A Metamodel-driven Interactive Framework for a Designer Assistance System

Stefan Menzel, Markus Olhofer, Bernhard Sendhoff

2010

Preprint:

This is an accepted article published in Proceedings of the 11th International Design Conference – Design 2010. The final authenticated version is available online at: https://doi.org/[DOI not available]

INTERNATIONAL DESIGN CONFERENCE - DESIGN 2010
Dubrovnik - Croatia, May 17 - 20, 2010.

DESIGN 2010

A METAMODEL-DRIVEN INTERACTIVE FRAMEWORK FOR A DESIGNER ASSISTANCE SYSTEM

S. Menzel, M. Olhofer and B. Sendhoff

Keywords: designer assistance system, interactive design, free-form deformation, metamodel, neural network

1. Introduction

The design of innovative objects which equally serve aesthetic and technical principles is a very complex process. With respect to automotive design, multiple criteria influence the final shape of the object, say shape of a car or of the car's interior, which are a combination of e.g. human creativity, aerodynamic performance, regulation constraints and stability issues. While it is often possible to assign an aerodynamic or stability performance index to the currently developed shape, the creative process as an interaction between human ideas and technical issues is very difficult to measure. Usually, a prototype design is generated iteratively: the designer suggests new concepts based on technical regulations and feedback from aerodynamic or stability evaluations of previous designs. However, one serious difficulty during the design process is the different times required for the different evaluation methods and the different frequency of evaluations. As far as stability or aerodynamics is concerned, the evaluation of the design performance is very time-consuming and costly. A Computational Fluid Dynamics (CFD) calculation of the flow around a car may take up to one or even several days depending on the required accuracy and available computing power. Thinking of wind tunnel experiments the costs are even higher and the number of possible experiments accordingly very low. As a consequence, providing the human designer with a direct, fast and visually understandable feedback on newly applied object modifications in CAE is very difficult.
As a solution for purely technical design optimizations, i.e., shape optimizations with no or only minor aesthetic requirements, it is common to integrate so-called metamodels, like e.g. neural networks, for modelling the quality landscape [Jin et al. 2002], [Menzel & Sendhoff 2008]. Practically, a neural network is learned on already available data samples each consisting of a parameter set and an associated quality value, like e.g. the result of a CFD simulation. Using the trained neural network the performance value for a given parameter set can be approximated very rapidly, hence, allowing a fast estimation of possible successful parameter variations.
In the present paper, we suggest an interactive framework which is based on metamodels to achieve a powerful designer assistance system. In this context, the term metamodel refers to any kind of computational algorithm which allows a fast estimation of the performance landscape for a given design space. By a combination and analysis of already existing domain data collected from e.g. wind tunnel experiments, CFD or FE (finite elements) calculations adequate metamodels are generated which should support (and subliminally influence) the decision making process of the human designer. Nevertheless, in our opinion it is important, that the metamodels should be understood as an *assisting* tool, the whole process must be driven by the creativity of the individual designer. However,

we believe that if reliable metamodels can be generated for a given problem, e.g. for estimating the aerodynamic flow around a car, the human designer may think of various alternatives to achieve both, aesthetic quality and a high technical performance at the same time.

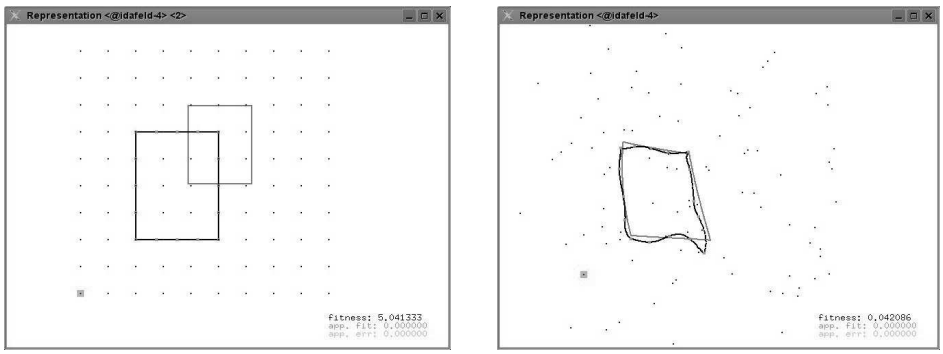

Figure 1. Designer Assistance Tool: Geometries a) before and b) after hybrid optimization

