



Institut für Numerische Simulation

Rheinische Friedrich-Wilhelms-Universität Bonn

Friedrich-Hirzebruch-Allee 7 • 53115 Bonn • Germany
phone +49 228 73-69828 • fax +49 228 73-69847
www.ins.uni-bonn.de

J. Garcke, S. Hahner and R. Iza-Teran

Alignment of Highly Resolved Time-Dependent Experimental and Simulated Crash Test Data

INS Preprint No. 2208

May 2022

Alignment of Highly Resolved Time-Dependent Experimental and Simulated Crash Test Data

Jochen Garcke^{1,2}, Sara Hahner¹, and Rodrigo Iza-Teran¹

¹Fraunhofer Center for Machine Learning and SCAI

²Institut für Numerische Simulation, Universität Bonn

August 26, 2022

We investigate for car and component crash tests the comparison of highly resolved experimental data with corresponding simulation data. Due to recent advances for optical measurement systems, one can nowadays obtain surface measurement data from a real crash experiment with high resolution in space and time. These advances call for new data processing methods that allow an alignment of this experimental data with numerical simulation results. We propose an approach based on a data representation stemming from a discrete Laplace–Beltrami operator, which allows such an alignment as well as a joint visual comparative analysis of both data sources. The method enables the identification of the best corresponding simulation among several numerical results, which allows inferring physical quantities that cannot be measured in experiments. We evaluate the procedure on synthetic and real experimental data from two different setups.

Keywords: CAE Analysis, Alignment of Experiment and Simulation, Temporal Coherence, Embedded Deformation.

1 Introduction

Crash tests with prototypes and human-like dummies are carried out in the automotive industry to investigate the deformation of the vehicle during a crash and its effect on occupants. Over the last 30 years, computer-aided crash simulations using finite element methods have become an elementary component in the car development process [32]. Without having to carry out time-consuming and expensive physical crash tests on

prototypes, variations in material properties or geometry of the vehicle components can be investigated in detail in view of possible effects on the crash behavior. On the one hand, engineers make such design changes to fulfill safety requirements, while on the other hand they observe guidelines, including weight, costs, or functional properties. Nowadays, physical destructive tests are usually only carried out in the late stages of the development process for verification and regulatory purposes. At this stage, a comparison of measurements from the physical crash test with the results of numerical simulation needs to be performed. Here, the simulation data is available in form of function values on surface meshes, for example, the vertex coordinates or physical values as plastic strain.

This comparison of experimental and simulation data is typically performed using so-called coded targets, which are photogrammetric markers that are placed on the surface of the vehicle and are tracked by cameras [11, 12, 24]. Additionally, before and after the crash test, the components can be measured in detail.

In recent years new optical measurement systems were introduced that can deliver 3D point cloud measurements with high resolution in space and time [16]. With these techniques, one can now obtain detailed surface measurement data over time from a real crash experiment. But, a direct alignment between 3D simulations and 3D data is not yet possible. The underlying task is the non-rigid alignment of point clouds to a surface mesh evolving over time with no correspondences. We address these limitations and propose an approach that allows an alignment of simulation and measurement data in an entirely new way. Our approach is based on an analytical approach that can efficiently compare surface mesh data from finite element simulations, introduced in [19]. Here, mesh functions are projected onto the eigenbasis of a discrete Laplace–Beltrami operator, which results in a form of geometric Fourier analysis and additionally reduces the dimensionality.

Alignment of measurement and simulation data allows the verification of the quality of the performed simulations. Our results on two different datasets show that working in a low-dimensional space stemming from the eigenbasis improves not only the runtime but also delivers better results. Furthermore, we can jointly visualize experimental and simulation results in the low-dimensional space. Additionally, from simulation results, one now can infer physical quantities such as plastic strain or stress for experimental data, where these quantities cannot be measured in real experiments. The proposed alignment procedure, therefore, delivers a substantial improvement for the automotive crashworthiness analysis.

Our contributions can be summarized as a) the introduction of a new approach to bring experimental point cloud observations in correspondence to meshes originating from finite element simulations, b) the fast selection of simulation meshes, which match the experimental observations over time, and, by this means, c) enabling the visualization of both data types in a low-dimensional space.

Further on in section 2, we discuss related work. In section 3, we present the two different data sources of experimental and simulation data, followed by our proposed approach to align these two data types in section 4. We present results in section 5.

2 Related Work

2.1 Observation of Experimental Data and Comparison to Simulation Data

In this work we consider the comparison of experimental and simulation data over time as based on the 3D coordinates of points from the surface of the observed workpieces. Whereas the 3D coordinates for simulation data are known for all vertices that define the finite element mesh, during an experiment the 3D coordinates of surface points over time are calculated using a stereo camera system. Other experimental verification of numerical crashworthiness simulations is based on measured physical quantities, for example using absorbed energy and crush force curves [6] or force–displacement curves [25].

In the automotive industry it is common to use photogrammetric markers, also referred to as coded targets, which are manually placed on the visible surface of the vehicle [11, 12, 24]. Using a stereo camera system, the markers’ 3D positions over time can be calculated. When using this small number of manually selected code targets or photogrammetric markers, their positions are calculated also on the simulation surfaces. Then, the partial correspondence between the experiment and the simulation surface can be determined based on those code targets’ positions. Nevertheless, this method only delivers information about a sparse point set covering visible surface areas. Consequently critical areas are possibly missing and a detailed alignment is not possible. Additionally, before and after the crash experiment, the workpieces can be measured in more detail. Clearly, this cannot capture deformation behavior during the crash.

In our case, we want to align experimental and simulation data without using photogrammetric markers. Instead, we apply the GOBO (GOes Before Optics) method explained in section 3.1. This allows us to calculate the 3D positions of a dense selection of points on the whole visible surface including critical deforming areas. Additionally, the method allows sampling at a higher sampling rate over time [16].

Once the 3D coordinates of the observed areas are known, we want to align the simulation and experimental data over time. The selected GOBO measurement method delivers more detailed results than a few hand-selected code targets. Therefore, the GOBO measurement allows a better comparison of experiments and simulations. Nevertheless, the alignment of experimental and simulation data over time, which we address in this work, is more challenging because we have to compare dense point clouds and not only selected targets.

There are existing algorithms to geometrically align three-dimensional point clouds. A widely used one is the ICP (iterative closest point) algorithm [1, 5, 30]. It requires an initial estimate of the relative pose and iteratively selects a subset of points in both sets. One of the subsets is transformed via scaling, rotation, and translation to align with the other subset by minimizing the distance between the points. Many different variants exist, where the runtime is generally high for dense data sets. We refer to this as the ICP based approach and explain it in more detail in section 5.1.

2.2 Computing Temporal Coherence from Point Clouds and Mesh Data

The comparison of data from experiments and simulations in the car design process can be regarded as a specific application of temporal coherence for point clouds and meshes. In this context, temporal coherence is the correspondence between deforming meshes or point clouds over time taking into account that the points vary in location and that occlusions are also time dependent. The extraction of the temporal dynamics from point clouds and the representation of this information on a mesh is an active research area in computer graphics. We can here only refer to a selection of relevant research, see the respective references for other work.

To establish temporal coherence, correspondences between time steps from a set of temporal point clouds are necessary. Approaches modeling unknown correspondences are available. Tracking of surfaces based on non-rigid alignment is possible but difficult, as it depends on the reference frame and is subject to error accumulation in time, especially due to correspondence computation under noise and occlusion [33].

A related field for temporal correspondence is the compression of time-varying mesh sequences of varying topology. Approaches use, for example, a special data structure over time, such as an octree or a graph [35].

Methodologies from video compression that use motion prediction based on previous frames extend to geometry processing, e.g., [23]. Extraction of temporal coherence includes the use of skeletons and decimated frames. Both approaches cannot deal with the temporal change of occlusions. Other approaches keep track of the volume within the surface and use energy optimization to keep track of the coherence in time [9, 10].

Bojsen-Hansen et al. [3] tracked the first frame of the sequence using embedded deformation graphs. The main problem with this approach as well as functional maps [29] is that one assumes that the surfaces correspond bijectively and continuously between frames.

More recent approaches for extracting temporal coherence are based on deep learning. Those approaches learn a low-dimensional representation of time-dependent structures while at the same time learning the inter-frame correspondence [34]. Note that deep learning approaches require a large amount of data, where for virtual car development and testing the availability of large datasets for training is an unrealistic assumption.

Our goal, guided by the application, goes further than finding the temporal coherence of a temporal point cloud. We would like to compare different temporal data sets originating from measurements with the finite element meshes that are obtained by simulations of the effect of different parameter combinations in the design and which represent deformations. Also, even if we do rely on correspondence over all timesteps, this is not computed explicitly but over the temporal coherence of the simulation mesh.

2.3 Dimensionality Reduction for Simulations

Our approach for alignment is based on dimensionality reduction using a specifically constructed data representation, which reduces runtime and allows further data analysis. Generally, methods for dimensionality reduction are based on the assumption that high-

dimensional data sets (approximately) have a much lower intrinsic dimension [22]. The high dimension in this context is the number of points in the cloud $n \in \mathbb{N}$.

