Interactive Visualization of transient 3D Electromagnetic and n-dimensional Parameter Spaces in Virtual Reality

Martin Hafner, Stefan Böhmer, François Henrotte and Kay Hameyer
RWTH Aachen University, Institute of Electrical Machines
Schinkelstraße 4, D-52056 Aachen, Germany,
Martin.Hafner@iem.rwth-aachen.de

1 Abstract

Purpose - Nowadays many parameter studies for the design and optimization of electromagnetic devices are carried out by means of 2D and 3D nonlinear finite element models. Through optimization algorithms selecting one design as optimal with respect to the chosen cost function, the user does not gain any intuitive clue of the interrelations existing between design parameters, although numerous computations have been performed across the whole parameter space of the system.
Design/methodology/approach - This paper presents a visualization approach for n-dimensional parameter spaces of FE solutions in virtual reality and the corresponding interpolation methods for enabling navigation through it.
Findings - The solution of an arbitrary electromagnetic FE problem is categorizes with respect to possible changes, due to chosen design parameters, within the solution itself and variations in the underlying mesh in order to find appropriate interpolation methods for the visualization of each type of parameter space.
Practical implications - The implementation bases on modifying the popular VTK framework.
Originality/value - Different solution approaches for the visualization of an interpolation between arbitrary different meshes are presented as well, but the problems remains unsolved and requires further research.
Keywords - Virtual reality, Visualization, Parameter space methods, VTK, Scientific visualization
Paper type - Research paper

2 Introduction

Along with the development of computers, electromagnetic finite element (FE) simulation software has attained nowadays a high level of sophistication. In many real-life engineering situations, the problem is no longer only to be able to simulate, but also to be able to interpret correctly and efficiently the huge amount of numerical information generated by the simulation.

Complex FE simulations are routinely used for design purposes of electric machines. With standard methods,e.g. Differential Evolution or Simulated Annealing, the optimization process is performed blindly, like a black box, with no prior knowledge of the system. At the end of the process, the engineer has gained no intuitive understanding of the existing interrelations between the different design parameters, and he must trust the optimum found by the algorithm. A number of optimization techniques attempt to build along the way some global knowledge about the objective function, and to perform some sensitivity analysis, but it is seldom pointed out how much this construction is precisely an aspect of technical design where the global cognitive capacities of the human brain could definitely be worth being more explicitly exploited with the help of virtual reality (VR) tools.

Therefore, the aim of this work is to visualize a complex nonlinear FE solution system, which is parameterized through a number of control variables, in virtual reality. This allows users to intuitively experience the sensitivity of response under specific parameter changes in the multi-parameter space. In case of a dynamic response, time can be chosen as the coordinate in a monodimensional parameter space and the user can sweep back and forth in time, with a cursor for instance, in order to analyze in detail the dynamic evolution of a quantity of interest.

All FE simulations reported in this paper have been done with the finite element package iMOOSE (van Riesen, Monzel, Kaehler, Schlensok & Henneberger 2004) whose numerical results were converted into the VTK data structure (Kitware 2006).

3 The visualization pipeline of VTK

The Visualization Toolkit VTK (Schroeder, Martin & Lorensen 2006) is an open source platform independent software library for 3D computer graphics, image processing and visualization. It is applied for scientific purposes in various fields of research (Joshi, Scheinost, Vives, Spencer, Staib & Papademetris 2008), (Sustersic, Kandemir, Phoha & Schmiedekamp 2008), (Badesa, Pinto, Sabater, Azorin, Sofrony & Cardenas 2009). This software package is capable of scalable parallel processing (Ahrens, Law, Schroeder, Martin, Inc & Papka 2000) and is also used for supercomputing visualizations (Ahrens, Lo, Nouanesengsy, Patchett & McPherson 2008). The working principle of VTK is based on visualization pipelines Fig. 1(a). These visualization networks are constructed by connecting data objects, representing and providing access to data, and process or filter objects that operate on those data objects (Schroeder, Avila & Hoffman 2000). Each pipeline object has an internal state control to detect when a reexecute command is necessary. Generally data update is performed from a source to a drain object, as shown in the figure.

4 Methodology

4.1 Requirements

Requirements on parameter space visualization arise from the data and the tasks to be fulfilled. The aim is to easily compare and evaluate the results of multiple FE computations with a changing set of parameter. More precisely, there is one FE dataset for each set of parameter values. This leads to a multidimensional parameter space with no restriction in dimension or in the number of available solutions. For better comparison and an intuitive recognition of the observed phenomena within the solution parameter space, an interpolation of the discrete steps should be performed to gain a continuous parameter space. This interpolation functionality has to consider two different cases occurring in electromagnetic FEA.

