Interactive Quality Inspection of Measured Deviations in Sheet Metal Assemblies

Felix Claus[1]
Computer Graphics & HCI Lab, Computer Science Department,
Technische Universität Kaiserslautern, Germany

Hans Hagen
Computer Graphics & HCI Lab, Computer Science Department,
Technische Universität Kaiserslautern, Germany

Viktor Leonhardt
Scientific Visualization Lab, Computer Science Department,
Technische Universität Kaiserslautern, Germany

Heike Leitte
Visual Information Analysis, Computer Science Department,
Technische Universität Kaiserslautern, Germany

Bernd Hamann
Department of Computer Science, University of California, Davis, CA 95616, USA

Abstract

We present an exploratory data analysis approach for finite element (FE) simulations to interactively inspect measured deviations in sheet metals arising in automotive applications. Exterior car body parts consist of large visible surfaces, and strict tolerances must be met by them to satisfy both aesthetic requirements and quality performance requirements. To fulfill quality requirements like gap and flushness, exterior vehicle components have adjustable mechanical boundaries. These boundaries are used to influence the shape and position of a sheet metal part relative to its chassis. We introduce a method that supports an inspection engineer with an interactive framework that makes possible a detailed analysis of measured sheet metal deviation fields generated from 3D scans. An engineer can interactively change boundary conditions and obtains the resulting deviation field in real-time. Thus, it is possible to determine viable and desirable adjustments efficiently, leading to time and cost savings in the assembly process.

2012 ACM Subject Classification Applied computing

Keywords and phrases Data Analysis, Interactive Inspection, 3D-Metrology, Finite Element Simulation

Digital Object Identifier 10.4230/OASIcs.iPMVM.2020.6

Funding This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 252408385 – IRTG 2057.

Acknowledgements We thank Q-DAS GmbH for providing CAD models.

1 Introduction

Measuring the dimensions of produced car body parts is one of the most important aspects of quality assurance. Regarding sheet metal assembly processes in today's automotive industry, dimension assessment is usually condensed by measuring single points on an assembled part. These arbitrary points, referred to as KPIs (key performance indicators) must be chosen

[1] Corresponding author

2nd International Conference of the DFG International Research Training Group 2057 – Physical Modeling for Virtual Manufacturing (iPMVM 2020).
Editors: Christoph Garth, Jan C. Aurich, Barbara Linke, Ralf Müller, Bahram Ravani, Gunther Weber, and Benjamin Kirsch; Article No. 6; pp. 6:1–6:18

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

carefully to obtain a satisfactory assessment of the quality of a measured part. We focus on the assembly of exterior automotive chassis components. KPIs are chosen such that gap and flushness between parts can be controlled. These properties have small tolerance ranges, and it is difficult to fine-tune them. The chosen tolerances are usually below the dimensional variability of the assembled parts. Adjustable boundaries are used to fine-tune gap and flushness for every assembly. When using the term "boundary" in the following, we refer to an actual physical boundary of a mechanical part. To support adaptive assembly, expensive tools with mounted measurement devices are employed to control gaps and flushness. However, additional inspection steps are needed to evaluate an assembly produced in an automated process. During detailed analysis, more inspection points are evaluated than can be actively controlled during the assembly process. To increase the amount of information captured by

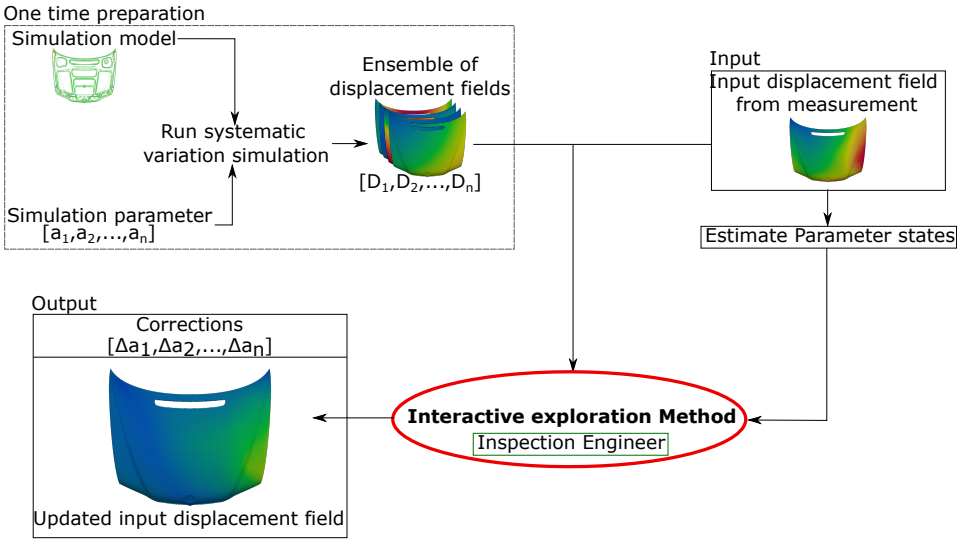

Figure 1 High-level description of workflow outline, grouped by the following tasks/data: "One-time preparation," "Input" and "Output".

a single measurement, optical measurement methods have become increasingly important for assessing sheet metal assemblies. Optical measurement technology captures the entire geometry of the visible surface of a produced assembly at high resolution. The "point clouds" generated by scanning can provide more detailed information about a measured geometry but, at the same time, it can be difficult to interpret the scanned data. Scans can be difficult to understand since a comparison between desired geometry and measured data results in 3D deviation (or difference) vector fields. To support simpler visual interpretation, we plot signed distance to the geometry instead of rendering the 3D offset vector field. When inspecting these color-coded distance fields, effects like dents, buckling, or elevations of large areas can be identified. However, the reason for identified effects can be a superposition of misplaced boundaries. Presumed geometrical compliance of the individual assembly components is already taken into account. The determination of appropriate countermeasures for greatly reducing identified deviations requires an experienced quality engineer with a substantial understanding of part behavior. Especially when parts are used for the first time it is often necessary to invest significant time to obtain the needed knowledge.

To speed up the mentioned trial and error experiments, this paper proposes an explorative approach to interact with the measured deviations. Figure 1 shows a high-level description of the workflow and the integration of the method. The figure is separated into "one-time

preparation", "Input" and "Output". As one-time preparation steps, an FE-simulation model must be set up from CAD geometry and an ensemble of displacement fields must be simulated sampling the solution space spanned by the simulation parameters that are considered adjustable. As input for the exploration method, a deviation field is calculated by comparing a measurement of an assembled part with its desired geometry. The obtained deviation field is then analyzed by a parameter estimation method to find a similar parameter state in the ensemble of pre-calculated displacement fields. Based on the obtained parameter state the input displacement field can then be updated by interpolating in between the pre-calculated displacement fields. So the user can change the parameters and gets a real-time update of the measurement. When the user found a satisfactory result, the output of the method is an updated displacement field and a set of necessary corrections. To meet the requirements of a system that is capable of updating measurements in real-time, our goal is to compute the update of the measurement below $0.5s$.

