Aligning the Element Orientation for Building up Simulation Models of Fiber Reinforced Structures Automatically

Andreas Wünsch[1], Robert Dienemann[2], Andreas Meyer[1] and Sándor Vajna[1]

[1]Chair of Information Technologies in Mechanical Engineering, Otto-von-Guericke University Magdeburg
[2]Chair of Optimization of Mechanical Structures, University of Wuppertal

Keywords:

Batch meshing, fiber reinforced materials, material axes

Abstract

The simulation of products that are made of fiber reinforced materials (FRM) is very useful to identify mechanical properties of these products in the early stages of product development. However, building up these models is a very time consuming process.

Automatic batch meshing can be used to reduce this time. With this process the order of the nodes of each finite element is unstructured, especially on complex shapes and free-form surfaces. However, when applying batch meshing the element orientation can't be used to define the material orientation.

We developed a code that aligns an existing mesh by selecting one or more start elements. Each element is aligned by looking at the elements next to it or at the boundaries of the surface. To verify we applied this code successfully after the batch meshing process to build up models for crash simulation in LS-DYNA. Since the code works directly on the ASCII-based FE-files, it can be adapted very easily to use it with other FE software, e.g. Abaqus or Nastran.

1 Introduction

It is more and more worth to use FRM for lightweight design in current products. E.g. [1] shows that savings of energy consumption and costs through lightweight design of electric vehicles enable the use of such lightweight measures, which previously had to be excluded.

In order to be able to use the full potential of FRM it is necessary to take into account their special design features early in the product development process. The mechanical properties can be calculated by the means of FEM analysis, which is the basis of simulation and automated optimization.

In contrast to isotropic materials, the anisotropic and orthotropic characteristics of FRM make it necessary not only to optimize the topology and the shape of these structures but also to optimize the size, placement and orientation of the fibers within the material. These optimizations must consider special manufacturing constraints. However, the manual creation of FEM simulation models, which are required for optimization, is very labor-intensive. So especially when using genetic optimization algorithms, where a lot of simulation models are needed, an automated model generation is almost mandatory [2].

One way to automate the subprocess of meshing is the use of automatic batch meshing. With this process the order of the nodes of each finite element is unstructured, especially on complex shapes and free-form surfaces. Consequently the definition of material orientation based on the element orientation is not possible just with batch meshing.

In LS-DYNA three options are available to define the material orientation for the simulation of fiber reinforced materials using shell meshes. These options can be selected by the parameter AOPT [3]. We show that none of these options is suitable for building up simulation models automatically after a batch meshing process. Due to this reason we decided to use the element edges to define the material orientation (AOPT=0) and developed a code that aligns an existing mesh by selecting one or more start elements.

2 Batch Meshing

The objective of this study is to build up simulation models of FRM automatically, which includes automatic meshing as well. One method to mesh geometry automatically is batch meshing. It reduces time and cost for meshing and improves the robustness of the created simulation models. Due to this fact it is a very efficient way to create 2D meshes with good quality by using tria or quad elements compared to manual meshing.

Several studies investigated the behavior of hand-meshed and automatically meshed models in crash simulations, including the effects of element size and meshing strategy ([4], [5]). The characteristics and the quality of the mesh show a strong impact on the results of numerical simulations. The automatically meshed models show less scattering and reduced sensitivity in the results than the models that were created manually. Thus, batch meshing as a fully automated process reduces the human factor in the meshing process [5].

For this study Altair BatchMesher was applied. It performs the fully automated cleanup of the geometry and the mesh generation, which is iteratively driven by a quality index to generate an optimal mesh [4]. The geometry cleanup and the mesh generation run automatically according to predefined cleanup parameters and element criteria.

The application of BatchMesher in the context of an automatic setup of simulation models is shown in Fig. 1. After the batch meshing process additional elements, e.g. sets, loadcases, and constraints can be defined.