Different methods have been developed for dimensionality reduction, where principal component analysis is the linear baseline method [22]. However, nonlinear behavior is present in the car crash simulation data. Nonlinear dimension reduction methods such as diffusion maps or t-SNE have been successfully applied to analyze crash simulations and other engineering data [2, 18, 20, 26, 31]. Nevertheless, these approaches only consider point clouds and do not use mesh information. Autoencoders [13, 17] are a type of neural network that calculates low-dimensional representations and has been applied to the analysis of crash deformation over time, e.g., [14]. Nevertheless, the number of samples has to be very high for successful training of the neural network.

In [19], a geometrical method for low-dimensional representations of functions on simulation meshes was introduced, which shows good results in the analysis of bundles of finite element mesh simulations. In [7, 8] a geometry-driven data simplification together with linear dimensionality reduction was proposed for data analysis and rule-finding.

3 Experimental and Simulation Data

The two considered data sources of experiments and simulations result in different data representations. From an optical measurement system, we obtain a time series of 3D point clouds for each physical crash experiment. Additionally, we have a set of numerical simulation results, where each sample is a possibly deformed finite element mesh at a specific timestep encoding the computed deformation.

3.1 Measurement Data from Physical Crash Experiments

To measure the deformation of a 3D structure, we employ a novel GOBO projection-based high-speed measurement technique introduced in [16], which is based on the pattern switching principle. A (GOes Before Optics) GOBO projector consists of a light source, a light collector, the GOBO itself made of steel, and an imaging optical system. The setup from [16] utilizes a metal-halide lamp, an ellipsoidal reflector as a light collector, and a two-lens zoom camera system. By moving the GOBO a switching geometric pattern is projected to the observed 3D structure. Using the known positions of the stereo high-speed video cameras, one can calculate the 3D coordinates of the surface at many visible points from the two camera images [16]. This novel measurement method was developed to allow an analysis of fast processes. Therefore, it can be applied to measure the deformation of structures in a real crash experiment.

The positions of the cameras allow the measurement of a dense set of 3D points covering parts of the upper beam of our employed test structure. Because some areas might be occluded and the deformation changes the size of the scanned surface, the resulting point clouds are of different sizes and there is no correspondence or tracking of the points over time. We will refer to the resulting data set as E_{obs} . An exemplary point cloud from E_{obs} obtained from the optical measurement system is shown for the employed test structure in Figure 1 (left).

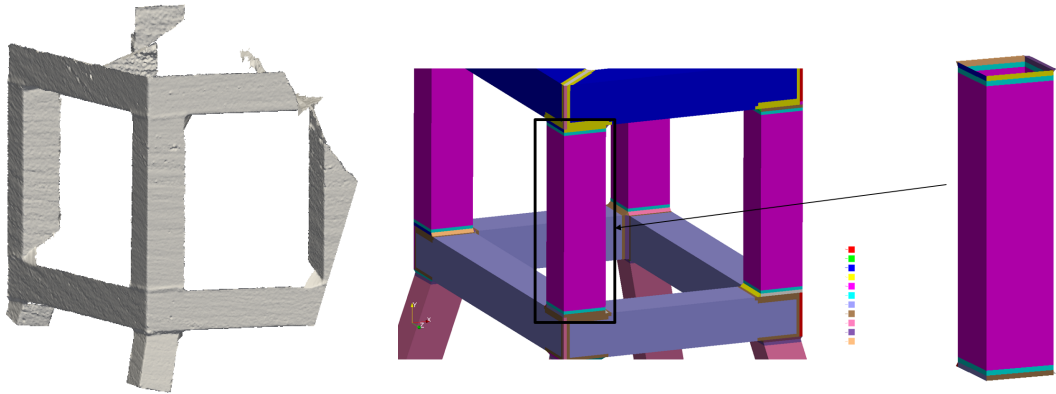


Figure 1: Data sources. Left: The real experiment is observed by a stereo camera system that outputs a dense point cloud registering visible areas. Right: A finite-element model of the test structure and the extracted beam for the analysis. For the alignment of simulation results and experimental data, we consider the visible two sides of the front beam, which is indicated by a rectangular frame.

We would like to stress that the extra quality obtained by the GOBO measurement system due to a highly detailed time and space resolution, results in challenging data processing. The mesh that can be created from such a dense point cloud is very fine and the processing is costly and time-consuming. Down-sampling the data is a possible approach, but then details are lost. In any case, the data between timesteps is not in correspondence so analyzing temporal coherence, meaning the temporal dynamics of the crash process, is not feasible. This is also due to occlusions as the camera focuses only on a part of the geometry that changes in time, which is observed from only one viewpoint.

3.2 Simulation Data from Virtual Crash Experiments

Simulation results for virtual crash experiments are represented as surface meshes in the three-dimensional space. Based on the test structure an initial mesh is generated by an engineer and used for the actual numerical simulation. The deformation of the parts is simulated using the finite element method [27], where sophisticated commercial solvers are available. One obtains a sequence of deforming meshes S_{obs} . The initial surface mesh for the crash simulations S_{obs} is shown for the employed test structure in Figure 1 (right).

Note that some mesh elements might be removed during the simulation by the numerical solver, often called failed elements. Therefore, although in principle the mesh is the same for the whole simulation, the number of nodes might differ slightly. Nevertheless, the meshes can be easily kept in correspondence, for example by reducing them to the common nodes and elements.

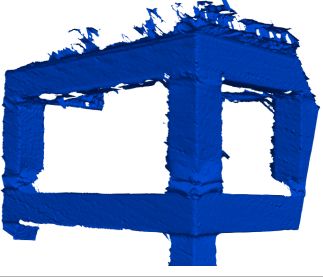
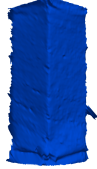


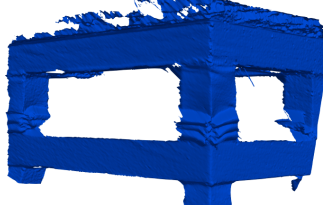



Time	Experiment		Simulations	
	Full	Cut	Full	Cut
$t = 10$				
$t = 20$				

Figure 2: Considered areas of experimental data.

4 Method: Alignment of Experimental and Simulation Data

We now aim to align the experimental observations E_{obs} (3D point clouds) to simulation results from S_{obs} (surface meshes) over time.

4.1 Preprocessing

The simulation results S_{obs} and the observations of real experiments E_{obs} are obtained by different methods, which lead to highly different data structures, see the previous section 3. Although the simulation results and experimental data represent the same deforming structure, we note that we require preprocessing for a comparison. The preprocessing establishes a projection of the data to a data representation that allows the comparison using dimensionality reduction. In particular, we need to establish vertex-to-point correspondence to a reference simulation sim_{ref} for all simulations in S_{obs} and experiments in E_{obs} . This will lead to two sets S and E containing point clouds of the same size n .

As part of the preprocessing, we identify the region captured by the optical device and take the corresponding one from the mesh in a way that both represent a similar area of the component. For example, surfaces of the component that are occluded in the experiment setup will be excluded from the surface mesh, as well as points of the point clouds that represent background or other components, see Figure 2.

The steps to establish the vertex-to-point correspondence can be structured into three and are visualized by the flow chart in Figure 3 and the plots in Figure 4.

1. **Rigid transformation:** The coordinate systems, in which the experimental data $exp \in E_{obs}$ and simulation results $sim \in S_{obs}$ are represented, do not necessarily