2 Background

This section summarizes the state of the art affecting the method this paper proposes.

Causal analysis of sheet metal assembly errors

The identification of potential causes for measured deviations is essential for quality assurance. Once the reason(s) is(are) identified, countermeasures can be executed. The concept of causal analysis is related to the topic of this paper, as the aim is to interactively identify proper countermeasures for measured deviations. [7] uses this concept for reducing the dimensional variation in multi-stage assemblies. When performing a causal analysis usually all possible influences on the process are considered and explanations for variation are identified. Typical causal factors in sheet metal assembly are, for example, assembly tool variation, part variation, the order of fastening screws or spot welds. Published approaches deal with optimizing these identified weak spots, see [17], [6], [16], [5], [15].

Parameter Estimation

Parameter estimation is widely used in different applications for identifying unknown quantities or states in simulation models, see [26], [3], [22]. In FE simulation, parameter estimation can be used for fine-tuning material parameters of a simulation model, feeding the estimator with experimental data, see [11] and [25]. [20] use parameter estimation as a kernel of a closed-loop control system for automotive purposes to predict critical motion states. The most relevant branch of parameter estimation methods used for our work is based on parameter estimation by least-squares methods. An overview of these methods is given in [13]. We already set up and validated a parameter estimation approach utilizing a least-squares method in connection with 3D-scanned deviations in sheet metal assemblies, presented in [8].

Proper Orthogonal Decomposition (POD)

POD is a branch of dimension reduction methods for finding a lower-dimensional basis for high-dimensional data. A review of POD methods is given in [18]. PODs are used in a wide variety of applications. For example, [4] use decomposition for reconstructing pressure fields on aerofoils from measured point data. [2] use POD for reducing the amount of necessary

single optical measures for recognizing a sheet metal part by reconstructing non-measured areas. The principle of POD is used in this paper to solve the parameter estimation problem for 3D-vector field applications.

Interactive/Explorative Design

Interactive design methods are commonly used for finding an optimal shape or design of a product to fulfill defined specifications by exploring the design space. These methods are used for a wide variety of applications and domains, see [21], [19], [24] . Especially in engineering applications, interactive design methods in combination with physical simulations become more popular, see [14], [9]. The method proposed in this paper is a hybrid approach containing aspects from interactive design methods as well as aspects from data exploration. To briefly cover literature from the field of data exploration we refer to an overview given in [12].

The presented literature covers many diverse topics of research spread over different disciplines. This work utilizes the knowledge from these different disciplines combining it to create a novel approach for interacting with measured sheet metal deviations. In particular, the contributions are:

- Method for supporting decision-making processes in the quality inspection of sheet metal assemblies
- Applicable for real-time interaction
- Validations performed with simulation- and experimental data

This paper unfolds as follows. First, the method is described in Section 3 covering different aspects like preparation steps and data acquisition, as well as a mathematical and algorithmic description of the method. Next, Section 4 presents a use case scenario showing validations with simulation- and experimental data. Last, the results are discussed in Section 5 and the paper is concluded by Section 6.

3 Method

In this section, the method that enables the interaction with the measurement data is explained in detail. Figure 2 depicts a detailed workflow that focuses on the “Interactive Exploration Method” mentioned in Figure 1 (red ellipse). In Figure 2 the interactive exploration method is also outlined with a red box. As input, we need an ensemble of pre-simulated displacement fields and the used simulation parameters. Also, the parameter states that led to the scanned deviation must be identified by a parameter estimation method. The estimated parameter states are compared to the pre-simulated parameters to determine the position of the scanned deviation field relative to the sampled points in the parameter space. Next, the deviation field can be represented by an interpolation between the pre-simulated displacement fields. This is achieved by weighting the pre-simulated fields corresponding to the distance calculated in the parameter space. At this point in the workflow, the user can make an input by changing the initially estimated parameter states of the scanned deviation. This kicks off an update of the calculated distances and thus the weights for the pre-simulated displacement fields become recalculated as well. By that, the scanned deviation field can be updated by the change of the new interpolation result. This update can be performed within a fraction of a second. So the user can interactively try out changes of parameters until he achieves a satisfying result. The loop depicted in Figure 2 is representing the update loop due to user input.

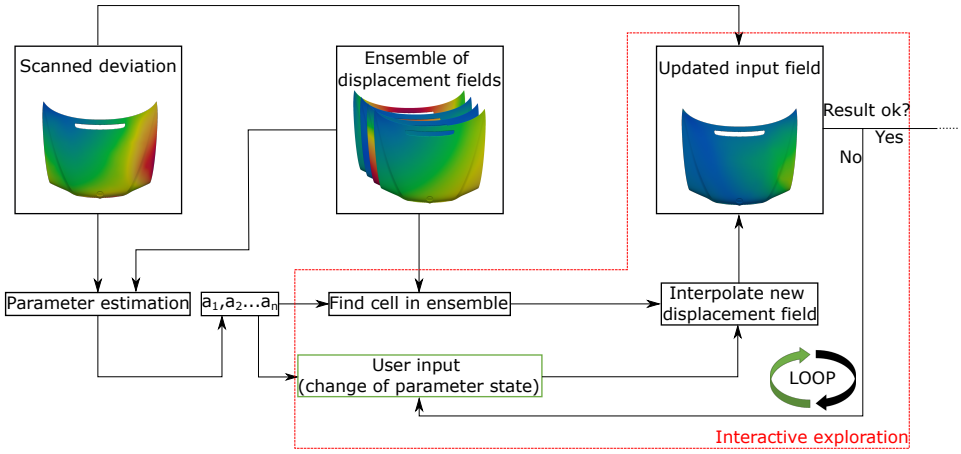

Figure 2 Detailed description of the workflow with attention to the interactive exploration loop marked in red. Inside this loop the user can change input parameters and gets an updated displacement field as response of the method.

3.1 Preparation steps

To be able to interact in real-time with measurement data, one-time preparation steps are necessary which are presented in this section. These preparations are already mentioned in Figure 1. First, an FE-simulation model must be created from a CAD-geometry for the part of interest. Also, the adjustable parameters and corresponding ranges must be defined and assigned to boundaries in the simulation model. Next, an ensemble of simulation runs needs to be created that samples the parameter space equally distributed. This ensemble can then be used to set up an estimator for solving the parameter estimation problem for a given deviation field. Different approaches can be used as kernel for the estimator. These already are highlighted in section 2. For this paper, the parameter estimation is considered as a problem that can be solved satisfactorily with methods already provided by the literature. With the estimator set up, the preparation steps are complete.

