9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization
4-6 September 2002, Atlanta, Georgia

AIAA 2002–5513

VisualDOC: A Software System for General-Purpose Integration and Design Optimization

Vladimir O. Balabanov (vladimir@vrand.com) *, Christophe Charpentier †, Dipankar Ghosh ‡, Gary Quinn §, Garret N. Vanderplaats ¶, and Gerhard Venter ‖

Vanderplaats Research and Development, Inc.
1767 S 8th Street, Colorado Springs, CO 80906

The main purpose of this paper is to draw attention to existing commercial general-purpose optimization tools. The representative capabilities of such tools are discussed using VisualDOC by Vanderplaats Research and Development, Inc. as an example. The ease of use of VisualDOC allows a person without an optimization background to start applying optimization to his particular problem within a couple of hours after first encountering VisualDOC. This is emphasized by discussing main VisualDOC features. Particular attention is paid to several ways VisualDOC can be interfaced and/or integrated with almost any analysis program. Practical examples of applying VisualDOC to actual industrial problems are presented to emphasize the benefits of applying optimization in any field.

Introduction

DESPITE many years of research, resulting in the availability of several quality commercial general-purpose optimization programs[1–3] and numerous research and development codes, optimization still has only a limited success in the industrial environment. There are many reasons for this lack of acceptance, including (a) lack of user familiarity with optimization concepts and thus the fear that optimization is a specialized technology which requires immense expertise to use, (b) cost of adding optimization capabilities to existing commercial software, (c) large computational resources required to perform general-purpose optimization.

The first issue is due to the fact that optimization is rarely taught at the undergraduate level, creating the need for some user training that companies are often unwilling to invest in. This issue can be addressed by creating user-friendly software that does not require the knowledge of optimization theory to run. This software should include real-time post-processing capabilities that graphically demonstrate the optimization process.

The second issue arises mostly because a clear market for optimization software has not been established yet, which in turn is a result of the fact that benefits of applying optimization techniques are not fully recognized in industry. This issue can be addressed by software that provides the ability to integrate several independent analysis packages without disrupting a normal flow of information into, out of, and inside of each package. Such software should also provide an easy linkage to the optimization process.

The high computational resources requirement of general-purpose optimization are partly addressed by high-speed computers already available and also by parallel computing. Thus, good optimization

*Senior R&D Engineer, Senior AIAA Member
†R&D Engineer
‡Product manager, AIAA Member
§Senior R&D Engineer
¶President, Fellow AIAA
‖Senior R&D Engineer, AIAA Member

software should provide an option to take advantage of parallel computing.

The VisualDOC[1] general-purpose design optimization software system from Vanderplaats Research and Development, Inc. (VR&D) was created with the purpose of addressing all the issues mentioned above.

Background

One of the main differences between VisualDOC and other available general-purpose optimization and multidisciplinary optimization packages is that VisualDOC was developed based upon the well-known general-purpose optimization package, DOT[4] which was introduced as a high-quality commercial product in 1987. Thus the optimization procedures in VisualDOC are well tested and robust. The VisualDOC software system includes a graphical user interface (GUI), a database, and several functional modules. It was first released as a commercial product in 1998. Since that time, VisualDOC has constantly been updated, refined, and improved. Some companies that provide general-purpose optimization systems are taking the approach of repackaging existing (sometimes even public domain) optimization programs written by somebody else. VR&D uses another approach. Almost all the methods used in VisualDOC are developed inside the company, providing much better control over the algorithms. The set of methods and algorithms used is constantly being updated and modified to better meet customer needs. New and prospective algorithms are being included on a regular basis. Particular attention is given to the customer requests. Many features and algorithms were added to VisualDOC in a short time according to our customers requests.

VisualDOC Features

VisualDOC can be used to add optimization to almost any analysis program. Using VisualDOC one can define optimization parameters based on the analysis program input/output files, run the optimization, and postprocess the results – all of it using specially designed user-friendly graphical utilities. Switching between different kind of tasks (gradient-based optimization, response surface optimization, design of experiments, evolutionary algorithms, etc.) does not require anything more than choosing the type of task. The default parameter settings for each type of task work well for the vast majority of cases. At the same time, experienced users may tune the algorithm parameters to accommodate their particular preferences. All these options along with a colorful short *Getting Started Manual* allow users who are not familiar with optimization to start running their own optimization problem within an hour, without writing a single line of a program code.

Additionally, programming experts may find the VisualDOC application programming interface (API) appealing. VisualDOC API allows them to use VisualDOC capabilities *inside* their own program. Such an option is not available in the majority of other general-purpose optimization software systems.

