 SOFTWARE TOOL OVERVIEW – GUSSET PLATE IMAGE PROCESSING TOOL (GPIPT)

A software tool was developed to process and analyze gusset plate images. It includes both automated and manual features: automated to speed processing time and manual for user direct processing. The user is required to provide a set of images containing the reference target in the GPIPT at the beginning of the process. The tool then guides the user through a series of processing stages. At each stage, the user can monitor the results and correct them. He/she can also change settings to recalculate for improved results. Results from each stage act as inputs to the next. The following list describes the stages that must be performed in order:

1. Reference target detection and perspective removal
2. Boundary marking
3. Fastener detection
4. Fastener grouping
5. Connection analysis

Given an image of a gusset plate containing the reference target (large or small), the first step involves removing the perspective as described in the previous chapter. In the next step, the boundary of the gusset plate is defined without the connecting truss members to allow the software to focus only on the region of interest. This is followed by the detection of fastener locations (bolts or rivets). The detected fasteners are then grouped or assigned to the associated truss members. In the analysis phase, a series of AASHTO-LRFR connection strength calculations are performed interactively including the fastener shear strength, bearing resistance at fastener holes, Whitmore and block shear checks, compression checks for each connected member and shear plane checks of the gusset plate. The final results are an input file for nonlinear finite element analysis (Detailed in Part 2 of this report) and output files reporting the AASHTO-LRFR strength limit state calculations, as well as metadata from the image process.

3.2 TASK 1 – TARGET DETECTION AND PERSPECTIVE REMOVAL

One of the reference targets, described previously, must be placed on the gusset plate prior to capturing the image. Images of gusset plates with targets will most often be taken by inspectors in the field from snoopers or by climbing the structures and the camera position will be arbitrary (images may not have the gusset plate orthogonal to the plates). Use of the reference target allows removal of perspective without camera calibration and establishes a physical scale in the digital picture in these conditions.

Upon uploading a gusset plate image containing an reference target, the target first must be detected and the nine (9) control points must be located. The user must specify which target size is used in the image.

3.2.1 Problem Statement – Target Detection and Perspective Removal

In order to correct a given image, all the target control points on the reference target must be determined. Examples of the target control points under different camera views are shown in Fig. 3.1.

Figure 3.1: Examples of target control points taken from different camera views

3.2.2 Literature Review – Target Detection and Perspective Removal

Template matching is a fundamental task used in many image analysis applications and is used in the present work to identify the nine (9) reference target control points. Template matching is the process of finding the location of a sub-image, called a template, within the larger image. There are number of methods for performing image registration. Once a number of corresponding templates are found, their centers are used as corresponding control points to determine the registration parameters. Template matching involves comparing a given template with windows of the same size in an image and identifying the window that is most similar to the template *(Di Stefano et al. 2003)*. This section discusses the template matching approach used for matching a small image which is a part of bigger image. The template matching process has two possible approaches. 1) feature-based of 2) template-based *(Brunelli and Poggio 1993)*.

3.2.2.1 Template Matching Using Normalized Cross Correlation

If the template image has strong features, a feature-based approach may be considered; the approach may prove further useful if the match in the search image can be transformed in some fashion. Since this approach does not consider the entirety of the template image, it can be more computationally efficient when working with source images of larger resolution. The alternative approach, template-based, may require

searching potentially large amounts of points in order to determine the best matching location.

For templates without strong features, or for when the most of the template image constitutes the matching image, a template-based approach may be effective. As aforementioned, since template-based template matching may potentially require sampling of a large number of points, it is possible to reduce the number of sampling points by reducing the resolution of the search and template images by the same factor and performing the operation on the resultant downsized images (multiple resolution, or pyramid, image processing), providing a search window of data points within the search image so that the template does not have to search every viable data point, or a combination of both.

Given a template t, whose position is to be determined in an image f, the basic idea of the algorithm is to represent the template, for which the normalized cross correlation is calculated, as a sum of rectangular basis functions. Then the correlation is calculated for each basis function instead of the whole template. The result of the correlation of the template t and the image f is obtained as the weighted sum of the correlation functions of the basic functions. Depending on the approximation, the algorithm can outperform Fourier-transform based implementations of the normalized cross correlation algorithm and it is especially suited to problems, where many different templates are to be found in the same image f.

3.2.2.2 Normalized Cross-Correlation

Normalized cross-correlation has found application in a broad range of computer vision tasks such as stereo vision, motion tracking, image mosaic-ing, etc. Normalized cross-correlation is the simplest but also an effective method as a similarity measure, which is invariant to linear brightness and contrast variations. Straightforward hardware implementation makes it useful for real-time applications *(Brunelli and Poggio 1993; Goshtasby et al. 1984)*.

For image-processing applications in which the brightness of the image and template can vary due to lighting and exposure conditions, the images can be first normalized. This is typically done at every step by subtracting the mean and dividing by the standard deviation. That is, the cross-correlation of a template, $t(x,y)$ with a sub-image $f(x,y)$ is:

$$\frac{1}{n}\sum_{x,y}\frac{(f(x,y)-\overline{f})(t(x,y)-\overline{t})}{\sigma_f\sigma_t}=\frac{\frac{1}{n}\sum_{x,y}f(x,y)t(x,y)-\overline{f}\overline{t}}{\sigma_f\sigma_t} \qquad \textbf{(3.1)}$$

where n is the number of pixels in $t(x,y)$ and $f(x,y)$, $\overline{f}$ is the average of f and σ_f is standard deviation of f. In functional analysis terms, this can be thought of as the dot product of two normalized vectors. That is, if $F(x,y)=f(x,y)-\overline{f}$ and $T(x,y)=t(x,y)-\overline{t}$ then the above sum is equal to $\left\langle \frac{F}{\|F\|}, \frac{T}{\|T\|} \right\rangle$ where $\langle\cdot,\cdot\rangle$ is the inner product and $\|\cdot\|$ is the L^2 norm. Thus, if f and t are real matrices, their normalized cross-

correlation equals the cosine of the angle between the unit vectors F and T, being thus 1 if and only if F equals T multiplied by a positive scalar.

3.2.3 Solution Approach – Target Detection and Perspective Removal

Template matching using a set of diverse target control points is a common approach as described above. However, owing to the wide variation of different scales and rotations of the template target control points and the need for precision in the output, the approach taken in the tool requests an initial input from the user and the rest of the target control points are identified based on this user selection using template matching.

Each gusset plate image is displayed to the user for marking the top-most control point on the reference target. In order to avoid faulty selections possibly induced by the human error, the user is requested to click on the center of the disk and anywhere outside the disk perimeter to identify the center location and disk radius.

The selected target disk is marked and is extracted from the image to act as a template/filter mask. This extraction is saved physically to the templates folder for further use. Given the top target control point, the extent of the target is calculated and noted based on the radius of the disk and the actual dimensions of the target arms. To prevent the possibility of losing image data containing the reference target, a sufficient buffer is induced in all directions around the target location. The image is then temporarily modified to focus only in the target section.

Using the template mask, template matching is performed on this image area containing only the reference target. The top nine (9) pixels with the highest similarity values are recorded from the output generated and are recorded as the target disk centers. These centers are marked on the image and the results are displayed to the user for approval or editing. The user can then edit the detection by adding or removing targets if unsatisfied. This approach requires user feedback making it semi-automatic. However, as compared to the procedure involving the user to select all target control points manually, this semi-automatic approach reducing the user effort and time. An example image with the 9 control points identified is shown in Figure 3.2.

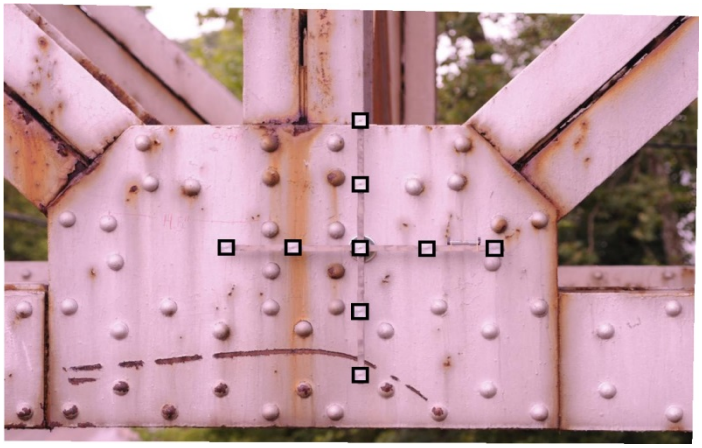


Figure 3.2: An example of a gusset plate connection with targets detected by GPIPT

Given the control point centers for all the nine (9) locations on the reference target and the actual target dimensions, the image is corrected by computing the transformation matrix first for the nine (9) control points and then extending the same for each pixel of the original distorted image. The *Image Processing Toolbox* of MATLAB includes an extensive set of spatial transformation matrix computation functions *(MATLAB 2013)* and these functions are implemented in the software tool. The output image from this step is the corrected image without perspective and with physical scale (pixel dimensions become inches). Examples of the perspective removal, before and after, are shown in Figure 3.3. In addition to displaying the rectified image, the resulting scale is also shown to the user after the transformation. The usual application of perspective removal in digital image processing results in a perspective that is controlled based on appearance. However, given the need for high accuracy in the present application, results are required to be precise to the pixel level.

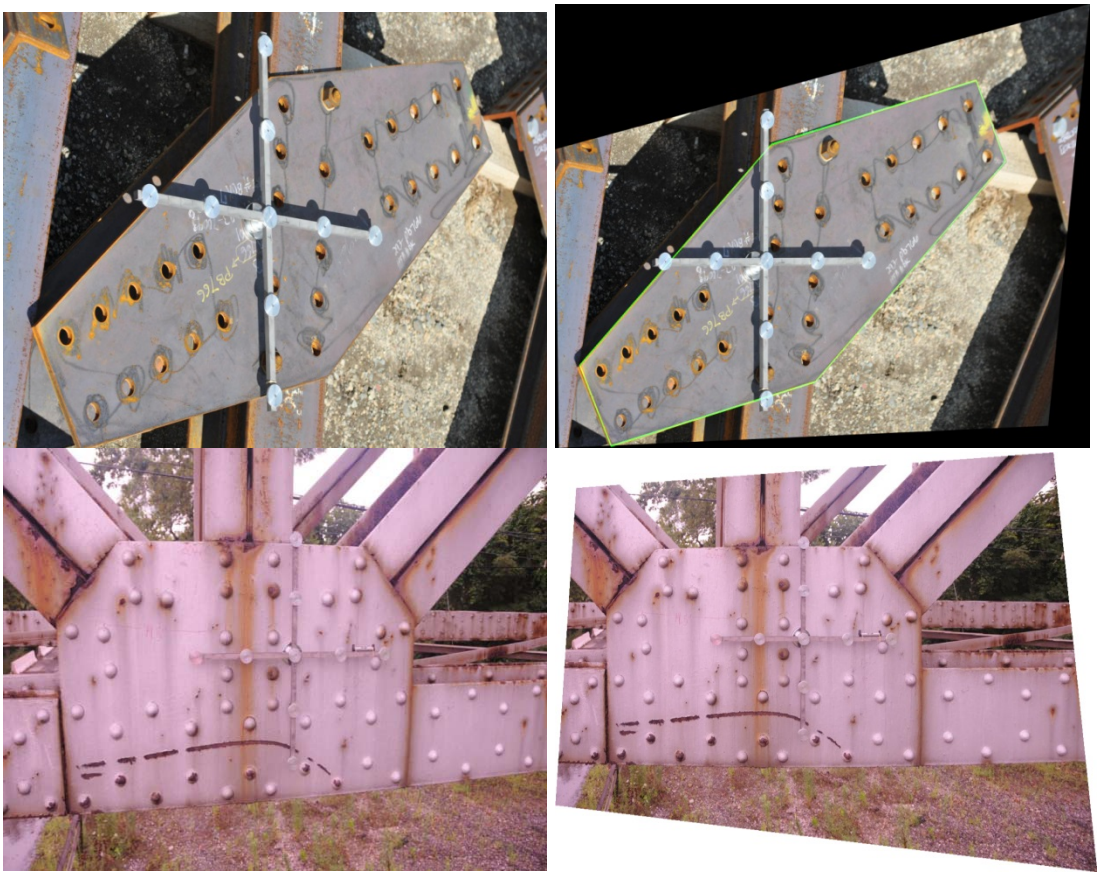


Figure 3.3: Examples of gusset plates before (left images) and after (right images) the perspective is removed by GPIPT

To assess the outcome of the transformation and the image scale, a set of tools called digital calipers becomes available to the user. Known physical dimensions on the gusset plate can be compared with measured values from the transformed image. If the scale is not confirmed to the satisfaction of the user, the user can repeat the process. Alternatively, if the transformation is adequate but the scale is not, the user can impose a scale in terms of pixels per inch for each of the axis coordinates. The ability to impose scale also allows use of scanned or other electronic drawings (including shop drawings with different scales on the ordinate and abscissa) to be used

in the subsequent analysis tasks rather than photographs. The imposed scale and digital calipers can be used repeatedly to converge image measurements to the known physical measurements.

3.2.4 Challenges – Target Detection and Perspective Removal

Here the challenges encountered in making the target detection process completely automated are presented.

3.2.4.1 Fasteners are Detected as Target Control Points

In an attempt to automatically detect target control points, when template matching is performed with a bank of different target templates, fasteners are often detected as target disks. There are various factors causing this phenomenon.

- Dimensions of the targets unknown: Targets can be of different shapes and sizes in the template bank relative to the disks in the image.

- Color and orientation varies: Each gusset plate is of a different color and depending upon the time of the image taken, the color of the target changes resulting in difficulty determining the control points.

Added to this, given the perspective of the image, the target disks do not necessarily face frontal and at times may also have strong perspective.

3.2.4.2 Extreme Precision Needed

As the determination of target control point coordinates provide perspective correction of the image and establish scale, a high degree of precision is demanded to ensure subsequent tasks that use the measured pixel distances as the actual scale. Hence, automatic target control point detection without user input and feedback might not provide the level of precision required for subsequent analysis tasks.

3.3 TASK 2 – BOUNDARY DETECTION

The gusset plate acts as the connection between the truss members. Hence every image of a gusset plate typically has the plate in the image center along with the attached truss members extending beyond to the edges. These members complicate the image processing, do not contain relevant data, and increase the computational time. Further, they are not needed for the subsequent analysis tasks. However, Identification of the gusset plate boundary is essential to establish the overall geometry of the plate, the edge distances for fasteners, and the shear planes.

3.3.1 Problem Statement – Boundary Detection

The extents of the gusset plate boundary are required from the image and need to be stored for subsequent tasks. A rectified and scaled gusset plate image is the prerequisite for the boundary detection phase.

3.3.2 Solution Approach – Boundary Detection

To automate boundary detection, several alternative approaches were pursued. These were all computationally expensive (required a long processing time) and produced inconsistent results over the range of images used in the calibration. For completeness and considering the significant effort expended to automate this task, the details of this approach are reported in Appendix A. Given the wide range of backgrounds, imperceptible boundaries between the gusset plate and the attached truss members, varying plate contrasts, colors, and field discontinuities, the most direct and least time consuming approach relied on the user manually selecting the gusset plate boundary using a click and drag tool developed for this purpose. After establishing the boundary, the user can accept the selection or redo it until satisfied with the result.

3.4 TASK 3 – FASTENER DETECTION

Fasteners on the gusset plate can be of two different types: bolts or rivets. A rivet is characterized by a smooth spherical head while a bolt consists of a hexagonal nut typically threaded onto the stick-out end of the fastener with a washer at the interface between the nut and plate. Examples of different fastener types are shown in Figures 3.4 and 3.5.

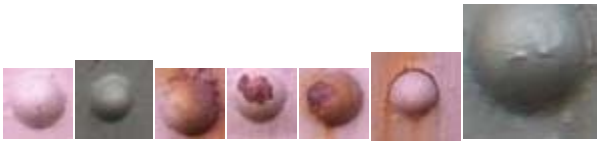


Figure 3.4: Examples of rivets on a gusset plate

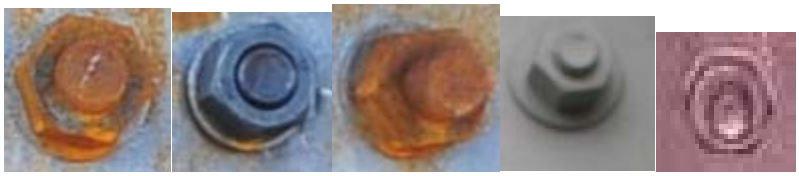


Figure 3.5: Examples of bolts on a gusset plate

3.4.1 Problem Statement – Fastener Detection

The precise coordinate locations of the all the fastener centers are required from the image and need to be stored for subsequent tasks. A rectified and scaled gusset plate image with the plate boundary identified are the prerequisites for the fastener detection phase.

3.4.2 Literature Review – Fastener Detection

The fastener detection process on the image involves searching for a specific pattern of pixel arrangements in the image. This is called template matching. This is the same concept as described previously using normalized cross correlation described earlier in the section 4.2.2.

3.4.2.1 Finding Circular Shaped Objects in an Image

Finding circles uses a Circular Hough Transform (CHT) based algorithm for finding circles in images. This approach is used because of its robustness in the presence of

noise, occlusion and varying illumination. The CHT is not a rigorously specified algorithm; rather there are a number of different approaches that can be taken in its implementation. However, there are three essential steps which are common to all.

3.4.2.2 Accumulator Array Computation

Foreground pixels of high gradient are designated as being candidate pixels and are allowed to cast ‘votes' in the accumulator array. In a standard CHT implementation, the candidate pixels vote in pattern around them that forms a full circle of a fixed radius. Figure 3.6 shows an example of a candidate pixel lying on an actual circle (solid circle) and the classical CHT voting pattern (dashed circles) for the candidate pixel.

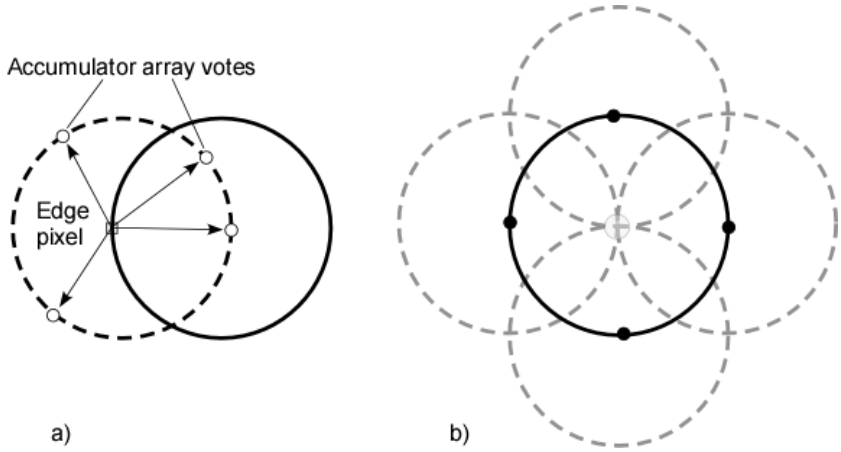


Figure 3.6: Classical CHT voting patterns a) accumulator array voting and b) center estimation.

3.4.2.3 Center Estimation

The votes of candidate pixels belonging to an image circle tend to accumulate at the accumulator array bin corresponding to the circle's center. Therefore, the circle centers are estimated by detecting the peaks in the accumulator array. Figure 3.6b shows an example of the candidate pixels (solid dots) lying on an actual circle (solid circle), and their voting patterns (dashed circles) which coincide at the center of the actual circle.

3.4.2.4 Radius Estimation

If the same accumulator array is used for more than one radius value, as is commonly done in CHT algorithms, radii of the detected circles have to be estimated as a separate step. The Matlab function *imfindcircles* provides two algorithms for finding circles in images: Phase-Coding (default) and Two-Stage. Both share some common computational steps, but each has its own unique aspects as well. The common computational features shared by both algorithms are as follows:

2-D Accumulator Array:

The classical Hough Transform requires a 3-D array for storing votes for multiple radii, which results in large storage requirements and long processing times. Both the Phase-Coding and Two-Stage methods solve this problem by using a single 2-D accumulator

array for all the radii. Although this approach requires an additional step of radius estimation, the overall computational load is typically lower, especially when working over large radius range. This is a widely adopted practice in modern CHT implementations.

Use of Edge Pixels: Overall memory requirements and speed is strongly governed by the number of candidate pixels. To limit their number, the gradient magnitude of the input image is threshold so that only pixels of high gradient are included in tallying votes.

Use of Edge Orientation Information: Another way to optimize performance is to restrict the number of bins available to candidate pixels. This is accomplished by utilizing locally available edge information to only permit voting in a limited interval along direction of the gradient as below. The two CHT methods employed by the Matlab function *imfindcircles* fundamentally differ in the manner by which the circle radii are computed.

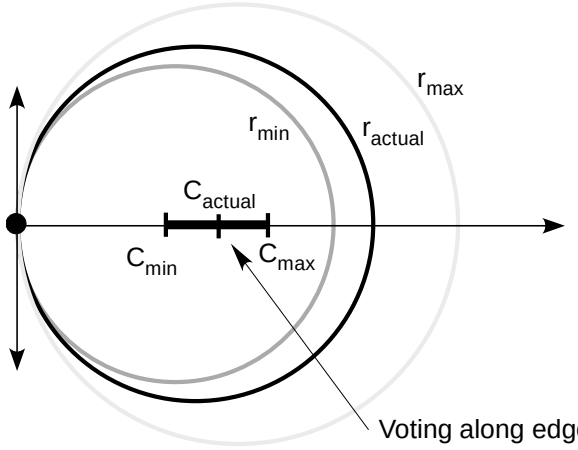


Figure 3.7: Voting mode: multiple radii, along direction of the gradient.

Two-Stage: Radii are explicitly estimated utilizing the estimated circle centers along with image information. The technique is based on computing radial histograms.

Phase-Coding: The key idea in Phase Coding is the use of complex values in the accumulator array with the radius information encoded in the phase of the array entries. The votes cast by the edge pixels contain information not only about the possible center locations but also about the radius of the circle associated with the center location. Unlike the Two-Stage method where radius has to be estimated explicitly using radial histograms, in Phase Coding, the radius can be estimated by simply decoding the phase information from the estimated center location in the accumulator array.

3.4.3 Solution Approach

As described above, one regular technique to detect fasteners on the image is to have template matching performed with a saved, diversified set of fastener templates on each image. However, the execution of such an approach results in a poor output due to the variation in many parameters at different levels. The following variations that impact the results include: different

intensity levels, sizes, brightness levels, fastener shapes, and types (bolts or rivets). In order to overcome these, the solution was to extract a few local fasteners from each image and then attempt to perform template matching strategy using high sensitivity. This method was the most promising because the fasteners extracted from the image possess similar intensity, size, and shape as the unknown fasteners in the search area. Using these detections as hard templates, the rest of the fasteners in the image are located using template matching with increased sensitivity.

The fastener detection algorithm takes as input a corrected image with the gusset plate boundary marked. The image is then converted to gray scale and an excerpt is taken from the center of the image. The most repeated value of the pixels in the image is chosen and all the pixels with the value that fall in the range of the most repeated value are chosen for further processing. If the image is smaller than the usual default area excerpted from the image, the most repeated value would be 0. If such is the case, to compensate for it, the excerpt of the image is made smaller in loop until the most repeated value is non zero. On the image that is now selected, a few morphological operations are performed such as filling the image and then executing the disk erosion so as to amplify the fastener pixels. Circles are then found on the image and saved to the metadata associated with each image in MATLAB. With the recorded radii, all the fasteners centered on the pixel list are extracted and saved to a folder. Using these as templates, convolution is performed on the edge output of the image with the edge output of each fastener template. Canny edge detector is used to compute these edge outputs. The convolution output returns the value at each pixel corresponding to the extent of the match with the template in the image. With a predetermined threshold, all the pixels that hold value greater than the threshold are recorded as the probable matches with the particular fastener template in the normalized convolution output.

All the pixel coordinates for each template are recorded and stacked with the rest of the templates. Duplicates and pixels with overlapping fastener radii range are removed. In order to suppress the cases of having multiple detections at close adjacent pixels, a mean shift cluster algorithm is performed that generates the mean point of all the pixels which are closer than a set threshold. At each point, the thresholds are adjusted to avoid the clearly false detections. That is, the cases where the detections exceed a set limit.

The user can manually add additional templates and rerun the detection algorithm. To help the user identify which fasteners were used as templates, they are marked with a dark colored border (black). Fasteners that were detected by the algorithm are marked with a light colored border (yellow) as shown in Figure 3.8.

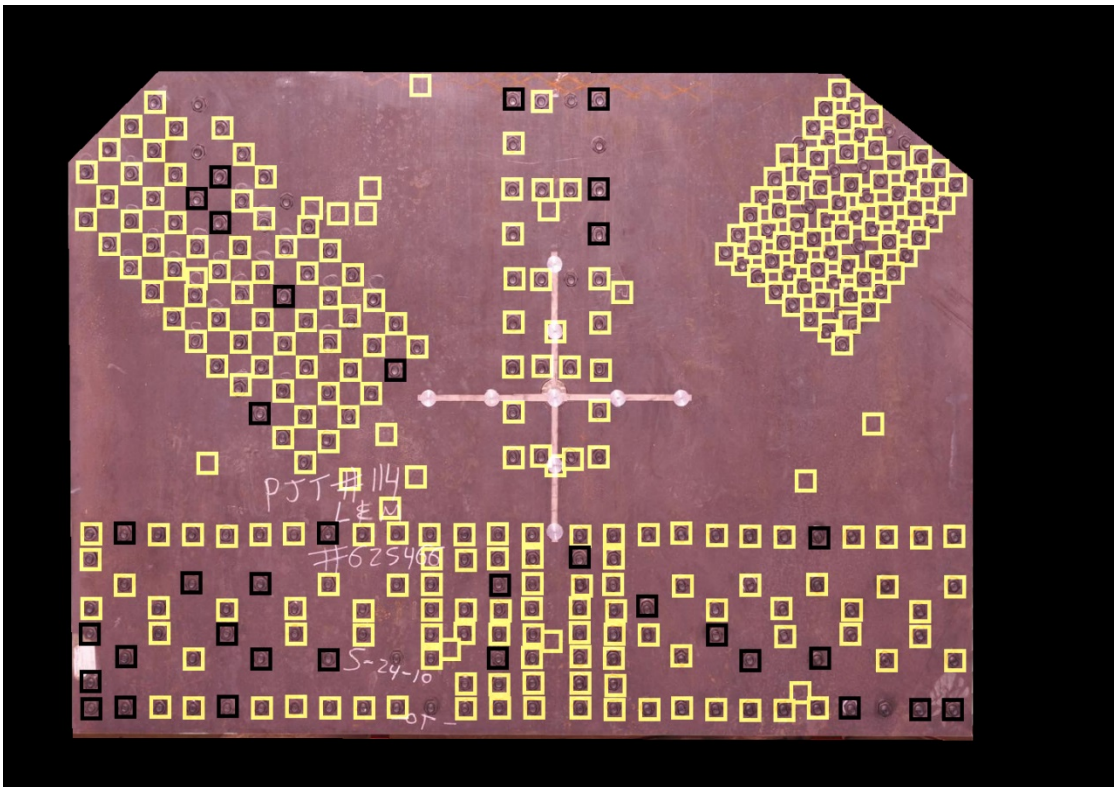


Figure 3.8: Example of a gusset plate with fasteners detected. Dark bordered fasteners act as the templates (extracted as circles from the image) and light bordered fasteners are detected using template matching (note there are false positives identified in the image that will need to be manually removed)

3.4.4 Challenges

There are a number of challenges to automating detection of fasteners in the gusset plate image, as detailed below.