3.2 Mathematical Aspects

In this section the mathematical details of the interactive exploration method are explained, especially the data acquisition and the interpolation/approximation scheme are detailed in the following.

3.2.1 Data Acquisition

The generation of the pre-calculated simulation runs, especially the choice of how to sample the parameter space to achieve good interpolation results, is a crucial point. This data generation will be discussed for a three-dimensional case for demonstration purposes and is then extended to a six-dimensional parameter space to match up with the validations shown later in this paper.

Before we can perform interpolations on pre-simulated displacement fields, we need to perform multiple simulation runs to generate the necessary information. The amount of simulation runs $s \in \mathbb{N}$ that need to be performed, highly depends on the number of simulation parameters that are varied and on the sampling resolution $r \in \mathbb{N}$ of the parameter space

$X \subseteq \mathbb{R}^n$. For example: Let $\vec{a} \in X$ be a vector of one simulation parameter, where every parameter a_i can be varied in a range $[min_i, max_i]$ on its parameter space X_i.

$$\vec{a} = (a_1, a_2, ..., a_n), a_i \in [min_i, max_i]. \tag{1}$$

Then, the number of possible combinations of parameter states s for all axis of X can be calculated with:

$$s = r^n \tag{2}$$

Sampling Distribution

The sampling distribution of simulation runs presumed in equation 2 is equally spaced on every parameter axis $\{X_0, \ldots, X_n\}$ with the same amount of sampling points r per axis. The reason for choosing such a structured data set will be clarified in section 3.2.2. To get a better understanding of the structure of the resulting data set, Figure 3 shows an example with three simulation parameters and three sampling points per axis. The number of simulation

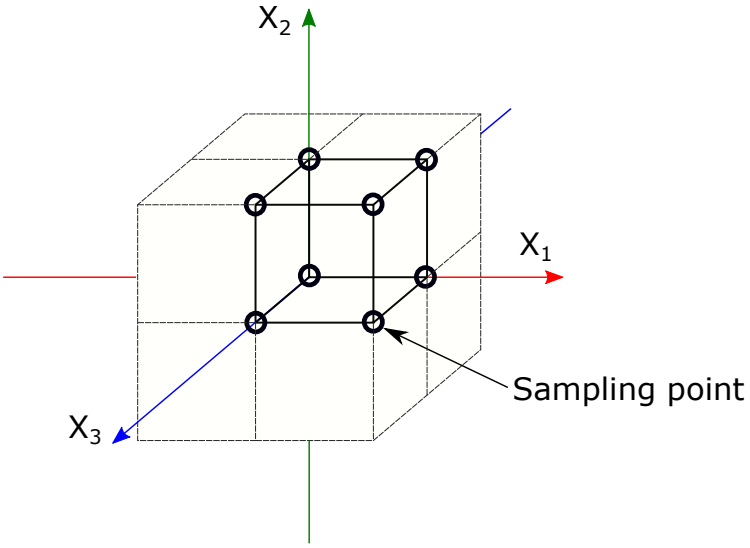

Figure 3 Simplified illustration in 3D of sampling one hypercube in a multidimensional parameter space.

parameters defines the order of dimension of the parameter space and the number of sampling points defines the number of (hyper)-cubes needed for filling the parameter space. In this particular example, we see three dimensions and three sampling points per axis resulting in eight cubes with a total of $3^3 = 27$ sampling points. One of these cubes and its sampling points (parameter 1,2,3 >= 0) is highlighted in the figure. Every sampling point in the parameter space is a combination of simulation parameters where an actual simulation run is performed, resulting in 27 simulation results which are the supports for the interpolation performed later on. As the simulation model of the use case presented in this paper has more than three adjustable simulation parameters, we next discuss how to apply this systematic sampling scheme on the general higher-dimensional case.

In the general case a geometric structure that subdivides a parameter space with the dimension n in an equally spaced manner, is called n-cube also named hypercube. In Table 1 the elements of the hypercubes up to $n = 6$ are listed. This table can be calculated by the

Table 1 Elements of an n-dimensional hypercube.

n	Name	0-face (vertex)	1-face	2-face	3-face	4-face	5-face	6-face	...
0	Point	1							
1	Line	2	1						
2	Square	4	4	1					
3	Cube	8	12	6	1				
4	Tesseract	16	32	24	8	1			
5	Penteract	32	80	80	40	10	1		
6	Hexeract	64	192	240	160	60	12	1	
⋮									

sequence A038207 that can be found in "The On-Line Encyclopedia of Integer Sequences®" (OEIS®) [1]. When we now consider six simulation parameters that are sampled with three points per axis we already need $3^6 = 729$ simulation runs that span 64 6D-hypercubes. Note that this exponential behavior for this systematic kind of sampling can result in a large number of simulation runs and thus in very expensive and time-consuming computations. To avoid an impossible amount of simulation runs, adaptive sampling strategies like provided by [23], can be applied. For the presented use-case no adaptive sampling was necessary.

3.2.2 Interpolation / Approximation Scheme

For generating information between sampled data points, we use interpolation. The interpolation scheme chosen in this paper for interpolating in a higher-dimensional space is called "inverse distance weighting". This scheme will now be explained in detail:

Let $\vec{a} = (a_1, a_2, ..., a_n)$ be a vector of parameters of one simulation and $\vec{u}(\vec{a})$ the corresponding simulation result represented by a displacement field, then we can express the pre-simulated displacement fields S as a set of simulation runs based on the parameter sets A:

$$A = \{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_m\} \tag{3}$$

$$S(A) = \{\vec{u}(\vec{a}_1), \vec{u}(\vec{a}_2), \ldots, \vec{u}(\vec{a}_m)\} = \{\vec{u}_1, \vec{u}_2, \ldots, \vec{u}_m\} \tag{4}$$

Every parameter set contained in the quantity from equation 3 is considered as a known or sampled point in the parameter space. The corresponding displacement field can be interpreted as the "value" of this point.

Let $\vec{b} \in X$ be also a set of simulation parameters but $\vec{u}(\vec{b}) = \vec{u}_b \notin S(A)$, thus we consider $\vec{u}_b$ as an unknown value in the solution space $S(A)$ that needs to be interpolated. To do so, every known value is weighted by the inverse of the distance using the function $\lambda : X \times X \to \mathbb{R}$, calculated in the parameter space:

$$\vec{u}_b \approx \frac{1}{\sum_{l=1}^{m} \lambda(\vec{b}, \vec{a}_l)} \sum_{i=1}^{m} \lambda(\vec{b}, \vec{a}_i)\vec{u}(\vec{a}_i) \tag{5}$$

$$\lambda(\vec{b}, \vec{a}_i) = \frac{1}{D(\vec{b}, \vec{a}_i)} \tag{6}$$

$$D(\vec{b}, \vec{a}_i) = \|\vec{b} - \vec{a}_i\|_2 \tag{7}$$

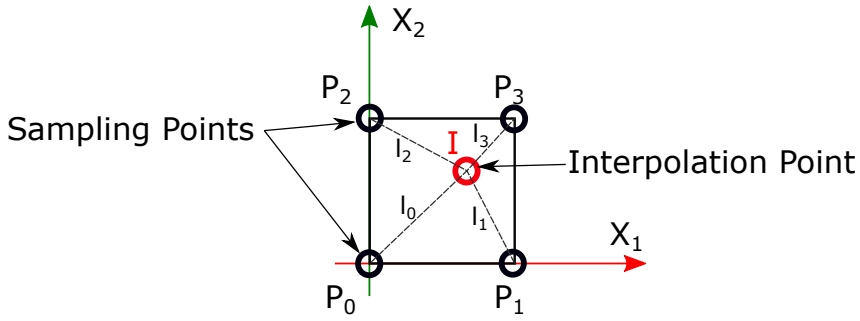

Figure 4 Calculation of the distances in the parameter space for an interpolation point (red) inside a sampled cell.

