Door Inner and Hinge Reinforcement Multi-Stage Assembly – Case Study

Version 0.0.1, 2020-01-13

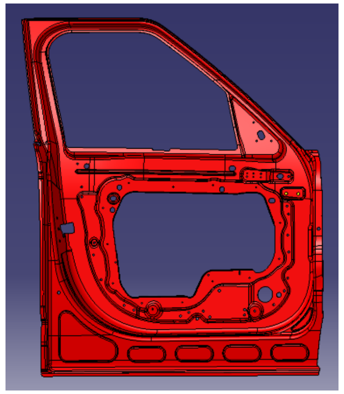
[Digital Lifecycle Management (DLM) Research Group]

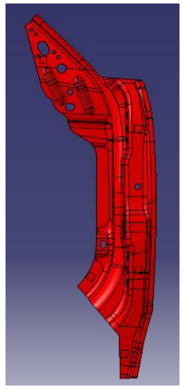
S. Sinha, E. Glorieux, P. Franciosa, D Ceglarek

CASE STUDY - OVERVIEW

Aim

- Study deformation patterns due to locator movement during the positioning, clamping, fastening and release steps for assembly of door inner and hinge reinforcement
- Perform Deep Learning using 3D CNN* on the deformation pattern data** to estimate process variations causing these deformation patterns
- The Two part assembly has a combined of 10841 Nodes


Door Inner


Hinge Reinforcement

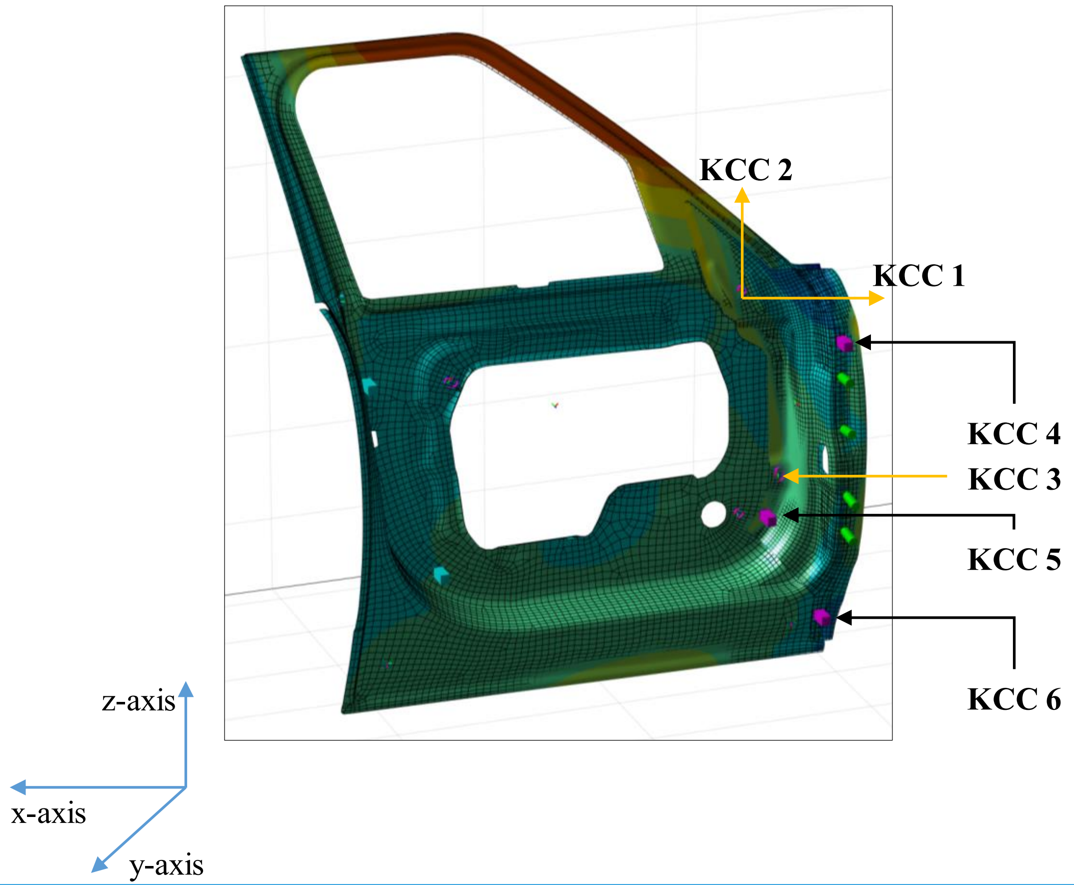
- Deviation patterns generated based on process parameter variations
- Different combination and magnitude of KCC variations give different patterns
- The deviation patterns are generated using VRM software

*Sinha, S., Glorieux, E., Franciosa, P., & Ceglarek, D. (2019). 3D convolutional neural networks to estimate assembly process parameters using 3D point-clouds (p. 10). SPIE-Intl Soc Optical Eng. https://doi.org/10.1117/12.2526062

** Franciosa, Pasquale, et al. "A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmarking and comparative study." Finite Elements in Analysis and Design 166 (2019): 103319.

