# A Study on the Extraction of Geometrical Parameters from Flexible Mechanical Components and Assemblies and their Impact on Performance: A Machine Learning Approach

by

Prakash Kumar

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved June 2024 by the
Graduate Supervisory Committee:

Joseph Davidson, Co-Chair
Houlong Zhuang, Co-Chair
Jami Shah

ARIZONA STATE UNIVERSITY

August 2024

ABSTRACT

Over a population of flexible sheet metal assemblies of the same design, such as an automobile door or hood, there typically are geometric variations that can become apparent as a changing gap between the door/hood and the auto body. The gap variations stem from allowable tolerances for the material (e.g., thickness, strength), spring-back of components post-stamping, and residual stress distortions following clamping and spot welding of the assemblies. To control the geometric effect from these input variations, multi-stage nonlinear finite element analyses (FEA) are used to relate the free-state nodal-point geometry to the inputs. Since the analyses are time-consuming, a sparse grid of input values (the design space) is used.

Given the sprung-back nodal data from FEA, the objectives of this research are to construct algorithms that generate geometric performance parameters (e.g. component or assembly twist) and to explore the application of machine learning (ML) for predicting these parameters in flexible assemblies at a greater resolution over the design space than is feasible with FEA. To achieve this goal, three datasets of FEA simulations were examined. Geometric parameters were extracted from two of these. The first was for stamped hat-section sheet metal components and for straight assemblies spot-welded from them, some with a flat unformed sheet as one component. Subsequently, one of the assembly parameters, a minimum-zone magnitude, was used to showcase the suitability of geometric parameters to be used for training a ML algorithm, specifically a fully connected Multi-Layer Perceptron Neural Network. The second dataset consisted of the nodal coordinates from FEA simulations of a Honda-suggested T-joint assembly. Here, algorithms were produced for computing 17 geometric performance parameters. The third dataset comprised multiview images of automobile hood structures. Three deep learning models were compared for accuracy of feature extraction from the images and for directly estimating maximum

stress without time-consuming FEA.

A last important contribution is the validation of the new iterative virtual-work-and-robotics method for fitting nodal points that deviate spatially from a straight line or circular arc. This allows geometric performance parameters for curved components to be included in future flexible assembly datasets.

# DEDICATION

*This thesis is dedicated to my parents, sister, and fiancé for their unwavering support and sacrifice throughout my journey.*

# ACKNOWLEDGMENTS

The completion of this thesis and the research it encompasses would not have been possible without the invaluable support and guidance of many individuals, to whom I extend my heartfelt gratitude.

First and foremost, I express my sincere appreciation to my advisor, Dr. Joseph Davidson, whose unwavering support, mentorship, and expertise were instrumental throughout my master's degree journey. His encouragement significantly influenced the direction of my research, inspiring a commitment to continuous improvement.

I also extend my gratitude to the members of my thesis committee for their invaluable guidance. Special thanks to Dr. Jami Shah for his insightful feedback and constructive criticism. His collaboration helped shape my best ideas and provided clear direction during this journey. Additionally, I am grateful to Dr. Houlong Zhuang for his generous time and support.

Lastly, I thank my family and friends for their unwavering encouragement and support throughout this academic journey. I also acknowledge the financial support provided by the National Science Foundation through the GOALI Award No. 2030093 to Arizona State University.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

# Chapter 1

# INTRODUCTION

In the realm of engineered products, particularly within the automotive industry, the assembly of sheet metal components plays a pivotal role in the construction of vehicle bodies. These assemblies are crucial not only for their structural integrity but also for meeting the increasingly stringent design goals aimed at reducing weight and enhancing safety. To achieve these objectives, design engineers are turning to advanced materials characterized by superior strength-to-weight ratios. However, the adoption of such materials introduces complex challenges in accurately predicting manufacturing variations that arise from cold rolling and forming processes. These challenges are compounded at the assembly level, where component-level variations of residual stresses and elastic spring-back necessitate meticulous compensation as parts are clamped and joined together, further complicating the prediction of final assembly variations.

Currently, the industry lacks a comprehensive tool capable of predicting these variations in sheet metal assemblies, taking into account the intricacies of tooling design and process parameters.The overall research project, of which this thesis is a part, seeks to bridge this gap by developing correlations between a large dataset of FEA simulations of manufacturing variations and geometric parameters that measure quality of the final assembly. The geometric parameters are extracted from a vast dataset of simulated components and assemblies, specifically stamped hat sections and spot welded assemblies made from them[1]. This work is described in Chapters 3 and 4.

Separately, this study harnesses machine learning algorithms to directly predict

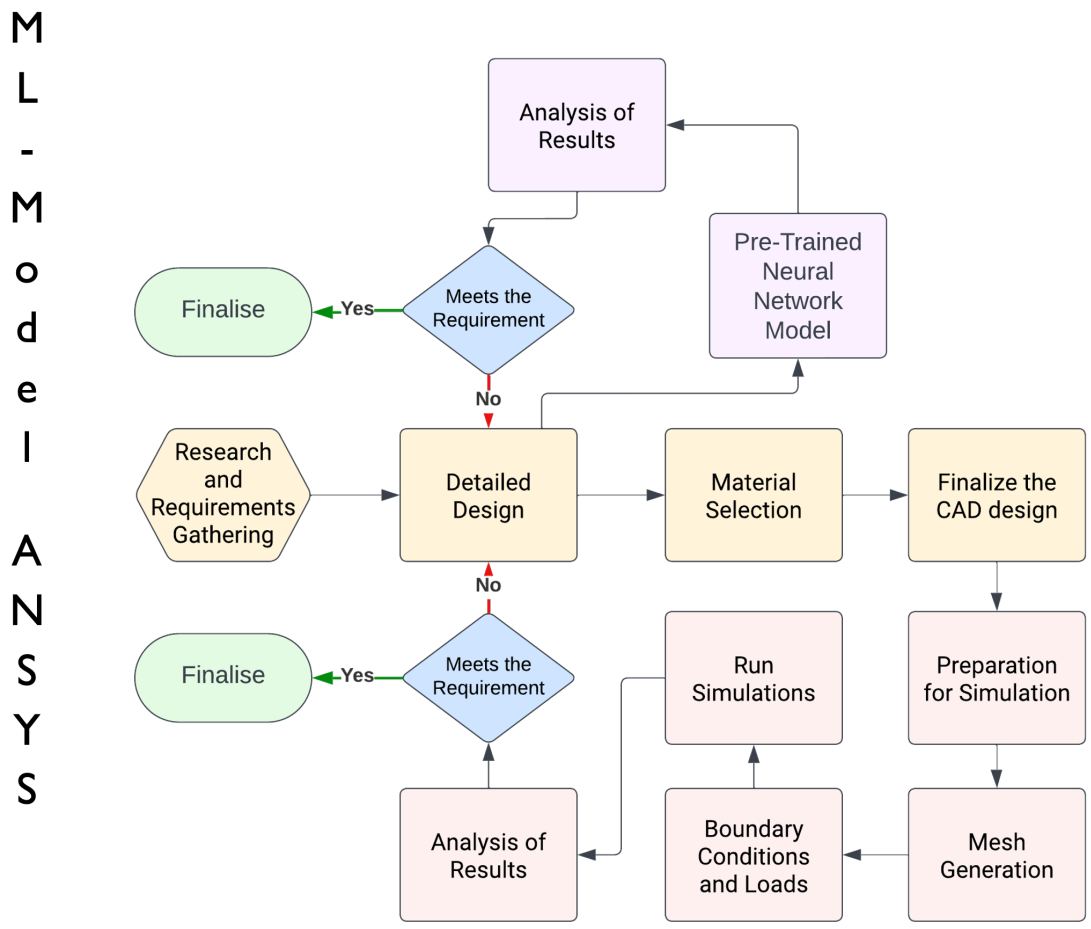



Figure 1.1: Comparison: Conventional Simulation (Lower Half of the Image) Vs ML Models (Upper Half of the Image).

these geometric parameters and deviations from the expected shape post stamping from the input conditions(Chapters 3). Additionally, images from multi-view stamped components, specifically car hood frames, are used as data for machine learning to accurately predict corresponding performance parameters, thus leveraging the synergy between machine learning and computer vision techniques (Chapter 5).

This approach represents a significant leap forward in structural design, offering a more efficient and cost-effective alternative to traditional simulation-based methods.

The integration of machine learning and computer vision not only streamlines the design process but also accelerates vehicle development, providing tangible benefits to manufacturers and consumers alike. By circumventing the need for resource-intensive simulations, as illustrated in Figure 1.1, the proposed method endeavors to accurately predict the performance parameters of stamped components, thereby streamlining the production of automotive bodies.

## 1.1 Background and Context

### *1.1.1 Variations in Flexible Sheet Metal*

Sheet metal components are typically much longer and wider than they are thick, which gives them a certain degree of flexibility. This flexibility is important to consider when designing assemblies using these components. By taking advantage of the wide array of geometric shapes that can be made by stamping or forming sheet metal, designers are able to minimize the amount of material subjected to low loads and, at the same time, meet stiffness requirements.

Automotive body structures are a good example of flexible sheet metal assemblies, as shown in Figure 1.2. For these, components are stamped from flat sheet stock, then spot-welded into subassemblies and ultimately into a larger final assembly. Tolerances on both thickness and strength of the sheet stock, and location and number of spot-welds, are examples of input variations that can produce geometric variations in the final assembly which deviate from the design intent for the shape of the final assembly. Another example of input variation is that cold rolling leads to variations in the material properties of the rolled sheets When blanks are cut from these sheets, the cutting patterns are often optimized to minimize waste rather than maintaining consistent alignment. This results in the anisotropic properties from cold rolling being exhibited in different directions among the blanks.

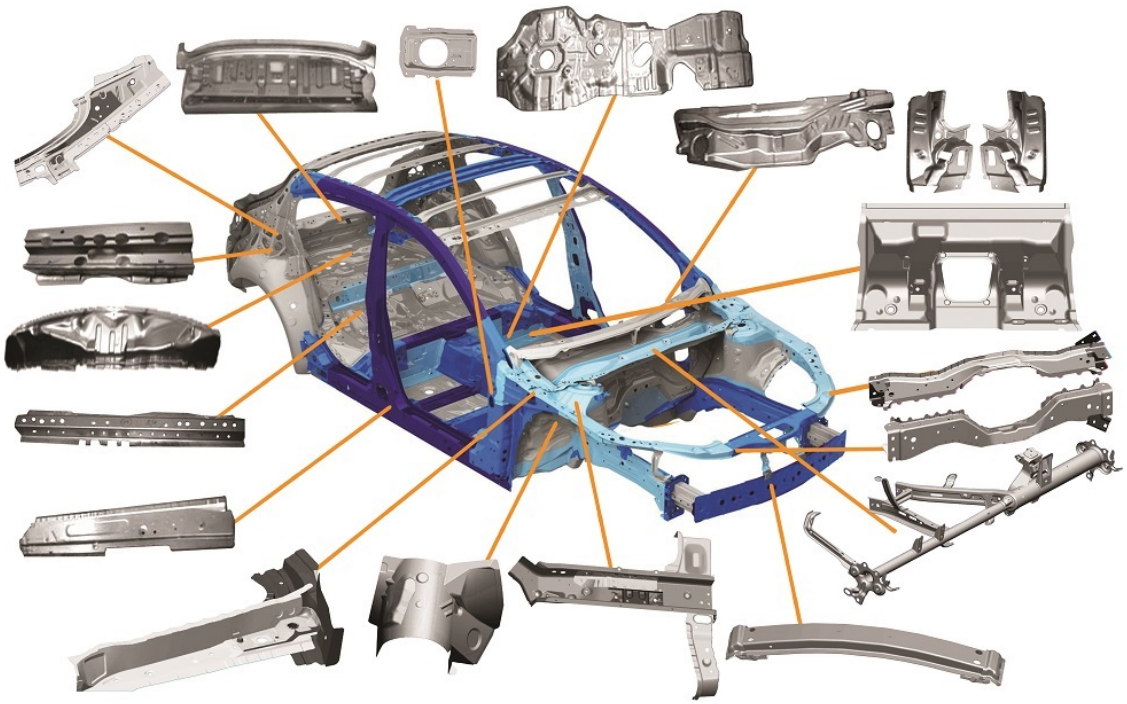


Figure 1.2: Exploded View of Automotive Body.

Additionally, forming operations, such as bending and stamping, cause elastic springback, where the components bounce back slightly after being shaped, leading to deviations from the desired geometry. Predicting springback can be challenging, especially for complex shapes and advanced materials. It can even vary between components of the same shape and material, depending on factors like the alignment of anisotropic material properties or minor misalignments in the tooling.

After forming, components are often joined into subassemblies concurrently to streamline the assembly process. However, any gaps at the interfaces between components due to imperfections from previous operations must be closed during clamping or fixturing before adding joints like spot welds. This forced closure introduces additional stresses, that affect the elastic spring-back in the subassembly once the clamps

are released. These subassembly-level deformations are influenced by the specific imperfections in each component and the constraints imposed by the joining method.

Historically, physical prototype assemblies were used to evaluate the geometric effects from all these sources of manufacturing variations. More recently, FEA simulations of the forming process and welding constraints provide a computer model for the geometric shapes of components and assemblies and how these geometries are affected by the different input variables.

For complete assemblies, the ultimate goal is to meet geometric tolerances for all combinations of allowable input variations.

### *1.1.2 Springback in Sheet Metal Forming*

As briefly mentioned above, springback is a term used to describe the recovery of elastic strains during unloading following a process of plastic deformation. This behavior is shown using the stress-strain plot in Figure 1.3. As the material is loaded past the yield strength ($\sigma_y$), the strain increases to the total strain ($\varepsilon_{total}$). During the unloading phase, the elastic strain ($\varepsilon_e$) is released (subtracted) from the total strain as stress and strain drop along the slope of the material's elastic modulus ($E$) until the load has been removed and the plastic strain ($\varepsilon_p$) remains.

While springback itself can refer to any type of elastic recovery after plastic deformation and release, its use and discussion in this thesis will be limited to springback in sheet metal forming. Most of the time, springback is a problematic side-effect of forming operations, as it causes formed components to deviate from their nominal design geometries. 2D representations of these deformations from springback [1] after forming processes are shown in Figure 1.4. Figure 1.4(a) shows the springback for a bending operation, while Figure 1.4(b) demonstrates springback following U-draw/bending. In both, springback causes bend angles to open and bend radii to

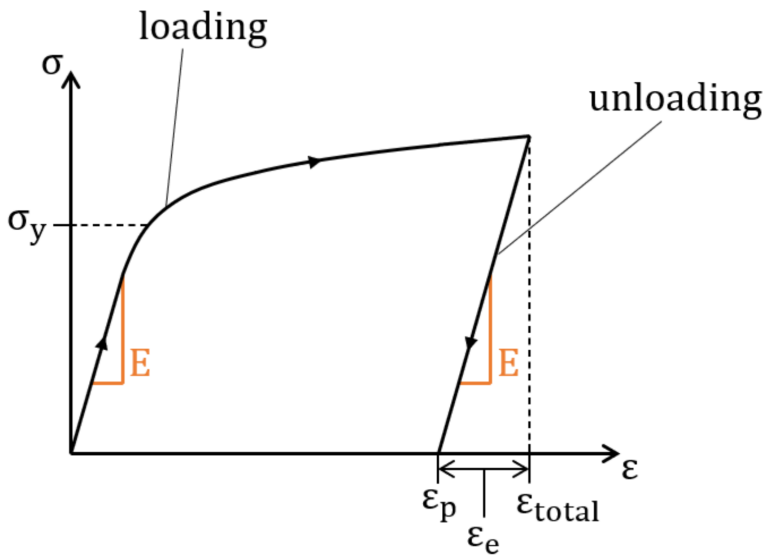



Figure 1.3: Stress-Strain Curve Showing Springback.

increase, but springback from U-draw/bending also induces curvature in the sidewalls after being drawn over the die radii. It is important to keep in mind that these are illustrations only; the actual amount of springback and final deformed shapes for each case are highly dependent on the materials, geometric dimensions of the blank and tooling, and process conditions. As such, prediction for even simple cases of springback can require the use of advanced analytical models or nonlinear FEA.

Bending and U-draw/bending processes are certainly utilized in the production of sheet metal components, but these geometries are much simpler than that for most of the components used in the complex automotive body assembly shown in Figure 1.2. To produce most of those components, stamping operations would be required. Figure 1.5 illustrates examples of springback along sections of stamped components similar to those that would be used in an automotive body. Components like these would not necessarily be extruded from 2D profiles, either; they are often intricate 3D designs. With such complex geometries, accurately predicting springback in an

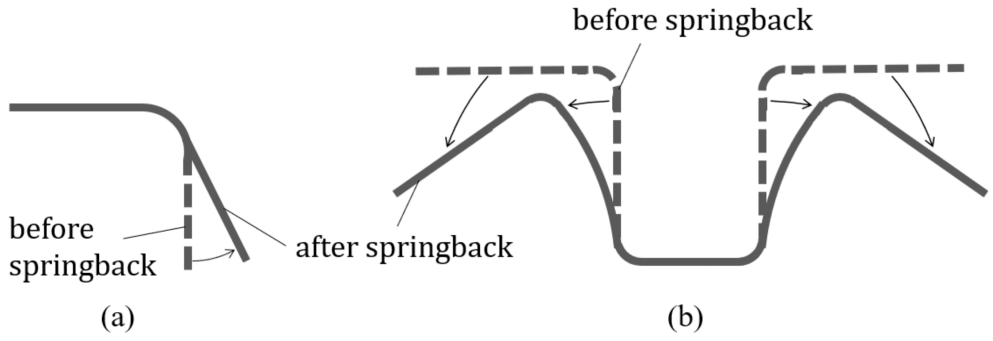



Figure 1.4: Illustrations of Springback in Sheet Metal Forming: (a) Bending Operation, (b) U-Draw/Bending [2].

efficient manner becomes a critical task in developing robust flexible assembly designs.

### *1.1.3 Least Squares Fittings of Geometric Shapes to FEA Simulated Data Points*

Once a component has been formed, or several components have been spot-welded together into an assembly (or subassembly), certain geometric parameters must be evaluated in order to qualify the part (assembly) for onward use. The FEA simulation(s) of this process provide sprung-back nodal points along edges, or within the material, that may be used to assess geometric quality. Least squares fitting is a mathematical method used to find the best-fitting profile (curve, line, plane) through such a set of data points. Although it is commonly used in regression analysis to estimate the parameters of a mathematical model that describes the relationship between variables, often the dependent variable is not very useful for evaluating quality of manufactured products. As one example, the usual regression fit of a line to the five points in Fig. 1.6 uses measures in the $y$-direction as deviations, yet, for evaluating manufacturing quality, the deviations at right angles to the fitted line, shown with dotted lines, are more useful. 'Orthogonal LS fitting' has been suggested as the

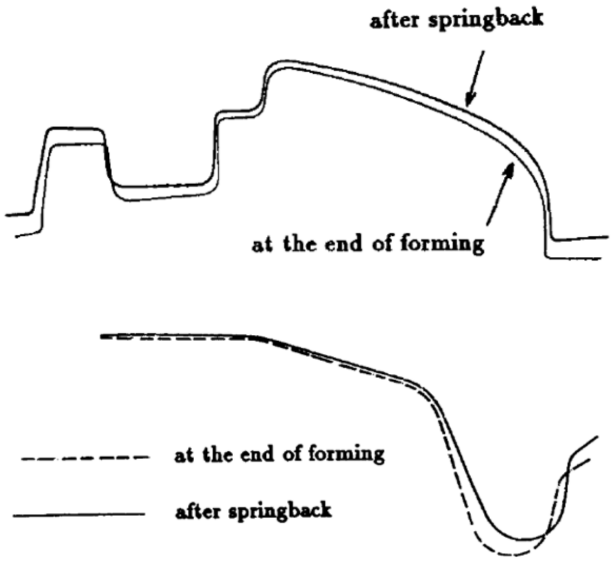



Figure 1.5: Examples of Springback in Sections of Complex Stamped Components [3].

accepted name for LS fits that use these deviations.

The basic idea behind least squares fitting is to locate the profile shape within the sprung-back data points so as to minimize the sum of the squared deviations between them and the fitted profile. This is achieved by adjusting the displacement parameters of the fitted shape until the sum of the squared differences is minimized.