3.4.4.1 Surface Stains on the Gusset Plates Identified as Fasteners

Gusset plates commonly exhibit rust staining in the field of the plates, around fastener locations, and at plate edges. These rust marks may falsely be detected as circles and hence as fastener templates. Such detections cannot automatically be corrected and thus the user needs to confirm detections and remove mis-identified fasteners.

3.4.4.2 Target Control Points Identified as Fasteners

Detecting fasteners that are around the same radii as the target control points sometimes results in the targets being detected as fasteners. This can be avoided by computing the new coordinates of the target centers and removing them from the fastener list.

3.4.4.3 Other Surface Defects on the Gusset Plates Identified as Fasteners

Some gusset plates have written notes, mismatched paint, and other types of staining on the surface. Such surface defects can sometimes be miscalculated as fasteners. Such detections need to be removed by the user interactively.

3.4.4.4 Fastener Shadows – False Positives

Gusset plates imaged in bright daylight can lead to dark shadows from the fasteners. Due to the high intensity difference, usually these shadows are interpreted as fastener circles. Such detections need to be removed by the user interactively.

3.4.4.5 Fasteners with Perspective Not Identified as Fasteners

Gusset plates have fasteners spanning across the entire plate and close to the edges. Images of these plates with strong initial perspective have edge fasteners that look quite different than those in the center, although in actuality they are similar. Even after the perspective removal for the plate, the fastener perspective is not completely lost. Hence, detection misses usually include fasteners at the edges of the gusset plates.

For all the challenges described above, the fastener detection phase is made semi-automatic in the GPIPT. Each fastener circle recorded on the image is saved and used in computing the rest of the fasteners with increased sensitivity. However, a single wrong fastener template may lead to additional false detections so corrections should be made by the user during the correction phase. In this way, the user can remove the incorrect template fasteners, add additional correct templates, and then rerun the detection for improved results. In addition, the user can manually add fasteners that were not detected by the template matching scheme.

3.5 TASK 4 – FASTENER GROUPING

The truss member forces flow through the fasteners that are used to join them to the gusset plate connection. Thus, to rate the connection, all the fasteners in the image need to be associated to one of the truss members. The exceptions to this are fasteners used to connect edge stiffening angles, which are not treated in the GPIPT and are neglected.

3.5.1 Problem Statement

All fasteners need to be associated with the truss member to which they are connected. No single fastener is to be simultaneously grouped into multiple groups. Also, no fastener should be left orphaned or ungrouped. The image with fasteners detected acts as a perquisite to the fastener grouping phase.

3.5.2 Literature Review

The solution to automatically assigning fasteners to the associated member centers on the concepts of clustering, which are detailed here.

3.5.2.1 Clustering

Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters). There are many kinds of clustering algorithms available. Hierarchical and Partition-based clustering are among them. The most appropriate clustering algorithm for a particular problem often needs to be chosen experimentally, unless there is a mathematical reason to prefer one cluster model over another. It should be noted that an algorithm that is designed for one kind of model has no chance on a data set that contains a radically different kind of model.

3.5.2.2 Hierarchical Clustering

Connectivity based clustering, also known as hierarchical clustering, is based on the idea that objects are more closely related to nearby objects than to objects farther away. These algorithms connect "objects" to form "clusters" based on their distance. A cluster can be described by the maximum distance needed to connect parts of the cluster. At different distances, different clusters will form, which can be represented using a dendrogram, which explains where the common name "hierarchical clustering" comes from: these algorithms do not provide a single partitioning of the data set, but instead provide an extensive hierarchy of clusters that merge with each other at certain distances. In a dendrogram, the y-axis marks the distance at which the clusters merge, while the objects are placed along the x-axis such that the clusters don't mix.

Connectivity based clustering is a whole family of methods that differ by the way distances are computed. Apart from the usual choice of distance functions, the user also needs to decide on the linkage criterion (since a cluster consists of multiple objects, there are multiple candidates to compute the distance to) to use. Popular choices are known as single-linkage clustering (the minimum of object distances), complete linkage clustering (the maximum of object distances) or UPGMA ("Unweighted Pair Group Method with Arithmetic Mean", also known as average linkage clustering). Additionally, hierarchical clustering can be agglomerative (starting with single elements and aggregating them into clusters) or divisive (starting with the complete data set and dividing it into partitions).

These methods will not produce a unique partitioning of the data set, but a hierarchy from which the user still needs to choose appropriate clusters. They are not very robust towards outliers, which will either show up as additional clusters or even cause other clusters to merge (known as "chaining phenomenon", in particular with single-linkage clustering). In the general case, the complexity is $\mathcal{O}(n^3)$, which makes them too slow for large data sets. For some special cases, optimal efficient methods (of complexity $\mathcal{O}(n^2)$) are known: SLINK for single-linkage and CLINK for complete-linkage clustering. In the data mining community, these methods are recognized as a theoretical foundation of cluster analysis, but now often considered obsolete. They were an inspiration for many later methods such as density based clustering.

Advantages of hierarchical clustering are that the algorithm doesn't need to know either the number of clusters or any random seed coordinates marking the locations of clusters. Disadvantages are that the output generated using this algorithm works only in the cases of clusters that are tightly bound. Single and complete-link clustering are two hierarchical cluster approaches as described below.

Single link clustering (also called the connectedness, the minimum method, or the nearest neighbor method) is a method that considers the distance between two clusters to be equal to the shortest distance from any member of one cluster to any member of the other cluster. If the data consist of similarities, the similarity between a pair of clusters is considered to be equal to the greatest similarity from any member of one cluster to any member of the other cluster. A major disadvantage associated with this type of clustering is that a presence of noise can completely affect the grouping.

Complete-link clustering (also called the diameter, the maximum method, or the furthest neighbor method) is a method that considers the distance between two clusters to be equal to the longest distance from any member of one cluster to any member of the other cluster.

3.5.2.3 Partition Based Clustering

Partitioning methods relocate instances by moving them from one cluster to another, starting from an initial partitioning. Such methods typically require that the number of clusters are pre-set by the user. To achieve global optimality in partitioned-based clustering, an exhaustive enumeration procedure of all possible partitions is required. Because, this is not feasible, certain heuristics are used in the form of iterative optimization. Namely, a relocation method iteratively relocates points between the so called "k clusters".

The k-means algorithm may be viewed as a gradient-descent procedure, which begins with an initial set of k cluster-centers and iteratively updates it so as to decrease the error function. It maintain a set of data points the same size as the input data set. Initially, this set is copied from the input set. Then this set is iteratively replaced by the mean of those points in the set that are within a given distance to that point. By contrast, k-means restricts this updated set to k points usually much less than the number of points in the input data set, and replaces each point in this set by the mean of all points in the input set that are closer to that point than any other (e.g. within the Voronoi partition of each updating point). A mean shift algorithm that is similar then to k-means, called likelihood mean shift, replaces the set of points undergoing replacement by the mean of all points in the input set that are within a given distance of the changing set *(Kanungo et al. 2002)*. One of the advantages of mean shift over k-means is that there is no need to choose the number of clusters, because mean shift is likely to find only a few clusters if only a small number exist. However, mean shift can be much slower than k-means, and requires selection of a bandwidth parameter. Mean shift has soft variants much as k-means does.

3.5.3 Solution Approach

Fasteners on gusset plates are grouped to the individual truss members to which they are connected. There are two different ways to group the fasteners in the GPIPT – Automatic and manual and the user can select which they would like to perform. The automatic grouping of fasteners is based on the method of clustering.

For the automatic grouping to be possible, an initial set of seed locations should be available for the algorithm along with the final number of expected clusters. For a common gusset plate arrangement connecting five (5) truss members and having 6 edges of the shape shown in Fig. 3.9, there would be five (5) clusters (top left, top right, top middle, bottom left and bottom right). To determine the cluster seeds, all the fasteners centers detected on the image are used to establish the coordinates of the far extremes, along with the midpoint of top two extremes are considered as the initial seeds to the k-means algorithm resulting in an output of clusters. Once the clusters are formed, the addition and removal of fasteners from and to these groups is achievable through the user interface. However if the image at hand is of a gusset plate of a different shape than the one described by Figure 3.9, automatic detection of members may not be possible due to various unknown factors like the number of clusters, seed locations, etc. To accommodate such cases, the application is made semi-automatic where the user can randomly click on the image once in each member and the result is computed according to the number and location of clicks.

Figure 3.9: A common gusset plate shape for connecting 5 truss members (2 diagonals, 1 vertical, and 2 chord members)

For the semi-automatic grouping of fasteners, the image is displayed to the user and the input seed locations are selected by the user on the image. Accordingly grouping is implemented and presented. For the manual grouping, user is provided with a click and drag option of drawing a bounding box (of any number of line segments) around the fastener groups.

3.5.4 Challenges

There is no common pattern of fastener arrangement that can be drawn from the population of gusset plates as they have different shapes, different number of connecting members, and different arrangement of fasteners. In order to automatically group fasteners, there exist many unknown factors that are described below.

3.5.4.1 Number of Members Varies

Each gusset plate has a different number of truss members attached to it at different locations. Since each gusset plate has different number of members, it is unlikely that automatic detect will work for all plates. As seen in Figure 4.10, gusset plate connections can vary drastically in terms of shape, size, and fastener arrangement. Hence unless the shape and fastener arrangement is known a priori, a fully automatic fastener grouping is close to unachievable. However, for the most commonly seen gusset plate template (5 members, as shown in Figure 4.9) algorithms have been implemented to execute automatic grouping.

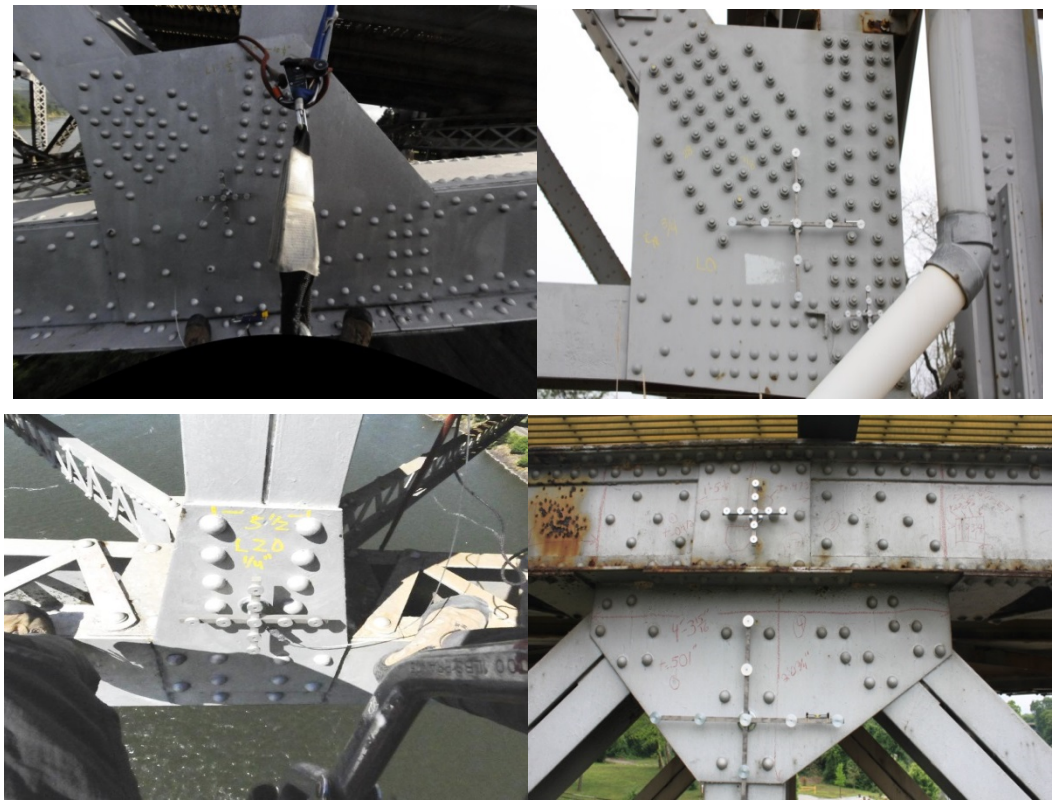


Figure 3.10: Examples of gusset plates with highly varied order of fasteners.

3.5.4.2

The fasteners are not uniformly spaced for automatic cluster determination to be possible. Even in the common five (5) member gusset plate template, the fastener arrangement is not guaranteed to be regular. Depending upon the connection, fasteners are either tightly or loosely bound. In cases of tightly bound fastener members, the complete linking Hierarchical clustering algorithm performed well even though seed locations were not provided. The only requirement was the number of clusters in the image. However, the disadvantage was poor performance in cases of loosely bound fastener groups. In cases of loosely bound members, the K-means algorithm with approximately calculated seed locations and number of clusters did a significantly better job. Results are compared for

the two different approached for automatic fastener grouping considering tightly and loosely bound fastener arrangements in gusset plates as seen in Figures 3.11 and 3.12.

Figure 3.11: Gusset plate with loosely arranged fasteners. Fasteners grouped using k-means on the left and complete link Hierarchical clustering in the right. (k-means clustering performed better)

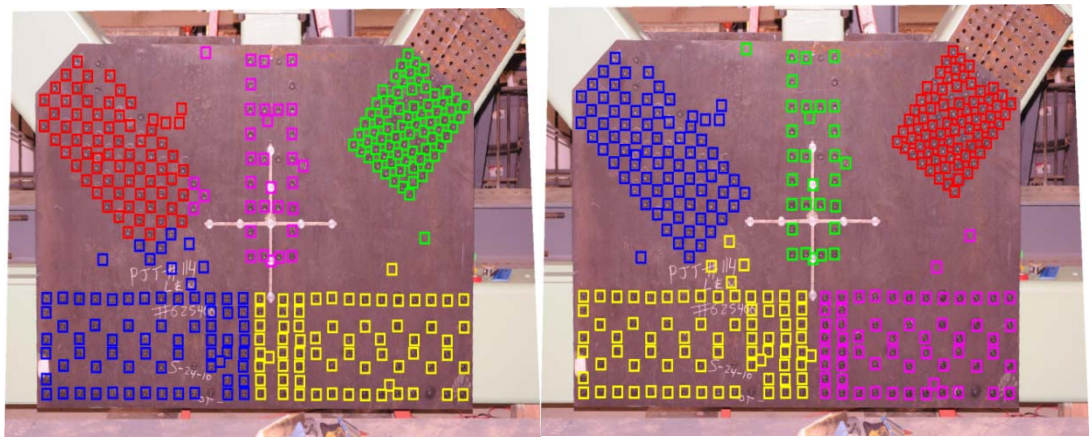


Figure 3.12: Gusset plate with tightly arranged fasteners. Fasteners grouped using k-means on the left and complete link Hierarchical clustering in the right (complete link Hierarchical clustering performed better)

In order to perform well in all the possible cases, the user is given an option of either choosing the semi-automatic fastener grouping or manually grouping the fasteners. In the semi-automatic grouping, the user can tell the GPIPT the number of truss members in each image or randomly click on the members. The number of clicks represents the number of clusters and the appropriate algorithm is implemented. In case of manual grouping, the interface allows the user to draw a bounding box around the fasteners associated with a particular member. The results using manual grouping for a loosely bound fastener pattern are shown in Figure 3.13.

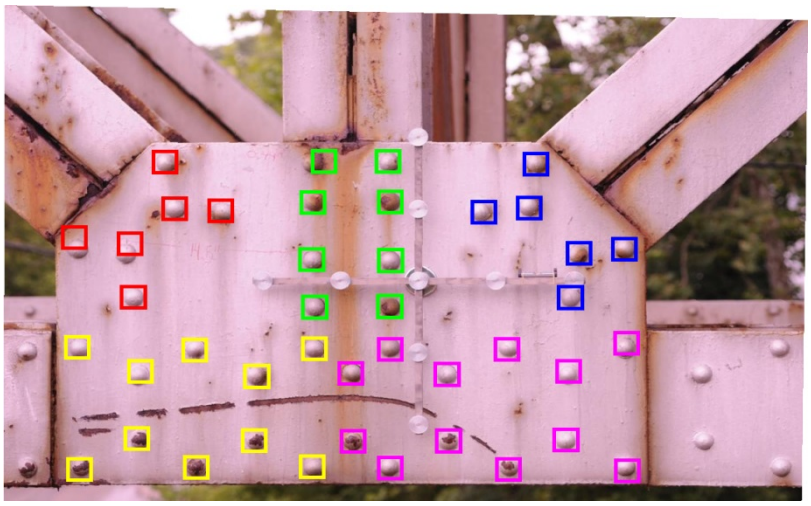


Figure 3.13: Example of a gusset plate with fasteners manually grouped in GPIPT

4.0 GUSSET PLATE ANALYSIS

The final step of the GPIPT is the analysis of the connection using AASHTO LRFD requirements. The analysis requires all the prior steps to have been completed previously (image converted to a scaled orthograph, plate boundary defined, fasteners identified and grouped). The analysis is performed for fastener shear strength, bearing resistance at fastener holes, tension yielding and fracture on Whitmore section, block shear, compression on the Whitmore effective column, and full and partial plate yielding and fracture shear strengths. The analysis results are for a single plate. To determine capacity of two plate connections, the results need to multiplied by two (2). The analysis cannot evaluate chord splice plates or shingled plates. However, these can be analyzed separately and superimposed with the main gusset plate results. In the case of shingle plates, they can be run through the tool a second time to perform the analysis separately.

In the following sections of this chapter, the variable names used in the GPIPT are shown in green to allow future users to make revisions. Red is used to highlight user inputs.

4.1 INITIAL CONSTANTS AND PLATE PROPERTIES

The following constants are used in the analysis:

- PI 3.1415927
- Young's modulus E=29,000.00 ksi
- Resistance factor for gusset plate compression phicg = 0.95
- Resistance factor for gusset plate chord splices phics = 0.85
- Resistance factor for gusset plate shear yielding phivy = 1.00
- Resistance factor for gusset plate block shear rupture phibs = 1.00
- Resistance factor for gusset plate shear fracture phivu = 0.80
- Resistance factor for tension, fracture in net section phiu = 0.80
- Resistance factor for tension, yielding in gross section phiy = 0.95
- Resistance factor for A325 and A490 bolts or rivets in shear phis= 0.80
- Resistance factor for A307 bolts in shear phis307 = 0.75
- Resistance factor for fasteners bearing on material phibb = 0.80

User input for the plate material properties and thickness:

- Yield stress of plate (ksi) FYPL
- Ultimate stress of plate (ksi) FUPL
- Plate thickness (in) TPL

The analysis process flows for each behavior (fastener shear for example) in order for each of the members (kth member) until all members are analyzed and then the next behavior is analyzed. The kth member corresponds to bolt group k. The number of shear planes considered is not dependent on the number of members and the user can choose to do as many shear planes as desired. The analysis results are reported in a text file that echoes the inputs and shows the design strengths for all the limit states detailed subsequently.

4.2 FASTENER SHEAR STRENGTH

After inputting the plate material properties and thickness, the analysis begins with the fastener shear strength. All fastener types are coded including rivet and bolts, as well as the different material properties. First, the center of gravity of the fastener group relative to origin (located at lower left hand corner of image) is computed for the kth member as:

$$xcg_k = \frac{\sum_{i=1}^{N1} x_i}{N_k} \quad \textbf{(4.1)}$$

$$ycg_k = \frac{\sum_{i=1}^{N1} y_i}{N_k} \quad \textbf{(4.2)}$$

where x and y are the coordinate locations of the fasteners and N_k is the total number of fasteners in the kth group. The result produces the location of the center of gravity of the bolt group as shown in Fig. 4.1. This is needed to assess later member behaviors.

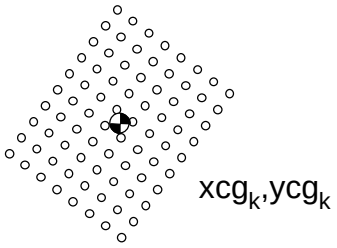


Figure 4.1: Computed center of gravity for bolt group k

After establishing the fastener center of gravity, the connection length must be determined. This requires the user to select two points that define overall length of the connection. This selection

also defines the vector orientation of the member force, which is required for any subsequent finite element analysis. User input: Click two extreme points that define the direction of the member force? The required points are illustrated in Fig. 4.2.

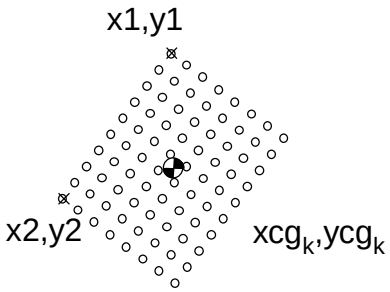


Figure 4.2: Two points produce vector defining direction of member force and establishe the connection length for kth member

The connection length of the kth connection, $LCON_k$, is computed as:

$$LCON_k = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad \textbf{(4.3)}$$

where x and y are the coordinates defining the ends of the connection. The user then must describe the fastener diameter for the kth group. User input: Nominal Fastener Diameter in group= ? (in). DB_k The fastener area for the kth group, AB_k, is computed as:

$$AB_k = 0.25 * PI * DB_k{}^2 \qquad \textbf{(4.4)}$$

Next, the user must determine if the fastener is a bolt or rivet and then establish the material properties of the fasteners. User input: Bolt or Rivets in group = ? $BOLT_k$ or $RIVET_k$. The logic flow is shown below.

If $BOLT_k$, User input: Select type of bolt: 1) A307, 2) A325, or 3) A490?

If A307 then FUB_k=60 ksi and the grip length need to be specified. User input: What is the grip length of the A307 bolts in the connection? $GLEN_k$. From this, the grip length reduction factor, $GREDF_k$, is computed as:

$$GREDF_k = 1 - \left(GLEN_k - 5 * DB_k\right) 0.16 \qquad \textbf{(4.5)}$$

If A490 then FUB_k=150 ksi and $GREDF_k$=1.0

IF A325 and DB_k<=1.0 then FUB_k=120 ksi and $GREDF_k$=1.0

IF A325 and DB_k>1.0 then FUB_k=105 ksi and $GREDF_k$=1.0

Next the user must determine if the threads are located in the shear plane. It is conservative to assume they are located within the shear plane. User input: Are threads

located in the shear plane (conservative to assume yes)? Yes=$THREAD_k$ =1
No=$THREAD_k$=0.

For long connections (exceeding 50 in.), a reduction factor of 0.8 is applied. The logic is expressed as:

IF $LCON_k$<= 50. Then $LCONF_k$=1.0 else $LCONF_k$=0.8

If $RIVET_k$, the user must set the material type. User input: Select type of rivet: 1) Unknown, 2) Carbon steel, ASTM A141, or ASTM A502 Grade I 3) ASTM A502 Grade II? From this selection, the material strength is established as:

If 1 then FUB_k=50 ksi
If 2 then FUB_k=60 ksi
If 3 then FUB_k=80 ksi

For long connections (exceeding 50 in.), a reduction factor is computed between 1 and 0.75. as: If $LCON_k$<= 50. Then the length reduction factor is computed as:

$$LCONF_k = 1 - (0.25 LCON_k / 50) \text{ else } LCONF_k = 0.75 \quad \textbf{(4.6)}$$

It is not possible to tell if fills are located in the connections or if they are developed. The reduction factor for the undeveloped fill plates must be specified by the user where applicable. User input: Are there undeveloped fill plates >=0.25 in. thick used in the member connection? $FILLY_k$ or $FILLN_k$. If yes ($FILLY_k$) , User input: What is the reduction factor for undeveloped fill plates? $RFILL_k$. If No, $RFILL_k$=1.0

Once all the inputs are collected, the individual fastener shear strength in a single shear plane, $RNFS_k$, is computed. For bolts, the shear strength is computed as one of the following:

If A307 and $THREAD_k$ =0 then

$$RNFS_k = phis307 * 0.48 * AB_k * FUB_k * GREDF_k * LCONF_k * RFILL_k \quad \textbf{(4.7)}$$

If A307 and $THREAD_k$ =1 then

$$RNFS_k = phis307 * 0.38 * AB_k * FUB_k * GREDF_k * LCONF_k * RFILL_k \quad \textbf{(4.8)}$$

If A325 or A490 and $THREAD_k$ =0 then

$$RNFS_k = phis * 0.48 * AB_k * FUB_k * GREDF_k * LCONF_k * RFILL_k \quad \textbf{(4.9)}$$

If A325 or A490 and $THREAD_k$ =1 then

$$RNFS_k = phis * 0.38 * AB_k * FUB_k * GREDF_k * LCONF_k * RFILL_k \quad \textbf{(4.10)}$$

If the fastener is a rivet then the shear strength of the rivet in a single shear plane is computed as:

$$RNFS_k = phis * AB_k * FUB_k * 0.67 * LCONF_k * RFILL_k \quad \textbf{(4.11)}$$

To account for additional shear planes which cannot be determined from the image view, the user must prescribe the number of shear planes in the group. If some fasteners have two (2) and some have only one (1) shear plane, the user can prescribe a fractional value that can account for the average numbers of shear planes within the group. User input: How many shear planes are available in group? NSP_k. The total fastener shear strength in group k, TFS_k, is determined by multiplying the total number of fasteners in the group with the shear strength of as single fastener and is computed as:

$$TFS_k = RNFS_k * NSP_k \quad \textbf{(4.12)}$$

The reporting features echo the inputs, report the connection length, the associated material properties, computed reduction factors, and output the single shear plane shear strength per fastener, and the total shear strength of the fastener group.

4.3 BEARING RESISTANCE AT FASTENER HOLES

In this step of the analysis, the bearing resistance at the fastener holes is calculated. The bearing resistance depends on the spacing of the fasteners as well as the distance between the plate edge and nearest fastener. The user must select points that indicate these distances. First the shortest spacing between fasteners in the line of action of the member force must be established as illustrated in Figure 5.3. User Input: Click closest distance between two fasteners in line of action of member?