In the following, we describe a simplified model for the assistance system. For the understanding of the paper it is very important to notice that we use target shape matching (described in Section 2) as a replacement for a costly technical evaluation method like e.g. complex fluid-dynamics or finite element computations. Also the design environment is highly simplified as can be seen from Figure 1. Nevertheless, the central elements of our interactive framework, namely, the use of metamodels for modelling the technical evaluation and the interaction with the designer are already included in the described framework. We combine the possibilities of integrating human object manipulation to create shape variations and visualizing the feedback given by a feed-forward neural network which we use as a metamodel for predicting the technical performance of the created shape. For the realization of human object manipulation standard Free-Form Deformation (FFD) algorithms are applied because of their intuitive and fast modification capabilities as well as their practical relevance.
Within the proposed framework, we will evaluate the performance of design development with and without neural networks for performance estimation. The experiments show that metamodels are capable of improving the development process towards performance increase. Especially in local search areas which allow only very small shape modifications to keep an already achieved design it is worthwhile to integrate metamodels.
The paper is organized as follows. In Section 2, the system layout is described which integrates a graphical visualizer, an interactive FFD model, neural networks as performance predictor and an evolutionary optimizer. FFD is explained in more detail in Section 3, followed by the technical description of the evolutionary optimizer in Section 4. In Section 5 neural networks are briefly introduced. Experimental results are discussed in Section 6, followed by concluding remarks.

2. System framework

To demonstrate the advantage of the proposed framework, a small-scale system has been realized which reflects the major characteristics of a comparable real-world application. The system requires a combination of several methods from different research fields in a graphical user interface which allows the human user to interact with.
Firstly, it is important to choose an object representation for the geometrical shapes which is computationally fast, flexible and whose changes are intuitive. For the presented application, standard Free-Form Deformation (FFD) [Sederberg & Parry 1986], [Coquillart 1990] is chosen because it allows intuitive design variations while keeping a manageable number of design parameters. As it will be described in Section 3 in more detail, shape variations are realized by a repositioning of visible control points which are the handles for geometric modifications.
Secondly, the system maintains a database. In real world applications, the database would contain domain knowledge which already exists from prior design studies or which is generated during the current ongoing development. Usually, each data sample consists of a set of parameters describing the

actual shape and one or more performance values. In our experiments, the parameter set consists of the x- and y-coordinates of a subset of points describing the design. Concerning the performance indicator it is important for the present study to circumvent time-consuming quality evaluations, like e.g. CFD simulations would be, for analyzing the system capabilities. Hence, we decided to indicate the technical performance, or so-called quality of each design by a calculation of a target shape match, expressed as mean distance *f*. As it will be explained in Section 3, a target shape is defined for each experiment and each quality corresponds to the distance *f* between the currently visible shape and a hidden target shape.