In this paper, we have employed various types of least squares fitting techniques to accurately determine the boundaries, planes, and curves of stamped components, particularly hat sections and spot welded assemblies of these sections. The least squares fittings utilized are:

**Principal Component Analysis (PCA) for 3D Line Fitting** [4] is a method used to find the best-fitting line through a set of 3D data points by analyzing the principal components of the data covariance matrix. This method has been used

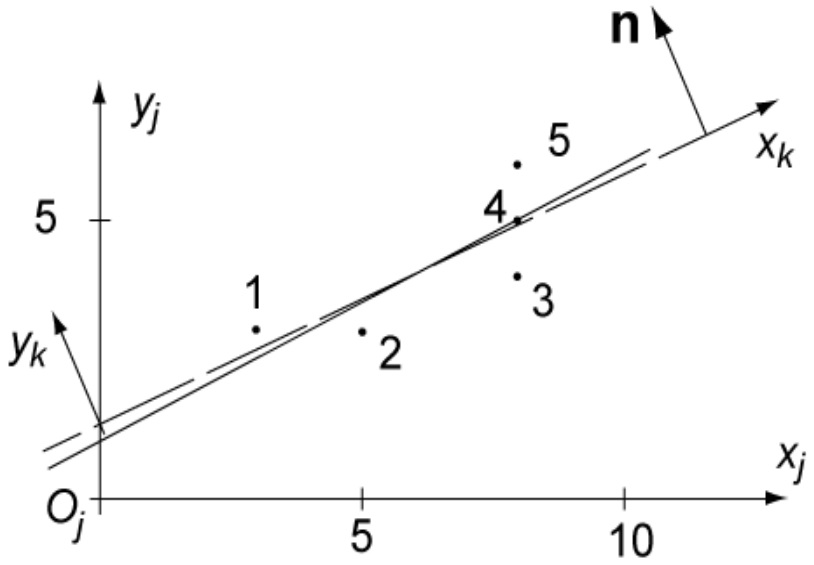



Figure 1.6: Five Points Equally Disposed About the Line $y = 1+x/2$ (Solid Line) in the $x_jy_j$-Frame and the Standard Regression Line (Dashed Line) for them.

in Chapter 2 for comparing the results with VW&R method and in Chapter 4 for fitting straight lines to calculate twist angles and angle between fitted planes. Here's a detailed description of the process along with the formulas involved:

1. **Compute the Mean of the Data**:

$$\bar{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \tag{1.1}$$

   Where $\mathbf{x}_i$ is the vector locating the $i^{th}$ data point in 3D space.

2. **Center the Data**: Subtract the mean from each data point:

$$\widetilde{\mathbf{X}}_i = \mathbf{x}_i - \bar{\mathbf{X}} \tag{1.2}$$

3. **Compute the Covariance Matrix**:

$$\mathbf{C} = \frac{1}{n}\sum_{i=1}^{n}\widetilde{\mathbf{X}}_i\widetilde{\mathbf{X}}_i^T \tag{1.3}$$

   Where $\mathbf{C}$ is the $3 \times 3$ covariance matrix.

4. **Find Eigenvectors and Eigenvalues**: Solve the eigenvalue problem for $\mathbf{C}$:

$$\mathbf{C}\mathbf{v}_k = \lambda_k \mathbf{v}_k \tag{1.4}$$

   where $\mathbf{v}_k$ is the $k^{th}$ eigenvector and $\lambda_k$ is the corresponding eigenvalue.

5. **Select the Largest Eigenvalue**: The eigenvector corresponding to the Largest eigenvalue represents the direction of the best-fitting line.

6. **Compute Line Parameters**: The line direction can be represented by the unit eigenvector $\mathbf{v}_{\max}$ corresponding to the largest eigenvalue.

7. **Compute Line Equation**: The line equation can be represented as:

$$\mathbf{x}_{\text{line}} = \bar{\mathbf{X}} + t\mathbf{v}_{\max} \tag{1.5}$$

   Where $t$ is a scalar parameter.

**The Kåsa Method** for 2D Arc Fitting [5] is a technique used to find the best-fitting circular arc through a set of 2D data points. This method has been used in Chapter 2 for comparing the results with VW&R method. Here's an explanation of the method mathematically: Given a set of data points $(x_i, y_i)$, where $i = 1, 2, ..., n$, representing points on or near the circumference of a circle, the goal is to find the parameters that define the circle's geometry, namely the center coordinates $(a, b)$ and the radius $r$.

The equation for a circle in 2D space is given by:

$$(x - a)^2 + (y - b)^2 = r^2 \tag{1.6}$$

To fit a circle to the given data points, the Kåsa Method aims to minimize the sum of squared distances between the data points and the circle's circumference.

The objective function to minimize can be formulated as:

$$F(a, b, r) = \sum_{i=1}^{n} \left( (x_i - a)^2 + (y_i - b)^2 - r^2 \right)^2 \tag{1.7}$$

The goal is to find the values of $a$, $b$, and $r$ that minimize this objective function.

This is a nonlinear least squares problem. The optimization process typically involves iterative numerical methods, such as the Levenberg-Marquardt algorithm [6] or gradient descent, to minimize the objective function and find the optimal values of $a$, $b$, and $r$.

Once the optimal parameters $a$, $b$, and $r$ are determined, they define the center and radius of the best-fitting circle to the given data points.

The Kåsa Method for 2D Arc Fitting is commonly used in various applications such as computer vision, image processing, and geometric modeling, where accurately fitting circular arcs to data points is essential.

**Fitting a Plane to Surface Points** using the pseudoinverse method involves finding the parameters of the plane equation that best fits a set of 3D data points. This method has been used in Chapter 4 for fitting planes to calculate angle between fitted planes. Here's an explanation of the method mathematically: Given a set of data points $(x_i, y_i, z_i)$, where i = 1, 2, ..., n, representing points on a flat surface or near a plane, the goal is to find the parameters that define the plane's geometry, namely the three coefficients of the plane equation $ax + by + cz = 1$. Steps involved:

1. **Formulate the Problem**: Given $n$ data points $\{(x_i, y_i, z_i)\}_{i=1}^{n}$, find $a$, $b$, and $c$ such that the plane $ax + by + cz = 1$ best fits these points in the least squares sense.

2. **Rewrite the Plane Equation**: The plane equation for each point $(x_i, y_i, z_i)$ can be written as:

$$ax_i + by_i + cz_i \approx 1 \tag{1.8}$$

3. **Set Up the System of Equations**: Express this system in matrix form:

$$A \begin{pmatrix} a \\ b \\ c \end{pmatrix} \approx \mathbf{1} \tag{1.9}$$

where

$$A = \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{pmatrix}, \quad \mathbf{1} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \tag{1.10}$$

4. **Multiply the equation (1.9) by transpose of** $A$ :

$$(A^T A) \begin{pmatrix} a \\ b \\ c \end{pmatrix} = A^T \mathbf{1} \tag{1.11}$$

5. **Solve the Equation**: Compute $(A^T A)^{-1}$ and $A^T \mathbf{1}$, then solve the resulting linear system:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = (A^T A)^{-1} A^T \mathbf{1} \tag{1.12}$$

Once we get the coefficient a, b and c , the parameters of the plane equation are determined, allowing us to describe the best-fitting plane to the given surface points

### *1.1.4 Minimum Zone and Zone Magnitude*

A zone magnitude is the normal distance between the closest pair of parallel planes that bound all the points. In the setting of geometric tolerances and tolerance analysis, it represents an unconstrained minimum zone that is used to quantify form

variations. The steps in computing the unconstrained minimum zone are as follows [7]:

1. **Point Selection**: Let $P = \{p_1, p_2, ..., p_n\}$ represent the set of relevant points in the region for which the zone magnitude is to be determined.

2. **Convex Hull Computation**: The "ConvexHull()" function from the "scipy.spatial" library is used to create a convex hull around the points in the data set. Let $H = \{h_1, h_2, ..., h_m\}$ denote the vertices of the convex hull and $F = \{f_1, f_2, ..., f_m\}$ denote the facets.

3. **Maximum Normal Distance of the Facets to the Vertices of the Convex Hull Computation**: For each facet of the convex hull, the normal distance to each vertex in the hull is computed, and the maximum one is selected and stored. This is achieved by iterating over each facet $f_i$, calculating the unit normal vector from the cross product of two vectors formed by three points on the facet, and then finding the dot product of this unit normal vector with all the vectors joining points in $H$ to one of the three points of the facet to get the normal distance of the point to the facet. Let $D = \{d_1, d_2, ..., d_m\}$ represent the maximum normal distances for each facet.

4. **Zone Magnitude Determination**: A search is undertaken on the list of maximum distances $D$, and then the minimum value is selected as the zone magnitude.

These steps provide a systematic approach to compute the unconstrained minimum zones, ensuring the accurate determination of zone magnitudes for the given components or assemblies under analysis. It is applied in Chapter 3 and 4.

### *1.1.5 Artificial Neural Networks (ANNs)*

Artificial Neural Networks (ANNs) are computational models inspired by the structure and function of biological neural networks in the human brain. They are a type of machine learning algorithm used for various tasks such as classification, regression, pattern recognition, and more.

An ANN consists of interconnected nodes or neurons organized into layers. The typical architecture of an ANN includes an input layer, one or more hidden layers, and an output layer. Each neuron in the network receives input signals, performs a computation, and passes the result to the neurons in the next layer.

The mathematics of implementing an ANN involves several key components discussed in [8]:

1. **Neuron Computation**: The computation performed by a neuron involves summing up the weighted inputs and applying an activation function. Let $x_1, x_2, ..., x_n$ be the input signals to the neuron, $w_1, w_2, ..., w_n$ be the corresponding weights, $b$ be the bias, and $f$ be the activation function. The output of the neuron $y$ is calculated as:

$$y = f(w_1x_1 + w_2x_2 + ... + w_nx_n + b) \tag{1.13}$$

2. **Feedforward Propagation**: Feedforward propagation involves passing the input data through the network to compute the output. The output of each neuron in a layer serves as the input to the neurons in the next layer, until the output layer is reached.

3. **Loss Function**: The loss function measures the difference between the predicted output of the network and the true output (labels) for a given set of

input data. Common loss functions include mean squared error (MSE) for regression tasks and categorical cross-entropy for classification tasks.

4. **Backpropagation Algorithm**: Backpropagation is an algorithm used to train the neural network by updating the weights and biases to minimize the loss function. It involves two main steps:

    - **Forward Pass**: Compute the output of the network using the current weights and biases.
    - **Backward Pass**: Compute the gradients of the loss function with respect to the weights and biases using the chain rule of calculus, and update the weights and biases using gradient descent or its variants.

5. **Optimization Algorithm**: Gradient descent or its variants such as stochastic gradient descent (SGD) [9], mini-batch gradient descent [10] , or Adam optimizer [11] are commonly used to update the weights and biases during the training process.

6. **Activation Functions**: Activation functions introduce non-linearity into the network and enable it to learn complex relationships in the data. Common activation functions include sigmoid, tanh, ReLU (Rectified Linear Unit), and softmax. A comparison of these activation functions is covered in [12].

7. **Regularization**: Techniques like L1 and L2 regularization, dropout, and batch normalization discussed in [13] are used to prevent overfitting and improve the generalization of the network.

By iteratively adjusting the weights and biases through the training process, the neural network learns to map input data to the desired output, effectively solving the task it was designed for.

### *1.1.6 Convolutional Neural Networks*

Convolutional Neural Networks (CNNs) [14] are a class of deep learning neural networks that are primarily used for image processing tasks such as image classification, object detection, and image segmentation. CNNs are particularly effective in capturing spatial hierarchies and patterns in images due to their ability to automatically learn hierarchical features from the data.

The key components of a CNN include convolutional layers, pooling layers, and fully connected layers. Here's an overview of the mathematics of implementing CNNs:

1. **Convolutional Layer**: The convolution operation involves sliding a filter (also known as a kernel) over the input image and computing the dot product between the filter and the corresponding patch of the input image. Mathematically, the output feature map $Y$ of a convolutional layer is computed as:

$$Y[i,j,k] = \sum_{m} \sum_{n} \sum_{l} X[i+m, j+n, l] \times W[m,n,l,k] + b[k] \tag{1.14}$$

   where: - $X$ is the input image. - $W$ is the filter weights. - $b$ is the bias term. - $i, j$ represent the spatial position in the output feature map. - $k$ represents the index of the filter. - $m, n$ iterate over the spatial dimensions of the filter. - $l$ represents the depth (or channels) of the input image.

2. **Activation Function**: After the convolution operation, an activation function like ReLU (Rectified Linear Unit) is applied element-wise to introduce non-linearity into the network.

3. **Pooling Layer**: Pooling layers reduce the spatial dimensions of the feature maps, helping in reducing computational complexity and controlling overfitting. Common pooling operations include max pooling and average pooling.

4. **Fully Connected Layer**: After several convolutional and pooling layers, the feature maps are flattened into a vector and fed into one or more fully connected layers. The fully connected layers perform classification or regression based on the learned features.

5. **Loss Function**: The choice of loss function depends on the task being performed, such as cross-entropy loss for classification tasks and mean squared error for regression tasks.

6. **Backpropagation and Optimization**: Backpropagation is used to compute the gradients of the loss function with respect to the network parameters, and optimization algorithms like gradient descent or its variants (e.g., Adam) are used to update the weights and biases of the network.

CNNs learn hierarchical features automatically from the data, making them powerful tools for image-related tasks. By adjusting the network architecture, parameters, and hyperparameters, CNNs can be tailored to a wide range of image processing tasks with impressive performance.

## 1.2 Research Objectives

the research specific to this thesis has four primary objectives:

1. Confirmation of the Virtual Work and Robotics (VW&R) method to get the least-squares fit of arc- and line-profiles to 3D arrays of CMM points (Chapter 2)

2. Construction of an automated workflow for extracting two geometric parameters from stamped hat component and their assemblies (Chapter 3) and 17 geometric parameters from Honda T joint (chapter 4).

3. Utilization of this dataset in training a machine learning algorithm for prediction of geometric Parameter (spot welded assemblies of stamped hat sections) with respect to manufacturing variations (chapter 3)

4. Extracting features and patterns from a large dataset of multi-view images of Honda hood frames to predict the maximum stress of the hoods using deep learning models.

Chapter 2

# THE VIRTUAL WORK AND ROBOTICS (VWR) METHOD FOR LEAST-SQUARES FITTING

The Virtual Work and Robotics (VWR) method is a refinement of the method described Sunkara [15] who produced locations of fitted profiles with a single pseudoinverse computation. But, when his single- pseudoinverse is applied to individual partitions of 3D data which are straight or circular, the results do not quite align with those from classical fitting methods. Therefore, the objective of this chapter is to explain why this discrepancy occurs and then to suggest an iterative solution to ensure that the VWR method converges to results consistent with those from classical methods. The chapter outlines a step-by-step procedure to use this refined method to find the least squares fits to line and circular arc profiles.

## 2.1 The VW&R Method in 2D

To illustrate the VW&R method, consider Fig. 2(a), which depicts the $x_n$-axis of a planar fixed reference frame $O_n x_n y_n$ aligned with the nominal straight edge-profile $\mathcal{P}$ (shown as a heavy dashed line) of a stamped component. The spring-back locations of five points, labeled 1 to 5, are shown lying in the same $x_n y_n$-plane. Now, against this plane, consider a separate phantom , or virtual, lamina. This lamina is able to move as the platform of a planar in-parallel and redundantly actuated robot, and has etched in it both the nominal shape of the profile (a straight line in Fig. 2(a)) and a reference frame $O_L x_L y_L$ initially coincident with $O_n x_n y_n$. Each sprung-back node is equipped with a linear actuator directed toward it at right angles to $\mathcal{P}$, although only the actuators for nodes 1 and 5 are fully shown in Fig. 2(a). Additional redundant

actuators are positioned at the remaining points, such as the one implied at node 2 with dashed lines. All five linear actuators of the platform robot apply vertical forces on the platform (depicted in Fig. 2(b)), and the small displacements of the platform are constrained to vertical translation $\Delta y$ and rotation $\Delta\theta_z$ in the plane. Consequently, except for one, all actuators are connected to both the base and the platform via rotary joints. The joint on the platform for each actuator is at the etched profile on the platform, not at the corresponding fixed data point as suggested in[16,17], and the base joints for the actuators are placed far away to maintain vertical forces on the platform during small displacements. Although the exceptional actuator can be any one of the five actuators, in Fig. 2(a) it is shown as actuator $\$_1$. It is firmly attached to the base to prevent platform translation in the $x$-direction. The five forces $F_i$ (where $i = 1, \ldots, 5$), applied along lines $i$ at the actuators corresponding to nodal points $i$, generate a wrench $(T; F)$ exerted by the platform on the environment, expressed in frame $O_n x_n y_n$ (Fig. 2(b)). Force $F$ represents the resultant of these forces, while $T$ is an accumulation of their moments about $O_n$.

Consider a hypothetical scenario where all actuators except one are free to move and exert no force to the platform except one, say $F_1$ acting on line $\$_1$ (Fig. 2.1 (c)), are free to move and exert no force on the platform. Then actuator force at line $\$_1$ generates $(T_1; F_1)$, a part of the total produced wrench $(T; F)$. The partial wrench $(T_1; F_1)$ is illustrated in Fig. 2.1(c) with the scalar components $T_{z1}$ and $F_{y1}$. Although not infinitesimal, spring-back displacements are small relative to part dimensions in relevant applications. Consequently, the principle of virtual work can be utilized to approximate closely relationships among the forces.

A virtual small displacement $d_1$ at $\$_1$ (Fig. 2.1(a)) would induce a kinematically consistent small displacement of both the platform and the moveable frame $O_L x_L y_L$ from their initial alignment with the nominal profile and fixed frame $O_n x_n y_n$.

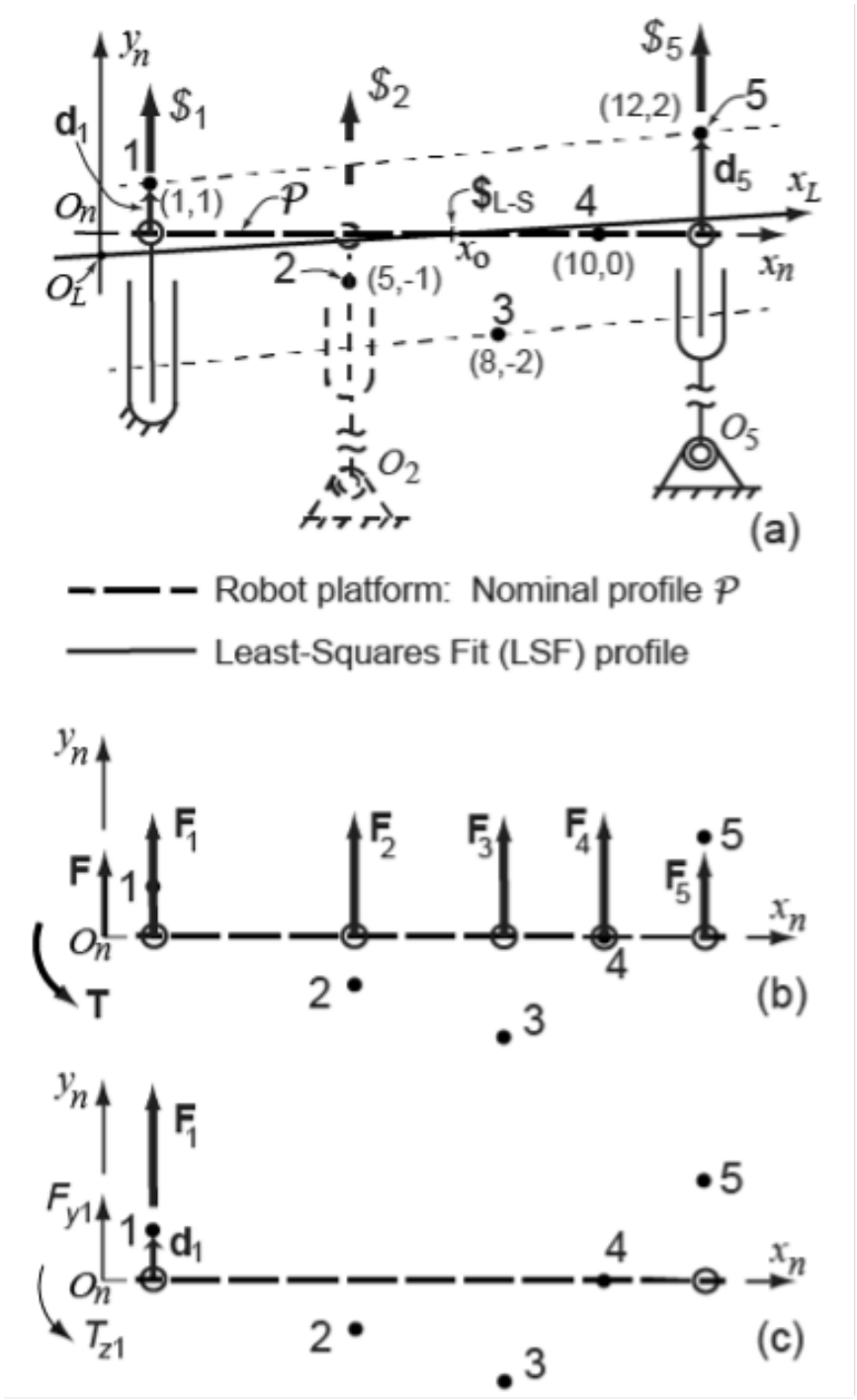



Figure 2.1: Least-Squares Fit (LSF) of a Line to Five Points in a Plane and Arrayed Along a Nominal Line P Lying on the $x_n$-Axis. (b) Wrench (**T** ;**F**) Produced by Forces Acting at all Actuators. (c) Wrench Produced by the Actuator at One Nodal Point.

