INTERNATIONAL DESIGN CONFERENCE – DESIGN 2022
https://doi.org/10.1017/pds.2022.178

A Machine Learning-Based Approach for Quick Evaluation of Live Simulations in Embodiment Design

C. Sauer [1,✉], B. Gerschütz [1], J. Bernsdorf [2], B. Schleich [1] and S. Wartzack [1]

[1] Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, [2] CADFEM GmbH, Germany

✉ sauer@mfk.fau.de

Abstract

Supporting product developers in early design phases with Live-Simulation can enhance the quality of early product designs. Live-Simulation can also facilitate a democratization of simulation and puts away pressure from simulation experts. In this paper, a machine learning based quick evaluation tool is proposed to support product developers in interpreting Live-Simulation results. The proposed tool enables a quick evaluation of the Live-Simulation results and enables product developers to further enhance their simulations. The tool is shown within a use case in bike rocker switch design.

Keywords: data-driven design, simulation-based design, structural analysis, machine learning, live-simulation

1. Introduction

One big trend in design is the democratization of simulation by giving more abilities to carry out simulations to the product developers. One aspect of this can be Live-Simulation, this means doing the simulations during or shortly after creating a new design variation of a product currently in design. This aspect enables Live-Simulation for product developers and puts away some work from simulation experts, as they get more sophisticated simulation models and fewer simulation requests from design departments.

Also, with rising graphical computing power, Live-Simulation can be carried out on the graphic processing units (GPU) directly on the hardware of the product developers (Abbey, 2019). Moreover, Live-Simulation is especially useful during the embodiment design phase because it enables the product developer to get a grasp on the physics of their product currently in design. However, user experience in simulation design and evaluation is often missing, so product developers struggle with creating correct and meaningful models that can be directly used for simulation (Kestel et al., 2019).

This paper presents an approach to support the product developers in carrying out their Live-Simulations via a proposed machine learning (ML) based quick evaluation tool, using existing finite element analysis (FEA) simulations and validated simulation results. The proposed evaluation tool therefore aims to evaluate the quality of each Live-Simulation. Moreover, ML in general also benefits heavily from the rising GPU power of the user.

The proposed ML-based quick evaluation tool combines the advantages of Live-Simulation with the knowledge of FEA simulations and validated simulation results, to further support the product developers. The goal of this support is to enable more sophisticated models for the product developers to hand them over to simulation experts and to reach a better level of communication in simulation and product development departments.

This paper first presents the related work in finite element analysis, Live-Simulation and ML. Then a focus is set on the research question and a concept for a ML-based quick evaluation tool of live

simulations is highlighted, before presenting a possible use case in mountain bike rocker switch design. The paper is rounded off by a discussion of the results and an outlook on further improvements of the presented approach.

2. Related Work

2.1. Finite-Element-Analysis

FEA is a numerical approach to solving engineering problems, via dividing solid or fluid bodies into a finite number of small elements (finite elements). Between those elements, small boundary conditions have to be applied. The sum of elements represents the behaviour of the whole model and the physics of the finite element model one wants to numerically predict. Foundations of FEA lie in the contributions of Courant (1942), Argyris (1955), Zienkiewicz (1977) and Ciarlet (1978).

FEA is currently one of the most used techniques in the realm of mechanical or fluid dynamical problems and allows predictions about the expected product behaviour in the real world. For example, the behaviour of a beam under load. On broader use case of FEA lies in the crash simulation of cars and car parts. The typical procedure of a FEA according to Vajna et al. (2018) can be the following. After specifying which predictions need to be made by the FEA, an adequate finite element model is produced. Product developers can use their off-the-shelf CAD software to create the geometry of real-world problems. Simulation engineers then enter a three-step process consisting of Preprocessing, Processing and Postprocessing. During Preprocessing, the geometry and product features are specified and boundary conditions, as well as forces, are applied. Lastly, the geometry gets discretized, resulting in a mesh. In the Processing step, the FEA is carried out by a solver, which numerically approximates the desired solution. Lastly, during Postprocessing, the results are projected onto the geometry of the finite element model. When a solution was found during Processing the technical job is finished and the job of the simulation experts is to interpret and assess the found solution, when there are no changes required an adequate solution was found.

Nowadays there exist several pieces of software for doing FEA, which cover either all three steps of a FEA or are useful for only one certain step in FEA. One piece of software is Ansys Mechanical in particular, which can be used to solve problems inside the mechanical realm (Alawadhi, 2009). In the realm of fluid dynamics on the other side, there exists different software like for example Flow-3D (Glatzel et. al, 2008).

