Automatic CAE Mesh and Analysis Results Export for Presentation in Virtual Reality Environments

Dimo Chotrov, Angel Bachvarov, Stoyan Maleshkov

***Abstract*— Using Virtual Reality (VR) interactive and immersive presentation of 3D CAE meshes and analysis results facilitates and enhances the engineering decision making which reduces the duration of the product development cycle leading to decreased product development costs and increased competitiveness. Due to the different purposes of CAE and VR, CAE aiming at precision and VR – at interactivity and immersion, there are significant differences in the input data formats which makes the transition between CAE and VR not a trivial task. The CAE data usually has to be exported to an intermediate format and then has to be imported in the VR environment. Here we propose a way to automate the workflow of exporting CAE meshes and results and importing them in VR directly from a CAE application. This is made possible through an add-in for the respective CAE application using its native Application Programming Interface (API) or Software Development Kit (SDK). Through the interface of the developed add-in the CAE application user can specify how to present the current CAE mesh and analysis results within VR.**

***Index Terms*— CAE Analysis Results Presentation, CAE VR Integration, Virtual Reality**

I. Introduction

Computer Aided Engineering (CAE) is an important integral part of the product development process for industry manufacturers. The CAE applications are constantly evolving towards a “virtualization” in term of using of immersion and interaction for presentation of the results obtained [1]. The core here is the idea for the correspondence between the real world and the artificially created virtual space in which the user can be immersed so that he/she will be able to experience more than in the physical space [2]. This is made in order to facilitate the interpretation of the CAE results obtained and to enhance the engineering design making on early stages of the product development process.

The Virtual Reality represents an effective medium for product validation and verification before “materializing” the product physical prototype which enables fast and uncomplicated problems identification and finding the necessary improvements. This shortens the development cycle and cuts down product development expenses respectively.

The ability of the Virtual Reality to "immerse" the user is achieved through a multi-modal presentation in terms of using multiple sensory channels (vision, hearing, touch). The information received from the various perception channels is processed in different ways and the individual presentation elements are combined and overlapped ensuring complementarity and redundancy, which means that the use of multimodality leads to significant expansion and intensification of information content transmission within the interaction between user and the object which is experienced. The users perceive this information as clear, logical, consistent and significant. With respect to above mentioned the immersive VR systems will be beneficial at post-processing and visualization of the results of engineering analyses and simulations.

II. Problem Statement

The presentation of the CAE meshes and analysis results in a VR environment is still not a trivial task. [3] and [4] give a detailed overview on the research topics in the field of VR based post-processing of CAE results. [5] outlines three important issues related to VR presentation of the CAE results as: (i) transformation of results to a format easily interpretable by a VR application; (ii) the computational problems related to visualization of a huge amount of continuous data generated; (iii) usability of the user experience and the immersion level offered.

The most of the currently available commercial CAE software packages don’t include any specific functionalities for export of the CAE meshes and analysis results to the VR applications. Typically to transfer CAE data to a VR environment usually these data should be converted to a specific VR format or to some other intermediate format supported by both the CAE and the VR applications [6]. As the two systems are also often physically different systems, the transition process requires exporting data from the CAE system, transferring the exported data to the VR system and importing them in the visualization application. A process which may be cumbersome and tedious, especially if performed frequently, as the transition process can only be done one-way from the CAE application to the VR environment. All this is time consuming and requires enormous efforts. It is less the time needed for executing the CAE simulation rather than the preparation of the CAE results for presentation within VR. Thus an automation of the CAE/VR pipeline is needed in order to be able to use immersive VR presentation as an efficient and useful tool for support of the routine

Manuscript received Dec 21, 2016

Dimo Chotrov, Department of Programming and Computer Technologies, Technical University Sofia, Sofia, Bulgaria
Angel Bachvarov, Faculty for Mechanical Engineering, Technical University Sofia, Sofia, Bulgaria
Stoyan Maleshkov, Department of Informatics, New Bulgarian University, Sofia, Bulgaria

engineering activities.