This displacement can be expressed using the small displacement torsor (SDT) [$\$_L$] =$[\Delta\theta_z \quad \Delta y]^{\mathrm{T}}$, where $\Delta y$ represents the resulting displacement of origin $O_L$ relative to $O_n$, and $\Delta\theta_z$ measures the rotation of the platform relative to $O_n x_n y_n$. Since the virtual work of all forces and moments on a free body must be in equilibrium for a kinematically valid small displacement [$\$_L$] of the platform consistent with $d_1$, the force and couple in the special case depicted in Fig. 2.1(c) lead to

$$F_1 d_1 = \begin{bmatrix} T_{z1} & F_{y1} \end{bmatrix} \begin{bmatrix} \Delta\theta_z \\ \Delta y \end{bmatrix} = F_{y1} \begin{bmatrix} x_1 & 1 \end{bmatrix} \begin{bmatrix} \Delta\theta_z \\ \Delta y \end{bmatrix} \tag{2.1}$$

The term $F_1 d_1$ represents the virtual work of force $F_1$ displacing the platform by small amount $d_1$, both of which are in the direction of $\$_1$, at nodal point 1. The product $[T_{z1} \quad F_{y1}] \begin{bmatrix} \Delta\theta_z \\ \Delta y \end{bmatrix}$ represents the corresponding virtual work produced by the resultant wrench $(T_1; F_1)$ that acts on the environment with virtual displacement $[S_L]$ of the platform. It's essential to note that Eq. (2.1) is purely geometric. In the loading condition of Fig. 2.1(c), where the platform behaves as a two-force member, the force magnitude remains the same on both sides, i.e., $F_1 = F_{y1}$. Removing this common multiplier yields the second form of Eq. (2.1), which is the first scalar geometric equation among five. Similar hypothetical scenarios lead to four more equations (2.1) for actuators $\$_2 ... \$_5$. Collecting all five geometric equations in matrix form,

$$[d_i] = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_5 \end{bmatrix} = [K][S_L] = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_5 & 1 \end{bmatrix} \begin{bmatrix} S_L \end{bmatrix} \tag{2.2}$$

Although there is no unique solution $[S_L]$ for the five linear equations in Exp. (2.2), the matrix $[K]$ can be inverted using a generalized inverse [18] to derive a single SDT

$[S_L]$ that meets a condition among the displacements $d_i$ of the nodal points.

A pertinent condition for metrology is to minimize the sum of squared deviations of the nodal points from the LS profile, expressed as $\Sigma\{d_i - (x_i\Delta\theta_z + \Delta x)\}^2$. This condition corresponds to the Moore-Penrose inverse (pseudoinverse) $[K]^{\#}$ of matrix $[K]$ as outlined in the second equation of

$$[S_L] = [K]^{\#}[d] = ([K]^T[K])^{-1}[K]^T[d] \tag{2.3}$$

[19] and to the least-squares location for the platform carrying $\mathcal{P}$, represented by SDT $[\mathbf{\$_L}]$ . For the five points in Fig. 2.1(a), the input vector $[d] = \begin{bmatrix} 1 & -1 & -2 & 0 & 2 \end{bmatrix}^T$ leads to the solution $[\mathbf{\$_L}]$ (rad, mm) scaled and aligned with the $x_L$-axis. The matrix $[K]$ in Exp. (2) is formed with five numerals 1 and the $x_i$-coordinates for the five points in Fig. 2.1(a).

Although we indicate in Fig. 2.1(a) that the constraint on displacement in the x-direction is at the hinge attachment for the actuator at nodal point 1, according to the platform SDT $[\mathbf{\$_L}]$, this constraint is located at that point $x_o = -\Delta y/\Delta_z$ on the platform and etched profile $\mathcal{P}$. This point can therefore be considered as the fixed center of the axis $\mathbf{\$_L}$ for a single rotation that aligns the nominal profile $\mathcal{P}$ with the LSF line. Based on the values provided in Fig. 2.1, VW&R yields $x_o = 7.2$ mm.

For comparison, another method used to fit a straight line to n points in any dimension is Principal Component Analysis (PCA) [4]. PCA is discussed in details in section 1.1.3.

The fitted PCA line aligns with this direction. The PCA-fitted line, for the points in Fig. 2.1, is oriented at a 3.524-degree angle from the $x_n$ -axis, equivalent to direction ratios (0.9981 , 0.0615). Additionally, the PCA-fitted line passes through the average point $(x, y)_{av} = (7.2, 0)$ mm. Perpendicular to this direction is the principal direction with the least variance, representing the direction with the smallest least squares sum

($LS\Sigma$) of point deviations from the PCA direction.

Table 2.1: LS$\Sigma$s and SDTs for Different Methods to Fit a Line to Five Points Lying in a Plane (Fig. 2).

| **Method** | **LS$\Sigma$** | **SDT ($\Delta\theta_z$, $\Delta y$) (rad, mm)** |
|---|---|---|
| VW&R, iteration 1 | 9.75819 | $(0.0535, -0.385)$ |
| VW&R, iteration 2 | 9.75409 | $(0.06077826, -0.43511632)$ |
| VW&R, iteration 3 | 9.754021 | $(0.06172738, -0.44162687)$ |
| PCA | 9.754022 | $(0.0615, -0.4428)$ |

The form of Exp (2.2) is the same as that used in the the forward velocity problem associated with redundantly actuatedplatform (parallel actuated) robots (referenced in, for instance, [20] or Box KIN.III in Chart 1 of Ch. 8 in [21]). Although in such a problem matrix $[K]$is the same as for fitting data points, the matrix $[d]$ becomes a list of velocities at the linear actuators, and the pseudoinverse solution produces the velocity state (twist) $\$_{vel} = (w_x, w_y, w_z; \dot{x}, \dot{y}, \dot{z})$ of the platform that corresponds to the minimum LS of all the actuator speeds. Here, $\dot{x}$, $\dot{y}$, and $\dot{z}$ denote the translational velocity components of origin $O_L$ on the platform relative to the fixed origin $O_n$ in Fig. 2.1. Such velocity relations have three characteristics that distinguish them from a fitting problem: (a) velocity relations are instantaneous; (b) all deviations (displacements) tend to zero in the limiting process; and (c) coordinate frames $O_L x_L y_L$ and fixed frame $O_n x_n y_n$ coincide and are never distinct.

However, in the context of fitting a profile to sprung-back nodal points, the passage of time becomes irrelevant, and all displacements are finite. This situation leads to frame $O_L x_L y_L$ no longer being coincident with $O_n x_n y_n$. To compute the Least Squares Sum ($LS\Sigma$) for a fitted line (represented by axis $x_L$ in Fig. 2.1(a)), it's necessary to

apply Eqn (A6) to transform all original deviations $d_i$, which are perpendicular to axis $x_n$ in Fig. 2.1(a), into deviations perpendicular to the fitted line $x_L$. As successive iterations are carried out, the $[\$_L]$ utilized in Eqn (A6) becomes the sum of $[\$_{L1}] + [\$_{L2}] + \ldots$, leading to decreasing differences in $LS\Sigma$ and other parameters between two successive sets of deviations until convergence is achieved. The computational process for the points in Fig. 2.1, when utilized for a single component direction, is outlined in the flowchart presented in Fig.2.2

## 2.2 The VW&R Method in 3D:

### *2.2.1 Straight Line Profile*

Figure 2.3 illustrates an example of a 3D array of points arranged along a straight profile $\mathcal{P}$ that is aligned with the $x_n$-axis of the nominal coordinate frame $O_n x_n y_n z_n$. Before a FEA simulated component is released from the forming press, or an assembly from its fixture, the points all lie on nominal profile $\mathcal{P}$, i.e. all deviations are zero. However, after release from the press or fixture, the simulations for residual stress and elastic spring-back lead to the coordinates $(x_i y_i z_i)$ that are used for obtaining the LSF profile. In general, all of these points are displaced laterally from $\mathcal{P}$. Although the points in Fig. 2.3 are a contrived set of data points, the large deviations make the figure easier to interpret.

It is important to note that, in 3D, the direction of a single actuator line $\$_i$ at a specific nodal point cannot be determined beforehand. This uncertainty leads to two consequences: (a) a single actuator line $\$_i$ and vector $d_i$, unlike the points in the 2D plane of Fig. 2.1, may not maintain alignment during the displacement of the nominal profile $\mathcal{P}$ to its location for the Least Squares Fit (LSF), and (b) representing virtual work at each actuator requires using the scalar of product of *two vectors*, $\mathbf{F}_i \cdot \mathbf{d}_i$,

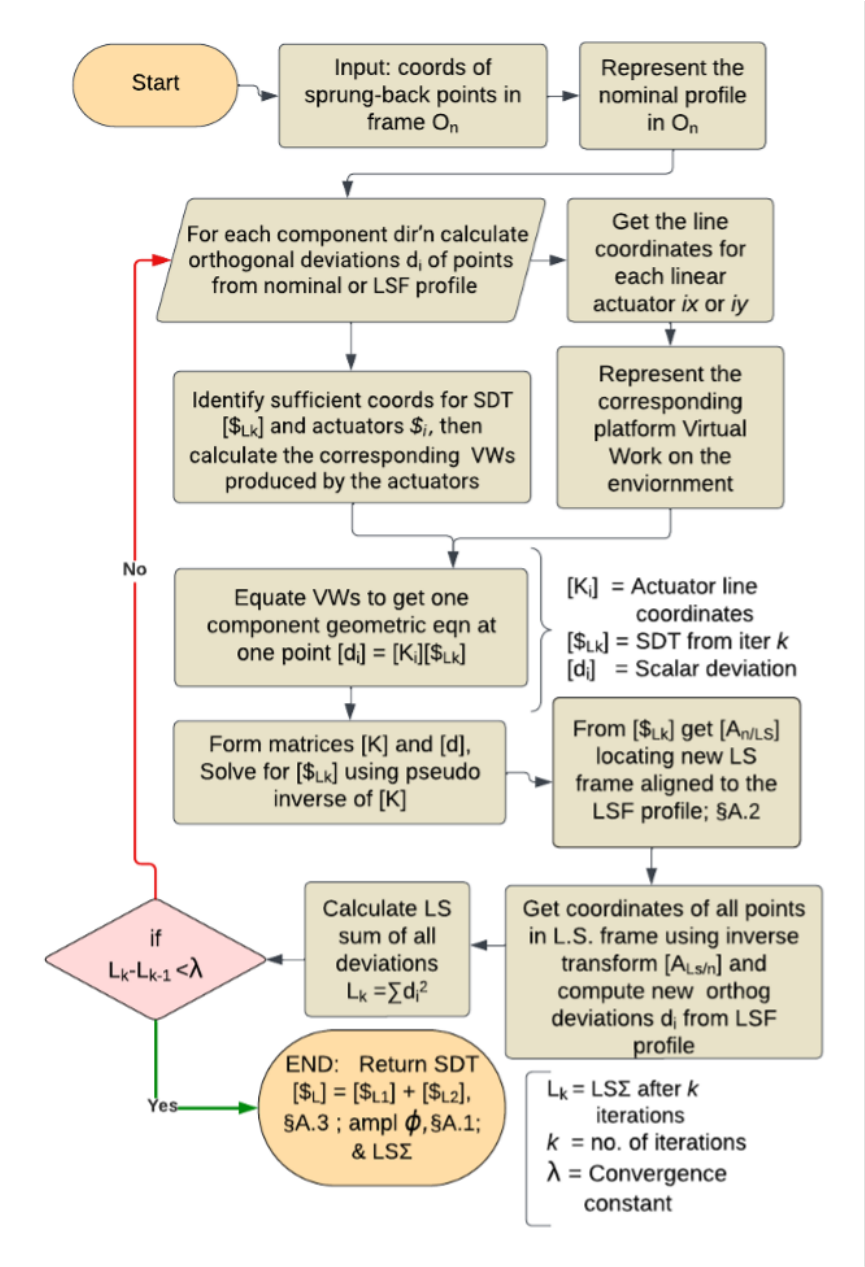



Figure 2.2: Flowchart of Computations for the VW&R Method. Symbol § Identifies a Subsection in Appendix A.

instead of the simple product of their magnitudes $F_i d_i$ as depicted on the left side of Eq.(2.1).

To accurately represent this scalar product for each nodal point i, we decompose both vectors $\mathbf{F}_i$ and $\mathbf{d}_i$ into their $x$- and $y$-components and assign an actuator for each component. Since this decomposition method necessitates two actuators for each point in the five-point cloud of Fig. 2.3, we require five pairs of linear equations organized as in Exp. (2.2). Each scalar force balance equation is then formed with a four-component wrench and SDT $[\$_L] = [\Delta\theta_x \quad \Delta\theta_y \quad \Delta x \quad \Delta y]^T$. Furthermore, all ten actuators are connected to both the base and the virtual platform using spherical joints. The base spheres are positioned far away to ensure that the forces applied on the platform remain perpendicular to $\mathcal{P}$ during small displacements. Optionally, one actuator could be linked to the base with a hinge, aligned parallel to the $z$-axis, to prevent translation of the platform in the z-direction. However, the Least Squares (LS) formulation automatically provides this constraint through the SDT $\$_L$ (Fig. 2.3), which carries the etched profile $\mathcal{P}$ on the robot platform (virtual body) from its nominal location aligned with $(O_n x_n y_n z_n)$ to its LS location shown $(O_L x_L y_L z_L)$. For any point $i$, the virtual work expressions in the $x$ - and $y$-directions lead to

$$F_{ix} d_{ix} = \begin{bmatrix} 0 & T_{iy} & F_{ix} & 0 \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_y \\ \Delta x \\ \Delta y \end{bmatrix} = F_{ix} \begin{bmatrix} 0 & z_i & 1 & 0 \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_y \\ \Delta x \\ \Delta y \end{bmatrix} \tag{2.4}$$

and

$$F_{iy}d_{iy} = \begin{bmatrix} T_{ix} & 0 & 0 & F_{iy} \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_y \\ \Delta x \\ \Delta y \end{bmatrix} = F_{iy} \begin{bmatrix} -z_i & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_y \\ \Delta x \\ \Delta y \end{bmatrix} \tag{2.5}$$

respectively, where $F_{ix}$ and $F_{iy}$ represent the x- and y-components of forces $\mathbf{F}_i$ acting at sprung-back nodal point $i$, while $T_{ix}$ and $T_{iy}$ denote the x- and y-components of the moment of $\mathbf{F}_i$ (i = 1,. . . ,5) about origin $O_n$ in the $O_n x_n y_n z_n$-frame (depicted in Fig. 2.3). Each of these equations serves as a broader version of Eq. (2.1), collectively forming the set of 10 over-constrained linear equations used to formulate the Least Squares Fit (LSF) of a straight line profile to the cloud of five points in Fig. 2.3. It's worth noting that the two actuators illustrated for point 2 are duplicated twice more, once for point 1 and again for point 3.

The flowchart presented in Fig. 2.2 outlines the complete computation process for the VW&R method applied to the points in Fig. 2.3. The outcomes for three iterations are depicted in the initial three columns of Table 2.2, while the results for the Principal Component Analysis (PCA) method (referenced at the end of §2.1) are detailed in the last column. For a fair comparison with the PCA method, the most suitable comparison is with the coordinates for $\$_L$ in the third column of Table 2.2 from the third iteration. The total Small Displacement Torsor (SDT) $\$_L = \$_{L1} + \$_{L2} + \$_{L3}$ (depicted in Fig. 2.3), encompassing its pitch, amplitude, and axis of rotation, carries the profile shape from its initial location to its LSF location referenced to frame $O_n x_n y_n z_n$. It's important to note that the axis of rotation is distinct from both the nominal and the LSF line-profiles. Parameters $\Delta\theta_x$ and $\Delta\theta_y$ represent the small displacements of the platform about the $x_n$- and $y_n$-axes, while $\Delta x$ and $\Delta y$ signify the $x_n$- and $y_n$-coordinates of origin $O_{L3}$ in the $O_n x_n y_n z_n$-frame.

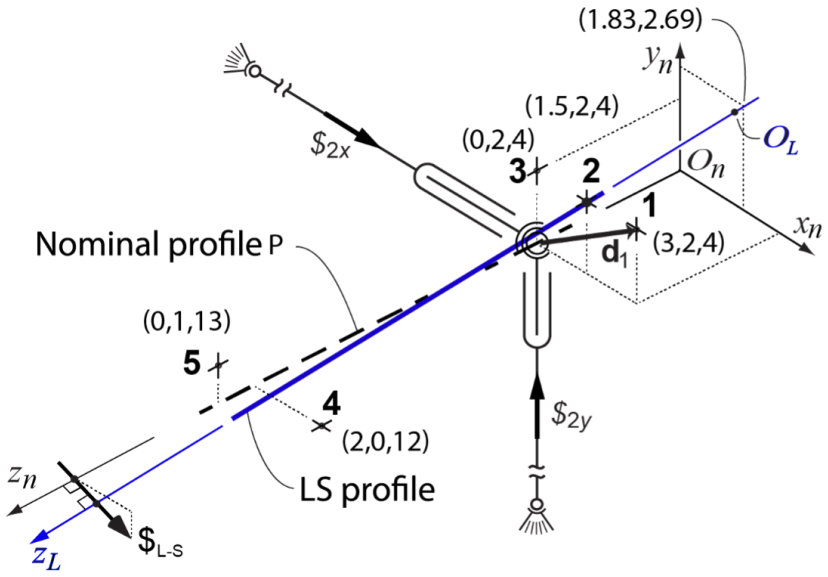



Figure 2.3: LSF of Five Points, with Coordinates $(x_n y_n z_n)$, Arranged in 3D Around a Nominal line $\mathcal{P}$ Lying on the $z_n$-axis. Figure is to Scale and Values are in mm.

The amplitude $\phi$ of $\$_L$ is $\phi = \sqrt{(\Delta\theta_x)^2 + (\Delta\theta_y)^2} = \sqrt{(0.16821)^2 + (0.07193)^2} = 0.18247$ rad, the pitch of $\$_L$ is $h = 3.52517$ mm/rad (App. A1), and the corresponding axial travel of the platform carrying the etched profile $\mathcal{P}$ is $s = h\phi = 0.64323$ mm. The location of $\$_L$ is determined from the coordinates of the line on which it lies. Using the relations listed in App. A1, the normalized coordinates of this line are $(L, M, N; P_l, Q_l, R_l) = (0.91946, -0.39318, 0; 6.73325, 15.74588, 0)$. Then, using App. (A2), we see that $\$_L$ intersects the $z_n$-axis ($(x_n, y_n) = (0, 0)$) at $z_n = \frac{Q_l}{L} = -\frac{P_l}{M} = 17.12510$ mm (Fig. 2.3).

### *2.2.2 Circular Arc Profile*

Similar to the line-fit discussed in §2.2.1, the robot platform acts as a phantom body that carries the etched profile $\mathcal{P}$ from its original position $(O_n x_n y_n z_n)$ to its location determined by the Least Squares (LS) fit $(O_L x_L y_L z_L)$. This displacement is

Table 2.2: For Fig. 2.3, Comparison of $\$_L$ Parameters Across Different Iterations and PCA.

| $\$_L$ parameters | Iter #1 | Iter #2 | Iter #3 | PCA |
|---|---|---|---|---|
| $\Delta\theta_x$, rad | 0.16972 | 0.16826 | 0.16821 | 0.16819 |
| $\Delta\theta_y$, rad | -0.06995 | -0.07192 | -0.07193 | -0.07196 |
| $\Delta x$, mm | 1.81766 | 1.82474 | 1.82478 | 1.82480 |
| $\Delta y$, mm | 2.65596 | 2.62711 | 2.62705 | 2.62702 |
| $\phi$, rad | 0.18356 | 0.18298 | 0.18294 | 0.18293 |
| $h$, mm/rad | 3.64144 | 3.52673 | 3.52517 | 3.52215 |
| $L$, rad | 0.92455 | 0.91952 | 0.91946 | 0.91938 |
| $M$, rad | -0.38105 | -0.39303 | -0.39318 | -0.39335 |
| $N$, rad | 0.0 | 0.0 | 0.0 | 0.0 |
| $P_l$, mm | 6.53503 | 6.72909 | 6.73325 | 6.73677 |
| $Q_l$, mm | 15.85597 | 15.74301 | 15.74588 | 15.74567 |
| $R_l$, mm | 0.0 | 0.0 | 0.0 | 0.0 |
| $zn$-intersect, mm | 17.14988 | 17.12084 | 17.12510 | 17.12630 |
| LS$\Sigma$, mm$^2$ | 7.03994 | 7.03734 | 7.03734 | 7.03734 |

represented by the Small Displacement Torsor (SDT) $\mathbf{\$_L}$ . When dealing with an arc, the breakdown of the dot product $\mathbf{F}_i \cdot \mathbf{d}_i$ necessitates two actuators, both oriented perpendicular to the identifiable tangent line at each sprung-back nodal point. These actuators exert force through that point ($A_i$ in Fig. 2.4) on the nominal profile closest to the nodal point $B_i$. One actuator is situated in the $zx$-plane, while the other is oriented in the $y$-direction (as depicted in Fig. 2.4).