Equation 6 is a weighting factor by inverting the distance calculated between the point $\vec{b}$ and the sampled point $\vec{a}_i$, while

$$\frac{1}{\sum_{l=1}^{m} \lambda(\vec{b}, \vec{a}_l)} \tag{8}$$

is a factor to scale the sum of weights to 1 so that we can introduce the scaled weight $\lambda^*(\vec{b}, \vec{a}_i)$:

$$\lambda^*(\vec{b}, \vec{a}_i) = \frac{\lambda(\vec{b}, \vec{a}_i)}{\sum_{l=1}^{m} \lambda(\vec{b}, \vec{a}_l)} \tag{9}$$

To locally increase the weight of a sampled point, the inverse lengths are raised to the power of k:

$$\lambda^{*k}(\vec{b}, \vec{a}_i) = \frac{(\lambda(\vec{b}, \vec{a}_i))^k}{\sum_{l=1}^{m} (\lambda(\vec{b}, \vec{a}_l))^k}, where \sum_{i=1}^{m} \lambda^{*k}(\vec{b}, \vec{a}_i) = 1 \tag{10}$$

With the introduced scaled weights and the notation $\lambda_i^{*k} = \lambda^{*k}(\vec{b}, \vec{a}_i)$ we can express the interpolation as:

$$\vec{u}_b \approx \sum_{i=1}^{m} \lambda_i^{*k}(\vec{b}, \vec{a}_i)\vec{u}(\vec{a}_i) = \lambda_1^{*k}\vec{u}(\vec{a}_1) + \lambda_2^{*k}\vec{u}(\vec{a}_2) + ... + \lambda_m^{*k}\vec{u}(\vec{a}_m) \tag{11}$$

Figrue 4 demonstrates a simple 2D example the interpolation scheme. The red point I can be approximated by calculating the distances l_{0-3} and applying equation 11.

Weak Spots of Inverse Distance Weighting

When using inverse distance weighting, some aspects that are affecting the interpolation results are discussed in the following.

First, the distribution of the sampled points in the parameter space plays a significant role in this interpolation scheme. In Figure 5 an example is given for an uneven distribution of sampling points. The red point again is the point that needs to be interpolated. When performing inverse distance weighting on this kind of data set the interpolation result will be shifted towards the dense cluster of sampling points. Thus, the result of the interpolation would not be plausible anymore. Another factor that influences the interpolation result is the threshold radius when filtering the distances so that the interpolation does not have

to calculate weights and the linear combination of Equation 11 for the whole data set. Depending on the location of the interpolation point a slight change of the filter radius can result again in an even or uneven set of sampling points that are used for the interpolation. To prevent these shortcomings, the data set of pre-simulated displacement fields is distributed in an even manner like presented in Section 3.2.1. To prevent uneven distribution due to distance filtering, our method uses just the nodes of the hypercube where the interpolation node is located in. These sampling points are found by sorting the whole data set by distance and use the n closest points, where n is equal to the number of 0-faces of the corresponding hypercube, see Table 1.

Although there are more general approaches available in the branch of multivariate interpolation commonly used like kriging, spline interpolation, or radial basis function interpolation, inverse distance weighting is preferred in this paper. Different methods for scattered data interpolation are discussed and tested in [10]. The reason for choosing inverse distance weighting for solving the interpolation problem in the higher dimensional parameter space is the simplicity, robustness, and easy to implement nature of the algorithm. Also, the mentioned shortcomings of this algorithm can be prevented as discussed.

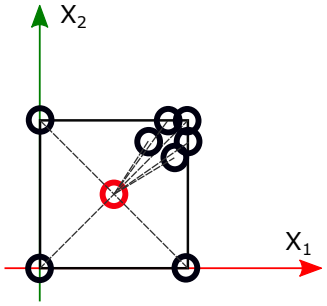

Figure 5 Influence of the distribution of sampling points. Local clusters lead to a higher weighting towards the cluster position.

3.3 Algorithm

In Algorithm 1, a pseudocode description of the interactive update is given. The input for this algorithm is an initial parameter set that is obtained from solving the parameter estimation problem for a measured deviation field. This initial parameter set is necessary to locate the measured deviations in the parameter space. Next, the measured displacement field is needed that will be updated by the algorithm. The third input is a list of parameters that were used to generate the pre-simulated displacement fields. To this list, the actual displacement fields are linked by a run-ID. The output of the algorithm is the updated measured displacement field.

At the beginning of the algorithm, a baseline is calculated by interpolating a displacement field for the initial parameter set. As we are interested in changes relative to this interpolation result, we need to subtract it from further interpolation results. For the interpolation, the steps that are described by the function “interpolate” have to be performed. First, the distances between the target point (in this case the initial parameter set) and every sampling point is calculated in the parameter space. Then the sampling points are sorted by distance and the 65 (64 points for the nodes of the 6D-hypercube and one point for the center of mass) closest points are used to perform the interpolation. For these 65 points, weights are calculated as described in Equation 11. Last, the weights are multiplied with the corresponding pre-

Algorithm 1 Pseudocode of Algorithm, structured by introducing two functions "calculate_weight" and "interpolate" that are used in the main body.