III. Export Automation

We propose a method for automating the export process of CAE mesh and analysis results from a CAE application for exploration and assessment in VR. Currently most of the necessary steps are performed manually by an operator. A detailed graphical presentation of these steps is presented on Figure 1. The operator has to export lists of the nodes, elements of the analysis mesh and analysis results for each performed analysis from the CAE application. These lists are imported in some conversion software which generates a triangle mesh and associates the results with the vertices of the mesh. Follows the setup of the analysis results presentation – whether the results are going to be presented visually, with sonification or with tactile feedback. This setup has been described in detail in our previous article [6]. The results are again exported from the conversion software and imported in a VR application to be explored and analyzed for decision making.

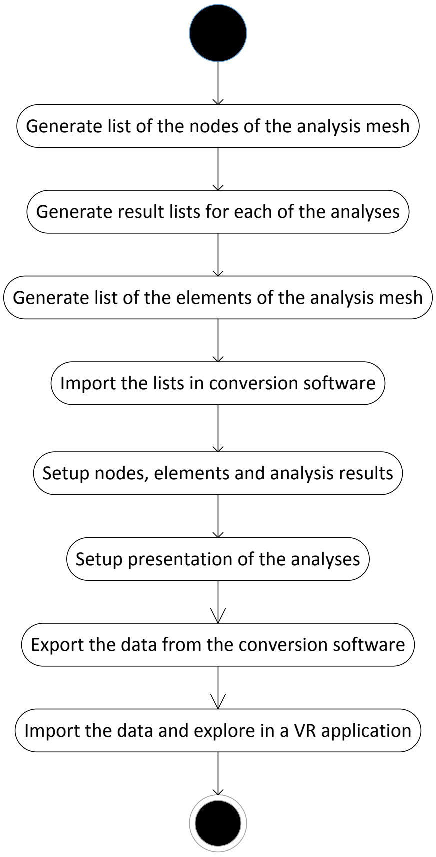

Generate list of the nodes of the analysis mesh
Generate result lists for each of the analyses
Generate list of the elements of the analysis mesh
Import the lists in conversion software
Setup nodes, elements and analysis results
Setup presentation of the analyses
Export the data from the conversion software
Import the data and explore in a VR application

Figure 1. Manual steps for CAE analyses results presentation in a VR application

If the CAE application exposes an API or SDK providing access to its internal data representation these steps can be very easily automated by developing an add-in or extension for the CAE application.

Figure 2 depicts the steps to be executed during the automatic export. Using the CAE application's SDK, the nodes and elements of the CAE mesh can be traversed collecting spatial data and analysis results. During the traversal, a triangular mesh suitable for the visualization of the model can be generated from the elements of the CAE mesh and their types, for example on the Boundary Representation (B-Rep) principle [8]. One such triangulation algorithm based on the elements of the analysis mesh has been described in our previous article [6].

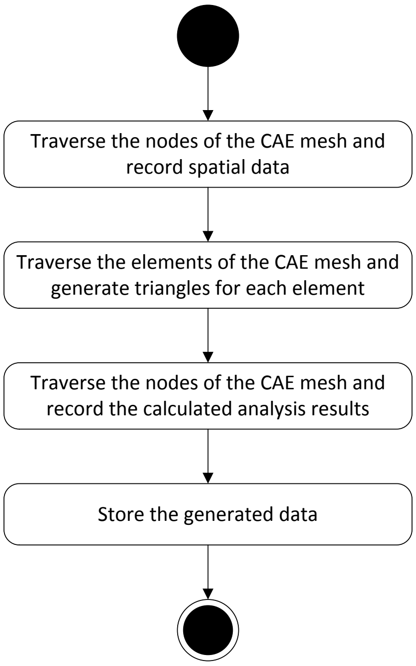

Traverse the nodes of the CAE mesh and record spatial data
Traverse the elements of the CAE mesh and generate triangles for each element
Traverse the nodes of the CAE mesh and record the calculated analysis results
Store the generated data

Figure 2. Export Automation

The data collected during the traversal of the CAE mesh has to be stored in a suitable data structure. The data structure needs to be designed to allow easy data transfer between the two (CAE and VR) applications. It should represent the data in a way suitable for import and easily interpretable by the VR application – geometrical data for the model visualization and analysis results data mapping to the geometrical data. Although not required for the model and analysis results presentation in VR it is desirable that the data structure is able to also store node / element reference identifiers in the CAE mesh to allow easy problem area identification for eventual feedback or corrections in the model after the analysis in VR.

