Applying Virtual Reality Techniques to Finite Element Solutions

Marc Schöning and Kay Hameyer

Institute of Electrial Machines, RWTH Aachen University, 52056 Aachen, Germany

Electrical machine design resorts more and more to numerical simulation techniques instead of prototyping, so as to reduce development time and costs. In order to replace prototypes advantageously, numerical simulations must be accurate. This is achieved by using the discretization techniques like the finite element method combined with a fine mesh. This results in a large amount of computed data that need to be postprocessed adequately to allow making a correct interpretation of the results and taking the right decisions. Virtual reality environments are suited to match these demands. They offer the possibility of exploring efficiently and in detail the computed results of complicated geometries. In this paper, a postprocessing software exploiting virtual reality capabilities is demonstrated and application examples are shown.

Index Terms—**Electrical machines and drives, software methodology, virtual reality.**

I. Introduction

Today, numerical simulations of electrical devices are becoming increasingly important. The finite element package iMOOSE [3] allows for simulating complex geometries with high precision and resolution in space and time. The postprocessing of the mass of numerical data generated by a 3-D magnetic field computation is not an easy matter. Various ways of visualization like 2-D plots or graphs have been well known for a long time, but the results obtained from the computation of 3-D fields in complex geometries are still difficult to evaluate. This paper demonstrates how the principles of virtual reality can be used to enhance design capabilities. With this technique, users can dive into a virtual representation of 3-D field lines and follow their path through the magnetic circuit, acquiring so an intuitive understanding of how the machine really works. Designers can also use the same technique to spot the location of some design problem and find out the possible causes of observed malfunctioning of the developed system. Virtual reality environments are suited for teaching purposes as well.

II. Computer Graphics Software

The postprocessing software named iMOOSE.trinity.vr uses the graphical package visualization toolkit (VTK) [1] for the visualization of finite element solutions. The VTK is an open source software system for 3-D computer graphics, image processing, and visualization. VTK consists of a C++ class library and several interpreted interface layers including Tcl/Tk, Java, and Python. It supports a wide variety of visualization algorithms for scalar, vector, and tensor quantities with different textures, and advanced modeling techniques such as implicit modeling, polygon reduction, mesh smoothing, cutting, contouring, and Delaunay triangulation. In addition, dozens of imaging algorithms have been directly integrated to allow the user to mix 2-D imaging/3-D graphics algorithms and data. VTK combines a powerful structure, object-oriented programming, platform independence and it is freely available. The object-oriented design of this software is characterized by

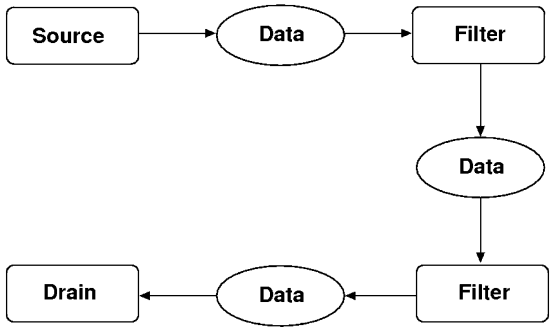

Fig. 1. Visualization pipeline of VTK.

general, easy to use data structures, whose versatility encourages a modular use of algorithms acting as filter objects. The working principle of VTK is based on visualization pipelines (see Fig. 1). Prior to the use of the filter algorithms provided by VTK, the results of the finite element computations must be converted into VTK data structures, more precisely into the so-called unstructured grids, which represents any combinations of points and cells. The main obstacle to using VTK for the visualization of finite element solutions is the discontinuity of the fields at material transitions in electromagnetic devices or in the presence of shocks in fluid dynamics. Finite element packages can handle discontinuous solutions of first or second order whereas the data structures of VTK can only handle first-order continuous solutions. Those discontinuities are, however, physically relevant and must, therefore, appear in the solution plot. When dealing with electromagnetic fields, the association between material boundaries and discontinuities can be utilized to comply with the standards of VTK. Because all physical characteristics are continuous inside a material domain, the solution in a first- or second-order element is continuous and discontinuities only occur at the boundaries between elements belonging to the different material domains. In consequence, the mesh is partitioned into several submeshes, one for each material domain, when converting the iMOOSE data into the VTK data. Boundary nodes are duplicated so that discontinuous solutions at finite element nodes of the original mesh translate into continuous solutions over the different submeshes. The toolkit VISTA [2] is used to display the VTK visualization data in a virtual reality environment. VISTA covers all aspects of