Since all actuators intersect the y-axis (those in the y-direction extending to

infinity), they cannot influence the SDT coordinate $\Delta\theta y$. Consequently, $[\mathbf{\$_L}] = [\Delta\theta_x \quad \Delta\theta_z \quad \Delta x \quad \Delta y \quad \Delta z \quad \Delta r]^T$, and coordinate $T_{iy}$ can be omitted from each row of the matrix [K ] (from Eqs. (2.4) and (2.5)) so that each actuator force (each forming a zero pitch wrench) comprises five coordinates: $(T; F) = (T_x, T_z; F_x, F_y, F_z)$. Then, if we take for an example, the nominal radius of the arc at the inner edge of the upper part of the assembly to be 250 mm, Eqs. (2.4) and (2.5) in §2.2.1 become

$$F_{izx}d_{izx} = \begin{bmatrix} 0 & 0 & F_{ix} & 0 & F_{iz} \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_z \\ \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = F_{izx} \begin{bmatrix} 0 & 0 & -\frac{A_{ix}}{\mu_i} & 0 & -\frac{A_{iz}}{\mu_i} \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_z \\ \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \quad (2.6)$$

and

$$F_{iy}d_{iy} = \begin{bmatrix} T_{ix} & T_{iz} & 0 & F_{iy} & 0 \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_z \\ \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = F_{iy} \begin{bmatrix} -250C_{\alpha_i} & 250S_{\alpha_i} & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \Delta\theta_x \\ \Delta\theta_z \\ \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \quad (2.7)$$

where $F_{izx}$ and $F_{iy}$ are the components of total force $F_i$ acting at point $A_i$ in the zx-plane and in the y-direction, respectively (Fig. 2.4), along the actuators $\$_{izx}$ and $\$_{iy}$; $T_{ix}$ and $T_{iz}$ are components of moment in xyz from force component $F_{iy}$; and deviations $d_{izx}$ and $d_{iy}$ are illustrated in Fig. 2.4. Further, $F_{izx} = \sqrt{F_{ix}^2 + F_{iz}^2}$, $\mu_i = \sqrt{A_{ix}^2 + A_{iz}^2}$, $C\alpha_i = \cos\alpha_i$, and $S_{\alpha_i} = \sin\alpha_i$, and $\alpha_i = \text{atan2}\left(\frac{A_{ix}}{A_{iz}}\right)$. Pairs of rows for matrix $[K]$ in Expr (2.2) are extracted as the second form from Eqs. (2.6)

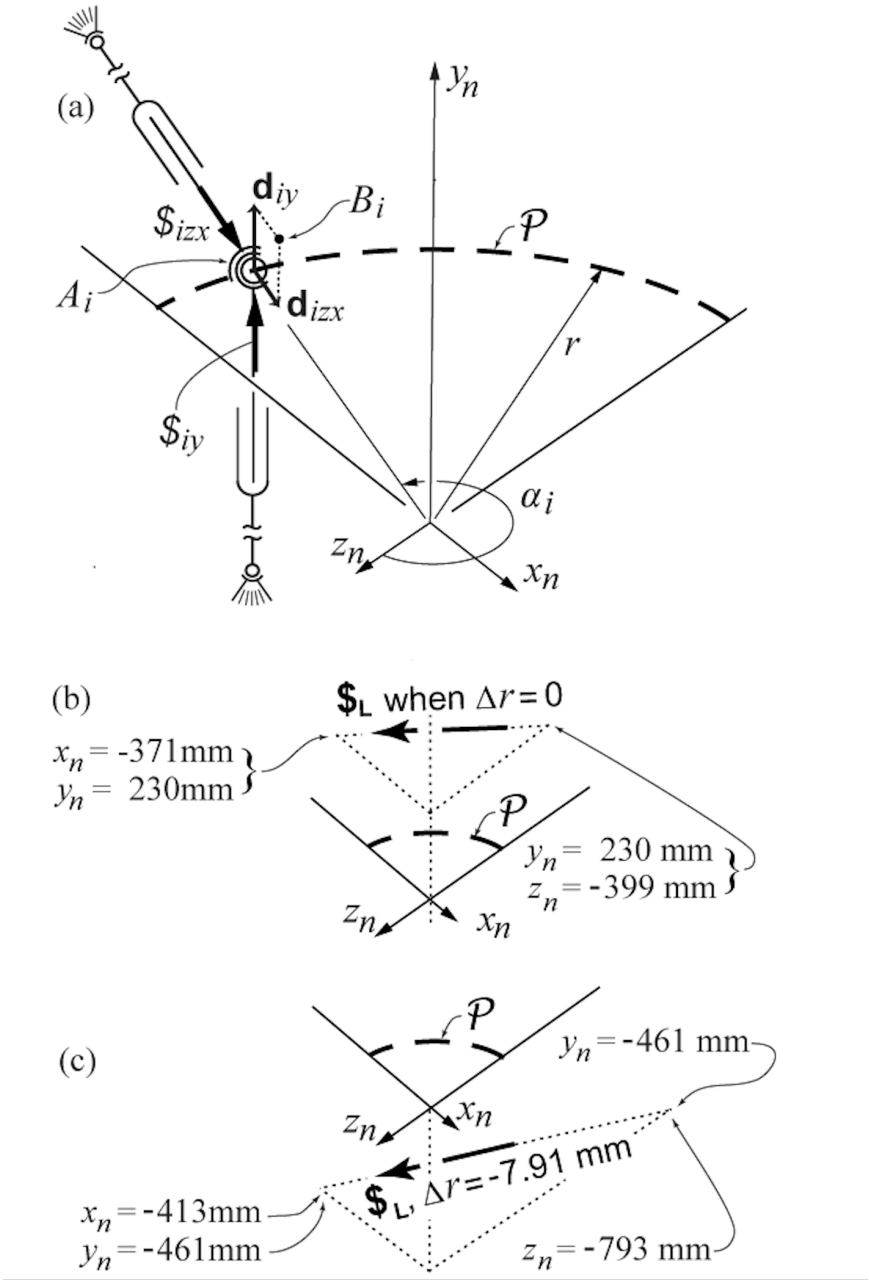



Figure 2.4: (a). Nominal 90° Arc-Profile $\mathcal{P}$ of Radius $r$=250 mm, Sample Spring-Back point $B_i$, its Deviation Vector $\mathbf{d}_i$ Decomposed into Components $d_{izx}$ and $d_{iy}$, its Actuators Represented with Lines $\$_{izx}$and $\$_{iy}$, and the Locations of SDT $\$_{\mathrm{L}}$ after Three Iterations of VW&R; (b). At a Smaller Scale, when Nominal Radius of 250 mm is Fitted to the Data. (c). Same as (b), but After Three Iterations of VW&R and when Change of Radius is Included in the Fitting.

and (2.7) found as the normalized coordinates of lines $\$izx = \begin{bmatrix} 0 & 0 & F_{ix} & 0 & F_{iz} \end{bmatrix}$ and $\$iy = \begin{bmatrix} T_{ix} & T_{iz} & 0 & F_{iy} & 0 \end{bmatrix}$. Therefore, Expr (2.2) becomes

$$
\begin{aligned}
[d_i] &= \begin{bmatrix} d_{1zx} & \cdots & d_{mzx} & d_{1y} & \cdots & d_{my} \end{bmatrix}^T \equiv [K][\$_L] \\
&= \begin{bmatrix} 0 & 0 & -\frac{A_{1x}}{\mu_1} & 0 & -\frac{A_{1z}}{\mu_1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & -\frac{A_{ix}}{\mu_i} & 0 & -\frac{A_{iz}}{\mu_i} \\ -250C_{\alpha_i} & 250S_{\alpha_i} & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -250C_{\alpha_n} & 250S_{\alpha_n} & 0 & 1 & 0 \end{bmatrix} [\$_L].
\end{aligned} \tag{2.8}
$$

Given that, during Finite Element Analysis (FEA) simulation of stamping from a flat blank, the center radius (which was 315 mm in this instance) of the 90° arc-component (depicted at the top of Fig. 2.1(a)) was restricted to vertical movement, the inner edge experienced tension post-forming, resulting in a reduction in radius during springback. Conversely, the outer edge was subjected to compression and sprang back to a larger radius. This implies that a more accurate fit can be achieved when the radii of the curved edges are allowed to vary, along with displacement.

Hence, Fig. 2.4 presents two outcomes for the Small Displacement Torsor (SDT) $\$_L$ after fitting the inner edge of the arc: one where the radius change is considered (fig 2.4(c)) and one where it is not (Fig 2.4(b)). In the scenario where radius change is disregarded, Eqs. (2.6) and (2.7), along with Expr. (2.8), are applicable, resulting in the fitted arc having a radius of 250 mm.

To incorporate size variation when fitting an arc to the points, we must not only consider the nominal profile etched on the phantom robot platform but also all concentric profiles. Furthermore, an additional coordinate $\Delta r$ needs to be added to the

$\$_L$ vector, i.e., $[\mathbf{\$_L}] = [\Delta\theta_x \quad \Delta\theta_z \quad \Delta x \quad \Delta y \quad \Delta z \quad \Delta r]^T$ , and another coordinate at the end of each row of the matrix $[K]$ in Expr. (2.8). This modification ensures that the two rows of $[K]$ for any given point $i$ become

$$\begin{bmatrix} 0 & 0 & -\frac{A_{ix}}{\mu_i} & 0 & -\frac{A_{iz}}{\mu_i} & -1 \\ -250C_{\alpha_i} & 250S_{\alpha_i} & 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.9}$$

The negative signs in the last column of $[K]$ have the effect of associating an increase in radius with $+\Delta r$, essentially reversing the positive orientations shown in Fig. 2.4 where extensions of actuators $\$_{izx}$ result in a reduced radius of the arc. It's important to note that the consideration of size change is selective, as it only impacts actuators in the zx-plane.

Table 2.3: Coordinates and Some Derived Values for Nine Sample Sprung-Back Points, Nearly Evenly Spaced, Along the Inner Edge of the Arc from A. Joshi[22].

| Pt | $x_n$ | $y_i, diy$ | $z_i$ | $di_{zx}$ |
|---|---|---|---|---|
| 1 | -241.50 | 0.9822 | 1.042 | 8.50 |
| 2 | -239.20 | 1.7540 | -45.51 | 6.509 |
| 3 | -226.11 | 1.5195 | -95.78 | 4.439 |
| 4 | -207.03 | 1.3136 | -135.02 | 2.837 |
| 5 | -175.07 | 1.2240 | -176.91 | 1.109 |
| 6 | -141.82 | 1.3043 | -205.99 | -0.090 |
| 7 | -95.55 | 1.6049 | -232.31 | -1.193 |
| 8 | -53.22 | 2.0476 | -246.17 | -1.857 |
| 9 | -0.024 | 2.7479 | -252.12 | -2.120 |

The FEA data utilized for fitting the arc are documented in Table 2.3 and origi-

nate from early FEA simulations of stamping and assembly conducted by Joshi [22]. The flowchart depicted in Fig. 2.2 outlines the computation process of VW&R for these data alongside the geometry presented in Fig. 2.4. The outcomes for three iterations are presented in Table 2.4, with the coordinates for $\$_L$ from the third iteration listed in the final row of Table 2.4. The total Small Displacement Torsor (SDT)

$\$_L$, which comprises $\$_{L1} + \$_{L2} + \$_{L3}$, is illustrated in Fig. 2.4(b) lying in a plane positioned 460.7 mm below the zx-plane containing the arc. The amplitude $\phi$ of $\$_L$ is calculated as $\phi = \sqrt{(\Delta\theta_x)^2 + (\Delta\theta_z)^2} = \sqrt{(-0.00436)^2 + (0.00838)^2} = 0.009446\,\text{rad}$, and the pitch $h$ of $\$_L$ is computed as $h = -957.40424\,\text{mm/rad}$ (refer to App. A1). Correspondingly, the axial travel $s$ of the platform carrying the etched profile $P$ is determined as $s = h\phi = -4.14282\,\text{mm}$. The position of $\$_L$ is determined based on the coordinates of the line it lies on, with normalized coordinates of this line calculated as $(L, M, N; P_l, Q_l, R_l) = (-0.46155, 0.0, 0.88711, -408.652, 365.960, -212.616)$ in units of (rad; mm). Using the relations outlined in App. A1, it is observed that $L$ intersects the plane $x_n = 0$ at $(y_n, z_n) = (-460.6564, -792.8935)\,\text{mm}$ and the plane $z_n = 0$ at $(x_n, y_n) = (-412.530, -460.655)\,\text{mm}$ (as shown in Fig. 2.4(b)).

For comparison purposes, a well-known method effective in fitting a circular arc to n points in 2D is described in Kåsa [5]. This method necessitates an initial estimation of an arc radius r and its center location $(x_C, y_C)$, for which we utilized the nominal arc from the CAD model for our FEA sprung-back points. The error measure minimized is a quadratic function involving $r$, $x_C$, and $y_C$. This error function, denoted as $e$, is computed by squaring the sum of the differences of two squares, i.e., $e = \{(x_i - x_C)^2 + (y_i - y_C)^2 - r^2\}^2$ . Optimization was carried out using the Nelder-Mead [23] method via the SciPy library [24], an effective non-linear optimization algorithm especially suitable for problems where derivative information is unavailable. The initial values

for radius and center coordinates were derived from the nominal values in Fig. 2.4, and the algorithm iteratively refined these parameters to minimize the objective function.

Table 2.4: $LS\Sigma s$ and SDTs for Three Iterations of VW&R To Fit an Arc to Nine Points in 3D (Table 2.3 and Fig. 2.4).

| Method | LSΣ, $mm^2$ | SDT ($\Delta\theta_x$, $\Delta\theta_z$; $\Delta x$, $\Delta y$, $\Delta z$; $\Delta r$), (rad; mm) |
|---|---|---|
| VW&R, iter 1 | 0.9517 | $(-0.00415, 0.00833; 0.4350, 3.5490, -10.1371; -8.0011)$ |
| VW&R, iter 2 | 0.8229 | $(-0.00436, 0.00838; 0.3140, 3.4570, -10.0315; -7.9064)$ |
| VW&R, iter 3 | 0.8229 | $(-0.00436, 0.00838; 0.3140, 3.4570, -10.0315; -7.9064)$ |

Since the Kåsa method is 2D and its result for $LS\Sigma$ disregards deviations in the y-direction (as shown in Fig. 2.4), a more appropriate comparison to our VW&R method can be made with values of radius and center coordinates of the fitted arc (as detailed in Table 2.5) instead of $LS\Sigma$.

To demonstrate the time efficiency of the virtual work and robotics method for fitting a circular arc around a large dataset, we conducted two experiments. In both experiments, we generated a substantial number of points along a circular arc with a radius of 250 units and a center at $(x_n, y_n, z_n) = (10, 0, 25)$. Random noise was added to these points using the NumPy library in Python to simulate real-world conditions.

In the first experiment, we generated 400,000 points and implemented a fully vectorized version of the VW&R method using NumPy and the math library. Remarkably, the algorithm converged after only 3 iterations, demonstrating its efficiency. The execution time for this experiment was approximately 0.4826 seconds and was performed on an M1 MacBook Pro equipped with an Apple M1 Pro chip featuring a 10-core CPU.

Additionally, we conducted a second experiment with a more extensive dataset

of 4 million points to further highlight the scalability and efficiency of the method. Surprisingly, despite the larger dataset, the computation time was proportionately even less. On the M1 MacBook Pro, the runtime for convergence was an impressive 4.3838 seconds.

Table 2.5: Radius and Two Arc Center Coordinates from Three Iterations of VW&R (Table 3 and Fig. 5) and from Kåsa 2D Method.

| Method | Radius $r$ | $x_C$ | $z_C$ |
|---|---|---|---|
| VW&R, iter 1 | 241.9988 | 0.4351 | -10.1371 |
| VW&R, iter 2 | 242.0999 | 0.3140 | -10.0314 |
| Kåsa | 242.09 | 0.30 | -10.04 |

## 2.3 Conclusion

We have explored the application of the VW&R method in fitting both 2D and 3D arrays of points to straight lines, as well as fitting circular arcs to 3D arrays of points. In the case of straight lines, two iterations have proven adequate to achieve alignment of the $LS\Sigma$ with a widely accepted fitting routine, reaching agreement up to four significant figures. Similarly, for fitting circular arcs, two iterations have sufficed to attain convergence of the $LS\Sigma$ up to four significant figures. However, when considering fitting arcs, it is more meaningful to compare with a well-established 2D fitting routine using the fitted radius as the guiding parameter (refer to Table 2.5) rather than the $LS\Sigma$ This is because a significant portion of the 3D deviation is lost when projected onto a 2D plane. Here as well, two iterations have yielded agreement up to four significant figures.

The aim of this chapter is not to advocate for the iterative VW&R method as a

replacement for traditional fitting methods for straight lines and circular arcs. Instead, by demonstrating that it produces identical results when fitting points arranged around these two features, we propose that iterative VW&R presents a viable approach for computing the Small Displacement Torsor (SDT) of any aggregate line-profile segmented into sequentially linked segments of straight lines, circular arcs, and potentially other analytically representable profile segments. A method for segmenting data into straight and circular arc-segments is elucidated in [25].

Chapter 3

# GEOMETRIC PARAMETERS FROM STAMPED COMPONENTS AND ASSEMBLIES MADE FROM THEM

The goal in this chapter is to describe algorithms that produce a small number of quality-related geometric parameters that characterize geometric deviations in components after the stamping process and in the spot welded assemblies made from them. Each dataset contains coordinates of simulated sprung-back nodal points along edge profiles, such as those shown in Figure 3.1 for an assembly in which a flat sheet is spot welded to the flanges of a straight hat-section having a channel width measuring 50mm. The components are constructed from DP 590 steel, characterized by nominal material qualities enhanced through MISO scale factors of 0.8, 1.0, and 1.2. The steel's thickness varies across five specific measurements: 0.8 mm, 1.0 mm, 1.2 mm, 1.4 mm, and 1.6 mm. Additionally, four draw depths (35 mm, 45 mm, 55 mm, and 70 mm) and three blank holding forces (15,500 N, 25,700 N, and 35,500 N) were used in creating the dataset. Taking all combinations of these inputs, a total of 180 distinct components were simulated. FEA simulations for two groups of straight assemblies were also generated from these components by our partners at Ohio State University. Some of these assemblies were formed by spot-welding a flat sheet to the flanges of a hat component; others were formed by spot-welding the flanges of one hat-section to those of another mounted over it. Altogether, 1,000 simulated assemblies were simulated by the OSU team in a manner to uniformly, although sparsely, fill the design space of input variables, and the ASU team generated the corresponding geometric parameters for quality. This dataset was produced as a part of a broader NSF collaborative project between the two universities [1], the goal of which is to

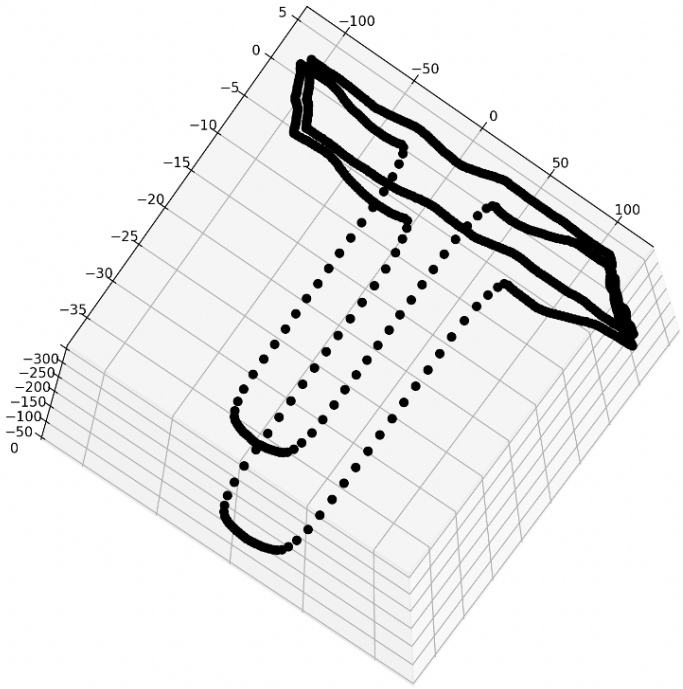



Figure 3.1: Nodal Point Data for an Assembly of a Flat Sheet Spot-Welded to a Stamped Hat-Section.

build a database of both (a) FEA simulations of component forming and spot-welded assemblies made from the, and (b) corresponding geometric parameters for both the components and assemblies in their free state [26]. Because the simulations are impractically time-consuming to use for every possible combination of input variables, Artificial Neural Networks (ANNs) may be used to forecast the geometric parameters to a much finer resolution of inputs than that used in generating the dataset. This approach aims to provide a predictive insight into manufacturing outcomes using the efficiency of ANNs

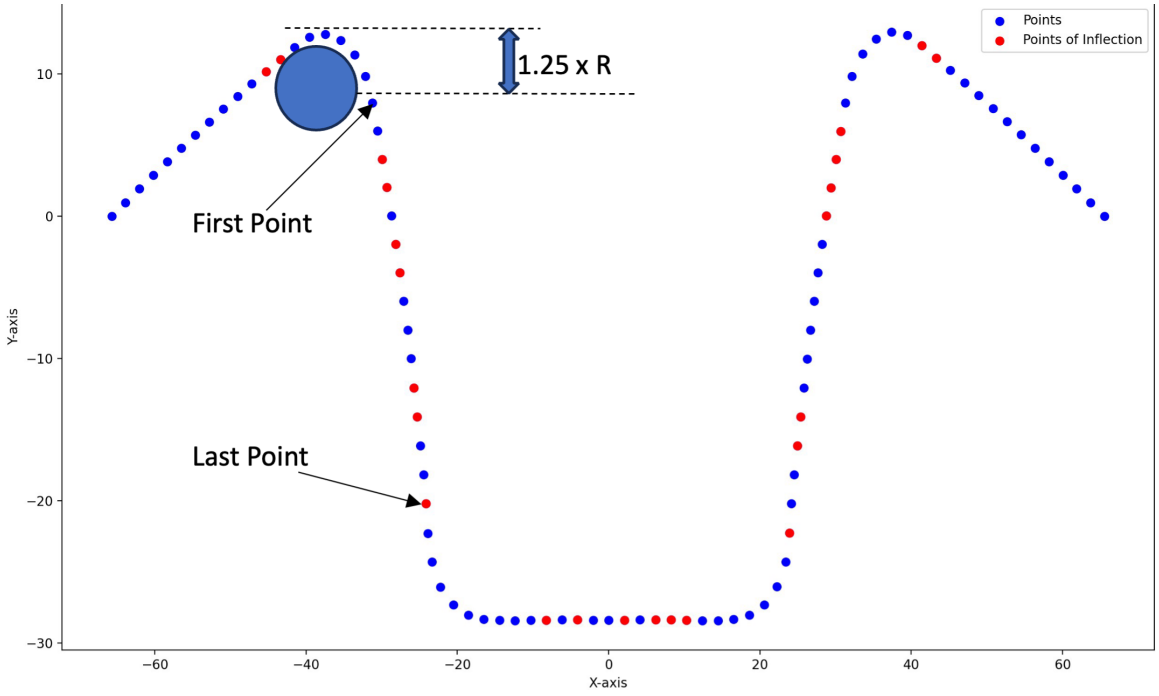



Figure 3.2: Range of Points used for Radius of Curvature Calculation on a Component. Last Point is the First Inflection Point Observed when Moving Upwards From the Bottom.

