On the Setup and Simulation of Large Scale LEGO® Models built with LS-DYNA® and LoCo

Thorsten Gerlinger[1], David Koch[1], Andre Haufe[1], Nils Karajan[2],
Thomas Weckesser[2], Pierre Glay[3], Alexandru Saharnean[4], Marko Thiele[4]

[1]DYNAmore GmbH
[2]DYNAmore Corporation
[3]DYNAmore France SAS
[4]SCALE GmbH

1 Introduction

Playing with LEGO® bricks is something many engineers might have enjoyed during their childhood. Building any kind of mechanical construction allows creativity and complexity to an extent which probably contributed to their fascination and finally to their decision of becoming engineers. It's interesting to see how many of them are still fascinated by LEGO® even in their adult life. Especially for children, or for those with an active inner child, crashing these models into each other is even more fun, because seeing all those bricks fly all over the place is just fascinating, beyond any scientific or professional aspect.
Given that in their professional live CAE engineers are dealing with highly sophisticated crash models on a daily basis, it seems obvious that, when confronted with videos such as this real crash of a Porsche 911 GT3 RS LEGO® Technic (42056) model on YouTube[2][4], one instantly wishes to simulate this with the LS-DYNA® [5] FEM solver.

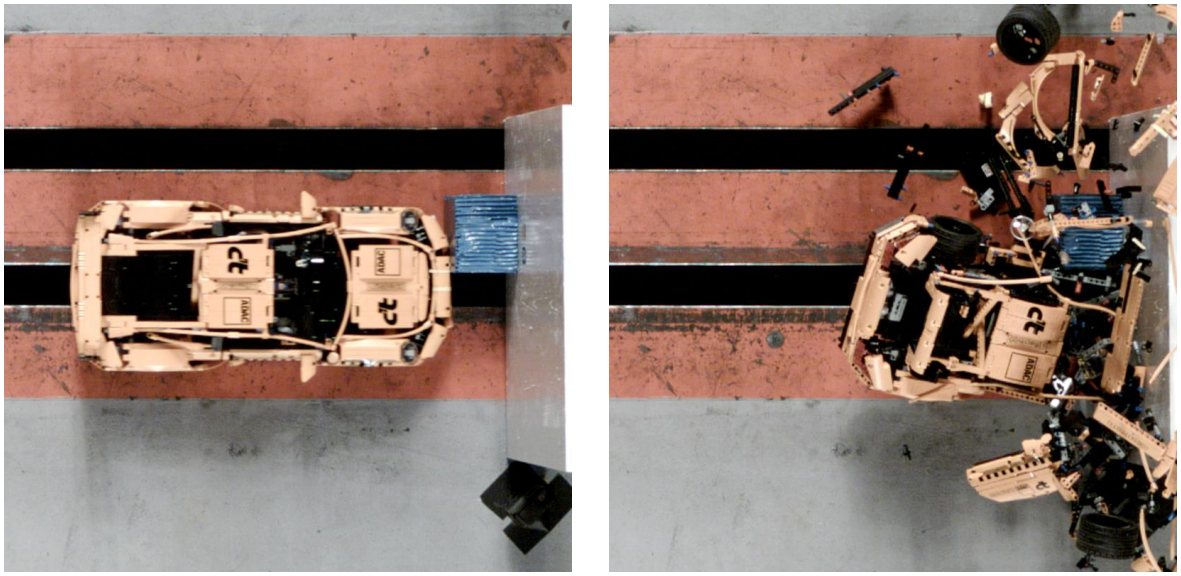

Fig. 1: Physical crash of Porsche 911 GT3 RS LEGO® Technic (42056) model by c't and ADAC[2][4].

Apart from the desire to do so and of course also just because it **is** possible, there are several reasons why it's actually worth doing it. Setting up a process, which involves every aspect of working with CAD data, meshing, dealing with solver files, submitting and monitoring the simulations, and finally handling the sparse result files of simulations, is an important step when developing a simulation process and data management (SPDM) system such as LoCo[6].
The real LEGO® models are often assembled with thousands of bricks. Handling so many parts in a SPDM system on one hand and maintaining the ability to work on such models in a collaborative way with multiple users on the other is quite challenging. Just opening and working with CAD data containing so many parts in modern CAE preprocessing tools is a resource demanding and complex task. Beyond that, a number of other issues also have to be solved. Avoiding redundant work by reusing already meshed bricks from one or multiple previous models is just one example. These issues are also common in real collaborative product developing projects and have to be dealt with accordingly.

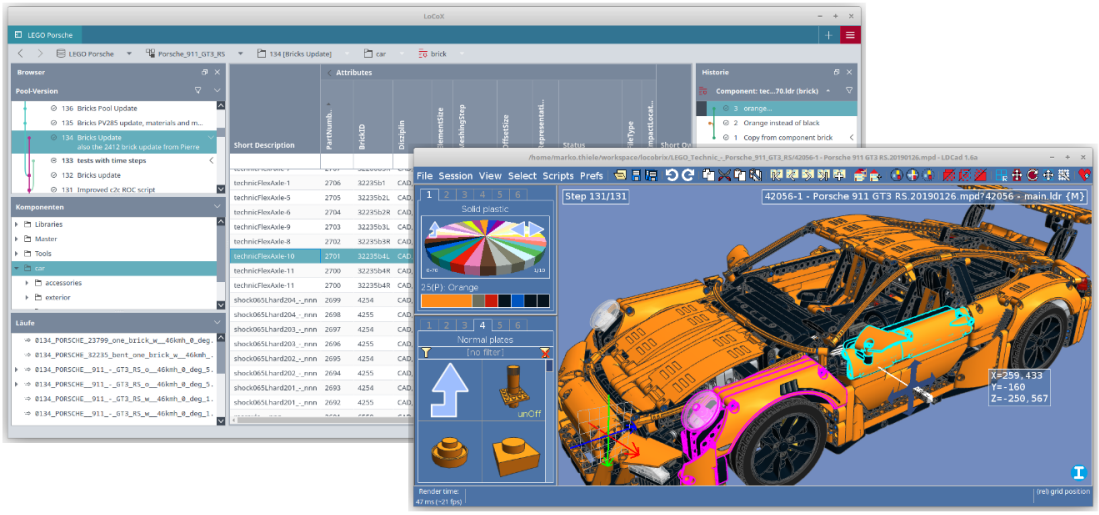

Fig. 2: Working with CAD within LoCo.

One further aspect is that using LEGO® for explaining a simulation process configured with LoCo, teaching mechanical engineering or Finite Element Methods (FEA) might be quite motivating. It could help class attendees or young students stay focused and pick up some FEA ideas, see the fun in physics and the challenges for team work in case of real and complex products.

2 Creating and working with LEGO® models using various software tools

Searching the internet, one can find a seemingly endless supply of resources, including pictures, videos, construction manuals and 3D CAD models on sites such as ldraw.org[7] or bricklink.com[8]. These 3D models are commonly made with special CAD software, like LDCad[9], LeoCAD[10], Studio[11] and complementary tools such as LDView[13]. Furthermore, there is a vivid community at ldraw.org[7], maintaining a library of CAD Data for all the numerous bricks that are available from LEGO®. This variety of source material also implies an advantage for using LEGO® focused CAD programs.
In order to get started with creating own CAD models, the aforementioned software Studio from BrickLink might be a good choice, since its easy to understand concepts allow even young students of age 10+ to build simple models.

