



COMPUTER SOFTWARE PRODUCT END ITEMS

CAD TOOLS

Submitted to:

SPAWAR Systems Center San Diego
53560 Hull Street, Code 2206
San Diego, California 92152-5001

Attention:

Mr. Mark Tukeman, Code 2712
(619) 553-1510

In fulfillment of the requirements for:

Contract No. N66001-02-D-0039
*Development of Port Related Transportation Technologies
to Advance Military Responsiveness to National Needs*

Task Order No. 0011
Computational Fluid Dynamic (CFD) Design Tool Development and Validation

CDRL Data Item No. A001BA

Security Classification: Unclassified

Prepared and Submitted by:

Center for the Commercial Deployment of Transportation Technologies
California State University, Long Beach Foundation
6300 State University Drive, Suite 220 • Long Beach, CA 90815 • 562.985.7394

Approved for public release: distribution is unlimited

September 30, 2003

Computer Software Product End Items

Computational Fluid Dynamics (CFD) Tool Development

Deliverable 3

CAD-Based CFD Tool development and Application

P/E Number: 2.20

Task Order Number: 11

POC

Hamid Hefazi, Professor & Chair

Mechanical and Aerospace Engineering Department

California State University, Long Beach

hefazi@csulb.edu

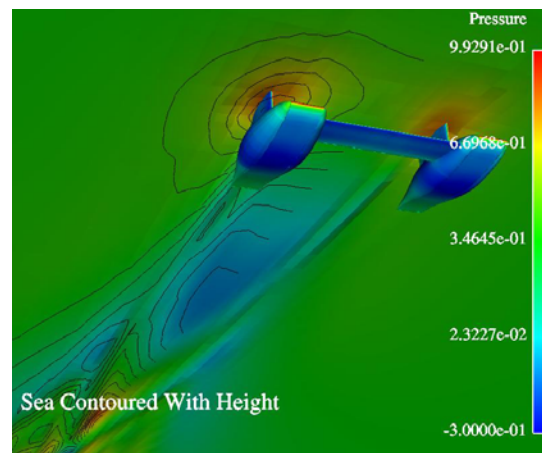


Table of Contents

1. Description of the method	1
1.1. Overall approach.....	1
1.2. Classical optimization approach	3
1.3. Neural-Network-based optimization approach.....	6
2. Use of the design/optimization method	7
2.1. Classical optimization approach and training/validation set generation.....	7
2.2. Neural-Network-based optimization approach.....	8
3. Summary/Conclusion	9
4. Glossary	9
5. Acknowledgments	9
6. References.....	10

This report presents the description of the optimization method based on Neural Networks and their integration into the CAD-based Computational Fluid Dynamics (CFD) tool design optimization process. Section 1 presents the approach and describes its components. It also compares and contrasts the Neural Network (NN) based method with the classical approach. Section 2 describes how the approach is implemented based on existing commercial products. The focus of the presentation is on the integration of the components of the design/optimization tool.

1. Description of the method

1.1. Overall approach

The three main components of a numerical optimization method for aerodynamic and/or hydrodynamic optimization are, (1) the representation of a configuration by a set of parameters called design variables (D.V.), \mathbf{x} , (2) the optimization method, and (3) the evaluation of the objective function (f), which, in CFD, is the aero- or hydrodynamic performance for that given configuration.

The general optimization process is illustrated in Fig. 1. An initial set of design variables, which might represent the configuration designed by experienced engineers, is supplied to the optimizer. Then, for this design, the objective function, f , is evaluated and the constraints, g_i , are analyzed to check whether they are violated or not. If the optimum is not reached, these values are fed back to the optimizer which modifies the D.V.'s. The process is repeated until convergence.

In the classical case, the objective function calculation requires the use of a CFD code that solves the equations of fluid dynamics and evaluates the aero- or hydrodynamic performance needed to perform the optimization. This calculation is usually very computer intensive – and drives the cost of the optimization process – with one flow solution taking from several minutes to several hours or days in the most complex problems, even using massive parallel computations.

In the approach using neural networks (NN), the process is divided in two parts: (1) generation of a training set and training the neural network, and (2) optimization of the configuration, like in Fig. 1, with the NN instead of the CFD code. The Network is trained with a relatively small set of data, which can be a mix of experimental data and/or

data generated with the CFD code. Then, this network is coupled with a global optimization method without the drastic increase in CPU time since the NN allows for a very fast evaluation of the aerodynamic performance. All other elements of the optimization approach, such as configuration representation and selection of the numerical optimization algorithm are currently independent of the use of Neural Networks to predict the aero/hydrodynamic properties.