2.2. Live-Simulation

In contrast to the well-known field of FEA, Live-Simulation is an approach to carry out simulations and get a prediction about resulting product behaviour directly when changing the geometry of the product. By harnessing GPU-based solver technology the simulation results get updated "on the fly" and mostly in sync with editing the geometry of the product. As it is a relatively new field of research, not many papers exist describing the general approach. However, Live-Simulation seems to be heavily influenced by mesh-free and voxelization techniques, also described by Zhou (2020).

Moreover, a very similar field besides Live-Simulation exists, which is called real-time finite element modelling. For example, Nikitin (2003) presented a system for the simulation of large elastic objects while using an offline inversion of the stiffness matrix. In real-time finite element modelling, linear elastic models are applied to mechanical structures. In Live-Simulation however, the goal is to incorporate not only linear elastic models. Therefore, meshfree and voxel-based methods are applied (Zhou, 2020). Live-Simulation allows faster results and faster predictions versus a "full-size" FEA by simulation engineers, with respect to the quality of the prediction. In the context of this contribution the term quality mainly refers to the aspect of prediction quaility regarding prediction errors. One can obtain faster results with the trade-off of a lower overall simulation prediction quality compared to the real-world behaviour of the product. Live-Simulation can be carried out for example by the software Ansys Discovery (Live) as shown by Fleischmann (2019). In this context Ansys Discovery can be seen as a mesh-free or voxel-based FEA solver.

Behind the background of putting simulation competencies into product developers hands, Live-Simulation has the potential to provide faster insights on the product for the product developers,

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

during the early phases of product design and mainly embodiment design and secondly reducing the load on the simulation engineers, for not having to carry out as many simulations anymore. Moreover, a higher maturity of the product and the simulation model can be reached before even carrying out a FEA.

2.3. Machine Learning (ML)

With rising amounts of data generated by finite element simulations and predictions about the overall product behaviour, a ML-based approach can be considered. For example, ML can be used on existing simulation results and allows fast design exploration and quick evaluation of designs based on previous simulation data. ML is a data-driven method that allows application inside the product development process (Vajna et al., 2018; Gerschütz, 2021). As mentioned above, ML can, besides other use cases, be used to predict product behaviour based on existing simulation results. The most common tasks for ML can be grouped into classification and regression tasks. For both tasks, a variety of different prediction models exists. The focus for this contribution lies on regression problems because this paper focusses on the prediction quality of a Live-Simulation with respect to a full-size FEA. ML can be divided into four methods, as shown in the following figure (Figure 1): Supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

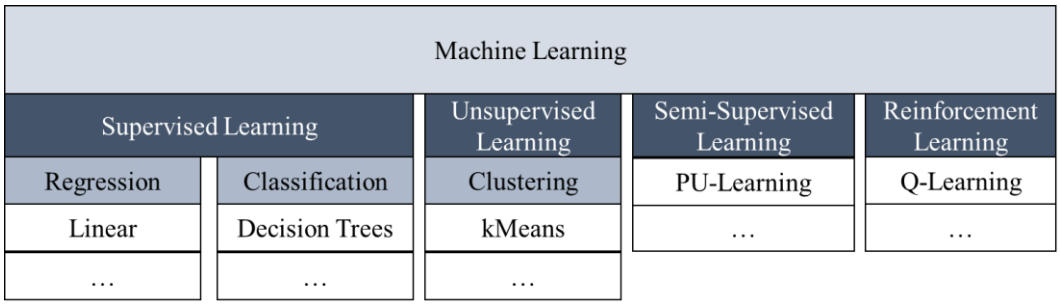

Figure 1. ML and its methods, tasks and algorithms

In Supervised Learning, the ML model is trained with labelled training data, therefore it is trained with previously known solutions. Unsupervised Learning focuses on unlabelled training data, for example when clustering unknown data. Semi-supervised Learning presents some sort of mixed procedure, where data gets labelled from the trained model. Lastly, Reinforcement Learning presents a novel approach where the machine learning model learns by exploring a design space based on a quality criterion of the underlying data.

Using ML inside product development focuses mainly on supervised learning methods, to be more specific regression or curve fitting tasks. For these ML tasks, one can start from data that describes a given input or feature set, such as the product features defined by product developers and attributed labels such as resulting product characteristics. ML mainly relies on numerical data in the form of n-dimensional arrays for regression-based supervised learning to first train and later predict via ML models. ML models therefore try to predict an input-output-correlation between input data, called features and output data, called targets. One ML basic model to highlight is a linear regression model with a varying polynomial degree. This model can predict very basic functional connections between the above-specified input and output data. Moreover, one more sophisticated algorithms used in modelling can be decision tree regressors (DTR). DTRs are piecewise-defined tree models consisting of many simple linear or polynomial functions as their leaves (Breimann, 1984). DTRs can also be called ensemble models. Decision trees can also be used for classification. The data for the models needs to be in the form of flat tables or tabular format, which also comes in handy, as FEA can produce big tables of resulting outputs. The prediction quality of ML models can be evaluated via quality criteria like the coefficient of prognosis (CoP) or the root-mean-squared-error (RMSE). According to the CoP is calculated with the following equation (Equation 1).