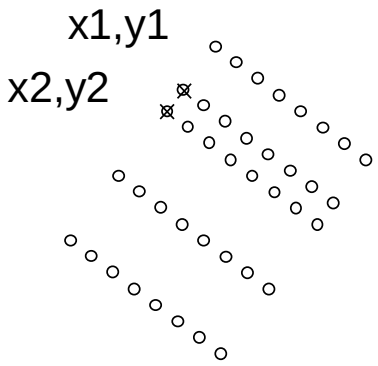


Figure 4.3: Two user selected points that define the shortest distance between the centers of two fasteners in the line of action of the member force for the kth member

The shortest distance between the two fasteners, $LFAST_k$, is computed as:

$$LFAST_k = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \textbf{(4.13)}$$

The clear distance between the two fasteners, $CLFAST_k$, is based on the fastener diameter prescribed previously for the member group and an assumed 1/8 inch tolerance on the hole sizes. The clear distance is computed as:

$$CLFAST = LFAST - DB - (1/8) \quad \textbf{(4.14)}$$

Next, the distance between the edge of the plate and closest fastener in the line of action of the member needs to be selected as illustrated in Fig. 5.4. User input: Click closest distance from edge of plate to fastener in line of action of member?

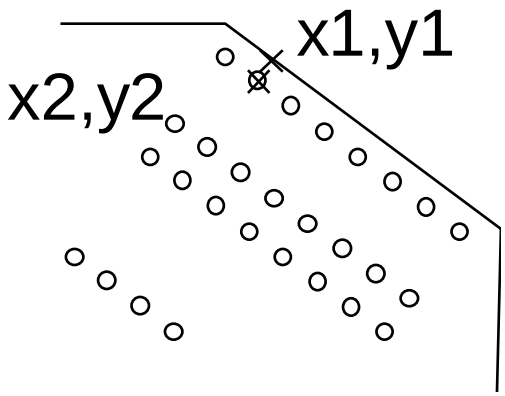


Figure 4.4: Two user selected points on that define the shortest distance between the edge of the plate and the closest fastener in the line of action of the member force for the kth member

The length of the edge distance, $LEDGE_k$, is computed as:

$$LEDGE_k = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \textbf{(4.15)}$$

and the clear distance between the fastener and plate edge, $CLEDGE_k$, is based on the fastener radius from the previously prescribed diameter for the member group and an assumed 1/16 inch oversized hole.

$$CLEDGE_k = LEDGE_k - DB_k * 0.5 - (1/16) \quad \textbf{(4.16)}$$

The limiting distance, LC_k, is determined based on the following logic:

IF $CLFAST_k < 2*DB_k$ or $CLEDGE_k < 2DB_k$ then

IF($CLFAST_k < CLEDGE_k$) then $LC_k = CLFAST_k$

Else $LC_k = CLEDGE_k$.

The bearing resistance of an individual fastener hole, $RNBEAR_k$, can then be computed as:

$$RNBEAR_k = phibb * 2.4 * DB_k * TPL * FUPL \quad \textbf{(4.17)}$$

Else

$$RNBEAR_k = phibb * 1.2 * LC_k * TPL * FUPL \quad \textbf{(4.18)}$$

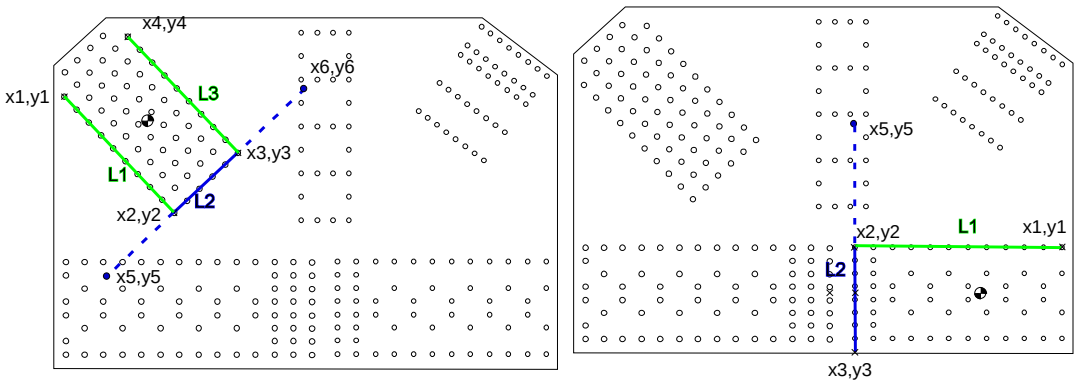
The total bearing strength of group k, $TFBEAR_k$, is determined based on the bearing resistance of the individual fasteners times the total number of fasteners in the group and is computed as:

$$TFBEAR_k = RNBEAR_k * N_k \quad \textbf{(4.19)}$$

The GPIPT reports the fastener spacing lengths, the bearing resistance of a single fastener hole, and the total bearing resistance of the fastener group.

4.4 WHITMORE AND BLOCK-SHEAR RESISTANCE

In the next phase of the analysis, the stresses on the Whitmore section and the block-shear resistance are determined. To perform these calculations, the user must identify whether the connected truss member is a vertical or diagonal compared vs a chord. Vertical and diagonals have three lines that define the block shear path compared to chord members that have only two lines that define the block sure path. Similarly to compute the Whitmore section, the assumed projection of the spread of stress is different for these two members. User input: Is member parallel and close to free edge of plate (like chord) (Y/N)? If "yes" the kth member is a chord and the user needs to define only 2 lines (using 3 points, as illustrated in Fig. 5.5a). If "no", the kth member is a vertical or diagonal and the user needs to define to define 3 lines (using 4 points, as illustrated in Fig. 5.5b). User input: Pick fasteners that define the perimeter of the member connection? Note, that only points 1 to 4 need to be selected for the verticals and diagonals and only points 1 to 3 need to be selected for chords. Also, the picking order of the points must be sequential.



a) b)

Figure 4.5: a) Whitmore section for 3 line member, b) Whitmore section for 2 line member

Given the selected points, the line lengths, $L1_k$, $L2_k$, and $L3_k$ are computed as:

$$L1 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \textbf{(4.20)}$$

$$L2 = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \quad \textbf{(4.21)}$$

$$L3 = \sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2} \quad \textbf{(4.22)}$$

(For 3 line members)

To establish the orientation to project the Whitmore section lines, the angle of the Whitmore section, $theta_k$, is *computed* with the following logic:

If $y_3 - y_2 = 0$ then $theta_k = 90$.

$$\text{Else } theta = ATAN\left(\frac{x_3 - x_2}{y_3 - y_2}\right) \quad \textbf{(4.23)}$$

The projection of two parts of the Whitmore length(s), $W1_k$ and $W2_k$, are computed as:

$$W1_k = Tan30 * L1_k \quad \textbf{(4.24)}$$

$$W3_k = Tan30 * L3_k \quad \textbf{(4.25)}$$

(For 3 line members)

The vertical and horizontal projections of the above computed Whitmore length(s) are used to graphically display the results to the user and are computed as:

$$dx1_k = W1_k * \sin(theta_k) \quad \textbf{(4.26)}$$

$$dy1_k = W1_k * \cos(theta_k) \quad \textbf{(4.27)}$$

$$dx3_k = W3_k * \sin(theta_k) \quad \textbf{(4.28)}$$

(For 3 line members)

$$dy3_k = W3_k * \cos(theta_k) \quad \textbf{(4.29)}$$

(For 3 line members)

The end points for Whitmore section are determined based on the orientation and then computed as:

$$signx_k = x_2 - x_3 \quad \textbf{(4.30)}$$

$$signy_k = y_2 - y_3 \quad \textbf{(4.31)}$$

$$x_5 = x_2 + signx_k * dx1_k \quad \textbf{(4.32)}$$

$$y_5 = y_2 + signy_k * dy1_k \quad \textbf{(4.33)}$$

$$x_6 = x_3 - signx_k * dx3_k \quad \textbf{(4.34)}$$

(For 3 line members)

$$y_6 = y_3 - signy * dy3 \quad \textbf{(4.35)}$$

(For 3 line members)

If the end points of the Whitmore section fall outside the plate, they are trimmed at the edge based on the plate boundary determined previously in the processing of the image. The line dimensions $W1_k$ or $W2_k$ are modified based on the final end points (that may rest at the plate edge). After completing the computations, the line segments that make up the input lines and the Whitmore section (x5,y5 -> x6,y6 for three line members or x3,y3->x5,y5 for two line members) are displayed on the image and the gross length of the Whitmore section is computed as:

$$LGW_k = L2_k + W1_k + W2_k \quad \textbf{(4.36)}$$

Note that $W2_k$ will be zero for two line members. Next the number of fasteners that cross line segments $L1_k$, $L2_k$, and $L3_k$ are determined based on the known fastener locations. The fasteners have a prescribed area defined as a bounding box based on the fastener diameter to allow for tolerance in identifying fastener-line intersections. The number of fasteners crossing lines $L1_k$, $L2_k$, and $L3_k$ are counted as $NFL1_k$, $NFL2_k$, $NFL3_k$, respectively. Care is taken to note double count the fasteners located at the intersections of $L1_k$ and $L2_k$, and $L2_k$, and $L3_k$. The net length of Whitmore section is computed by taking the gross length and subtracting the number of fastener diameters that intersect it as:

$$LNW_k = L2_k + W1_k + W2_k - \left(NFL2_k - 1\right)\left(DB_k + 1/8\right) \quad \textbf{(4.37)}$$

For the above calculation, $W2_k$ will be zero for two line members. For block sure calculations the net line lengths, $LN1_k$, $LN2_k$, $LN3_k$, need to be determined for each of the relevant line sections by removing all the fastener holes that intersect the lines. These are computed as:

$$LN1_k = L1_k - (NFL1-1)*(DB_k+1/8)+LEDGE_k \quad \textbf{(4.38)}$$

$$LN2_k = L2_k - (NFL2_k-1)*(DB_k+1/8) \quad \textbf{(4.39)}$$

$$LN3_k = L3_k - (NFL3_k-1)*(DB_k+1/8)+LEDGE_k \quad \textbf{(4.40)}$$

The gross lengths of the shear lines along the fastener pattern, $LG1_k$ and $LG3_k$,are determined from the originally picked line segments plus the edge distance from the plate boundary to the first fastener line and are computed as:

$$LG1_k = L1_k + LEDGE_k \quad \textbf{(4.41)}$$

$$LG3_k = L3_k + LEDGE_k \quad \textbf{(4.42)}$$

Note that the edge length is introduced in the above calculations and comes from the edge length determined for the fastener bearing calculations in the earlier step.

The force required to produce yielding on the kth member gross Whitmore section, $TFYW_k$, is computed as:

$$TFYW_k = phiy * FYPL * TPL * LGW_k \quad \textbf{(4.43)}$$

while force required to produce fracture on the kth member net Whitmore section, $TFFW_k$, is computed as:

$$TFFW_k = phiu * FUPL * TPL * LNW_k * 1.0 \quad \textbf{(4.44)}$$

The controlling block-shear strength of the kth member, $TFBS_k$, is taken as the smaller of yielding on the gross tension area with fracture of the net shear areas, $TFBS1_k$, as:

$$TFBS1_k = phibs * 0.9 * (0.58 * FUPL * TPL * (LN1_k + LN3_k) + FYPL * TPL * LN2_k) \quad \textbf{(4.45)}$$

or fracture of the net tension area with yielding on the net shear areas, $TFBS2_k$, as:

$$TFBS2_k = phibs * 0.9 * (0.58 * FYPL * TPL * (LG1_k + LG3_k) + FUPL * TPL * LN2_k) \quad \textbf{(4.46)}$$

To determine which one controls, the following logic is used:

$$\text{If } TFBS1_k > TFBS2_k, \text{ then } TFBS_k = TFBS2_k \text{ Else } TFBS_k = TFBS1_k \quad \textbf{(4.47)}$$

The GPIPT reports the gross Whitmore length, number of fasteners crossing each of the line segments, the number of fasteners crossing the Whitmore length, the net Whitmore length, the Whitmore yielding force, the Whitmore fracture force, and controlling block-shear strength.

4.5 COMPRESSION RESISTANCE

To determine the compression strength of the gusset plate for members that apply compressive forces to the connection, the previously computed Whitmore section is combined with an effective column length. The effective column length is computed as a line segment, originating at and orthogonal to the Whitmore section that intersects the first adjacent member fastener line. Here a reference line is produced on the image that originates at the connection center of gravity and projects in the direction of the member force to the edge of the image (all necessary data are produced in the previous steps). The user must trim the projected line at the adjacent member fastener interface by drawing a line that intersects the projected line at the member line. User input: Pick equivalent column length from the red line shown in the image. The user must click two points that establish the trim line at the end of the effective column length. The results are illustrated in Figs. 5.6a and b. The effective length of the column, $LMID_k$, is the distance between the intersection of the Whitmore section and the user selected trim location Compute length of line.

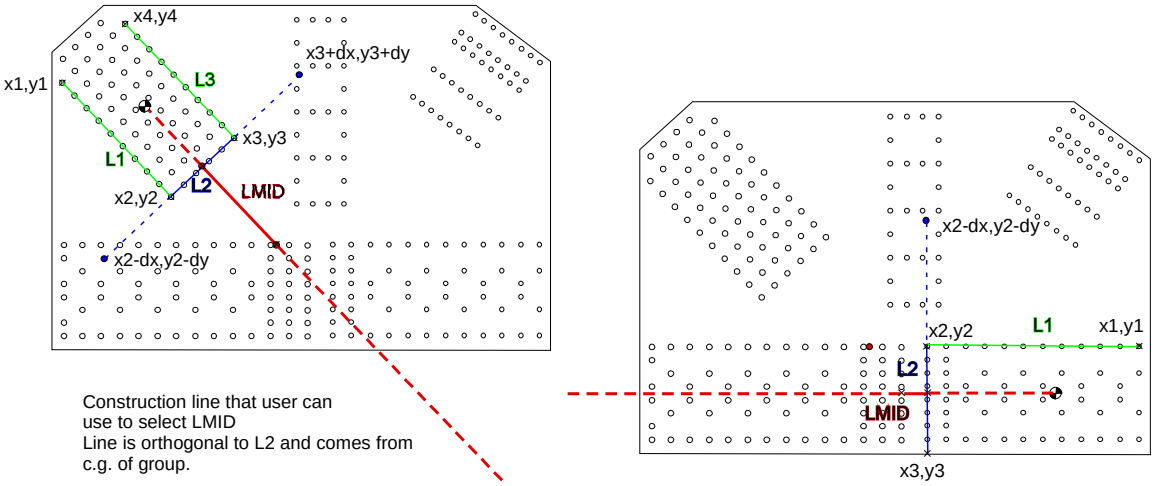


Figure 4.6: Construction line (dashed) projected in the line of action for the member from center of gravity of the fastener group that the user trims at the location where it intersects the nearest adjacent member fastener line a) diagonal and b) chord

Once the effective column length is established, the elastic buckling load, PE_k, is computed as:

$$PE_k = \frac{3.29 * E * TPL * LGW_k}{\left(\frac{LMID_k}{TPL}\right)^2} \quad \textbf{(4.48)}$$

The compressive yielding force on the Whitmore section, PO_k, is computed as:

$$PO_k = FYPL * TPL * LGW_k \quad \textbf{(4.49)}$$

The ratio of the elastic buckling load to the compressive yielding load, $RATIO_k$, is computed as:

$$RATIO_k = PE_k / PO_k \quad \textbf{(4.50)}$$

If $RATIO_k$<0.44 then the nominal compressive force, PN_k, is computed as:

$$PN_k = 8.877 * PE_k \quad \textbf{(4.51)}$$

Else

$$PN_k = PO_k * 0.658^{(PO_k / PE_k)} \quad \textbf{(4.52)}$$

The effective compressive stress, $COMPST_k$, on the Whitmore section is computed as:

$$COMPST_k = PN_k / (\mathrm{TPL} * \mathrm{LGW}_k) \quad \textbf{(4.53)}$$

The design buckling force, TFC_k, is computed as:

$$TFC_k = phicg * PN_k \quad \textbf{(4.54)}$$

The GPIPT reports the effective column length, nominal elastic buckling force, nominal yielding force, and nominal effective compressive force on the effective column, as well as the effective compressive stress on the Whitmore section, and the design compressive force on the effective column.

4.6 SHEAR PLANE RESISTANCE

The user can choose to check as many partial and/or full shear planes as is desired. The user must specify how many planes to check. User input: How many shear planes to check? NV. After specifying the number of planes to check, the GPIPT automatically loops through NV times to have the user define the lines. The user clicks two points (x_1,y_1 and x_2,y_2) that define the shear plane as illustrated in Fig. 5.7. If fasteners are located along the line, the number of fasteners will be computed to determine the net section. Note software accounts for different possible fastener diameters within the k different fastener groups that intersect with the given line.

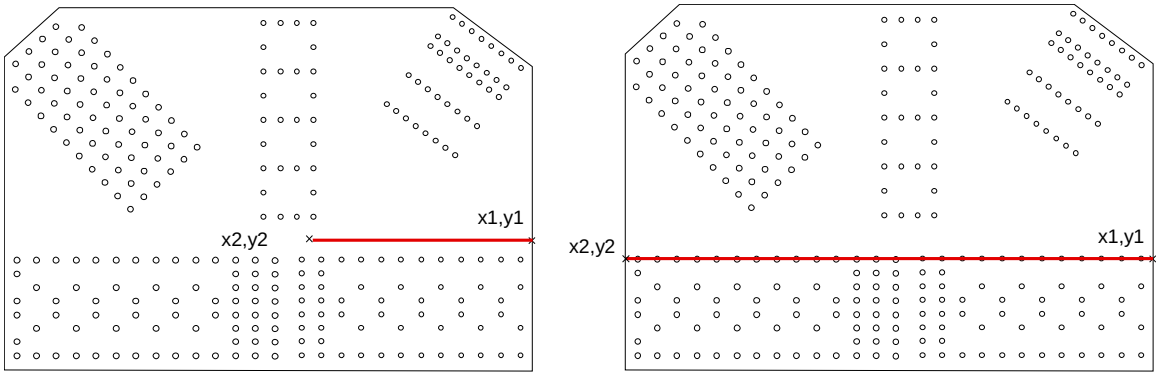


Figure 4.7: Example lines defining shear planes a) partial plane (with gross section) and b) full plane (with net section)

The line length of the NVth shear plane, LV_{NV}, is computed as:

$$LV_{NV} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \textbf{(4.55)}$$

The gross shear yielding design strength of the NVth plane, $TFVY_{NV}$, is computed as:

$$TFVY_{NV} = phivy * 0.58 * FYPL * TPL * LV_{NV} * 0.88 \quad \textbf{(4.56)}$$

The net shear fracture design strength of the NVth plane, $TFVF_{NV}$, is computed as:

$$TFVF_{NV} = phivu * 0.58 * FUPL * TPL * \left[LV_{NV} - \sum N_{NV,k} * (DB_{NV,k} + 1/8) \right] \quad \textbf{(4.57)}$$

The GPIPT reports the shear plane number, the number of fasteners crossing the shear plane (if any), the gross length of the shear plane, the shear yielding design strength, and the shear fracture design strength.

All analysis results are reported to an output file in .txt format in the text files folder of the working directory.

5.0 CONCLUSIONS

The GPIPT software includes all features detailed about to process the raw digital photograph with an image target, collect the geometric information of the gusset plate and fasteners, and to conduct the AASHTO-LRFR rating calculations, as well as to develop the input file for subsequent finite element analysis. The software runs Java (presently jre1.8.0_77) and requires MATLAB (presently version 2013b or 2014b) with the Image Processing Toolbox. Detailed instructions for running the software are found in a set of videos in .MP4 format that are contained in the folder structure shown in Fig. 5.1.

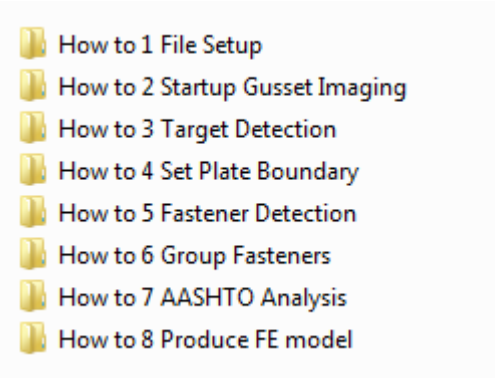


Figure 5.1: Folder structure for instructional videos

All software associated with this project comes with the following disclaimer:

Permission to use, copy, modify, and distribute this software and its documentation for educational, research and non-profit purposes, without fee, and without a written agreement is hereby granted, provided that the above copyright notice, this paragraph and the following three paragraphs appear in all copies.

Permission to incorporate this software into commercial products may be obtained by contacting OREGON STATE UNIVERSITY Office for Commercialization and Corporate Development.

This software program and documentation are copyrighted by OREGON STATE UNIVERSITY. The software program and documentation are supplied "as is", without any accompanying services from the University. The University does not warrant that the operation of the program will be uninterrupted or error-free. The end-user understands that the program was developed for research purposes and is advised not to rely exclusively on the program for any reason.

IN NO EVENT SHALL OREGON STATE UNIVERSITY BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE OREGON STATE UNIVERSITY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. OREGON STATE UNIVERSITY SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO,

THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND ANY STATUTORY WARRANTY OF NON-INFRINGEMENT. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND OREGON STATE UNIVERSITY HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

PART 2: FINITE ELEMENT MODELING AND ANALYSIS OF GUSSET PLATES FROM IMAGES

Michael H. Scott, ph.D.
and
Christopher Higgins, Ph.D., P.E.

1.0 GETTING STARTED

The Gusset Plate analysis software is built upon the OpenSees finite element software framework *(McKenna et al. 2000, 2010)*. OpenSees is linked with the Tcl/Tk scripting language *(Ousterhout 1994 and Welch 2000)*, offering software developers the opportunity to build graphical user interfaces for customized finite element software applications. Documentation of the Gusset Plate analysis software, developed using this OpenSees/Tcl/Tk approach, is presented herein.

1.1 INSTALLATION INSTRUCTIONS

To use the Gusset Plate analysis software, the user must install version 8.5.18 of Tcl/Tk (64-bit), which can be downloaded from the following site: http://downloads.activestate.com/ActiveTcl/releases/8.5.18.0/ActiveTcl8.5.18.0.298892-win32-x86_64-threaded.exe In addition, the following two files should be installed in a local directory where the user has write access:

- OpenSeesTk.exe (64-bit)
  https://www.dropbox.com/s/b98q6uwyfv0k0lf/openSeesTk.exe

- zGusset.tcl – Available via anonymous ftp

The following section describes how to use the software.

1.2 STARTUP INSTRUCTIONS

To start the Gusset Plate analysis software, double click the OpenSeesTk.exe icon. This will launch an MS-DOS window. Then, at the command line, type source zGusset.tcl, as shown in Figure 1.1.

Upon successful launch of the software, the main window shown in Figure 1.2 appears with two editable fields:

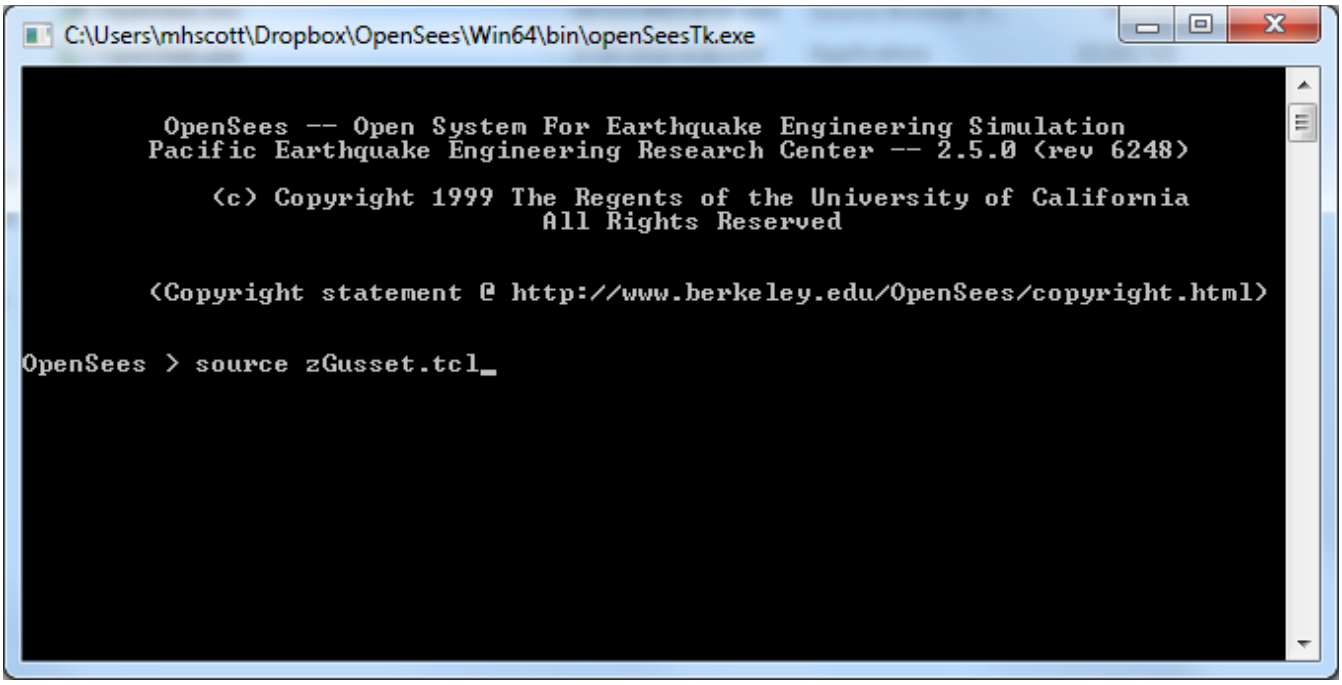


Figure 1.1: MS-DOS window at start of OpenSeesTK

- Gusset Plate – Name of the gusset plate to be analyzed. The default name is GussetPlateXYZ where XYZ is a system-dependent number returned by the [clock clicks] function of Tcl.

- Engineer – Name of the analyst. The default name is standard.

Each of these fields will appear in the report generated after the gusset plate analysis.