Figure 3 shows the data structure hierarchy we have developed to store the data collected during the CAE mesh node and element traversal. The ***TesselatedBody*** class is used to store the generated triangle mesh. It contains a list of all points (identical to the nodes from the analysis mesh) with their coordinates, optional normals, if such information is available, and a list of triangles described through indices in the points list. The ***CAEAnalysisResults*** class stores analysis results for one performed analysis. The ***CAEBody*** class contains a ***TesselatedBody*** instance and a dictionary with all the analyses performed for the model.

Once the data has been recorded it can be saved in a file to be imported in the VR application or transmitted over a network depending on the available communication interfaces and implementation requirements.

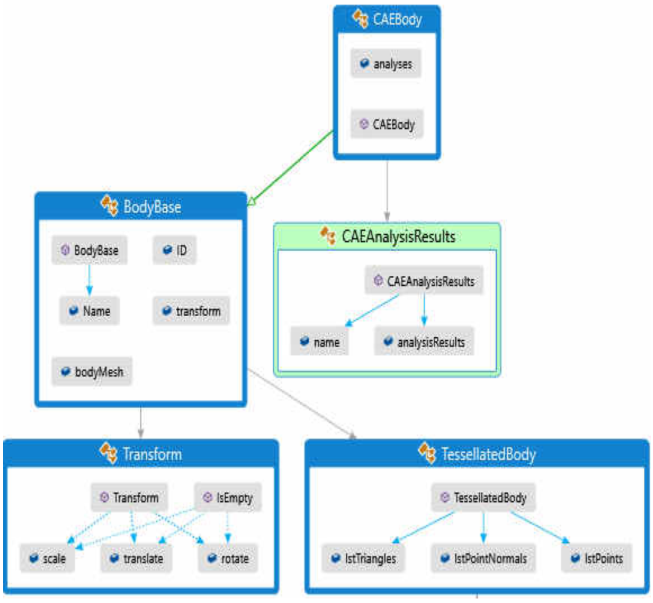


Figure 3. Data structure hierarchy for storing the data collected during the traversal

The VR application itself needs to be able to interpret the received data. It has to transform the triangular mesh to its internal representation for 3D model visualization and to assign presentation stimuli for the presentation of the analysis results. The first one is a trivial task, as applications dealing with 3D model visualization mostly use a triangle or quad mesh of the model. A simple solution for the second task would be to alternate the available presentation possibilities for multiple analyses. A better alternative would be to have a user configuration specifying the kind of presentation to be used for each specific type of analysis or to allow the user to switch between presentation alternatives during runtime.

IV. Experiment Results

For the implementation of the proposed method we have used the ANSYS Application Customization Toolkit (ACT) to develop an extension for ANSYS Mechanical. ANSYS ACT makes it possible to develop a wide range of extensions for the ANSYS family of products like:

- adding toolbars to the graphical user interface
- definition of custom loads and constraints
- definition and calculation of custom results
- third party integration [7].

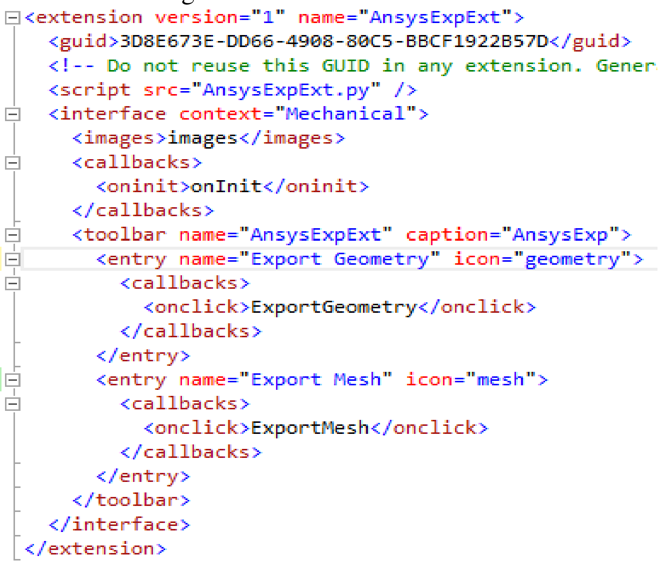
An experimental realization of the proposed method has been developed for the ANSYS Mechanical CAE application and an in-house developed VR application. The implementation of the proposed method falls into the last group of described applications for the ANSYS ACT – third party integration.