Data:
- $\vec{a}\_init$: Initial parameter set obtained from parameter estimator
- *measurement*: Measured displacement field
- *parameter_list*: Ensemble of pre-calculated displacement fields and corresponding list of simulation parameter

Result: Updated measured displacement field
Function `calculate_weight(`*distances*`):`
  # calculates the weight using Equation 11.
Function `interpolate(`*point*`):`
  distance_list=parameter_list - point
  distance_list.sort_by_distance()
  parameter_list.sort_by_distance()
  cut_off_list=distance_list[:65] # get the first 65 entries
  for *i in length(cut_off_list)* **do**
    weight=calculate_weight(cut_off_list[i])
    Output+=weight * simulation_result(parameter_list[i])
  end
  return Output
Main
baseline=interpolate($\vec{a}_{init}$)
#interactive Loop
while *TRUE* **do**
  get user_input
  interpolation_result=interpolate(user_input)
  updated_measurement=measurement - baseline + interpolation_result
  render(updated_measurement)
end

simulated displacement fields. The weighted displacement fields are summed up and returned by the function. After the baseline is calculated, the interactive loop starts, where the user can make changes in the form of a parameter set. This input parameter set is then also interpolated by the same function "interpolate". The measured displacements are then updated by subtracting the baseline and adding the interpolation result of the user input resulting in an approximated displacement field for the new parameter state. Finally, the updated field is rendered. The interactive loop runs until it is stopped by the user.

4 Use Case

We present a real-world use case scenario with validations performed on simulation and experimental data sets.

4.1 Simulation Setup

The chosen part is the engine hood of a car body. This example represents an assembly consisting of seven individual parts. The assembly structure is shown in Figure 6. Besides the visible outer skin, two inner parts are visible when lifting the hood. Four thicker sheet

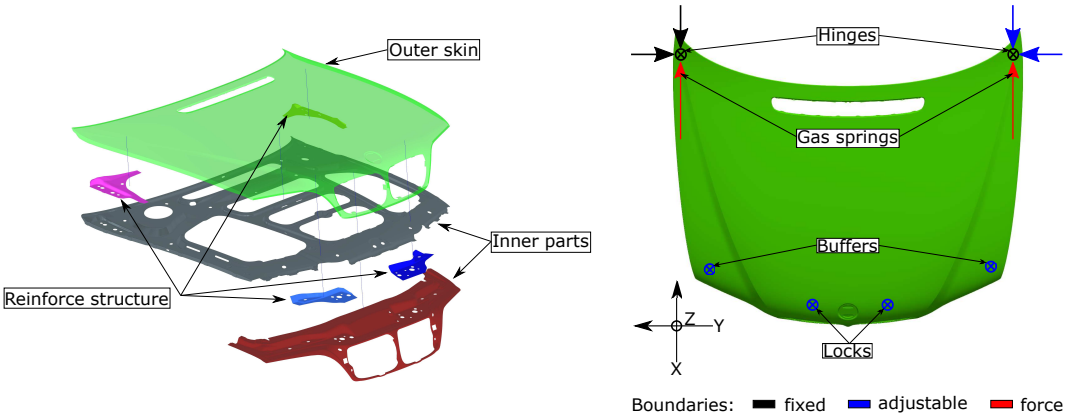

Figure 6 Left side: assembly structure of used hood. Right side: boundaries used for simulation with the coordinate system. The boundaries are "fixed" (cannot move at all), "adjustable" (to manipulate hood shape), or "force" (by using defined forces to model gas springs attached to the hood).

Table 2 Simulation parameters and corresponding value ranges.

Simulation parameter	Label	Description	Value range [mm]
a_1	HINGE_X	Hinge left, X-direction	[-1;1]
a_2	HINGE_Y	Hinge left, Y-direction	[-1;1]
a_3	BUFFER_L	Buffer left, Z-direction	[-1;1]
a_4	BUFFER_R	Buffer right, Z-direction	[-1;1]
a_5	LOCK_L	Lock left, Z-direction	[-1;1]
a_6	LOCK_R	Lock right, Z-direction	[-1;1]

metal parts are inside the hood to reinforce its structure. An FE simulation model based on the CAD geometry is created. Every component of the hood has meshed with first-order 3D thin-shell elements. The total number of mesh nodes is 365,683 and the number of mesh elements is 365,154. All components are joined together by modeling all spot welds as well as structural and acoustic adhesives. Finally, mechanical boundaries are added, as shown in Figure 6. This figure introduces the chosen coordinate system (X-, Y-, and Z-directions). The hood is attached with two hinges to the chassis. The right hinge (direction-of-travel) is held in position while the left hinge is adjustable in X- and Y-directions. In the closed state one gas spring per side is pushing with 580N in X-Z-direction against the hood near the hinges. In the front, two locks and two buffers are modeled as boundaries in Z-direction. While the buffers can only push against the hood, the locks can push and pull in both Z-directions. All boundaries are applied to the inner parts or the reinforced structure. Figure 6 shows an overview of the locations of all boundaries and Table 2 lists the adjustable boundaries with the corresponding value ranges. Based on the adjustable boundaries and the range of values, the data set of pre-simulated displacement fields is generated. We sample three points per axis. The number of simulation parameters is six, and by applying Equation 2 a total of 729 simulation runs are performed, used to define 64 6D-hypercubes. Further, 64 additional simulation runs are performed by sampling the center point of every 6D-hypercube. This approach produces a denser sampling without affecting the underlying data structure and not affecting interpolation result negatively.

4.2 Validation with Simulation Data

To assess the performance of the interpolation using inverse distance weighting, a validation with simulation data is performed. The goal of this validation is to understand the scale of the interpolation error. A "worst-case scenario" is set up by measuring the interpolation quality of a point in parameter space that is as far away from all sample points as possible. Therefore, we exclude the 64 simulation runs from the data set that were generated at the center point of the 6D-hypercubes. Instead, we interpolate the data values for these center points to compare the result with the excluded simulation-based value. The result is shown in Figure 7 (parameter colored in red) for one hypercube. We argue that for every other interpolation point inside a 6D-hypercube, the result generated via inverse distance weight interpolation would be better, as the interpolation location will be closer to one of the 6D-hypercube's sampled corner points. Additionally, the samples of the center points are part of the data set for further interpolations. To show that the quality of interpolation improves for different parameters, random parameter sets are generated in the allowable range specified in Table 2 and full simulation runs are performed. An interpolation result is calculated for the same parameter sets. The simulation results are compared with the interpolation results. This comparison is shown in Figure 7 for three randomly generated parameter sets. The first row in the figure shows the simulation parameters used for simulating/interpolating the results for each column. Rows two and three show these results (displacement magnitude) as color-coded images. As differences are difficult to see when visually comparing these two rows, the last row shows the error distribution of the comparison of the simulated-based and interpolated displacement fields. For all three examples, the RMS error is below the worst-case scenario.