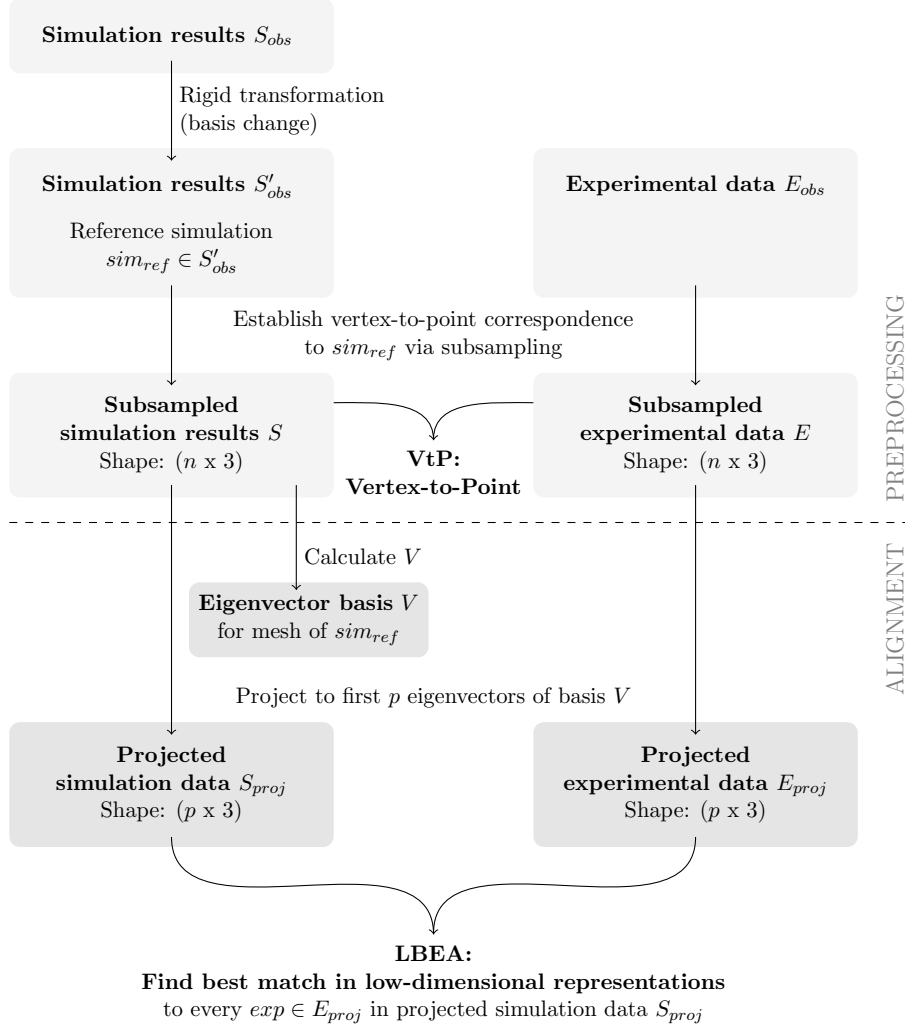


Figure 3: Preprocessing steps to create the vertex-to-point correspondence between simulation results and experimental data and the steps of the method LBEA (Laplace Beltrami Eigen Alignment) to align the data over time. The VtP (Vertex-to-Point) approach compares the subsampled samples in S and E .

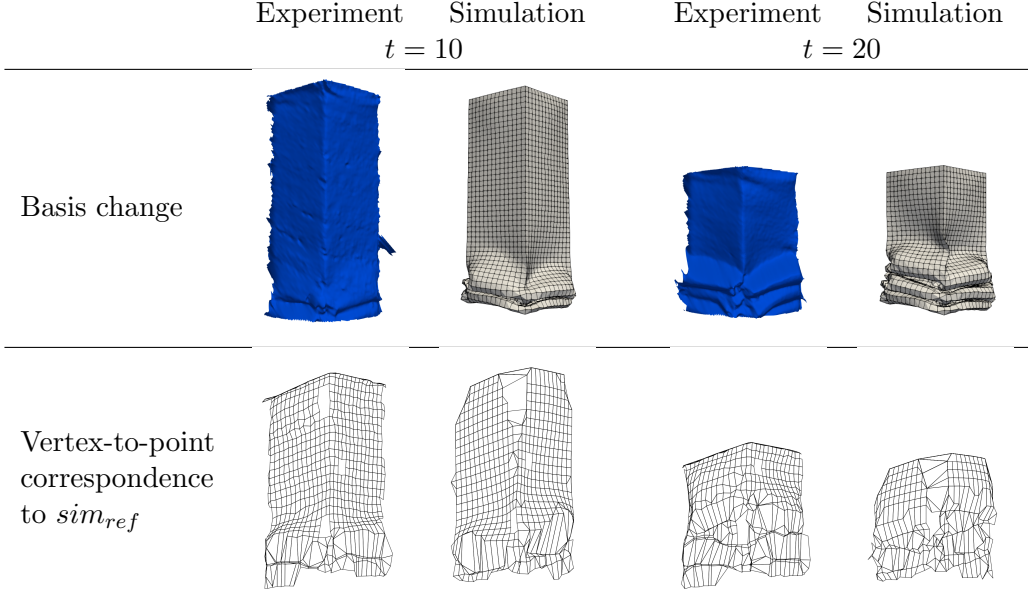


Figure 4: Vertex-to-point correspondence of simulation results and experimental data to the reference mesh sim_{ref} for dataset C-STRUCTURE. Since the experiment and the simulations are not aligned over time, the correspondence is to the undeformed mesh at $t = 0$.

coincide. We transform the simulation data, by applying a basis change that is calculated using the camera angle and distance as employed for the GOBO measurements. The experimental data E_{obs} and simulation data S'_{obs} are then represented in the same coordinate system, see Figure 1.

2. **Select a reference simulation sim_{ref} :** This step is necessary to establish vertex-to-point correspondence for all simulations $sim \in S'_{obs}$ and experiments $exp \in E_{obs}$. In particular, for the experimental data this is not given due to occlusions and deformations. Additionally, the visible area changes during deformation, which leads to a different size of the point clouds. The vertex-to-point correspondence is necessary to allow successful comparison of simulation results and experimental data, as well as the application of dimensionality reduction. There are two approaches to align the data into vertex-to-point correspondence:

- a) We chose a non-deformed mesh of a random simulation at the impact time as the reference simulation sim_{ref} . The impact time is the time step when the car or crash structure hits the barrier. All timesteps are therefore mapped to the same timestep.
- b) In case experiments and simulations are sampled along the same time scale, we chose a random reference simulation sim_{ref} and establish the correspondence to the simulation mesh at the corresponding timestep t .

If we consider only one timestep as in 2 a), the error in the alignment rises over time, because the translation and the increasing deformation lead to a lower quality of the correspondence. This effect is reduced by establishing the correspondence to the corresponding timestep as in 2 b).

3. **Establish the vertex-to-point correspondence:** We establish the vertex-to-point correspondence based on comparison to the reference simulation sim_{ref} . For all experiments $exp \in E_{obs}$ the algorithm finds the nearest point in exp to each of the vertices of the reference simulation sim_{ref} . The selected points of exp describe the new representation for the experimental samples. The same procedure is applied to all simulations sim in S'_{obs} (after the basis change). This way, all experiments and simulations are represented by n 3D coordinates (the number of vertices of sim_{ref}) and the vertex-to-point correspondence to the mesh of sim_{ref} is established. Therefore, all samples are represented by a surface mesh. We call the resulting sets E and S .

All samples from the preprocessed sets E and S are now in vertex-to-point correspondence to the mesh of sim_{ref} , see some examples in Figure 4. Note that more than one vertex of sim_{ref} can be in correspondence to a given point of the sample.

4.2 Alignment over time

The presented preprocessing routine establishes the vertex-to-point correspondence from the reference simulation mesh sim_{ref} to both, the simulation meshes and experimental data obtained from the 3D measurement method. Therefore, all simulation results in S and experiment results in E are point clouds of size n ($n \in \mathbb{N}$), and the data can be interpreted as a function on the mesh of the reference simulation sim_{ref} .

We investigate two approaches for an alignment over time, called VtP (Vertex-to-Point) and LBEA (Laplace Beltrami Eigen Alignment), respectively. For any experimental data in E , we look for the closest mesh in S , which consists of all time steps of all simulations.

4.2.1 Vertex-to-Point (VtP)

Given the vertex-to-point correspondence from section 4.1, each vertex of the mesh is associated with one point of each point cloud. Thereby, the experiment point clouds in E are in correspondence to the simulation meshes in S and we can calculate a distance between the experimental and simulation data on the meshes. This distance is calculated as the average distance between the positions of the corresponding vertices of the mesh. After calculating the pair-wise distances between all experiments and simulations, we define the best match by the experiment-simulation pair with minimum distance:

$$s^* = \arg \min_{s \in S} \sum_{i=1}^n \|e_i - s_i\|_2, \quad (1)$$

where e_i and s_i are the 3D coordinates for vertex i of the considered experimental or simulation data. We refer to this approach as Vertex-to-Point (VtP).

4.2.2 Laplace Beltrami Eigen Alignment (LBEA)

The second approach is based on the representation derived from a discrete Laplace-Beltrami operator, following [19]. Here, a function f given on the mesh vertices can be represented by the eigenvectors $\{\psi_j\}_{j=1}^p$ of this operator:

$$f = \sum_{j=1}^p \alpha_j \psi_j, \quad \alpha_j = \langle f, \psi_j \rangle. \quad (2)$$

We use the p eigenvectors associated to the p largest eigenvalues, in descending order. All the considered functions are thereby projected into a low-dimensional space of p dimension, where the analysis takes place. The eigenvectors can be understood as representing elementary components of the mesh’s deformation behavior. For more details see [19]. In our case of 3D coordinates, we have three functions per simulation and timestep to consider, i.e. separate functions for x, y, z .

The theoretical background of the approach requires that the data samples are represented on meshes that are invariant to approximately isometric transformations, in this case, deformations that preserve distances on the surface. If the data fulfills the requirements, the method can represent mesh functions from all timesteps of the simulation using one eigenbasis. Note that the data after vertex-to-point correspondence fulfills this assumption, because the deformation information is mapped to one reference simulation mesh sim_{ref} . On the other hand, a point can be assigned to several vertices of the reference mesh, which means that isometric transformations in the original data are not preserved in this preprocessing step. Nevertheless, the alignment procedure still works, since experimental and simulated data are processed in the same fashion.