Figure 1 presents the overall structure of the VisualDOC software system and a brief description of the main VisualDOC features is provided below.

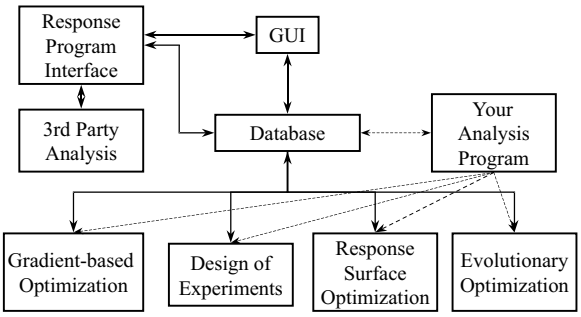

Fig. 1 General VisualDOC structure

Database

The central part of the VisualDOC software system is the database. The database employed is object relational and multiuser. It is also platform independent, which means that the user may define his optimization problem on one operating system, copy the database to a computer with another operating system, run it there, and postprocess the results on a third computer, if desired. The database acts as an efficient container for all design information. Thus changing optimization task types and using results from one optimization task in another task becomes easy.

Graphical User Interface

A graphical user interface is typically the feature that users deal with most. It is thus important to provide both ease of use for the novice and control and flexibility for advanced users. VisualDOC provides a powerful graphical user interface to define and post process design information. The VisualDOC GUI is developed in JAVA and thus the GUI look and feel is very similar on all supported platforms. The GUI allows users to launch all design tasks, perform real time optimization progress monitoring, and post process the results. Users interact

with the program using a spreadsheet metaphor. Figure 2 shows a view of the VisualDOC GUI.

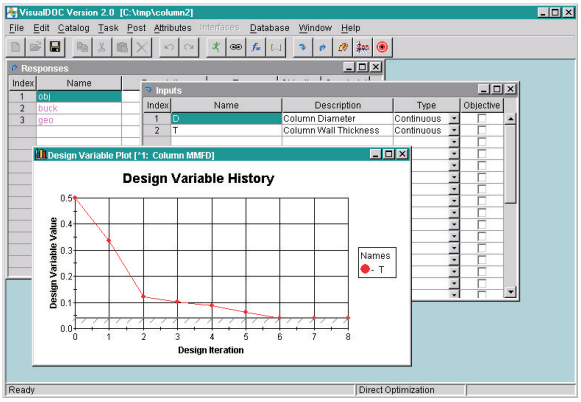

Fig. 2 VisualDOC GUI

Functional Modules

If the GUI can be considered a facade of the software system, the functional modules are its backbone. The functional modules are the part of the software that do the actual work: perform optimization, design study, etc. It should be noted that in VisualDOC, inputs (design variables) may be continuous, integer, discrete, or any combination of these. Below is a brief description of the capabilities of the main VisualDOC functional modules.

Gradient Based Optimization

VisualDOC was developed based upon the well-known gradient based general-purpose optimization package, DOT. The VisualDOC gradient based algorithms include:

- Sequential Quadratic Programming (SQP)
- Modified Method of Feasible Directions (MMFD)
- Sequential Linear Programming (SLP)
- Fletcher-Reeves
- Broydon-Fletcher-Goldfarb-Shanno (BFGS)
- Sequential Unconstrained Minimization Technique (SUMT) for very large constrained and unconstrained problems

All of these methods require gradients of responses (objective functions and constraints) to operate. By default VisualDOC calculates the gradients using finite differences. However, advanced users have an option of providing their own gradients. The above mentioned algorithms have been extensively developed and tested. They have been demonstrated to be both efficient and reliable for a wide range of engineering applications. It is possible to perform multi-objective optimization in VisualDOC, wherein the user may minimize or maximize one or more objective functions while at the same time driving one or more objective functions to some target values.

Response Surface Optimization

In general, both Design of Experiments (DOE) and Response Surface methodologies are used to establish empirical relationships (approximations) between design variables and responses. This is particularly useful when no such relationship explicitly exists (for example, physical experiments), or when such a relationship is very complicated (for example, non-linear finite element codes). Additionally, these methodologies have the advantage of filtering out numerical noise present in the analysis, because the created response approximations are smooth functions (typically low order polynomials) of design variables.