Digital Object Identifier 10.1109/TMAG.2007.916142

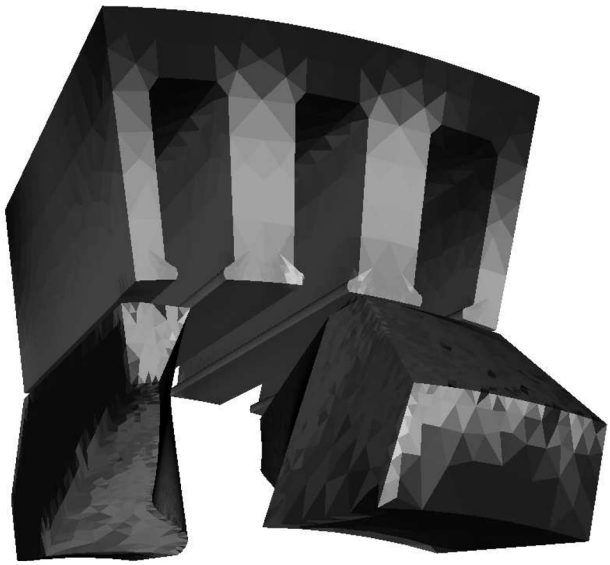

Fig. 2. Flux density distribution in claw pole generator.

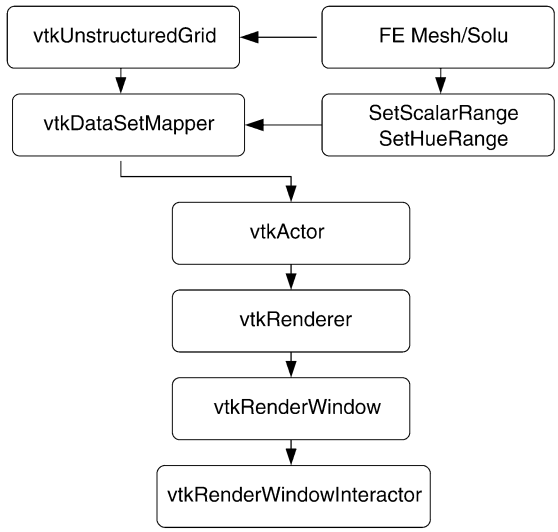

Fig. 3. VTK filter chain for a scalar plot.

virtual reality technology. It is scalable in order to integrate all kinds of display systems, ranging from high-end visualization displays like CAVE-style (see Fig. 9) systems down to standard desktop monitors, and it offers access to a variety of 3-D input devices. Special features like force feedback and 3-D acoustics are also covered. VISTA is implemented as a library, so that virtual reality applications can be developed rapidly for specific purposes. By using additional hardware, like a holobench or a CAVE, combined with additional features like force-feedback system and head position tracking system, the realistic effect of virtual reality environments can be increased significantly.

III. Application

By using VTK, various new visualization methods in the form of filter algorithms are available. This includes the extraction of equipotential surfaces or field lines in 3-D models (see Fig. 4). Furthermore, the deformation and the generation of vectors can be controlled very accurately and additional functionalities like smoothing meshes and decreasing the number of shown vectors are available. All visualizations presented here are created utilizing the nodal or element solution data calculated by the iMOOSE solver environment. Fig. 2 illustrates a flux density plot of a claw pole generator. The necessary VTK filter chain is shown in Fig. 3.

First, the finite element mesh and solution are converted into a *vtkUnstructuredGrid* data set for each material. These grids are mapped to graphics primitives by the *vtkDataSetMapper*, who also maps the scalar range of the finite element solution to a given color range, specified by *SetHueRange*. The next element of the filter chain is the *vtkActor* representing an entity of the rendering scene. In particular, *vtkActor* combines object properties (color, shading type, etc.), geometric definition, and orientation in the world coordinate system. The class *vtkRenderer* is responsible for coordinating its lights, camera, and actors to produce an image. *vtkRenderWindow* ties the rendering process together. This filter manages a window on the display device including functionality for stereoscopic visualizations.