## 3.1 Radius of Curvature

In the case of the hat-shaped components, a noticeable curvature emerged in the vertical sections due to springback in the stamped parts, as depicted in figure 4.2. This curvature transitioned at a point of inflection located at the base of the vertical section, where the direction of curvature shifts. To accurately determine the curvature of the vertical section of the hat, measurements were taken from points below 1.25 times the radius of the die used, extending to the observed point of inflection at the bottom. To determine the point of inflection within a given set of points, which are organized in ascending order of their $x$-coordinates, the process begins with the calculation of the first derivative. This is achieved by determining the slope between

two consecutive points. Subsequently, the second derivative is computed by employing a formula that involves three consecutive points [27].

$$\frac{2\left[(y_3 - y_2)(x_2 - x_1) - (y_2 - y_1)(x_3 - x_2)\right]}{(x_3 - x_2)(x_2 - x_1)(x_3 - x_1)}$$

Now, to calculate the points of inflection we essentially look for where the second derivative of a set of points either equals zero or changes sign.

To calculate the radius of curvature accurately, we fit a circle to the points in question using a least squares fitting approach.

1. Establish Circle Equation: Start with the general equation of a circle $(x_i - x_c)^2 + (y_i - y_c)^2 = r^2$ where $(x_i, y_i)$ are the coordinates of points on the perimeter of the circle, $(x_c, y_c)$ represents the circle's center, and $r$ is the radius.

2. Rearrange the Equation: Expand and rearrange the circle's equation to get $ax_i + by_i + c = x_i^2 + y_i^2$ where $a = 2x_c$, $b = 2y_c$, and $c = r^2 - x_c^2 - y_c^2$.

3. Formulate a Linear System: For all the points under consideration, set up a system of linear equations based on the rearranged equation of the circle.

4. Matrix Representation: Convert the system of equations into matrix form, $A \cdot X = B$ , where:

   - $A$ is a $n \times 3$ matrix with known $x_i$ and $y_i$ values, plus a column of ones.
   - $X$ is a column vector of the unknowns $a$, $b$, and $c$.
   - $B$ is a column vector containing $x_i^2 + y_i^2$ for each point.

5. Calculate the Pseudoinverse: Find the pseudoinverse of matrix $A$, represented as $A^+$, using the formula $A^+ = (A^T \cdot A)^{-1} \cdot A^T$ where $A^T$ is the transpose of $A$.

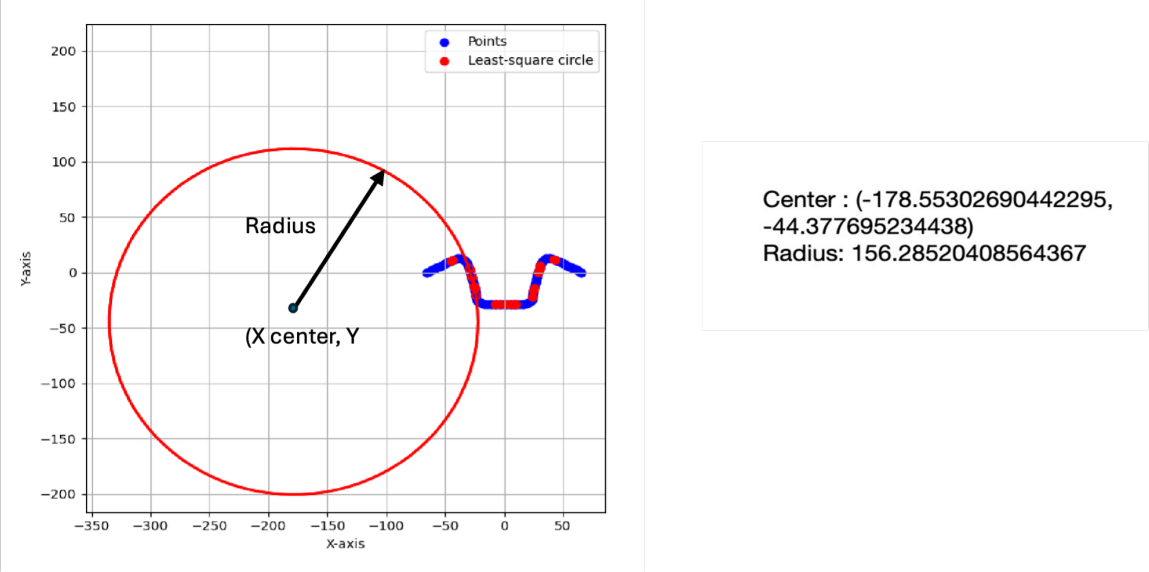



Figure 3.3: Arc Fitted to the Left Vertical Part of One Stamped Hat Section. Values in mm.

6. Optimal Solution: Solve for the vector $X$ by multiplying the pseudoinverse $A^+$ by matrix $B$, thus $X = A^+ \cdot B$.

7. Determine Circle's Center and Radius: With the optimal solution for $a$, $b$, and $c$, calculate the circle's center $(x_c, y_c)$ with $x_c = \frac{a}{2}$ and $y_c = \frac{b}{2}$, and the radius $r = \frac{\sqrt{4c+a^2+b^2}}{2}$.

Following these steps allows for a precise determination of the circle's center and radius of curvature for a least-squares fit of the set of points. The results for one arc-fit on one component are shown in Figs. 3.3 and 3.4. A compilation of results for all 180 hat sections appears in Table 3.1 and Fig. 3.5. .

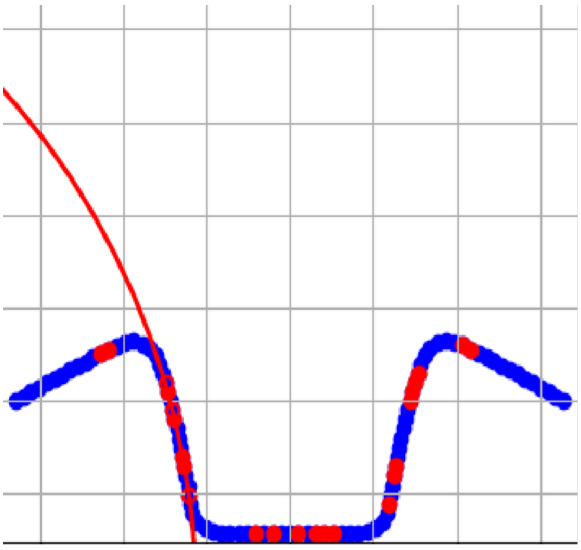


Figure 3.4: Enlarged view of the arc fit.

Table 3.1: Location of Center of Curvature and Radius for Stamped Hat Sections.

| **Filename** | **X center** | **Y center** | **Radius** |
|---|---|---|---|
| S_50_80DP590_140_45_35500.xlsx | -385.99182 | 3.74326471 | 413.426631 |
| S_50_120DP590_120_70_25700.xlsx | 1338.958459 | -97.349602 | 1313.833205 |
| S_50_100DP590_100_35_25700.xlsx | -278.394458 | 6.207085055 | 305.369882 |
| S_50_120DP590_100_55_15500.xlsx | -1222.856676 | 92.91268779 | 1255.428577 |
| S_50_100DP590_120_45_15500.xlsx | -567.466386 | 35.68184411 | 596.925917 |
| S_50_80DP590_80_45_25700.xlsx | 234.266194 | -36.807231 | 208.425081 |
| S_50_80DP590_160_70_25700.xlsx | 1298.208376 | -121.947619 | 1273.845979 |
| S_50_120DP590_100_45_35500.xlsx | 690.027785 | -62.420257 | 664.946553 |
| S_50_100DP590_120_55_35500.xlsx | 328.038069 | -40.059677 | 301.555054 |
| S_50_80DP590_140_55_15500.xlsx | -418.452680 | 5.437130135 | 446.763913 |
| S_50_100DP590_160_70_15500.xlsx | 464.682451 | -51.981358 | 437.890712 |
| S_50_100DP590_80_45_15500.xlsx | -366.125379 | 10.90092111 | 393.728166 |

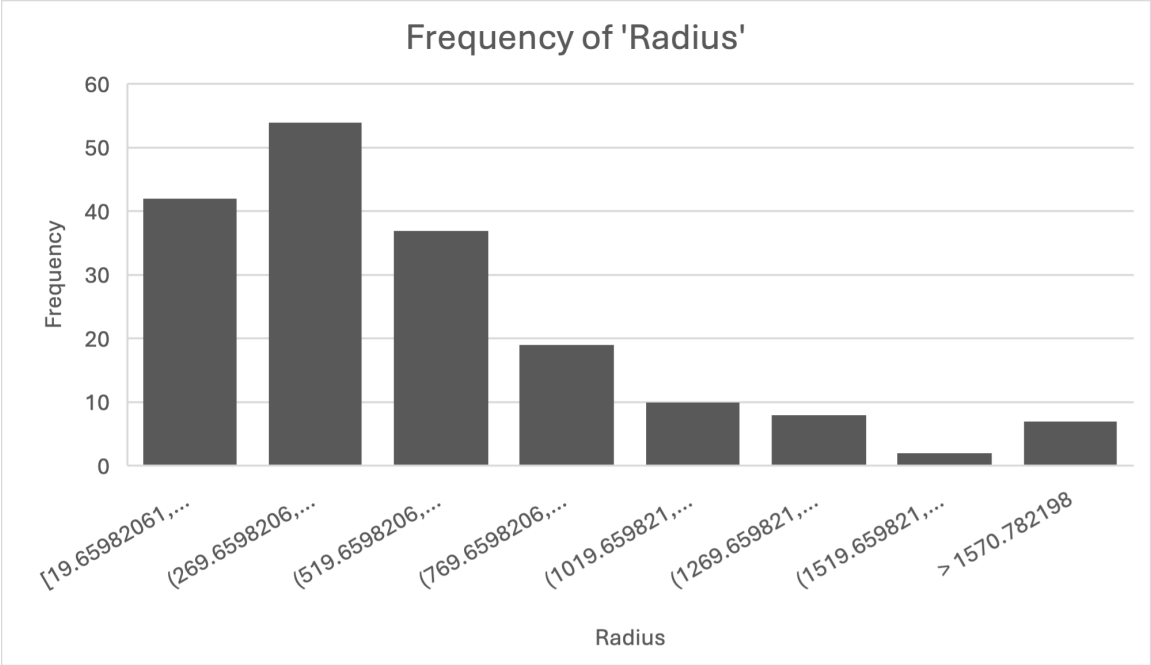



Figure 3.5: Frequency of Radius of Curvature for Stamped Hat Sections.

## 3.2 Zone Magnitude

All of the sprung-back coordinate data provided by the Ohio State team were for points along the edges of components and assemblies. One parameter used to judge the quality of assemblies was the zone-magnitude for points along both components that had been spot-welded (simulated) at the flanges. As an example, data points for the assembly in Fig. 3.6 were selected from the entire periphery of the sheet (marked red in Fig. 3.6) and from all flange-points of the hat section (marked green). Specifically, for the hat sections, points were included from both the left and right flange edges. Additionally, for the central part of the hat sections, points were incorporated extending from the edge up to the commencement of the die curvature, as illustrated in the Fig 3.6.

Since the nodes used in all components of the FE simulations are located midway between the two surface boundaries of the sheet material, the half-thickness from

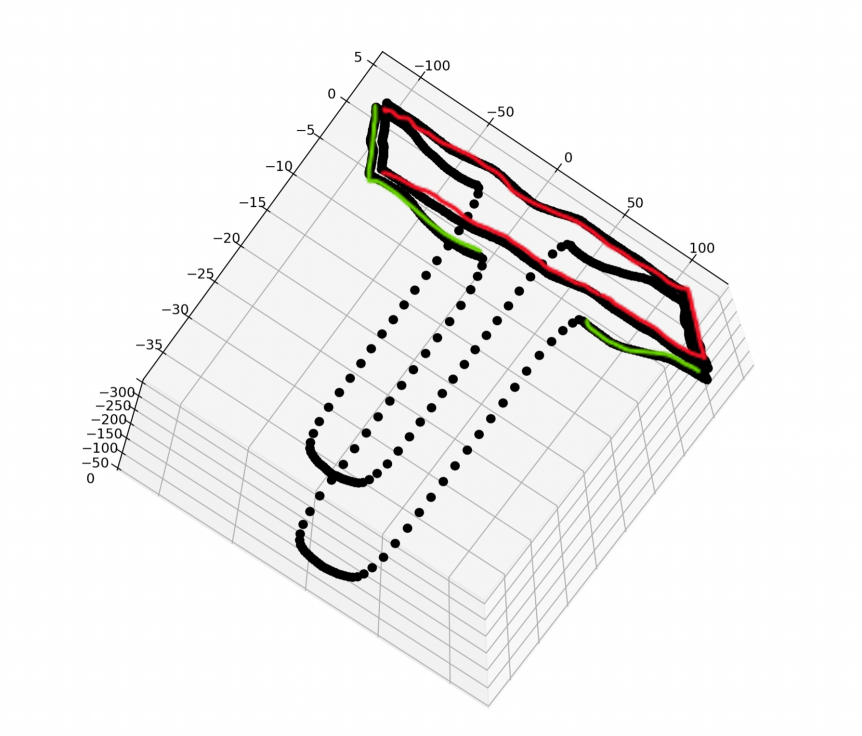


Figure 3.6: Data Points: Entire Periphery of the Sheet (marked Red) and the Apex Points of the Hat Sections (Marked Green).

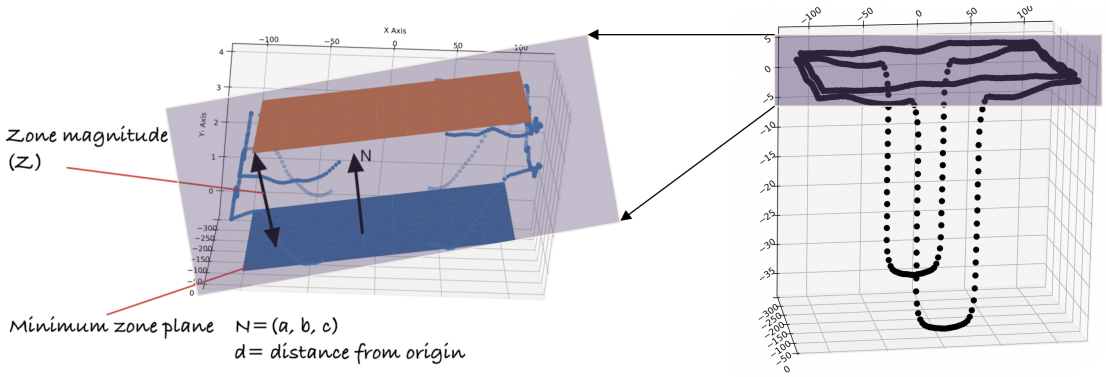



Figure 3.7: Zone-Magnitude for Points along both Components that had been Spot-Welded (Simulated) at the Flanges.

each of the two joined components is included in the corresponding zone magnitudes. The steps in computing the unconstrained minimum zones are discussed in details in section 1.1.4:

## 3.3 Multi-Layer Perceptron Neural Networks (MLPNN) for Stamped Components and Hat Subassemblies.

The objective is to develop a multi-layer perceptron network designed to learn from the features (inputs) of simulated assemblies. Subsequently, by entering other input values for these same assemblies, the network aims to accurately predict both the magnitude of the minimum zone and the location of its mid-plane. This predictive model, then, allows a user to estimate the outcomes without the need for additional simulations, thereby streamlining the design process and saving valuable resources.

For the training of the multi-layer perceptron neural network, 14 input features (manufacturing variations) of spot-welded assemblies of hat sections with plane sheets or another hat section were utilized. The features are:

1. Top Component Shape: The shape category of the top component. All the hats were of straight hat section.

2. Top Component Channel Width: The channel width of the top component measured in millimeters (50 mm).

3. Top Component Material: The material specification for the top component, indicating both the MISO scale factor (0.8 and 1.2) and the nominal material type.

4. Top Component Thickness: The material thickness (80, 100, 120, 140, 160, 180) of the top component in millimeters.

5. Top Component Draw Depth: The vertical draw depth of the top component, measured in millimeters ( 70, 55, 45, 35).

6. Top Component Blank Holding Force: The blank holding force exerted on the top component during its formation, measured in Newtons (25700N, 35500N, 15500N).

7. Bottom Component Shape: The shape classification of the bottom component. All the hats were of straight hat section.

8. Bottom Component Channel Width: The channel width of the bottom component in millimeters (50 mm).

9. Bottom Component Material: The material used for the bottom component, including the MISO scale factor (0.8 and 1.2) and the material type.

10. Bottom Component Thickness: The thickness of the bottom component's material (80, 100, 120, 140, 160, 180).

11. Bottom Component Draw Depth: The draw depth of the bottom component in millimeters ( 70, 55, 45, 35).

12. Bottom Component Blank Holding Force: The force applied to the blank of the bottom component, measured in Newtons (25700N, 35500N, 15500N).

13. Weld Count: The number of spot welds applied to the flanges (3 welds).

14. Weld Pattern: The configuration in which the welds are distributed (Linear).

*Learning Rate and Architecture Details*

- Initial Learning Rate: 0.01

- Learning Rate Schedule: Adaptive (decreases by 10% every 5 epochs without improvement)
- Optimizer: Adam (Adaptive Moment Estimation)
- Input Layer: 14 neurons (matching the number of input features)
- Hidden Layers: Configurations with 4, 7, 10, and 13 layers, using ReLU activation
- Output Layer: Single neuron with linear activation for regression (predicting zone size)

*Regularization, Dropout, Batch, and Epochs*

- Dropout Rate: 0.5 for layers with >5 neurons to prevent overfitting
- L2 Regularization: Lambda value of 0.001 on weights
- Batch Size: 32
- Number of Epochs: 100 with early stopping if no improvement for 10 epochs

*ML Performance Metrics and Weight Initialization*

- Loss Function: Mean Squared Error (MSE).
- Evaluation Metric for ML: Accuracy within 5% of error
- Method: He Initialization (for ReLU activations)

MLP Structures and Performance

The dataset was divided into two sets: 70% for training and 30% for testing. The MLP configurations and their performance are detailed below:

**MLP Configuration 1:** 4 hidden layers, each with 3 neurons, Accuracy: 57.07%

**MLP Configuration 2:** 7 hidden layers, each with 5 neurons, Accuracy: 78.41%

**MLP Configuration 3:** 10 hidden layers, each with 5 neurons, Accuracy: 95.93%

**MLP Configuration 4:** 13 hidden layers, each with 5 neurons, Accuracy: 99.91%

**MLP Configuration 5:** 13 hidden layers, each with 7 neurons, Accuracy: 99.67%

**MLP Configuration 6:** 13 hidden layers, each with 9 neurons, Accuracy: 99.58%

Through iterative optimization, the neural network's structure was fine-tuned, resulting in a marked increase in predictive accuracy. The highest accuracy was achieved with a network comprising 13 hidden layers and 5 neurons in each layer. Subsequent increases in the number of neurons within this configuration showed a marginal increase in performance, suggesting a potential plateau in the effectiveness of adding more neurons.