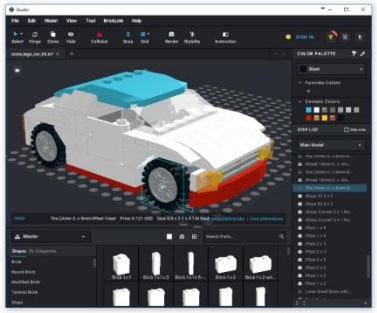

Fig. 3: Model opened in Studio LEGO® CAD software.

Fig. 4: Photorealistic rendering generated with Studio.

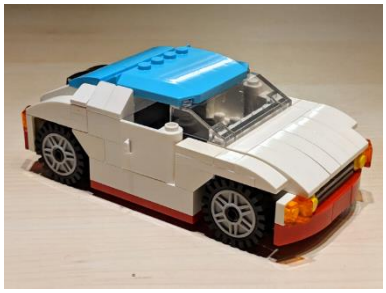

Fig. 5: Photo of physical build of scale car model.

Before actually starting with the sophisticated Porsche 911 GT3 RS LEGO® Technic (42056) model, henceforth referred to as Porsche model, a model of a smaller car was created. This gave the possibility to get used to the tools and the setup of the processes needed later for working with the considerably larger Porsche model.
The model shown in Fig. 3 and Fig. 4 is publicly available and can be obtained from bricklink.com[8]. So if you want to take a look, just go ahead and try yourself. On BrickLink, as well as on the OMR (Official Model Repository)[12] at ldraw.org[7] you will also find many other models of the same kind, just to investigate and play with.
Furthermore, the online store at the BrickLink website offers LEGO® bricks. This makes it quite convenient to build a physical model, which was previously created with the software. Fig. 5 shows the physical build of our starting model.

For more advanced models, like the Porsche model that will be shown henceforth, or for those favoring open source, LDCad might be the preferred choice. LDCad is also better suited for integration with LoCo and offers a significantly better performance and handling of flexible parts.

3 Setting up the model and the simulation process with LoCo

3.1 Aspects of the teamwork involved

When the idea took off to set up a simulation process for generating LEGO® models made out of thousands of bricks, it became clear that this ambitious task would require teamwork. Therefore, the whole process was setup using LoCo by colleagues from SCALE GmbH in the offices of Ingolstadt and Dresden, Germany. LoCo is a SPDM system that allows to setup simulation processes and to handle the data required for simulation and sharing that data among various team members.

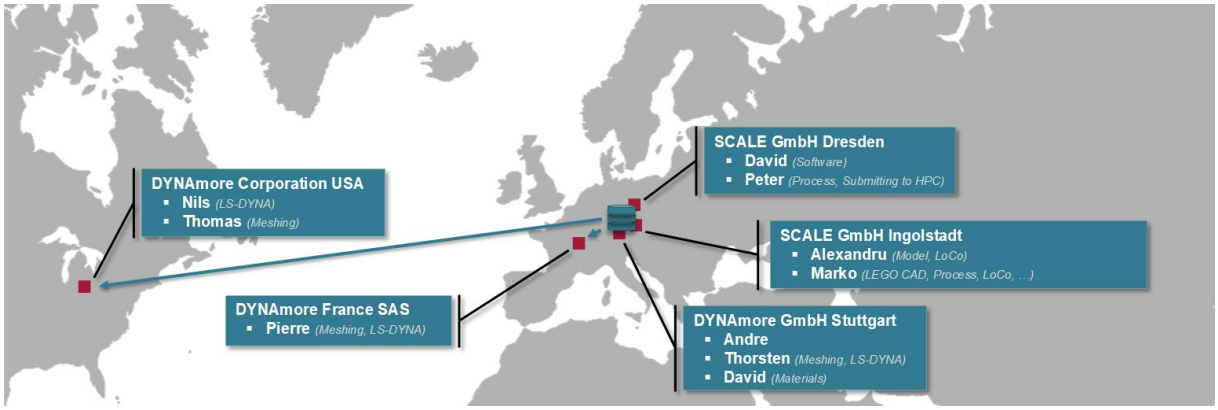

Fig. 6: Tasks during collaborative effort to setup the simulation processes with LoCo SPDM system.

The main challenge when working together on the same model is that no more than one person can work on the same pieces of data at the same time. There are several approaches to deal with this problem. The most user friendly approach is the realization of a system that allows for real-time editing among several team members within the same files. This has been done for simple text editors in projects such as Etherpad[14] and lately also for office documents, e.g. Google Docs[15] or Collabora Online[16]. However, the typical CAD and CAE preprocessing tools are not capable of such simultaneous editing. In order to collaborate on the same CAD or CAE model, a commonly used approach is to split up the model into several files. As the size of the team grows, the importance of splitting up the model into ever smaller parts increases. This is also something that is seen in productive examples of real car projects. If consequently applied, this approach results in setups where each part in a whole car assembly is held in an individual file. For complete car projects the amount of parts/files to be handled becomes quite large such that the SPDM system has to be able to deal with several thousands of parts/files at the same time. This is especially challenging with regard to the user interface as well as the underlying mechanisms such as synchronization of data. Therefore, creating such a “worst case” scenario is important for the development of a SPDM system and was part of the main motivation for creating the LEGO® examples.

3.2 Process setup

The most used format to describe LEGO® models is LDraw[7]. In this format basically every line is just providing the transformation information, a color and a reference to one brick in the standardized LDraw brick library. When trying to go from the LDraw format to actual LS-DYNA® simulation models it seemed natural to use a similar approach as used in the LDraw format. The goal is to be able to import LDraw models relatively easy and use the positional information of the bricks in order to position the actual meshed bricks. This way each brick would have to be meshed only once and can be used instantly wherever it appears in the model. The idea is, that, if a library of meshed bricks in the same positions as the bricks in the LDraw library is available, it should be easy to create simulation models for any of the countless LDraw models available online. Maybe at some point even easy enough for young students. A created mesh of one brick can be used over and over again by utilizing `*INCLUDE_TRANSFORM` cards of LS-DYNA® to import the brick to its various locations. Within LoCo the meshed bricks are also maintained in a shared library pool that can be used throughout various different projects. This makes it easy to share the same version of the meshed bricks throughout all projects.

3.2.1 CAD Data

The CAD Data for the Porsche model[17] has been obtained as LDraw file from the OMR (Official Model Repository)[12]. It was published by Philippe Hurbain and released under the CCAL version 2.0[18] which allows every kind of usage. Within the LDraw file a typical line contains the transformation and color information as well as a unique brick ID for one brick of the model.

```
0 FILE 42056 - main.ldr
0 main
0 Name: 42056 - main.ldr
0 Author: Philippe Hurbain [Philo]
0 !LDRAW_ORG Model
0 !LICENSE Redistributable under CCAL version 2.0 : see CAreadme.txt

0 !THEME Technic

0 ROTATION CENTER 0 0 0 1 "Custom"
0 ROTATION CONFIG 0 0
1 71 -0.567 0 -180.567 0 0 1 0 1 0 -1 0 0 64179.dat
1 1 -60.567 0 -160.567 -1 0 0 0 0 -1 0 -1 0 6558.dat
1 1 -60.567 0 -200.567 -1 0 0 0 0 -1 0 -1 0 6558.dat
1 0 -40.567 -40 -180.567 0 1 0 0 0 1 1 0 0 60484.dat
1 0 -40.567 0 -230.567 0 1 0 0 0 1 1 0 0 2780.dat
1 0 39.433 0 -230.567 0 1 0 0 0 1 1 0 0 2780.dat
```

Fig. 7: First lines of the LDraw model file of the Porsche model.