The proposed approach based on a comparison in low dimensions has the advantages of robust results and low runtime. Furthermore, the representation allows identifying the importance of features, which allows further analysis of the deformation behavior.

Now, the alignment can be calculated by the following steps, see also Figure 3:

1. We calculate the eigenvector basis V for the mesh of the reference simulation $sim_{ref} \in S$ based on its discrete Laplace-Beltrami operator. Since all samples in S and E are in vertex-to-point correspondence to the reference simulation mesh, the eigenvector basis can be used to represent all samples. To reduce the dimension we chose the first $p \ll n$ eigenvectors resulting in the eigenvector basis V_p . For our experiments we chose $p = 100$.
2. Now, the low-dimensional representations for all $sim \in S$ are obtained by projection to the basis V_p . The resulting vectors are of size $p \ll n$ for each of the x, y , and z coordinates and we refer to the set containing the low-dimensional representation of the simulations as S_{proj} .

3. Since the experimental data in E is also in vertex-to-point correspondence to the reference simulation sim_{ref} we can project the experiment sample $exp \in E$, whose best matching simulation result we want to find, to the basis V_p of the low-dimensional space. We refer to the set containing the low-dimensional representation of the experiments as E_{proj} .
4. For the projected experiment sample exp_{proj} we can determine the best matching simulation by calculating the nearest neighbor of exp_{proj} in S_{proj} :

$$s^* = \arg \min_{s \in S_{proj}} \sum_{j=1}^p \|e_j - s_j\|_2, \quad (3)$$

where e_i and s_i are the respective 3 spectral coordinates α_j from (2) for the x, y, z functions for the experiment sample exp_{proj} or simulation sample from S_{proj} .

We refer to this approach, which selects the nearest neighbor of the projected experiment point cloud $exp_{proj} \in E_{proj}$ in the low-dimensional space containing the simulation data S_{proj} , as LBEA (Laplace Beltrami Eigen Alignment).

4.3 Runtime Comparison

The method LBEA has two components, for which the runtime can be determined separately. The first step includes the eigenvector calculation resulting in the basis V_p and the projection of the simulation data S to S_{proj} using the basis V_p . Afterward, the nearest neighbors in S_{proj} for the projected experimental data are determined.

For an alignment in the high-dimensional spaces S and E , the eigenvectors are not calculated, nevertheless, the nearest neighbor search of the method Vertex-to-Point (VtP) is more costly since the point clouds in S and E are of size n .

For the runtime of the nearest neighbor search in the approach VtP, let s be the cardinality of the set of simulations S . The best high-dimensional match to experiment sample $exp \in E$ is defined as the nearest neighbor of exp in the set of simulations S .

The runtime for the nearest neighbor search for the samples of size n is $s \cdot \mathcal{O}(n)$ for the distance calculation, which can be done in linear time, plus $\mathcal{O}(s)$ for finding the sample with minimum distance. Therefore, the total runtime for the approach VtP is

$$s \cdot \mathcal{O}(n) + \mathcal{O}(s)$$

for one nearest neighbor search in the high-dimensional space S .

If we reduce the dimensionality to $p \ll n$ by projecting to the basis V_p , the runtime for the approach LBEA reduces to

$$\mathcal{O}(n) + s \cdot \mathcal{O}(p) + \mathcal{O}(s)$$

for the projection to the basis V_p and one nearest neighbor search in the low-dimensional space S_{proj} . Computing the first p eigenvectors of the Laplace-Beltrami operator is done only once and achieved with complexity [21]

$$\mathcal{O}(p \cdot n^2 \log(n)).$$

Generally, the projection to the eigenvector basis V_p reduces the number of coordinates to less than 10% of the original number n . Therefore, especially if we have to compare to a high number of samples in S , the extra effort of computing the eigenvectors and the projection is easily compensated.

4.4 Postprocessing of the Alignment

Additionally, the approach LBEA allows us to jointly visualize the behavior of experiments and simulations for further data analysis. We consider the sets of the projected samples S_{proj} and

$$E_{proj} = \{exp_{proj} \mid \text{for all } exp \in E\}.$$

The low-dimensional embedding corresponds to the projection of the data to a small eigenvector-basis (up to 3 vectors to be able to visualize the result). We typically chose the three eigenvectors from the basis V_p with the highest variance computed for all the samples as proposed in [19].

In the low-dimensional embedding, we expect similar deformation patterns to be located in the same cluster. This way many coarse patterns in the deformation behavior of the simulations and experimental data can be visualized.

Non-observable physical functions for example plastic strain or stress cannot be measured during real experiments, whereas for finite element simulation these functions are calculated over time. Our alignment of simulations and experiments over time allows a projection of the simulation’s physical quantities to the experiment and gives an estimate of quantities, that cannot be measured. We refer to this projection of non-observable physical quantities from simulations to experimental results as indirect inference.

5 Results

We investigate the alignment between simulation results and experimental observations over time on synthetic and real experimental data, where the synthetic experimental observations are generated with a simulated light detection and ranging (LiDAR) sensor [4]. Both datasets contain simulation meshes (S_{obs}) and point clouds (E_{obs}) at different timesteps. To evaluate the results we have a detailed look at the best matches obtained from different approaches for the synthetic and real experimental data. We are comparing against a baseline method, which is based on the ICP (iterative closest point) algorithm. Additionally, we compare the runtimes and in particular analyze the joint low-dimensional visualizations that the LBEA approach provides.

5.1 Iterative Closest Point

Iterative closest point (ICP) is an algorithm that aligns two point sets [1, 5, 30]. It assumes that the two point clouds are similar and are sampled from the same object. In

comparison to the methods VtP and LBEA, the algorithm does not require an established correspondence between the sets, because itself iteratively creates this.

For a selected $exp \in E_{obs}$, we apply ICP to the pairs (exp, sim) for all $sim \in S'_{obs}$. From the meshes in S'_{obs} , we only consider the mesh’s vertex coordinates, because the ICP algorithm handles point clouds. The ICP algorithm iterates the following two steps to align the two point clouds exp and sim :

- (i) given a rigid transformation, find the closest point in sim for every point in exp
- (ii) given the correspondence calculated in (i), calculate the rigid transform between the two sets by minimizing the distances between the corresponding points. This is achieved by solving a least-squares problem.

This is iterated for a maximum of 30 iterations. The simulation sim with the smallest final error in step (ii) of the last iteration of ICP is considered the best match since the deformation described by the two point clouds is then similar.

Note that in step (i) of each iteration the point-wise distances between all points in both point clouds have to be calculated. The point clouds in E_{obs} contain between 16,000 and 62,000 points for the real experimental dataset and between 15,800 and 25,000 points for the synthetic experimental dataset.

It is possible to apply ICP to a random subsample of points from exp to reduce the runtime. But this means that the result is not deterministic anymore. Also, the quality of the registrations decreases with a decreasing number of subsampled points because of outliers. Therefore, down-sampling of the point clouds is not necessarily a solution as information is lost.

5.2 Measuring the Error

For the synthetic experimental data, we can compare the result of the alignment to the ground truth. To quantify this, we calculate an error between the simulation mesh $s^* \in S_{obs}$ that the algorithm identifies as the best match to the selected experiment $exp \in E_{obs}$, and the simulation mesh s^e corresponding to the synthetic LiDAR observation exp , i.e. using the original simulation meshes before the LiDAR processing. The error is the average Euclidean distance between the corresponding vertices of the two meshes in mm:

$$EE(s^e, s^*) = \frac{1}{n} \sum_{i=1}^n \|s_i^e - s_i^*\|_2, \quad (4)$$

where s_i^e and s_i^* are the 3D coordinates for vertex i of the respective experimental and simulation data.

For the real experimental data, the point clouds in E_{obs} have no corresponding simulation meshes. Therefore, we cannot calculate a Euclidean error to a ground truth mesh. The matching results of different alignment methods have to be compared visually. The visual comparison takes place not only in the high-dimensional space by comparing the experiment point clouds to the matched simulation meshes, but, in the case of the LBEA approach, also in the low-dimensional space.

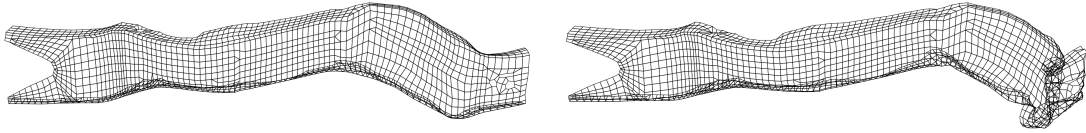


Figure 5: Simulation mesh of the left front beam of the TRUCK model at impact time in the undeformed state (left) and in a deformed state after approximately one-third of the simulation time.