When an optimization problem has relatively few design variables (up to about 20) and when the computational cost of performing a single analysis is high, the response surface techniques in VisualDOC may be the most efficient method to use for optimization. In this method VisualDOC fits an approximate curve to each independent response. Then the optimization is performed using created approximations rather than the exact response values. Thus the cost of calculating gradients is significantly reduced, because gradients of the approximations are calculated analytically inside of VisualDOC. When the approximate optimization is complete, VisualDOC will call the analysis program, evaluate the actual responses, add this new point to a pool of points that are already available and re-fit the approximation curves to all the responses. This process is repeated until the specified convergence criteria is met. To start the response surface optimization process an initial set of points is required. These points could be provided by the user with or without responses. Otherwise VisualDOC will propose a default ordered set of points. The user can select one of several starting strategies.

Design of Experiments

Design of Experiments is a specific systematic arrangement of points in the design space. DOE provides data to explore the design space and to construct response surface approximations. Statistical criteria for distributing the points in the design space typically involve either gaining the most information from a specified number of points or obtaining

a reasonably accurate fit of the approximation using as few points as possible. Often DOE is employed to identify the design variables that have the most influence on the responses in a specified portion of the design space and construct response surface approximations for responses using the most significant design variables.

There is a significant overlap in definitions of Response Surface Methodology and Design of Experiments. In VisualDOC the DOE module is used to do the following:

- Distribute points in the design space according to a specified pattern.
- Perform analysis (evaluate responses) at the specified points.
- Fit various approximations to the obtained response values.
- Perform statistical analysis of the resulting approximations.

All these steps could be very helpful for a designer who wants to find a better region of the design space in terms of the response values.

VisualDOC offers an extensive set of standard statistical design of experiments: full and fractional factorial designs, composite designs (including small composite designs based on fractional factorial designs), simplex design, Koshal designs, Box-Behnken designs, random design, Latin Hypercube designs, Taguchi orthogonal arrays, and several others. The user can also provide his own design.

A unique feature of the VisualDOC DOE module is the ability to account for an irregularly shaped design space when creating the design points. As long as the shape of the design space is governed by the design variables values only, VisualDOC will be able to distribute the points in this design space without placing any points outside of the design space.

Additionally, VisualDOC offers a D-optimal design. This design is useful for spreading out points in an irregularly shaped design space. An additional advantage of the D-optimal design is that the user controls the number of points in the design.

The post processing capabilities of VisualDOC allow the user to compare various designs, analyze the spread of the design points, and perform statistical analysis of the approximations that were fitted to the responses at the specified points.

Evolutionary Optimization

Evolutionary optimization algorithms in VisualDOC include Genetic Algorithm (GA)[5] and Particle Swarm Optimization (PSO)[6] methods. Both of these methods have a better chance of finding a global optimum than gradient-based optimization methods. However, the user should remember that these methods are computationally expensive. It is often cheaper to start a gradient-based optimization or a response surface optimization from several initial points in the design space to ensure that a global optimum is found, than to employ a single run of evolutionary optimization algorithms. Thus, in most cases it is unreasonable to apply evolutionary algorithms to optimization problems of more than about 30 design variables.

Despite this deficiency, evolutionary algorithms have some definite advantages over gradient-based optimization techniques: they do not require gradient information and they are extremely good in handling numerical noise.

The VisualDOC implementation of GA and PSO algorithms follows the general VisualDOC policy: to provide users with a default set of robust control parameters that works for the vast majority of problems. The novice user does not have to set any parameters to solve his problem. At the same time the advanced users are given an opportunity to tune the control parameters according to their needs.

VisualDOC Interfaces with Analysis Programs

To perform the majority of its functions VisualDOC needs to interact with engineering analysis programs. Specifically, VisualDOC needs to provide the values of the design variables to the analysis program and get back the corresponding values of the responses. The goal of the VisualDOC interfaces is to make the process of interfacing any analysis program with optimization as easy as possible. In most cases, no programming is required at all. At the same time advanced users have the option of writing their own programs, provided that a few simple guidelines are followed when interacting with VisualDOC.

Simple Text File Interface

When employing the Simple Text File Interface, VisualDOC communicates with an analysis program by means of ASCII text files: VisualDOC writes out an ASCII text file with the values of design variables and the analysis program is expected to read this file, evaluate the corresponding responses, and write the responses into another ASCII text file that will be read by VisualDOC. Both text files are organized in such a way that the values of the design variables or responses are written one number per line.

This is the most simple and flexible interface. It gives users an opportunity to create a program that in turn calls several other programs (one of the ap-

proaches to multidisciplinary optimization), process data from many sources, etc. But it also requires the user to provide an executable that will read and write the corresponding ASCII files. The user's executable can be written in any programming language or script.