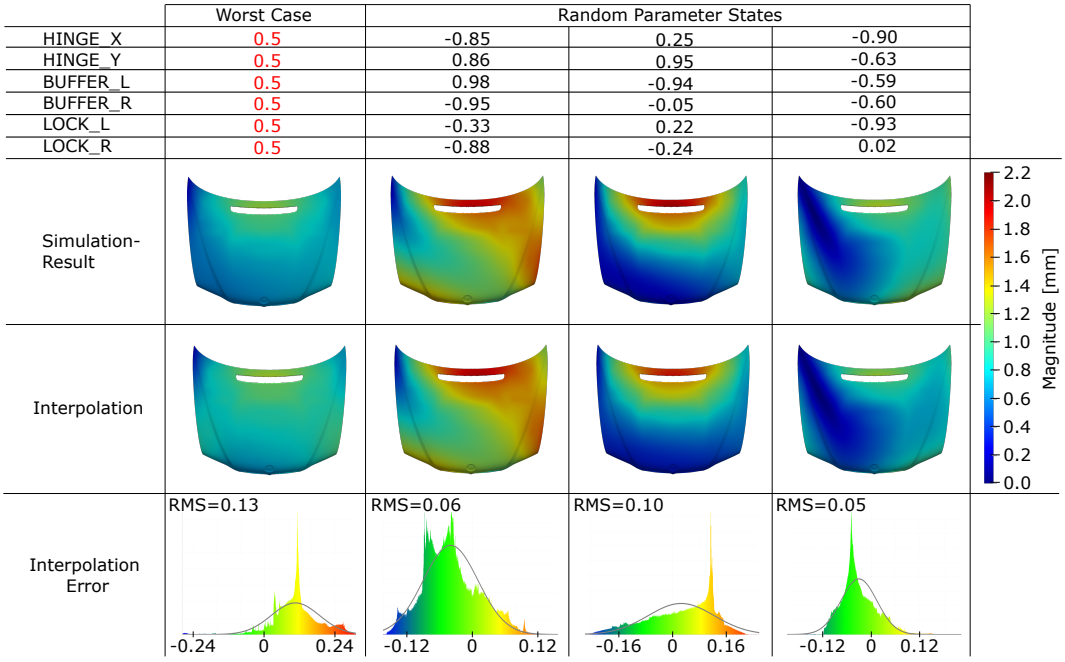

Figure 7 Comparison of simulation and interpolation. Top row: used parameter (red = worst-case scenario) to generate the simulation-based and interpolation results; second/third row: calculated displacement fields; bottom row: histogram for distribution of interpolation error evaluated at all vertices, RMS[2] error value.

[1] Root-Mean-Square

4.3 Validation with Experimental Data

The quality of the interpolation result is crucial for practical use. The results presented in Section 4.2 show promising performance concerning the precision, but further investigation with experimental data should be made. We describe our experimental setup and present results.

4.3.1 Hardware Set-up

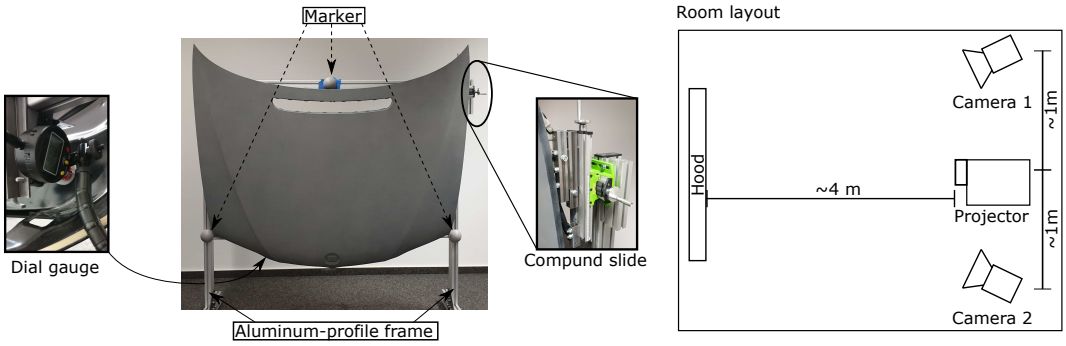

Figure 8 Left-hand side: experimental setup used for validation against measurement data; right-hand side: set-up used for scanning the engine hood.

The hood is prepared for measurement experiments by applying a light gray paint to it. This is necessary to generate a 3D scan with a structured light scanner. Besides the part itself, a fixture is designed and built from aluminum profiles. On top of this aluminum frame the locks, buffers, hinges, and gas springs are mounted. The set-up is depicted in Figure 8. To make the relative positions of the hinges adjustable, the left hinge is mounted onto a compound slide. The locks and buffers have proprietary adjustment options. The frame is equipped with markers to align different measurements. A dial gauge is used to change boundaries precisely. The hood is not mounted in the orientation as it would be in the assembled in a car, but this fact only affects the direction of gravity considered in the simulation. For the purpose of our validation, the orientation of the hood does not matter. Nevertheless, we point out that orientation matters when considering the scanning set-up, as depicted in Figure 8 on the right-hand side. The cameras and the projector of the structured light scan system must have a certain distance to the scanned object to capture it in its entirety. Usually, complex parts are captured with many pictures that are later stitched together. However, to reduce uncertainty caused by data registration, we capture every measured state of the hood in a single picture. This approach results in a relatively coarser resolution and noisier measurement data, but the quality of the result data is still sufficiently precise for performing our validation. A significant advantage of this set-up is the fact that the hood and the hardware components of the 3D scanner are not moved between scans. All measurements are made in the same coordinate system. This method almost removes the need for alignment. The markers mounted to the fixture are primarily used to determine whether something has moved between scans. (The measurement system used is a modified HP pro S3 structured light scanner, calibrated with custom calibration panels for the large size of the scan window.)

4.4 Experimental Validation Results

To assess the performance of the entire workflow, presented in Figure 2, a prototype implementation of Algorithm 1 is discussed next. We consider optical measurement data obtained for different boundary situations. Figure 9 shows the sequence of steps of the validation. First, a reference measurement is taken, and geometrical changes are computed relative to

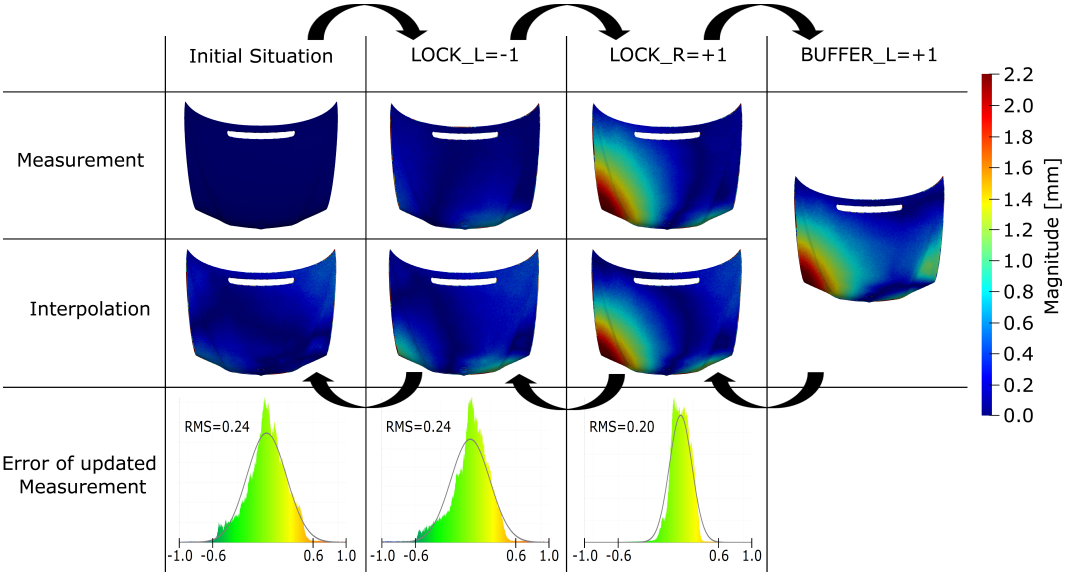

Figure 9 Experimental results. Top row: measured displacements caused by adjusting boundaries, compared to reference measurement (dark blue, leftmost); second row: interpolation results reconstructed "backward" from the measured displacement field (rightmost); third row: histogram for distribution of error caused by measurement update, evaluated at all vertices, and RMS error value.