The files can also contain nested structures where parts of the model are referenced and transformed as a whole. This makes working with part assemblies, for example a wheel assembly, easier as they can be built once and placed on various positions in the model.
For importing these structures into LoCo, a script was implemented, which splits the LDraw file into individual files for each brick. Afterwards, the individual files where imported in a batch operation and also, the nested structure for subassemblies was rebuilt within LoCo. This allows some team members to work on one subassembly while others work on another one. This becomes especially handy when using the LiveMode feature of LoCo[3]. Here the SPDM system keeps track of who is working on which parts of the model, such that users do not need to merge the versions of the model parts they were working on later in time.
Once the CAD data was imported from the LDraw file to LoCo, the users are able to work directly with e.g. LDCad within LoCo as shown in Fig. 2. For this purpose a small wrapper script was written, which combines the files with single lines per brick into one big file and opens it in LDCad. After saving the changed assembly in LDCad, the file is split again and only the changed and new bricks are saved back to LoCo. The user does not notice this process. He just clicks on an assembly and requests it to be opened in LDCad. New versions of all files and folders are created and synchronized automatically to all other team members working on the project. In this case LoCo also acts as a collaboration platform, giving the possibility to try out different versions of the actual LEGO® model or create a model in cooperation with others.

3.2.2 Meshing

The main collaborative effort was to do the actual meshing. This has been done by colleagues from DYNAmore GmbH, DYNAmore Corporation in USA and DYNAmore France SAS using LoCo to organize and exchange the meshed bricks (Fig. 6). Each brick was meshed as a 2 mm as well as a 1 mm variant which are kept together in LoCo in one place such that one is able to easily generate simulations using either of the mesh resolutions. Together with the LS-DYNA® models for each brick, the CAD data and cleaned geometry were also stored in LoCo as ANSA[24] DB files.
The original CAD data for each brick from the LDraw brick library is most often available in the form of a simple triangulation representing the surface of the brick. This can easily be converted to `*.stl` or `*.obj` files using LeoCAD or LDView and imported to ANSA or Hypermesh. However, these meshes are not suitable at all for conducting simulations. Initial experiments trying to use the simple original STL meshes as rigid bodies have not been very promising and therefore it was decided to create primarily tetrahedral volume meshes with a target edge length of 1 mm and 2 mm.

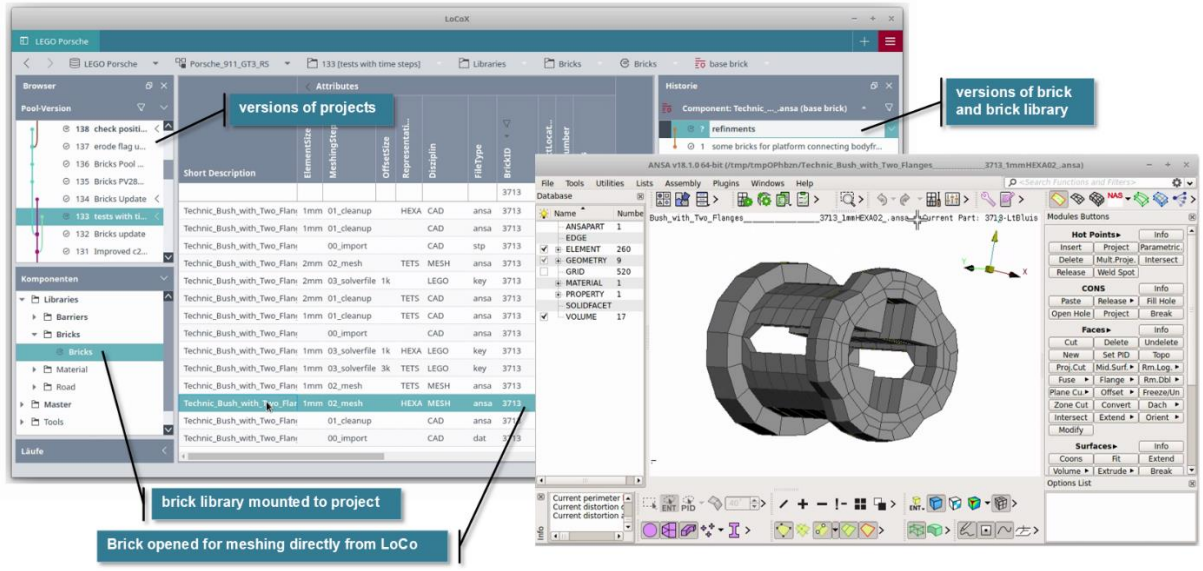

Fig. 8: Managing the library of meshed bricks in LoCo

Even though the `*.stl` files could be used as a basis for creating the required FEM meshes, they are actually not very convenient to work with in meshing tools. This is because they are not made out of real primitives and therefore already contain an inaccuracy, since they are composed of polygons instead of native geometries. Fortunately, the Porsche model has also been available as "real" CAD data on GRABCAD[19] where the user dk[20] has kindly allowed us to use the geometric data of the bricks for our investigations. The geometry from GRABCAD only had to be positioned coincidental to the geometry of the LDraw geometry and could then be used for the subsequent meshing step.