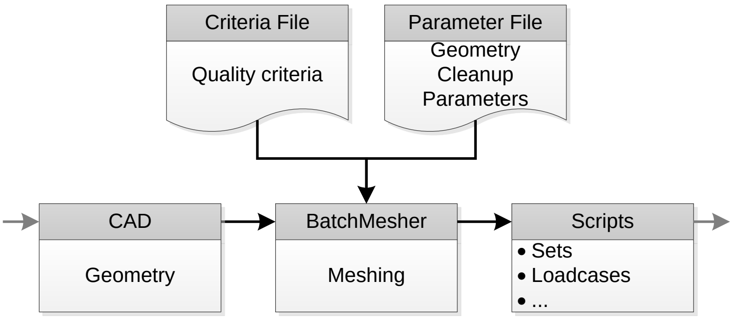

Criteria File
Quality criteria
Parameter File
Geometry Cleanup Parameters
CAD
Geometry
BatchMesher
Meshing
Scripts
- Sets
- Loadcases
- ...

Fig. 1: The application of BatchMesher in the context of an automatic setup of simulation models

By investigating the material axes of different meshes we found, that the material axes are unstructured and not suitable to use them directly to define vectors of any orthotropic material (see Fig. 2). To use the material axes to define an orthotropic material the element orientation has to be aligned.

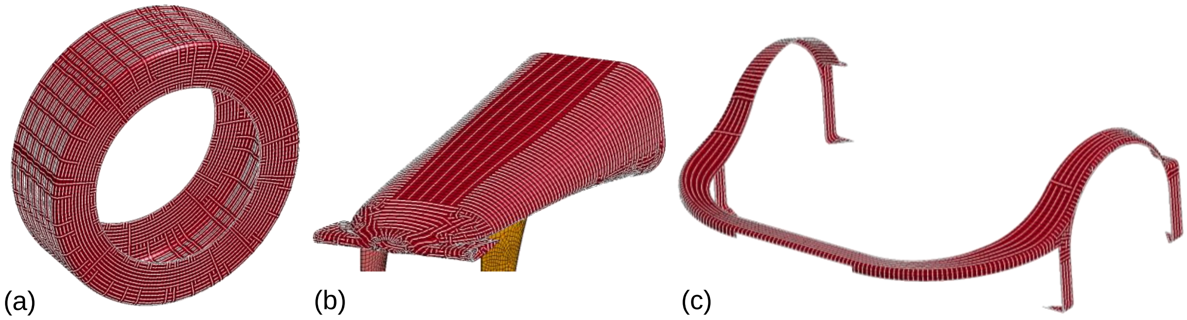


Fig. 2: Unstructured mesh after batch meshing, (a) wheel, (b) nose of a sports car, (c) bumper. White lines show basic material orientation.

3 Material Model

For the simulation of FRM in LS-DYNA we use the material type 54 (**`*MAT_ENHANCED_COMPOSITE_DAMAGE`**). This material type is an enhanced version of the composite model material type 22 (**`*MAT_COMPOSITE_DAMAGE`**) and can be used for different orthotropic materials. However, it is only valid for thin shell elements. Additionally different types of failure can be specified [3]. The material orientation of the composite in different layers can be defined relative to the element's material axes.

There are various options to define the material axes for using 2D meshes, which are determined by the AOPT parameter [3]:

- AOPT=0.0: This option determines a locally orthotropic material in which the material axes are defined by the element nodes N1, N2, and N4 (see Table 1). The axes could be rotated by an angle (MANGLE) about the element normal.
- AOPT=2.0: This setting determines a globally orthotropic material in which the axes determined by a vector defined in the material type.
- AOPT=3.0: A locally orthotropic material is defined. The material axes are determined by the cross product of a predefined vector with the element normal. The axes could be rotated by an angle (MANGLE) about the element normal.
- AOPT<0.0: The absolute value of this option indicates to an ID number of a coordinate system.