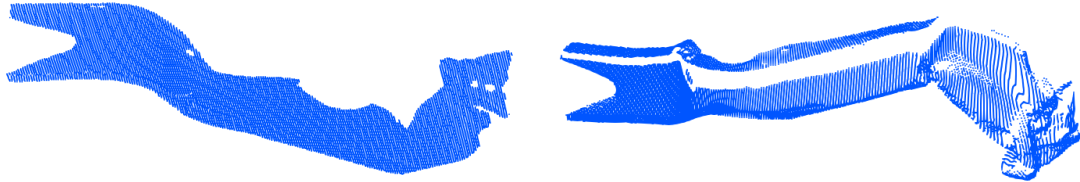


Figure 6: Simulated LiDAR observation of the left front beam (TRUCK dataset) from a laser. Left: the view of the simulated LiDAR laser. Right: View as in Figure 5. Notice that some areas are not scanned due to occlusion.

5.3 TRUCK - Synthetic Experimental Data

The simulation dataset S_{obs} contains simulation meshes of a left front beam that deforms over time, see Figure 5. We artificially create the synthetic experimental observations E_{obs} by applying a simulated LiDAR sensor [4] to 50 simulations, see Figure 6. Since the experimental data is created artificially based on simulation results, we can measure a true error and use (4) for that.

We use frontal crash simulations of a Chevrolet C2500 pick-up truck (from NCAC [28]), where material characteristics were changed, which is a setup similar to [2]. Each simulation result has 100 equally distributed timesteps, of which we chose every second timestep. As the impact time, we identify timestep $t = 19$, at which we start our alignment. For our analysis, we chose the left front beam. Figure 5 illustrates an undeformed beam at the impact time and a deformed beam. To create the two sets we randomly assigned 150 completed simulations to either S_{obs} or E_{obs} , such that S_{obs} contains 100 and E_{obs} 50 samples. This data will be made available after acceptance of the article.

For the TRUCK dataset, when calculating the vertex-to-point correspondence we consider both approaches for selecting sim_{ref} , because we know the respective timesteps for all experimental and simulation data:

1. sim_{ref} is one simulation over time since the experimental and simulation data are aligned over time. For every timestep, we then establish the correspondence to the corresponding timestep from sim_{ref} .
2. sim_{ref} is one reference simulation mesh at impact time $t = 19$.

For the three methods, Table 1 shows the Euclidean errors in mm between the iden-

Method	sim_{ref} at impact time		sim_{ref} over time		ICP*
	VtP	LBEA	VtP	LBEA	
Mean EE (mm)	10.01	9.71	9.01	8.64	13.13
Median EE (mm)	9.67	9.70	10.06	8.70	10.38
<i>Perfect match possible</i>					
Mean EE (mm)	4.03	3.79	2.69	2.46	-
Median EE (mm)	1.88	2.10	2.35	1.65	-

Table 1: On the top mean and median of the average Euclidean errors (EE), in mm, per experiment over time for the TRUCK dataset for unseen data. Since the experimental data is synthetic, we can calculate the distance between the matched simulation and the ground truth. Under *perfect match possible* we show results for experimental observations that are included in the simulation data.

*For the ICP based approach, the errors are calculated for only two experiments because of the high runtime.

tified best matches and the ground truth simulation meshes. We first average the Euclidean errors per experiment over time and then take the mean and median over the 50 experiments. The errors between best match and true mesh obtained with our approach LBEA are slightly lower compared to the error of the VtP approach. Additionally, the LBEA approach has a lower runtime. The ICP based approach is not able to match the selected experimental observations to the simulation meshes in a satisfying way.

Figure 7 visualizes where the ICP based approach fails in comparison to our alignment approaches VtP and LBEA. For the ICP alignments, also undeformed areas of the components are not well aligned, which leads to a higher alignment Euclidean error. The best matches from our approaches VtP and LBEA align well in most of the areas of the component. In some cases, the aligned mesh is tilted towards the end of the component.

The errors must be strictly larger than zero because the ground truth meshes of the experimental data E_{obs} are not included in the simulation data S_{obs} . Consequently, selecting the ground truth mesh, which would produce an error of zero, is not possible.

Therefore, we consider a second experiment dataset containing 25 experiments, where the underlying simulation meshes to the experimental observations are included in the simulation data, see Table 1 (Perfect match possible). If for all the samples the ground truth meshes were selected as the best match, the errors would be zero. Because the synthetic LiDAR observations miss occluded areas (Figure 6) and the samples are aligned in the low-dimensional space, not for every sample the best match is found. This suggests that more sophisticated preprocessing steps could further improve the results. Note, the results with our LBEA approach are still better compared to the VtP results.

When selecting a simulation sim_{ref} over time, the results are only slightly better in comparison to those when selecting an undeformed simulation mesh at the impact time. Especially the error in later timesteps is similar, as the curves in Figure 8 for a reference simulation sim_{ref} over time for a reference simulation at the impact time show.

If we select an undeformed simulation mesh at the impact time as sim_{ref} , we observe

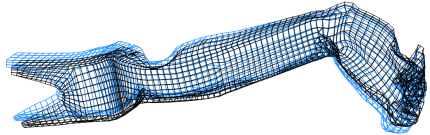
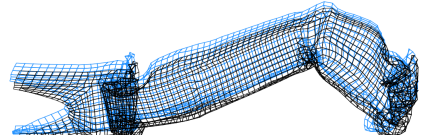
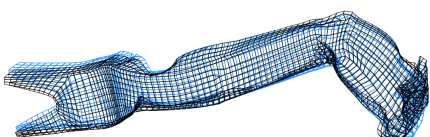
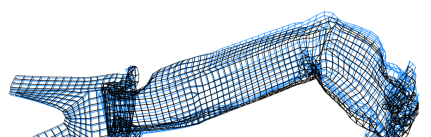
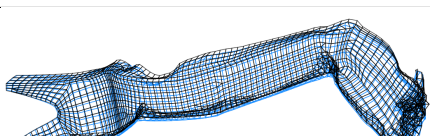
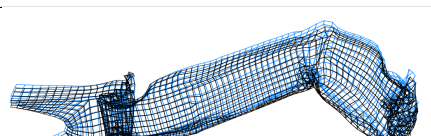
Method	Branch 1	Branch 2
ICP	 19.73 mm	 23.77 mm
VtP	 12.80 mm	 12.59 mm
LBEA	 9.90 mm	 12.95 mm

Figure 7: Best Matches at $t = 83$ using the approaches ICP, VtP, and LBEA (sim_{ref} over time) for two simulations from the TRUCK dataset that deform in different deformation patterns. The deformed meshes in black are the corresponding surface meshes to the LiDAR observations from E_{obs} and the aligned simulation meshes from S_{obs} are visualized in blue. The alignment Euclidean errors are given in mm.

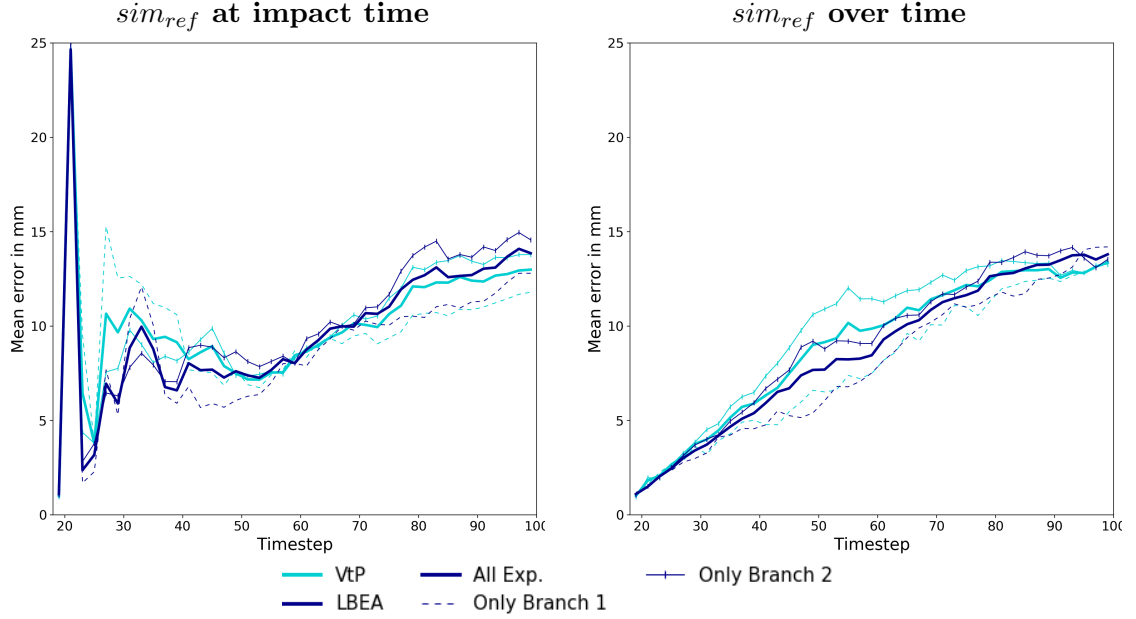


Figure 8: Euclidean errors (EE) in mm over time for the TRUCK dataset, when selecting a reference simulation sim_{ref} over time or at impact time for the two approaches VtP and LBEA. The errors are calculated for the two different branches corresponding to different deformation patterns. Note that the alignment error is lower for the pattern, in which the reference simulation deforms.

outliers for errors in early timesteps. This is due to the translation of the meshes, which is the highest in the early timesteps. If the meshes are matched to a slightly translated mesh, the deformation might be similar, but the vertex-wise errors increase because of the translation. For later timesteps, the two ways of selecting the reference simulation sim_{ref} lead to similar errors. This means, that if experimental observations and simulations are not aligned over time, the methods LBEA and VtP are still able to find a corresponding deformed simulation mesh to each experimental observation.