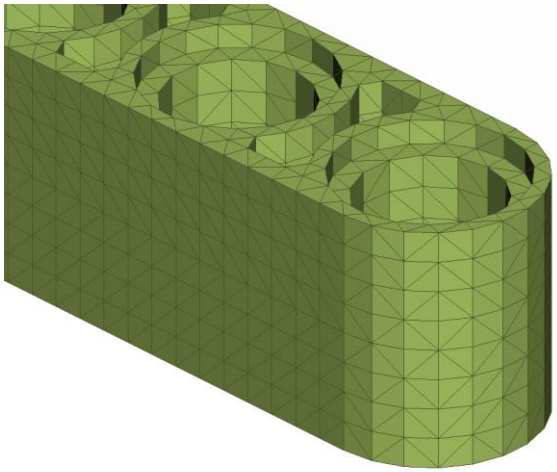

Fig. 9: Technic Beam 13, BrickID 41239 1mm Tetra mesh, 33123 elements

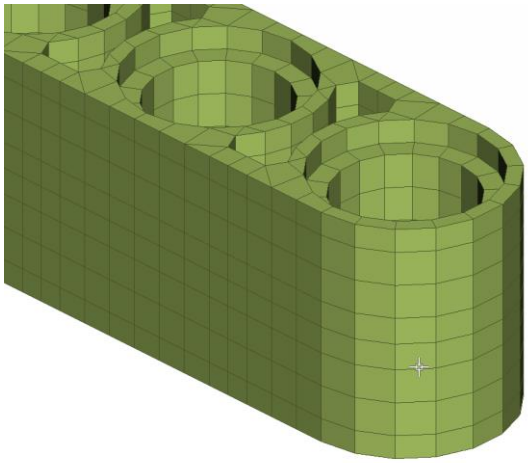

Fig. 10: Technic Beam 13, BrickID 41239 1mm Hexa mesh, 5124 elements

One drawback when using volume elements is that this results in models with many elements. This can be improved by using hexahedral elements. As shown in Fig. 9 and Fig. 10 this can dramatically reduce the number of elements. For the final model, where bricks such as #41239 are used 29 times throughout the whole model, this saves more than 800,000 elements for the final simulation.
The resulting complete Porsche model contained more than 2700 bricks, each handled as an individual part. For the variant using the bricks meshed with 1 mm target element size this model contained an astonishing amount of roughly 19.5 million elements. This exceeds the usual model size for integrated simulations of full vehicle crashes used in real virtual product development at OEMs by a factor greater than two. However, using the 2 mm variant results in loss of many details of the bricks since the contact situation, i.e. the way how the bricks stick to each other, is crucial for the overall model behavior. Hence, it seems advisable to use at least the 1 mm variant.

The average run time on a high performance cluster (HPC) system using 192 CPUs for the simulation was 22 hours for 120 ms of simulated time.

3.2.3 Assembly

Other files needed for the final simulation, such as a master file with all the control cards, barrier models, guiding blocks (that had been attached below the vehicle to make sure it would run straight), files with material cards and so on are also maintained in LoCo. Finally, a script is stored together with all the components of the model, which is executed by LoCo when assembling an actual LS-DYNA® solver deck. This script takes the transformation information of each brick and translates them to LS-DYNA® **`*INCLUDE_TRANSFORM`** and **`*DEFINE_TRANSFORMATION`** cards such that each **`*.key`** file of the bricks is imported over and over again.

```
$#############################################################################
$ Include - Transform for:
$ dashboard2_-_nnn_____________558_3070b_000000_----_3b16a7a2.ldr
$#############################################################################
$
*DEFINE_TRANSFORMATION
  20115001
$
$  Rotation:
$
    ROTATE       1.0       0.0       0.0       0.0       0.0       0.0      90.0
$
$ um X
    ROTATE       1.0       0.0       0.0       0.0       0.0       0.0       -90
$ um Y
    ROTATE       0.0       1.0       0.0       0.0       0.0       0.0        -0
$ um Z
    ROTATE       0.0       0.0       1.0       0.0       0.0       0.0       180
$
$  Translation:
$          x         y         z
    TRANSL   -0.2268       -80  -109.827
$
$ final rotations
$
    ROTATE       1.0       0.0       0.0       0.0       0.0       0.0     -90.0
    ROTATE       0.0       0.0       1.0       0.0       0.0       0.0     -90.0
$_____________________________________________________________________________
$
*INCLUDE_TRANSFORM
Tile__1_x__1_with_Groove_____________________3070b_1mm03_.key
$#  idnoff    ideoff    idpoff    idmoff    idsoff    idfoff    iddoff
  20115000  20115000  20115000         0  20115000  20115000  20115000
$#  idroff
  20115000
$#  fctmas    fcttim    fctlen    fcttem    incout

$#  tranid
  20115001
$
```

Fig. 11: Automatically generated solver cards for including bricks with transformation.

The same script also creates session files for LS-PREPOST[21], Animator[22] and META[23] for correct coloring when visualizing the resulting crash animations during post-processing.
With this setup in place it becomes easy to apply changes to the LEGO® CAD model by using e.g. LDCad or importing other LEGO® models and run these as new simulations. Provided the meshed bricks already exist for the bricks used in the model, the simulations can be run right away.

3.3 Setup of different load cases and model variants

Once the entire simulation process has been set up and a running simulation model has been created it becomes easy to set up different load cases using LoCo. Besides the 40% ODB offset barrier crash shown on YouTube[2][4], load cases for frontal rigid wall, 50% and 25% offset barrier, crashes with 30° impact angle and various different velocities were set up as well.
LoCo has the ability to use attributes defined on components and parameter values in order to relate them automatically to specific simulations. This makes it possible to create a high amount of different variations on one virtual product without having to relate all components (files, includes) individually to each load case. Therefore a change applied to e.g. one brick is automatically taken over to all defined use cases and submitting the jobs for many load cases to evaluate an overall status of a simulation project becomes effortless.
A car to car crash simulation using the multi stage assembly feature of LoCo was also set up. With this feature one can use the result of one assembly, e.g. the assembled Porsche model, within another assembly for a different simulation.

4 Modeling aspects of LS-DYNA®

4.1 Global Contact

Thanks to LS-DYNA® [5] contact algorithms, the strategy of the contact is relatively straightforward for users in automotive crashworthiness applications. Only one single surface contact is needed to take care of the full vehicle, including the ODB offset barrier. This has the huge advantage to eliminate the need to know contacting surfaces. **`*CONTACT_AUTOMATIC_SINGLE_SURFACE`** has been used on this purpose. It provides a way of treating interaction between disjoint elements and therefore, the goal is to prevent penetration between parts or between elements of the same part.
In order to include all the bricks into the contact surface a **`*SET_PART_COLLECT_TITLE`** card has been defined for each brick.

```
$
*SET_PART_COLLECT_TITLE
global contact
      1001       0.0       0.0       0.0       0.0
         2
$
```

Fig. 12: Definition for the set defining the contact surface.

This is the identical card in each brick. And since the bricks are imported multiple times with offsets for the individual part IDs all the parts for the overall simulation are collected automatically depending on which bricks are actually used in the models. The big advantage of this approach is that one does not have to define a contact surface prior to the simulation but instead the contact surface depends on which bricks are included at the time of assembling the model. Therefore it becomes particularly easy to interchange bricks, introduce new bricks or delete undesired bricks without having to manually update the contact surface.

4.2 Material definition of the LEGO® bricks

In order to be able to reproduce realistic part deformations and material behavior, tensile samples were taken from LEGO® components and tested at the DYNAmore materials test laboratory in Stuttgart, Germany. Due to its large flat surface, brick #59349 is very well suited for this purpose. The specimens were then investigated quasi-static in a tensile testing machine, accompanied by an digital image correlation (DIC) measuring system to capture the local straining of the samples. Finally, the **`*MAT_024`** material card (elasto-plastic material model with isotropic hardening) was calibrated using a reverse engineering strategy within a state-of-the-art industry accepted process.

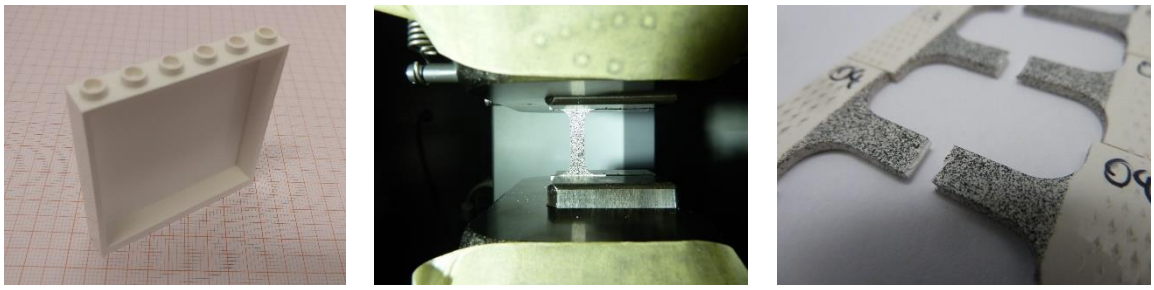

Fig. 13: Material testing to calibrate LS-DYNA® MAT24 cards.