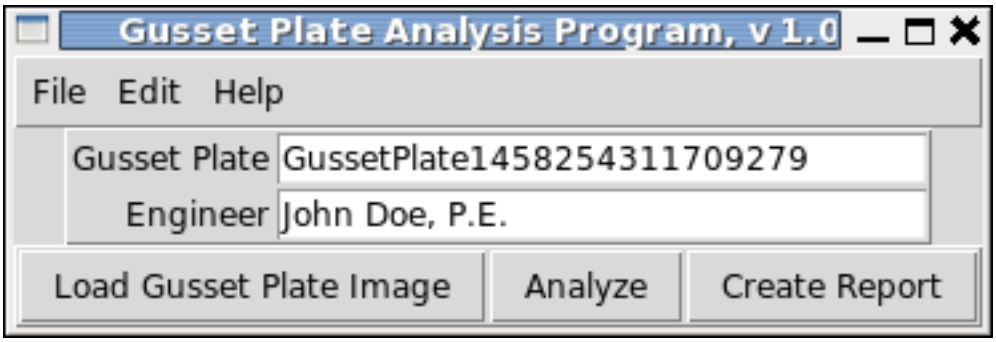


Figure 1.2: Main window at start of Gusset Plate analysis software

1.3 LOADING A GUSSET PLATE IMAGE FILE

To load a gusset plate image file, the Load Gusset Plate Image option is selected from the File menu shown in Figure 1.3. Alternatively, the user can use the keyboard shortcut Ctrl+L to load an image file or press the Load Gusset Plate Image button on the main window.

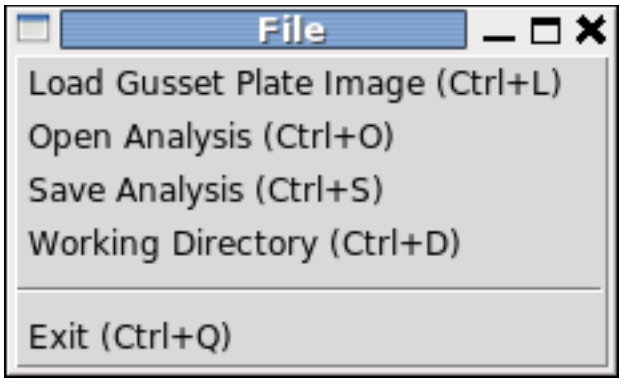


Figure 1.3: File menu for Gusset Plate analysis software

After selecting a gusset plate image file, the data is loaded and the main window expands to incorporate the boundary of the gusset plate, the bolt locations, and the number assigned to each bolt group. The example gusset plate shown in Figure 1.4 has five bolt groups. The group numbers are drawn at the geometric centroid of the bolts in each group.

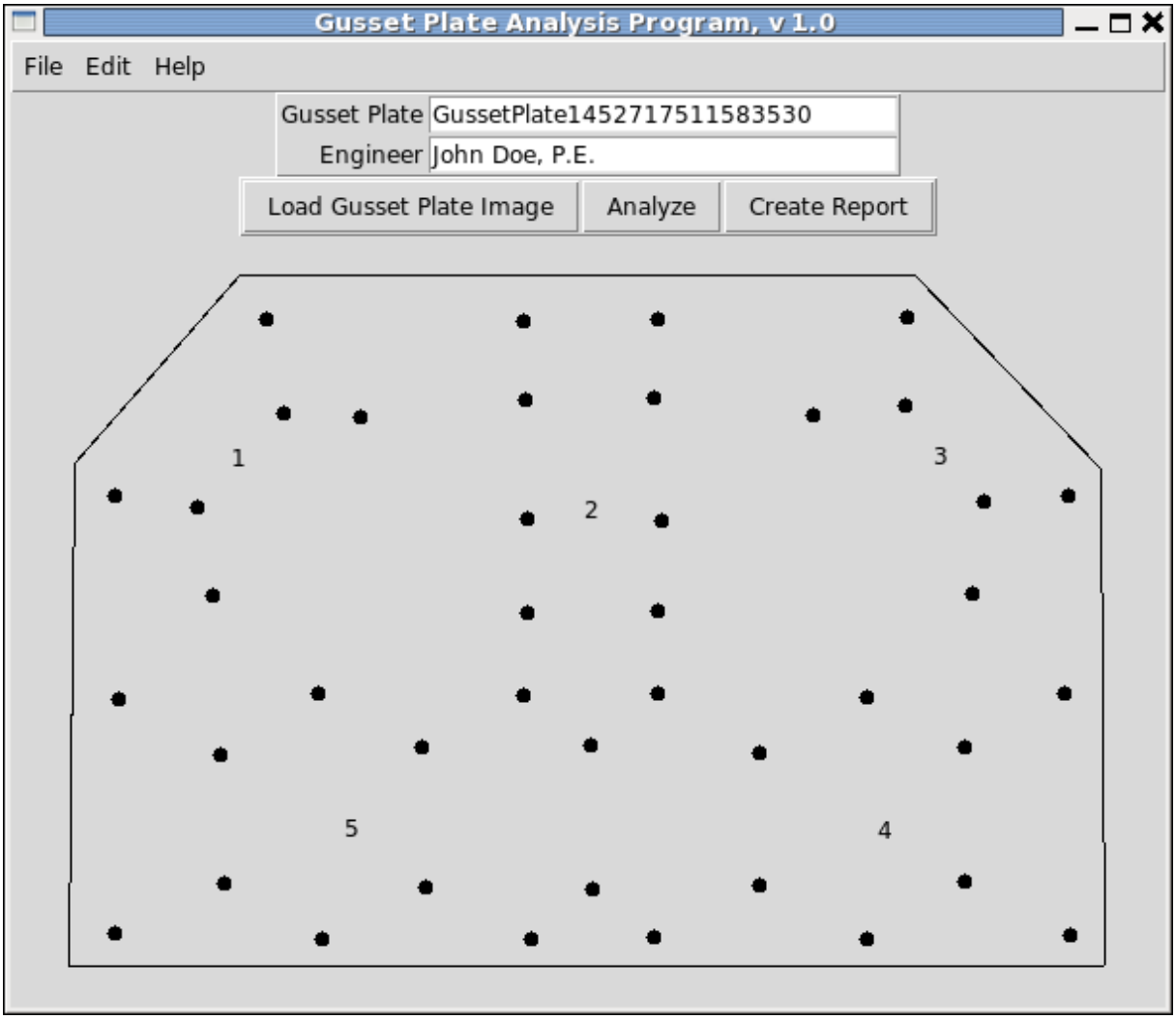


Figure 1.4: Main window after gusset plate image file is loaded

2.0 FINITE ELEMENT MODEL

A three-dimensional (3D) finite element model is developed from the plate corner locations and fastener locations contained in the image input file. A mesh of triangular shell elements is created using Delaunay triangulation *(Dvorkin and Bathe 1984)* with nodes corresponding to each fastener location (in addition to nodes created by Delaunay triangulation away from the fasteners). Properties of the finite element model and options for the finite element analysis of the gusset plate can be edited via the Edit menu shown in Figure 2.1.

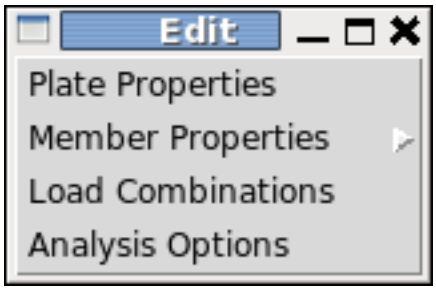


Figure 2.1: Edit menu for Gusset Plate analysis software

2.1 PLATE PROPERTIES

Material and geometric properties of the plate are input via the Plate Properties window shown in Figure 2.2. The first four inputs relate to the steel material properties:

- fy – yield stress (default: 36 ksi, range: > 0)
- E – elastic modulus (default: 29,000 ksi, range: > 0)
- v – Poisson ratio (default: 0.3, range: 0 to 0.5)
- α – strain hardening ratio (default: 1% (0.01), range: 0 to 100% (1.0))

The final three inputs allow the user to specify the plate thickness (default: 0.4375 in), the Number of Plates, either one or two (default: two), and, if there are two plates, the Plate Separation (default: 8 in). If Number of Plates is set to one, then the Plate Separation input is disabled.

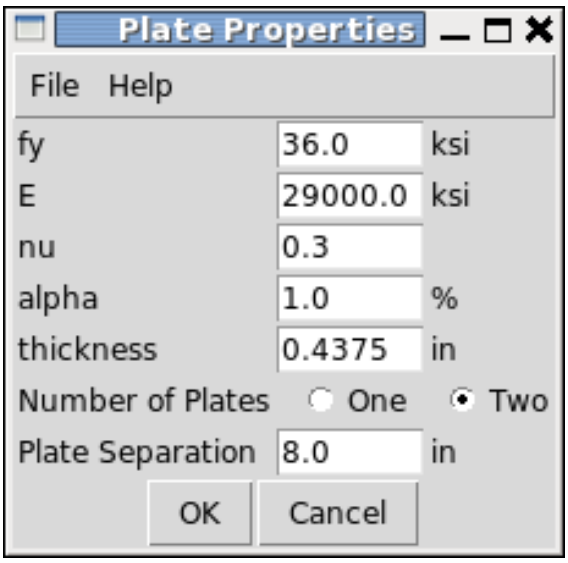


Figure 2.2: Window showing material and geometric properties of the gusset plate

2.1.1 Steel Constitutive Behavior

The steel stress-strain response is simulated using a J_2 plasticity model with linear strain- hardening using the parameters input via the window shown in Figure 2.2. The one- dimensional stress-strain response of this model is shown in Figure 2.3 with linear strain hardening.

2.1.2 Formulation for Plate Stress Condition

To represent the state of stress at each point in the 3D gusset plate model, the constitutive model is formulated considering $\sigma_{33} = 0$, i.e., that the stress in the out of plane direction of the gusset plate is zero. Five components of stress remain

$$\boldsymbol{\sigma} = \left[\begin{array}{ccccc} \sigma_{11} & \sigma_{22} & \sigma_{12} & \sigma_{23} & \sigma_{13} \end{array} \right]^T \tag{2.1}$$

with associated strains

$$\boldsymbol{\varepsilon} = \left[\begin{array}{ccccc} \varepsilon_{11} & \varepsilon_{22} & 2\varepsilon_{12} & 2\varepsilon_{23} & 2\varepsilon_{13} \end{array} \right]^T \tag{2.2}$$

where the factor of two on the shear strains corresponds to the definition of engineering shear strain.

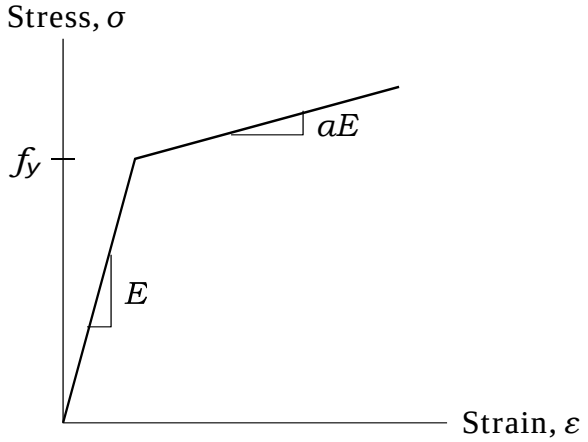


Figure 2.3: Uniaxial stress-strain response of the J2 plasticity model

For linear-elastic, isotropic behavior, the relationship between stress and strain is given by the constitutive matrix defined in terms of the elastic modulus, E, and Poisson ratio, v. After static condensation of the standard 6×6 elastic, isotropic matrix based on the condition $\sigma_{33} = 0$, the modified elastic constitutive matrix is

$$\mathbf{C} = \begin{bmatrix} \frac{E}{(1-\nu^2)} & \nu\frac{E}{(1-\nu^2)} & 0 & 0 & 0 \\ \nu\frac{E}{(1-\nu^2)} & \frac{E}{(1-\nu^2)} & 0 & 0 & 0 \\ 0 & 0 & G & 0 & 0 \\ 0 & 0 & 0 & G & 0 \\ 0 & 0 & 0 & 0 & G \end{bmatrix} \tag{2.3}$$

where the shear modulus, G, is defined by

$$G = \frac{E}{2(1+\nu)} \tag{2.4}$$

The J_2 plasticity model tracks the evolution of plastic strain

$$\boldsymbol{\varepsilon}^p = \begin{bmatrix} \varepsilon_{11}^p & \varepsilon_{22}^p & 2\varepsilon_{12}^p & 2\varepsilon_{23}^p & 2\varepsilon_{13}^p \end{bmatrix}^T \tag{2.5}$$

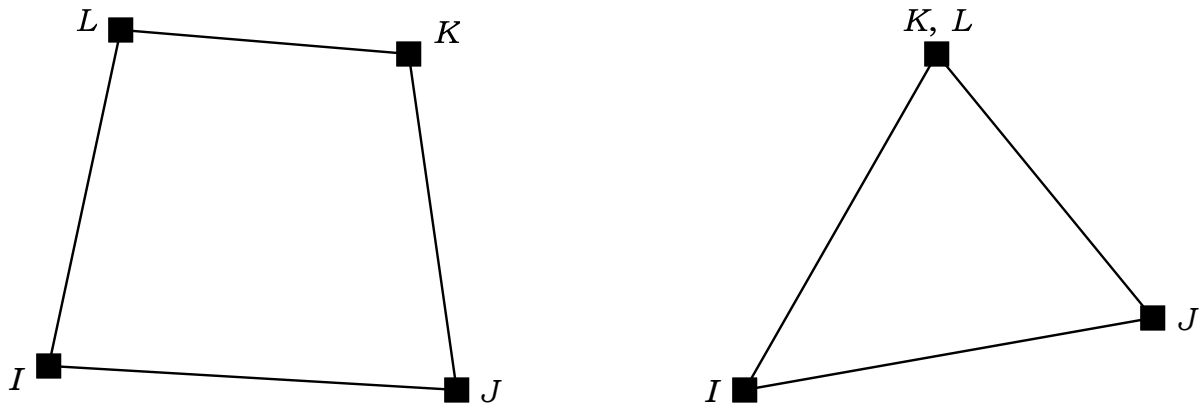
The state determination for the J_2 plate fi formulation is analogous to that presented in Simo and Hughes *(Simo and Hughes 1998)* for the plate stress condition. The matrix that relates deviatoric strains to deviatoric stresses for the plate fiber formulation is

$$\mathbf{P} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ -\frac{1}{3} & \frac{2}{3} & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix} \tag{2.6}$$

Algorithm details on the return mapping algorithm are found in Simo and Hughes *(Simo and Hughes 1998)*. This constitutive model is invoked in OpenSees using the *ndMaterial J2PlateFibre* command.

2.2 SHELL ELEMENT FORMULATION

Using the Delaunay triangulation of the plate area, a mesh of MITC4 shell elements *(Dvorkin and Bathe 1984)* is created. The MITC4 formulation is of a four-node shell element that uses a bilinear isoparametric formulation in combination with a modifi shear interpolation in order to improve thin-plate bending performance. Both membrane and drill effects are included in the formulation. To conform with the Delaunay triangulation, nodes *K* and *L* of each shell element map to the same node in the gusset plate model.



(a) Four noded shell element (b) Nodes assigned for gusset plate model

Figure 2.4: Assignment of nodes for the use of MITC4 shell elements in triangulated finite element mesh

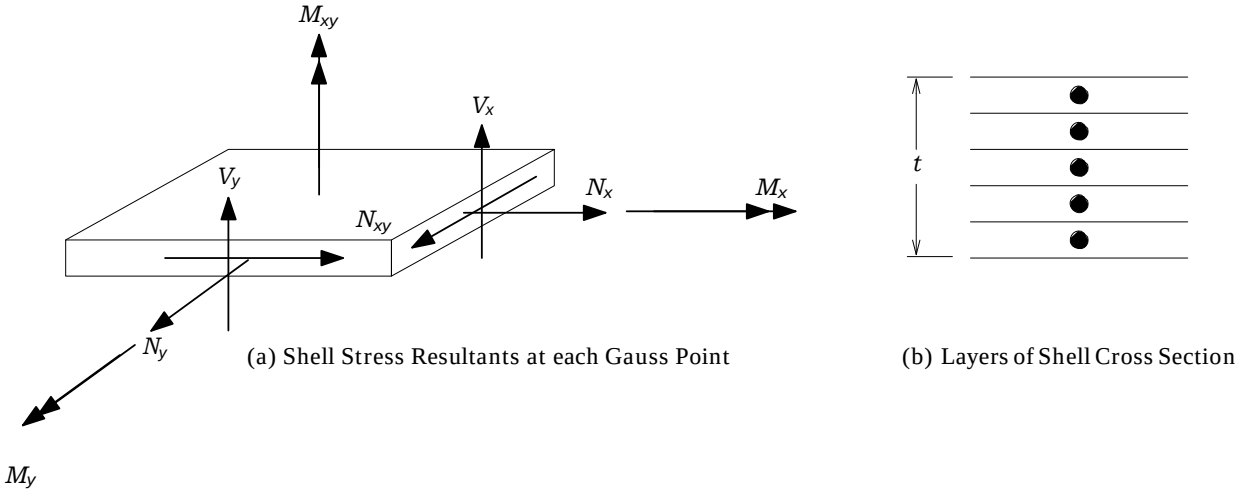
Each shell element contains four Gauss points where stress resultants are evaluated in order to obtain the element forces by integration over the element domain. The stress resultants are shown in Figure 2.5 (a) where membrane forces N_x, N_y, and N_{xy} act in the *x*-*y* plane of the element; bending moments M_x and M_y act about the *x*- and *y*-axes, respectively; shear forces V_x and V_y act along the *z*-axis; and the drilling moment M_{xy} acts about the *z*-axis. The stress resultants are obtained by integration of the aforementioned "plate fiber" stress-strain response through five layers as shown in Figure 2.5 (b).

This element is invoked in OpenSees using the element *ShellMITC4* command.

2.3 CONNECTED MEMBERS

Material and geometric properties of the members connected to each bolt group can be modifi by the user. The options for connected member properties are shown in Figure 2.6. There is a radio button for whether or not the selected member is connected to its associated bolt group. If the member is not connected, the input of subsequent member properties is disabled.

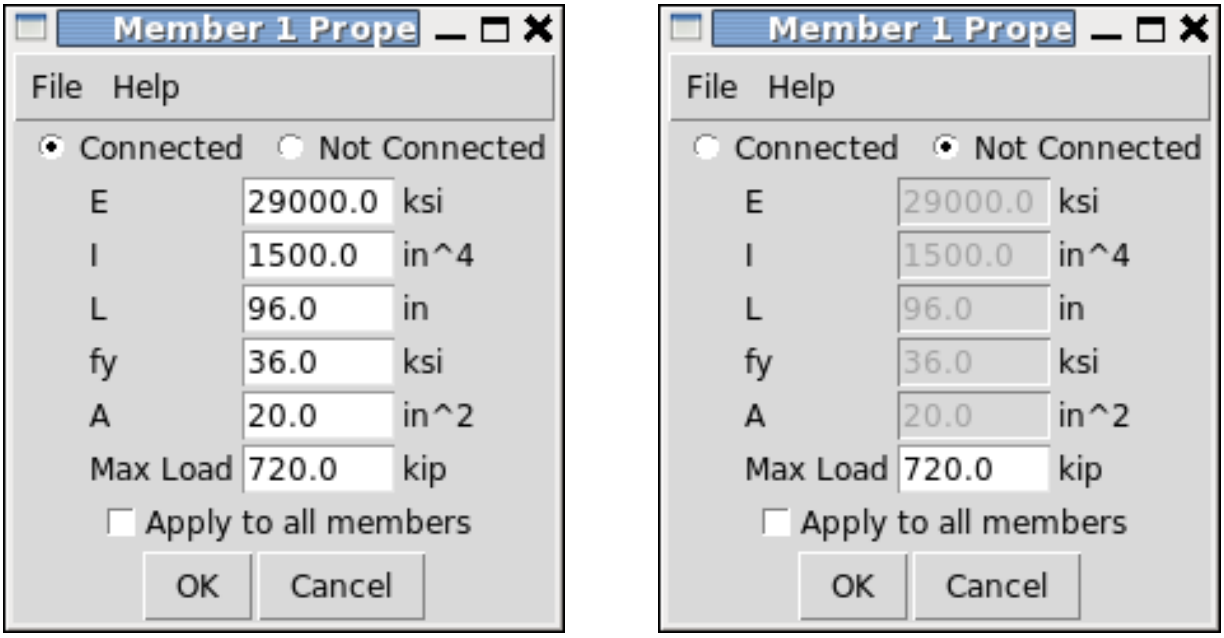
The member properties shown in Figure 2.6a are as follows:



(a) Shell Stress Resultants at each Gauss Point

(b) Layers of Shell Cross Section

Figure 2.5: Shell section stress resultants and layers through the shell thickness



(a) Connected Member (b) Member Not Connected

Figure 2.6: Window showing options for defining the properties of members connected to the gusset plate

- E – elastic modulus (default: 29,000 ksi, range: > 0)
- I – second moment of the cross-sectional area (default: 1500 in^4, range: > 0)
- L – beam length measured from the centroid of the associated bolt group (default: 96 in, range: > 0)
- f_y – yield stress (default: 36 ksi, range: > 0)
- A – cross-sectional area (default: 20 in^2, range: > 0)
- *Max Load* – maximum load applied to the gusset plate by this member (default: 720.0 kip, range: > 0)

Only the flexural stiffness of the connected member is considered in the gusset plate fi element model. The member is pinned at its far end, making its flexural contributions to the gusset plate proportional to $3EI/L$. In the finite element model, the cross-sectional area of each connected member is set to zero so that loads applied along the member centerline are not absorbed by the axial stiffness of the member and are instead transmitted entirely to the plate, as intended. The maximum load applied to the gusset plate by each member is equal to the smaller of the *Max Load* and the product of *fy* and *A* shown in Figure 2.6a.

If the *Apply to all members* box is checked, the properties assigned to the selected member are applied to all connecting members of the gusset plate model.

2.4 GUSSET PLATE ANALYSIS OPTIONS

Basic options for the gusset plate analysis are shown in Figure 2.7. These options control how the gusset plate fi element model is created along with the failure criteria for terminating the load-displacement analysis described in Chapter 3.

2.4.1 Maximum Element Area

The *Maximum Element Size* option shown in Figure 2.7 allows the analyst to specify the maximum area of each element created by the Delaunay triangulation procedure. The area of each triangular element will be less than or equal to the input value. The default value for the maximum element size is 2.0 in^2. Two fi element models created for the example gusset plate using a maximum area of 2.0 and 4.0 in^2 is shown in Figure 2.8.

Consistent with fi element theory, the computed results will converge to a solution as the number of elements increases. On the other hand, the computational expense, i.e., the analysis run time, increases accordingly when using more elements.

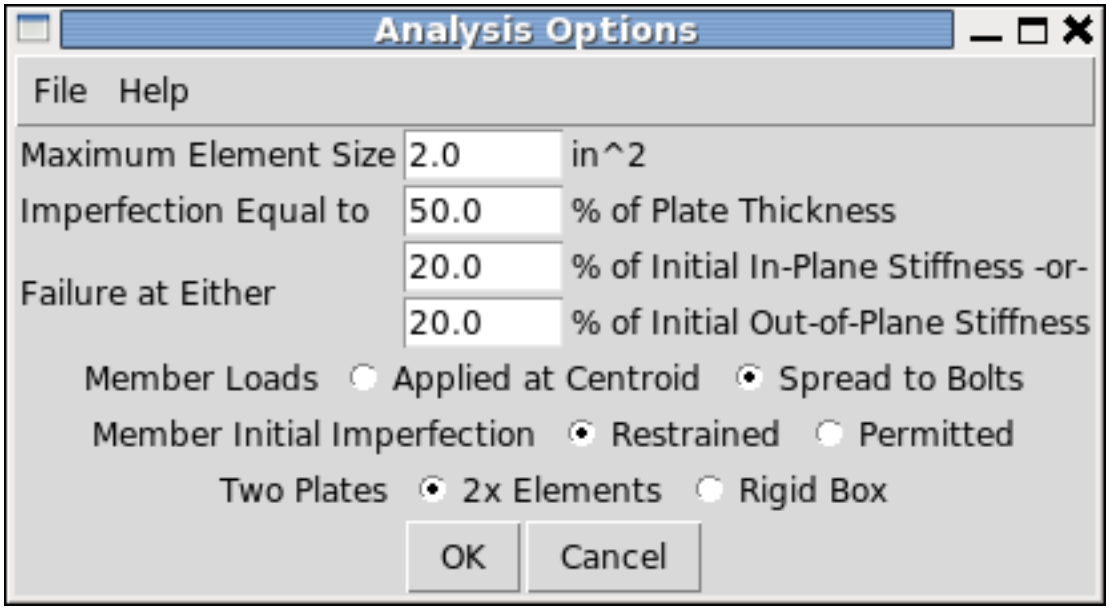
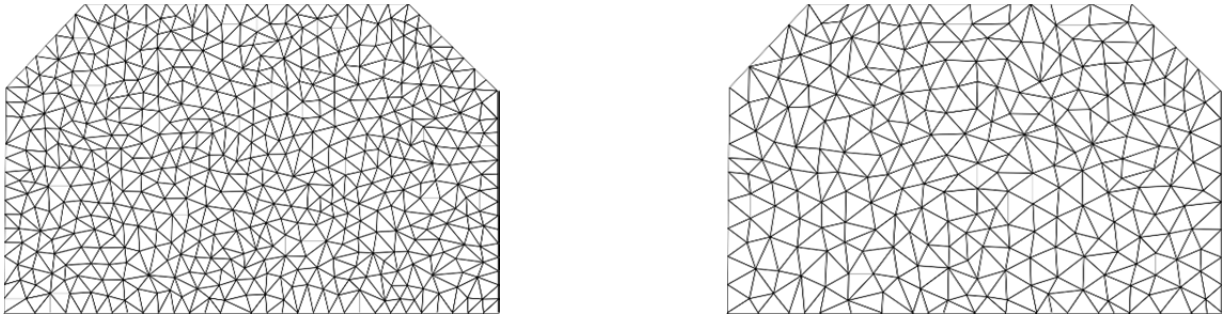


Figure 2.7: Window showing gusset plate analysis options



(a) A_{max}=2.0 in^2 (b) A_{max}=4.0 in^2

Figure 2.8: Gusset plate finite element model showing different element area sizes

2.4.2 Member Initial Imperfection

To trigger failure of the gusset plate under member loading, the plate is given an initial out-of-plane imperfection (deformed shape) based on eigenvalue analysis of the initial stiff- ness matrix, **K**, of the gusset plate finite element model. Using the *eigen* command in OpenSees, the eigenvector, $\boldsymbol{\varphi}$, associated with the smallest eigenvalue, λ, that both satisfy the eigenvalue equation is chosen for the deformed shape

$$\mathbf{K}\boldsymbol{\varphi} = \lambda\boldsymbol{\varphi} \tag{2.7}$$

The eigenvector that corresponds to the smallest eigenvalue, collectively the smallest eigen- pair, is chosen for the deformed shape as this corresponds to the mode that requires the least strain energy to deform the plate. Each node in the gusset plate model is displaced in the z-direction according to the eigenvector

$$z_i = z^o{}_i + (Ct_{plate}) \quad \frac{\varphi_{zi}}{\varphi_{zi,max}} \tag{2.8}$$

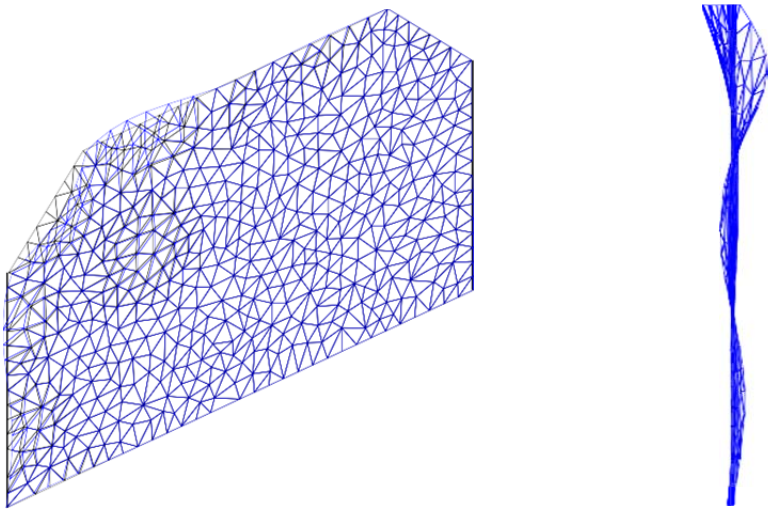
where z^o_i is the initial z-coordinate of node i, φ_{zi} is the z-component of the eigenvector corresponding to node i, t_{plate} is the user-specified plate thickness, and $\varphi_{zi,max}$ is the maximum value of φ_{zi} over all nodes in the gusset plate model. The variable C specifies the maximum out-of-plane imperfection as a fraction (percentage) of the plate thickness, which is the input value in the *Imperfection Equal to ___% of Plate Thickness* field of the analysis options shown in Figure 2.7. The default value for this parameter is 50%, i.e., the maximum out-of-plane imperfection for the plate is 50% of the plate thickness.