For the integration of an extension in the graphical user interface (GUI) of an ANSYS application a .xml file is used, which describes how the extension is incorporated in the GUI, and a IronPython script to handle GUI events and perform the necessary processing. Another possibility is to use .NET assemblies, for example to define custom loads / constraints / results.

In this case the extension was developed using a hybrid approach – an IronPython script which receives the GUI events and then uses C# .NET assemblies for the data processing. This approach was chosen for a number of reasons:

- IronPython had to be used for the GUI event handling. It was not possible to describe in the extension's xml file that an event should be handled by a method in a class in a .NET assembly.
- Using a .NET assembly provides better performance than an IronPython script [7].
- Using a .NET assembly was also preconditioned by the need for the extension and the VR application to use common class libraries to simplify the data transfer from ANSYS Mechanical to the VR application without the need for additional conversion.

The XML file shown on Figure 4 adds a toolbar with caption AnsysExp to the interface of ANSYS Mechanical. The toolbar contains two buttons – one for geometry only export and one for exporting the analysis mesh together with the results from the performed analyses. The resulting toolbar can be seen on Figure 5.



```
<extension version="1" name="AnsysExpExt">
  <guid>3D8E673E-DD66-4908-80C5-BBCF1922B57D</guid>
  <!-- Do not reuse this GUID in any extension. Gener
  <script src="AnsysExpExt.py" />
  <interface context="Mechanical">
    <images>images</images>
    <callbacks>
      <oninit>onInit</oninit>
    </callbacks>
    <toolbar name="AnsysExpExt" caption="AnsysExp">
      <entry name="Export Geometry" icon="geometry">
        <callbacks>
          <onclick>ExportGeometry</onclick>
        </callbacks>
      </entry>
      <entry name="Export Mesh" icon="mesh">
        <callbacks>
          <onclick>ExportMesh</onclick>
        </callbacks>
      </entry>
    </toolbar>
  </interface>
</extension>
```

Figure 4. The XML file used for the AnsysExp extension toolbar in ANSYS Mechanical

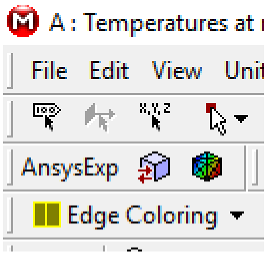


Figure 5. The AnsysExp toolbar in the ANSYS Mechanical GUI

As was mentioned earlier, the CAE export add-in and the VR application use a common class library for the data transfer between the two applications. This is the **CADLib** library on Figure 6 which contains the in the previous part described data structure hierarchy. The **AnsysExp** class library, also shown on Figure 6 contains the actual implementation of the export. The ***ExportMesh*** method of the ***AnsysExp*** class first traverses the nodes and elements of the analysis mesh, generates triangular polygons from each element depending on its type and stores the resulting polygonal mesh in an instance of the ***TessellatedBody*** class.