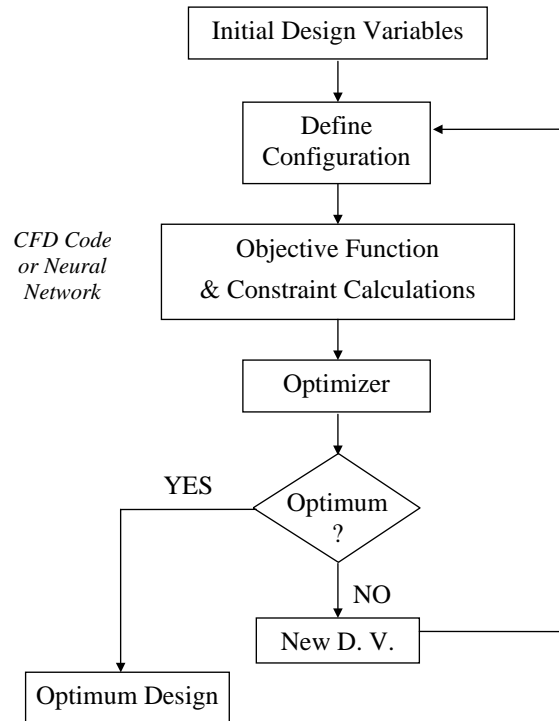


Fig. 1 Flowchart of the numerical optimization. In the NN approach, the CFD code is replaced by a NN evaluator which has been trained *a priori*

Instead of having the cost buried in the function evaluations during the optimization, and therefore not easily known *a priori*, the approach using Neural Networks (NN) moves the cost to the generation of the training set. By controlling the size of the training set (TS), not only the cost can be known *a priori*, but also reduced. Knowing the cost of the optimization offers advantages from a programmatic stand point because it allows managers to know exactly when they will get a solution to the problem at hand and also what the cost will be. In addition, the algorithm allows for external data, such as previous suitable computations or experimental data, to be included in the training set if desired.

The elements of classical and neural-network-based optimization methods are discussed in Sects 1.2 and 1.3, respectively.

1.2. Classical optimization approach

The classical optimization approach is described in detail in Ref. 1. It is included here because it allows comparing the classical process with that of the neural network-based approach, and because the latter approach will use a process similar to this one for training and validation set generation. See Sect. 1.3 for details.

The method integrates several commercially available software packages with several scripts, which, once setup, perform all tasks automatically, without user intervention. This automatic setup is critical in the optimization process. *iSIGHT* [2] controls the process and calls the various scripts, which, **from the design variables**,

1. Define and generate the new geometry (Pro-Engineer, [3]);
2. Check for any constraint violation (from output of Pro-Engineer);
3. Generates a suitable mesh (ICEM-CFD, [4]);
4. Executes the CFD method; and
5. Extracts the data needed by *iSIGHT* (**objective function and constraint values**) and calculate the **constrained objective function**, f_c , for the next iteration.

The process is described in Fig. 2. The constrained objective function is such that, when at least one constraint is strongly violated, f_c is set close to f_{max} . f_{max} is typically on the order of 1 and corresponds to a normalized value of the maximum objective function. The normalization value is given by the user and is typically chosen at the higher values of expected f over the search space. For example, with an optimization where L/D is the objective function on the order of 10-15, a normalization value of 10 to 20 can be chosen. Choosing 10 would mean that f_c might reach 1.5.

Two positive parameters ε_1 and ε_2 are now defined. If $\forall i, g_i \leq -\varepsilon_1$, then the constrained objective function is f . If $\exists i \mid g_i \geq \varepsilon_2$, then the penalized cost function gets close to f_{max} depending on how the constraints are violated. In other words, ε_1 decides when constraints become active, and ε_2 when they become prohibitive.