Two options are provided for computing the initial deformed shape based on the participation of loaded members and are listed under Member Initial Imperfection field shown in Figure 2.7. The first option is Restrained, which restrains the out-of-plane motion for all bolt groups prior to forming the stiffness matrix, K, and computing its smallest eigenpair according to Equation (2.7). With this option, the eigenvalue analysis is performed once and the resulting deformed shape is used for all load cases applied to the gusset plate. The initial deformed shape using this option for the example gusset plate is shown in Figure 2.9.

The second option is *Permitted*, which allows out-of-plane motion of the bolt groups associated with the members to be loaded in each load case. If flexible beam members are connected, their stiffness is included in **K** prior to computing the smallest eigenpair according to Equation (2.7) and the member stiffness will influence the initial deformed shape of the plate. With this option, the smallest eigenpair is re-computed for each load case applied to the gusset plate as the loaded bolt groups generally change from case to case. The initial deformed shape using this option for the example gusset plate is shown in Figure 2.10 for the case of one loaded member.

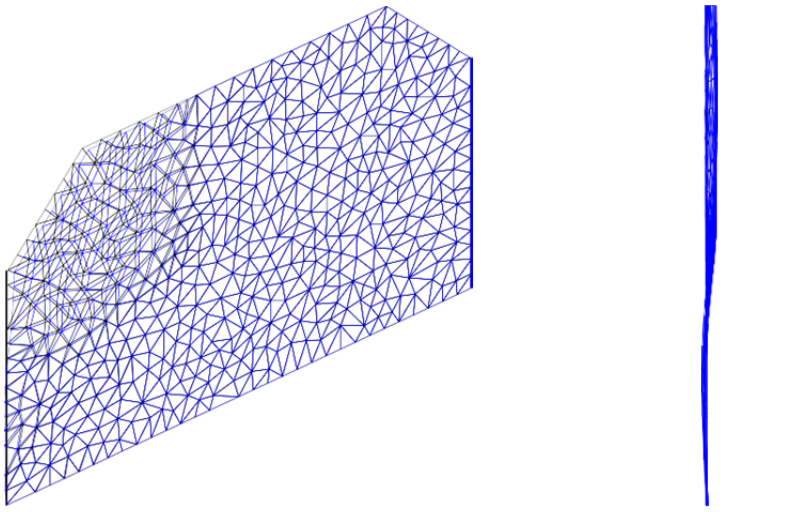
2.4.3 Member Loads

The *Member Loads* option shown in Figure 2.7 determines how member loads are applied to the gusset plate. The analysis program *allows two choices*. First, is *Apply at Centroid*,



(a) Perspective (b) Profile

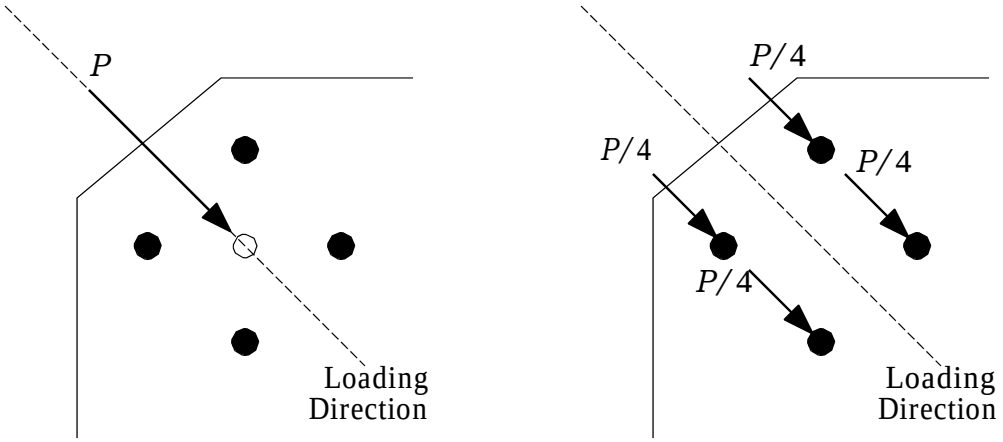
Figure 2.9: Member initial imperfection with motion restrained for all members. (z_{max} = $0.5t_{plate}$ and magnification factor = 5.0)



(a) Perspective (b)Profile

Figure 2.10: Member initial imperfection with motion permitted for one loaded member and restrained for all other members. ($z_{max} = 0.5t_{plate}$ and magnification factor = 5.0)

which applies the entire member load at the geometric centroid of the bolt group, as depicted in Figure 2.11 (a). Second is *Spread to Bolts*, which evenly distributes the member load among the bolts in the group, as shown in Figure 2.11 (b). The default member load approach is *Spread to Bolts* as this mitigates the stress concentrations that would appear if the entire member load were applied at a single point.



(a) Member Load Applied at Centroid (b) Member Load Spread to Bolts

Figure 2.11: Options for applying member loads to the gusset plate

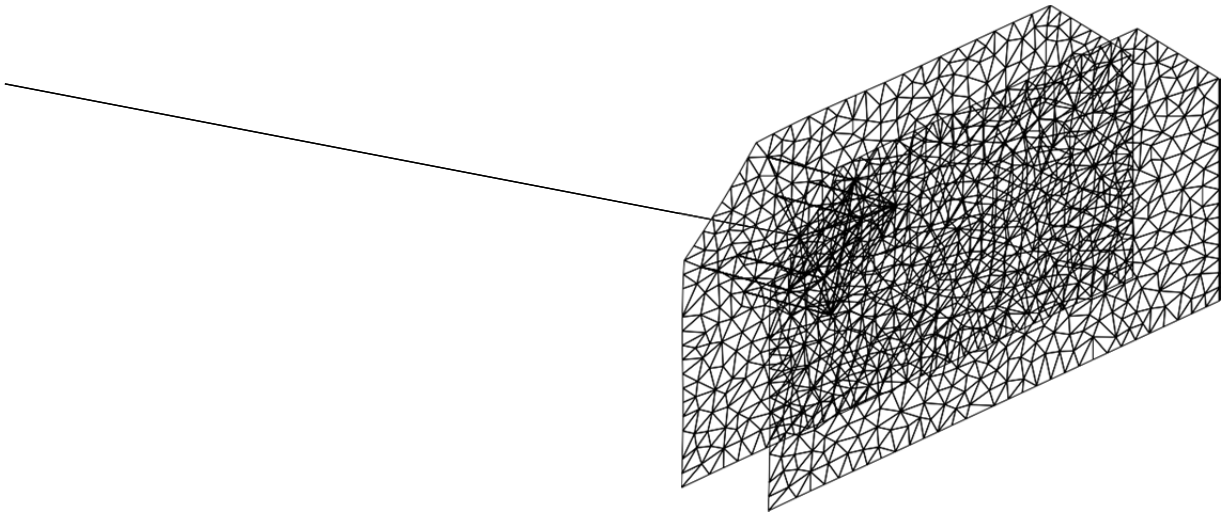
2.4.4 Double Gusset Plate

The final analysis option shown in Figure 2.7 is the modeling approach for the case where the model consists of two plates. The fi option is 2x Elements where the mesh of shell elements is "doubled up" in the nodal mesh. This option is analogous to two plates with zero separation and full compatibility of deformations.

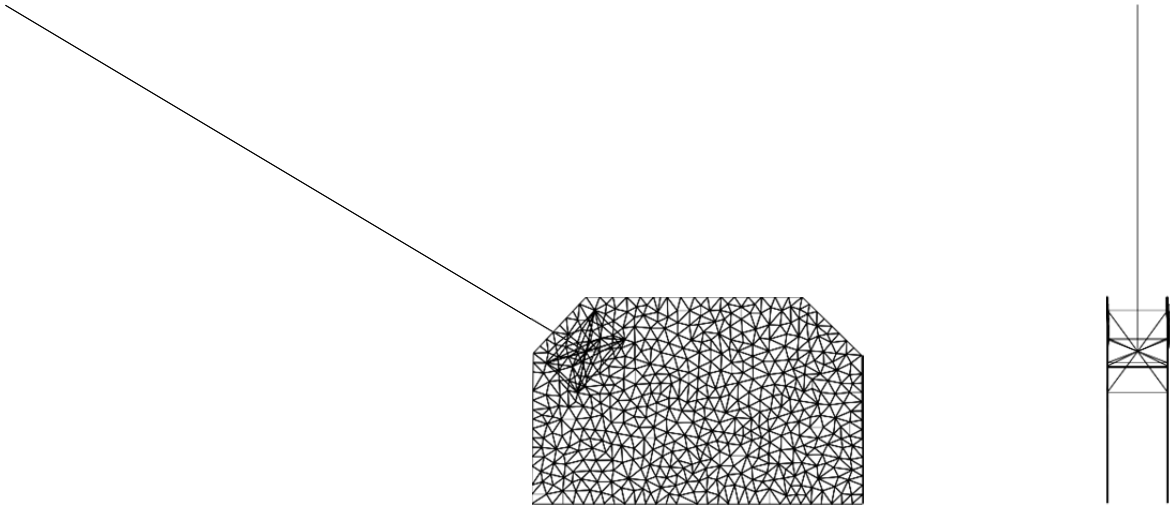
The second option is *Rigid Box*, where the mesh of shell elements and nodes is repeated at the plate separation distance specifed in the *Plate Properties* window of Figure 2.2. For each bolt group, beam elements connect each bolt on one plate to its counterpart on the opposite plate and to the centroid of the rigid box, as shown in Figure 2.12c. This figure also shows that the loaded member is connected at the centroid of the rigid box and that a rigid box is created for only the loaded members in each load case. The elastic properties assigned to the beams that comprise the rigid box are taken from the material and cross-section properties of the associated member input via the window shown in Figure 2.6.

2.5 LOAD COMBINATIONS

The analysis software allows the user to specify an arbitrary number of load combinations to be applied to the gusset plate. The default load combinations correspond to each member loaded individually, e.g., if there are fi e members, there are fi e load combinations, as shown in Figure 2.13. The user can add a new load combination by clicking the *Add New* button and deleted selected load combinations by checking the box of the load combinations to be deleted then pressing the *Delete Selected* button.



(a) Perspective



(b) Plan

(c) Profile

Figure 2.12: Gusset plate finite element model using the *Rigid Box* option with one loaded member

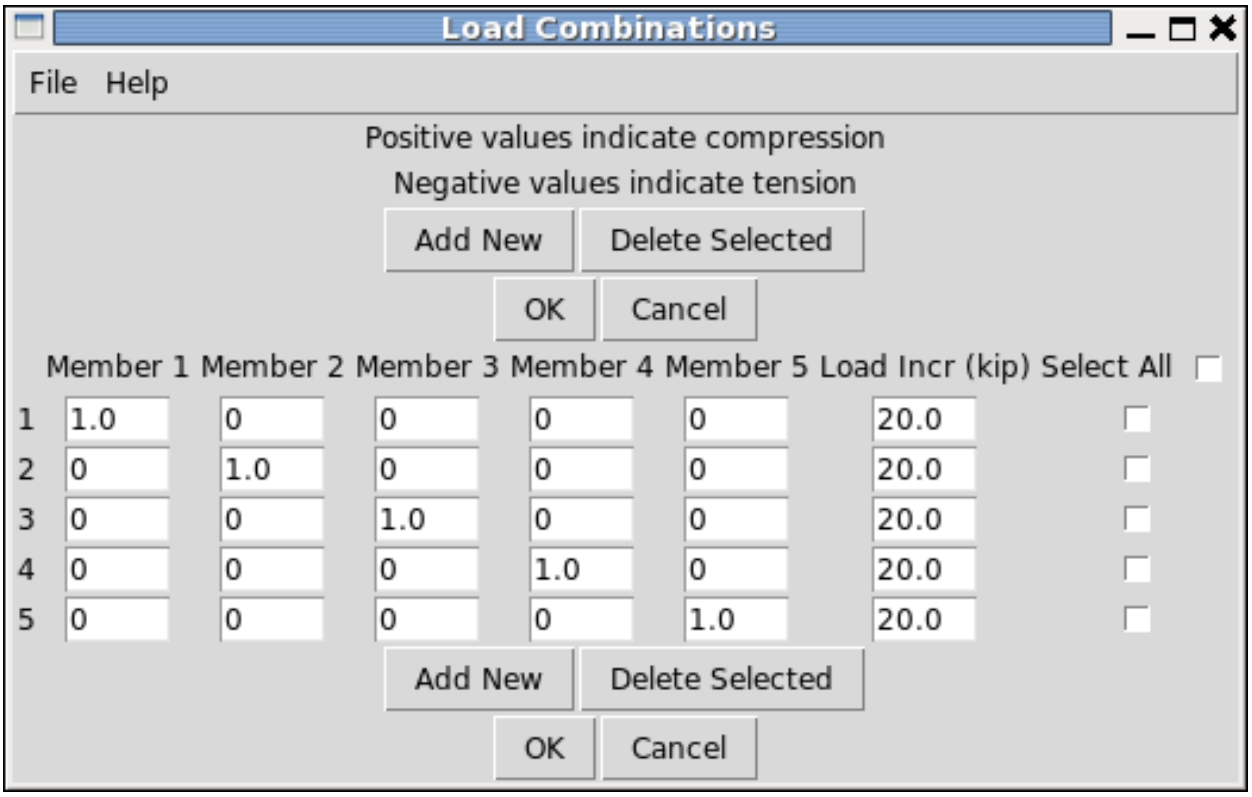


Figure 2.13: Window showing load combinations to be applied to the gusset plate model

The input for each load combination (each row of input in Figure 2.13 is a pattern of reference member loads and a load increment, $\Delta\lambda$. The incremental load applied to the gusset plate is equal to the specified load increment, $\Delta\lambda$, multiplied by a normalized reference load vector

$$\Delta\mathbf{P} = \Delta\lambda \frac{\mathbf{P}_{ref}}{\|\mathbf{P}_{ref}\|} \tag{2.9}$$

The vector $\mathbf{P}_{ref}$ is obtained from user input via the window shown in Figure 2.13

$$\mathbf{P}_{ref} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_N \end{bmatrix} \tag{2.10}$$

where N is the number of members and P_i is the reference load of the i^{th} member. The magnitude of the reference load vector is

$$\|\mathbf{P}_{ref}\| = \sqrt{\sum_{i=1}^{N} P_i^2} \tag{2.11}$$

With this normalization of the member loads, the total load increment applied to the gusset plate is equal to $\Delta\lambda$ regardless of the number of loaded members. The default load increment, $\Delta\lambda$, is 20 kip for each load combination. In addition, as indicated in Figure 2.13, positive values of reference member load indicate compression, or pushing on the gusset plate, while negative values correspond to tension, or pulling on the gusset plate.

3.0 FINITE ELEMENT ANALYSIS

The Gusset Plate analysis software performs nonlinear static analysis of the gusset plate model for the given load combinations. Details of the nonlinear fi element solution and the results generated by the software are described in this chapter.

3.1 SOLUTION STRATEGY

Although several solution strategies are available in the OpenSees fi element framework, the Gusset Plate analysis software uses a specific set of strategies chosen for efficient solution of gusset plate response.

- **Equilibrium Solution** – The system of nonlinear equilbrium equations is solved at each load step using a Krylov-subspace accelerated Newton algorithm.

  algorithm KrylovNewton

  As the size of the gusset plate fi element mesh can be large, this Krylov-Newton algorithm strikes a good balance between the Newton-Raphson and Modified Newton algorithms in terms of computational efficiency and convergence rate *(Scott and Fenves 2010)*.

- **Convergence Test** – The convergence criterion of the Krylov-Newton algorithm is specified in terms of the norm of the incremental displacement vector relative to what it was at the first iteration of the load step.

  test RelativeNormDispIncr

  This option is chosen due to the relatively stiff beam elements used for the Rigid Box option where small changes in displacement can lead to large changes in forces and pose challenges for convergence criteria based on residual forces. The convergence tolerance is 10^{-4} and the maximum number of iterations is 50.

- **Linear Equation Solver** – At each iteration of the Krylov-Newton equilibrium solu- tion algorithm, a linear system of equations is formed. The solution to this system of equations is obtained using the UMFPACK library available in OpenSees.

  system UmfPack

  The UMFPACK library is a multi-frontal solver that is efficient for large, sparse systems of simultaneous equations.

- **Load Steps** – The simulation advances through pseudo-time using constant load in- crements in a load-control integration method.

  integrator LoadControl

  The load increment is specified for each load combination by the user, as shown in Figure 2.13. In addition, the analysis will reduce the load increment by up to 50% from the user-specified value when the equilibrium solution takes more than six iterations to converge.

- **Constraints** – Multi-point constraints, e.g., slaving all nodes in a bolt group to have the same in-plane displacement, are enforced using the Transformation method (Cook et al. 1989).

  constraints Transformation

Further details on the selected solution strategies can be found in the given references and at the OpenSees website *(McKenna et al. 2000)*.

3.2 ANALYSIS PROGRESS

After loading a gusset plate image file the user launches the fi element analysis by pressing the Analyze button on the main window shown in Figure 1.4. The analysis progress window shown in Figure 3.1 becomes visible and displays the load combination number and the current load applied to each member. The user can press the Stop button at any time in order to terminate the finite element analysis and return to editing of the gusset plate properties and analysis options.

In addition to the member loads shown in Figure 3.1, stress contours over the gusset plate are displayed and updated in the main window as shown in Figure 3.2. The contour is of the J_2 stress averaged over each element from the four Gauss points of the element.

Using the gusset plate finite element model with user-defined parameters along with the user-specified load combinations, the analysis software loops through all combinations performing nonlinear static analysis until one of the following failure states is reached:

- The load applied via any member exceeds the member's *Max Load* input via Figure 2.6.

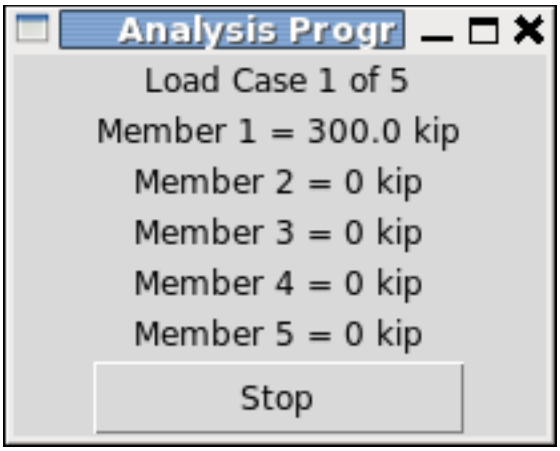


Figure 3.1: Analysis progress window showing current member loads for each load combination

- The load applied via any member exceeds the member's plastic axial capacity, as defined by the product of *fy* (yield stress) and *A* (cross-section area) input via Figure 2.6.

- The slope of the in-plane load-displacement response for any member reduces to a specified fraction of the initial stiffness. The in-plane displacement is recorded at the centroid of each bolt group.

- The slope of the out-of-plane load-displacement response for any member reduces to a specified fraction of the initial stiffness. The out-of-plane displacement is recorded at the node with the largest eigenvector component, $\varphi_{z,max}$. In this case, the load-displacement response for each member has the same displacements but potentially different loads.

- The numerical solution for equilbrium fails to converge. This can occur when the load increment specified in Figure 2.13 is too large, as well as due to a variety of other factors.

When one of these failure states is reached for a load combination, the software advances to the next load combination and performs a new analysis.

3.3 REPORT GENERATION

After the analysis is complete, or has been terminated by the user, the software will create a PDF report summarizing the plate properties, analysis options, load combinations, and analysis results. To generate a report, the user clicks the *Create Report* button in the Analysis Complete window shown in Figure 3.3 that appears after the analysis is complete. This will prompt the user to select a directory and filename to which the report will be saved, as shown in Figure 3.4.