Enhanced Text File Interface (VisualScript)

The enhanced text file interface in VisualDOC is called VisualScript. VisualScript provides an easy-to-use GUI for interfacing external analysis programs with VisualDOC. VisualScript is a flexible tool for creating general interfaces without requiring the user to do any programming.

VisualScript works with any analysis program that reads its input data from one or more ASCII input files and writes out response data into one or more ASCII output files. The user graphically interacts with the VisualScript GUI to define an interface between the analysis program (or programs) and VisualDOC. VisualScript automatically converts the user's graphical definitions into a Perl script. The novice user does not have to worry about the Perl script at all, but advanced users can modify the resulting Perl script to accommodate their specialized needs.

VisualScript allows the user to:

- set up working directories
- search for keywords in input/output files and NOT base this search on line and column numbers if it is not specifically required
- modify input files
- extract results from output files
- use multiple analysis and input/output files
- transfer data between various input and output files
- specify a transfer order for data in and out of VisualScript
- test the resulting interface with or without running the actual analysis program(s)

VisualScript uses two levels of detail when interacting with the user:

1. The *outline* level represents the overall flow of the script and shows the connectivity between different analysis programs. At this level the user has an option of defining *IF* statements that will redirect the program flow based on the results from one or more analysis programs.
2. The *detailed* script element level deals with the low level details associated with each analysis program. Here the user may specify what input parameters should be changed, in what order and how, and what output parameters should be monitored and how.

VisualScript can be used separately from VisualDOC for linking together several independent programs. Figures 3 and 4 show examples of the outline and detailed script level interfaces.

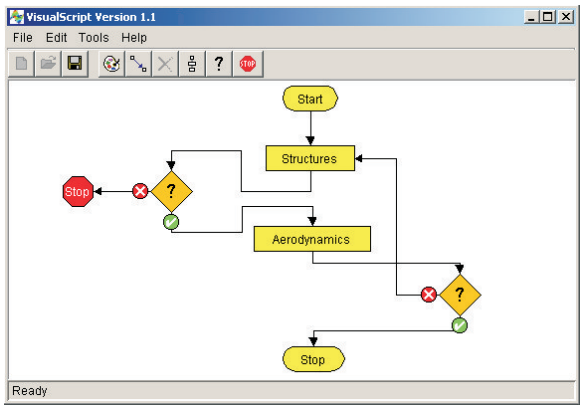

Fig. 3 Example of VisualScript Outline Level GUI

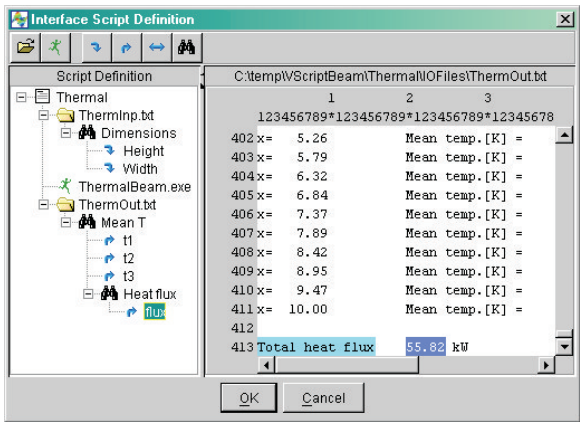

Fig. 4 Example of VisualScript Detailed Level GUI

Interface to MATLAB

A large community of engineers and researchers are conducting their work using MATLAB. MATLAB has its own computational and graphical environment. In order to make VisualDOC useful for this community, VR&D has created a specialized interface between MATLAB and VisualDOC. In this interface the MATLAB computational engine is used

to evaluate responses. VisualDOC starts the MATLAB engine and directly transfers design variables and responses as MATLAB matrix objects. The user should supply his analysis as a MATLAB function M file. Thus users have full access to all the built-in functionality of MATLAB, while gaining access to all VisualDOC capabilities at the same time.

Interface to Microsoft Excel

A lot of PC users take advantage of the powerful Excel capabilities. For users who do all or part of their calculations in Excel VR&D developed an interface between Excel and VisualDOC. Similar to the MATLAB interface, VisualDOC starts the Excel computational engine and transfers design variables and responses directly to and from Excel. The user has to specify what cells contain the design variable information and what cells contain responses. Such an interface provides the user with all VisualDOC capabilities while retaining the ability to exploit all the power of Excel.

VisualDOC API

All VisualDOC interfaces to analysis programs described above could be referred to as interfaces where the VisualDOC engine and the VisualDOC GUI guide the optimization or the study process. Because the VisualDOC GUI is very easy to use, such an arrangement is convenient for most users. However, some advanced users may want to take complete programming control over the whole process. They write their own programs and want VisualDOC to be part of them. Such programming experts may find the VisualDOC API appealing. The VisualDOC API allows them to put all VisualDOC capabilities inside of their own code. VR&D clients use a variety of programming languages to put VisualDOC capabilities into their own programs and products: C/C++, FORTRAN, Visual Basic, MATLAB, etc.