*Best Performing Model Configuration*

- Number of Hidden Layers: 13
- Neurons in Each Hidden Layer: 5
- Achieved Accuracy: 99.91%

The MLP's performance was optimized through iterative architectural adjustments, balanced with regularization and learning rate adaptations to prevent overfitting and ensure effective training convergence.

Chapter 4

# GEOMETRIC PARAMETERS FROM T JOINT ASSEMBLY

The second dataset is for the T joint assembly in Fig 4.1, Each T-joint consists of two hat-and-flat-sheet subassemblies joined together into the T. The hat-and-flat-sheet subassemblies are similar to the assemblies described in chapter 3, but here additional variations to both material (Table 4.1) and geometry (Table 4.2) are considered in the workflow for generating additional variety in dataset. The geometric variations include changes in the section depth and width, sheet thickness, as well as changes to the tooling geometry. The material variations include modifications to the nominal materials (i.e., DP590, Aluminum Alloy) which include variations to the elastic modulus and stress strain curves. In the case of DP590 the stress strain values are provided using a table of values for plastic strain region (multilinear isotropic hardening). For the aluminum alloy, a tangent modulus for the bilinear isotropic hardening model is used. Table 4.1 shows the material properties of the two materials considered.

Table 4.1: Material Properties for Simulated DP590 Steel and Al Alloy.

| **Material** | **Density [Kg/mm$^3$]** | **E [GPa]** | **Poisson's Ratio** | **Yield Strength [MPa]** |
|---|---|---|---|---|
| DP590 | $7.85 \times 10^{-6}$ | 191 | 0.3 | 411 |
| Al Alloy | $2.70 \times 10^{-6}$ | 71 | 0.3 | 135 |

To quantify the geometric variations of these assemblies, 17 parameters were extracted. These parameters are:

- 6 Minimum zone magnitudes
- 8 Twist Angles.

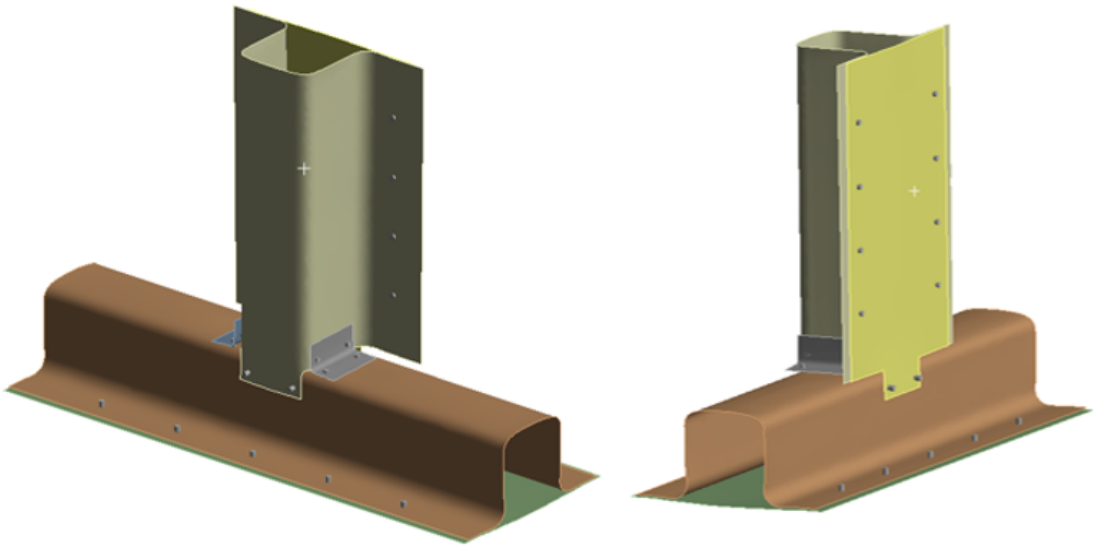


Figure 4.1: T Joint Configuration, which Consists of Two Hat Sections Coupled with Two Flat Sheets

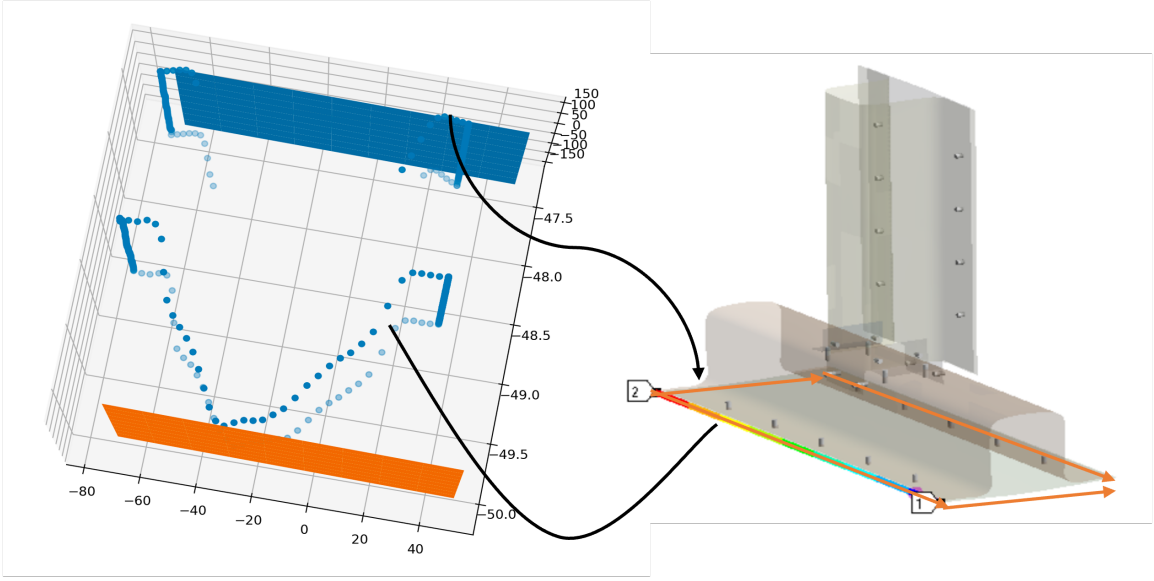
Table 4.2: Material Properties for Simulated DP590 Steel and Al Alloy.

| **Component** | **Material** | **Thickness [mm]** | **Width [mm]** | **Depth [mm]** |
|---|---|---|---|---|
| Horizontal | DP590 | 1.5 | 45 | 50 |
| Horizontal | Al | 2.0 | 60 | 60 |
| Vertical | DP590 | 2.0 | 50 | 48.0 |
| Vertical | Al | 1.5 | 55 | 63.5 |

- 3 Angles between Fitted planes

## 4.1 Zone Magnitudes

The method to calculate zone magnitude has been discussed in details in section 1.1.4 and 3.2 . The sample of the zone magnitudes which have been obtained for Honda T Joint are in shown in the Table 4.3 and in the figure 4.2.




Plane Normal vector: [-2.62807438e-04 -9.99999953e-01 -1.55799681e-04]
Plane Distance from origin: 47.38717742977702

Figure 4.2: Honda T joint in the Right. In the Left, Points from the Bottom Part of the T-Joint. The Points Near the Blue Plane are from the Hat and the Points Near the Orange Plane are from the Plane Sheet.

Table 4.3: Minimum Zone Magnitudes for Assemblies and Components.

| **Component/Assembly** | **Zone Magnitudes** |
|---|---|
| Bottom sheet and horizontal hat assembly | 2.6297607298821717 |
| Top sheet and vertical Hat Assembly | 2.6297607298821717 |
| Horizontal Sheet | 1.4294150077366226 |
| Vertical Sheet | 1.4383860156661976 |
| Base of Horizontal Hat | 0.4647751897650423 |
| Base of Vertical Hat | 0.20937025169485002 |

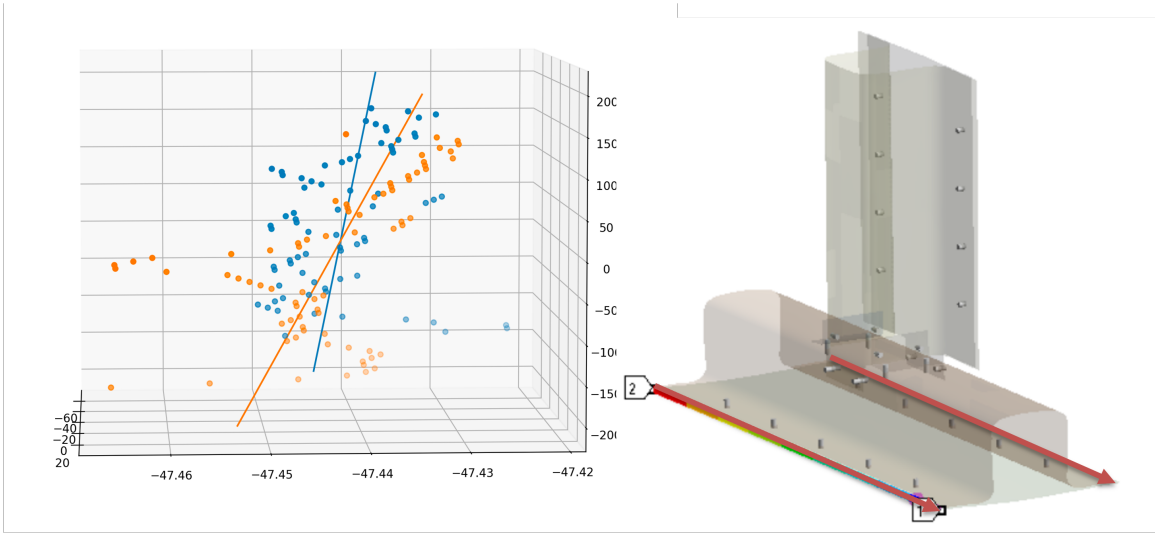



Figure 4.3: Opposite Edges Shown in Red (Right), Angle Between Fitted Lines to the Points Along the Edges (Left).

## 4.2 Calculation of Twist Angles

A second geometric parameter is the twist angle between opposite edges of any component or assembly as shown in figure 4.3. To calculate the twist angle between two parallel edges of any component or assembly, the least square fit line is first calculated for each edge using PCA, discussed in details in section 1.1.3. To calculate the twist angle between the two lines, each line is projected onto two global coordinate frame planes. If the edges were almost parallel to the z-axis, the lines were projected onto the XZ and YZ planes as shown in Fig. 4.4 to calculate the twist angles. For the XZ projection, the x and z components of the direction vector of each line are extracted. For the YZ projection, the y and z components are extracted. Finally, the twist angle between the two lines is calculated using the arcsine of the cross product of the two direction vectors. The angle is then converted from radians to degrees.

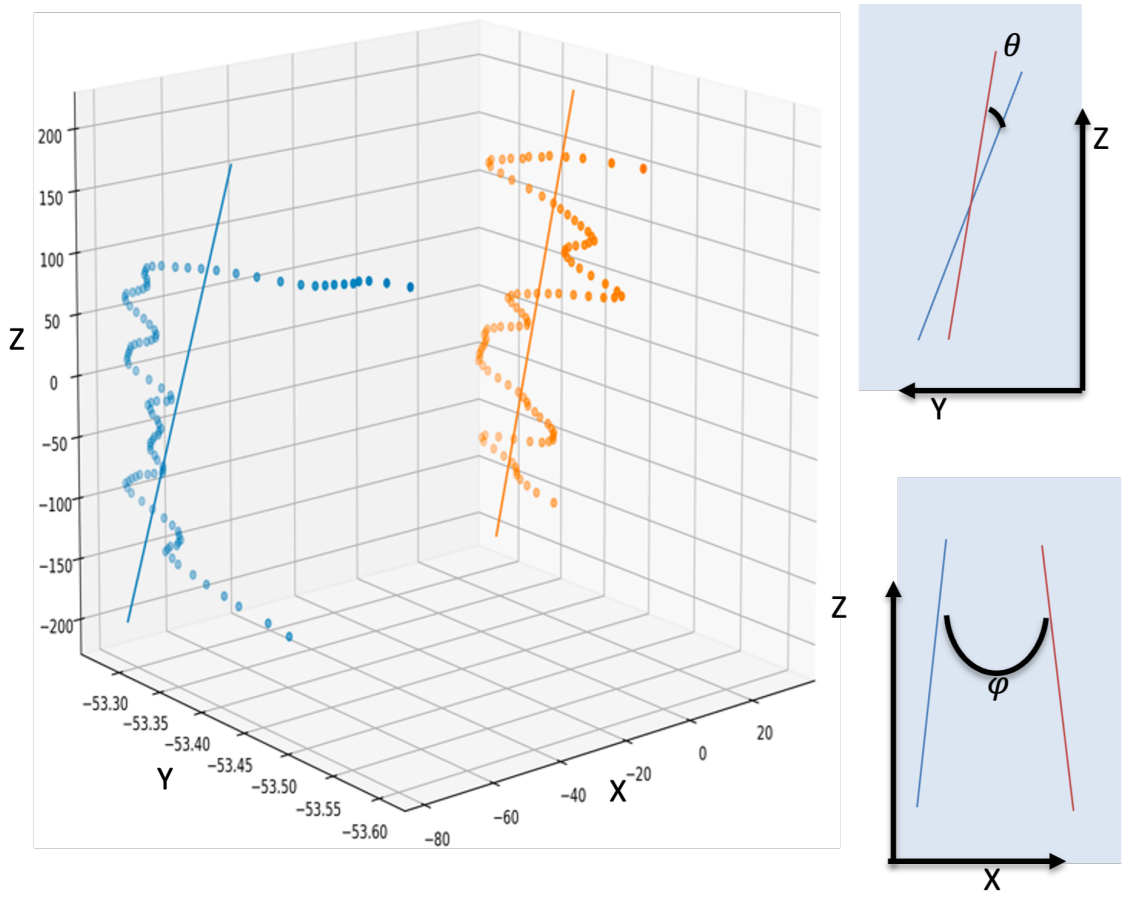



Figure 4.4: (a) Data from Two Opposite Edges of a Component or Assembly. (b) The Two Projected Angles of Twist.

## 4.3 Angles Between Fitted Planes

To estimate the perpendicularity of a T-joint, two planes are fitted to the base of the horizontal and vertical components of the joint, shown in Fig. 4.5. This process involves collecting two sets of data from the horizontal and vertical subassemblies of the T-joint. The best-fit planes for each subassembly are found using the pseudoinverse method, discussed in section 1.1.3, and are represented by the planes PQRS and ABCD, respectively in Fig 4.5. The normal vectors for both planes are calculated

and named $N_1$ and $N_2$, respectively.

To determine the relative rotation of the plane ABCD with respect to the plane PQRS in the Z, X, and Y directions, three sets of angles are defined: $\alpha$, $\beta$, and $\gamma$. A coordinate frame is fixed to the PQRS plane so that PQRS is parallel to the XZ plane. To calculate the angle $\alpha$, $N_1$ is projected onto the XY plane, and the angle between the projected normal and the X axis is calculated. The angle $\alpha$ is then determined as 90° minus this angle. To calculate the angle $\beta$, a line is constructed through the midpoints of AB and CD and is projected onto the YZ plane. The angle $\beta$ is determined as the angle between the projected line and the Y axis. Finally, to calculate angle $\gamma$, $N_1$ is projected onto the XZ plane, and the angle $\gamma$ is determined as the angle between the negative X axis and the projected line. Sample values of $\alpha$, $\beta$, and $\gamma$ for one assembly are 87.97°, 0.041° and 0.097° respectively (Fig. 4.5).

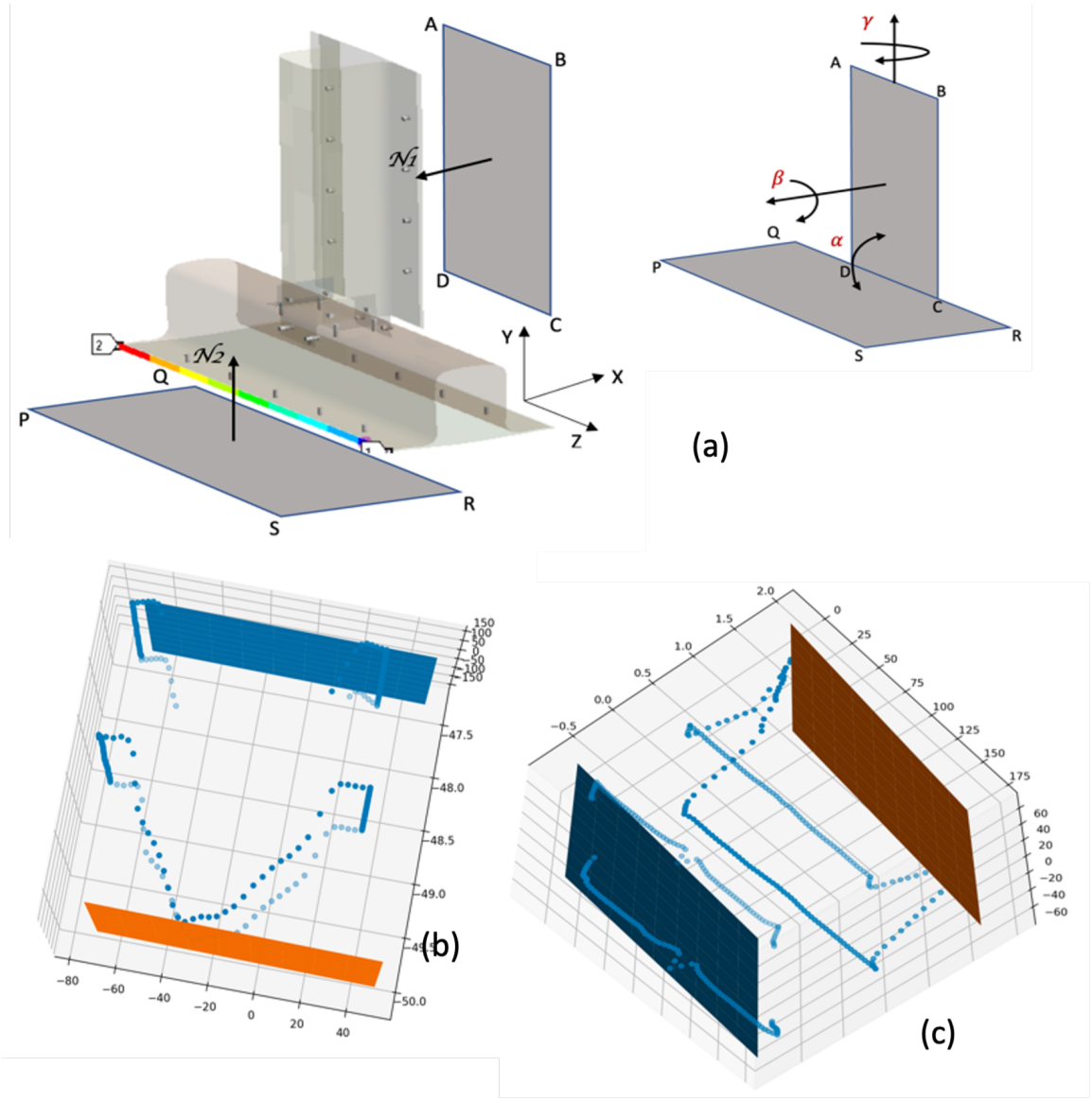



Figure 4.5: (a.) The Exploded T-Joint Assembly Showing Locations of Points for the Horizontal Planar Zone Fit and Points for the Vertical Planar Zone Fit. (b.) Points from the Flat and Hat-Section Components and the Resulting Parallel Boundaries for the Horizontal Planar Zone Fit. (c.) Points from the Flat and Hat-Section Components and the Resulting Parallel Boundaries for the Vertical Planar Zone Fit.

Chapter 5

# DEEP LEARNING MODEL FOR PREDICTING MAXIMUM STRESS FROM MULTI-VIEW IMAGES OF HONDA HOOD FRAMES

The main aim of this chapter is to create a predictive model capable of accurately estimating the Von Mises stress of Honda car hood frames based on their multi-view images. The model development utilized data from the carhood10k dataset [28], which comprises 3D FEA simulations of 10,000 hoods in STL (Standard Triangle Language) format, featuring 100 skins with 100 designs each, shown in Fig. 5.1. In addition to STL files, the dataset includes Excel files detailing geometric and performance attributes of the hoods, though only the maximum stress was used as a measure of performance in this study.