To validate the application of these options we used the model of a simple wheel as previously illustrated in Fig. 2. As shown in Table 1 none of the described options to define the material axes led to suitable results in the case of keeping the wheel as a single part and avoiding to split it in different sections and defining the material axes of these sections separately. Note that the material axes using the options AOPT=2.0 and AOPT=3.0 are shown in dependency of three different vectors to define the axes. Using AOPT=0.0 there is no possibility to use a specified vector.

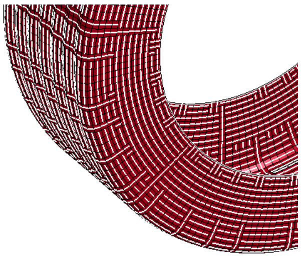
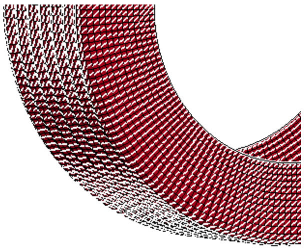
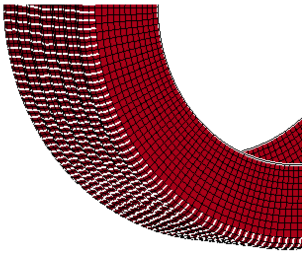
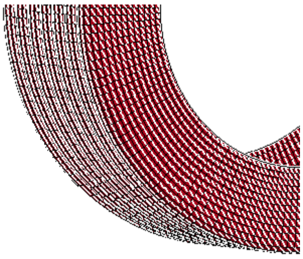
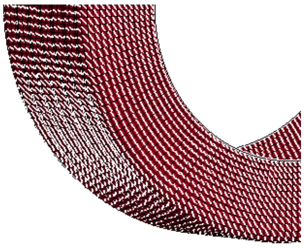
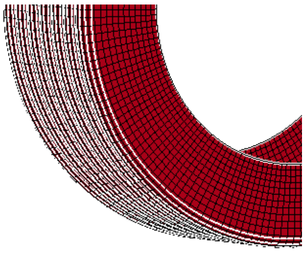
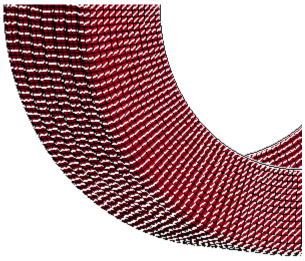
AOPT	**Vector = (1,0,0)**	**Vector = (0,1,0)**	**Vector = (0,0,1)**
AOPT=0.0			
AOPT=2.0			
AOPT=3.0			

Table 1: Element axes determined by different options applied on the mesh of a simple wheel model

Due to the previously described results we can show that with the used methods and tools it is not possible to use batch meshing and one of the options in LS-DYNA to build up simulation models of FRM automatically without adapting the mesh.

We chose the option AOPT=0.0 to determine the material axes. In this option the material axes are defined by the vector between the nodes N1 and N2 (direction A), and by the vector between the nodes N1 and N4 (direction B). These vectors for one single element are shown in Fig. 3.

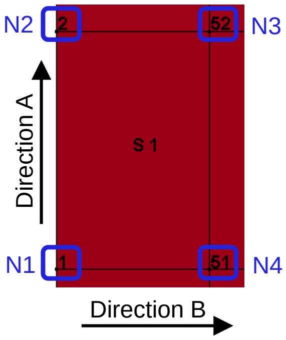


Fig. 3: Material axes determined by element nodes N1, N2, and N4

4 The Algorithm

The basic idea of the algorithm is a successive alignment of elements, which are next to the elements already aligned (see Fig. 4). In the algorithm two lists are used: One with the aligned elements, which initially consists of the start element(s), and one with the unaligned elements. If an unaligned element shares at least two nodes with an already aligned element, it is called neighbor element. The first step to adjust the unaligned neighbor element is the calculation of the angles between all element edges of this element and the material orientation of the aligned neighbor element. In order to find an element edge direction a vector is computed by subtracting the position vectors of the corresponding nodes.