***Process Parameters are termed as Key Control Characteristics (KCCs) in a manufacturing assembly setup

CASE STUDY – PROCESS PARAMETERS (KCCs)

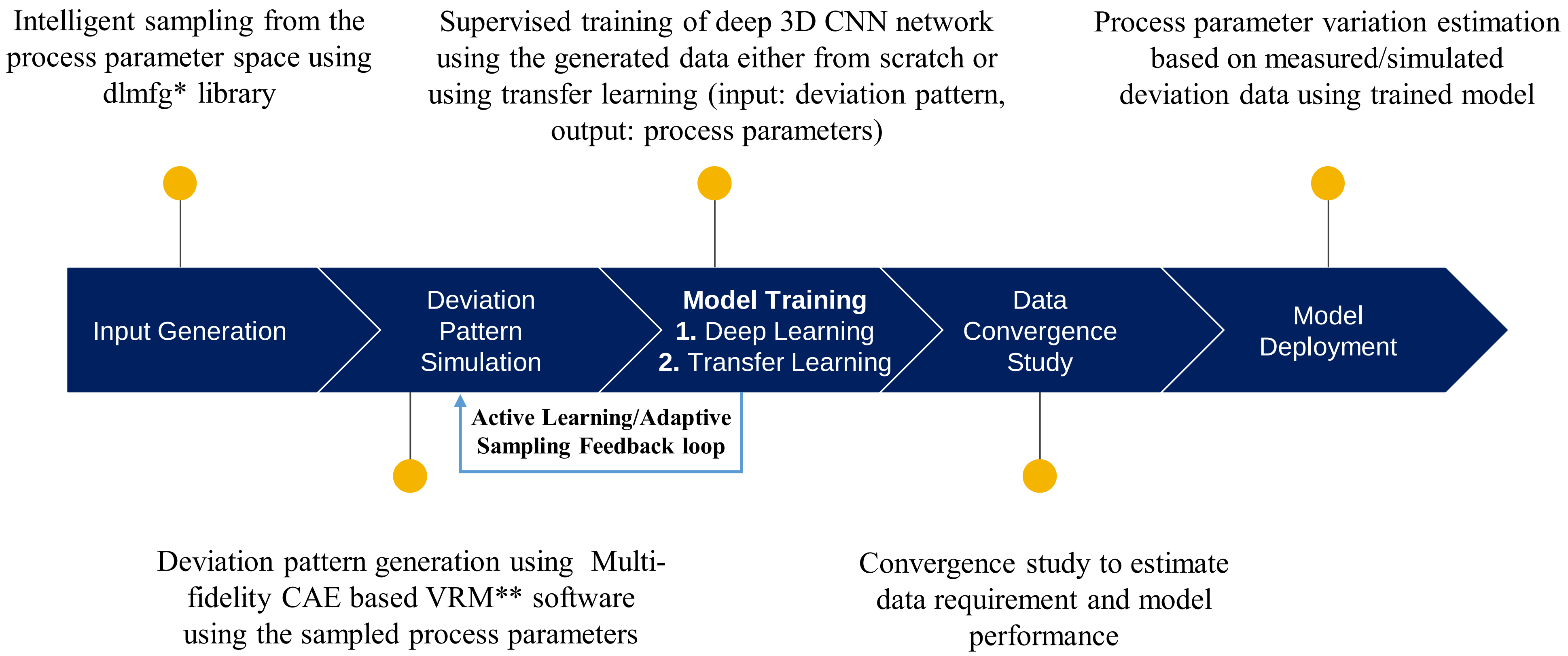


Process Parameters (KCCs)

Process Parameter*	Description	Unit	Range
KCC 1	Rotation around the pinhole	degree	[-1,1]
KCC 2	Pin hole displacement in x	mm	[-1,1]
KCC 3	Pin hole displacement in z	mm	[-1,1]
KCC 4	Clamp 1 displacement in y	mm	[-2,2]
KCC 5	Clamp 2 displacement in y	mm	[-2,2]
KCC 6	Clamp 3 displacement in y	mm	[-2,2]