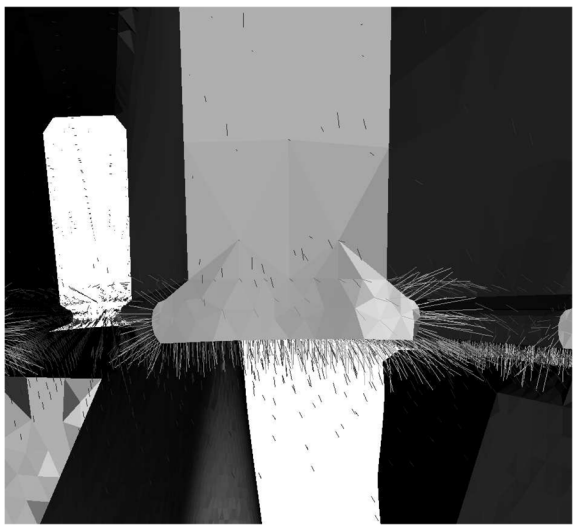

Fig. 4. Field lines in air volume.

Finally, *vtkRenderWindowInteractor* provides a platform-independent interaction mechanism for mouse, key, and time events.

Another example is the visualization of field lines exemplarily in the air gap (see Fig. 4). This visualization mechanism can be used to indicate the flux distribution and, therefore, the flux leakage at every desired position in an simulation model. The corresponding VTK filter chain is shown in Fig. 5.

As mentioned before, the finite element mesh and solution is converted into an *vtkUnstructuredGrid*. Afterwards, the *vtkCellDataToPointData* filter transforms cell data (i.e., data specified per cell) into point data (i.e., data specified at cell points). In this example, the vectorial flux density solution is transformed to scalar values. The method of transformation is based on averaging the data values of all cells using a particular point. *vtkHedgeHog* is the next element of the filter chain creating oriented lines from the input data set at every given point (the mentioned cell points). The line length is controlled by magnitude of the solution time-scale factor and can be colored by scalar data. Because the filter chain converted the data

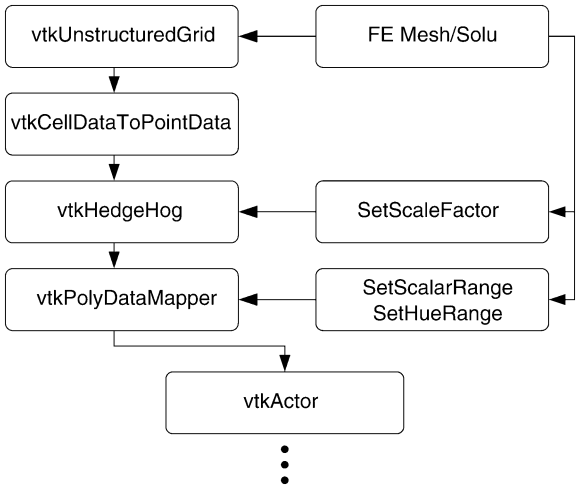

Fig. 5. VTK filter chain for flux lines.

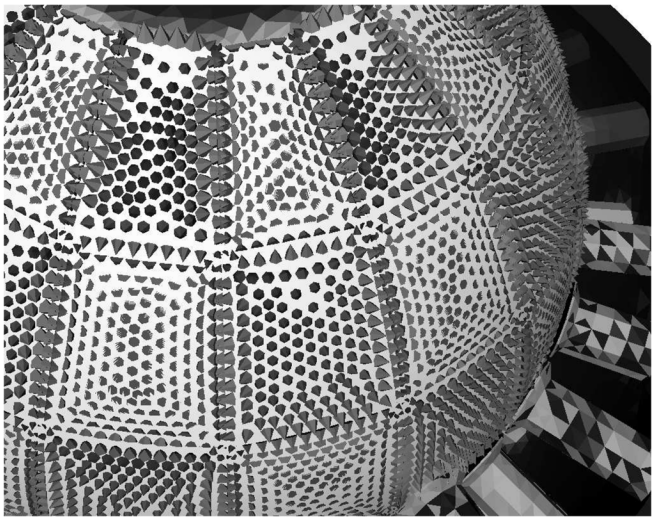

Fig. 6. Vectorial plot in spherical motor.