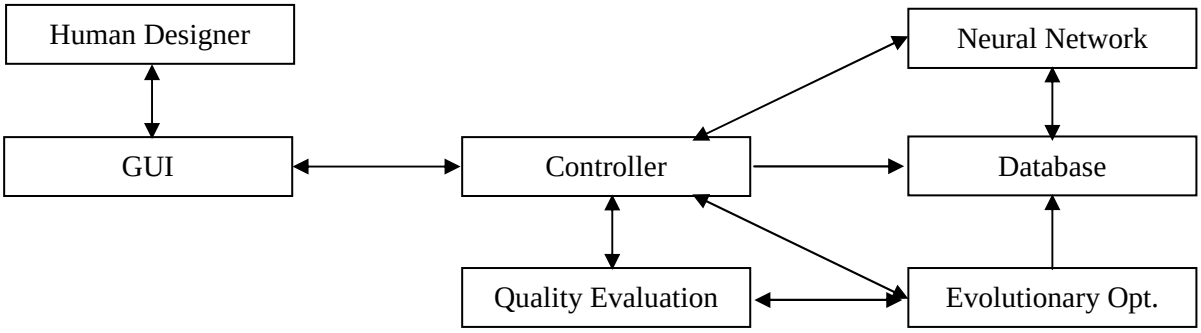

Figure 2. Schematic overview of the system

To generate new data samples, the system provides two alternative ways. On the one hand the human user may modify the currently visible shape via the control point handles. On the other hand an evolutionary algorithm may be started which carries out a design optimization to minimize the distance between the actual and the target shape, thus, generating a large number of data samples and indicating the closest optimal shape. The optimizer is introduced in more detail in Section 4.
At the heart of the systems is the metamodel which is learned on the data available in the database. The model will be discussed in more detail in Section 5. Currently, a feed-forward neural network is used as a metamodel which allows the estimation of the quality landscape. The neural network results are displayed in the GUI to support the human user in his/her decision making process.
Summarizing from the point of view of the human user, the designer interacts with the geometry via the GUI as exemplarily depicted in Figure 1. The designer chooses whether the actual visible shape is deformed manually using the control point handles or if a computational optimization is carried out. It is also possible to iteratively apply manual and computational shape variations (hybrid method). In any case, both modifications result in a deformation of the initial geometry and the associated technical performance, i.e. the mean distance *f*, of the actual design is visible in the lower right corner of the GUI. Upon request by the designer a neural network is learned on the existing data and used to estimate the quality for possible control point movements. The predicted quality values are graphically visualized in the GUI, hence, providing the human designer with information on possible ways to improve the design technically before actually carrying out the design modification. Thus, the human user is free to accept or to ignore the neural network information and is still responsible for further design development steps.

3. Representing design changes by Free-Form Deformation

The choice of an adequate representation is very important for the performance of the whole system and its ability to generalize. We require three important characteristics towards the representation. Firstly, it should be able to deal both with rather complex and rather simple geometries to allow a good scaling behaviour. Secondly, the human interaction should be intuitive and close to systems which may be used for practical applications. Thirdly, the parameter set for the metamodel should be easily extractable to evaluate the performance of the model and to allow a fast visualization of the prediction results. We have chosen standard Free-Form Deformation (FFD) [Sederberg & Parry 1986], [Coquillart 1990] as the representation of the shapes. FFD has been originally developed for soft object animation in the field of computer graphics. To realize design modifications, in a first step

a chosen base design which can be of any kind of complexity is embedded in a lattice of control points which is called control volume. Thereafter, the base design is transferred to the parameter space of the control volume, a process which is called freezing. Mathematically, cubic B-splines are used as base functions which provide a high degree of locality and flexibility. The deformations are achieved by the repositioning of control points and the calculation of the new shape based on the frozen parameter coordinates. An example is depicted in Figures 3.a) and 3.b). A simple square is initially embedded in a 10x10 grid of equidistantly distributed control points and frozen to the control volume. A movement of the marked control point $P_{5/3}$ by 1.6 in positive x-direction (Figure 3.b) generates the illustrated shape.