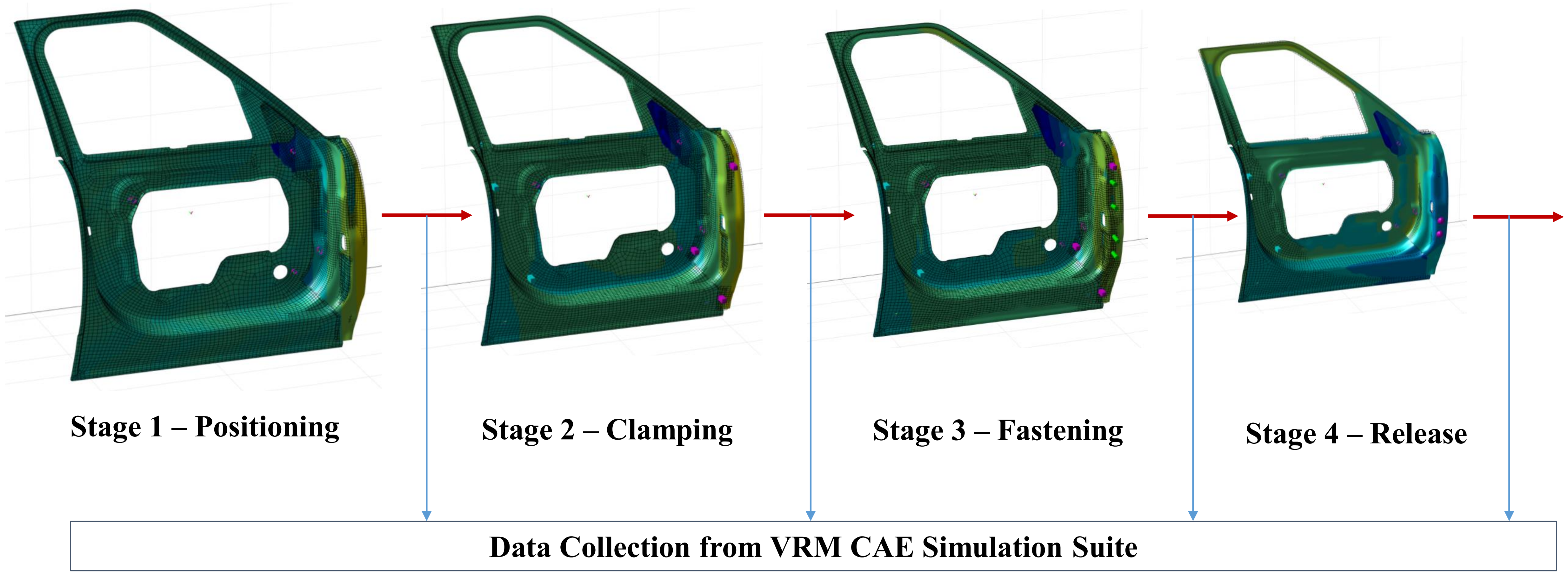
**Process Parameters are termed as Key Control Characteristics (KCCs) in a manufacturing assembly setup*

CASE STUDY – DATA GENERATION & LEARNING WORKFLOW


Intelligent sampling from the process parameter space using dlmfg* library
Supervised training of deep 3D CNN network using the generated data either from scratch or using transfer learning (input: deviation pattern, output: process parameters)
Process parameter variation estimation based on measured/simulated deviation data using trained model
Input Generation
Deviation Pattern Simulation
Model Training
1. Deep Learning
2. Transfer Learning
Data Convergence Study
Model Deployment
Active Learning/Adaptive Sampling Feedback loop
Deviation pattern generation using Multi-fidelity CAE based VRM** software using the sampled process parameters
Convergence study to estimate data requirement and model performance

- **dlmfg code and software link: https://github.com/sumitsinha/Deep_Learning_for_Manufacturing*
- *** VRM code and software link:*