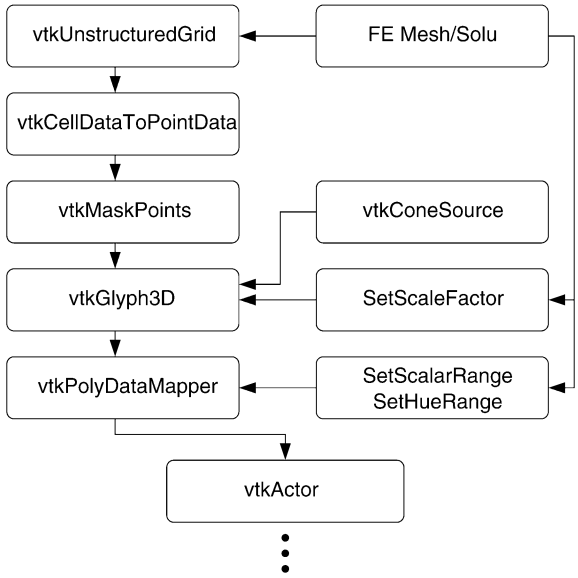

Fig. 7. VTK filter chain for a vectorial plot.

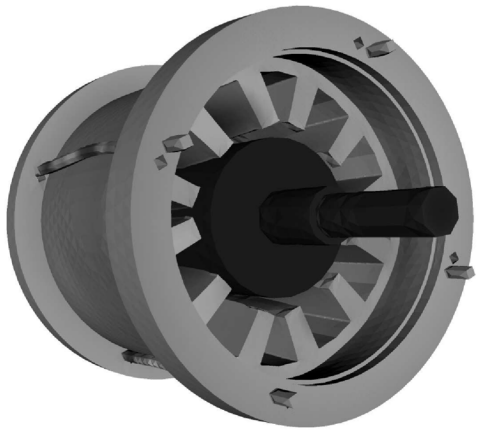

Fig. 8. Deformation of a reluctance machine.

already to the polygonal data, a *vtkPolyDataMapper* is needed. The filter chain is completed as mentioned for the scalar plot before.

Another important visualization method is the plotting of vectors. An example is illustrated in Fig. 6. The necessary filter chain is shown in Fig. 7. Compared to the flux line filter chain, the cell data is connected to *vtkMaskPoints*. This algorithm reduces the number of shown vectors to increase the clearness of the rendering scene. Afterwards, the *vtkGlyph3D* filter is applied. Glyphing is a visualization technique that represents the data by using symbols or glyphs. The symbols can be simple or complex, ranging from oriented cones to show vector data to complex, multivariate glyphs such as Chernoff faces. The *vtkGlyph3D* class allows to create glyphs that can be scaled, colored, and oriented along a direction. The glyphs are copied to each point of the input data set. The glyphs are represented by cones, which are generated by the class *vtkConeSource*. The filter chain is rounded up by *vtkPolyDataMapper* and *vtkActor*.

For structure dynamic simulations, deformations have to be visualized. The filter chain equals the filter chain of the scalar solution (see Fig. 3), except the fact that *vtkWarpVector* is placed between the *vtkUnstructuredGrid* and *vtkDataSetMapper*. *vtkWarpVector* modifies point coordinates by scaling in the direction of the point vector. The scaling is controlled by the method *SetScalingFactor*. An example is show in Fig. 8.

The presented visualization methods combined with virtual reality environments (see Fig. 9) improve the understanding of finite element solutions significantly.

IV. Virtual Reality Hardware

The best virtual reality system available is a CAVE-style system, where the virtual environment is projected from the outside onto the surfaces of a hollow cube so that the user is completely surrounded by the projection panels (Fig. 10). The cave located in the computer center of the RWTH Aachen University, Aachen, Germany, consists of a bottom panel and four side panels creating a nearly perfect immersion effect of 360°. The stereoscopic effect is achieved by using two digital video projectors for each panel and circular polarization. The reference user standing in the cave wears the polarization glasses. In addition, a tracking system detects the position and

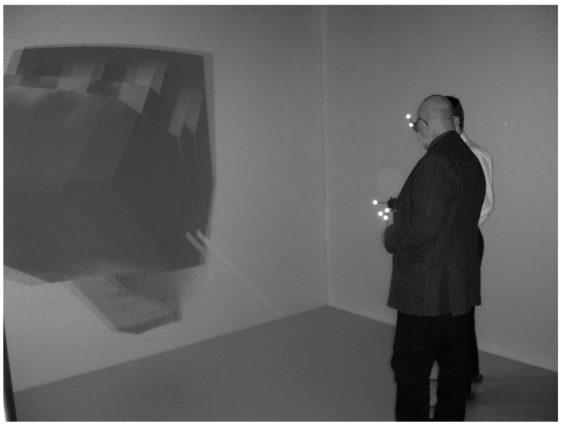

Fig. 9. User in a CAVE-style system.