The generalized constraint G is defined as

$$G(x) = \frac{1}{n_{con}} \left(\sum_{g_i \geq -\varepsilon_1} g_i(x) + \varepsilon_1 \right)$$

and the constrained objective function becomes

$$f_c(x) = [1 - \varphi(y)] \cdot f(x) + \varphi(y) \cdot f_{\max} \cdot \left(1 - \frac{e^{-y}}{2}\right)$$

where

$$\varphi(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ e^{-1/x^2} & \text{if } x > 0 \end{cases}$$

and

$$y = \frac{10G(x)}{(\varepsilon_1 + \varepsilon_2)}$$

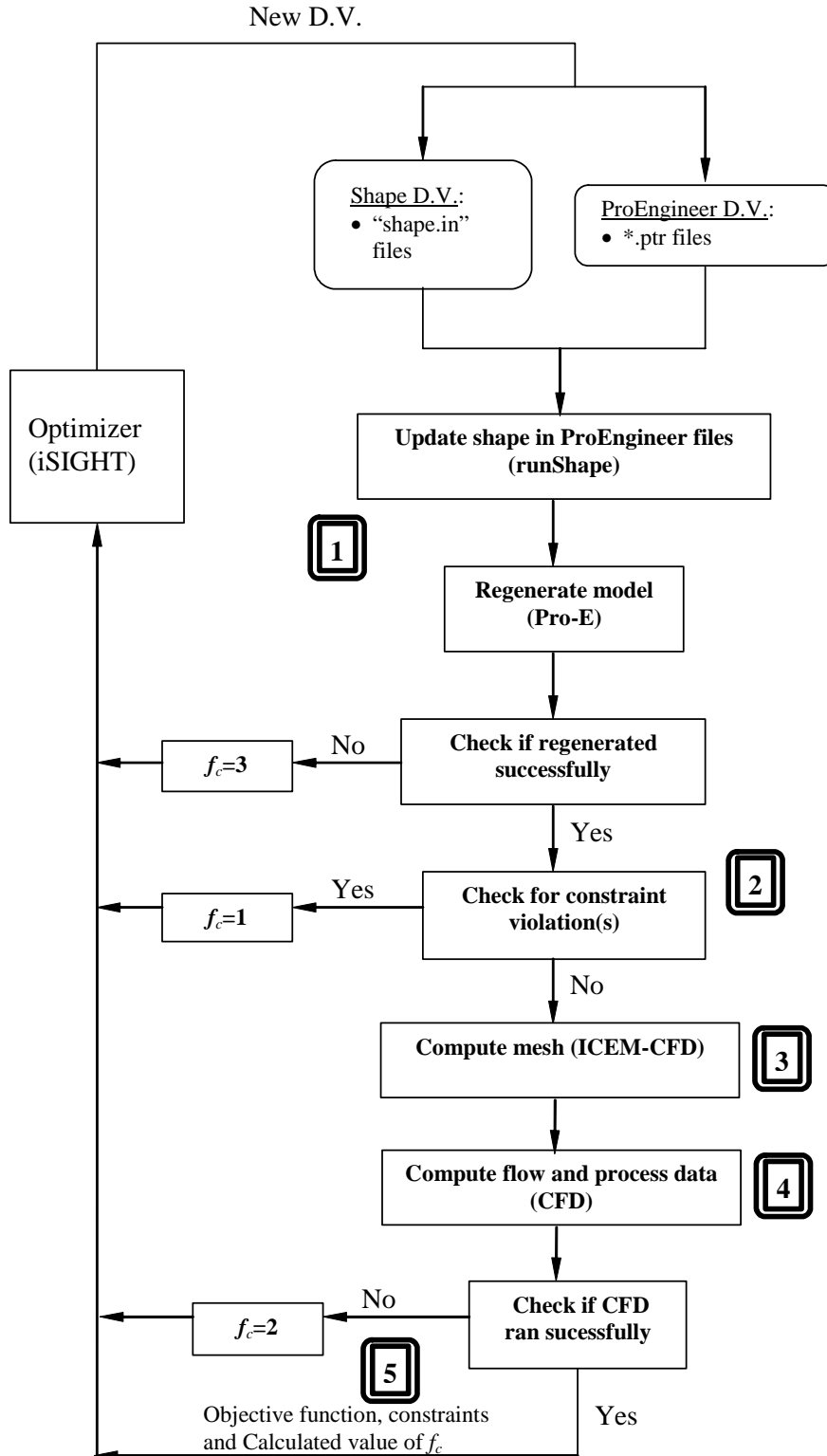


Fig. 2 Optimization process with classical approach – Also used in Neural Network (NN) Training Set (TS) generation with Latin Hypercube in iSIGHT instead of optimizer

1.3. Neural-Network-based optimization approach

As discussed in Sect. 1.1, the use of the Neural Network (NN) approach encompasses three steps:

1. Generation of the Training Set (TS) & Validation Set (VS)
2. NN training to obtain a NN “evaluator(s)”
3. Optimization as in Fig. 1 with the NN evaluator(s)

A training set (TS) corresponds to a set of known data points (design variables and their associated values, such as objective function(s) and constraints) used to train the NN, i.e. the network attempts to achieve an output which matches the input (training set). A validation set (VS) is a set which, unlike the TS, is *not* used for training per say, but rather is used for *stopping* the training. The purpose of the VS is to avoid over-fitting which can occur with Cascade Correlation. These are described in detail along with the theoretical foundation and NN methodology in Ref. [5].