One can observe that the left front beam of the TRUCK deforms in two different branches [2, 15], depending on the material characteristics. The reference simulation sim_{ref} is from the second branch. Therefore, the alignment error for experiments from the second branch is also somewhat lower than for the other branch, see Figure 8.

5.4 C-STRUCTURE - Real Experimental Data

A generic crash structure was designed that shows deformation behavior typical during a crash test. It is inspired by the front part of a car and based on crash structures currently used in automotive engineering. During the crash, the kinetic energy is absorbed by buckling. This deformation is characterized by statistically strongly different deformation behavior during a front crash under identical test conditions due to different weld

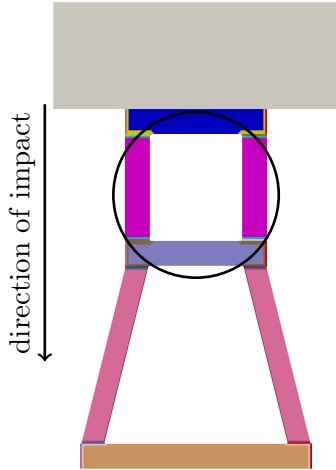


Figure 9: Generation of the real experimental data. The crash structure’s spot weld thickness is varied for the different experiments. The deformation in the upper part of the structure will be analyzed (the area that is marked by an ellipse). The arrow indicates the direction of the load.

seams or heat-affected zones. The crash structure was hand-welded out of aluminum.

The structure gets under load by an accelerated mass simulating a front crash, see Figure 9. The deformation behavior is observed in a real experiment using the GOBO 3D measurement method [16] and using numerical simulations. The buckling behavior can be observed in the beams, which are highlighted in Figure 9. The measurement data from the real experiment consists of a dense point cloud of 42 observed timesteps.

The structure and its characteristics are represented in a finite element mesh. Since the deformation behavior has proven to be dependent on the spot weld thickness, this parameter has been randomly chosen, creating different setups for the simulation. We refer to the resulting meshes as S_{obs} , which contains a mesh for 46 simulations at 61 timesteps. Note that the experiments and simulations are not synchronized along the time scale. While in the course of the original research several experiments were performed, we focus on one to illustrate our approach for comparing simulation and experiment.

As mentioned in section 5.2, the error between the best match in S_{obs} and the experiment $exp \in E_{obs}$ cannot be measured, because there is no ground truth for the real data. Therefore, we evaluate the quality of the approaches using visualizations of the results in Figure 10. For the selected examples we can observe that the ICP based approach seems to focus on the non-deformed areas of the beam, which leads to suboptimal alignment results. For example, the timesteps that are selected as the best match are from earlier in the simulation (see $t = 10, 13$), or the best match manifests a different deformation pattern (see $t = 10$).

At the same time, however, the alignment approach VtP after establishing the vertex-to-point correspondence and the LBEA approach in the low-dimensional space success-

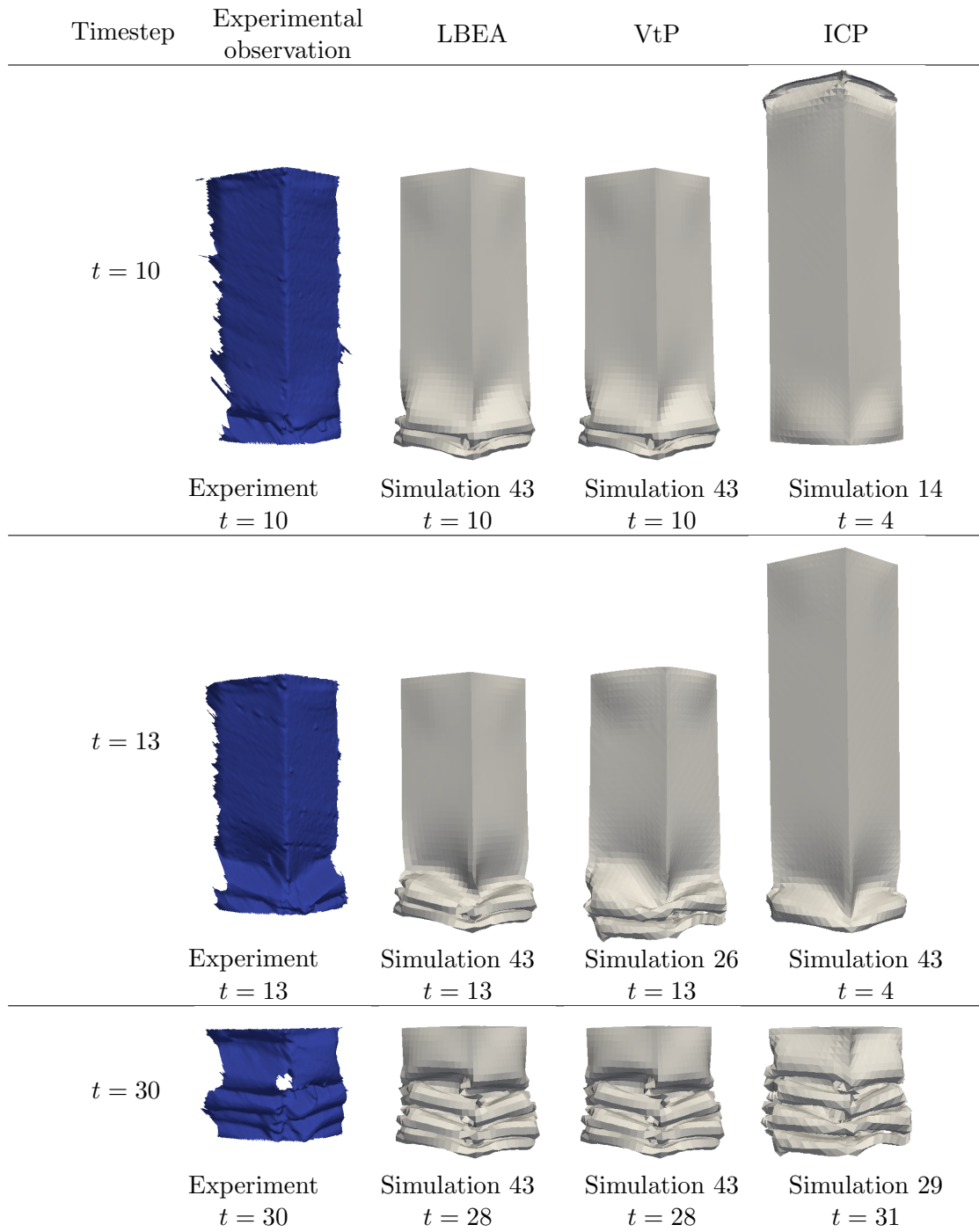


Figure 10: Visualization of the alignment using ICP, VtP, and LBEA on the real observed data E_{obs} from the C-STRUCTURE dataset. The alignment fails when using the ICP based approach, whereas the results from the LBEA and VtP are similarly good.

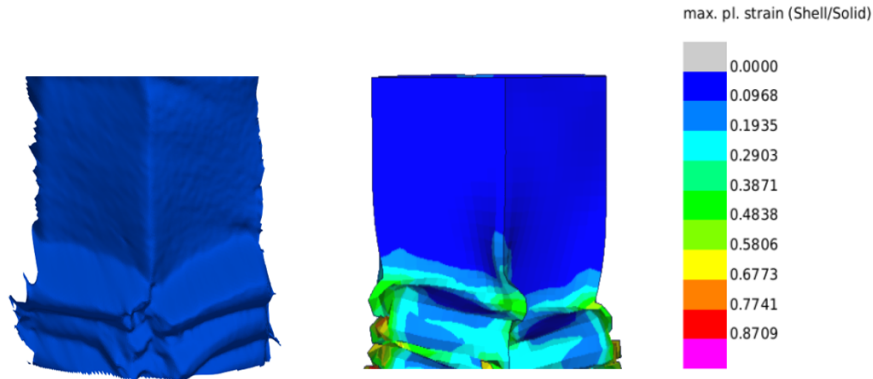


Figure 11: Indirect inference of physical quantities. Left: Point cloud of the real experiment. Right: The finite-element model of the test structure that best fits the experiment. We show the values of the plastic strain on the mesh.

fully align experiments and simulations given the location of the buckling on the beam. The results of the two methods VtP and LBEA are very similar. For the 42 timesteps of the experiment in E_{obs} , 25 are matched to the same simulation in S_{obs} . The differences in the matching are similar to the ones at $t = 13$ visualized in Figure 10.