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

$$R^2CoP = \left(\frac{\sum_{i=1}^{n} (y_p - \bar{y}_p)(y_t - \bar{y}_t)}{(n-1)\sigma_p \sigma_t} \right)^2 \quad (1)$$

Predicted values from the ML model are arranged in a vector called yp. Its mean value is calculated and described by the barred yp. The ground truth target values of the samples used for testing the models quality are stored in the vector yt. Following the same scheme as the predicted values, barred yt represents the mean of the true target values. On the other hand, the standard deviations of prediction and the true target are described by the sigma variables. The total count of data samples is represented by the variable n. The main advantage of the calculated performance value is its scalability to the size of the data, by employing the total number of samples and calculating their performance with respect to their mean and standard deviation (Most T. and Will J., 2008). The RMSE on the other hand, is the root of the mean squared error between the predicted values of the model and the ground truth or test dataset.

Besides using ML for regression tasks, works of Sprügel et al. (2018) and Bickel et al. (2019) exist for plausibility checking of FEA simulations. Both use a Deep Learning Neural Network approach to check on existing simulation data if errors in the build-up of a finite element analysis (FEA) were made by the simulation engineers and if the results of the FEA seem plausible against stored results.

Moreover, new FEA simulations can be completely replaced by ML predictions when there is previous simulation data available as Martinez-Martinez et al. (2017) have shown. In their contribution three ML models were used to predict real-time biomechanical behaviour from FEA data, replacing new FEA simulations to acquire the resulting behaviour.

2.4. Summary of Related Work and Main Research Question

As mentioned above FEA provides meaningful predictions for product behaviour, Live-Simulation provides almost real-time results of possible product behaviour based on finite element methods and ML can be used to harness data, mainly produced by simulation models from both worlds FEA and Live-Simulation.

One aspect of Live-Simulation which was not stressed out enough before is that if product developers get a Live-Simulation result, no statement about the quality of this result is given. Live-Simulation solvers simply do not present any form of confidence interval about their predictions. So maybe ML can be used to get a statement about the quality of the Live-Simulation results, during the design and at the same time as the Live-Simulation is carried out?

Following this, the main research question in this paper is: Is there a way to equip product developers with a tool for getting a prediction about the quality of their Live-Simulation at the same time as they carry out the product design with the help of Live-Simulation?

As stated above, ML needs existing data for these types of predictions, but with rising amounts of simulation data, this problem can be addressed.

3. Concept for a ML-based quick evaluation of Live-Simulations

To answer the formulated research question above, a concept was needed. Some initial considerations around the structure of a Live-Simulation were made. It was found that on the Input side of a Live-Simulation product developers need to always defines parameters, boundary conditions and forces. Aside from the geometry of the product these are the most important things to consider. The Live-Simulation software then produces outputs in the form of deformations, stresses, and more FEA or CFD relevant output values. The proposed solution should receive those outputs and transfer them via a ML-approach into deviations between FEA and Live-Simulation, overall plausibility checks of the Live-Simulation results and further improvements to the Live-Simulation inputs and structure of the Live-Simulation. In the developed concept the ML-approach which brings together the initial considerations needs to facilitate an adequate architecture and needs to be model agnostic. Model agnostic means that later different ML models can be applied to the concept. The created concept is shown in the following figure (Figure 2).