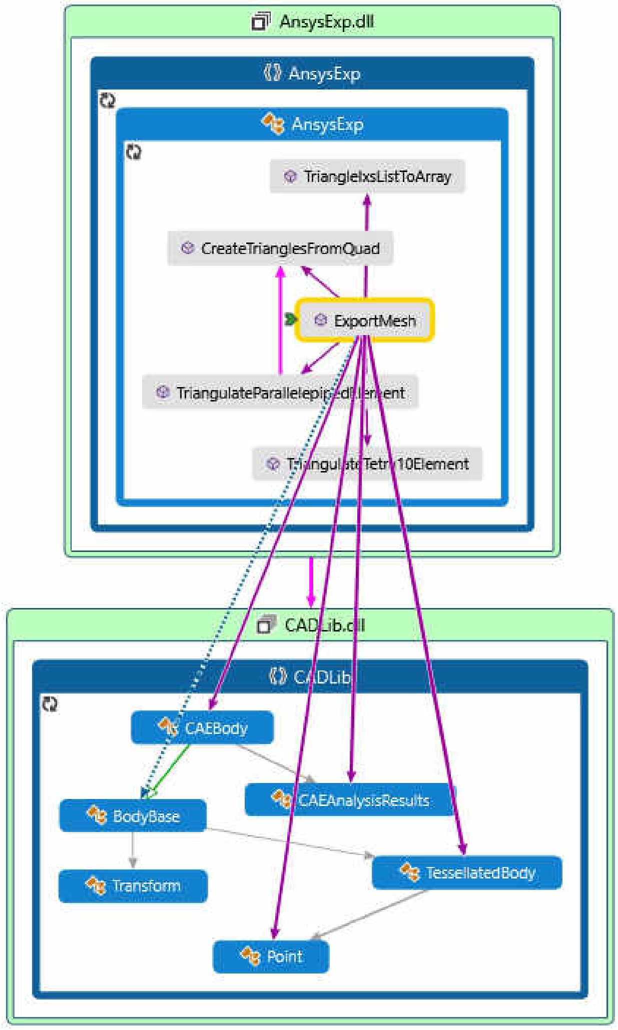


Figure 6. Generation of a CAEBody instance by the ExportMesh method

Follows a second traversal of the analysis mesh during which the results from the performed CAE analyses are stored in instances of the ***CAEAnalysisResults*** class. Both the ***TessellatedBody*** and ***CAEAnalysisResults*** instances are added to an instance of the ***CAEBody*** class. The ***CAEBody*** instance is finally serialized using binary serialization and saved to a file to be imported in the VR application.

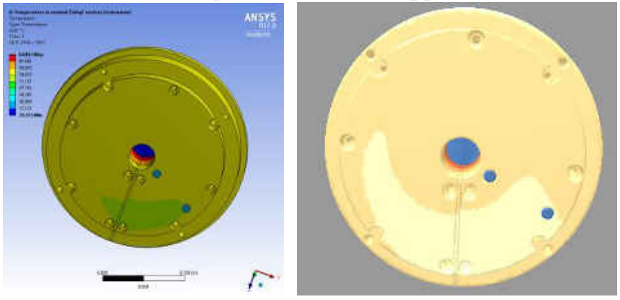


Figure 7. Comparison of model with analysis results in ANSYS Mechanical and in a VR application

The result demonstrating the application of the proposed method for automated export of analysis mesh and results from a CAE application to a VR application can be seen on Figure 7. The figure shows the model of a cooling radiator with results from a temperature distribution analysis in ANSYS Mechanical and the same model and results in our in-house VR application after using the experimental implementation described above. There is dissimilarity in the coloring which is due to difference in the color scales used in the two applications for the visual presentation, as well as the value distribution in the color scales.

Figure 8 shows the same CAE mesh and analysis results after export from ANSYS Mechanical visualized stereoscopically on a VR system with a 2x2 tiled display.

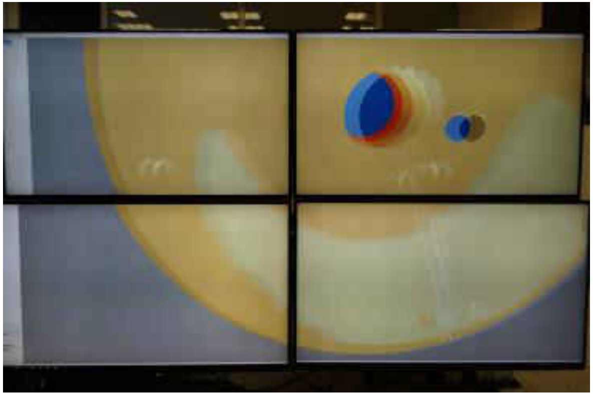


Figure 8. Stereoscopic visualization of the CAE mesh and analysis results on a 2x2 tiled display

CONCLUSION

The proposed method greatly simplifies the otherwise mainly manual export of CAE mesh and analysis results for visualization in virtual environment and improves the workflow for VR supported decision making.