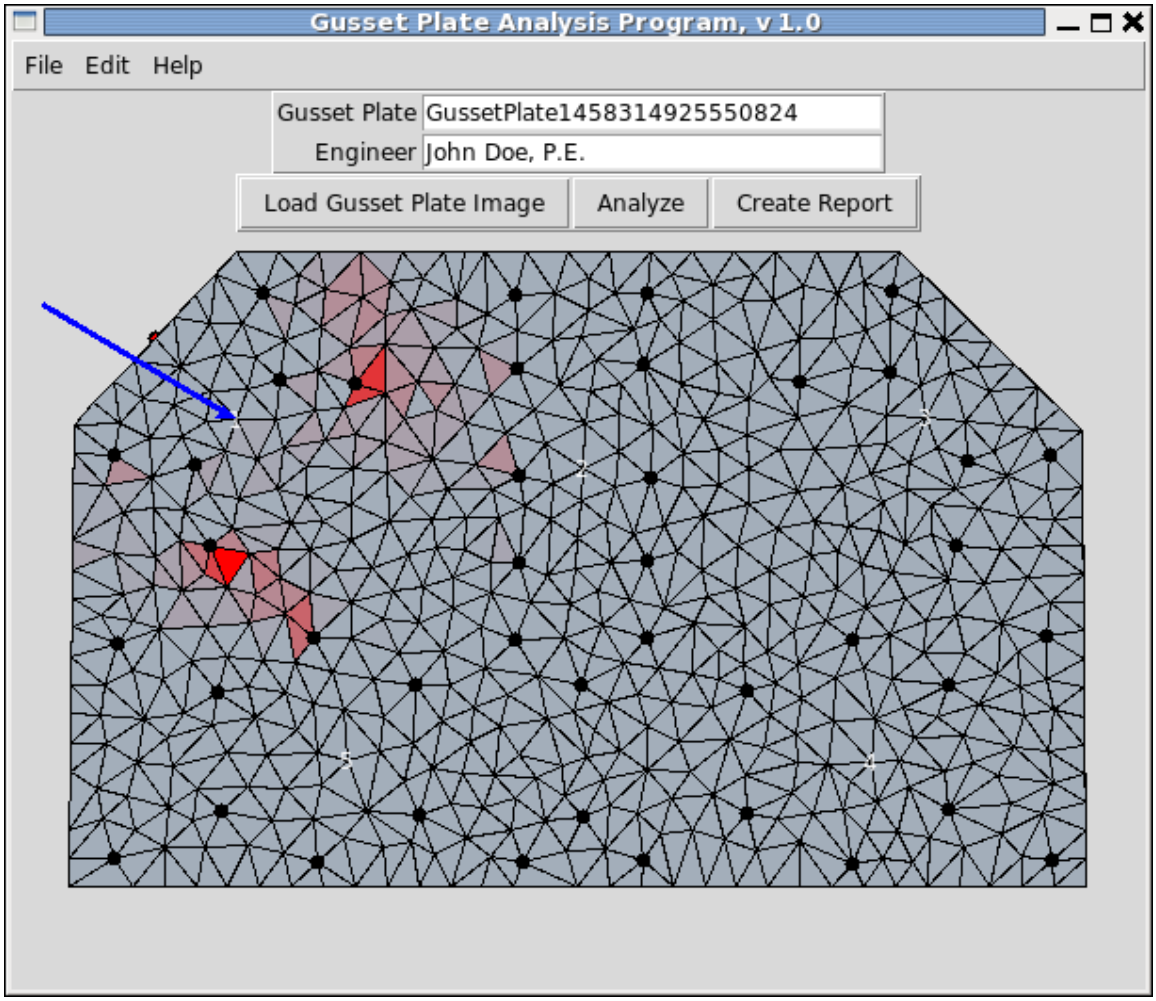


Figure 3.2: Stress contours shown during finite element analysis

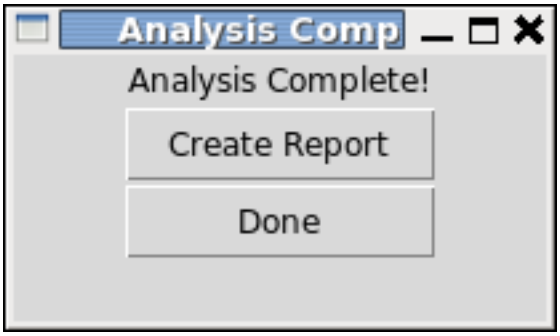


Figure 3.3: Window that appears after gusset plate analysis is complete

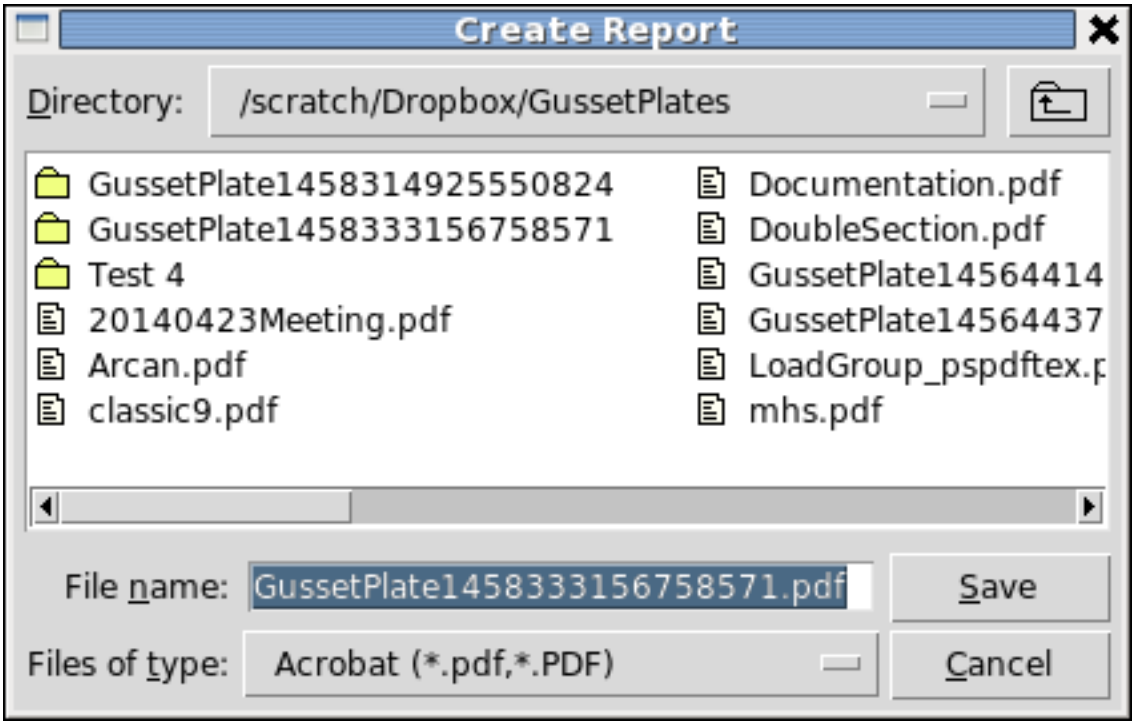
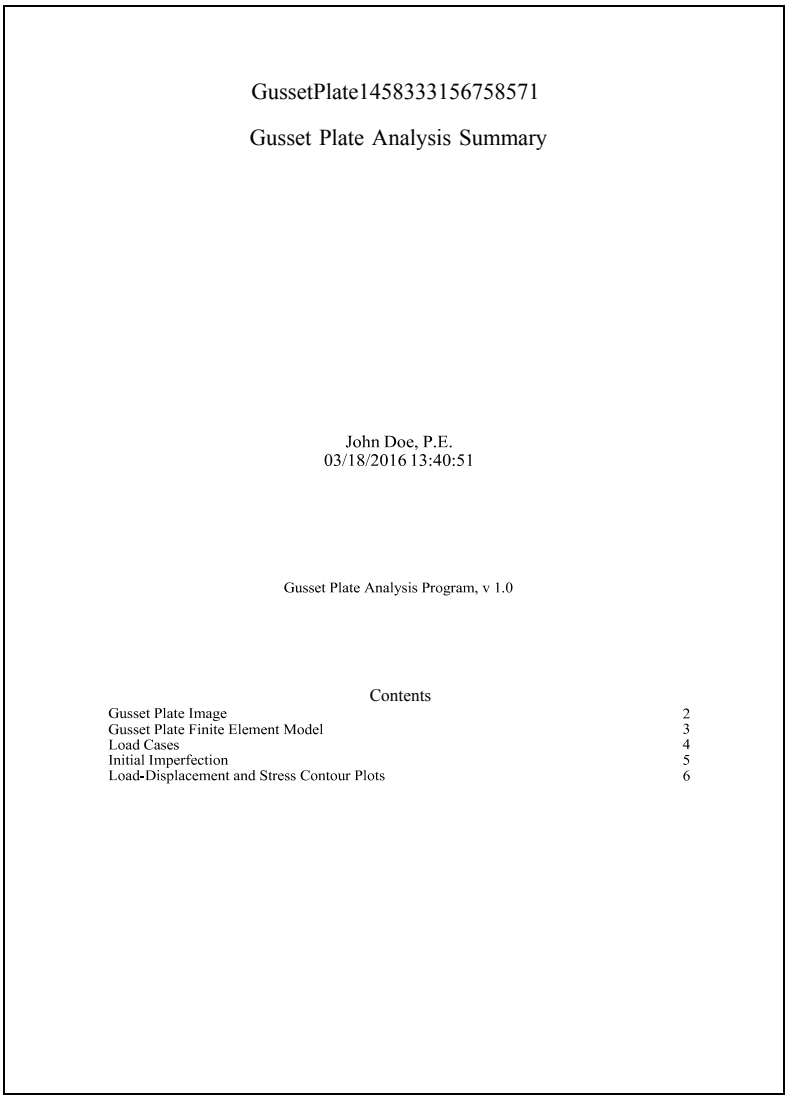


Figure 3.4: Window that appears after gusset plate analysis is complete

3.3.1 Title Page and Table of Contents

The title page echoes information from the main window (Figure 1.2) including the gusset plate name, analyst name, and date of analysis. There is also a table of contents as shown in Figure 3.5.



GussetPlate1458333156758571

Gusset Plate Analysis Summary

John Doe, P.E.
03/18/2016 13:40:51

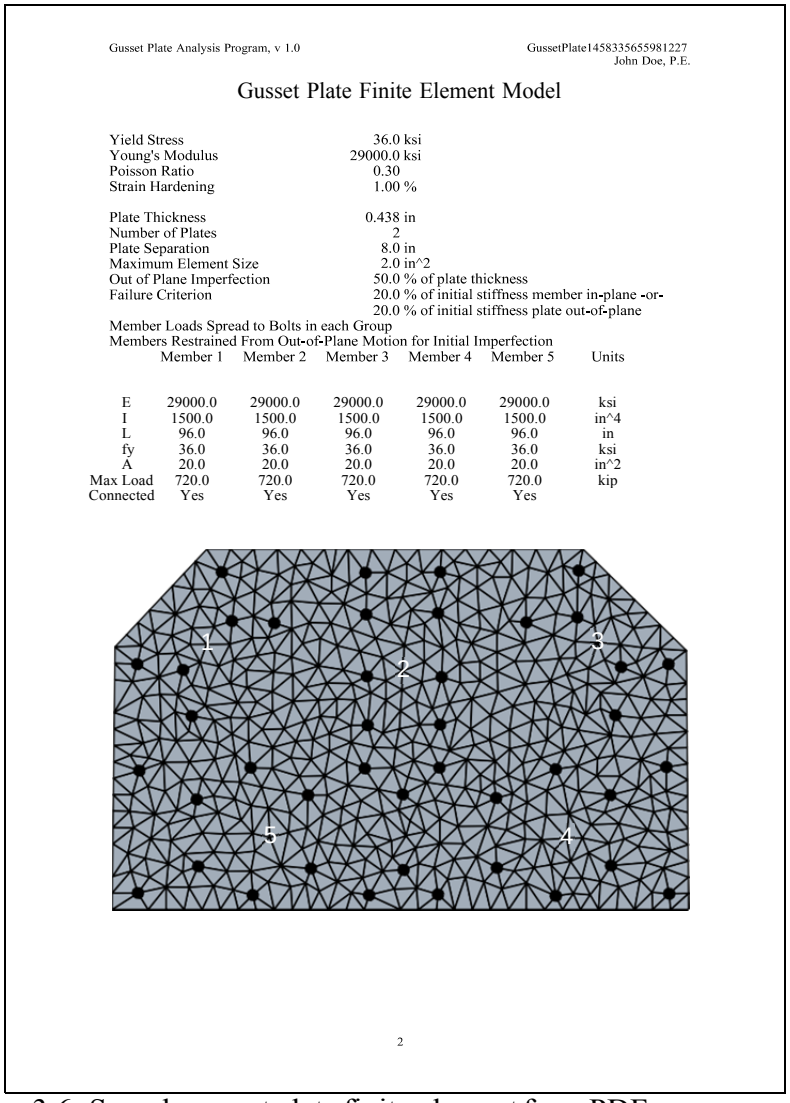
Gusset Plate Analysis Program, v 1.0

Contents

Figure 3.5: Sample title page and table of contents from PDF summary report

3.3.2 Gusset Plate Finite Element Model

The second page of the report shows the gusset plate material properties, properties of connected members, and analysis options, as shown in Figure 3.6. In addition, an image of the finite element mesh is shown along with the location of bolt group centroids.



Gusset Plate Analysis Program, v 1.0 GussetPlate1458335655981227
John Doe, P.E.

Gusset Plate Finite Element Model

Yield Stress	36.0 ksi
Young's Modulus	29000.0 ksi
Poisson Ratio	0.30
Strain Hardening	1.00 %
Plate Thickness	0.438 in
Number of Plates	2
Plate Separation	8.0 in
Maximum Element Size	2.0 in^2
Out of Plane Imperfection	50.0 % of plate thickness
Failure Criterion	20.0 % of initial stiffness member in-plane -or- 20.0 % of initial stiffness plate out-of-plane

Member Loads Spread to Bolts in each Group
Members Restrained From Out-of-Plane Motion for Initial Imperfection

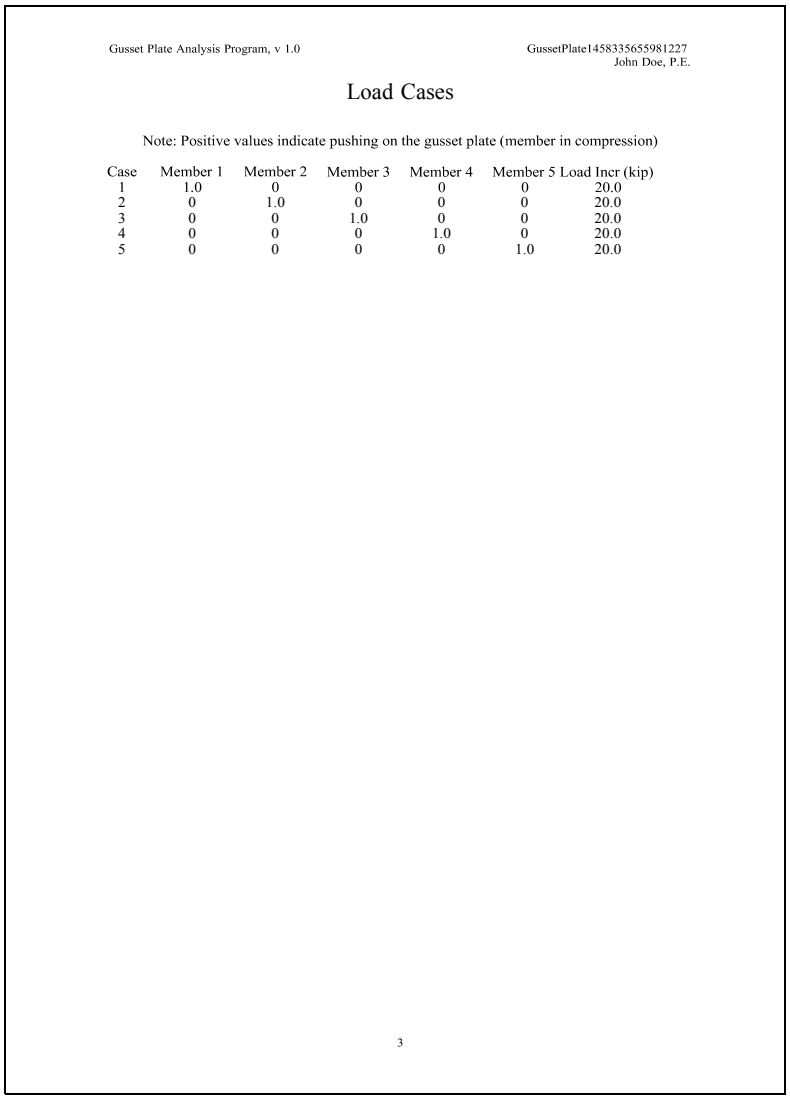
	Member 1	Member 2	Member 3	Member 4	Member 5	Units
E	29000.0	29000.0	29000.0	29000.0	29000.0	ksi
I	1500.0	1500.0	1500.0	1500.0	1500.0	in^4
L	96.0	96.0	96.0	96.0	96.0	in
fy	36.0	36.0	36.0	36.0	36.0	ksi
A	20.0	20.0	20.0	20.0	20.0	in^2
Max Load	720.0	720.0	720.0	720.0	720.0	kip
Connected	Yes	Yes	Yes	Yes	Yes	

2

Figure 3.6: Sample gusset plate finite element from PDF summary report

3.3.3 Load Combinations

The third page of the report shows the load combinations applied to the gusset plate.



Gusset Plate Analysis Program, v 1.0 GussetPlate1458335655981227
John Doe, P.E.

Load Cases

Note: Positive values indicate pushing on the gusset plate (member in compression)

Case	Member 1	Member 2	Member 3	Member 4	Member 5	Load Incr (kip)
1	1.0	0	0	0	0	20.0
2	0	1.0	0	0	0	20.0
3	0	0	1.0	0	0	20.0
4	0	0	0	1.0	0	20.0
5	0	0	0	0	1.0	20.0

3

Figure 3.7: Sample load combinations from PDF summary report

3.3.4 Initial Imperfection

If the *Member Initial Imperfection* option shown in Figure 2.7 is set to Restrained, the fourth page of the report shows the initial imperfection that is used for all load combinations. If this option is set to *Permitted*, the initial imperfections are shown in subsequent pages of the report for each load combination.

The color gradient on this page indicates the magnitude of the initial imperfection and the red dot shows the node with the large out-of-plane eigenvector component. In addition, paths to two files are

shown on this page. This first file stores nodal and element connectivity of the gusset plate model for import into third party software and the second fi stores the out-of-plane imperfection for each node.

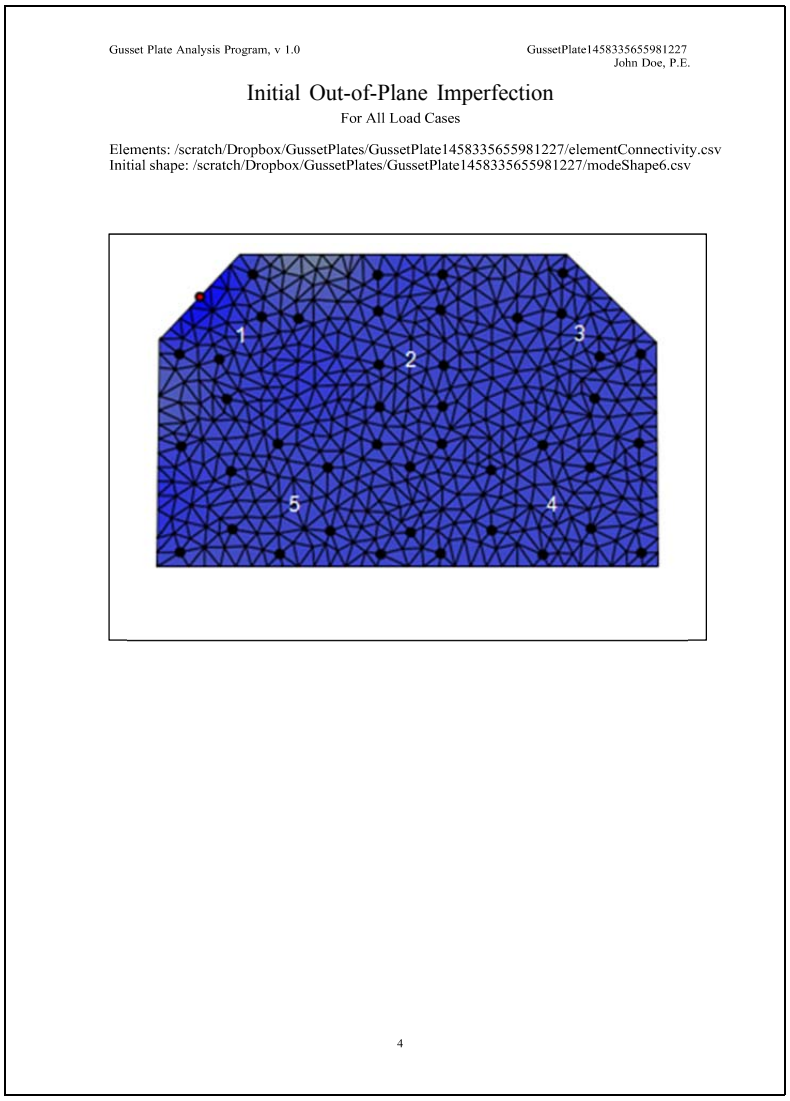


Figure 3.8: Sample initial imperfection (members restrained) from PDF summary report

3.3.5 Load-Displacement and Stress Contour Plots

The remaining pages of the report contain analysis results for each load combination. There are four pages per load combination containing the member in-plane load-displacement response, the member out-of-plane load-displacement response, the member load vs. plate out-of-plane displacement response, and the figure stress contour. If the *Member Initial Imperfection* option shown in Figure 2.7 is set to *Permitted*, a fifth page is added for each load combination showing the initial imperfection.

3.3.5.1 *Member In-Plane Load-Displacement*

This page shows the in-plane load-displacement computed at the bolt group centroid of each loaded member(s). The final member load(s) are shown, along with the path to a .csv file that contains the load-displacement data.

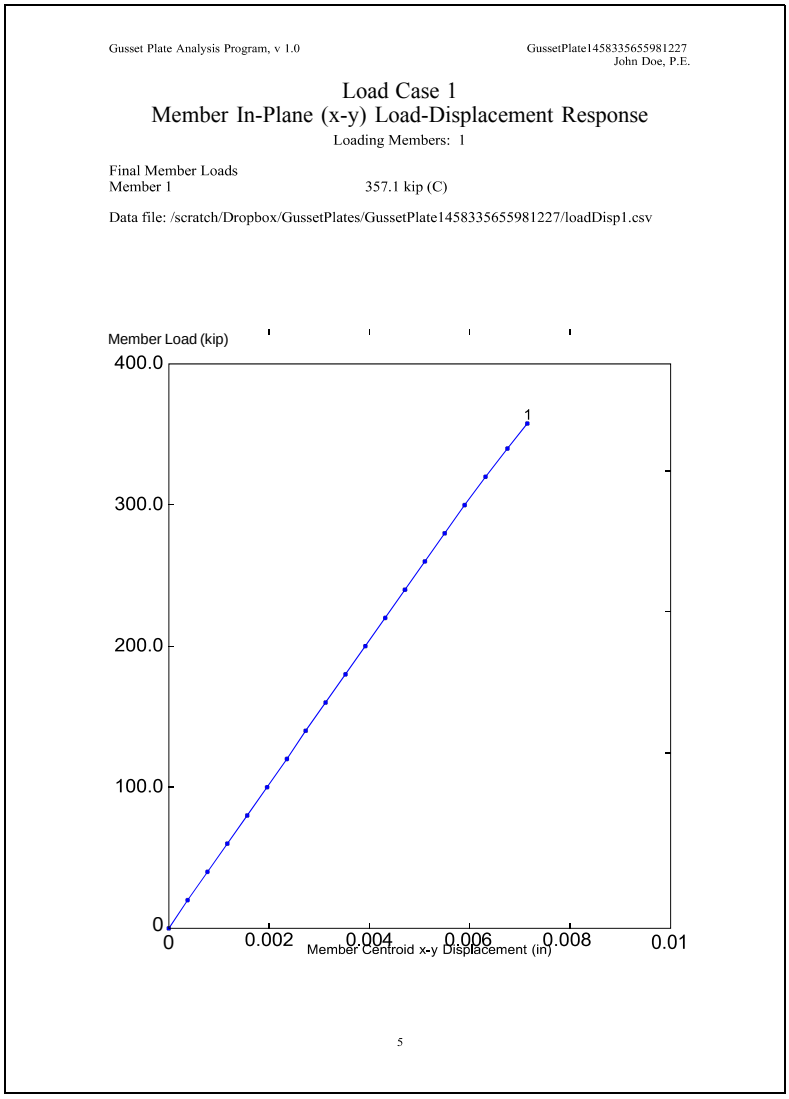


Figure 3.9: Sample initial imperfection (members restrained) from PDF summary report

3.3.5.2 *Member Out-of-Plane Load-Displacement*

This page shows the out-of-plane load-displacement computed at the bolt group centroid of each loaded member(s). The final member load(s) are shown, along with the path to a .csv file that contains the load-displacement data.

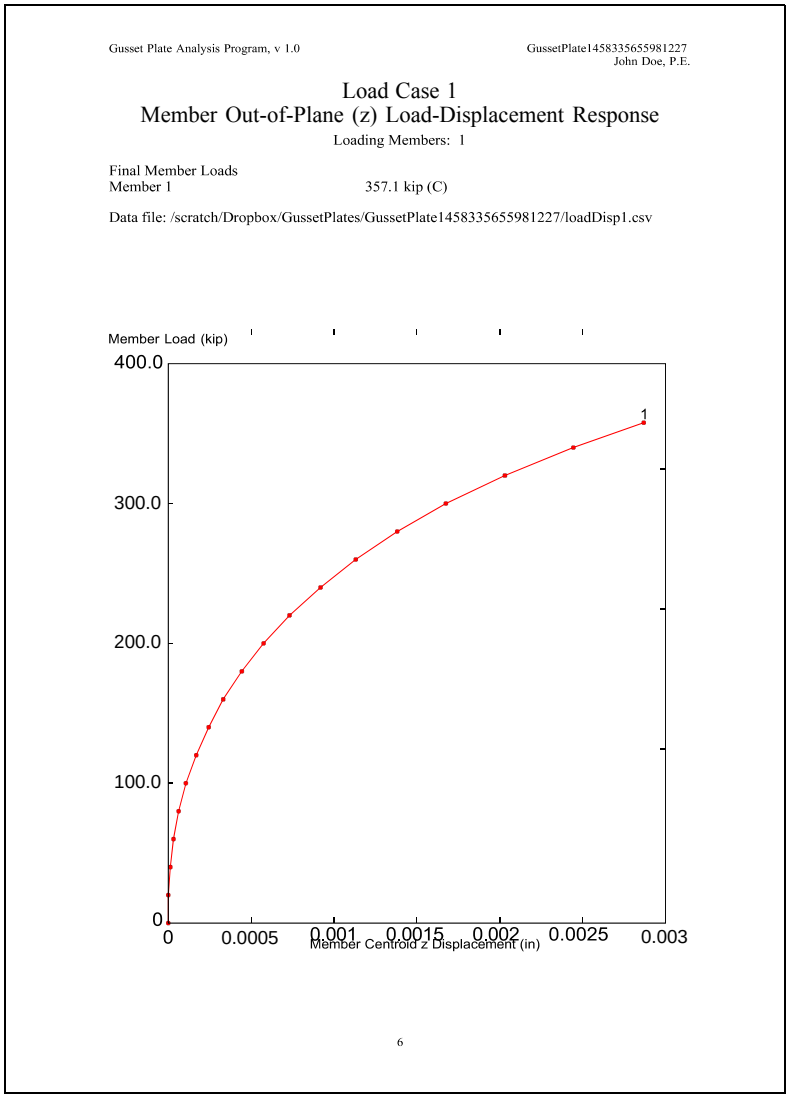


Figure 3.10: Sample member out-of-plane load-displacement plot from PDF summary report

3.3.5.3 *Member Load vs. Plate Out-of-Plane Displacement*

This page shows the relationship between member load and out-of-plane displacement at the gusset plate node with the largest initial imperfection. The final member load(s) are shown, along with the path to a .csv file that contains the load-displacement data.

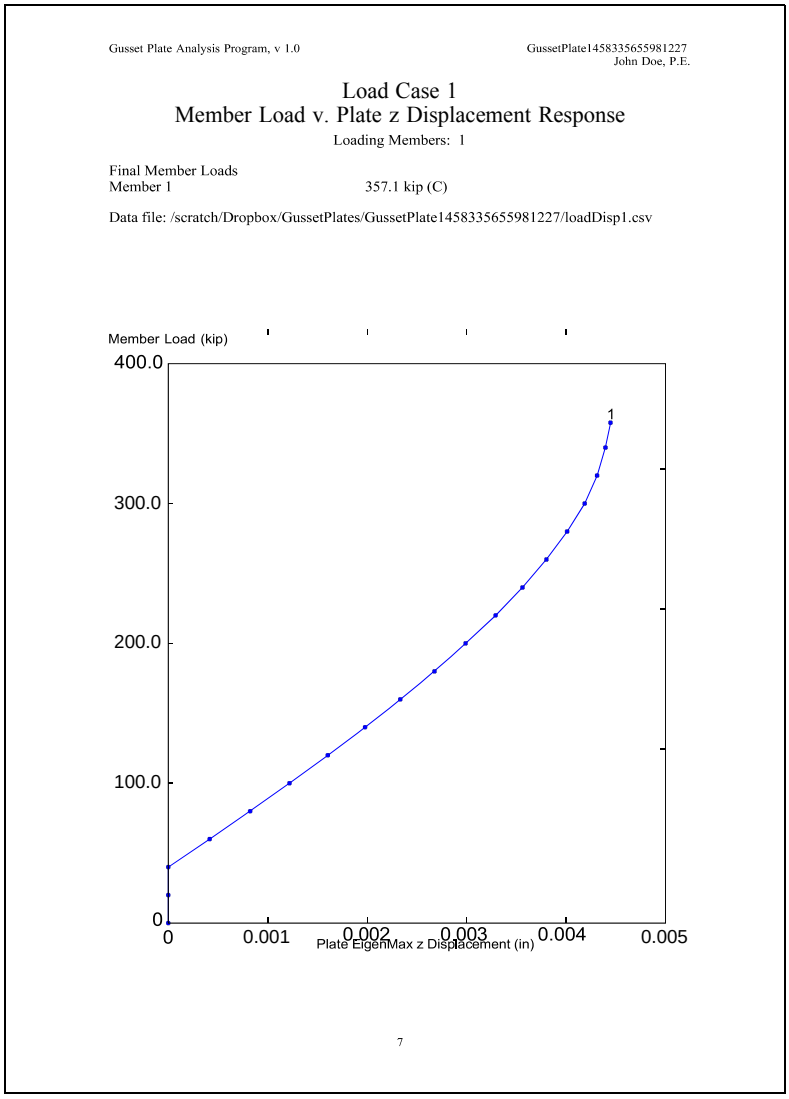


Figure 3.11: Sample member out-of-plane load-displacement plot from PDF summary report

3.3.5.4 *Final Stress Contour*

This page shows the stress contour (J_2 component) of the gusset plate at the fi load step for the current load combination. In addition, paths to two files are shown on this page. This first file stores nodal and element connectivity of the gusset plate model for import into third party software and the second fi stores the final out-of-plane displacement for each node.