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

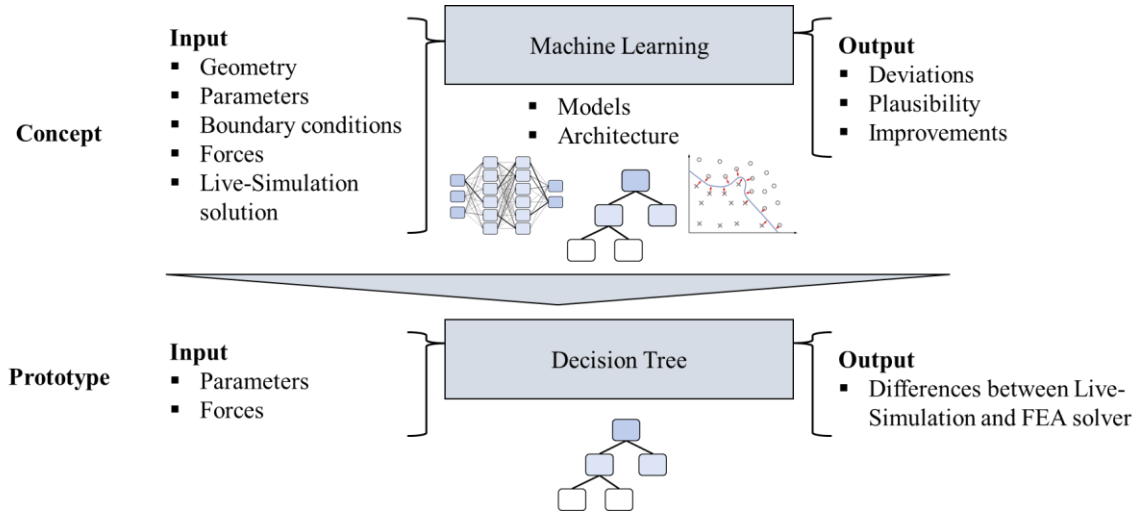

Figure 2. Concept and prototypical realisation for a ML-based quick evaluation of Live-Simulations

One aspect of the concept is that on the input side different Live-Simulation parameters need to be considered (as mentioned above). For example, a parametric CAD model of the geometry, different load cases, simulation boundary conditions and external forces on the products. The ML-based quick evaluation needs to also receive the solution of the Live-Simulation software to generate outputs. On the output side, the ML-based quick evaluation should be able to predict resulting deviations between the Live-Simulation and FEA solutions. Moreover, the quick evaluation should allow a plausibility check of the Live-Simulation result and automatically present possible improvements in the Live-Simulation to rise the resulting quality. In between Input and Output, the ML-based quick evaluation should provide the necessary architecture to facilitate an adequate prediction of the output parameters, based on the respective inputs described above. For this prediction, it should be possible to select from different machine learning models, e.g. the ones stated in the related work section above. To get from this very abstract concept to a more feasible approach a prototypical realisation was carried out. The different parameters of this realisation are shown in the lower part Figure 2.

For our prototypical realisation, the only inputs selected are the actual geometry of the product, mostly based on the geometric parameters inside the CAD and the forces applied to the respective product. On the output side, the existing differences between FEA and Live-Simulation get aggregated. These differences result from stresses or deformations predicted for sooner product designs and can support product designers, especially in the embodiment design phase.

To facilitate the ML side, input and output data needs to be of a certain format and form. The different data types and structures are shown in the table below (Table 1).

Table 1. Data types and structures needed

Parameter	Data type	Data structure
Input - geometry	CAD parameters	Float Vector or Matrix
Input - forces	Live-Simulation parameters	Float Vector or Matrix
Input - Accuracies	FEA/Live-Simulation accuracy	Float number
Output - stresses	FEA/Live-Simulation result	Float number
Output - deformations	FEA/Live-Simulation result	Float number

Resulting from this, one can see that the data structures needed are always in a vectorial or matrix-like form. To build up training data for the ML-based quick evaluation the storage of the in- and outputs inside flat table format is advised. Because a supervised learning approach is selected, the data points need to always contain the inputs alongside their respective outputs on which the ML model is then trained (Figure 3).

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

Bar width in mm	Distance hole to frame	Solver accuracy	Element size of FEM	Deviatons deformations	Deviations stresses
6	10	0,25	10	-0,000180223	-4,94E+08
6	11	0,25	10	1,27924E-05	-3,12E+08
6	12	0,25	10	-2,49183E-05	-3,98E+08
6	13	0,25	10	-4,45504E-05	-4,23E+08
Geometry		**Accuracies**		**Deviations**	

Figure 3. Tabular structure for ML model training

To create such a flat table the scripting interfaces of the different tools can be used. Every major FEA or Live-Simulation software has some scripting capability, which allows users to automatically change a parametric simulation model and later extract solutions found for this model. In our protype case we used the programming language Python and the tools provided by Ansys Mechanical and Discovery software.

Once the flat-table data was created, ML models need to be trained on the dataset. This can also happen via Python and the ML library scikit-learn (Pedregosa, 2011). Moreover scikit-learn allows via a ML model API the creation and addition of custom ML models to the prototype. The training of the ML model can be carried out via a train-test-split. This means the dataset is split into a training subset and a test subset, the test subset of the data is not used during training and can be used to validate the prediction quality of the model after training.

Prediction quality is not only interesting for Live-Simulation results. In our case, the ML-based quick evaluation models also need to be checked for their prediction quality of resulting deviations between the Live-Simulations and FEA. For the initial use case those deviations were calculated via simple subtraction of the maximum values for deformation and Von-Mises-Stress found either from Live-Simulation or FEA. To check the prediction quality of those resulting deviations a model quality statement needs to be introduced, so for every prediction, the ML-based quick evaluation makes, a model quality statement is added. Users can easily see how certain their quick evaluations are with the guess on the resulting deviations.