The element will be aligned, if the smallest angle difference between one of the element edges and the material orientation in the neighborhood goes below the defined angle limit (10° by default). If that is not the case the search for further unaligned neighbor elements continues. If no element is found for alignment the angle limit will be increased.

This algorithm continues until the angle limit exceeds 90°. This means all elements of connected mesh regions with at least one start element and consisting of quad and/or tria elements are aligned.

In Fig. 4 a simple mesh is shown as an example for the first 9 steps of the alignment process and the final orientation of the material axes.

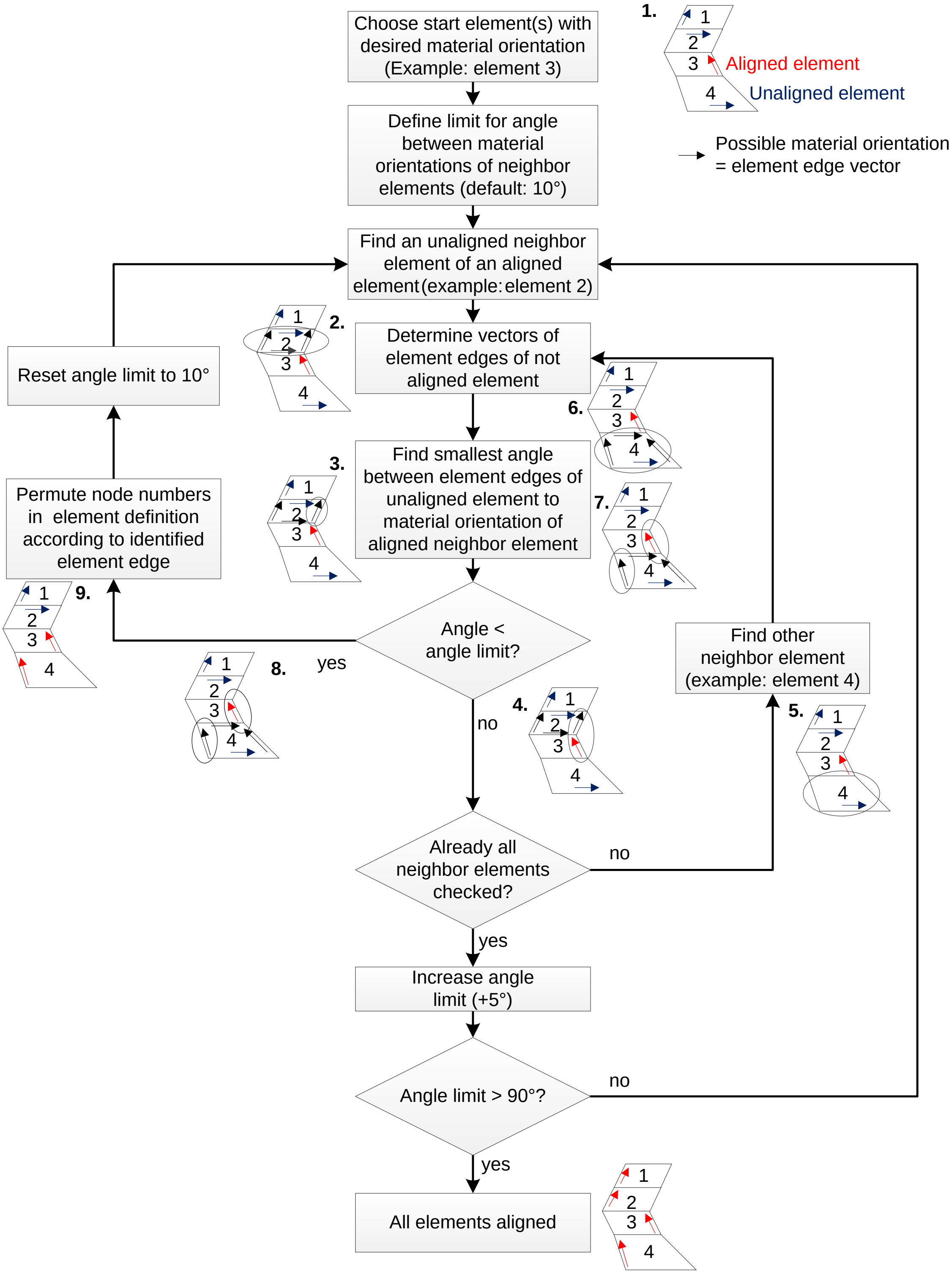

Choose start element(s) with desired material orientation (Example: element 3)
Define limit for angle between material orientations of neighbor elements (default: 10°)
Find an unaligned neighbor element of an aligned element (example: element 2)
Determine vectors of element edges of not aligned element
Find smallest angle between element edges of unaligned element to material orientation of aligned neighbor element
Angle < angle limit?
yes
no
Permute node numbers in element definition according to identified element edge
Reset angle limit to 10°
Already all neighbor elements checked?
no
yes
Find other neighbor element (example: element 4)
Increase angle limit (+5°)
Angle limit > 90°?
no
yes
All elements aligned
1.
Aligned element
Unaligned element
Possible material orientation = element edge vector
2.
3.
4.
5.
6.
7.
8.
9.
1
2
3
4

Fig. 4: Operating principle of the aligning algorithm

The algorithm was programmed in Python 3.3 that is a free programming language with many useful libraries for scientific and engineering work [6]. In our opinion Python is a very intuitive and easy-to-learn programming language as well. We made good experiences with this language in several projects. The processing time of Python scripts is very fast and due to the fact that the code hast not to be compiled it can be adapted very fast as well.

5 Examples and Validation

To validate the developed algorithm we tested it on different examples. The results are shown in the following subsection. Furthermore, we investigated the performance of the algorithm related to the choice of the start element. This investigation leads to the need of regarding more than one start elements to avoid subdivisions of the aligned mesh. At the end of this section the investigation of the processing time is presented.

5.1 Examples

In the following some examples are shown, which demonstrate the success and the characteristics of the introduced algorithm.

The first example shown in Fig. 5 is the wheel (see Fig. 5). In this case a very regularly, exclusively with quad elements meshed part results in a perfect material orientation all over the model. Because the wheel mesh is axis-symmetric, any start element results also in a perfectly axisymmetric material orientation. It depends on the start element's material direction whether the material orientation is along or across the wheel's rotation direction.