The core concept, as outlined in the chapter 1, involves training a deep learning model to extract features from Honda hood images and then predict the hood's maximum stress. Furthermore, this chapter includes a comparative analysis of three neural network models: a custom-designed convolutional neural network [14], VGG-16 with transfer learning [29], and ResNet-50 with transfer learning [30]. The comparison focuses on the models' ability to predict maximum stress within accuracy thresholds of 5%, 10%, 15%, and 20%. Along with these plots of Residual analysis, Learning curves gives a comprehensive understanding of the performance of the models.

## 5.1 Data Preparation & Filtering

Three perspectives of the 3D STL files, namely the top, front, and side views, shown in Fig. 5.2, were captured using AutoCAD. During the data filtering process, consistency between the image and Excel data was ensured. Any discrepancies, such

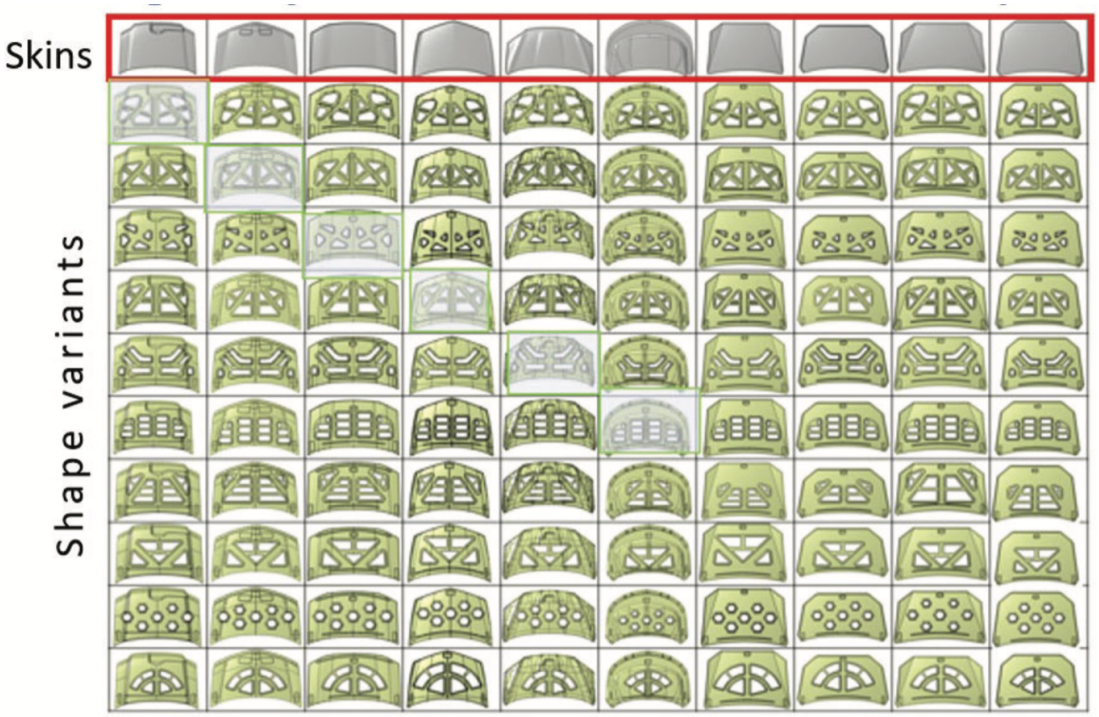



Figure 5.1: All Shape Variants and 10 Sample Skins from the Hood Frame Dataset.

as missing image data for an Excel file, resulted in the deletion of both the Excel file and its corresponding image data to maintain dataset integrity. A structured directory was created where each 'skin' type from the CSV files had its designated folder in the "Final Data" directory.

## 5.2 Predictive Model using Custom-CNN Architecture

The Custom-CNN architecture is tailored for the multi-view image analysis of hood frames, and its design (Fig. 5.3). Here's a detailed explanation of its architecture:

- Three Separate Channels: The architecture has three distinct channels, each dedicated to processing images from one of the three views of the hood frame: top, front, and side (Fig. 5.2). This allows the network to learn view-specific

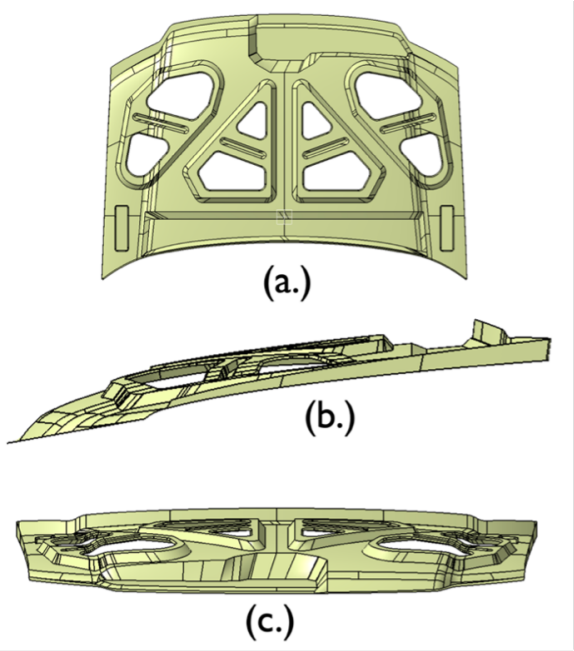



Figure 5.2: (a.) Top View, (b.) Side View, (c.) Front View.

features.

- Convolutional Layers: Each channel contains three convolutional layers [14]. Each layer uses 64 filters with 3×3 kernel sizes, which are small enough to capture detailed features while allowing for a deeper network that can learn complex patterns.

- ReLU Activation Function: The Rectified Linear Unit (ReLU) activation function is used in these layers to introduce non-linearity, helping the network learn complex patterns in the data. ReLU was chosen for its computational efficiency and its ability to mitigate the vanishing gradient problem.

- Max-Pooling Layers: Interspersed with the convolutional layers, the Max-Pooling layers (1 to 5) use a 2×2 pooling size. These layers reduce the spatial dimensions of the feature maps, decreasing the amount of computation needed for

subsequent layers and helping to prevent overfitting by providing an abstracted form of the features.

- Global Average Pooling (GAP): Following the convolutional and Max-Pooling layers, a GAP layer in each channel helps to reduce the feature dimensions further. It calculates the average output of each feature channel, condensing the data into a single value per feature map. This reduces the total number of parameters and helps in combating overfitting.

- Fully Connected Layers: The concatenated outputs, i.e., after stacking the features from the GAP layers end-to-end of all three channels are passed through three fully connected (dense) layers with 768, 256, and 128 units, respectively. These layers serve to integrate the features learned from all views of the hood, creating a comprehensive representation.

- Dropout Regularization: A dropout rate of 0.5 is applied after every fully connected layer, randomly setting half of the units to zero during training. This regularization technique helps to prevent overfitting by ensuring that the model does not become overly reliant on any one node.

- Single Unit Output Layer: The final layer is a single-unit output layer for regression, designed to predict the maximum stress of the hood frame. This layer consolidates the learned features into a single predictive value, representing the estimated maximum stress.

### *5.2.1 Results*

Accuracy within ±5%: 24.26%

Accuracy within ±10%: 35.38%

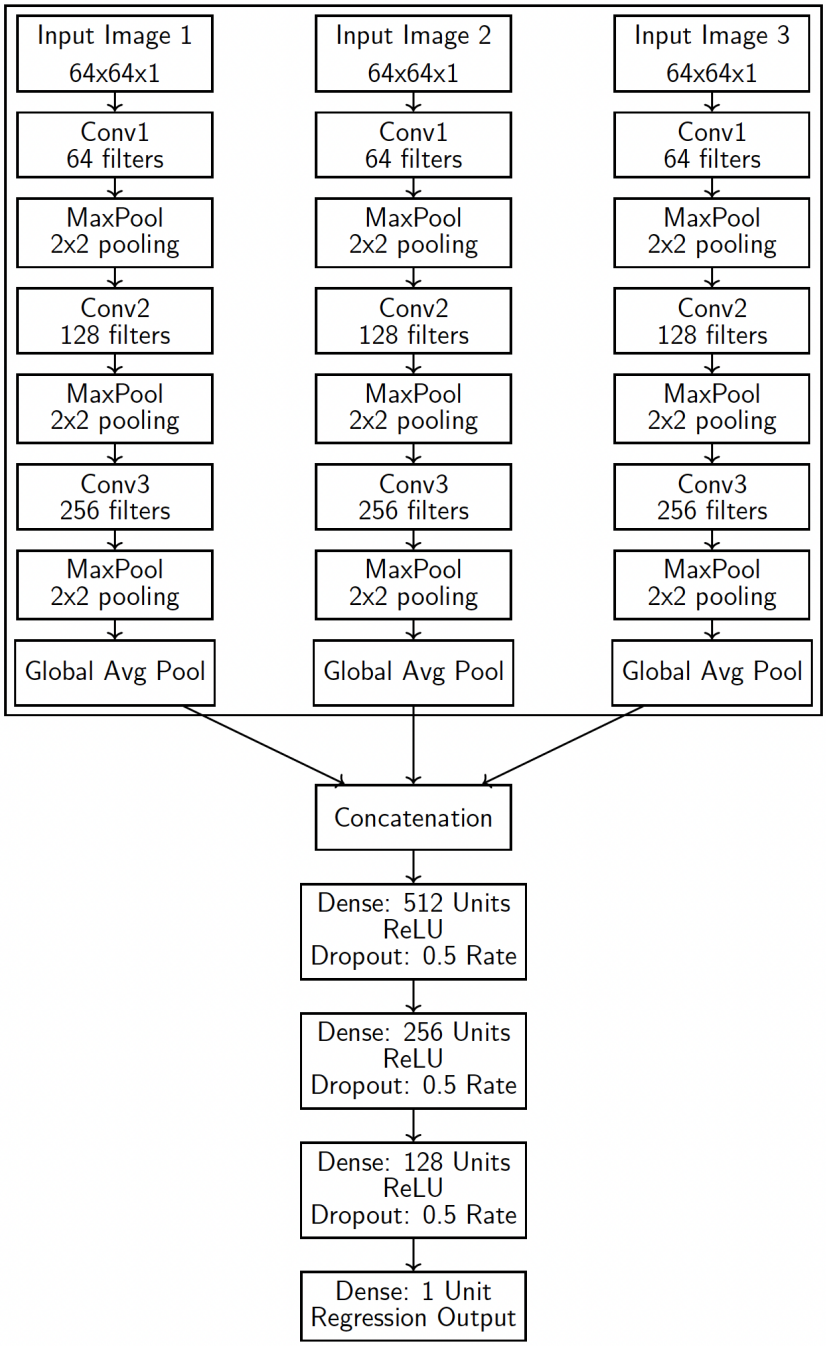



Figure 5.3: The Custom-CNN Architecture, Tailored for the Multi-View Image Analysis of the Honda Hood Frames.

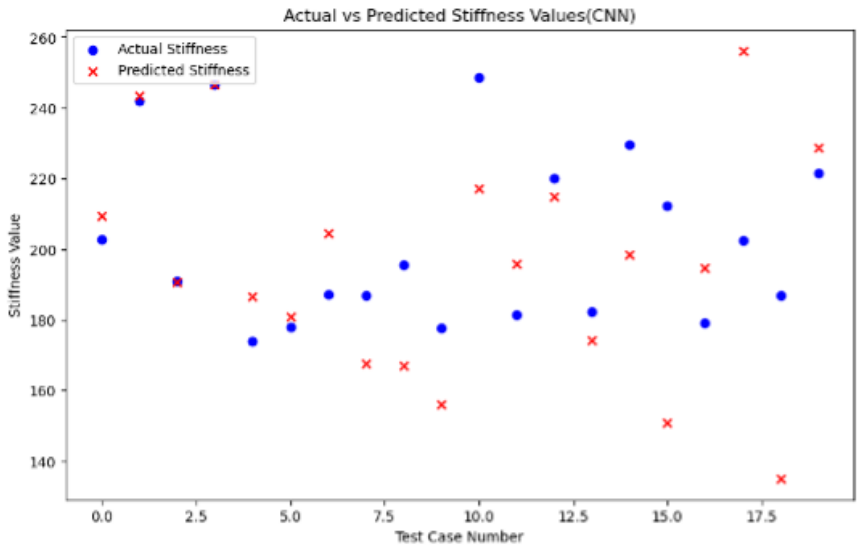



Figure 5.4: Actual maximum stress (Blue) vs Predicted maximum stress(Red) for the Custom-CNN Architecture.

Accuracy within ±15%: 55.79%

Accuracy within ±20%: 77.22%

The Custom CNN network's predictive accuracy shows a sliding scale of correctness with increasing tolerances. Within a narrow margin of 5%, the network is correct about a quarter of the time. As the margin widens to 10% and then 15%, the accuracy improves to 35.38% and 55.79%, respectively. Finally, with a broad margin of 20%, the network's predictions are 77.22% accurate, indicating that while the network captures the general trend, its precision needs enhancement for tasks requiring high accuracy.

## 5.3 Predictive Model using VGG-16 Architecture: Transfer Learning

VGG-16 is a convolutional neural network architecture that was designed by the Visual Graphics Group at Oxford, from which it gets its name. The "16" refers to the fact that it has 16 layers that have weights. The core characteristics of VGG16


Figure 5.5: Training and Validation Loss for the Custom-CNN Architecture.

has been discussed in [29]. The pre-trained VGG16 model has been trained on the ImageNet dataset [31], a large database of images across 1000 different categories. Here I have used transfer Learning using VGG-16 [29] with deep neural network for predicting maximum stress of Honda hood frames (Fig. 5.6).

- Convolutional Layers: VGG16 consists of 13 convolutional layers. It uses 3x3 convolutional filters exclusively with a stride of 1 pixel, which is a smaller filter than was popularly used before VGG16 demonstrated its effectiveness.

- Max Pooling Layers: The convolutional layers are followed by max pooling layers with a 2x2 pixel window, with a stride of 2 pixels.

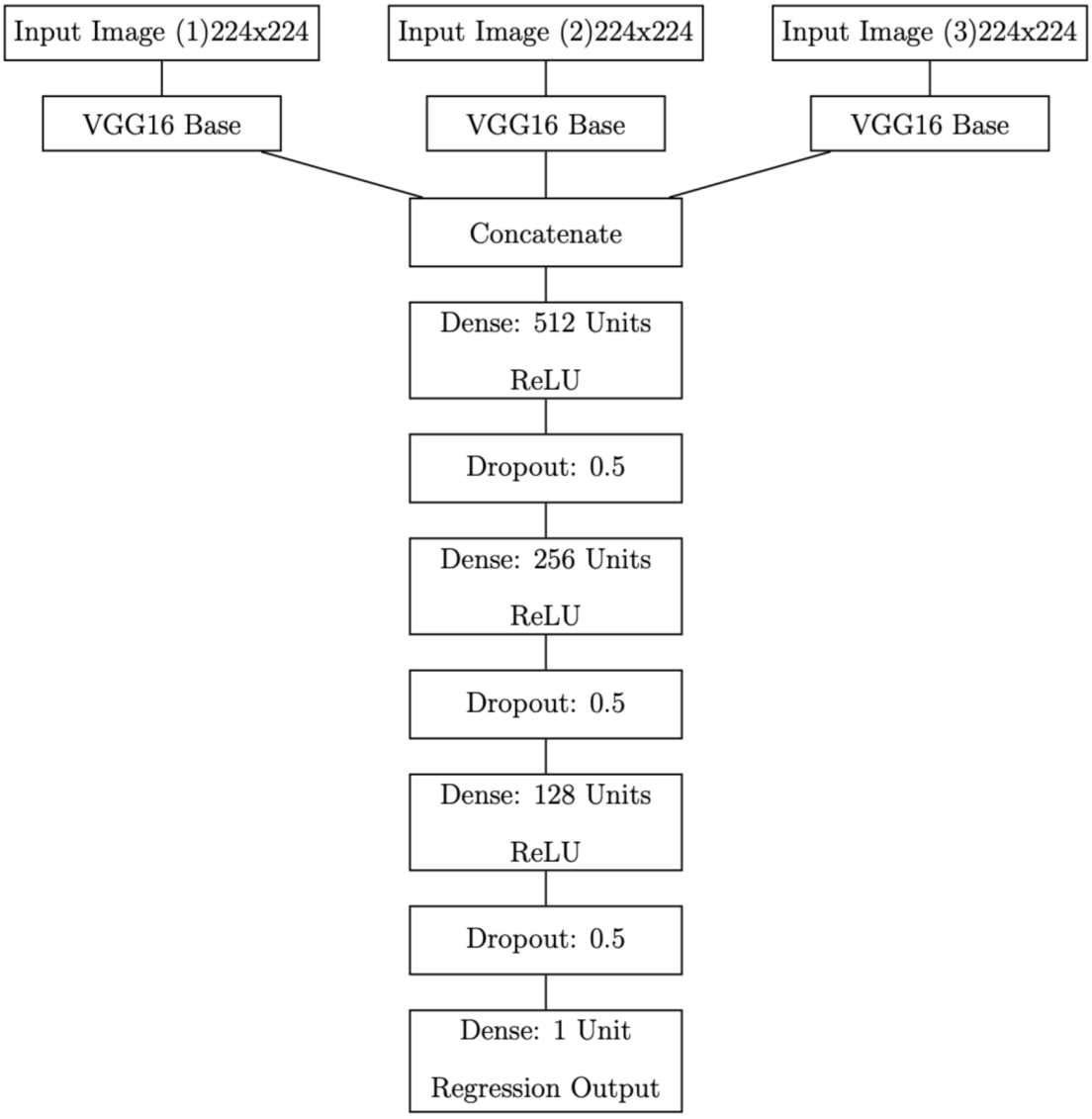



Figure 5.6: VGG Architecture, Tailored for the Multi-View Image Analysis of the Honda Hood Frames.

- Fully Connected Layers: At the end of the network, there are 3 fully connected layers. The first two have 4096 channels each, and the third performs 1000-way classification (which corresponds to the 1000 classes of the ImageNet dataset) and uses a softmax activation function.

- Fixed Input Size: VGG16 requires a fixed input size of 224x224 RGB images.

- ReLU Activation: ReLU activation is used throughout the network

- Here for multi-view, i.e., top, front, and side view regression task, I have used VGG16 as a feature extractor for each channel. The structure is described below,

- Input Layer: Each of the three input images is pre-processed to fit the input requirements of VGG16, i.e., resized to 224x224x3 pixels.

- VGG16 Base: The processed images were then input into their respective VGG16 pre-trained networks. The convolutional base of VGG16, which includes all the convolutional and pooling layers but not the fully connected layers at the top was used. These pre-trained convolutional layers were frozen while training and only the fully connected dense layers were trained.

- Feature Concatenation: After passing through their respective VGG16 bases, the feature maps were pooled (using global average pooling) and concatenated.

- Fully Connected Layers: The concatenated features then went through several fully connected layers with ReLU activations. Dropout was applied between these layers for regularization.

- Output Layer: The last fully connected layer was tailored for regression and output the maximum stress of the Hood.

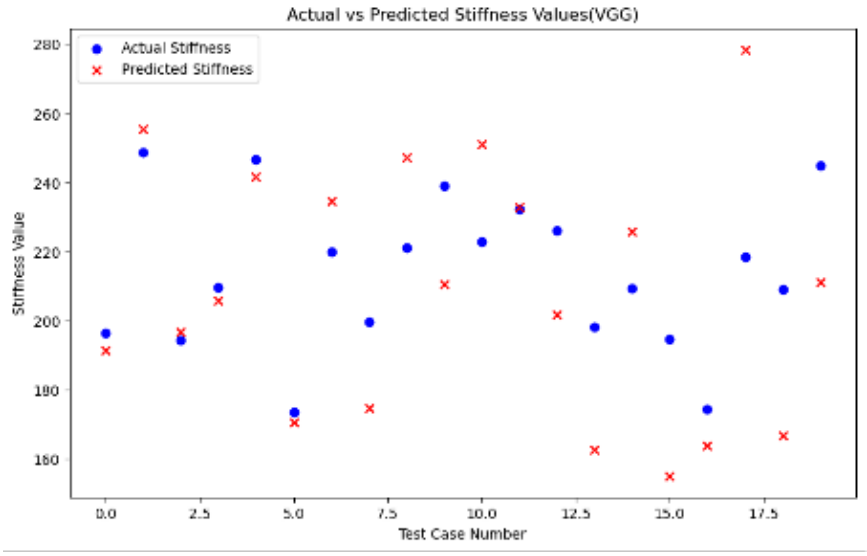



Figure 5.7: Actual vs Predicted maximum stress Value (VGG).

### *5.3.1 Results*

Implementing VGG16 led to improved accuracy across all specified tolerance levels compared to the Custom CNN network. Specifically, accuracy within a strict ±5% margin increased to 32.16%, suggesting that the VGG16-based model provided more precise predictions for nearly a third of the dataset. The trend continued with wider tolerances: accuracy within ±10% rose to 42.24%, and ±15% reached 65.79%, reflecting a substantial improvement in the model's ability to capture the correct trends within these ranges. Most notably, within a ±20% tolerance, the accuracy ascended to 82.52%, indicating a strong general prediction capability.