The presented concept and the prototypical realisation are now completed and in the next section a possible use case in mountain biker rocker switch design is shown and the prototypical realisation is implemented for this specific use case.

4. Use Case

This section is to show a prototypical implementation of our concept based on the mountain bike rocker switch design. This part can be used in full-suspension mountain bikes for facilitating the suspension of the rear wheel. In the use case force is always applied from the rear wheel side. The other holes of the product are fixed in Y-direction and the other side is fixed against the suspension. The use case rocker switch is shown in the figure below (Figure 4).

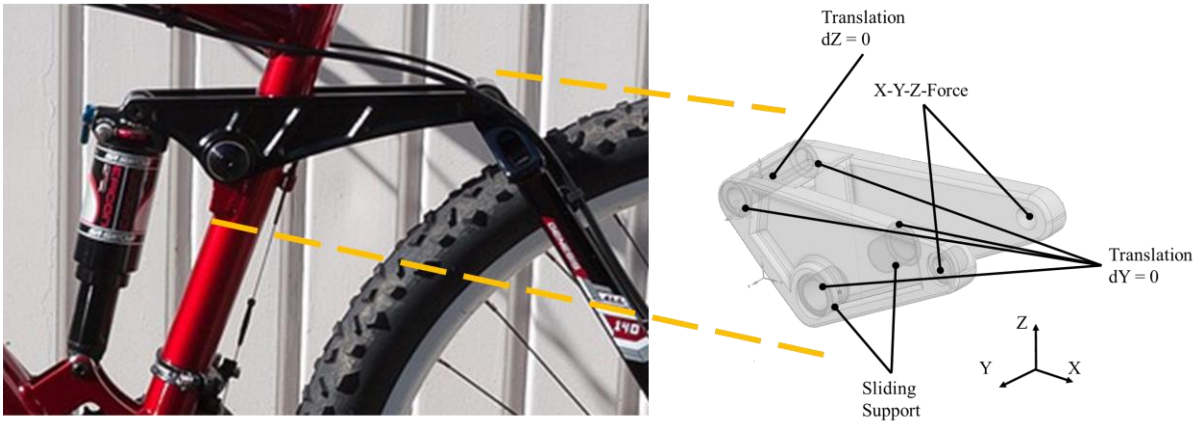

Figure 4. Parametric CAD model of the use case mountain bike rocker switch

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

In the figure above (Figure 4) one can see the parametric CAD model of our mountain bike rocker switch. The CAD model is fully parametrized; therefore, the product developers can design their mountain bike rocker switch by simply accessing the CAD parameters. Moreover, different boundary conditions are applied to the model. The translation in Z-direction is fixed in the top hole of the rocker switch. The translation in Y-direction is fixed in the top hole as well as in the bottom holes. In the bigger bottom hole also a sliding support was added. On the rocker side, a 3D force is applied (X-Y-Z force).

To get an initial dataset an initial set of FEA simulations was carried out in Ansys Mechanical and around 20 design points were created. Those design points were created via a full-factorial design of experiments. The same models were also run inside Ansys Discovery, as an example for Live-Simulation software. The resulting 20 design points were validated, and plausibility checked by a simulation expert either for Ansys Mechanical and Ansys Discovery. From these 20 design points, an initial dataset was created. This dataset holds the following eight parameters (Table 2).

Table 2. Parameters of the initial dataset

Parameter	Input/output parameter	Type and unit
Bar width of the rocker switch	Geometrical input	Float number in mm
Distance hole to frame	Geometrical input	Float number in mm
Solver accuracy of Live-Simulation	Live-Simulation input	Float number in between 0.0 - 1.0
Maximum deformation Live-Simulation	Live-Simulation result	Float number in m
Maximum Von-Mises-Stress Live-Sim.	Live-Simulation result	Float number in Pa
Element size of FEA	FEA input	Float number in mm
Maximum deformation FEA	FEA result	Float number in m
Maximum Von-Mises-Stress FEA	FEA result	Float number in MPa

One can see that two geometrical inputs from CAD were used, which were directly extracted from the CAD system (in this case PTC Creo). The current prototype was also focused on parametric CAD models with a fixed set of parameters. For future use cases it is thinkable to use a Deep Learning approach to facilitate predictions for similar geometric shapes.