VisualDOC Parallel Capabilities

Practical problems that are solved in industry often require high computational resources. Because optimization needs results not from one, but many analyses, the requirements on computational resources for general-purpose optimization are even higher. These requirements are partly addressed by high-speed computers and also by parallel computing. VisualDOC provides an option to take advantage of parallel computing. If the user has Message Passing Interface (MPI) set up on his computer network, with a simple switch VisualDOC will run optimization jobs at the designated computers of the computer network. Any type of VisualDOC task can take advantage of parallel computing: gradient-based optimization, response surface optimization, DOE, evolutionary algorithms.

Application Examples

VisualDOC is a general-purpose design optimization software system that can be used to couple optimization with a wide range of applications: from financial analysis to auto engine and rocket design. A large number of users both from industry and research institutions are using VisualDOC to improve performance of their products. Many of these applications are proprietary in nature and can not be discussed here. However, a few examples of VisualDOC applications in different spheres of interest are presented below.

VisualDOC was not specifically designed for multidisciplinary optimization. However, because of the general nature of the VisualDOC approach and because of VisualDOC expendability, VisualDOC was successfully used for solving multidisciplinary design optimization problems.[7]

Hybrid Car Design

VR&D and the National Renewable Energy Laboratory (NREL) collaborated on optimizing the control system for a hybrid car. The schematic picture of a possible hybrid car configuration is presented in Figure 5.

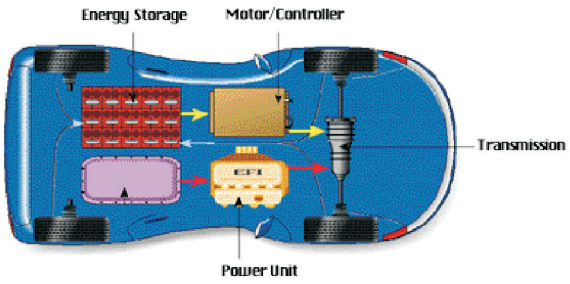

Fig. 5 Scheme of a Hybrid Car Parallel Control Strategy

The overall vehicle was modeled using the ADVISOR program[8] developed by NREL. The ADVISOR program was coupled with VisualDOC to perform optimization of the control system.

The objective in designing the control strategy of the hybrid car was to minimize the nitrous oxide and hydrocarbon emissions while maximizing the fuel economy at the same time, with constraints on

acceleration and grade climbing. There were 8 design variables in the problem. They were various characteristics of battery state of charge and also some parameters of the fuel converter. As a result of the joint work, the fuel economy was increased by 6.5%, while at the same time the nitrous oxide emission was reduced by 11.5% and the hydrocarbons emission was reduced by 3.6%.

A group of researchers was trying to achieve similar goals, but without optimization their results were much worse. Note, that the results were obtained for a fixed mass vehicle. If optimization was also used to reduce the vehicle mass, further improvements in economy could have been made. More details on this work and related efforts are provided in papers by Garcelon[9] and Garcelon et al.[10]

Design of Equivalent Material Properties

Current state of the art of CAD systems and finite element modeling is such that sometimes designers do not pay attention to how their CAD model is meshed into finite elements and how many of these elements exist in the model. Typically finite element meshing is done very reliably. However, in spite of the available computer power, modal analysis for example, may not be easily performed for finite element models with several million degrees of freedom. Still it may not be wise to abandon the whole model, even though certain types of analyses can not be performed on the resulting finite element model. To compromise in such cases it is possible to introduce equivalent elements and replace finely meshed parts of the finite element model with much simpler parts that will have little effect on the model analysis results. However, making a correct substitution of equivalent parts is not a trivial task.

VR&D was tasked to reduce the size of a heat exchanger finite element model. This model exceeded three million degrees of freedom and the customer was not able to perform a modal analysis of the model. The finite element model was obtained from the CAD model using automatic meshing. Many of the degrees of freedom were embedded in seven layers of copper air fins, that were modeled using shell elements. Such level of detail was possibly needed for a heat transfer analysis, but prevented performing modal analysis due to a large number of closely spaced eigenvalues during eigenvalue solution.