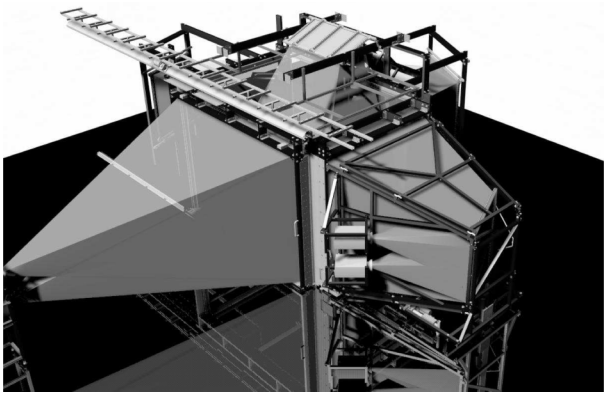

Fig. 10. CAVE-style system.

movements of the reference user and adjusts the visualization accordingly. Other users in the cave wear polarization glasses without tracking, because the stereo effect can only be adjusted for one observer. The visualization quality for the other users depends on their distance to the reference user. This remark holds for all systems discussed in this paper.

A less elaborated virtual reality system is the holobench. It consists of an L-shaped projection table with two orthogonal projection panels. Portable systems also exist based on the same principle. The portable systems are equipped with two digital light processing (DLP) video projectors, generating linearly polarized light with a 90° shift to each other. Thanks to the glasses mounted with lenses polarized accordingly, each eye receives its own stereoscopic picture flow. The projector can either be placed at the front side of an opaque panel or at the back side of a semitransparent one in order to avoid shadowing problems.

Finally, the combination of a standard desktop monitor with shutter glasses (see Fig. 11) is already enough to allow stereoscopic visualization. The monitor is driven to produce alternative pictures for the right and the left eye. The shutter glasses mask alternatively one eye and the other with the same frequency so that each eye gets again its own picture flow. For this environment, the software ViSTA is not needed because VTK already offers all needed features inside the class *vtkRenderWindow*.

Fig. 11. Desktop PC with shutter glasses.

V. Conclusion

We have shown in this paper that computer graphics packages like VTK can provide, with minor adaptations, all required functionalities for an efficient and user-friendly postprocessing of finite element computation results. They fulfill all requirements of modern code development (object oriented, portability, and modularity) but provide also valuable additional functionalities like virtual reality representation of electromagnetic fields in complex structures. There are indeed many situations where such functionalities can be exploited advantageously. Virtual reality allows a better intuitive understanding of the complex systems and assists the developers in branches where simulations and prototyping are used intensively. It provides an impressive way to present to a scientific audience the research objectives and development results. A product presentation to customers is also possible, already at the preprototype state. Finally, because abstract theoretical notions can be made clearer by an adequate visualization, virtual reality environments can also be used fruitfully for teaching purposes. There is a wide range of hardware available for building virtual reality environments, from a normal desktop PC equipped with shutter glasses, up to the CAVE-style systems. They differ in price and size and have to be selected in the function of the field of application.

References

[1] The VTK User's Guide. ver. 4.4., Kitware Inc., 2004.
[2] T. van Reimersdahl, "ViSTA: A multimodal, platform-independent vr-toolkit based on WTK, VTK and MPI," presented at the 4th Int. Immersive Projection Technol. Workshop, Ames, IA, 2000.
[3] D. van Riesen, C. Monzel, C. Kaehler, C. Schlensok, and G. Henneberger, "iMOOSE—An open-source environment for finite-element calculations," *IEEE Trans. Magn.*, vol. 40, no. 2, pp. 1390–1393, 2004.

Manuscript received June 24, 2007 (e-mail: marc.schoening@iem.rwth-aachen.de).