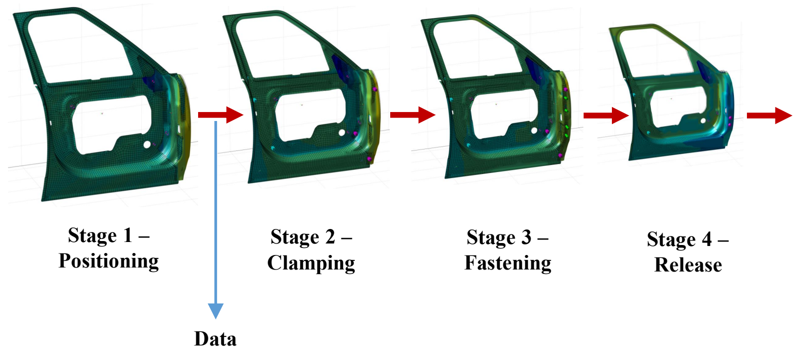
Case Study – Assembly Sequence

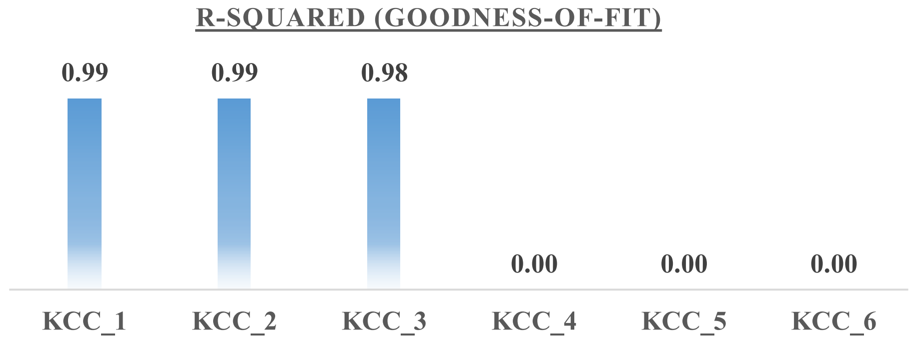


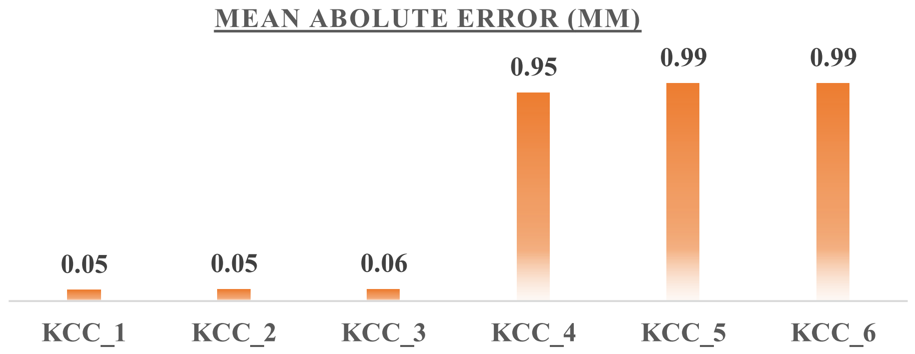
DIGITAL LIFECYCLE MANAGEMENT

CASE STUDY - DIAGNOSIS ANALYSIS*

Case 1: Data Collection after stage 1 (end of positioning sequence)







- **Given data is collection after stage 1, only KCC 1, 2 and 3 are discriminable and diagnosable**
- **There is no inferential information for the rest of the KCCs**

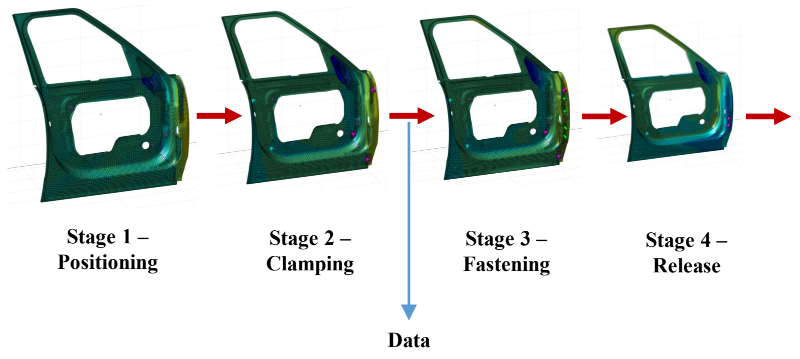
**All diagnosis analysis is done considering worst case scenario i.e. simultaneous variations in all process parameters (or all root causes present in the system)*