Step 1 is done as described in Sect. 1.2, using the same approach as the classical one, but one in which the “optimizer” tool within iSIGHT replaced by a Latin Hypercube sampling program. Step 1 is done for the VS and for the TS. The outputs are 2 datasets which are used in Step 2 for training the network.

The training program is a C++ software in which the Cascade Correlation algorithm. The outcome of the training is a NN in the form of an executable in which the proper number of hidden units and corresponding weights – found during training – have been implemented. This approach of generating such executable for each network was selected for efficiency during the optimization process. The other approach would consist in printing a list of weights and hidden unit information. This information would have to be read at each function call during the optimization process, thus requiring substantial amount of computer input/output (I/O) time when compared with the short CPU time needed for the actual computation. Also printed is error information (w.r.t. TS and VS) for the user to see what kind of accuracy should be expected during optimization. Ref. [5] describes the Neural Network and its characteristics, as well as how the software is used. Ref. [6] shows its application to a model problem.

The third step is the same as that of the classical problem, but instead of linking the optimization tool to the CFD code, the tool is linked to the executable generated in Step 2. Fig. 3 shows how the elements of the method are linked together.

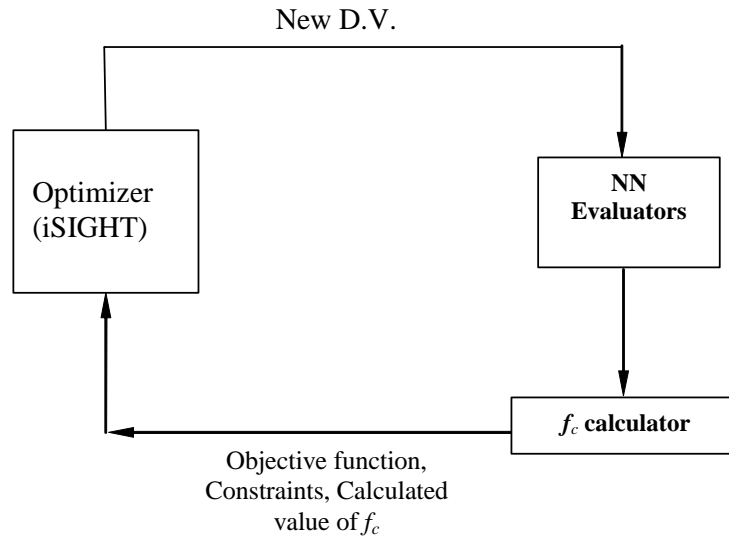


Fig. 3 Optimization process with NN approach

2. Use of the design/optimization method

2.1. Classical optimization approach and training/validation set generation

As discussed above, the training/validation set generation and the classical optimization method differ only in the choice of process within the iSIGHT tool. Latin Hypercube sampling is used in the former when an optimization scheme, such as Genetic Algorithm, is used in the latter. This process is shown in Fig. 2 and includes the steps of Sect. 1.1.

Step 1, geometry definition from design variables, is problem dependent in its implementation and, in general, involves two parts:

1. Airfoil shape definitions (“shape.in” files)
2. Configuration parameterization (“.ptr” files)

Each individual component of the configuration is represented by a set of parameters and, once assembled, these components describe the entire configuration. Simple geometrical parameters, such as width, length, chord, etc., are used to characterize the elements. These correspond to the “Configuration parameterization”. Foil cross-sections are defined by their mean camber line and thickness distributions. Camber is parameterized by classical NACA 2-parameter camber [7]. The thickness distribution, y_{th} , is represented by a 6-deg. polynomial, with an additional term for controlling the leading edge radius:

$$y_{th}(x) = a_0 \sqrt{x} + \sum_{i=1}^6 a_i x^i, \quad 0 \leq x \leq 1$$

In general, the optimizer updates values of parameters both sets of shape and configuration parameter files. Some or all parameters may be updated. Depending on the problem, one or several independent scripts or programs are used to regenerate updated Pro-Engineer files (“.ptr” files). Then, Pro-Engineer automatically updates the geometry based on the revised data.