- **Solution Interpolation -** By changing material characteristic or excitation conditions, the magnitude and direction of the obtained field solution differs, but remain defined on the same mesh. We only have to differ between continuous but discretized parameters (e.g. current density) and non-continuous values (steel properties).

- **Mesh Interpolation -** A variation of geometric parameters, such as teeth or yoke width typically, leads to FE results, where the underlying mesh changes between two different parameter values. Here, a mesh to mesh interpolation is required to map between field nodes which do not necessarily have the same global coordinates.

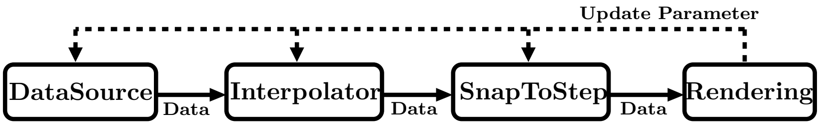
The method proposed here, also works for FE models with motion, where a particular part of the mesh is rotated or translated in function of time. An application of arbitrary hp-adaption techniques within the FEA is unaffected by the visualization procedure, but requires further processing within the interpolation, see 5.3, 5.4.

Finally, the tool has to be integrated seamlessly into the existing interface (Schoning & Hameyer 2008), (Hafner, Schoening, Antczak, Demenko & Hameyer 2009) between iMOOSE and VTK, but should also be able to run independently of iMOOSE in conjunction with any other VTK task. Furthermore maintainability and compatibility regarding future releases of VTK are two important demands.

4.2 Design and Implementation

Given the fact that VTK is used for visualization and that there is an existing, recently published, implementation for time support (Biddiscombe, Geveci, Martin, Moreland & Thompson 2007) (which essentially is a mono-dimensional parameter space), it is considered to extend this implementation concept to hold a multidimensional parameter space. The time support in VTK is realized by attaching time specific meta information to the pipeline architecture which can be passed up **and** down the pipeline, see Fig. 1(b). Hereby filters are enabled to request specific time steps from a previous filter or source in the pipeline and can pass the data down the pipeline. By activating the **SnapToStep** functionality, the interpolation between solutions is deactivated for performance issues

(a) Classical visualization pipeline of VTK.



(b) Pipeline concept with time meta informations.

Figure 1: Different pipeline concepts of VTK.

and the navigation is limited to in memory available cases. This time support represents exactly the desired behavior for the special case with only one parameter. So the idea of our design is to replace the scalar time by a vector whose size equals the dimension of the parameter space, respectively the number of parameters.

Figure 2 shows a two dimensional parameter space, where a time-transient FE computation is carried out for a series of different excitation currents. The crosses symbolize the computed FE solutions corresponding to a unique parameter set (t,J). The path represents the interactive navigation through the solutions in virtual reality. At step ① a new FE solution is loaded and projected in VR. At step ② and ③ a solution interpolation is required, hence the underlying mesh is constant. A navigation through time at steps ④ and ⑤ requires both a mesh and a solution interpolation for a mono-dimensional direction (time axis). The same operation in bi-direction is given in ⑥.

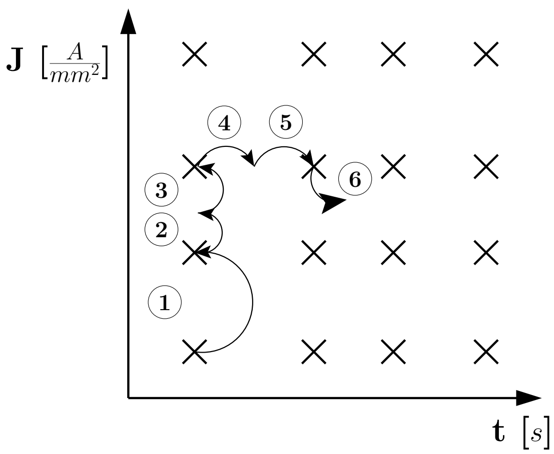


Figure 2: Interactive navigation through a 2D parameter space.