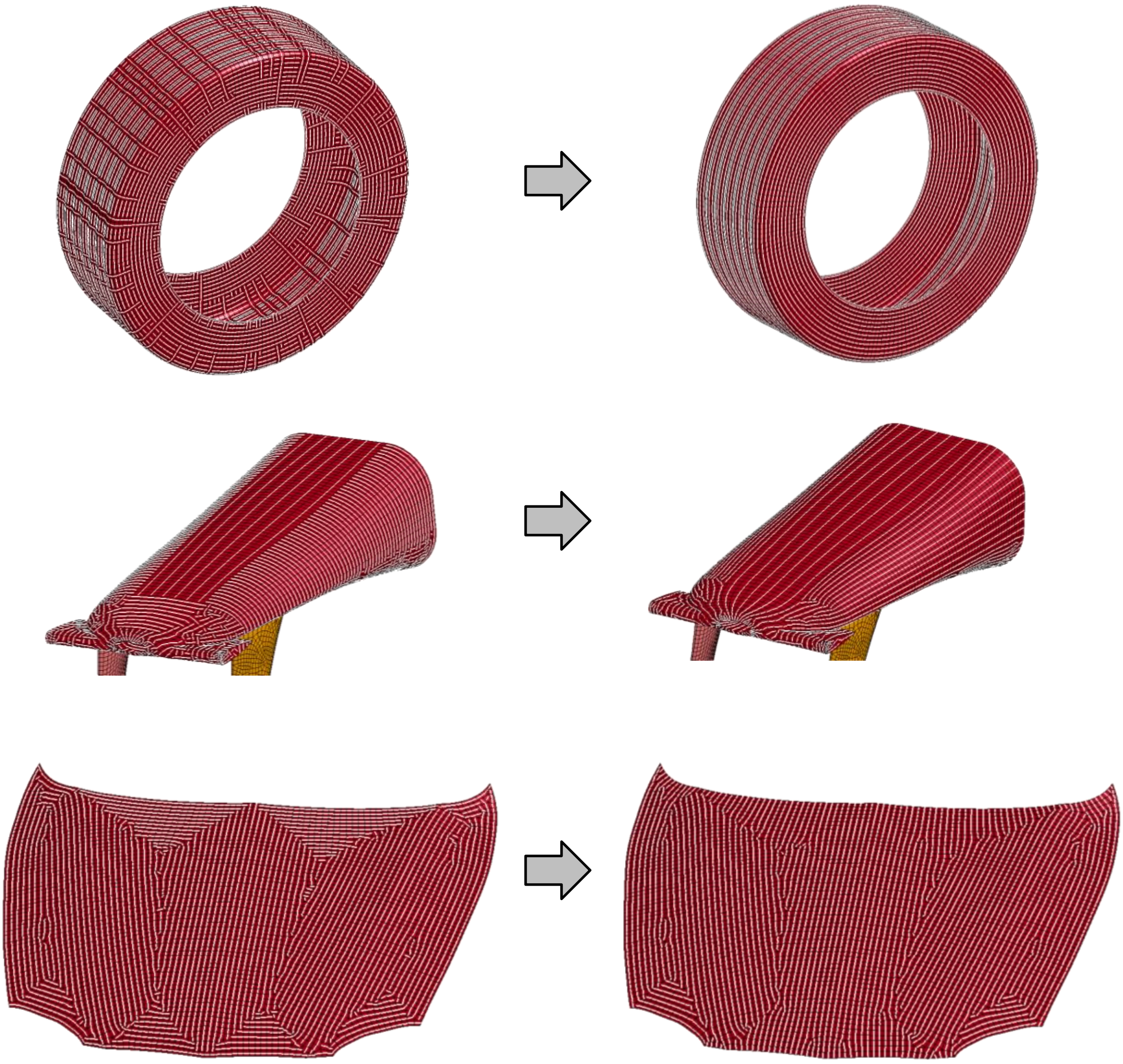


Fig. 5: Examples before (left hand side) and after (right hand side) aligning the elements

A more complex geometry is provided by the sport car nose in Fig. 5. The material orientation of every single part has to be aligned separately. The algorithm achieves an understandable material orientation without manual interference. The algorithm's result in this example depends highly on the choice of the start element(s). By choosing a start element far away from the symmetry plane, the material orientations can become asymmetric although the mesh is symmetric. In regions with a high percentage of tria elements the material directions can differ a lot between two neighbor elements.

The hood shows that the material axes are especially good along geometric edges of the part, where the mesh is highly regular. The provided material orientation of the whole part is a good basis for a producible optimized design of a long-fiber-reinforced hood. Nevertheless, the material orientation of such an extensive surface can’t be chosen intuitively. Consequently the material orientation algorithm can’t replace the manual process of finding a reasonable and producible material orientation.

5.2 Start Elements

The dependency of the material orientations on the start element is shown in Figure 6. The material orientation can depend on the start element’s position, its material orientation, and the mesh itself.

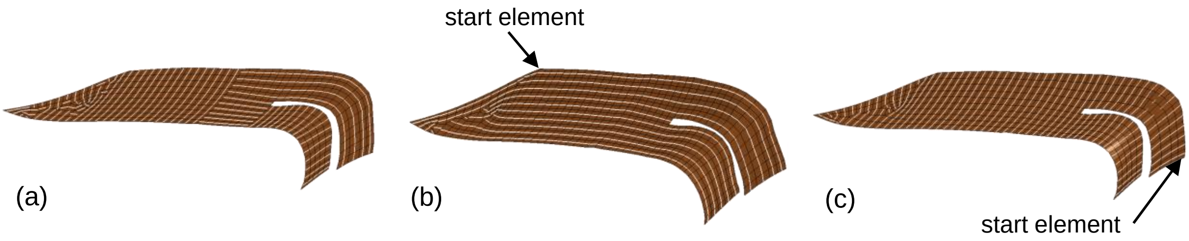


Fig. 6: Dependency from the start element, (a) before, (b) and (c) after aligning the elements