As outlined, such an alignment to a simulation allows the indirect inference of quantities, available in the simulation, that cannot be measured in the experiment. This allows further insight into the experiment. As an example, Figure 11 shows the indirect inference of the maximal plastic strain from the simulation matching the experiment.

5.5 Runtime

In Table 2 the runtimes are specified for the three different alignment methods applied to both datasets. We separate the measurements into preprocessing and the actual alignment. The preprocessing is necessary for establishing the vertex-to-point correspondence for the approaches VtP and LBEA. Additionally, for the latter, the eigenbasis is calculated and the data is projected to the basis V_p .

Although the preprocessing takes additional time, the total runtime of preprocessing plus the alignment for the LBEA approach is reduced to approximately half in comparison to VtP. At the same time, the results are of the same quality as before the dimension reduction. The runtime of the alignment depends on the size of the data samples and this size is reduced from the number of vertices in the reference mesh sim_{ref} for VtP to the number of eigenvectors p for LBEA. Notice that the larger the datasets E and S , the bigger is also the runtime advantage of the low-dimensional comparison LBEA compared to VtP.

The calculation of the best matches applying the ICP based approach takes hours for

Data	Method	Preprocessing		Alignment	Total
		Correspondence to sim_{ref}	Calculation of and projection to basis V_p	Alignment of samples in E_{obs}	
C-STRUCTURE	ICP	-	-	> 10 h	> 10 h
	VtP	5 sec	-	17 sec	22 sec
	LBEA	5 sec	0.5 sec	5 sec	11 sec
TRUCK	ICP	-	-	> 40 days*	> 40 days*
	VtP	15 sec	-	1257 sec	21.2 min
	LBEA	15 sec	0.5 + 1 sec	496 sec	8.5 min

Table 2: Average runtimes for the three alignment methods for all experiments from the C-STRUCTURE (1 experiment and 46 simulations a 61 timesteps in E_{obs} and S_{obs}) and TRUCK (50 experiments and 100 simulations a 41 timesteps in E_{obs} and S_{obs}) datasets.

*: For the application of the ICP based approach to the TRUCK dataset, the runtime has been estimated based on the alignment of two experiments, because of the high runtime.

the C-STRUCTURE dataset. The application of ICP is not feasible for the full TRUCK dataset, the runtimes have been extrapolated based on two experiments for it to be more than a month.

5.6 Joint Low-dimensional Visualization of Simulations and Experiments

The alignment with the LBEA approach is based on a decomposition of the deformation behavior into elementary components. This allows a joint visualization of an experiment together with the simulations in a low-dimensional space, as shown in Figure 12. Each point of the 3D scatter plot represents a simulation at a certain time step. The points that are bigger in size correspond to the most similar mesh to the experimental data as calculated by the LBEA approach. In this way, experiment and simulation data are shown jointly.

As time advances, the points follow curves that start at the bottom and end at the top of the 3D scatter plot. We clearly observe two different patterns in the deformation behavior: the right side corresponds to deformations starting in the upper area of the beam, whereas the left side corresponds to those starting in the lower area. Furthermore, one can also observe that the deformation branch, to which the simulation belongs, depends on specific parameter ranges. Values of the weld thickness greater than two seem to correspond to simulations on the left, while values lower than two correspond to the ones on the right. This shows that the LB-based approach allows in addition to the alignment also an effective analysis of ensembles of numerical simulations, as originally proposed in [19], with or without experimental data.

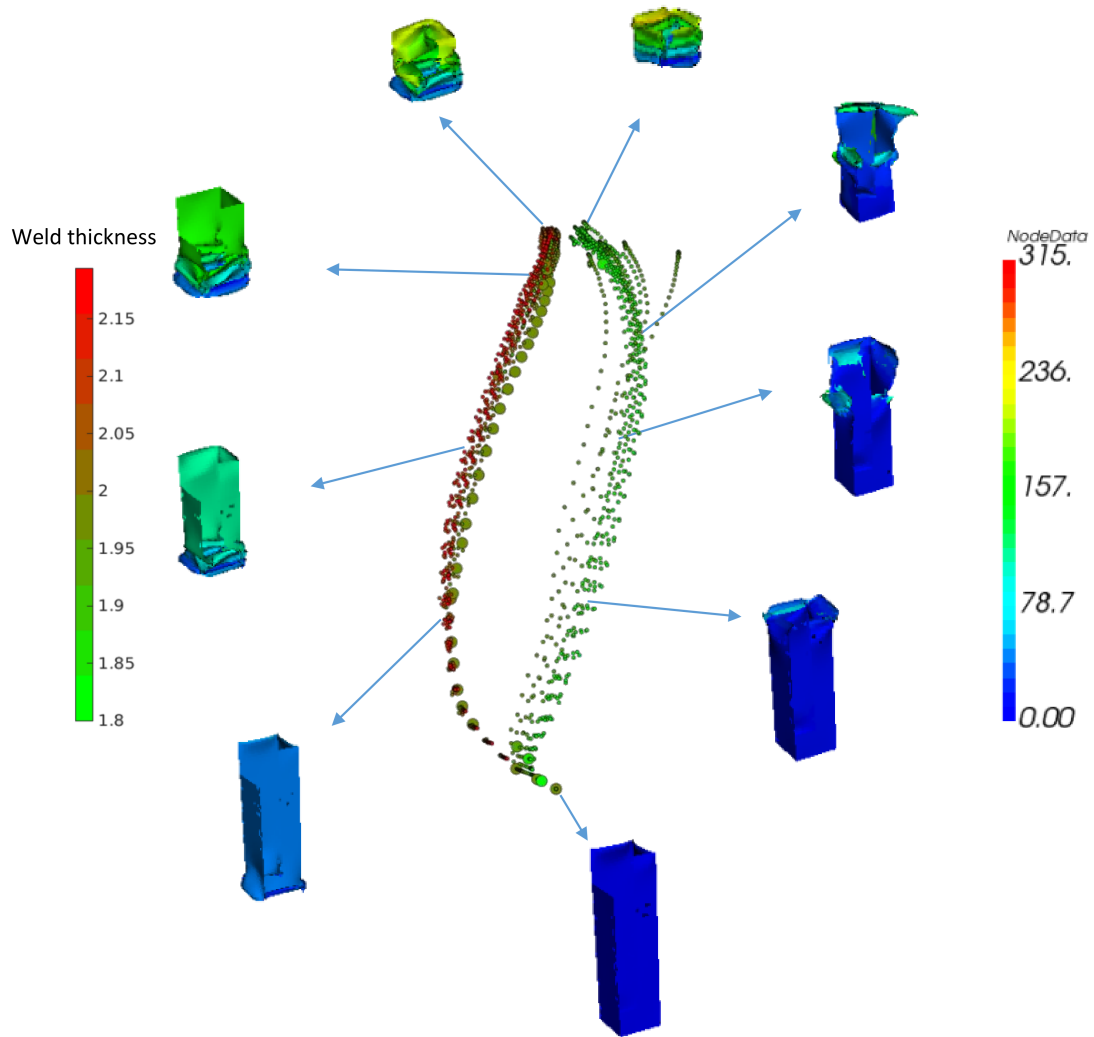


Figure 12: Embedding of 46 simulations at 60 timesteps and the matched results for the real crash experiment. We use the three Laplace-Beltrami coefficients with the highest variance, which are all for the coordinate function in the direction of the crash. The experimental data, marked by the bigger points, is following the left branch. The color of the points, range on the left, corresponds to the changed input parameter, which is the thickness of the welding. We exemplarily show the part that differs between the simulations. The color on the meshes, range on the right, represents the difference to the selected reference, shown at the bottom.

6 Conclusion

As we have seen, our novel Laplace Beltrami Eigen Alignment (LBEA) determines a correspondence between highly resolved optical experimental observations, stemming from a GOBO method [16], and a match from a simulation ensemble. In particular, it allows a joint comparison and visualization in a low-dimensional space. This results in an intuitive overview of the parameter-dependent behavior in a collision for experimental data and many simulation results. Usage of said methods for industrial applications promises very relevant results for the engineering practice.

In the considered examples the arising more global behavioral changes were investigated to show the potential of the approach. For future work, it would be interesting to explore up to which small scale and fine detail such matches are possible and what kind of insight into the experiment they could allow, e.g. by indirect inference. Different preprocessing approaches might be needed for this, which could also improve the overall quality of the alignment procedure. Furthermore, instead of looking for an existing simulation, one could aim to find the best possible simulation via the small LB eigenbasis, by means of an interpolation in the low-dimensional space. Here, one assumes that the LB eigenbasis represents the simulation manifold in a reasonable fashion and that the modelling of the component by the employed simulation mesh is suitable. This could allow an estimation of the gap between the best matching simulation and the best possible simulation. Future research could also explore the analysis of components that are not visible by a LiDAR sensor or the GOBO method, since they are not in the field of vision or covered by other components, which is another application of indirect inference.