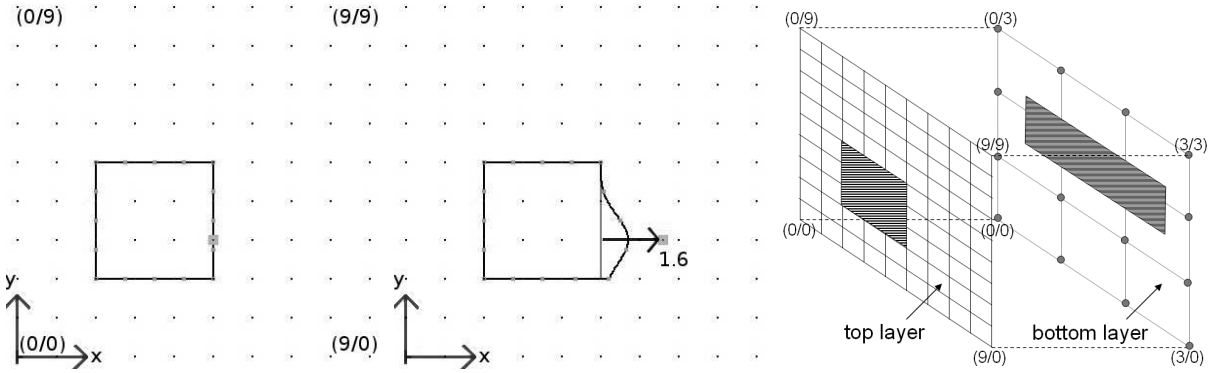

Figure 3. a) A square embedded in a standard FFD control volume, b) deformation of a square by FFD, c) realization of dynamic FFD by coupling a top and bottom layer control point grid

Summarizing, FFD allows the realization of interactive and intuitive shape modifications and additionally decouples the complexity of a chosen base design from the parameter set on which the designer, either a human or a computational algorithm, is working on. Because of these advantages FFD has already been applied to the evolutionary design optimization of complex systems in the aerodynamic problem domain [Menzel & Sendhoff 2008]. Aerodynamic problems are characterized by a non-linear quality landscape, thus making it very difficult for a human designer to technically improve an already optimized design. Often shape modifications and performance changes are coupled indirectly and, thus, are very difficult to predict. Nevertheless, especially these kinds of problems are targeted by our conceptual framework. However, in the simplified set-up we want to avoid the time-consuming CFD calculations in the conceptual research phase. Hence, a substitute system is suggested which can be used to validate our methods and to demonstrate the performance of the outlined framework without costly computational simulations.

As depicted in Figure 3.c) we propose a system of two coupled FFD control volumes, each one with an associated base geometry, to reproduce similar non-linear quality landscapes. One FFD control volume which is called top layer is modifiable by the designer, a second control volume which is called bottom layer is not modifyable by the designer. Instead, each modification which the designer applies to the control points of the top layer is transferred by a set of pre-defined equations to the control points of the bottom layer. These pre-defined equations are the key for adjusting the complexity of the quality landscape.

For the experiments presented in this paper the bottom layer is calculated as follows: Given the initial absolute positions of the bottom layer control points $P,B_{i,j}$ and the relative modifications of the top layer control points $\Delta P,T_{i,j}$, the modified absolute positions of the bottom layer control points are computed by

$$P,B_{i,j}{}^{*} = P,B_{i,j} + \sum_{a=-2}^{2}\sum_{b=-2}^{2} m_{a,b} \cdot \Delta P,T_{(3i+a),(3j+b)} \quad \forall i,j \in [0,3]. \tag{1}$$

where $m_{a,b}$ is a factor according to Figure 4.a). The transfer functions and different sizes of control volumes have been chosen in such a way to generate local design deformations on the top layer and more global ones on the bottom layer.

m	a=-2	a=-1	a=0	a=+1	a=+2
b=+2	0.00	0.00	0.33	0.00	0.00
b=+1	0.00	0.33	0.67	0.33	0.00
b=0	0.33	0.67	1.00	0.67	0.33
b=-1	0.00	0.33	0.67	0.33	0.00
b=-2	0.00	0.00	0.33	0.00	0.00