5 Implementation details

5.1 Pipeline architecture

Coming up with version 5.0 VTK got a new pipeline architecture (Aylward, Barre, Cedilnik, Hoffman, Ibanez, King, Martin, Moreland, Schroeder & Squillacote 2006) - the official documentation of each class can found in (*VTK Online Documentation* 2010). The old pipeline concept handled both pipeline execution and algorithm operation in every filter or data source, allowing computing only in forward direction. Performing an update or refresh command on the render in Fig. 1(b) provokes the rendere to reload the data output of the **SnapToStep** filter - no further request etc. are initiated. The new architecture splits the pipeline in two separate classes. **VtkAlgorithm** is responsible for providing and manipulation of data and **vtkExecutive** connects the different filter of the pipeline and handles execution and communication. Following this approach it is possible to transfer additional information up and down the pipeline by using the executives. Furthermore all the data transfer between two objects in the pipeline is handled by the corresponding executives. This enables the executives to change the transmitted data before passing it to the next filter. In the VTK 5.0 concept, the exemplary refresh command of the render is communicated backwards down to the **DataSource** and afterwards thought all contributing filters back to the renderer.

VTK time support (Biddiscombe et al. 2007) uses the new architecture to attach information about given or requested time steps and time range to the communication in the pipeline. It is realized by extending the pipeline operation by the following requests:

- REQUEST_INFORMATION communicates information about time steps and time range down the pipeline.
- REQUEST_UPDATE_EXTENT communicates desired time steps up the pipeline.
- REQUEST_DATA generates data in the filter and populates it down the pipeline.

In this paper, this approach is extended to fulfill the requirements of a multidimensional parameter space. As sketched in 4.2, the idea is to replace the scalar time by a vectorial representation t

$$t \longmapsto \begin{pmatrix} p_1 \\ : \\ p_n \end{pmatrix}, \tag{1}$$

where each entry p_i in this vector represents one dimension of the parameter space and the value of p_i indicates a specific point within the field range of the space dimension. This results in a vector holding n elements in the case of a n-dimensional parameter space.

The VTK pipeline requests of the time support are designed to handle multiple time steps within one request. Based on this characteristic, the enhanced n-D time vector, holding one parameter set, can be transcipted into a sequence of time steps. By this, the basing pipelining functionality of VTK can be used without modification. The time mapping and its reverse has to be performed in each filter which manipulates the timing vector t, e.g. **SnapToStep**, or processes data in function of t, e.g. **Interpolator**, cf. Fig. 1(b).

5.2 Source object

The new object **DataSource** is the first element in the execution pipeline of Fig.1(b). This object is a container for all the solutions of a previously performed FE computation. The solutions are internally stored in a vtkTemporalDataSet which simply represents a sequence of data sets. Therefore, this class requires the time to sequence transcription to identify every single solution by its unique parameter set t.

5.3 Interpolator

Say one has got a FE model Ω depending on n design parameters spanning a n-dimensional parameter space Λ, and assume the model has been evaluated for a cloud of points $\lambda_i \in \Lambda$. For each sampled point λ_i, the solution of the FE problem for a physical quantity ν has been visualized over the domain of analysis as a colormap of the solution:

$$v\left(\underline{x}\right), \underline{x} \in \Omega \tag{2}$$

From a functional point of view, the displayed solution consists of 2 elements: the evaluation point $\underline{x} \in \Omega$ and the value $v\left(\underline{x}\right)$ associated by the model to the point.

For the purpose of a smooth visualization of the solution $v\left(\underline{x}\right)$ when navigating the parameter space Λ, interpolation methods between several solution maps must be applied.

In order to formalize this problem mathematically one has to distinguish amongst the n design parameters between the geometrical parameters $\varsigma_i, i = 1, .., m$ and the non-geometrical parameters $\mu_j = m, .., n$, so that the solution of the FE problem can be written

$$v\left(\underline{x}\left(\varsigma_i\right), \mu_j\right). \tag{3}$$

The interpolation is now obtained by the combination of a number of neighboring pre-computed solutions. For the sake of simplicity, we assume only the closest available solutions, represented by $\left(\varsigma_i^A, \mu_j^A\right)$ and $\left(\varsigma_i^B, \mu_j^B\right)$ respectively, are used. The intermediary states between them, assuming an affine linear combination, are

$$\varsigma_i\left(\zeta\right) = \varsigma_i^A + \zeta\left(\varsigma_i^B - \varsigma_i^A\right), \zeta \in [0,1] \tag{4}$$

$$\mu_j\left(\zeta\right) = \mu_j^A + \zeta\left(\mu_j^B - \mu_j^A\right) \tag{5}$$

and the purpose of the interpolation method is to provide an approximate color map

$$\tilde{v}\left(\zeta\right) \cong v\left(\underline{x}\left(\varsigma_i\left(\zeta\right)\right), \mu_j\left(\zeta\right)\right). \tag{6}$$

This setting with only two solutions and a linear affine interpolation is rather simple but it suffices to explore the main difficulties encountered. The μ-parameters can be treated by blending techniques.

$$\tilde{v}_{blend} = \left(1 - \zeta\right) v\left(\underline{x}, \mu_j^A\right) + \zeta\left(\underline{x}, \mu_j^B\right) \tag{7}$$

This operation is implemented in 5.4.