Since the Porsche model contains also a few rubber parts, generic data collected from public sources has been used to feed **`*MAT_077_O`** (general hyperelastic rubber model) in the final model.

4.3 Clamping force

For the moment the models are held together solely by contact and friction which is not quite capturing the physical truth because in reality the bricks are also held together by a significant amount of clamping force. Various options to include tiebreak contacts are under investigation at present to improve this behavior but even without them the simulation results already look very impressive when compared to reality. However, one can clearly see that in many occasions the bricks separate too easily. Especially smaller bricks that are only connected through studs are behaving differently than what can be seen in the physical tests.

4.4 Rotation of wheels

The initial velocity of the model is defined using LoCo-parameters which then are used in a parameterized `*INITIAL_VELOCITY_GENERATION` card. In this card the initial velocities are applied to a given set of nodes. This set of nodes contains all the nodes of the model. This is achieved by defining a `*SET_NODE_LIST_GENERATE_COLLECT_TITLE` card in each key file, using the same ID for the set. In this manner this set collects all the nodes of the model.
To simulate the rotation of the wheels, one needs to define different initial velocities for the parts resembling the wheel. Hence for each wheel part or brick an `*INITIAL_VELOCITY_GENERATION` card is defined additionally in the corresponding ldr file. In this case the card contains both the translational as well as the angular velocity values. The values for the angular velocities are also freely defined by LoCo-parameters with expressions depending on the wheel radius as well as the LoCo-parameters defining the vehicle translational velocity.
In the assembly process the same script that interprets the ldr files to write the include cards into the master file also copies any following `*INITIAL_VELOCITY_GENERATION` cards from the ldr files into the master file. The ID of the card is set correspondingly to the ID offset for the entire key file.

4.5 Use as benchmark model

One further aspect in regard to the practical importance of setting up such a large model is the use as a benchmark model. With this, one could evaluate how such big models behave on HPC systems and use this experience in preparation for the time, when actual real car crashworthiness models are getting in the same range in terms of model size, memory usage and computing time.
The MPP (Message Passing Parallel) version of LS-DYNA® uses a decomposition technique in subdomains in order to divide the model and the MPI protocol to communicate. Each subdomain is managed by one processor. Because just simply increasing the number of processors would not ensure to speed up the simulation significantly, particular attention was paid to finding better domain decomposition than the default one. The Fig. 14 shows the selected decomposition principle on eight processors. The model is cut into slices according to the Y-axis which ensured a better load balancing of calculation cost among processors.

Fig. 14: Domain decomposition for simulation with 8 CPUs.

5 Validation

5.1 Small scale car model

As already mentioned in section 2 as a predecessor of the complex Porsche model a much more simple and smaller scale car model was built with only 134 bricks. However, the first simulation results seemed unconvincing, since the car was heavily bouncing backwards. The expectation was that it gets a lot more spin when hitting the barrier with only one edge of the car. Therefore, the decision was made to construct a small wooden crash sled in order to get some means of physical validation. The process of building this sled took about a weekend's time.

Fig. 15: Crash sled (left) and positioned car for 25% overlap frontal crash (right).

The sled is driven by a slingshot and can accelerate the car to approximately 15-20km/h which is much higher than one would estimate and made the bricks of the car fly all over the place.
The test was then filmed using the slow motion mode of two smartphones. One was placed on top of the housing to create a video showing the crash from above. The other was placed on the left side of the crash sled to provide a side view. Both cameras (Samsung S7 and Pixel 2) have been able to capture video at 240fps and produced reasonable results for such a low budget setup.
The achieved velocity in the physical test has been obtained by analyzing the frames of the video right before the test. Since the physical dimensions of the car, the time between two frames and the amount of pixels the car moved between frames were known, it was easy to obtain the actual velocity of the car right before the crash. The determined speed of 17km/h was then used to create a simulation for validation.
Fig. 16 shows the results for the side view comparing images from the test videos and the simulation. In accordance to the simulation, the physical test also confirmed that the car is bouncing straight back from the barrier right after the impact and does not get much spin.
Two videos comparing the side view[25] results from simulation and crash as well as the top view[26] are also posted on YouTube®. For a simulation with such little tweaking for validation it is already impressive how well the results of the simulation and the test compare.
Having the simulation process in place and the confirmation that the obtained results could be within a reasonable range the road was paved to approach the more challenging task of creating the simulation for the Porsche model.

5.2 heise c't / ADAC crash of Porsche model

When heise published their challenge for the crash test[27] of the Porsche model on YouTube, they left a big cliffhanger by initially not showing the actual crash and asking the community how, in their opinion, the model would behave during the crash. At that point many CAE colleagues instantly stated that LS-DYNA should be able to predict that behavior by simulation. Some of them also downloaded the model from GRABCAD[19] and opened it in ANSA[24]. However, almost immediately it became clear that one would not be able to create any simulation right in time for the heise challenge.
However, the videos later posted by heise and ADAC on YouTube provided good material in order to be used as virtual validation of the simulations to come. For example, the exact positioning of the car as well as the dimensions and the positioning of the barrier have been identified from single images of the videos posted on YouTube.

5.2.1 Element size

Surprisingly the used target element size had a relatively big impact on model behavior. As explained in section 2, meshes where created with 2 mm and 1 mm target element size. Furthermore, when looking at simulations which otherwise use the same parameters, one can clearly see that the behavior of the models are quite different.