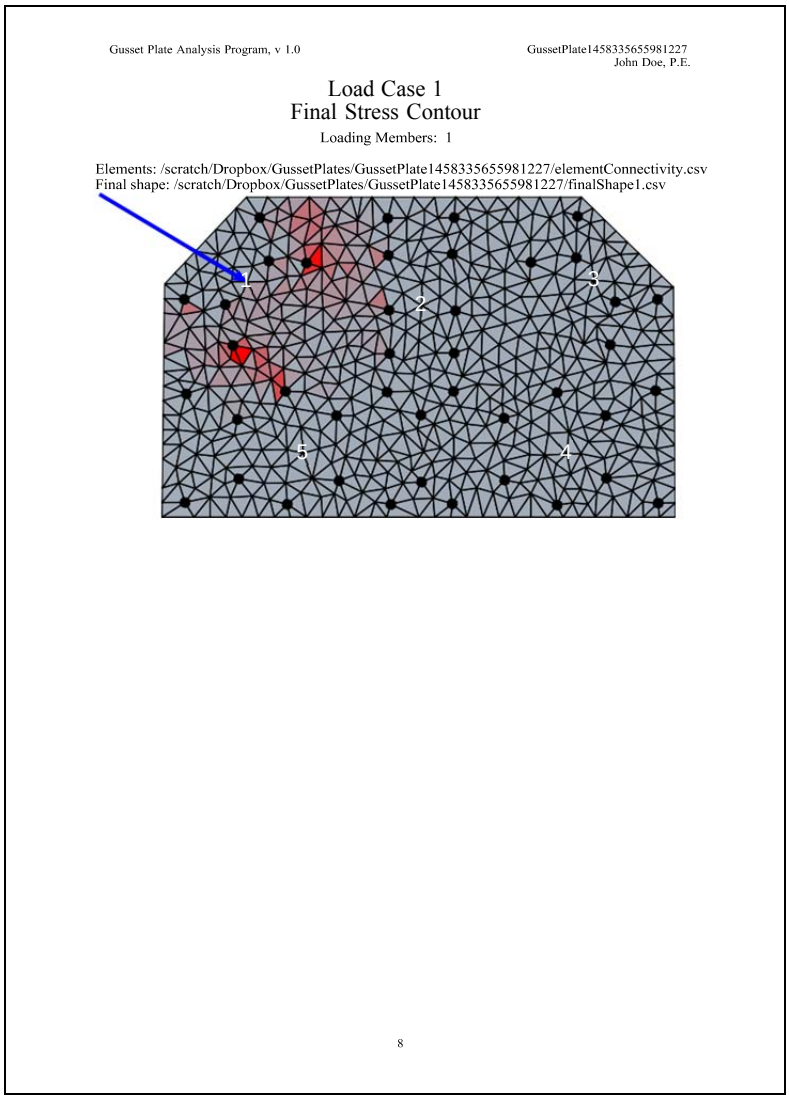


Figure 3.12: Sample final stress contour from PDF summary report

The PDF report pages shown in Figures 3.9 through 3.12 are generated for the remaining load combinations.

PART 3: IBIM (INTERACTIVE BRIDGE IMAGE MANAGEMENT): A WEB GIS TOOL FOR MANAGING BRIDGE INSPECTION IMAGERY AND METADATA

Farid Javadnejad
Daniel T. Gillins, Ph.D., P.L.S.
and
Christopher Higgins, Ph.D., P.E.

1.0 INTRODUCTION

The proposed collection of large numbers of high-fidelity quantitative inspection images of bridges members and connections requires a new approach to organize and query this data. One advantage of metric images is the opportunity to quantify changes that occur over time and thus better informed bridge maintenance and management decisions. To leverage this opportunity requires the image data to be interrogated both spatially and temporally and to link it with other metadata such as inspection notes, design drawings, nondestructive test results, and other relevant data related to the image. Such an integrated inspection image data management system does not exist. To fill this void, a new approach was developed using Geographic Information System (GIS) as a framework to link images spatially across various scales, temporally across inspection intervals, and associate metadata relevant to the image view. The details of this development are described in this section of the report.

2.0 BRIEF BACKGROUND ON WEB GIS

The term *web GIS* is used for any Geographic Information System (GIS) that utilizes web technologies to communicate between its components. A web GIS has many advantages, such as it: enables access for a large number of users over the web, has an independent platform, can be easily updated and maintained, does not require the user to install software or download updates, and is composed of simple scripting and coding. All of these advantages make web GIS a preferred platform for mapping, visualization, data collection, data distribution and web-based geospatial analysis tasks *(Fu and Sun 2011)*.

A web GIS consists of at least two components: a web application server and a client (e.g., a web browser or mobile application). The *web GIS server* is a critical component of the web GIS that provides spatial analysis services for web services. An example is the ArcGIS® Server which is the core GIS service for Esri products and provides web GIS services through ArcGIS® online. ArcGIS® Server is also used for managing multi-user *GIS databases*. GIS databases are spatial data storage and management frameworks that can store a collection of GIS datasets. The manager of the web GIS can update, manage, and store GIS data on the server.

A user will be able to view the data and any updates by accessing or refreshing the web page of the web GIS. Every web page has a unique *URL (Uniform Resource Locator)* address that the client can use to locate the page. The client communicates with the server based on *HTTP (Hypertext Transfer Protocol)* rules and procedures, the foundation of request and response communication for the web. The source code for most web pages is HTML (Hypertext Markup Language). HTML code is interpreted by the web browser to generate the content and the structure of the web page.

Cascading Style Sheets (CSS) are used to improve the appearance, style and layout of pages coded in HTML. Scripting languages are often used to make the web pages dynamic and interactive *(Duckett 2011)*. The most common *scripting language* for web applications is JavaScript, which is a simple, secure, and cross-platform web development language *(Fu and Sun 2011)*.

Large applications often require lengthy amounts of coding, which can be challenging to write and difficult to troubleshoot or update. One way to address this problem is to use patterns of *object-oriented programming*. In *object-oriented programming*, templates or blueprints are used to regenerate *objects* that are identical. Such templates are called *classes*. Classes make code easier to handle and maintain, and they increase the likelihood of their use in the future. In addition, it is possible to use existing classes for constructing an object or a *function* instead of needing to write more code. The constructed object has all of the attributes and methods of the *class*. *Classes* are stored in the components of the system called *modules*, which are developed independent from the system, and the details of what is happing inside the *modules* has no effect on the system as a whole *(Eckel 2006)*.

Application programming interfaces (APIs) are developed for web applications, operating systems, and database systems. APIs can call modules and implement their classes in the application. Some popular examples of APIs include the DirectX for Microsoft Windows®, OpenGL cross-platform graphics API, and the Android API for mobile applications development. *Web APIs* are *mashups* that dynamically integrate complementary elements of web pages or applications. Web APIs provide facilities such as built-in and plug-in blocks, tools and functions (e.g., classes) that maximize programming productivity. ArcGIS® JavaScript API, Google Maps JavaScript API and Leaflet are some commonly used APIs for developments of interactive web mapping applications.

3.0 IBIM (INTERACTIVE BRIDGE INVENTORY MANAGEMENT)

For this project, iBIM is proposed as a web GIS tool that allows interactive access and management of bridge images, inventory data, plans, and inspection reports. It was developed with HTML coding, using ArcGIS® JavaScript API for enabling GIS tools and geospatial data on a web environment. The style and layout of iBIM was coded with CSS. The GIS databases were geo-referenced and constructed in Esri's ArcMap. The databases were then placed on ArcGIS Server and were thereby made accessible for the online applications.

The base section of the tool is called iBIM-Plan, which provides a plan map of the locations of bridges included in the database. Markers depict the geographical location of each bridge. Clicking on a marker opens a pop-up window that displays a description and a profile image of the bridge. The pop-up window also provides a hyperlink to the second section of the tool called iBIM-Profile. As the name implies, iBIM-Profile provides a profile view of the selected bridge and allows interaction with bridge imagery in two different domains: 1) at varying zoom levels, and 2) at different moments in time. The user can view higher resolution imagery by selecting closer zoom levels. In addition, it is also possible to select imagery from different years in order to monitor and detect changes in the bridge with time. The iBIM-Profile also includes hyperlinks to scanned inspection reports, repairs, and redesign reports on the bridge.

The following sections provide much greater details on the data, architecture, graphical user interface (GUI), and coding of iBIM.

3.1 DATASETS AND ARCHITECTURE

All of the web pages and their components, including scripts, images, and data (except map service data) were stored in a folder placed on the web server. A URL address is allocated to this folder that defines a global web address to the application. Figure 3.1 shows the hierarchical structure of the files and folders of iBIM. The root folder contains iBIM-Plan files and iBIM-Profile subfolders for each bridge in the database.

The HTML files contain the code for the structure of the web page, including the headers, text, interactive forms, panels, and map container frame. The CSS files define the layout, style and appearance of the web page contents, and the JS files contain the JavaScript code that make the web pages interactive. More details on the JS codes and functions are provided below. In addition to communication with local JS and CSS codes, the HTML file also calls remote library scripts, such as mapping API modules that empower the web spatial capabilities.

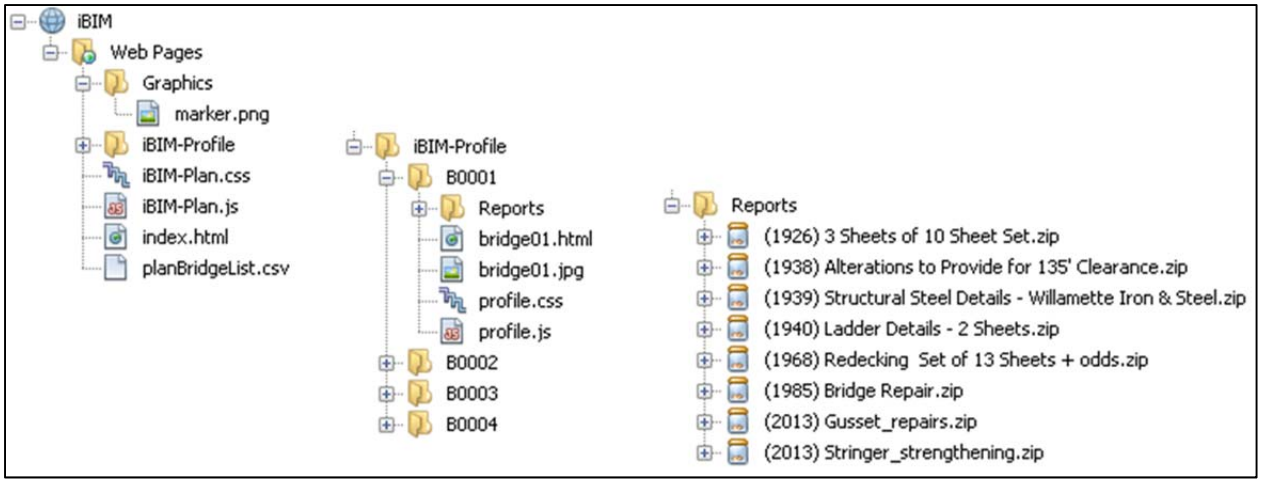


Figure 3.1: The hierarchical structure of the files and folders of the iBIM-Plan and iBIM-Profile tools

Tabular information on the bridges is stored in the CSV (Comma Separated Values) file. Table 3.1 is a simplified example of data in this file. The iBIM-Plan tool queries the latitude and longitude fields in this table in order to display a marker (i.e., a point) at each bridge on the digital map. The "Image Link" field provides the relative address to the overview profile image of the bridge. Similarly, the "Page Link" field lists the address to the iBIM-Profile web page (HTML file) for the bridge. As mentioned previously, when the user clicks on the marker, a popup window will appear which displays the attribute information from the CSV file, the overview image, and a hyperlink that will open the iBIM-Profile for the bridge.

Table 3.1: A simplified example of a CSV file for storing bridge inventory data

Name	Latitude	Longitude	Cross	Image Link	Page Link
Bridge of the Gods	45.66242	-121.90128	Columbia River	iBIM-Profile/B0001/bridge01.jpg	iBIM-Profile/B0001/bridge01.html
Steel Bridge	45.52778	-122.66778	Willamette River	iBIM-Profile/B0002/ bridge02.jpg	iBIM-Profile/B0002/bridge02.html
Yaquina Bay Bridge	44.62207	-124.05636	Yaquina Bay	iBIM-Profile/B0003/ bridge03.jpg	iBIM-Profile/B0003/bridge03.html
Astoria - Megler Bridge	46.21725	-123.86291	Columbia River	iBIM-Profile/B0004/ bridge04.jpg	iBIM-Profile/B0004/bridge04.html

One of the objectives of the iBIM-Profile tool is to enable the use of web-based GIS tools in order to explore bridge inspection and inventory images acquired at different standoff distances and in different times. In order to make use of web GIS tools, images of the bridge were first geo-referenced (rotated and translated) to a geographic coordinate system. To accomplish this task, overview image(s) of the entire bridge were first imported in Esri ArcGIS software. Figure 3.2 shows an example set of images of an entire bridge. The overview images were roughly geo-referenced to the location of the bridge using the "georeferencing tool" in ArcGIS and by applying a reasonable scale factor. Afterwards, the closer-up, higher resolution images of the

bridge were imported and georeferenced on top of the overview images. This process was followed for all images, and the time each photo was taken was also tagged for all of the images (Figure 3.3).

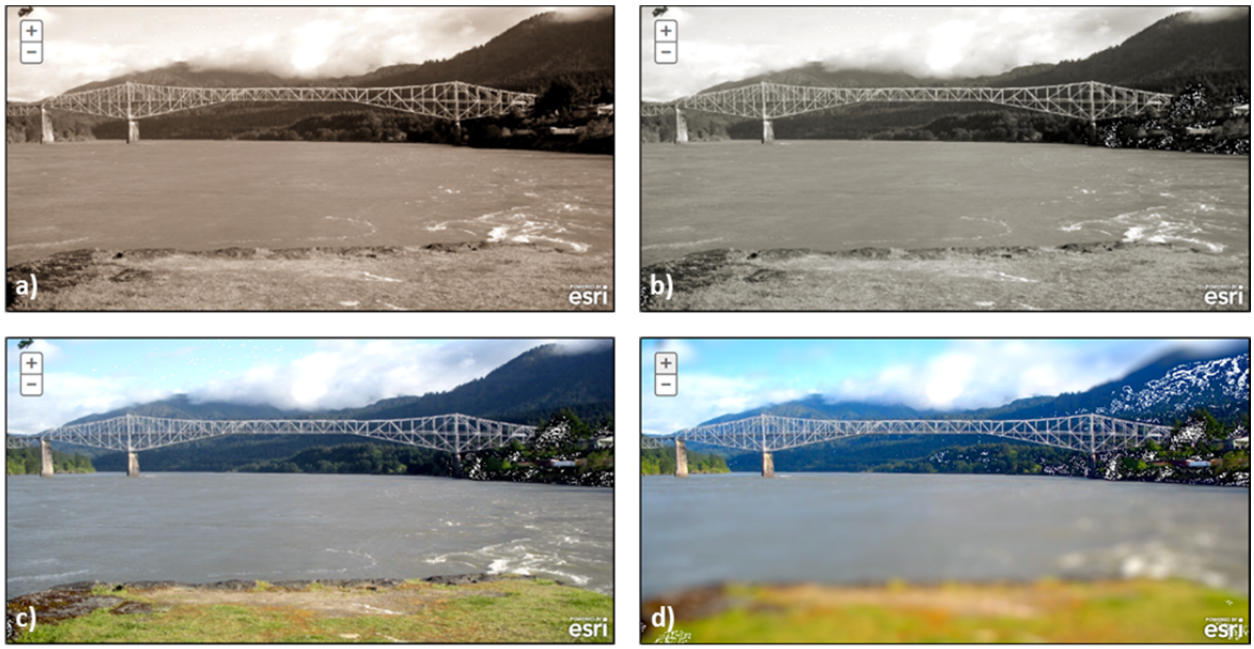


Figure 3.2: Example overview images for Bridge of the Gods in different years: a) year 0, b) year 4, c) year 8, and d) year 12

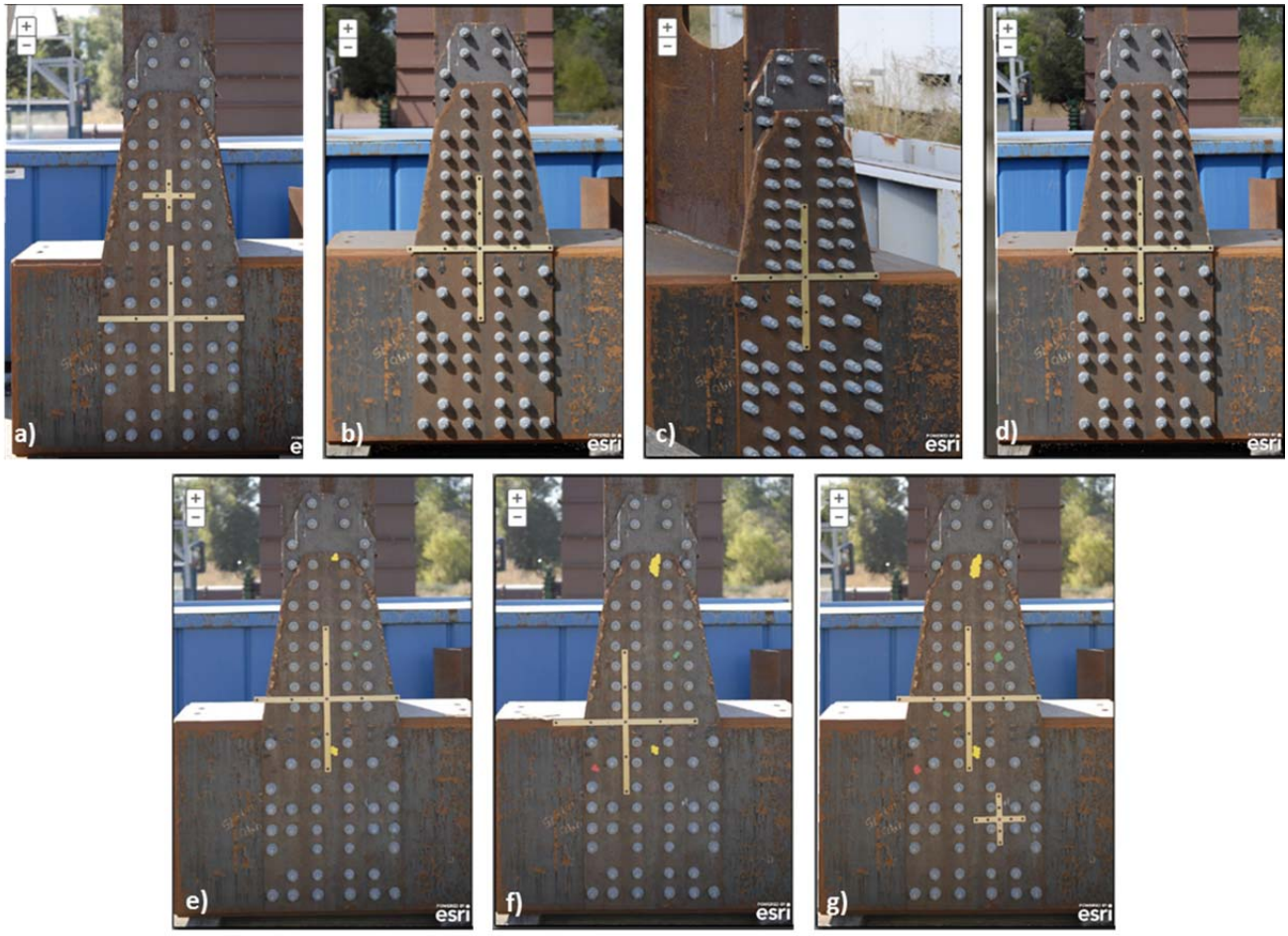


Figure 3.3: Images of a gusset plate in different years: a) year 0, b) year 2, c) year 4, d) year 6, e) year 8, f) year 10, and g) year 12

In order to provide online access to the images, the geo-referenced images were shared on a host server. ArcGIS Server allows *publishing* images or raster databases as *image services* on a server (ArcGIS Image extension is required). Image services are *tiles* of raster data divided into smaller, more manageable chunks that are ordered in pyramidal scheme. A raster pyramid is a series of raster data of the same area that has been resampled into coarser resolutions. Instead of always attempting to display the full resolution overview image, raster pyramids and tiles of the image can be streamed by the tool more quickly depending on the level of zoom, thereby improving visual performance and speed. Figure 3.2 presents the raster tiles for the overview images of the Bridge of the Gods for different years at zoom level 24. (Note that zoom levels are defined in greater detail below). Figure 3.3 displays high-resolution images of a joint at the same bridge captured during different years. The images in Figure 3.3 are online streams from image service in iBIM-Profile, at zoom level 29.

The iBIM-Profile interacts with the image services and displays them in a map container. The coarse or fine resolution raster pyramids appear depending on the zoom level. In addition to automatically representing the raster pyramids at different zoom levels (based on the definition given in the image server), the iBIM-Profile also dynamically overlays the close-up images on top of the overview image as the user zooms into a specific portion of the bridge. A reverse

action occurs when the user zooms out from the close-up view, and the higher detailed image is removed from the overview image. Figure 3.4 shows an example for the dynamic addition and removal of image services to the display based on the level of zoom, from the overview image at zoom level 24 (L:24) to a close up image at zoom level 35 (L:35).

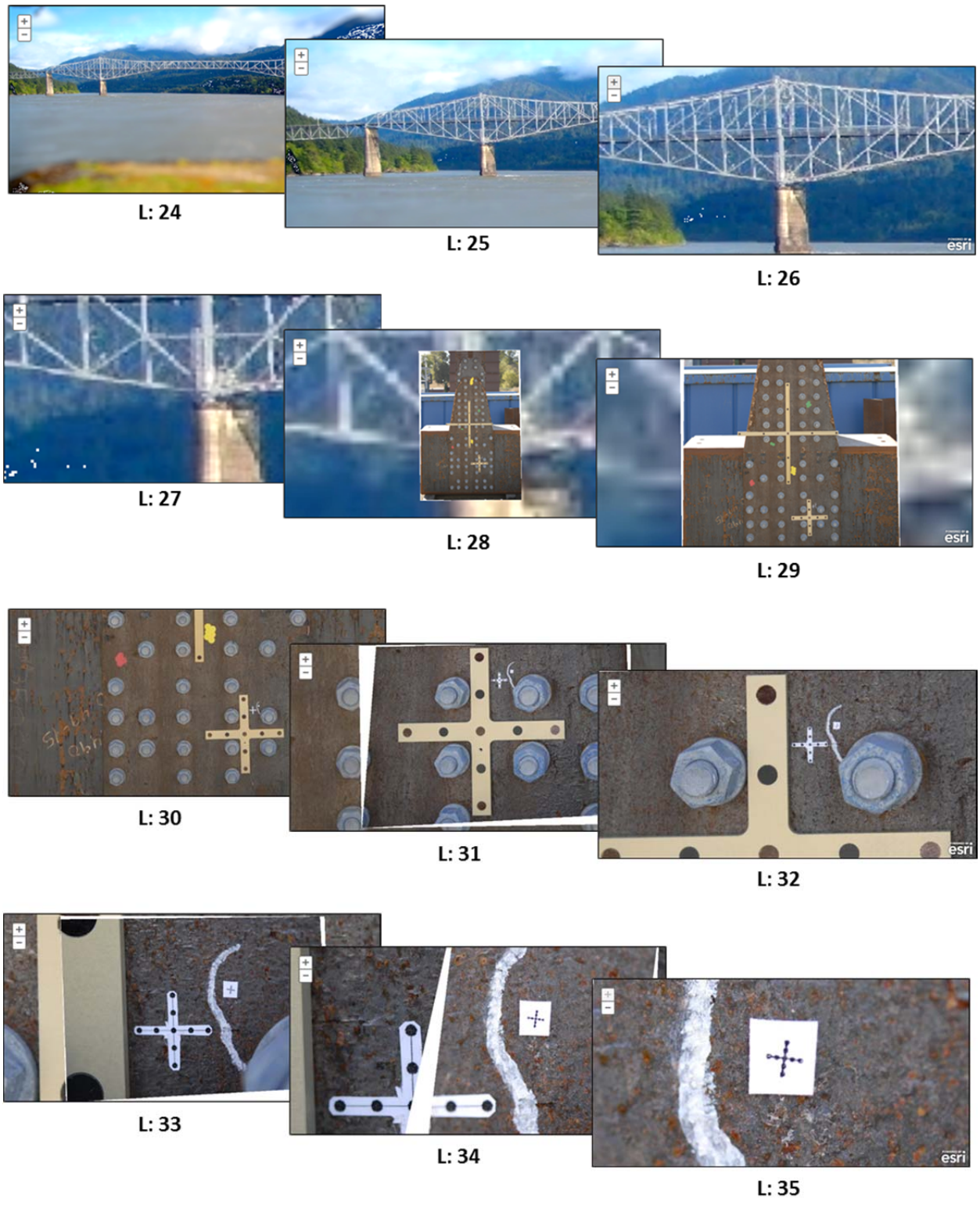


Figure 3.4: Bridge inventory images for a gusset plate at different zoom levels (L). Image services are represented in coarse or fine resolution tiles. The image services are dynamically added or removed based on the defined zoom levels

3.2 GRAPHICAL USER INTERFACT (GUI)

The Graphical User Interface (GUI) for iBIM-Plan and iBIM-Profile are designed slightly different; however, both tools are based on web GIS tools and capabilities.

3.2.1 iBIM-Plan

A screenshot of the iBIM-Plan is shown in Figure 3.5 and its components (numbered in the figure) are described below.