WMG THE UNIVERSITY OF WARWICK

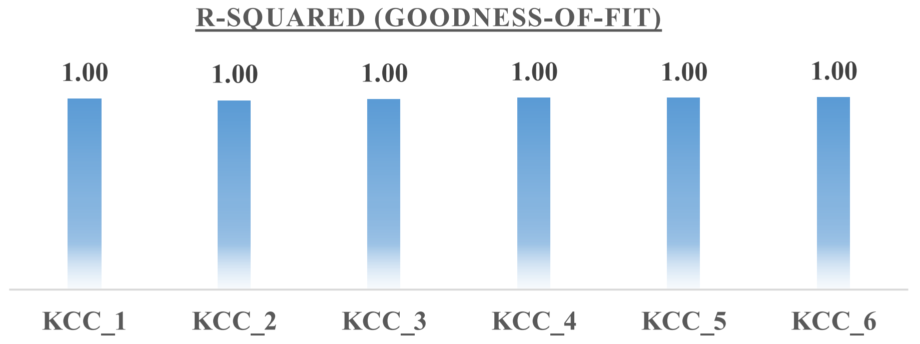
DIGITAL LIFECYCLE MANAGEMENT

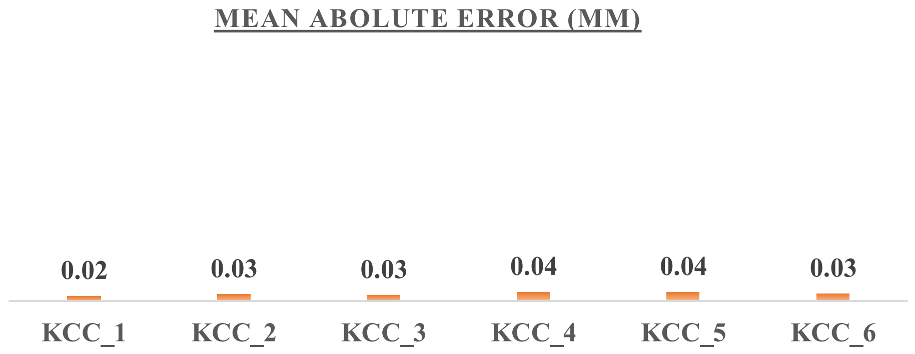
CASE STUDY - DIAGNOSIS ANALYSIS*

Case 2: Data Collection after stage 2 (end of clamping sequence)



- **Given data is collection after stage 2 all KCCs are perfectly diagnosable given at this stage the relation is approximately linear**



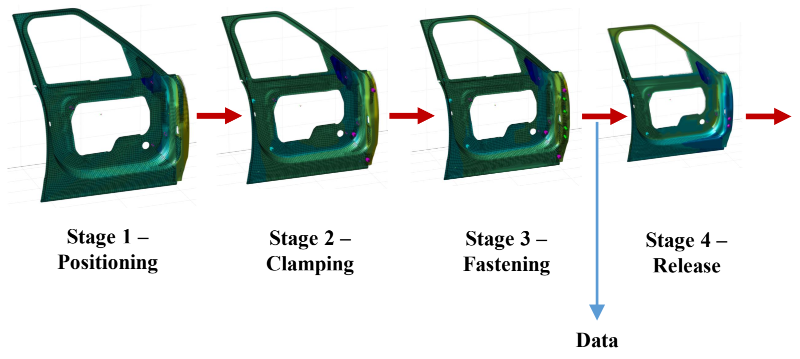


All diagnosis analysis is done considering worst case scenario i.e. simultaneous variations in all process parameters (or all root causes present in the system)

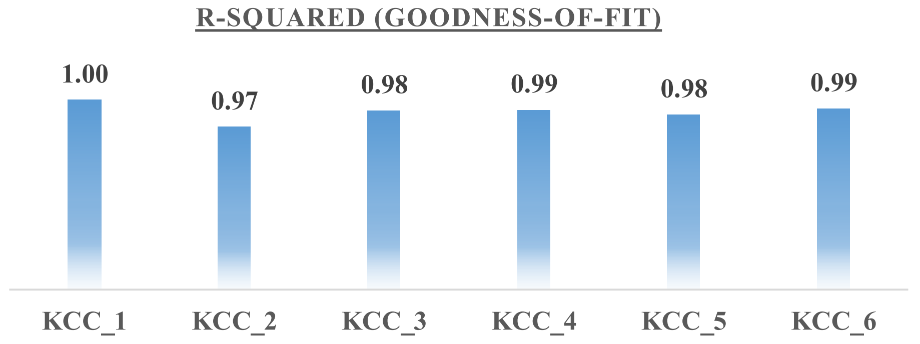
DIGITAL LIFECYCLE MANAGEMENT

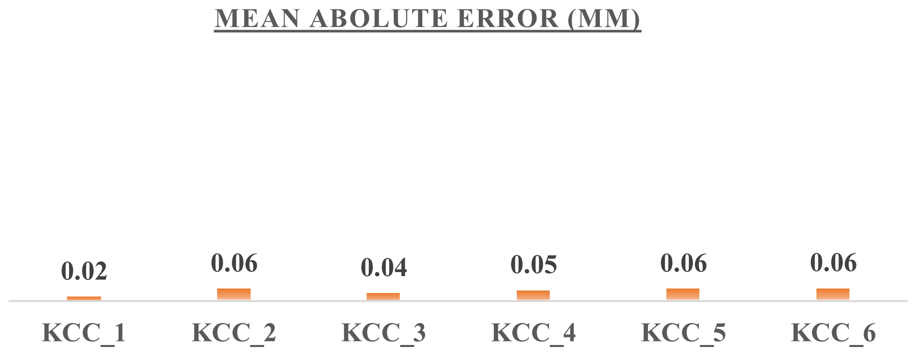
CASE STUDY - DIAGNOSIS ANALYSIS*

Case 3: Data Collection after stage 3 (end of fastening sequence)



- **Given data is collection after stage 3 all KCCs are diagnosable with an error up to 0.6 mm**