In Step 2, constraints which may be determined from the newly generated solid model, such as volume, structural constraint, etc., are evaluated. In Step 3, a suitable mesh is automatically generated using ICEM-CFD. This mesh is then used along with the CFD input data file to execute the CFD code (Step 4). When using an Interactive Boundary Layer approach [8] for the CFD, a surface mesh is sufficient. Note that this surface mesh includes, in general, trailing wakes behind lifting surfaces as well as free-surface definition. See Ref. 9 for more details. On the other hand, the use of a Reynolds-Averaged Navier-Stokes method requires the use of a volume mesh.

Step 5 involves the use of a constrained objective function which integrates both the objective function and constraints. It is used when a global optimization method (such as Genetic Algorithms) is used within iSIGHT. If a gradient-based optimization method is used, objective function and constraints are used separately from one another.

This method is described in more detail and applied to the design/optimization of several lifting body configurations in Refs. 1 and 9.

2.2. Neural-Network-based optimization approach

The application of the NN-based optimization method involves the steps described in Sect. 1.2. As discussed in that section, generation of the TS and VS is performed using the process of Sect. 1.1 and described in more detail from a user point of view in Sect. 2.1.

Once training and validation sets have been obtained, they are used to generate an executable. The process for generating such executable as well as validation test cases are described in Refs. 5 and 6. The outcome of the training is a set of evaluators which come in the form of executable files. These executable files are then linked to one another following the standard procedures described in iSIGHT’s User’s Manual and online help.

This latter part does not require the use of any special scripts but relies exclusively on the ability of the NN “evaluator” to replace the CFD tool.

3. Summary/Conclusion

This report describes the development of a CAD-Based CFD and optimization tool. This work is an extension of previous CCdoTT work. Classic optimization approach used previously is replaced by a Neural Network based optimization approach. The new approach substantially reduces required computer resources and makes the method more suitable for hull design application. The accompanying Software Test Report describes the application of this new method to one of the Pacific Marines underwater hull forms.

4. Glossary

Acronyms

CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
NN	Neural Network
D.V.	Design Variable
CPU	Central Processing Unit
TS	Training Set
L/D	Lift to Drag Ratio
VS	Validation Set
I/O	Input/Output

5. Acknowledgments:

The CCDoTT project is a collaborative effort involving several faculty and students at California State University, Long Beach. Staffs primarily responsible for this portion of the work are Dr. Eric Besnard Associate Professor, Adeline Schmitz, Research Associate and Stanley Baksi, graduate student in the Department of Mechanical and Aerospace Engineering.

6. References

1. H. Hefazi, et al “CFD Design Tool Development and Validation, CCDoTT FY00 Task 2.8”, Center for the Commercial Deployment of Transportation Technologies, Long Beach, CA, Feb. 2002. Available on line at www.ccdot.org
2. Engineous Software Inc., iSIGHT software, <http://www.engineous.com/>
3. Parametric Technology Corporation, Pro/Engineer software, <http://www.ptc.com/>
4. ICEMCFD Engineering Inc., ICEMCFD software, <http://www.icemcfd.com/>
5. H. Hefazi, et al. “Computer Software Product End Item, Deliverable 2, Optimization Tool Development Based on Neural Networks Computer DI-MCCR-80700 report” Center for the Commercial Deployment of Transportation Technologies, Long Beach, CA, and Sept. 2003.
6. H. Hefazi, et al. “Software Test Report, Deliverable 2, Optimization Tool Development Based on Neural Networks DI-IPSC-81440A report” Center for the Commercial Deployment of Transportation Technologies, Long Beach, CA, and Sept. 2003.
7. Abbott, I.H. and Von Doenhoff, A.E. *Theory of Wing Sections*, Dover Publications, Inc., NY, 1959.
8. T. Cebeci, *An Engineering Approach to the Calculation of Aerodynamic Flows*, Horizon Pub., July 1999.
9. H. Hefazi et al, “CFD Design Tool development and Validation CCDoTT FY01 Task 2.14”, Center for the Commercial Deployment of Transportation Technologies, Long Beach, CA, January 2003. Available on line at www.ccdot.org