If the mesh is not fixed during the FE computations (ς-parameters), for example due to rotation, geometric parameter variation or mesh refinement techniques, an additional mesh interpolation has to be performed. This interpolation consists of two tasks:

- **Vertex correspondence**: A mapping between the vertices (nodes) of the two data sets has to be computed.
- **Vertex path**: After the mapping is found the displacement of each vertex from its initial position to the new one has to be done.

Amongst the ς-parameters, one has to further distinguish between those that represent a rigid body motion (angular position of the rotor for instance) from those that describe a deformation of the underlying geometry.

In the first case, the vertex correspondence depends explicitly and exclusively on the design parameter ς, and it can be treated rather easily. If one is aiming at an intuitive access to the visualized field quantities and a minor loss accuracy is acceptable in this case, the vertex path can be interpolated linearly between the two given vertices, since the discretization of rotation is a small fraction of an electrical period. This is also implemented in 5.4. If this simplification is not sufficient, one can apply advanced approaches which try to preserve the mesh structure, e.g. the intrinsic solution of (Sederberg, Gao, Wang & Mu 1993), which modify the angles and edge lengths of the mesh. In case of an rotating FE-part, the proposed approach can be simplified, since the rotation transformation for each time step is explicitly known.

In the second case, morphing techniques must be applied:

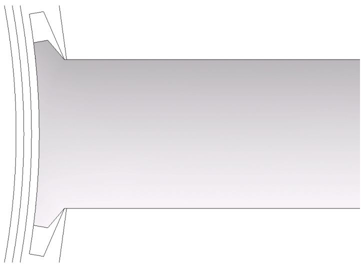
$$\tilde{v}_{morph} = v\left((1-\zeta)\,\underline{x}\left(\varsigma_i^A\right) + \zeta\underline{x}\left(\varsigma_i^B\right), \mu_j\right) \tag{8}$$

A general solution, which does not require any further information about the mesh or of its solution is given by Sederberg et al. (Sederberg & Greenwood 1992), who use a physically based model: one of the meshes is modeled as a construct of wire. Then a vertex mapping is chosen which minimizes the amount of work needed to transform this mesh to the second one by bending and stretching. A more sophisticated solution is presented in (Tian, He, Cai & Feng 2006), where arbitrary meshed geometries are transformed into each other. A satisfying solution for the visualization by smooth interpolation between such types of solution is still an open question.

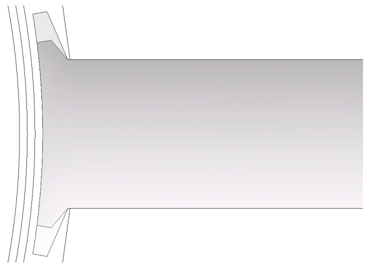
To demonstrate this situation, Fig. 3 shows the image blending between two almost identical FE simulations, where the tooth head varies. Even if this method only consists of a cost effective pixel-wise addition of the rendered solution, tendencies are slightly distinguishable. Since FE electromagnetic governing equations are usually first-order vector potential formulations, leading to zero-order magnetic flux density solutions, an application of time-consuming field projection would lead to a visualization of comparable quality.