VR&D replaced the air fin shell elements with equivalent anisotropic solid elements with far fewer degrees of freedom. The equivalent solid elements did not have identical stiffness coupling as the shell air fins, but the whole purpose of the task was to reduce the finite element model size to provide a good approximation of the model shape and frequencies for a limited number of target modes. Data recovery on the air fins was not possible with the equivalent solid elements, but their purpose was to mimic the mass and stiffness properties of the fins with far fewer degrees of freedom.

To find good properties of the equivalent solid elements VR&D used VisualDOC as an optimization tool and finite element analysis and optimization code GENESIS[11] as an analysis tool.

To specify appropriate data for anisotropic solid elements a typical finite element code requires the input of the 6x6 material property matrix $[\boldsymbol{G}]$ according to Equation 1 below.

$$\{\boldsymbol{\sigma}\} = [\boldsymbol{G}]\{\boldsymbol{\epsilon}\} \tag{1}$$

Only diagonal elements of $[\boldsymbol{G}]$ were considered as design variables in this problem. VisualDOC was used to find the values of these diagonal elements that would: bring the first five frequencies close to their target values while simultaneously constraining them to no more than three percent error, and at the same time constraining displacements produced by the pinch and twist static load cases to no more than five percent error. Three and five percent error values were considered a reasonable engineering tolerance.

All static and modal comparisons were done on a small segment of the heat exchanger (Figures 6 and 7). These comparisons were done for three configurations: (a) air fins modeled with shell elements (original model), (b) air fins completely removed, and (c) equivalent solid elements substituted for the original shell elements. The results for one of several load cases considered simultaneously are shown in Table 1.

Summarizing the results obtained, the model with optimized equivalent material properties show no more than 5% error in the selected displacements, while the model frequencies for the first five elastic modes were within 3% error. This very favorably compares to 50% error and 20% error respectively for the model with no air fins at all. It was also confirmed that equivalent solid elements did not change the overall mode shapes when compared to the original model (Figures 6 and 7). Using such a substitution the size of the full model was reduced by approximately one million degrees of freedom.

It should be noted that calculating the equivalent properties of the solid elements using VisualDOC did not require any specific knowledge of structures or composite materials except for Equation 1.

Table 1 Results for modal analysis

Mode	With air fins Freq (Hz)	No air fins Freq (Hz)	Percent Error	Equiv solid Freq (Hz)	Percent Error
1	3169	2544	19.7	3106	2.0
2	3460	3318	4.1	3499	−1.1
3	4291	4049	5.6	4256	0.8
4	6086	5908	2.9	5943	2.3
5	8091	7154	11.6	8040	0.6

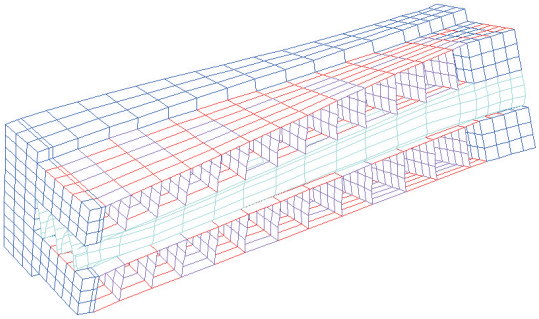

Fig. 6 The first twist mode of part of the original heat exchanger

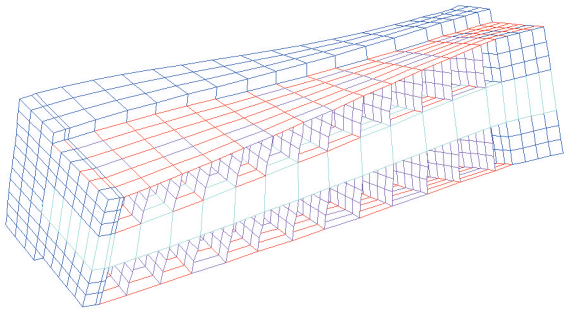

Fig. 7 The first twist mode of part of the heat exchanger with equivalent solid elements.

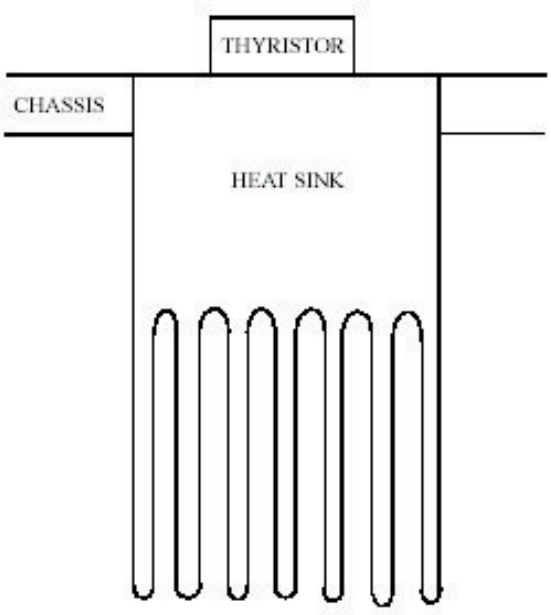

Fig. 8 Original configuration of a heat sink.