this reference model (dark blue - top left). Single boundaries are changed, one after the other (top row). After every change of boundaries, the new geometry is captured by a scan. The resulting geometries are compared relative to the reference measurement, producing the displacement fields shown in the first row of the figure. The displacement field in the rightmost column is used as an input field for interactive interpolation. To validate the proposed method, Algorithm 1 is used to calculate the measured states, based on the displacement field. To reduce the influence of an interpolation error, only boundary states that match a sampled point in the pre-calculated simulation ensemble are used. When traversing the second row from right to left, we see the results produced by the proposed algorithm. The initial displacement field is updated by the change of the interpolation towards the geometry of the reference measurement, where the displacement should ideally be zero for every vertex. It can be observed that the resulting displacement fields do not match the measured data exactly.

4.5 Run Times

To assess the interactive aspect of our method, we measured run times, listed in Table 3. The results show that creating the simulation data ensemble – which is a one-time pre-computation step – requires about 103.5 hours. Nevertheless, as this is a pre-computation step that can

also be performed in parallel, it scales with the available computational resources. The last column, "Interpolation", is the relevant column, it documents interactive response capability of our presented method; the average computational time is only 0.2s.

Table 3 Computational times.

	Simulation Ensemble	Comparison of measurements	Interpolation
Computational time	793 x 470s = 103.5 h	2s	0.2s

5 Discussion

This paper addresses different aspects of realizing its goal. Thus, the following topics from the presented use case scenario are discussed in this section: The data generation of the pre-calculated simulation ensemble, the results from the validation with simulation data as well as the results from the validation with experimental data, and finally, the achieved run times.

To generate sampling data for a higher dimensional parameter space can result in a large number of necessary simulation runs. Already for the shown use case of six parameters, a coarse sampling resulted in 793 runs. As the range of values is small relative to the size of the part, a linear behavior can be presumed, making a coarse sampling sufficient. For the use case of tuning in gap and flushness in sheet metal assemblies, this assumption is appropriate. For parts that behave nonlinearly like ones that are made of composite materials, different sampling and interpolation strategies might be necessary.

The interpolation/approximation scheme used in this paper, to achieve the results presented in Section 4.2, is very simple and easy to implement. Although the achieved results are sufficient for the presented use case it is hard to generalize this specific approach to other use cases. The results in Figure 7 show an interpolation error (RMS) up to 0.1 (worst case excluded). Considering the used sampling grid resolution of $1mm$ on each parameter axis, extended by the additional sampling at the center of mass for each hypercube, this error is indeed not negligible. Nevertheless, for exploitative purposes we consider the achieved precision as sufficient, as the goal of the interpolation is not to replace simulation runs, but to provide an impression of what's happening in between the sampled points. When the user visually compares the simulation and interpolation result, it is hard to spot out differences, so we can argue that the achieved precision is sufficient for the presented use-case.

When using experimental data, things become more complicated. The results presented in Figure 9 show on the same scale visible differences. To justify these higher errors, different sources of errors were identified. First, every measurement system has uncertainties. The measurement system includes the 3D-scanner itself, the fixture, and environmental influences. The 3D-scanner is capable of providing scans with a precision of up to $0.05mm$. But as the measurement conditions were not ideal (compare Section 4.3.1) we assume a higher uncertainty, estimated at $0.1mm$. The second source of error is the fixture which is assumed as entirely rigid. During the experiments, it turned out that some boundaries show a slight runaway when changing the boundary state. This was recognized and re-adjusted, but anyway causes errors when comparing set parameters. Besides uncertainties caused by the physical setup, there are also sources of errors in the workflow of processing the measurement data. For example, smoothing and outlier removal as well as solving the registration problem to match measurement and simulation vertices is a crucial step. For this experiment, the assumption was made, that only displacements perpendicular to the surface of the part occur

as in-plane movements cannot be captured by a 3D-scan. The registration was simply made by finding the closest point when comparing two meshes. This assumption might lead locally to large errors. Nevertheless, when visually comparing the real measurements and due to interpolation updated measurement, we see highly similar behavior. So we can argue that the goal of providing information about the behavior of the measured part due to boundary changes is achieved.

Last, the interactive aspect needs to be addressed. Therefore, the run times of different tasks were recorded and listed in Table 3. The crucial step is the calculation of the interpolation result, which takes on average $0.2s$. This value meets the pre-defined design goal of below $0.5s$.

To summarize the discussion, we can say that the proposed method works satisfyingly, although individual components could be improved. Especially for the interpolation scheme, as well as for the point cloud registration problem, other methods might improve the overall result.

6 Conclusions

We have introduced an interactive approach for exploring deviation measurements of sheet metals, to help with the determination of viable boundary adjustments necessary to eliminate or significantly reduce deviations. Our improvement of existing methods is the design and implementation of an interactive system framework that can be utilized intuitively by an inspection engineer to identify quality shortcomings and quickly enforce countermeasures to positively affect sheet metal behavior in an assembly. Our system was driven by the design and quality objectives employed in today's automotive industry. Our system is especially helpful to an inspection engineer during the early stages of production when it is desirable to evaluate multiple combinations of boundary adjustments in a small amount of time. As our interactive system supports the real-time exploration of such combinations, it represents a substantial acceleration of the decision-making process thereby reducing time and cost. We have validated our methods and prototype system by applying them to simulated and experimental data. Our validation results support the effectiveness of our system for real-world scenarios.

Regarding potential directions for future research, it is possible to consider the use of other data interpolation or approximation schemes, instead of inverse distance weighting. Further, other point cloud registration methods could be utilized.

References

1 OEIS Foundation Inc. (2020). *The On-Line Encyclopedia of Integer Sequences*, 2020. URL: https://oeis.org/A038207.

2 Manoj Babu, Pasquale Franciosa, and Dariusz Ceglarek. Spatio-temporal adaptive sampling for effective coverage measurement planning during quality inspection of free form surfaces using robotic 3d optical scanner. *Journal of Manufacturing Systems*, 53:93–108, 2019. doi:10.1016/j.jmsy.2019.08.003.