5.4 Enhanced VTK Interpolator

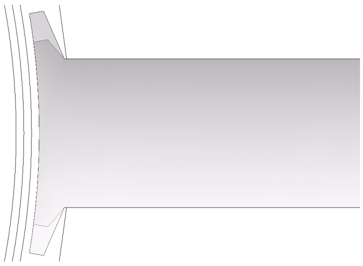
Interpolation is necessary to navigate smoothly through the parameter space because containing data is given by discrete points representing different parameter sets. Therefor a new filter called **Interpolator** is build based on the VTK



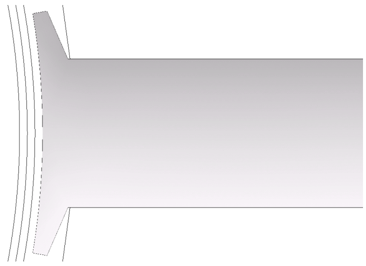
(a) Present FE solution ONE



(b) Image blending 33%



(c) Image blending 66%



(d) Present FE solution TWO

Figure 3: Image blending between two FE solutions with a tooth head variation.

time support interpolator. **Interpolator** is placed in the execution pipeline and changes parameter information in the REQUEST_INFORMATION phase by deleting the parameter points and only passing the parameter range down the pipeline. Thereby, the next filter object in the pipeline is able to request any point in the parameter space. If this point is not equal to one of the given discrete points the interpolator generates it on demand.

The multidimensional parameter space interpolation is carried out by the inverse distance weighting (IDW) (Shepard 1968), as sketched in case of 2D in Fig. 4 to compute the function value at x, y by the surrounding points 0-3. The value of a desired point is computed by a linearly weighted (affine) combination of the surrounding discrete parameter points. Weighting is done by applying a function of the inverse distance between the two points. Formally the method is given by

$$v(x) = \frac{\sum_{i=0}^{N} \frac{v_i}{d_i}}{\sum_{i=0}^{N} \frac{1}{d_i}}, \tag{9}$$

where x denotes the desired point, $v(x)$ the value of the desired point, v_i the value of the given point x_i and d_i the distance between x and x_i in the n-dimensional space. N denotes the number of neighbor points which equals 2^n in a regular n-dimensional grid. A intermediate position between two points in one dimension is simply identified as weighted combination of the two points. The advantage of this simple identification is, that the navigation in higher dimensional spaces can be based on exactly the same principle. Moreover, this allows a combination with arbitrary user interfaces or navigation models, such as the high sophisticated time model of (Wolter 2010).

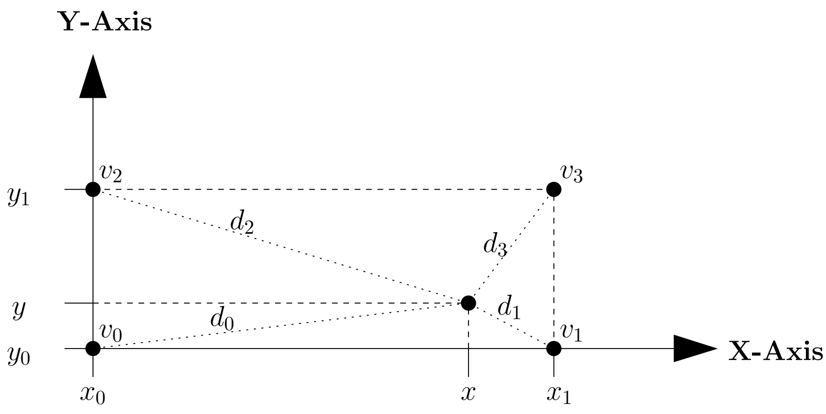


Figure 4: A exemplary 2D-Interpolation by IDW.

6 Conclusion and further prospects

In this paper, the popular open-source framework VTK is applied and expanded to visualize multi-parameter spaces of finite element solutions. The corresponding VTK filter chains are presented - necessary adaption within the VTK structure are proposed. To navigate smoothly through the parameter space, which assists an intuitive interpretation of the ongoing changes within the FE solution, a VTK based mesh and solution interpolation is applied. In present state, this interpolation fails in case of a variation of the CAD model and consequently of the underlying mesh, since this requires a mapping function between geometrically different discretization. Even if several approaches from literature are discussed, the development of an generally applicable solution for that problem is current state of research.

To improve the immersion of such techniques, multi-parameter visualizations should be projected in CAVE (Cave Automatic Virtual Environment) or CAVE-like environments. In a further step, this would require an integration of a virtual reality library, e.g. VISTA (van Reimersdahl, Kuhlen, Gerndt, Henrichs & Bischof 2000). Further, the implementation of FEA restarting facilities would enable to perform simulations as a user's reaction within the VR scene. The implementation of an adequate input device for the navigation through the multi-parameter spaces would also be of assistance.

7 ACKNOWLEDGEMENT

This research was funded by the SEED Fund project "‘Interactive Data Visualization in 3D Electromagnetics"’ of the Jülich Aachen Research Alliance (JARA) (*JARA - Jülich Aachen Research Alliance - Jara* 2010). We are grateful to all members of the Virtual Reality Group of the RWTH Aachen University headed by Prof. Kuhlen for their support.