There is also a constraint on the maximum temperature allowed in the thyristor.

The initial design was chosen to have an unreasonably thick base to test the optimization (Figure 8). The optimum design is shown in Figure 9 and is very

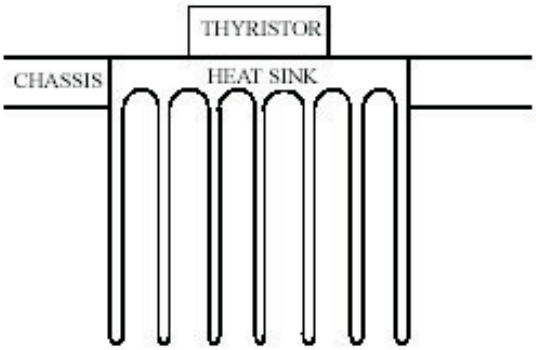

Fig. 9 Optimal configuration of a heat sink.

Heat Sink Design

Another example of applying VisualDOC capabilities is the design of a heat sink for electronic applications. Here VisualDOC was coupled with the FLUX2D thermal analysis program from CEDRAT Corporation.[12] The initial design is shown in Figure 8.

Heat is generated by the thyristor and dissipated by the heat sink. The objective is to minimize the material of the heat sink. The design variables are the thickness of the base, height and width of the fins. Constraints are imposed on the heat dissipated to the air and the heat transferred between the heat sink and the supporting chassis.

similar to heat sinks commonly found in electronic devices. This demonstrates the ease with which a commercial analysis program can be coupled with optimization.

Design Optimization of a Mixing Elbow

Computational Fluid Dynamics (CFD) problems are known for their nonlinearity. Even relatively simple problems require iterative solution that could provide additional source of noise to the optimizer. As an example of VisualDOC application to CFD problems we considered designing a mixing elbow for better temperature distribution at the outlet (Figure 10). This example is familiar to both FLUENT[13] and GAMBIT[14] users. The core of this example is taken from *FLUENT Tutorial 1* and *GAMBIT Tutorial 2*.

Cold water at 26 °C enters through the large pipe and mixes with the warmer water at 40 °C in the elbow. The Reynolds number at the main entrance is 2.03 x 10^5, so that a turbulent model was required for solution.

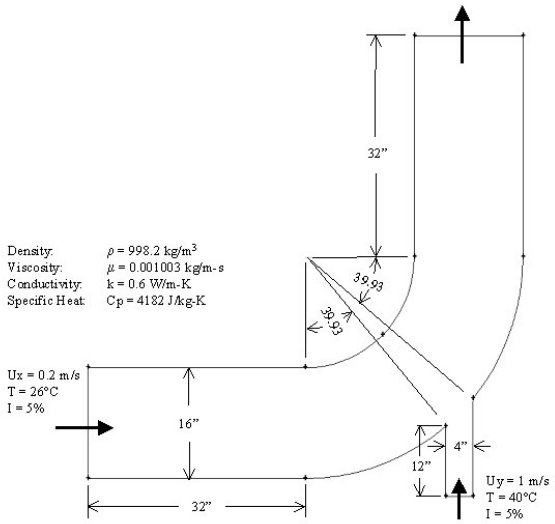

Fig. 10 Schematic representation of a mixing elbow.

In the original example from *FLUENT Tutorial 1* and *GAMBIT Tutorial 2* a small vertical mixing tube was aligned with the center of the pipe. However, the large temperature gradient at the pipe outlet produced by this configuration may not represent optimal mixing (Figure 11).

We selected three parameters as design variables to improve the temperature distribution at the pipe outlet: radius of the pipe elbow, location of the mixing pipe, and angle of the mixing pipe. The objective

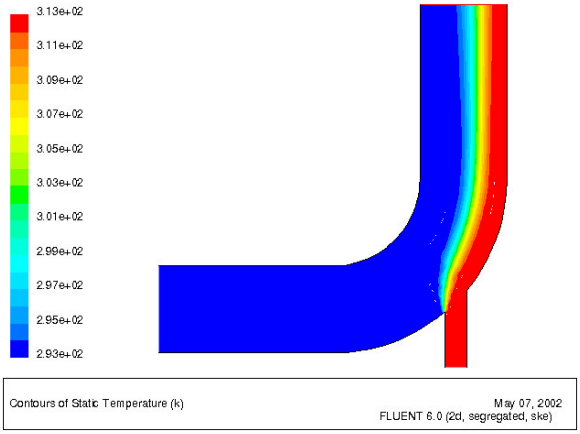

Fig. 11 Initial temperature distribution in a mixing elbow.

was to reduce the standard deviation from the mean temperature at the pipe outlet.