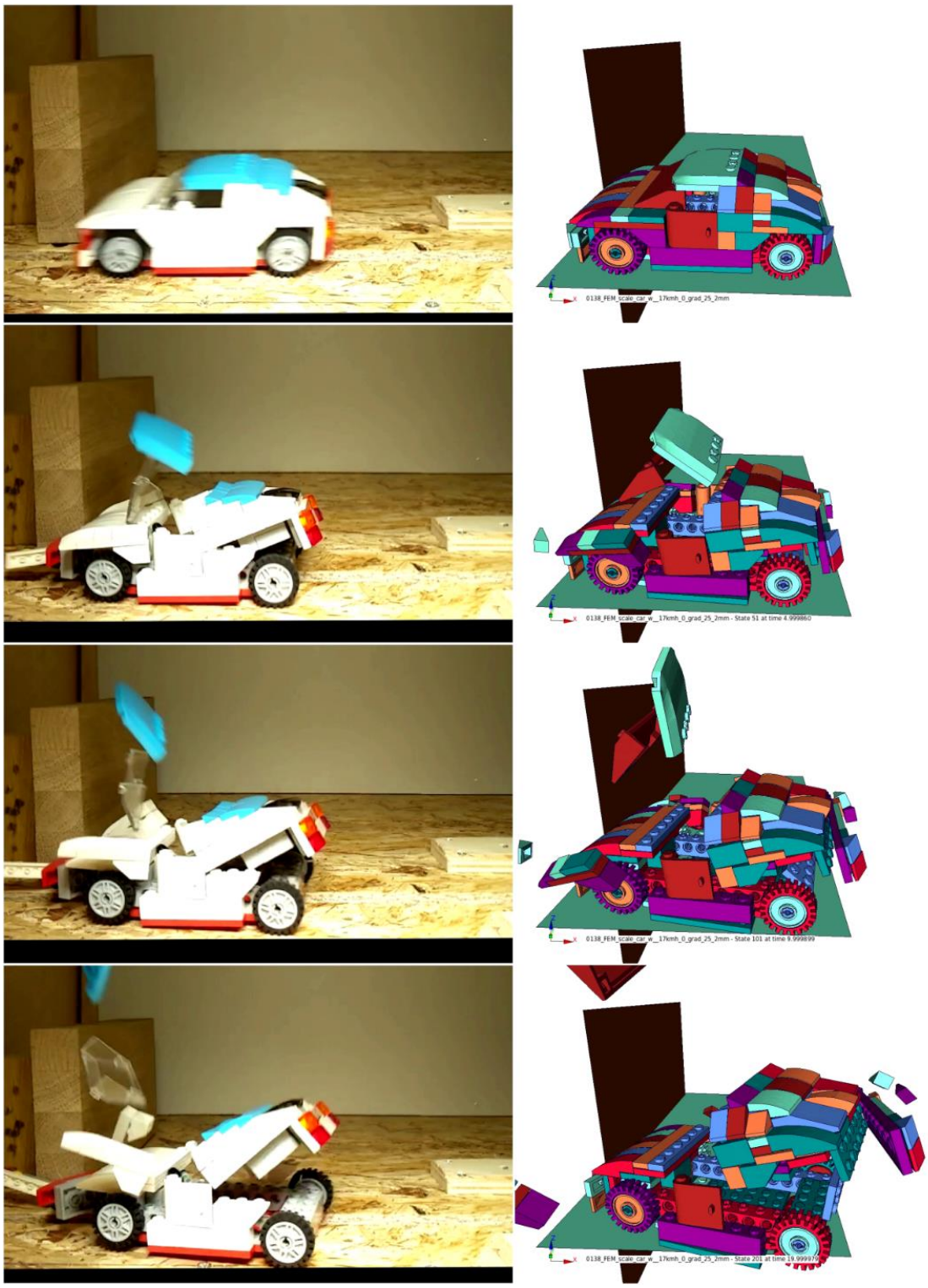

Fig. 16: Physical test compared to simulation for 25% offset crash at 0, 5, 10 and 20 ms of the scale model.

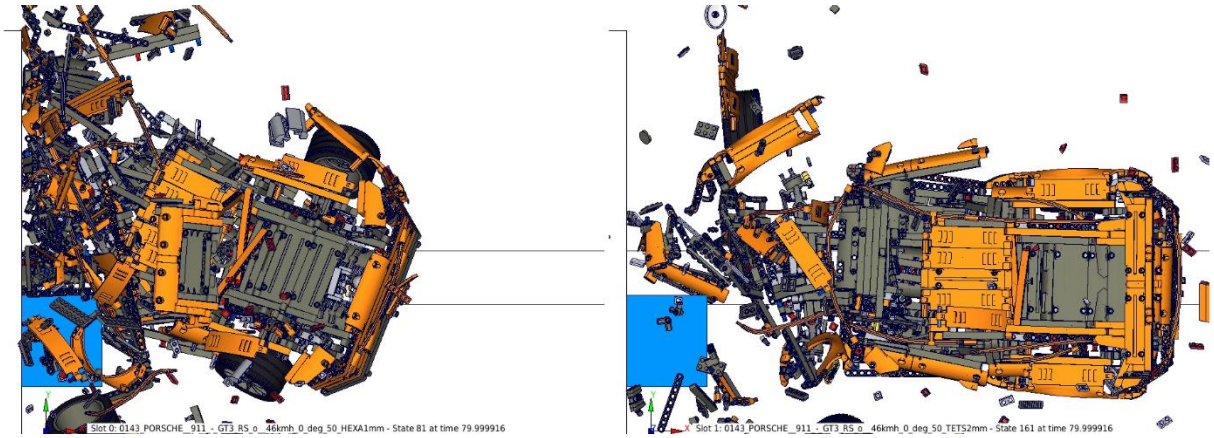

Fig. 17: 1 mm (left) and 2 mm (right) variant of the same simulation at 80 ms of the Porsche model.

As seen in Fig. 17 the 2 mm variant has not the same degree of separation of the bricks. Certainly, the reasons for this are the contact situation as well as a higher discretization error in the 2 mm variant which results in more intersections.

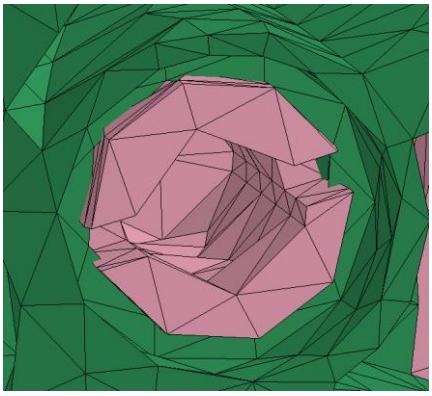

Fig. 18: Intersections between connector and technic brick due to the discretization error for the 2 mm variant.

When looking at the modelling in Fig. 18 it becomes clear that these kinds of intersections will obviously lead to nonphysical behavior of the contact between these bricks and therefore affect the way how they separate.

5.2.2 Friction

A similar impact on model behavior may be clearly attributed to the effect of friction. Naturally, it is very hard to determine correct physical values for static and dynamic friction coefficients. So, at the beginning friction parameters of 0.1 for dynamic and 0.2 for static friction were assumed. However, by changing these values one could observe that this – as usually – has huge impact on the model behavior. Similar to the target element size this might be because the friction directly affects how high the required forces are in order to separate bricks.
A higher friction value seemingly results in higher forces needed to separate bricks and therefore results in less separation of bricks. This is illustrated in Fig. 19 where one can clearly see that lower friction results in higher degree of destruction.
This big impact of coefficients that are not really deterministic is problematic. However, it also gives an opportunity to counter-effect the influence of the element size and choose friction coefficients that result in an overall model behavior similar to the tests.

5.2.3 Plastic material model

As described in section 4.2 a material card `*MAT_024` was created considering actual plastic deformation of the bricks. However, as shown in Fig. 20 this static definition resulted in heavily deformed bricks.