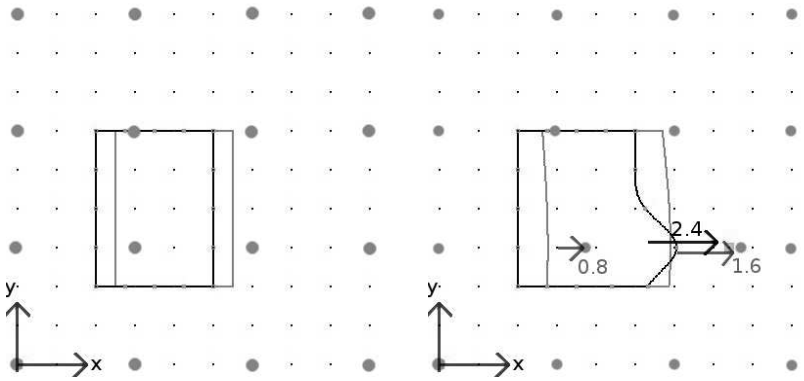

Figure 4. a) Table of multiplier values $m_{i,j}$, b) Initial top and bottom layer rectangle, c) Coupled deformation of top and bottom layer geometry by manual repositioning of control point $P,T_{5,3}$

An exemplary system of two coupled FFD volumes is depicted in Figure 4.b) and 4.c). For the evaluation scenario presented in this paper, the modifyable top layer consists of 10x10 equidistantly distributed control points marked by small black dots, the bottom layer of 4x4 equidistantly distributed control points marked by large grey dots. The geometries placed in both layers can be chosen arbitrarily. Nevertheless, for the introduction and understanding of the effects of the coupled FFD volumes, two rectangles are generated but with varying positions on the grid. The black rectangle which is associated with the top layer is positioned at the coordinates (2/2, 5/6) whereas the grey rectangle associated with the bottom layer is positioned at (2.5/2, 5.5/6). To demonstrate the local influence of the top layer $P,T_{5,3}$ has been moved by 2.4 in x-direction. Because of the cubic B-splines the influence on the black rectangle is locally in the lower right corner. Following eq. (1) $P,B_{1,1}$ and $P,B_{2,1}$ have to be updated. The resulting modification of the grey rectangle is also visible in Figure 4.c). Because of the smaller number of control points of the bottom layer the whole rectangle is deformed. For calculating the "technical" performance, i.e. the shape matching quality expressed as mean distance *f*, both rectangles are represented by 400 points starting in the lower left corner counting clockwise with 100 points lying on each edge. Each point is given by *x* and *y* coordinates ($x_{T,i}/y_{T,i}$) and ($x_{B,i}/y_{B,i}$) respectively. Hence, the quality value *f* is calculated by the mean distance of both shapes in the following way:

$$f = \frac{1}{400}\sum_{i=1}^{400}\sqrt{(x_{T,i}-x_{B,i})^2\cdot(y_{T,i}-y_{B,i})^2}\ . \tag{2}$$

Thus, the optimal design is found if both shapes are congruent. During the experiments which are explained in Section 6 the bottom layer geometry is *not* visible to the human designer. Furthermore, since each top layer modification is transferred to the bottom layer in a way which is also unknown to the human designer, it is a rather difficult task to achieve two rather congruent shapes manually. But since we are also interested in the comparison between the quality values resulting from manual modifications and the maximally achievable quality, an evolutionary optimizer has been integrated into the framework which is able to compute the optimal shape for a given initial shape. The algorithm is presented in Section 4.

4. Design optimization using evolutionary algorithms

The framework which is proposed in the present paper allows two different ways for shape variation. On the one hand, the human user can manually reposition the control points to deform the design. Hence, the technical performance of the design may or may not improve. On the other hand it should also be possible to search for the design which is optimal in a technical sense, i.e. which provides the minimal distance to the target shape. One possible way to calculate these designs is the application of an evolutionary optimization which can deal with a larger number of optimization parameters and non-linear quality landscapes.

Evolutionary algorithms belong to the group of stochastic optimisation algorithms. They mimic the principles of Neo-Darwinian evolution, see e.g. [Fogel 1995], by applying operators for reproduction, mutation and/or recombination and selection. Prominent examples are Evolution Strategies (ES), Genetic Algorithms (GA) or Genetic Programming (GP), respectively. Among the advantages of evolutionary algorithms are robustness against noisy or discontinuous quality functions, the ability to escape from local optima and to enable global search. In the course of optimization a population of possible solutions, e.g. a vector of continuous parameters, is adapted to solve a given problem over several generations. The adaptation occurs by variation of solutions contained in a population and by selection of the best solutions for the next generation. The variations can be classified as purely stochastic, usually called mutation, and combinatoric/stochastic, usually called recombination or in the context of genetic algorithms crossover.
In this paper, a special variant of Evolution Strategies, the Covariance Matrix Adaptation-ES (CMA-ES) [Hansen & Ostermeier 2001], is applied which has the advantage of a high convergence rate for real-valued problems compared to other evolutionary algorithms. The successful application of this type of algorithm has been shown previously for two- and three-dimensional turbine blade optimizations in the aerodynamic domain [Menzel & Sendhoff 2008].