The temperature mixing for the optimal configuration (Figure 12) was much better than that for the original configuration (Figure 11).

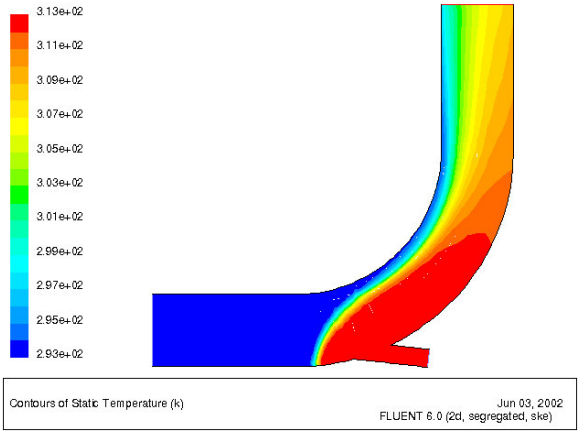

Fig. 12 Final temperature distribution in a mixing elbow.

Parameterization of the GAMBIT and FLUENT journal files was critical for the optimization implementation. After that parameterization, VisualDOC was able to control both meshing in GAMBIT and the CFD solution in FLUENT automatically. Using Response Surface optimization technique, only 26 FLUENT calls were required to get to the optimal solution from the initial configuration.

Conclusions

VisualDOC is a software package that expands the optimization technology base and at the same

time greatly improves ease of use of optimization. Particularly, conducting design studies, storing and interpreting results, and integrating and linking various analysis codes together is greatly simplified. The approach to general-purpose optimization used in VisualDOC is considered to be the most effective way to expand the use of optimization in industry, to make optimization a true everyday design tool.

References

[1] *VisualDOC Design Optimization Software, Version 2.0, Getting Started Manual*. Vanderplaats Research and Development, Inc., 1767 S. 8th St., Suite 100, Colorado Springs, CO, February 2001.

[2] *iSIGHT, Version 3.1, Developer's Guide*. Engineous Software, Inc., Morrisville, NC, April 1998.

[3] *LMS Optimus, Revision 2.0, Users Manual*. LMS Numerical Technologies, Leuven, Belgium, October 1997.

[4] *DOT. Design Optimization Tools, Version 5.0, Users Manual*. Vanderplaats Research and Development, Inc., 1767 S. 8th St., Suite 100, Colorado Springs, CO, January 1999.

[5] Z. Michalewicz and D. Dasgupta, editors. *Evolutionary Algorithms in Engineering Applications*. Springer Verlag, 1997.

[6] J. Kennedy and R. C. Eberhart. Particle Swarm Optimization. *Proceedings of the 1995 IEEE International Conference on Nerual Networks, Perth, Australia*, pages 1942–1948, 1995.

[7] J. H. Garcelon, V. O. Balabanov, and J. Sobieski. Multidisciplinary Optimization of a Transport Aircraft Wing using VisualDOC. *Proceedings of the AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit, St. Louis, MO*, AIAA-1999-1349:1306–1313, April 12-15 1999.

[8] K. Wipke, M. Cuddy, and S. Burch. ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach. *IEEE Transactions on Vehicular Technology*, 48(6), ISSN 0018-9545, November 1999.

[9] J. H. Garcelon. Implementing Optimization in ADVISOR using VisualDOC. *Proceedings of the ADVISOR Users Conference, Costa Mesa, CA*, pages 54–72, August 24-25 2000.

[10] J. H. Garcelon, K. Wipke, and T. Markel. Hybrid Vehicle Design Optimization. *Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA*, AIAA-2000-4745, September 6-8 2000.

[11] *GENESIS Version 7.0 Users Manual*. Vanderplaats Research and Development, Inc., 2001.

[12] *FLUX2D. Users Manual*. CEDRAT Corporation, Meylan, France, 2001.

[13] *FLUENT 6.0 Getting Sstarted*. Fluent, Inc., 2001.

[14] *GAMBIT 2.0 Getting Sstarted*. Fluent, Inc., 2001.