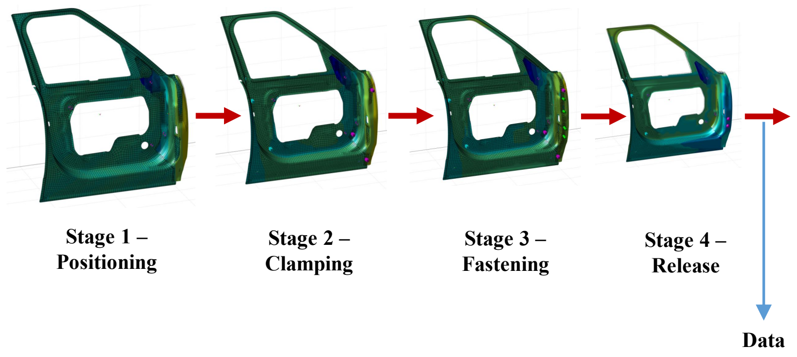
**All diagnosis analysis is done considering worst case scenario i.e. simultaneous variations in all process parameters (or all root causes present in the system)*

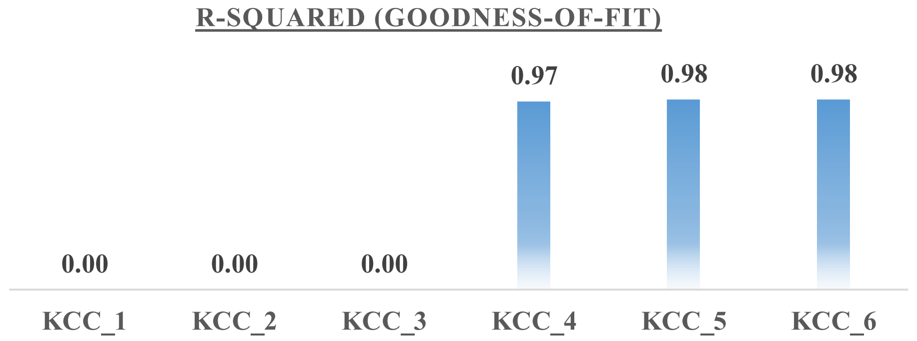
WMG THE UNIVERSITY OF WARWICK

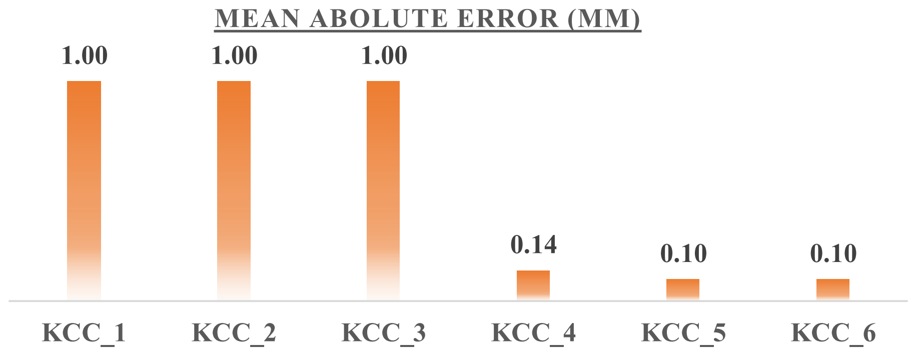
DIGITAL LIFECYCLE MANAGEMENT

CASE STUDY - DIAGNOSIS ANALYSIS*

Case 4: Data Collection after stage 4 (end of assembly sequence)