Acknowledgements:

This work was developed in the Fraunhofer project “Horus - Hochgeschwindigkeits-3D-Messdatenerfassung zur Validierung von Experiment und Simulation in der Crashbewertung”. We thank our colleagues from the Fraunhofer institutes IOF and EMI for the experimental setup and data as well as the corresponding simulation model. We cordially thank Nikhil Prabakaran for initial experiments.

References

- [1] P. J. Besl and N. D. McKay. “Method for registration of 3-D shapes”. In: *Sensor Fusion IV: Control Paradigms and Data Structures*. SPIE, 1992, pp. 586–606. DOI: 10.1117/12.57955.
- [2] B. Bohn, J. Garcke, R. Iza-Teran, A. Paprotny, B. Peherstorfer, U. Schepsmeier, and C. A. Thole. “Analysis of car crash simulation data with nonlinear machine learning methods”. In: *Procedia Computer Science* 18 (2013), pp. 621–630. DOI: 10.1016/j.procs.2013.05.226.
- [3] M. Bojsen-Hansen, H. Li, and C. Wojtan. “Tracking surfaces with evolving topology.” In: *ACM Trans. Graph.* 31.4 (2012), pp. 53–1.

- [4] J. B. Campbell and R. H. Wynne. *Introduction to Remote Sensing*. Guilford Press, 2011.
- [5] Y. Chen and G. Medioni. “Object modelling by registration of multiple range images”. In: *Image and Vision Computing* 10.3 (1992), pp. 145–155. DOI: 10.1016/0262-8856(92)90066-C.
- [6] E. Demirci and A. R. Yildiz. “An experimental and numerical investigation of the effects of geometry and spot welds on the crashworthiness of vehicle thin-walled structures”. In: *Materialpruefung/Materials Testing* 60.6 (2018), pp. 553–561. DOI: 10.3139/120.111187.
- [7] C. Diez, L. Harzheim, and A. Schumacher. “Effiziente Wissensgenerierung zur Robustheitsuntersuchung von Fahrzeugstrukturen mittels Modellreduktion und Ähnlichkeitsanalyse Big Data in der Crashsimulation”. In: *VDI-Berichte 2279*. 2016.
- [8] C. Diez, P. Kunze, D. Toewe, C. Wieser, L. Harzheim, and A. Schumacher. “Big-Data based rule-finding for analysis of crash simulations”. In: *World Congress on Structural and Multidisciplinary Optimization*. June. 2017.
- [9] J. Dvořák, P. Vanecek, and L. Váša. “Towards Understanding Time Varying Triangle Meshes”. In: *International Conference on Computational Science (ICCS) 2021*. Springer, 2021, pp. 45–58.
- [10] J. Dvořák, Z. Káčereková, P. Vaněček, L. Hruďa, and L. Váša. “As-rigid-as-possible volume tracking for time-varying surfaces”. In: *Computers & Graphics* (2021). DOI: <https://doi.org/10.1016/j.cag.2021.10.015>.
- [11] M. S. Erickson, J. J. Bauer, and W. C. Hayes. “The Accuracy of Photo-Based Three-Dimensional Scanning for Collision Reconstruction Using 123D Catch”. In: *SAE Technical Paper*. SAE International, 2013. DOI: 10.4271/2013-01-0784.
- [12] J. Gall, B. Rosenhahn, S. Gehrig, and H.-P. Seidel. “Model-Based Motion Capture for Crash Test Video Analysis”. In: *Pattern Recognition*. Berlin, Heidelberg: Springer, 2008, pp. 92–101.
- [13] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
- [14] S. Hahner and J. Garcke. “Mesh Convolutional Autoencoder for Semi-Regular Meshes of Different Sizes”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2022, pp. 885–894.
- [15] S. Hahner, R. Iza-Teran, and J. Garcke. “Analysis and Prediction of Deforming 3D Shapes using Oriented Bounding Boxes and LSTM Autoencoders”. In: *Artificial Neural Networks and Machine Learning*. Springer, 2020, pp. 284–296.
- [16] S. Heist, P. Lutzke, I. Schmidt, P. Dietrich, P. Kühmstedt, A. Tünnermann, and G. Notni. “High-speed three-dimensional shape measurement using GOBO projection”. In: *Optics and Lasers in Engineering* 87 (2016), pp. 90–96. DOI: 10.1016/j.optlaseng.2016.02.017.

- [17] G. E. Hinton and R. R. Salakhutdinov. “Reducing the Dimensionality of Data with Neural Networks”. In: *Science* 313.5786 (2006), pp. 504–507. DOI: 10.1126/science.1127647.
- [18] R. Iza-Teran. “Enabling the analysis of finite element simulation bundles”. In: *International Journal for Uncertainty Quantification* 4.2 (2014), pp. 95–110. DOI: 10.1615/int.j.uncertaintyquantification.2013005436.
- [19] R. Iza-Teran and J. Garcke. “A Geometrical Method for Low-Dimensional Representations of Simulations”. In: *SIAM/ASA Journal on Uncertainty Quantification* 7.2 (2019), pp. 472–496.
- [20] D. Kracker, J. Garcke, and A. Schuhmacher. “Automatic analysis of crash simulations with dimensionality reduction algorithms such as PCA and t-SNE”. In: *16th International LS-DYNA Users Conference*. accepted. 2020.
- [21] J. Kuczyński and H. Woźniakowski. “Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with a Random Start”. In: *SIAM Journal on Matrix Analysis and Applications* 13.4 (1992), pp. 1094–1122. DOI: 10.1137/0613066.
- [22] J. A. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Information Science and Statistics. New York, NY: Springer New York, 2007.
- [23] J. Liu, N. Akhtar, and A. Mian. “Temporally Coherent Full 3D Mesh Human Pose Recovery from Monocular Video”. In: *arXiv:1906.00161* (2019).
- [24] R. V. McClenathan, S. S. Nakhla, R. W. McCoy, and C. C. Chou. “Use of photogrammetry in extracting 3d structural deformation/dummy occupant movement time history during vehicle crashes”. In: *SAE Transactions* (2005), pp. 736–742.
- [25] S. K. Mert, M. Demiral, M. Altin, E. Acar, and M. A. Güler. “Experimental and numerical investigation on the crashworthiness optimization of thin-walled aluminum tubes considering damage criteria”. In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 43.2 (2021), pp. 1–22. DOI: 10.1007/s40430-020-02793-6.
- [26] S. Mertler. “Comparative Analysis of Crash Simulation Results using Generative Nonlinear Dimensionality Reduction”. Dissertation. Universität Wuppertal, 2022.
- [27] M. Meywerk. *CAE-Methoden in der Fahrzeugtechnik*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. DOI: 10.1007/978-3-540-49867-4.
- [28] National Crash Analysis Center (NCAC). *Finite Element Model Archive*. URL: <http://web.archive.org/web/20160110143219/www.ncac.gwu.edu/vml/models.html> (visited on 01/10/2016).
- [29] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. J. Guibas. “Functional maps: a flexible representation of maps between shapes”. In: *ACM Trans. Graph.* 31.4 (2012), 30:1–30:11. DOI: 10.1145/2185520.2185526.
- [30] S. Rusinkiewicz and M. Levoy. “Efficient variants of the ICP algorithm”. In: *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM* (2001), pp. 145–152. DOI: 10.1109/IM.2001.924423.

- [31] A.-L. Schwartz. “Analysis of crash simulation data using spectral embedding with histogram distances”. In: *Informatik 2014*. Ed. by E. Plödereder, L. Grunske, E. Schneider, and D. Ull. Bonn: Gesellschaft für Informatik e.V., 2014, pp. 2449–2460.
- [32] P. Spethmann, C. Herstatt, and S. H. Thomke. “Crash simulation evolution and its impact on RD in the automotive applications”. In: *International Journal of Product Development* 8.3 (2009), pp. 291–305. DOI: 10.1504/IJPD.2009.024202.
- [33] G. K. Tam et al. “Registration of 3D Point Clouds and Meshes: A Survey from Rigid to Nonrigid”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.7 (2013), pp. 1199–1217. DOI: 10.1109/TVCG.2012.310.
- [34] J. Tang, D. Xu, K. Jia, and L. Zhang. “Learning Parallel Dense Correspondence from Spatio-Temporal Descriptors for Efficient and Robust 4D Reconstruction”. In: *Proceedings of CVPR*. 2021, pp. 6018–6027. DOI: 10.1109/CVPR46437.2021.00596.
- [35] D. Thanou, P. A. Chou, and P. Frossard. “Graph-based compression of dynamic 3D point cloud sequences”. In: *IEEE Transactions on Image Processing* 25.4 (2016), pp. 1765–1778. DOI: 10.1109/TIP.2016.2529506.