With the help of an extension like the one presented in this paper, the complex process presented on Figure 1 is reduced to two simple steps - export from the CAE application and import in the VR one. This means less time spent performing a trivial but time consuming task and less commitment on the side of the operator.

A next step to even further simplifying the process would be to create a direct interoperability between the CAE and VR applications, allowing the operator to transfer the CAE mesh and analysis results directly to the VR application without the need for data export and import.

REFERENCES

[1] Horvath, I & Vroom, RW (2015). Ubiquitous computer aided design: A broken promise or a sleeping beauty? In. Computer-Aided Design, 59(February), 161-175.

[2] Coyne R. (1999). Technoromanticism: digital narrative, holism, and the romance of the real. Cambridge (MA): MIT Press.

[3] Scherer, S., Wabner, M. (2008). Advanced visualization for finite elements analysis in virtual reality environments, In: International Journal on Interactive Design and Manufacturing, 2:169–173

[4] H. Graf (2011). A "Change 'n Play" Software Architecture Integrating CAD, CAE and Immersive Real-Time Environments, 12th International Conference on Computer-Aided Design and Computer Graphics, Jinan, pp. 3-10.

[5] F. Argese, A. Martini, L. Colizzi, M. Fina, G. Reo, F. Ambrosino, P. Bene (2015). A Virtual Prototyping Platform to Improve CAE Analysis Workflow, Second International Conference Salento AVR 2015, At Lecce, Italy, Augmented and Virtual Reality - Lecture Notes in Computer Science Volume 9254, pp 267-286

[6] Maleshkov S., Chotrov D. (2013). Post-processing of Engineering Analysis Results for Visualization in VR System, IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 2

[7] ANSYS Inc. (2015). ANSYS ACT Developer's Guide, SAS IP Inc.

[8] Krysl, P. and Ortiz, M. (2001), Extraction of boundary representation from surface triangulations. Int. J. Numer. Meth. Engng., 50: 1737–1758. doi:10.1002/nme.94

Dimo Chotrov has received BSc. (2007) and MSc. (2009) degrees in computer systems and technologies from the Technical University Sofia. Finished his PhD in 2015 with subject Integration of CAD and VR applications. Part of the development team of the Technical University Sofia Virtual Reality Laboratory since 2009. Since 2016 an Assistant Professor at the Technical University Sofia.

Angel Bachvarov studied mechanical engineering at the TU-Sofia and specialized in the field of Virtual Engineering (Karlsruhe Institute of Technology) and Mechatronics (TU-Ilmenau). He actively participated in the initial setting up and commissioning of the Laboratory for Virtual Reality of TU-Sofia and now is valuable member of the laboratory team working on development and implementation of concepts and paradigms for introducing VR and AR technologies as supporting tools in the product development routines. His main focus is on product customization, customer co-design, product development and engineering decision making using Virtual Reality, Tangible Interfaces and Virtual Environments.

Stoyan MALESHKOV has Eng. degree in system and control engineering (1975), master in applied mathematics (1977) and PhD in computer aided system design (1981), all received from the Technical University (TU) - Sofia, Bulgaria. Fulbright scholar (1989–1990) at the Interactive Modeling Research Lab, Louisiana State University, Baton Rouge, USA. Assoc. professor (1989-2013) and professor (2014-2016) of computer aided engineering and computer graphics at the TU of Sofia. Department chair (2000-2004) and vice dean (2004-2008), both at the TU Sofia. Head of the Virtual reality lab, TU Sofia (2008-2016). Since 2015: Professor of computer graphics and computer aided design at the New Bulgarian University, Sofia. Major Fields of Scientific Research: Computer graphics, Virtual reality. IEEE member.