From the Live-Simulation software, the solver accuracy was considered. This is an Ansys Discovery specific value for Live-Simulation to control the speed of the Live-Simulation. For example, one can select very low accuracy to get near-real-time results and very high accuracy to get near FEA simulation time and high accuracy results. On the FEA side, the element size of the Ansys Mechanical solution was considered. Results from Live-Simulation and FEA were collected for the maximum deformations and the maximum Von-Mises-Stresses of the model. It is worth noting, those different units must be considered, for example, the maximum Von-Mises-Stress of the FEA solver was in MPa, the maximum Von-Mises-Stress of the Live-Simulation was in Pa. Both results get aggregated to calculate a difference between resulting deformations and Von-Mises-Stresses between FEA and Live-Simulation. It is worth noting that the prototype focuses on static FEA simulations, as the dynamic ones are more difficult to predict with state of the art Live-Simulation tools.

Based on this initial dataset a decision tree regression model was trained. The decision tree ML models were selected because of their prediction quality for this type of tabular data structure (Kim, 2008). The model was trained with a train-test-split of 70-30 and later evaluated via RMSE and CoP values being inside specified quality criteria. For example, the model quality for the deformations got a CoP of around 95%, the model quality for the Von-Mises-Stress got a CoP of around 91%. The resulting software prototype is shown in the following figure (Figure 5).

The resulting software prototype is running alongside Ansys Discovery and can be accessed via a user-generated function inside the user interface. On every design change, an automatic quick evaluation of the resulting Live-Simulation result is made. Product developers then receive a prediction of how far off their Live-Simulation result is from a possible FEA solution. Moreover, the model quality of the ML is also presented to the user and shows feasible results for our use case

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

approach. This finishes our use case, and the next section is to further discuss our results and show the actual strengths and weaknesses of our presented approach.

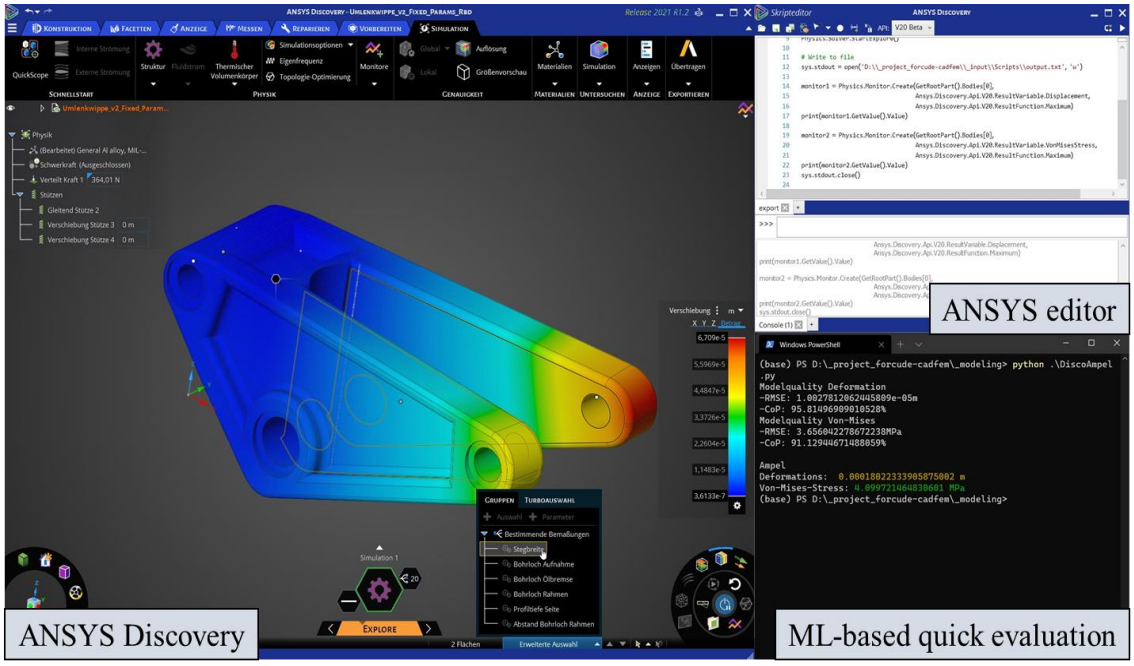

Figure 5. Resulting software prototype with Ansys Discovery and quick evaluation tool running alongside

5. Discussion

As stated above resulting CoP values for the trained decision tree regression models look sufficient, as they are both above 90%, same applies to resulting RSME values. Thus one can trust our trained models in predicting the differences between the Live-Simulation displayed to the user and the FEA simulation run before. In getting such high CoP values there was also no need for us to further investigate different ML models to do the task, as they were good enough. However, the prototype implementation uses a standard scikit-learn API for different ML models, so one can add ML models using this API without any further need for implementations (Pedregosa et al., 2011). Product developers can use the model quality prediction of the ML quick check to validate their Live-Simulation results. This means for example the solution of the Live-Simulation presents a positive output, but the ML-based quick evaluation predicts a low Live-Simulation result quality. Product developers then have the option to refine their simulation. So, the ML-based quick evaluation can for example provide a way to counteract under sizing of specific product areas.