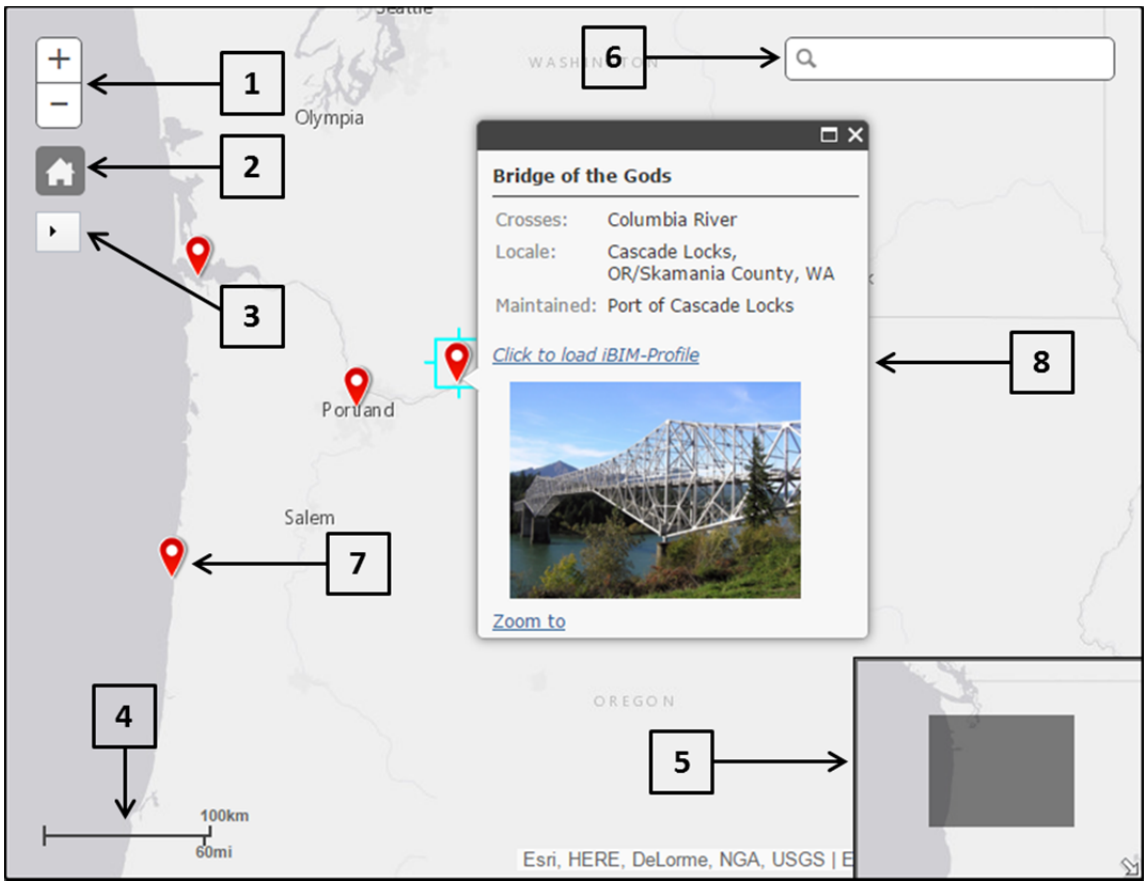


Figure 3.5: Outline view of the iBIM-Plan Tool with various feature tools and their screen location

1. *Zooming and Panning*: The user can navigate through the map by clicking on the zoom in and out buttons or by using a mouse wheel.

2. *Home Button:* The default view of the map is set to the State of Oregon. If the user changes the zoom level or pans the map to a different extent, the zoom level and map extent will return back to this view by clicking the Home Button.

3. *Basemaps:* By clicking on this icon, a new box will appear that includes a list of available basemaps for display (e.g., Esri Streets, Imagery and Topography, OpenStreetMaps, USGS National Maps etc.). Once the selection is done, the user can close the box by clicking on the base map icon.

4. *Scalebar:* The scalebar is interactive and provides map scale both in metric and imperial units.

5. *Overview Map:* This coarser map shows the spatial extent of the view of the main window. The box is located on the lower right corner of the map container. The user can hide the overview map box by click on the arrow at the corner of the box.

6. *Address Locator:* This tool allows searching for a specific location on the map by entering its address. Running this command centers and zooms the view of the main window to the address.

7. *Marker:* Each marker shows the latitude and longitude of a bridge included in the database.

8. *Popup Information Window:* The user can access basic information about the bridge via an information window that will appear when clicking on the marker. The information window displays the name of the bridge, the name of the feature that the bridge crosses, the organization that is responsible for maintaining the bridge, and other general information. The window also contains an overall image of the bridge and a hyperlink that takes the user to iBIM-Profile tool.

3.2.2 iBIM-Profile

A screenshot of the iBIM-Profile is shown in Figure 3.6 and its components (numbered in the figure) are described below.

1. *Zooming and Panning*: Similar to iBIM-Plan, the zoom in or out buttons can be used to navigate the data. Changing the zoom level will load different image service tiles. In addition, image services will be added or removed dynamically based on the zoom level. For example, as the user zooms in, increasingly closer-up, higher-resolution images appear on the screen.

2. *Select Year*: The user can select a year of interest for filtering the imagery. Only image services for a selected year will be shown on the screen. The user can change the year at a certain level of zoom, and the raster image for that year at the level of zoom will be displayed. This enables the user to "travel back and forward in time" in order to help with detecting changes to the bridge.

3. *Reports:* The user can download a scanned report for the bridge by clicking on a hyperlink. The hyperlink text indicates the year and title of the report.

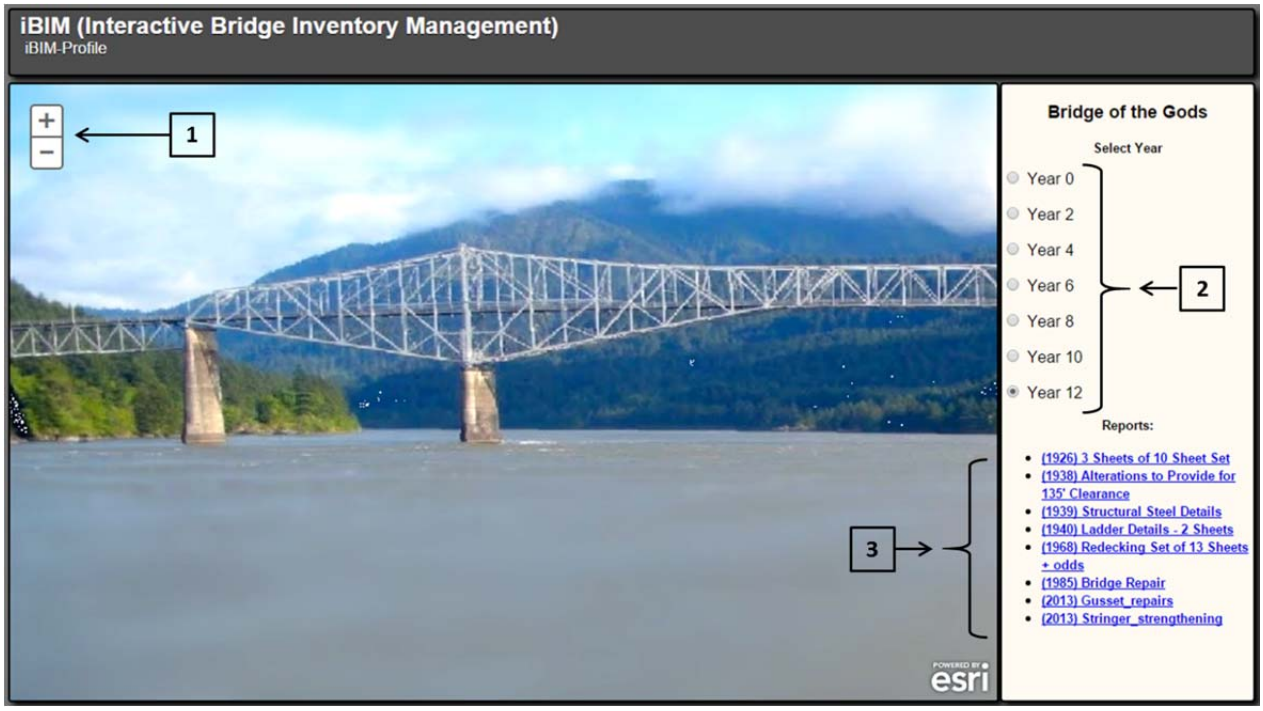


Figure 3.6: The overall design of iBIM-Profile with the time tools and report hyperlinks

3.3 CODE

The iBIM uses ArcGIS JavaScript API for the web mapping applications and accessing the GIS capabilities on ArcGIS Server. The following section describes some important parts of the JavaScript code used in the iBIM tool.

The "main" classes required for the application (e.g., "Map", "OverviewMap", "BasemapGallery", "Scalebar", "ArcGIS®ImageServiceLayer", "ImageServiceParameters") were from Esri modules (e.g., "esri", "esri/dijit", "dojo", etc.). These classes were used to create different parts of the tool without the need to write detailed code.

The basic framework for both iBIM-Plan and iBIM-Profile was a map class that was defined as a *variable*, constructed inside of the given HTML container. The following construction properties defined the center of the map extent, the zoom level and the Esri basemap:

```
var map;
map = new Map("map", {
   center: [-122.5, 45],
   zoom: 8,
   basemap: "gray"
});
```

The marker was defined as a picture symbol using the "PictureMarkerSymbol" class, with construction properties (e.g., type, height, width, and the URL of the graphics) specified in constructer properties.

```
var marker = new esri.symbol.PictureMarkerSymbol({
```

```
});
```

Each time that the tool is loaded on the web page, the CSV file is parsed, and the marker is rendered in the map container based on the latitude and longitude fields.

```
csv = new CSVLayer("planBridgeList.csv");
var renderer = new SimpleRenderer(marker);
csv.setRenderer(renderer);
map.addLayer(csv);
```

The popup window was constructed from the "esri.dijit.PopupTemplate" and will display the field information from the CSV file when the user clicks on the marker. The following code was used for defining the variable "template", which includes the name field of the bridge as its "title", and the various data from the CSV file as "fieldnames". Also, the popup window loads an image of the bridge (e.g., a media file) by reading the URL address of the image (i.e., using "sourceURL") from the CSV file, and the window converts the image and its caption into hyperlinks to the iBIM-Profile by reading the URL page address field (i.e., using "linkURL") from the CSV file.

```
var template = new esri.dijit.PopupTemplate({
   title: "{Name}",
   fieldInfos: [
      {fieldName: "Cross", visible: true, label: "Crosses:"},
      {fieldName: "Locale", visible: true, label: "Locale:"},
      {fieldName: "Maintained", visible: true, label:
      "Maintained:"}
   ],
   mediaInfos: [{
      "caption": "<a href={Page Link}> Click to load iBIM-
      Profile</a>",
      "type": "image",
      "value": {
          "sourceURL": "{Image Link}",
          "linkURL": "{Page Link}"
                  }
      }]
});
```

Finally, a new title is set to the "template" variable and the CSV file, previously defined as a "CSVLayer". This new title comprises all of the bridge data read from the CSV file.

```
template.setTitle("Bridge Attributes");
csv.setInfoTemplate(template);
```

The hyperlink in the popup window opens the iBIM-Profile tool. The iBIM-Profile tool uses a different set of web mapping tools. Typically, the default zoom level in a web GIS is 22; however, this tool must zoom into very close-up, high-resolution bridge images. Therefore, it is necessary to manually define the scale and resolution of each zoom level to the application. The following code defines zoom levels 1 to 35, where the resolution field defines the size of an

individual raster pixel (for this case, the units are considered in a relative scale, but technically they are in WGS84 Web Mercator (Auxiliary Sphere) units).

```
var lods = [
    {"level": 0, "scale": 3000000, "resolution": 793.75158750317507},
    {"level": 1, "scale": 2000000, "resolution": 529.167725002116},
    {"level": 2, "scale": 1000000, "resolution": 264.583862501058},
    {"level": 3, "scale": 500000, "resolution": 132.291931250529},
    {"level": 4, "scale": 250000, "resolution": 66.1459656252645},
    {"level": 5, "scale": 125000, "resolution": 33.0729828126322},
    {"level": 6, "scale": 50000, "resolution": 13.2291931250529},
    {"level": 7, "scale": 30000, "resolution": 7.93751587503175},
    {"level": 8, "scale": 12500, "resolution": 3.30729828126322},
    {"level": 9, "scale": 5000, "resolution": 1.32291931250529},
    {"level": 10, "scale": 2500, "resolution": 0.661459656252645},
    {"level": 11, "scale": 1000, "resolution": 0.264583862501058},
    {"level": 12, "scale": 500, "resolution": 0.132291931250529},
    {"level": 13, "scale": 250, "resolution": 0.0661459656252645},
    {"level": 14, "scale": 100, "resolution": 0.0264583862501058},
    {"level": 15, "scale": 51.932690695019, "resolution": 0.0138940720685454},
    {"level": 16, "scale": 24.1660431071863, "resolution": 0.00646538317289663},
    {"level": 17, "scale": 11.2452798351617, "resolution": 0.00300856216709916},
    {"level": 18, "scale": 5.23281027060198, "resolution": 0.00139998605979685},
    {"level": 19, "scale": 2.43500417326198, "resolution": 0.00065146101651451},
    {"level": 20, "scale": 1.13309006388285, "resolution": 0.000303146915691219},
    {"level": 21, "scale": 0.527265253574545, "resolution": 0.000141064545941334},
    {"level": 22, "scale": 0.245354413112011, "resolution": 6.56421196839874E-05},
    {"level": 23, "scale": 0.114171733535308, "resolution": 3.05455055900363E-05},
    {"level": 24, "scale": 0.0531279815721367, "resolution": 1.42138601898094E-05},
    {"level": 25, "scale": 0.0247222525096932, "resolution": 6.61419143644332E-06},
    {"level": 26, "scale": 0.0115041029428751, "resolution": 3.07780770133681E-06},
    {"level": 27, "scale": 0.00535324944474125, "resolution": 1.43220835644607E-06},
    {"level": 28, "scale": 0.0024910486076075, "resolution": 6.66455144479311E-07},
    {"level": 29, "scale": 0.00115916944082609, "resolution": 3.10124192198613E-07},
    {"level": 30, "scale": 0.000539400872564902, "resolution": 1.44311309446014E-07},
    {"level": 31, "scale": 0.0002510015284016, "resolution": 6.71529489085642E-08},
    {"level": 32, "scale": 0.000116799527891677, "resolution": 3.12485456921394E-08},
    {"level": 33, "scale": 5.43507834497782E-05, "resolution": 1.45410086041537E-08},
    {"level": 34, "scale": 2.52912637142189E-05, "resolution": 6.76642469410213E-09},
    {"level": 35, "scale": 1.17688831634455E-05, "resolution": 3.14864700154821E-09}
];
```

In order to load imagery for a specific time, one of the necessary steps is to read the year selected by the user from the right panel of the GUI. The following code reads the selected *radio button*, and parses it to an integer value for the given year.

```
$('.radio-layers').bind('click', function () {
    var year = [parseInt($(this)[0].id)];
});
```

In addition, four cutoff zoom levels were defined, which were used to dynamically add and remove the image services based on the zoom levels.

```
var L2 = 28;
var L3 = 31;
var L4 = 33;
var L5 = 34;
```

In order to handle the image service layers on the ArcGIS Server, iBIM uses the "ArcGIS®ImageServiceLayer" class from the "esri/layers" module. The code below allowed communication with the image service dataset and read the layer parameters. A dummy value of 1 was assigned in case of no data.

```
var params = new ImageServiceParameters();
    params.noData = 1;
```

The URL location and the parameters from the image service layer needed to be specified. The following code reads the parameters from the server, "arcweld.engr.oregonstate.edu", and stores them as ArcGIS®ImageServiceLayer variables.

```
var yr12_L5a = new
ArcGIS®ImageServiceLayer("http://arcweld.engr.oregonstate.edu:6080/ArcGIS®/re
st/services//Bridge_GeoRef/yr12_L5a/ImageServer", {
        imageServiceParameters: params,
        opacity: 1.0
    });

var yr_1 = new
ArcGIS®ImageServiceLayer("http://arcweld.engr.oregonstate.edu:6080/ArcGIS®/re
st/services//Bridge_GeoRef/yr_1/ImageServer", {
        imageServiceParameters: params,
        opacity: 1.0
    });
```

The tool starts by adding a default layer to the web GIS map by following the sample code below. In this case, the default layer is the overview image of the entire bridge for year 1.

```
    map.addLayer(yr_1);
```

As the user zooms in or out, the tool adds or removes additional images (i.e., layers) according to zoom cutoff levels (e.g. L2, L3, L4, L5). The "On Style Events" of the "Map" class identifies the "extent-change" and reads the zoom level selected by the user. By using *if* statements based on the zoom level, imagery is coded to be added or removed. The "year" variable is a parameter that specifies the year of the imagery. The following code is an example of adding imagery with detail greater than cutoff zoom level "L5" to the map according to a year selected by the user.

```
map.on("extent-change", function () {
        level = map.getLevel();


        if (level >= L5) {
            switch (year) {
                case 8:
                    map.addLayer(yr08_L5);
                    break;

                case 10:
                    map.addLayer(yr10_L5a);
                    break;
```

```
                case 12:
                    map.addLayer(yr12_L5a);
                    break;
            }
        }

    });
```

If the user selects a different year without changing the zoom level, the following code removes all of the imagery from the overview image, then adds new imagery for the selected year according to the zoom level (please note that the example code below only includes year 0 and year 12):

```
$('.radio-layers').bind('click', function () {
        year = parseInt($(this)[0].id);

        switch (year) {
            case 0:
                map.removeAllLayers();
                map.addLayer(yr_1);

                if (level >= L2 && level < L3) {
                    map.addLayer(yr00_L2);
                } else {
                }
                break;


            case 12:
                map.removeAllLayers();
                map.addLayer(yr_4);
                if (level >= L2 && level < L3) {
                    map.addLayer(yr12_L2);

                } else if (level >= L3 && level < L4) {
                    map.addLayer(yr12_L3a);
                    map.addLayer(yr12_L3b);
                } else if (level >= L4 && level < L5) {
                    map.addLayer(yr12_L4a);

                } else if (level >= L5) {
                    map.addLayer(yr12_L5a);
                } else {
                }
                break;
        }

    });
```

4.0 SUMMARY

This paper proposed the framework for a new web GIS tool for managing bridge inspection images and other inventory data related to the images. The iBIM is a web-based prototype that provides the user a simple interface for viewing, panning, zooming in and out of bridge imagery collected over the years as a result of numerous bridge inspections and interventions. The tool is meant to provide the user an intuitive, organized method for evaluating and managing bridge inspection data. The iBIM has two main parts: 1) iBIM-Plan, which is a web GIS tool that provides plan view of the locations of bridges that are included in the database, and 2) iBIM-Profile that is a profile view of each bridge that allows the user to navigate through large volumes of bridge images collected at different physical scales and at different times. The user is also provided links to a library of scanned bridge inventory data such as inspection notes, nondestructive test results, structural drawings, etc.

This paper presented the architecture for the tool, listed the required data, and demonstrated the steps necessary for preparing and publishing the required data. Example data for The Bridge of the Gods was developed and implemented in the database. Georeferenced raster photos of the bridge were published as image service layers on an ArcGIS Server and were used in the application.

An explanation of the Graphical User Interface and the capabilities of the tool was also given. Some of the important code was described, and the paper discussed some of the classes and modules used in development of the tool.

5.0 FUTURE DEVELOPMENTS

The iBIM was designed as a prototype web GIS for enhancing the management of bridge inspection images and inventory data. Several items could be added to enhance iBIM.

First, as the number of bridges increase in the database, it may become difficult for the user to find particular bridges in iBIM-Plan. A tool could be developed that allows the user to search, sort, and filter the bridges in the database based on bridge attributes in the CSV file.

Second, the current design only shows one profile view of each bridge. It is proposed that additional map panels be generated to iBIM-Profile so that each panel can show the upstream and downstream sides of the bridge superstructure.

Third, one of the limitations of the current iBIM development is how it handles time. Based on the current design, each image is defined as one image service layer that is dynamically added or removed with respect to the zoom level of the map. It is recommended to use sliders instead of radio buttons in the future, and to develop web-based code that can use a raster mosaic database capable of handling time-series data in multiple dimensions instead of using separate raster layer for each image.

In most cases, the close-up, higher definition images were taken for a small portion of the bridge. For example, often the imagery was limited to the areas that were identified by the inspector to be important, such as at a few connectors or along a problematic structural member. One idea is to add markers on the overview profile image that allows users to quickly identify the areas on the bridge that contain higher resolution images when zoomed out. The current development lacks such highlighting markers.

5.1 SOFTWARE VERSIONS AND REQUIREMENTS

The following software and versions were used in the development of iBIM:

- NetBeans IDE 8.0 was used for web development. (Any IDE or editor, such as Aptana Studio, Visual Studio, Sublime Text, Notepad ++, etc would work for this purpose).

- ArcGIS 10.2 was used for generating raster databases and publishing online image services.

To run iBIM and access the data requires the user to have only a web browser and an internet connection.

6.0 REFERENCES

Arias, P., C. Ordóñez, H. Lorenzo, J. Herraez, and J. Armesto. Low-Cost Documentation of Traditional Agro-Industrial Buildings by Close-Range Photogrammetry. *Building and Environment*, Vol. 42, 2004, pp 1817-1827.

Brunelli, R., and T. Poggio. Face Recognition: Features versus Templates. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 15, No. 10, 1993, pp. 1042-1052.

Cook, R.D., D.S. Malkus, and M.E. Plesha. *Concepts and Applications of Finite Element Analysis*. John Wiley & Sons, 1989.

Criminisi, A., I. Reid, and A. Zisserman. Single View Metrology. *International Journal of Computer Vision*, No. 40, 2000, pp. 123-148.

Dare, P.M., H.B. Hanley, C.S. Fraser, B. Riedel, and W. Niemeier. An Operational Application of Automatic Feature Extraction: The Measurement of Cracks in Concrete Structures. *The Photogrammetric Record*, Vol. 17, No. 99, 2002, pp. 453-464.

Di Stefano, L., S. Mattoccia, and M. Mola. DEIS-ARCES, An Efficient Algorithm for Exhaustive Template Matching based on Normalized Cross Correlation. IEEE computer society, Proceedings of the 12th International Conference on Image Analysis and Processing. 2003.

Duckett, J. *HTML and CSS Design and Build Websites*. 1st Ed., John Wiley & Sons, Inc., Indianapolis, IN, 2011.

Dvorkin, E.N., and K.J. Bathe. A Continuum Mechanics Based Four Node Shell Element for General Nonlinear Analysis. *Engineering Computations*, Vol. 1, No. 1, 1984, pp.77–88.

Eckel, B. *Thinking in Java*. 4th Ed. Pearson Education, Upper Saddle River, NJ. 2006.

Fu, P., and J. Sun. *Web GIS: Principles and Applications*. Esri Press. 2011.

Fujita, Y., Y. Mitane, and Y. Hamamoto. A Method for Crack Detection on a Concrete Structure. The 18th International Conference on Pattern Recognition. 2006.

Goshtasby, A., S.H. Gage, and J.F. Bartholic. A Two-Stage Cross-Correlation Approach to Template Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, No. 3, 1984, pp. 374-378.

Hartley, R.I., and A. Zisserman. *Multiple View Geometry in Computer Vision*, Second Edition, Cambridge University Press. 2004.

Heuvel, F. 3D reconstruction From a Single Image Using Geometric Constraints. *ISPRS Journal of Photogrammetry and Remote Sensing*, No. 53, 1998, pp. 354-368.

Ito, A., Y. Aoki, and S. Hashimoto. Accurate Extraction and Measurement of Fine Cracks from Concrete Block Surface Image. IECON Proceedings (Industrial Electronics Conference),Vol. 3, 2002, pp. 2202-2207.

Jauregui, D., Y. Tian, and R. Jiang. Photogrammetry Applications in Routine Bridge Inspection and Historic Bridge Documentation. In *Transportation Research Record*: Journal of the Transportation Research Board, No. 1958, Transportation Research Board of the National Academies, Washington, D.C., 2006, pp. 24-32.

Kanungo, T., D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. An Efficient K-means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 7, 2002, pp. 881-892.

Lee, S., and L. Chang (2005). Digital Image Processing Methods for Assessing Bridge Painting Rust Defects and Their Limitations. ASCE International Conference on Computing in Civil Engineering. 2005.

Liu, Z., D.S. Forsyth, A. Marincak, and P. Vesley. Automated Rivet Detection in the EOL Image for Aircraft Lap Joints Inspection. *NDT&E International*, Vol. 39, 2006, pp. 441-448.

MATLAB. *Image Processing Toolbox.* The MathWorks Inc., Natick, MA. 2013.

McKenna, F., G.L. Fenves, and M.H. Scott. *Open System for Earthquake Engineering Simulation*. University of California, Berkeley, CA, 2000. http://opensees.berkeley.edu.

McKenna, F., M.H. Scott, and G.L. Fenves. Nonlinear Finite-Element Analysis Software Architecture using Object Composition. *Journal of Computing in Civil Engineering*, Vol. 24, No. 1, 2010, pp. 95–107.

Mills, J., and D. Barber. Geomatics Techniques for Structural Surveying. *Journal of Surveying Engineering*, Vol. 130, No. 2, 2004, pp. 56-64.

Ousterhout, J.K. *Tcl and the Tk Toolkit*. Addison-Wesley, 1994.

Rodríguez, J., M.T. Martín, P. Arias, C. Ordóñez, and J. Herráez. Flat Elements on Buildings using Close-Range Photogrammetry and Laser Distance Measurement. *Optics and Lasers in Engineering*, No. 46, 2008, pp. 541-545.

Scott, M.H., and G.L. Fenves. Krylov Subspace Accelerated Newton Algorithm: Application to Dynamic Progressive Sollapse Simulation of Frames. *Journal of Structural Engineering*, Vol. 136, No. 5, 2010.

Simo, J.C., and T.J.R. Hughes. *Computational Inelasticity*. Springer-Verlag, New York, NY, 1998.

Tommaselli, A., and M. Reiss. A Photogrammetric Method for Single Image Orientation and Measurement. *Photogrammetric Engineering and Remote Sensing*, Vol. 71, No. 6, 2005, pp. 727-732.

Welch, B.B. *Practical Programming in Tcl and Tk*. Prentice-Hall, Upper Saddle River, NJ, 2000.

Yamaguchi T., and S. Hashimoto. Automated Crack Detection for Concrete Surface Image using Percolation Model and Edge Information. Proceedings of the 32nd Annual Conference of the IEEE Industrial Electronics Society. 2006.

Yamaguchi, T., and S. Hashimoto. Fast Crack Detection Method for Large Size Concrete Surface Images using Percolation Based Image Processing. *Machine Vision and Applications*, Vol. 21, No.5, 2009, pp. 797-809.

Yamaguchi, T., S. Nakamura, and S. Hashimoto. An Efficient Crack Detection Method using Percolation-Based Image Processing. Proceedings of the 3rd IEEE Conference on Industrial Electronics and Applications. 2008.