Figure 7 shows the advantage of choosing more than one start element. In (a) the initial material orientation is shown. In (b) a start element at the symmetry plane was selected. Due to a better performance and manufacturability in (c) one start element per strut highlighted by the arrows were chosen. This results in a better material orientation around the start elements, but produces sharp changes of the material orientation elsewhere (d).

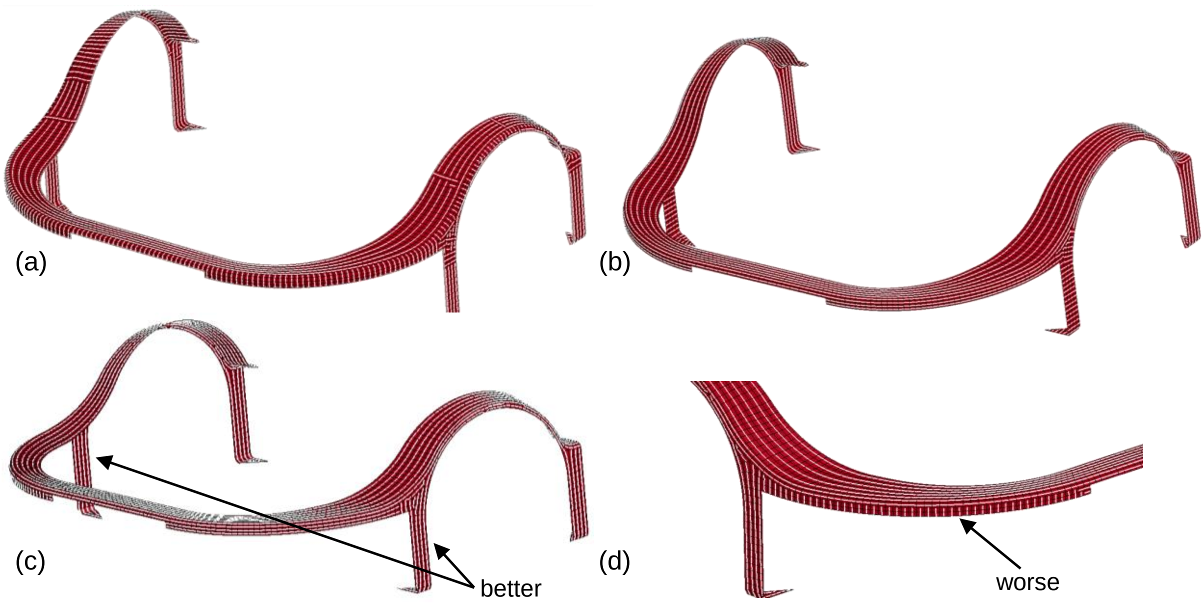


Fig. 7: Using two start elements, (a) before, (b), (c) and (d) after aligning the elements

The algorithm with several start elements works exactly as the one with only a single start element. Initially the start elements are taken from the list of unaligned elements to the list of aligned elements as described in chapter 4.

5.3 Processing time

To analyze the algorithm's efficiency we measured its processing time on different parts with different numbers of elements. As shown in Fig. 8 the processing time increases with the number of elements which is obvious because the algorithm iterates over the elements. Furthermore, the processing time depends on the part's geometrical shape and the initial element orientation of the part. If the initial element orientation is close to the final orientation the algorithm do not need to permute as many elements as in the case of a disadvantageous initial element orientation.