One aspect of this implementation is the model complexity of our ML-based quick evaluation. If more inputs are introduced and different types of inputs for example the boundary conditions are added, other types of models needed to be considered. Moreover, the prototype DTR needs to be retrained for a new geometry or parameter set. When switching from parametric models to a geometric approach, Deep Learning methods could be applied to predict results for similar geometries. For example, different models that can handle more and different types of inputs (e.g. Deep Learning models).

Another weakness of the presented approach is the need for an initial set of FEA simulations for the specific use case, in our case the mountain bike rocker switch design. This weakness is tackled via newer and more sophisticated training data in the future.

On the other hand, one strength of the presented approach is that in the first studies the ML-based quick evaluation together with the Live-Simulation is better than a simpler, in terms of simply training the model based on the existing FEA data, ML-based prediction of the FEA results without using any type of Live-Simulation as shown in the figure below (Figure 6).

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

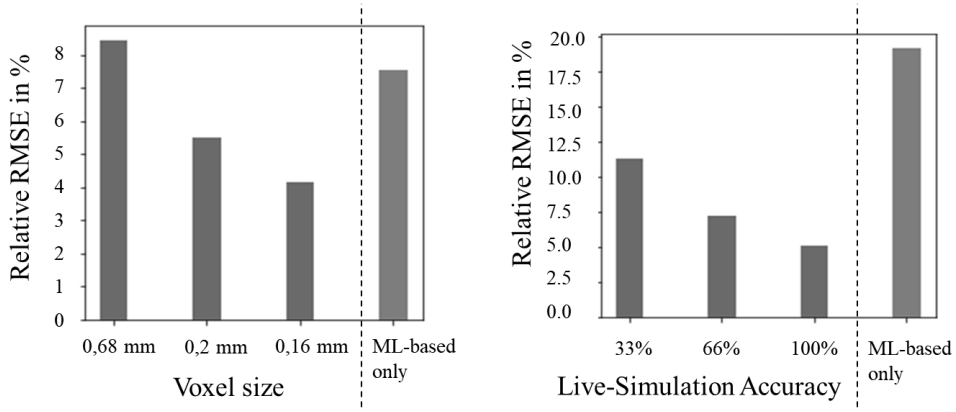

Figure 6. ML-based quick evaluation of Live-Simulation vs. ML-based prediction of FEA results

One can see that in every case and for every Live-Simulation accuracy tested with quick evaluation provides better or near ML-based (for the 0,68mm voxel size approach) prediction quality than a solo ML-based prediction of the results based on the FEA results. Even when changing the Live-Simulation solver accuracy this statement holds. This means that the presented approach of a ML-based quick evaluation can support the product developers during their design and when using Live-Simulation in early phases of design, like the embodiment design.

This answers the initial research question of this paper. Moreover, it even exceeds the initial goal of supporting product developers because it presents a viable tool to enhance Live-Simulation prediction quality.

6. Summary and Outlook

To put it in a nutshell, this contribution presents an approach for a ML-based quick evaluation of Live-Simulation results. For this, a concept for quick evaluating Live-Simulation results was introduced, consisting of an Input-Model-Output declaration. The concept uses ML to train models on available data. The concept was then implemented in a prototypical implementation and a use case inside the mountain bike rocker switch design was given. The first prediction quality studies look very promising and show potential beyond quick checking, this means enhancing Live-Simulation via our ML-based approach.

The next steps include the addition of more input parameters, such as the simulation structure and included boundary conditions and loads. The database also needs further improvement and more and different types of simulation models should be added. The presented system is applicable to any domain with recent simulation solutions and can be applied to different domains, such as computational fluid dynamics (CFD).

Acknowledgements

This research work is part of “FORCuDE@BEV - Bavarian research association for customized digital engineering for bavarian SME's“ and is funded by the “Bayerische Forschungsstiftung (BFS)”. The authors are responsible for the content of this publication. Special thanks are directed to the Bayerische Forschungsstiftung (BFS) for financial support of the whole research project. The authors would also like to thank Ansys in supporting us with their latest Ansys Software release and continued technical support by the CADFEM GmbH.

References

Abbey, T. (2019), *Meshless FEA Opportunities. [online] Digital Engineering 247. Available at:* http://web.archive.org/web/20211021121117/https://www.digitalengineering247.com/article/meshless-fea-opportunities/simulate *(accessed 21.10.2021)*.

Alawadhi, E. M., 2009, *Finite Element Simulations Using Ansys*, CRC Press, Boca Raton, Florida, USA. https://doi.org/10.1201/9781439801611

Argyris, J. H. (1955), "Energy theorems and structural analysis", *Aircraft Engineering, Vol.* 27, pp. 125-144.