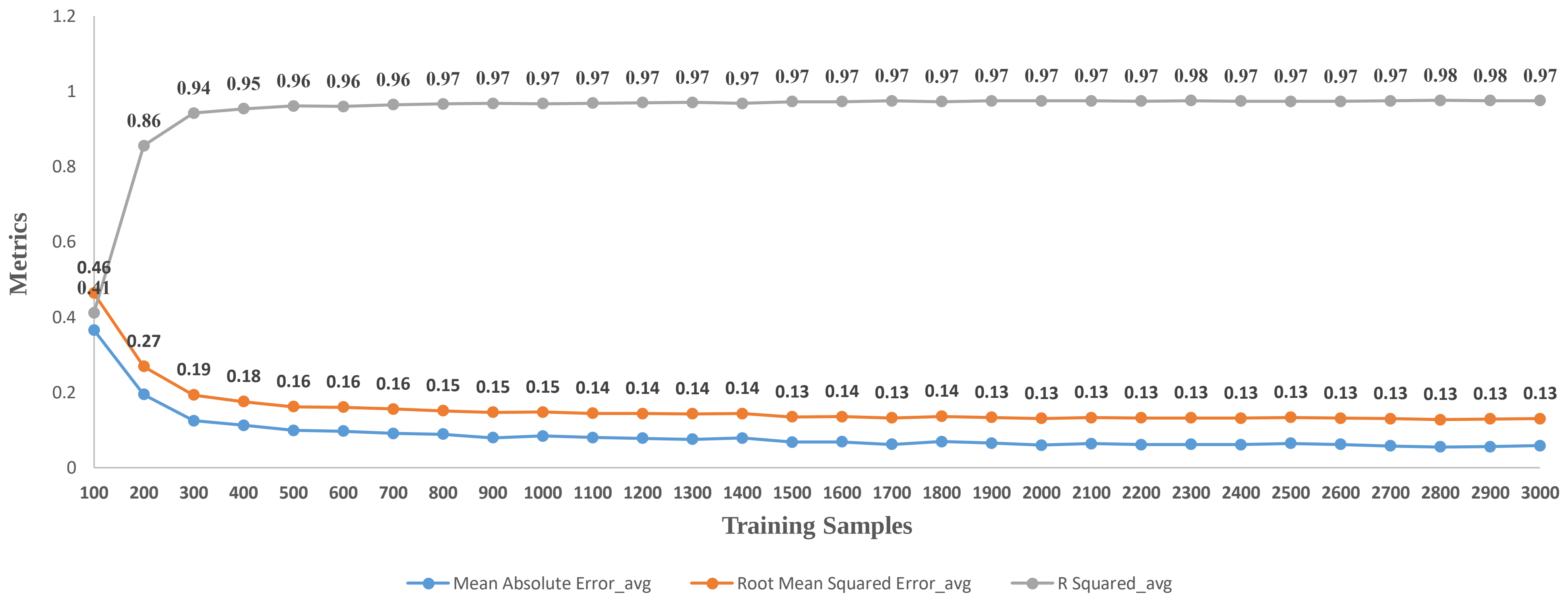
- **Given data is collection after stage 4 KCC 4, KCC 5, KCC 6 are diagnosable with an error up to 0.14 mm**
- **Analysis while varying KCC 1, KCC 2, KCC 3 are yet to be done**

**All diagnosis analysis is done considering worst case scenario i.e. simultaneous variations in all process parameters (or all root causes present in the system)*

WMG THE UNIVERSITY OF WARWICK

DIGITAL LIFECYCLE MANAGEMENT

CASE STUDY – DATA CONVERGENCE STUDY



- If data is collected at stage 3, the model converges with 800 training samples with a root mean square error (RMSE) of 0.14 and R-squared value of 0.97

Html based plot consisting of metrics for each KCC generated using Plotly Python Library embedded below