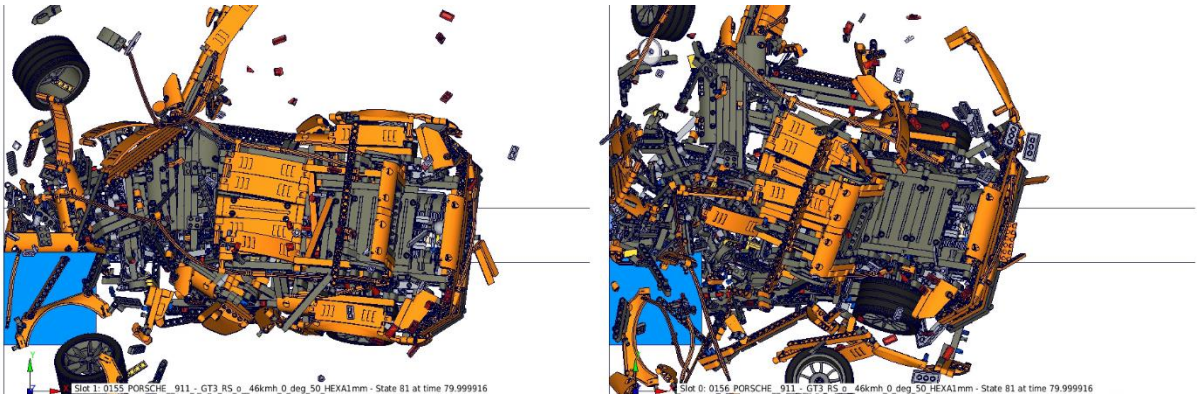

Fig. 19: Influence on model behavior for 0.05/0.1 (right) vs. 0.1/0.2 (left) dynamic/static friction coefficients at 80ms

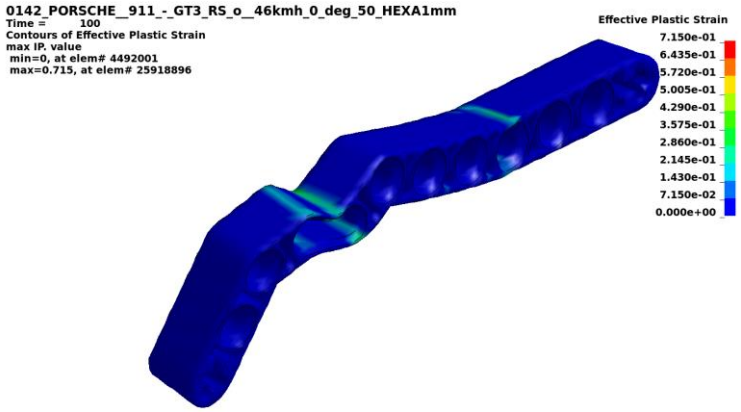

Fig. 20: Plastic deformation of brick during crash

Since this behavior was not seen in the actual physical full car test the material parameters were adjusted such that plastic behavior is not considered for the moment. One possible reason for this discrepancy is seen in the fact that clamping forces between the individual bricks were not defined yet leading to much more disintegration of the model compared to reality. Another reason is attributed to the material parameters that were calibrated for quasi static loading only. Hence any strain rate dependent effects, like work hardening under elevated load velocities are currently not taken into account. In order to capture these effects more comprehensive testing of coupons and small assemblies at higher load velocities are necessary.

5.2.4 Guiding blocks

In the videos, and especially in the Making of Video[28] at YouTube, it was seen that two guiding blocks have been used in order to keep the car on track. These blocks are fixed by screws directly to bricks on the bottom floor of the car. In order to create a model of these blocks the dimensions were extracted from pictures provided by heise and a FreeCAD model was created. This was in turn meshed with ANSA using shell elements. Simple rigid body material parameters were used in order to act as corresponding contact surface. The mass was added by `*PART_INERTIA` cards.

The guiding blocks have a considerable effect to the total rotation of the model during the crash. Especially the front guiding block prevents the car to bounce back and to the side and gives it some spin about the Z axis similar to what can be observed on the videos on YouTube.

Although the model behaves in many aspects differently compared to the footage from YouTube and a lot of things remain to be improved with the simulation, the results are already impressive. The overall behavior of the model is already captured to a reasonable degree with respect to the relatively little effort that was put into validation.

FEA models used in real virtual product development usually undergo a lot more validation and fine tuning. This often involves many different people collaborating on material models, validation of physical parameters, calibration with the test setup and fine tuning many more details in order to capture the physical truth as good as possible.

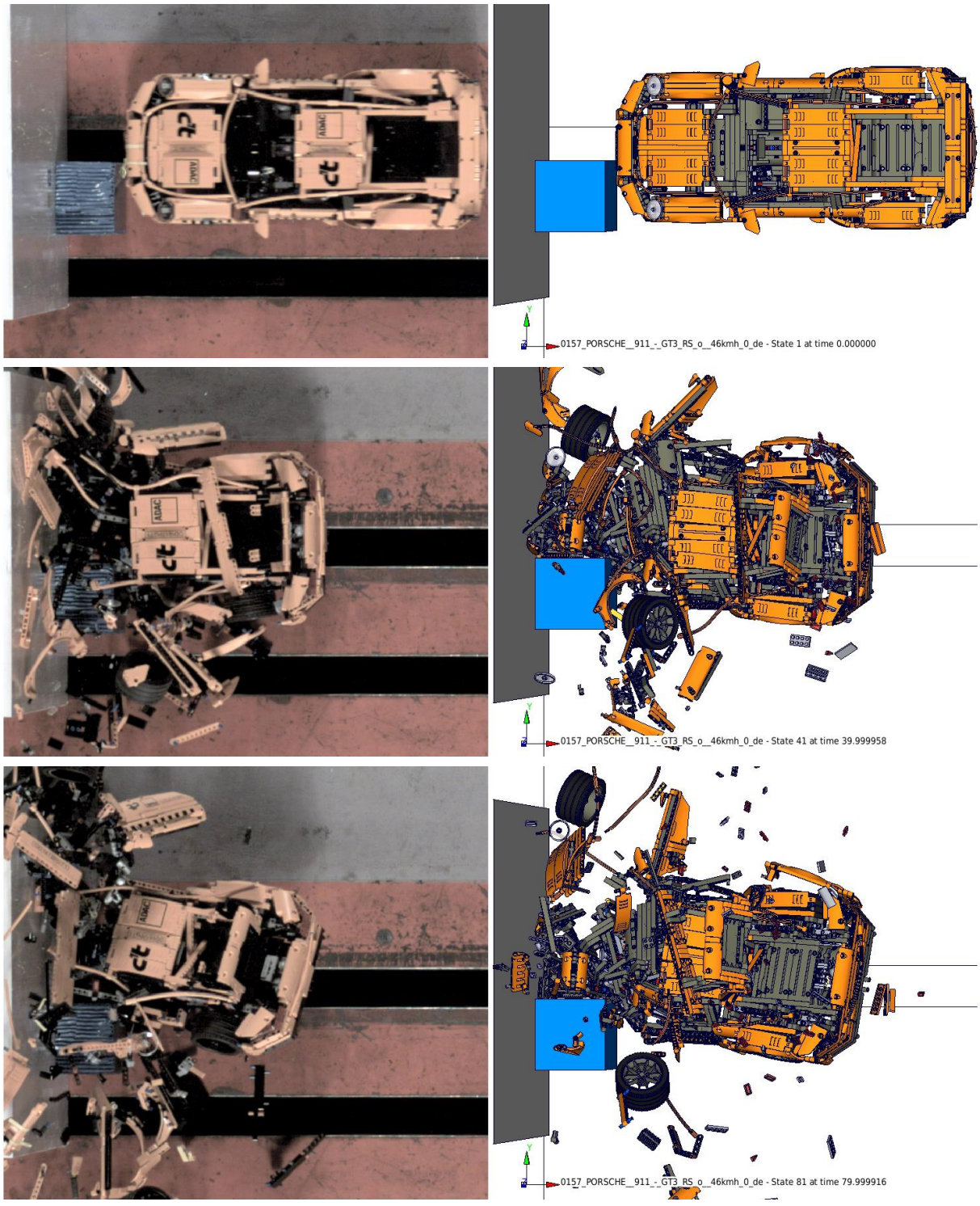

Fig. 21: Current simulation results compared to images from test videos at 0, 40 and 80 ms