Bickel, S., Sprügel, T., Schleich, B., Wartzack, S. (2019), "How Do Digital Engineering and Included AI Based Assistance Tools Change the Product Development Process and the Involved Engineers", *Proceedings of the 22nd International Conference on Engineering Design* (ICED19), Delft, NL: Cambridge, United Kingdom: Cambridge University Press, pp. 2567-2576.

https://doi.org/10.1017/pds.2022.178 Published online by Cambridge University Press

Breiman, L., Friedman, J., Olshen, R., Stone, C. (1984) *Classification and Regression Trees*, Wadsworth, Belmont, CA, 1984.
Ciarlet, P. (1978), *The Finite Element Method for Elliptic Problems*, North Holland, New York, USA.
Courant, R. (1942), "Variational methods for the solution of problems of equilibrium and vibrations", *Bulletin American Mathematical Society, Vol.* 69, pp. 1-23.
Fleischmann, C., Leher, I., Hartwich, R., Hainke, M., Sesselmann, S., (2019), "A new approach to quickly edit geometries and estimate stresses and displacements of implants in real-time", *Current Directions in Biomedical Engineering*, Vol. 5, pp. 553-556. https://doi.org/10.1515/cdbme-2019-0139
Gerschütz, B.; Sauer, C., Wallisch, A., Mehlstäubl, J., Kormann, A. Schleich, B., Alber-Laukant, B., Paetzold, K., Rieg, F., Wartzack, S. (2021), "Towards Customized Digital Engineering: Herausforderungen und Potentiale bei der Anpassung von Digital Engineering Methoden für den Produktentwicklungsprozess", *Stuttgarter Symposium für Produktentwicklung SSP* 2021, Stuttgart, Deutschland, pp. 93-104. https://doi.org/10.18419/opus-11478
Glatzel, T., Litterst, C., Cupelli, C., Lindemann, T., Moosmann, C., Niekrawietz, R., Streule, W., Zengerle, R., Koltay, P., 2008, "Computational fluid dynamics (CFD) software tools for microfluidic applications – A case study", *Computers & Fluids*, *Vol* 37, pp. 218-235. https://doi.org/10.1016/j.compfluid.2007.07.014
Kestel, P., Kügler, P., Zirngibl, C., Schleich, B., Wartzack, S. (2019), "Ontology-based approach for the provision of simulation knowledge acquired by Data and Text Mining processes", *Advanced Engineering Informatics*, Vol. 39, pp. 292-305. https://doi.org/10.1016/j.aei.2019.02.001
Kim, Y. S. (2008), "Comparison of the decision tree, artificial neural network, and linear regression methods based on the number and types of independent variables and sample size", *Expert Systems with Applications*, Vol. 34(2), pp. 1227-1234. https://doi.org/10.1016/j.eswa.2006.12.017
Martínez-Martínez, F., et al. (2017) “A Finite Element-Based Machine Learning Approach for Modeling the Mechanical Behavior of the Breast Tissues under Compression in Real-Time.” Computers in Biology and Medicine, vol. 90, Nov. 2017, pp. 116–124. https://doi.org/10.1016/j.compbiomed.2017.09.019
Most, T., Will, J. (2008), Metamodel of Optimal Prognosis - an automatic approach for variable reduction and optimal metamodel selection, Proceedings Weimarer Optimierungs- und Stochastiktage 5 (2008): 20-21
Nikitin, I., Nikitina, L., Frolov, P., Goebbels, G., Göbel, M., Klimenko, S., Nielson, G. M. (2003), Real-time simulation of elastic objects in virtual environments using finite element method and pre-computed Green's functions, *Исследовано в России*, Vol 6.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E. (2011), "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research*, Vol. 12, pp. 2825-2830.
Sprügel, T., Rothfelder, R., Bickel, S., Grauf, A., Sauer, C., Schleich, B., Wartzack, S. (2018), "Methodology for plausibility checking of structural mechanics simulations using Deep Learning on existing simulation data", *Proceedings of NordDesign* 2018. Linköping, SE: The Design Society.
Vajna, S.,Weber, C., Zeman, K., Hehenberger, P., Gerhard, D., Wartzack, S. (2018) *CAx für Ingenieure*, 3 Ed., Springer, Berlin/Heidelberg. https://doi.org/10.1007/978-3-662-54624-6
Zhou, Y., Lu, H., Wang, G., Wang, J., Li, W., (2020). "Voxelization modelling based finite element simulation and process parameter optimization for Fused Filament Fabrication", *Materials & Design*, Vol 187, 108409. https://doi.org/10.1016/j.matdes.2019.108409
Zienkiewicz, O. C. (1977), *The Finite Element Method*, McGraw-Hill, New York, USA.