## 5.4 Predictive Model using ResNet-50 Architecture: Transfer Learning

ResNet-50 is a deep convolutional neural network architecture that gained significant popularity for its effectiveness in image recognition tasks, especially in the context of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). It

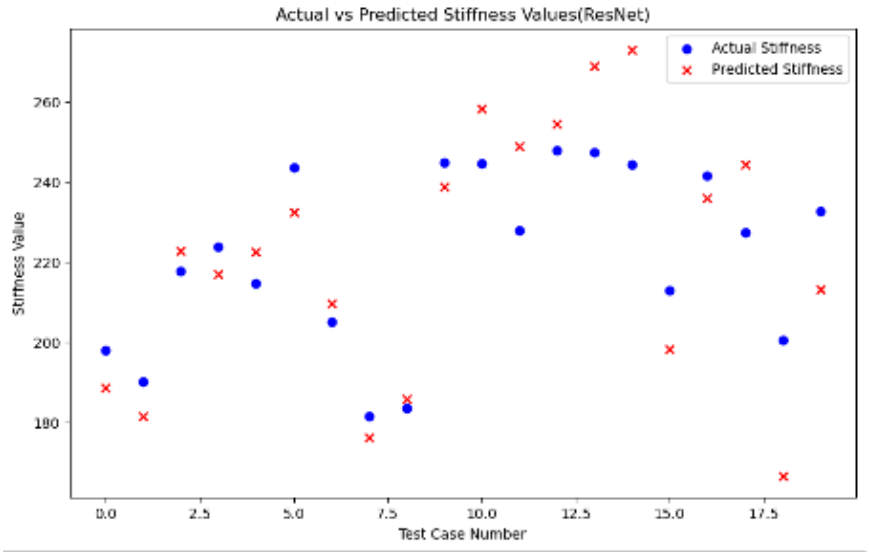



Figure 5.8: Actual vs Predicted maximum stress Value (ResNet-50).

is part of the ResNet (Residual Network) family developed by Microsoft Research, which introduced the concept of residual learning to address the problem of vanishing gradients in very deep neural networks.

Just like VGG16, using ResNet-50 with transfer learning for each channel involves leveraging a pre-trained network to harness its deep layers for feature extraction. ResNet-50, with its 50 layers, includes a series of residual blocks that make it adept at training very deep networks efficiently. The defining feature of ResNet-50 is its skip connections, which allow gradients to flow through the network directly, combating the vanishing gradient problem often encountered in deeper networks.

The original ResNet architecture was introduced in the paper titled "Deep Residual Learning for Image Recognition" by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun [32]. We do not here attempt to repeat that discussion.

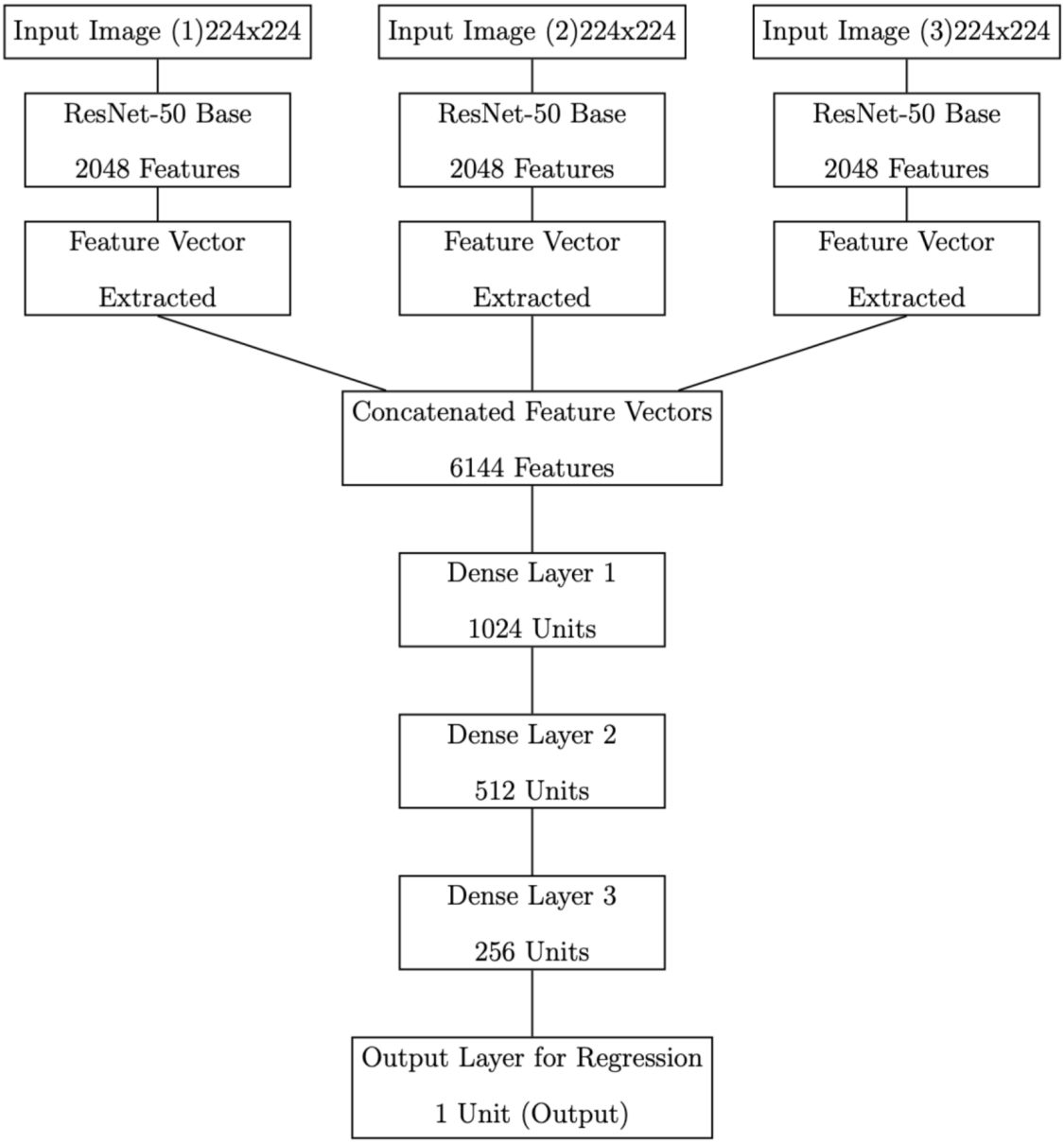



Figure 5.9: The ResNet Architecture, Tailored for the Multi-View Image Analysis of the Honda Hood Frames.

### *5.4.1 Results*

The application of ResNet-50 has notably enhanced prediction accuracy within various margins of error compared to the previously discussed two methods. The model now accurately predicts within ±5% for 45.54% of the dataset, showcasing a significant improvement for the most stringent margin. As the tolerance increases to ±10% and ±15%, the accuracy climbs to 62.43% and 74.52% respectively, indicating a robustness in capturing closer approximations of the target values. Impressively, within the broadest margin of ±20%, the model achieves over 90% accuracy, demonstrating a strong overall predictive capability with the ResNet-50 architecture.

# Chapter 6

# CONCLUSION

The primary objective of this study was to extract geometric parameters from simulation data and subsequently train machine learning algorithms with this information. The results and conclusions are detailed in each respective chapter. In Table 6.1, I have summarized all the new contributions presented in this thesis.

Table 6.1: List of Contributions in this Thesis.

| **S. No.** | **Topic** | **Type** |
|---|---|---|
| 1 | Radius of curvature due to spring back in hat-shaped stamped components | Method & Application |
| 2 | Zone Sizes for the spot welded assemblies of hat section of flat sheet | Application |
| 3 | Neural network model to predict Zone Size | Application |
| 4 | Twist Angle between two parallel edges of hat sections and spot welded assemblies of flat sheet | Method & Application |
| 5 | Angles between fitted planes to base of horizontal & vertical hat in a T-Joint | Method & Application |
| 6 | Adding iterations to virtual work & Robotics method in 2D & 3D | Method & Improved results |
| 7 | Deep learning models to predict maximum stress of Honda car hood frames | Application |

REFRENCES


[1] Adrian A., Automation and Validation of Big Data Generation via Simulation Pipeline for Flexible Assemblies, MS Thesis, Ohio State University, 2022.

[2] M. Makinouchi, E. Nakamichi, E. Onate, R. Wagoner, In: Proceedings of the NUMISHEET'93 Second International Conference on Numerical Simulation of 3D Sheet Metal Forming Processes, Tokyo, Japan, 1993.

[3] On˜ate, E., J. Rojek and C. Garc´ıa Garino, "Numistamp: a research project for assessment of finite-element models for stamping processes", Journal of Materials Processing Technology 50, 1-4, 17–38 (1995).

[4] Srinivasan, Vijay & Shakarji, Craig & Morse, Edward. (2012). On the Enduring Appeal of Least-Squares Fitting in Computational Coordinate Metrology. Journal of computing and information science in engineering. 12. 10.1115/1.3647877. Jolliffe IT. Principal Component Analysis. New York: Springer; 2002.

[5] Kåsa I. A circle fitting procedure and its error analysis. IEEE Trans on Instr and Meas, vol. IM-25, no. 1, pp. 8-14, March 1976, DOI: 10.1109/TIM.1976.6312298.

[6] Lourakis, Manolis IA. "A brief description of the Levenberg-Marquardt algorithm implemented by levmar." *Foundation of Research and Technology* 4.1 (2005): 1-6

[7] Mohan, P., Haghighi, P., Shah, J. J., and Davidson, J. K. (2015). Development of a library of feature fitting algorithms for CMMs. International Journal of Precision Engineering and Manufacturing, 16(10), 2101–2113. https://doi.org/10.1007/s12541- 015-0272-1

[8] Yegnanarayana, Bayya. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009

[9] Amari, Shun-ichi. "Backpropagation and stochastic gradient descent method." *Neurocomputing* 5.4-5 (1993): 185-196

[10] Hinton, Geoffrey, Nitish Srivastava, and Kevin Swersky. "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent." *Cited on* 14.8 (2012)

[11] Zhang, Zijun. "Improved Adam optimizer for deep neural networks." *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*. Ieee, 2018

[12] Szandała, Tomasz. "Review and comparison of commonly used activation functions for deep neural networks." *Bio-inspired neurocomputing* (2021): 203-224

[13] Badola, Akshay, Vineet Padmanabhan Nair, and Rajendra Prasad Lal. "An analysis of regularization methods in deep neural networks." *2020 IEEE 17th India Council International Conference (INDICON)*. IEEE, 2020

[14] O'shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." *arXiv preprint arXiv:1511.08458* (2015)

[15] Sunkara, S.C., Statics with Robotics to Get the Least-squares Fit of Profiles for Evaluating FEA Simulations of Flexible Components and Assemblies, MS Thesis, Arizona State University, 2023

[16] Davidson JK, Savaliya SB, He Y, Shah, JJ. Methods of robotics and the pseudoinverse to obtain the least-squares fit of measured points on line-profiles. CD-Rom Proc, 17th ASME Des for Mfg & the Life Cycle Conf. Chicago, IL, DOI: #10.1115/DETC2012-70203.

[17] Davidson JK, Savaliya SB, Shah JJ. Least-squares fit of measured points for square line-profiles. Procedia CIRP 2013;10:203-10

[18] Ben-Israel A, Greville TNE. Generalized Inverse: Theory and Application. 2nd ed. New York: Springer; 2003.

[19] Strang G. Linear Algebra and its Applications. 4th ed. India: Thomson Brooks/Cole; 2006.

[20] Nakamura Y. Advanced Robotics: Redundancy and Optimization. New York: Addison-Wesley; 1991.

[21] Davidson JK, Hunt KH. Robots and Screw Theory. Oxford, UK: Oxford; 2004.

[22] Joshi A. Quantifying deformations in flexible assemblies. MSc thesis, Arizona State University, 2020

[23] Nelder JA, Mead R. A Simplex method for function minimization. The Computer Jnl 1965; 7: 308-13. doi.org/10.1093/comjnl/7.4.308.

[24] Virtanen P, Gommers R, Oliphant TE et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. Nature Methods 2020; 17: 261–72. doi.org/10.1038/s41592-019-0686-2.

[25] Zhang X, Zhang H, Zhang L, Shen, J. High-precision extraction of segment points of 2D profiles based.on a dynamic grid method. Math Probs in Engr 2019. DOI:10.1155/2019/2052319

[26] A. Adrian, S. Ramnath, S. C. Sunkara, Y. Korkolis, J. K. Davidson and J. J. Shah, "Curating Datasets of Flexible Assemblies to Predict Spring-back Behavior for Machine Learning Purposes," in Proceedings of MSEC2022, West Lafayette, IN, 2022. doi.org/10.1115/MSEC2022-85718.

[27] An, Yi, Cheng Shao, Xiaoliang Wang, and Zhuohan Li. "Geometric properties estimation from discrete curves using discrete derivatives." *Computers & Graphics* 35, no. 4 (2011): 916-930

[28] Ramnath, Satchit et al. (2022). OSU-Honda automobile hood dataset (CarHoods10k) [Dataset]. Dryad. https://doi.org/10.5061/dryad.2fqz612pt

[29] Tammina, S., 2019. Transfer learning using vgg-16 with deep convolutional neural network for classifying images. International Journal of Scientific and Research Publications (IJSRP), 9(10), pp.143-150.[76]

[30] Sahaai, Madona B., G. R. Jothilakshmi, D. Ravikumar, Raghavendra Prasath, and Saurav Singh. "ResNet-50 based deep neural network using transfer learning for brain tumor classification." In AIP Conference Proceedings, vol. 2463, no. 1. AIP Publishing, 2022.

[31] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In 2009 IEEE conference on computer vision and pattern recognition, pp. 248-255. Ieee, 2009.

[32] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

# APPENDIX A

# FLOWCHART ALGORITHMS

The following definitions and equations form the basis of several computations in the flowchart of Fig. 3. They are listed here in the terminology of this thesis, but without proof. Derivations and more elaborate discussions about them may be found in Chapters 3 and 4 and Appendix A from [21].

*Coordinates of Lines, Screws, Wrenches, SDTs, and Finite Twists*

The six coordinates $(L, M, N; P, Q, R)$, which correspond to the two concatenated vectors $(\mathbf{\Lambda}; \mathbf{\Pi})$, may be used to represent a line or a screw in 3D space; a wrench $(\mathbf{F}; \mathbf{T}) = (F_x, F_y, F_z; T_x, T_y, T_z)$, i.e., a combined force and moment; a small displacement torsor (SDT) $\$ = (\phi; \mathbf{X}) = (\Delta\theta_x, \Delta\theta_y, \Delta\theta_z; \Delta x, \Delta y, \Delta z)$; or a finite twist $(\phi; \mathbf{X})$ when the amplitude of displacement is not small.

For all of these, the first vector $\mathbf{\Lambda}$ acts along a line having direction ratios $\mathbf{\Lambda} = (L, M, N)$; these are direction cosines when the six coordinates are normalized. The second vector $\mathbf{\Pi}$ is formulated as the sum of the cross-product $\mathbf{\Lambda} \times \mathbf{r}$ and a component $h\mathbf{\Lambda}$ along the line, where $\mathbf{r}$ is any vector from the line to the origin of coordinates and $h$ is the pitch. For three of these entities, except lines, the pitch $h = \frac{\mathbf{\Lambda} \cdot \mathbf{\Pi}}{\mathbf{\Lambda} \cdot \mathbf{\Lambda}}$ in units of length (L) of axial progression per radian of rotation, and each lies on its (unit) screw of the same pitch $h$. When the pitch $h = 0$, the entity lies on a line (a screw of pitch zero), such as any $\$i$ in Figs. 1, 2, 4, and 5. (Since $h = 0$ for every line, the formulation for $\mathbf{\Pi}$ requires that coordinates for a line satisfy the compatibility condition $\mathbf{\Lambda} \cdot \mathbf{\Pi} = 0$).

For lines and screws, the six coordinates $(L, M, N; P, Q, R)$ are superabundant, so only their ratios are independent. Therefore, $\frac{1}{\rho}(L, M, N; P, Q, R)$ represents the same line or screw for any scalar $\rho$. The normalized coordinates of a line or screw are those for which $\rho$ is chosen so that

$$L^2 + M^2 + N^2 = \rho^2. \tag{A1}$$

A screw, a SDT, a twist, or a wrench of non-zero pitch, lies on a line that has coordinates $(L, M, N; P_l, Q_l, R_l) = (\mathbf{\Lambda}; \mathbf{\Pi}_l) = (\mathbf{\Lambda}; \mathbf{\Pi} - h\mathbf{\Lambda})$. Further, the incidence conditions for a point $(x, y, z)$ to lie on a line $(L, M, N; P_l, Q_l, R_l)$ are the three equations [30]:

$$\begin{bmatrix} P_l \\ Q_l \\ R_l \end{bmatrix} = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \begin{bmatrix} L \\ M \\ N \end{bmatrix}. \tag{A2}$$

The purpose of normalized coordinates is that they formalize the separation of mechanics (intensity of a wrench or amplitude of a twist or SDT) from the geometry, i.e., from the six normalized coordinates that represent the line or screw along which it acts. The amplitude of a SDT or finite twist is

$$\phi = \left(\Delta\theta_x^2 + \Delta\theta_y^2 + \Delta\theta_z^2\right)^{1/2} \tag{1}$$

and the intensity of a wrench is

$$F = \left(F_x^2 + F_y^2 + F_z^2\right)^{1/2}. \tag{2}$$

In each case, the $\phi$, or $F$, takes the role of $\rho$ in Eq. (A1).

In this paper, we take the vectors $\mathbf{\Lambda}$ and $\mathbf{\Pi}$ in the order $(\mathbf{\Pi};\mathbf{\Lambda})$ for wrenches $(\mathbf{T};\mathbf{F})$ and the lines at actuators $\$i$ , but the reverse order $(\mathbf{\Lambda};\mathbf{\Pi})$ for both SDTs and finite twists. These arrangements allow every term of a matrix product of a wrench and SDT to be in units of mechanical work.

Transforming a SDT $\$_L = (L, M, N; P, Q, R)$

When the deviations $d_i$ are larger than first-order small quantities, no components of displacement of frame $O_L x_L y_L z_L$ relative to frame $O_n x_n y_n z_n$ are second-order small quantities that can be ignored. Instead, the full $4 \times 4$ homogeneous transformation $[A_{nL}]$ relating the frames is needed. Its formulation, in Eqs. (A2), (A3), and (A4) below, is taken from Eqs. (4.62) and (4.63) in §4.6.3 in [30]. Given the SDT $\$L - S$ in terms of its small rotation amplitude $\phi$ and small axial translation $s = h\phi$, and given the normalized coordinates $(L, M, N; P_l, Q_l, R_l)$ of the line on which it lies, the required transformation is

$$[A_{nL}] = \begin{bmatrix} & [\mathbf{R}_{nL}] & & \begin{matrix} x_{nL} \\ y_{nL} \\ z_{nL} \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x & o_x & a_x & x_{nL} \\ n_y & o_y & a_y & y_{nL} \\ n_z & o_z & a_z & z_{nL} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A3}$$

in which $[\mathbf{R}_{nL}]$ is the $3 \times 3$ rotation matrix

$$[\mathbf{R}_{nL}] = \begin{bmatrix} L^2V_\phi + C_\phi & MLV_\phi - NS_\phi & NLV_\phi + MS_\phi \\ LMV_\phi + NS_\phi & M^2V_\phi + C_\phi & NMV_\phi - LS_\phi \\ LNV_\phi - MS_\phi & MNV_\phi + LS_\phi & N^2V_\phi + C_\phi \end{bmatrix} \tag{A4}$$

and

$$\begin{aligned} x_{nL} &= V_\phi(MR_l - NQ_l) + P_l S_\phi + Ls \\ y_{nL} &= V_\phi(NP_l - LR_l) + Q_l S_\phi + Ms \\ z_{nL} &= V_\phi(LQ_l - MP_l) + R_l S_\phi + Ns \end{aligned} \tag{A5}$$

The three abbreviated notations are defined as $V_\phi = \text{versine}\,\phi = (1 - \cos\phi)$, $S_\phi = \sin\phi$, and $C_\phi = \cos\phi$.

Obtaining deviations in frame $O_L x_L y_L z_L$ from the values in $O_n x_n y_n z_n$ requires the use of transformation

$$[A_{Ln}] = [A_{nL}]^{-1} = \begin{bmatrix} & [\mathbf{R}_{nL}]^T & & \begin{matrix} -\mathbf{n}\cdot\mathbf{x}_{nL} \\ -\mathbf{o}\cdot\mathbf{y}_{nL} \\ -\mathbf{a}\cdot\mathbf{z}_{nL} \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A6}$$