References

Ahrens, J., Law, C., Schroeder, W., Martin, K., Inc, K. & Papka, M. (2000). A parallel approach for efficiently visualizing extremely large, Time-Varying datasets.

Ahrens, J., Lo, L., Nouanesengsy, B., Patchett, J. & McPherson, A. (2008). Petascale visualization: Approaches and initial results, *Ultrascale Visualization, 2008. UltraVis 2008. Workshop on*, pp. 24–28.

Aylward, S., Barre, S., Cedilnik, A., Hoffman, B., Ibanez, L., King, B., Martin, K., Moreland, K., Schroeder, W. & Squillacote, A. (2006). Kitware source - software developer's quarterly (Issue 1 - july 2006).

Badesa, F., Pinto, M., Sabater, J., Azorin, J., Sofrony, J. & Cardenas, P. (2009). Efficient collision algorithm for the 3D haptic interaction with solid organs in medical environments, *Advances in Computer-Human Interactions, 2009. ACHI '09. Second International Conferences on*, pp. 187–192.

Biddiscombe, J., Geveci, B., Martin, K., Moreland, K. & Thompson, D. (2007). Time dependent processing in a parallel pipeline architecture, *Visualization and Computer Graphics, IEEE Transactions on* **13**(6): 1376–1383.

Hafner, M., Schoening, M., Antczak, M., Demenko, A. & Hameyer, K. (2009). Interactive postprocessing in 3D electromagnetics, *17th Conference on the Computation of Electromagnetic Fields*, Florianopolis, Brazil.

JARA - Jülich Aachen Research Alliance - Jara (2010). http://www.jara.org/.

Joshi, A., Scheinost, D., Vives, K., Spencer, D., Staib, L. & Papademetris, X. (2008). Novel interaction techniques for neurosurgical planning and stereotactic navigation, *Visualization and Computer Graphics, IEEE Transactions on* **14**(6): 1587–1594.

Kitware, I. (2006). *VTK User's Guide Version 5*, 5 edn, Kitware, Inc.

Schoning, M. & Hameyer, K. (2008). Applying virtual reality techniques to finite element solutions, *Magnetics, IEEE Transactions on* **44**(6): 1422–1425.

Schroeder, W., Avila, L. & Hoffman, W. (2000). Visualizing with VTK: a tutorial, *Computer Graphics and Applications, IEEE* **20**(5): 20–27.

Schroeder, W., Martin, K. & Lorensen, B. (2006). *Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th Edition*, 4th edn, Kitware.

Sederberg, T. W., Gao, P., Wang, G. & Mu, H. (1993). 2-D shape blending: an intrinsic solution to the vertex path problem, *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, ACM, Anaheim, CA, pp. 15–18.

Sederberg, T. W. & Greenwood, E. (1992). A physically based approach to 2D shape blending, *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM, pp. 25–34.

Shepard, D. (1968). A two-dimensional interpolation function for irregularly-spaced data, *Proceedings of the 1968 23rd ACM national conference*, ACM, pp. 517–524.

Sustersic, J., Kandemir, M., Phoha, S. & Schmiedekamp, M. (2008). High-performance visualizations and simulations for ocean environments and the mine countermeasure mission using C3L, *OCEANS 2008*, pp. 1–8.

Tian, H., He, Y., Cai, H. & Feng, L. (2006). Efficient metamorphosis of Point-Sampled geometry, *International Conference on Artificial Reality and Telexistence*, Vol. 0, IEEE Computer Society, Los Alamitos, CA, USA, pp. 260–263.

van Reimersdahl, T., Kuhlen, T., Gerndt, A., Henrichs, J. & Bischof, C. (2000). ViSTA: a multimodal, platform-independent VR-Toolkit based on WTK, VTK, and MPI, *Fourth International Immersive Projection Technology Workshop (IPT2000)*, Ames, Iowa.

van Riesen, D., Monzel, C., Kaehler, C., Schlensok, C. & Henneberger, G. (2004). iMOOSE-an open-source environment for finite-element calculations, *Magnetics, IEEE Transactions on* **40**(2): 1390–1393.

VTK Online Documentation (2010). http://www.vtk.org/doc/nightly/html/classes.html.

Wolter, M. (2010). *Navigation in Time-Varying Scientific Data*, PhD thesis, RWTH Aachen University.