3 Erin Santini Bell, Masoud Sanayei, Chitra N. Javdekar, and Eugene Slavsky. Multiresponse parameter estimation for finite-element model updating using nondestructive test data. *Journal of Structural Engineering*, 133(8):1067–1079, 2007. doi:10.1061/(ASCE)0733-9445(2007)133:8(1067).

4 T. Bui-Thanh, M. Damodaran, and K. Willcox. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal*, 42(8):1505–1516, 2004. doi:10.2514/1.2159.

5 Jaime Camelio, S. Jack Hu, and Dariusz Ceglarek. Modeling Variation Propagation of Multi-Station Assembly Systems With Compliant Parts . *Journal of Mechanical Design*, 125(4):673–681, January 2004. doi:10.1115/1.1631574.

6 Jaime A. Camelio, S. Jack Hu, and Dariusz Ceglarek. Impact of fixture design on sheet metal assembly variation. *Journal of Manufacturing Systems*, 23(3):182–193, 2004. doi:10.1016/S0278-6125(05)00006-3.

7 Johan S. Carlson and Rikard Söderberg. Assembly root cause analysis: A way to reduce dimensional variation in assembled products. *International Journal of Flexible Manufacturing Systems*, 15(2):113–150, 2003. doi:10.1023/A:1024453207632.

8 F Claus, H Hagen, H Leitte, and B Hamann. Decomposing deviations of scanned surfaces of sheet metal assemblies[submitted]. *Journal of Manufacturing Systems*, 2020.

9 Christopher Fleischmann, Irina Leher, Reinhold Hartwich, Marc Hainke, and Stefan Sesselmann. A new approach to quickly edit geometries and estimate stresses and displacements of implants in real-time. *Current Directions in Biomedical Engineering*, 5(1):553–556, 2019. doi:10.1515/cdbme-2019-0139.

10 Richard Franke. Scattered data interpolation: Tests of some method. *Mathematics of Computation*, 38(157):181–200, 1982. URL: http://www.jstor.org/stable/2007474.

11 A. S. Gendy and A. F. Saleeb. Nonlinear material parameter estimation for characterizing hyper elastic large strain models. *Computational Mechanics*, 25(1):66–77, 2000. doi:10.1007/s004660050016.

12 Stratos Idreos, Olga Papaemmanouil, and Surajit Chaudhuri. Overview of data exploration techniques. In Timos Sellis, Susan B. Davidson, and Zack Ives, editors, *Compilation proceedings of the 2015 ACM Symposium on Principles of Database Systems, ACM SIGMOD International Conference on Management of Data, and SIGMOD/PODS 2015 PhD symposium, May 31 - June 4, 2015, Melbourne, VIC, Australia*, pages 277–281, New York, NY, 2015. ACM. doi:10.1145/2723372.2731084.

13 Michael L. Johnson and Lindsay M. Faunt. Parameter estimation by least-squares methods. In *Numerical Computer Methods*, volume 210 of *Methods in Enzymology*, pages 1–37. Academic Press, 1992. doi:10.1016/0076-6879(92)10003-V.

14 Gergely Kristóf and Bálint Papp. Application of gpu-based large eddy simulation in urban dispersion studies. *Atmosphere*, 9(11):442, 2018. doi:10.3390/atmos9110442.

15 Xiaoyun Liao and G. Gary Wang. Simultaneous optimization of fixture and joint positions for non-rigid sheet metal assembly. *The International Journal of Advanced Manufacturing Technology*, 36(3):386–394, 2008. doi:10.1007/s00170-006-0827-5.

16 S. C. Liu, S. J. Hu, and T. C. Woo. Tolerance Analysis for Sheet Metal Assemblies. *Journal of Mechanical Design*, 118(1):62–67, March 1996. doi:10.1115/1.2826857.

17 Cong Lu and Hong-Wang Zhao. Fixture layout optimization for deformable sheet metal workpiece. *The International Journal of Advanced Manufacturing Technology*, 78(1):85–98, 2015. doi:10.1007/s00170-014-6647-0.

18 Kuan Lu, Yulin Jin, Yushu Chen, Yongfeng Yang, Lei Hou, Zhiyong Zhang, Zhonggang Li, and Chao Fu. Review for order reduction based on proper orthogonal decomposition and outlooks of applications in mechanical systems. *Mechanical Systems and Signal Processing*, 123:264–297, 2019. doi:10.1016/j.ymssp.2019.01.018.

19 Yuki Mori and Takeo Igarashi. Plushie. *ACM Transactions on Graphics*, 26(99):45, 2007. doi:10.1145/1239451.1239496.

20 Giulio Reina, Matilde Paiano, and Jose-Luis Blanco-Claraco. Vehicle parameter estimation using a model-based estimator. *Mechanical Systems and Signal Processing*, 87:227–241, 2017. Signal Processing and Control challenges for Smart Vehicles. doi:10.1016/j.ymssp.2016.06.038.

21 Evan Strasnick, Maneesh Agrawala, and Sean Follmer. Scanalog. In Krzysztof Gajos, Jennifer Mankoff, and Chris Harrison, editors, *Proceedings of the 30th Annual ACM Symposium on*

User Interface Software and Technology, pages 321–330, New York, NY, USA, 10202017. ACM. doi:10.1145/3126594.3126618.

22 Mario Teixeira Parente, Steven Mattis, Shubhangi Gupta, Christian Deusner, and Barbara Wohlmuth. Efficient parameter estimation for a methane hydrate model with active subspaces. *Computational Geosciences*, 23(2):355–372, 2019. doi:10.1007/s10596-018-9769-x.

23 Steven K. Thompson. Adaptive cluster sampling. *Journal of the American Statistical Association*, 85(412):1050–1059, 1990. doi:10.1080/01621459.1990.10474975.

24 J. Whyte, N. Bouchlaghem, A. Thorpe, and R. McCaffer. From cad to virtual reality: modelling approaches, data exchange and interactive 3d building design tools. *Automation in Construction*, 10(1):43–55, 2000. doi:10.1016/S0926-5805(99)00012-6.

25 F. Yoshida, M. Urabe, and V.V. Toropov. Identification of material parameters in constitutive model for sheet metals from cyclic bending tests. *International Journal of Mechanical Sciences*, 40(2):237–249, 1998. doi:10.1016/S0020-7403(97)00052-0.

26 A. Zerwer, G. Cascante, and J. Hutchinson. Parameter estimation in finite element simulations of rayleigh waves. *Journal of Geotechnical and Geoenvironmental Engineering*, 128(3):250–261, 2002. doi:10.1061/(ASCE)1090-0241(2002)128:3(250).