6 Significance for the software development process of LoCoX

The current SPDM product offered by the SCALE GmbH is the LoCo2 rich client which has been in development since 2010. This application is showing its age given that the technological decisions are outdated, in some respects, when compared to modern standards.
For this reason, there is work ongoing to derive a new client application, referred to as LoCoX, which uses the old solution as basis. With LoCoX our customers will see an update of the implementation language from Python 2 (currently being phased out) to Python 3 while at the same time replacing the user interface library from wxWidgets[30] to Qt5[31]. Furthermore, in preparation for a web based solution, the development team is also replacing the core components of the user interface with web based technology.
The development of LoCo2 benefited from immediate user feedback whilst being used in the production environment of our customers. This was enabled by applying an agile release cycle. Developing LoCoX lacks this advantage, since its core features are still in active development and only a subset is deemed ready for rollout. However, feedback from workshops as well as tests from key users, albeit not fully, iron out these circumstances. While enjoying the liberty of polishing the concepts behind the different features that will be part of the final release of LoCoX, there is the risk these features do not cover all necessary functionalities for an active production environment.
For this reason, the LEGO® simulation efforts provide a welcome opportunity to test drive LoCoX for reliability, functionality and, especially, performance in a production environment. Performance, as mentioned in previous chapters, represents a major challenge when dealing with a high number of parts for a single model. Given that the number of parts in a LEGO® model exceeds those of a real car by a significant margin, demonstrating good performance for this project provides for some buffer for “real” applications.

7 Summary

Creating this model was a lot of fun because of the teamwork involved. Many of the colleagues who helped in making this possible worked on the model during their spare time or whenever there was some time in between real project work. Solving the problems associated with the process and the LS-DYNA® model is quite some pleasure on its own and doing all this with an actual LEGO® model made it even more interesting and enjoyable.
The current setup with LoCo makes it now relatively easy to setup a simulation model from LEGO® models. However, the usability of all this is not quite there yet such that young students or non FEA affine people can handle it. The final goal would be that one could download models from the OMR of Ldraw[12], bricklink.com[8] or create an own model, define easily some boundary conditions and run crash simulations. This will require further improvements of our Simulation Data Management System LoCo that might be achievable in the present development path of LoCoX - the successor of LoCo. At least this will be the benchmark for the usability and ease of use for the new user interface.
The end of this journey might be that simulation has become a child’s play in order to influence new generations and pass on the fascination of physics and simulation.

8 Literature and references

[1] M. Thiele: "Investigation of the crash behavior of a LEGO® car ", FEA Information Engineering Solutions Volume 8, Issue 2, February 2019, pages 42-49
[2] S. Hansen: "Lass krachen! Der Lego-Porsche-Crash bei c't", issue 12, 2017, pages 74-79
[3] R. Bitsche, M. Thiele, T. Landschoff, M. Koch: "Recent Developments in LoCo - Instant Collaboration in Simulation Data Management", 11th European LS-DYNA Conference 2017, Salzburg, Austria
[4] heise online, 23 May 2017, "LEGO Porsche Crashtest in Slow-Motion", [online] Available at: https://youtu.be/dCPWPj4JHqg
[5] Livermore Software Technology Corporation (1987). LS-DYNA (Version R931) [Software], http://lstc.com/products/ls-dyna
[6] SCALE GmbH (2014). LoCo (Version 1.89) [Software], https://www.scale.eu/en/products/loco
[7] LDraw.org, http://www.ldraw.org
[8] BrickLink Limited, https://www.bricklink.com
[9] Roland Melkert (2011). LDCad (Version 1.6b) [Software], http://www.melkert.net/LDCad
[10] Leonardo Zide (1998). LeoCAD (Version 18.02) [Software], https://www.leocad.org
[11] BrickLink Limited (2018). Studio (Version 2.0.1_93) [Software], https://www.bricklink.com/v3/studio/download.page
[12] LDraw OMR (Official Model Repository), http://omr.ldraw.org
[13] Travis Cobbs (2009). LDView (Version 4.2) [Software], http://ldview.sourceforge.net
[14] The Etherpad Foundation (2008). Etherpad (Version 1.7.5) [Software], https://etherpad.org
[15] Google LLC (2006). Google Docs [Software], https://www.google.com/docs
[16] Collabora Ltd (2005). Collabora Online (Version 4.0.3) [Software], https://www.collaboraoffice.com
[17] P. Hurbain, 2016, "42056-1 - Porsche 911 GT3 RS", CAD Data from LDraw OMR, [online] Available at: http://omr.ldraw.org/set/42056-1
[18] CCAL version 2.0, http://creativecommons.org/licenses/by/2.0
[19] dk, 30 Jul. 2016, "LEGO Technic - Porsche 911 GT3 RS (42056)", CAD Data from GRABCAD, [online] Available at: https://grabcad.com/library/lego-technic-porsche-911-gt3-rs-42056-1
[20] dk, https://grabcad.com/dk
[21] Livermore Software Technology Corporation (1987). LS-PREPOST (Version 4.6) [Software], http://lstc.com/products/ls-prepost
[22] Gesellschaft für Numerische Simulation mbH (2004). Animator (Version 2.4.0) [Software], https://gns-mbh.com/products/animator
[23] BETA CAE Systems (1990). META (Version 15.1.0) [Software], https://www.beta-cae.com/meta.htm
[24] BETA CAE Systems (1990). ANSA (Version 18.1.0) [Software], https://www.beta-cea.com/ansa.html
[25] M. Thiele, 4 Dec. 2018, "LEGO Crash 25% offset barrier 17km/h: left camera compared to simulation", [online] Available at: https://youtu.be/fYfpCnJf0Ak
[26] M. Thiele, 4 Dec. 2018, "LEGO Crash 25% offset barrier 17km/h: top camera compared to simulation", [online] Available at: https://youtu.be/jwsiMlig1Ds
[27] heise online, 17 May 2017, "Lego Porsche Crashtest Teaser", [online] Available at: https://youtu.be/LyJ5B---Zdo
[28] heise online, 28 May 2017, "Lego-Crash: Making Of", [online] Available at: https://youtu.be/ZeBReuh4tP4?t=1100
[29] The FreeCAD Team (2001). FreeCAD (Version 0.18) [Software], https://www.freecadweb.org
[30] Julian Smart (1992). WxWidgets (Version 3.1.2) [Software], https://www.wxwidgets.org
[31] Qt Group (2014). Qt5 (Version 5.12) [Software], https://www.qt.io