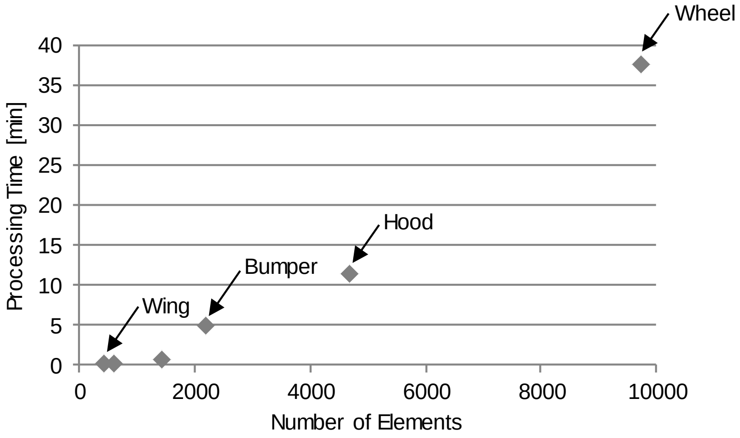


Fig. 8: Processing time of and number of elements of different example parts

6 Discussion

The algorithm developed here aligns the element axes of a FE model following one or more common direction(s) that is determined by one or more start element(s). The algorithm was tested on different parts and scenarios. The results show that the algorithm works very well if the initial mesh is very regular and has a good quality. This is mostly the case in the simulation of extrusion parts and parts with tangent continuous edges.

The example of the hood (see Fig. 5) shows that the material axes are especially good along geometric edges of the part, where the mesh is highly regular. When the edges are not tangent continuous and are connected in small angles (e.g. 30°) the mesh becomes non-regular and the material axes will be non-regular, too. In this case the use of the algorithm is not recommended.

Considering the processing time the introduced algorithm could be improved. Due to the fact that it works with two included iterations over the elements makes the alignment quite costly. This approach of using two included iterations could be improved as well.

7 Summary and conclusions

In this paper we introduced a code that aligns an existing mesh by selecting one or more start elements. Each element is aligned by looking at the elements next to it. To verify we tested this code in different scenarios. It was successfully applied after the batch meshing process to build up simulation models for LS-DYNA automatically. Using the developed code after automatic batch meshing reduces the time for building up simulation models of fiber reinforced structures.

The developed code works efficiently to align the material axes of an existing mesh by selecting one or more start elements. The results depend strongly on the selected start elements and the initial mesh. Using more than one start element can avoid subdivisions of the aligned mesh. The investigation of the processing time of the algorithm leads to the conclusion that the efficiency of the code could be improved. Furthermore, the code uses two included iterations over the elements that makes it quite costly.

Instead of defining one or more reference elements at the beginning another approach could be the definition of a start vector or a spline to determine the direction of the material axes. Since the code works directly on the ASCII-based FE-files, it can be adapted very easily to use it with other FE-software, e.g. Abaqus or Nastran.

8 Literature

[1] Eckstein, L.; Schmitt, F.; Hartmann, B.: "Leichtbau bei Elektrofahrzeugen", IN: ATZ, 2010

[2] Rahul; Sandeep, G.; Chakraborty, D.; Dutta, A.: ”Multi-objective optimization of hybrid laminates subjected to transverse impact”, IN: Composite Structures 73 , 2006, pp. 360-369

[3] Livermore Software Technology Corporation (LSTC): "LS-DYNA Keyword User's Manual", Volume II, February 2012, pp. 1-46, 1-233 - 1-235

[4] Ratzel, M.; Du Bois, P.; Fredriksson L. A.; Schneider, D.: "A study of mesh sensitivity for crash simulations: comparison of manually and batch meshed models", 4. LS-DYNA Anwenderforum, 2005, pp. 25, 26, 32

[5] Seybold, J.; Bäuerle, C.; Eick, M.; Ratzel, M.: "BatchMeshing and CAE Data Management as Key Technologies for Six Sigma Compliant CAE Processes for LS-DYNA Simulations", 5. LS-DYNA Anwenderforum, 2006, pp. 15, 21

[6] www.python.org, (27.09.2014)