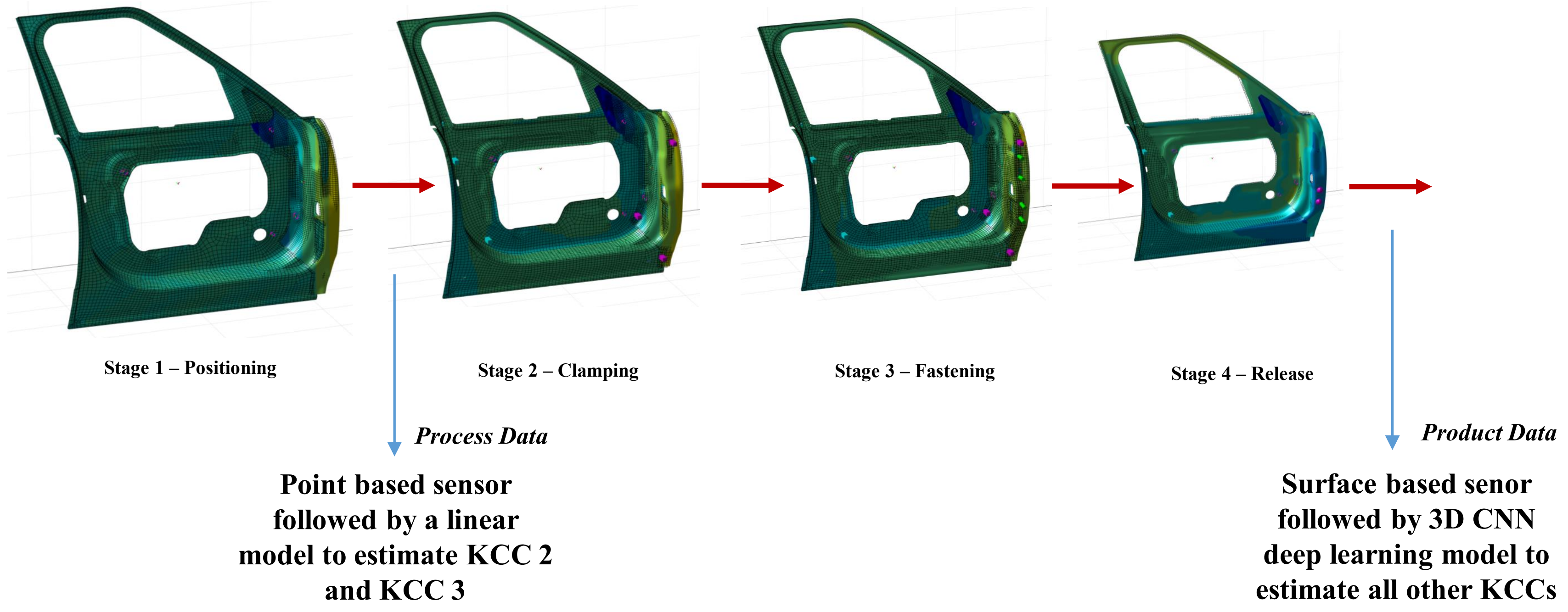
data_study_plot_test.html

WMG THE UNIVERSITY OF WARWICK

CASE STUDY - RESULTS

The optimized data collection and diagnosis architecture to ensure complete diagnosis ability would require:

- Process (KCC point based data) data after positioning (stage 1) to estimate in-plane movement due to movement in pin and slot (KCC 2 and KCC 3)
- Product (Cloud-of-point) data at the of line to estimate all other KCCs of the system


Stage 1 – Positioning
Stage 2 – Clamping
Stage 3 – Fastening
Stage 4 – Release
Process Data
Point based sensor followed by a linear model to estimate KCC 2 and KCC 3
Product Data
Surface based senor followed by 3D CNN deep learning model to estimate all other KCCs

**All diagnosis analysis is done considering worst case scenario i.e. simultaneous variations in all process parameters (or all root causes present in the system)*

CASE STUDY – STEPS FOR MODEL TRAINING AND DEPLOYMENT

1. Download the **dlmfg library** to your python environment *(it is recommended to create a new environment to prevent version mismatch of the prerequisites e.g. TensorFlow, Keras etc)* using Git clone https://github.com/sumitsinha/Deep_Learning_for_Manufacturing or pip install dlmfg *(if you are using git clone you would need to install the requirements by running pip install –r requirements.txt from the root folder)*

2. Transfer the configuration files from inner_rf_config_file to the config root folder

3. Run **data_download.py*** located within core that creates the required folder structure and downloads the files within them they consist of :
   1. a folder within datasets with same name of the case study consisting of the model input i.e. node deviations *(Files:)*
   2. a folder within active learning with the same name of the case study consisting of the model output i.e. process parameters or KCCs *(Files:)*
   3. a file within resources/mapping files consisting of the voxel mapping *(File: voxel_mapping_inner_rf_64.csv)*
   4. a file within resources/nominal_cop consisting of the nominal Cop in oder of node IDs *(File: nominal_cop_inner_rf.csv)*

4. Run **model.train.py*** located within core that trains the model by spitting the train data into train and validation sets and creates an output folder within the trained_models folder with the same name of case consisting of the following file structure:
   1. Logs – model training logs and training metrics (R-squared, RMSE, MAE, MSE)
   2. Model – trained model in Keras trained_model.h5 format
   3. Plots – training loss convergence visualization

5. Run **model_deployment.py*** within core to test the trained model on the test set and obtain accuracy metrics within the logs folder

6. Run **data_study.py*** within model core to train model incrementally by increasing the training data in steps, this obtains accuracy metric at each step by testing the trained model on the test set, this generates html plotly based plots for better visualization and training/testing metrics for each step within the logs folder

** All these files parse the requirement parameters from the case study configuration files within the config folder, each case study has a different set of configuration files which need to be changed in case a new case study within different parameters needs to be trained and deployed*

CASE STUDY – DATA DESCRIPTION

Input Data to the 3D CNN model (Independent variables/Predictor Variables)*:

- x, y, z deviations of 10841 nodes generated as output from the VRM for all the stages of the assembly system
- These correspond to the output of the VRM simulation software

Output Data of the 3D CNN model (Dependent Variables/Predicted Variables)*:

- Process parameters (KCCs)
- These correspond as input to the VRM simulation software

Mapping Files to map deviation nodes to Voxel locations (64*64*64)

- Voxel index (i, j, k) for each node (x, y, z) *(File: voxel_inner_rf_halo_64.dat)*

Nominal Cloud of Point with node ID

- Node nominal location in 3D Space *(File: nominal_cop_inner_rf.csv)*

**Split into train and test set, test set contains samples beyond the range of the train set to check for better generalization and prevent use of overfit models*

Thank-you

Digital Lifecycle Management (DLM) Research Group

Please contact Sumit Sinha (email: sumit.sinha.1@warwick.ac.uk) in case of any doubts

S. Sinha, E. Glorieux, P. Franciosa, D Ceglarek