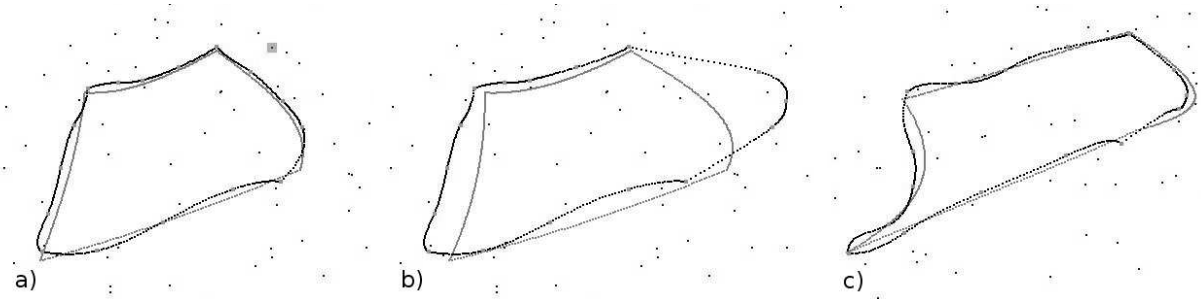

Figure 5. Hybrid optimization: a) 1. optimized design (f=0.056), b) manually modified design (f=1.006), c) optimized design (f=0.049)

The evolutionary optimization is carried out in four steps. After setting up the FFD system as described in Section 3, at first, the current 10x10 x- and y-control point positions of the top layer $P,T_{i,j}$ are stored sequentially in the parents' chromosome vector. Next, the chromosome vector is copied for each newly generated offspring and each value in the vector is slightly modified according to the mutation operator. Based on the mutated control point positions $P,T^*_{i,j}$ the deformed top layer shape is calculated. At the same time, the control points of the bottom layer $P,B^*_{i,j}$ are updated following eq. (1) and the bottom layer shape is deformed. The mean distance f between both shapes is computed by eq. (2) and assigned to the offspring. Finally, the selection is carried out based on a quality comparison and the chromosome vectors of the successful offsprings are copied to the parents' chromosome. By repeating these steps the distance f between both shapes is minimized over the course of generations. Figure 5 depicts an examplary hybrid optimization. Initially, a rectangular shape similar to the one shown in Figure 4.b) has been optimized by the evolutionary algorithm resulting in design 5.a) with an associated shape matching quality f of 0.056. Secondly, the human user has modified the shape by manually introducing a bump on the right side of the design which downgrades the quality to 1.006. As it is visible in Figure 5.b), the bottom layer geometry is also modified because of the transfer functions as described in Section 3. Thirdly, restarting the evolutionary optimization leads to a different optimal design as found in 5.a) with an associated quality of 0.049. This example illustrates the existence of different local optima which are dependent on the current position in the design space. For aerodynamic problems this effect is very common since many spatially distributed design parts interact in the system as a whole.

5. Performance prediction by Feed-Forward Networks

Metamodels are widely used in the artificial intelligence domain for substituting costly computations which themselves are models simulating mathematical equations like e.g. the Navier-Stokes equations

in fluid-dynamics. Therefore, the name metamodel refers to a model of a model. Artificial neural networks are one possibility to realize a metamodel for modelling linear or non-linear systems. An artificial neural network is given by a composition of single neurons according to a specified topological arrangement. The topology is given by the neural connectivity and the connection weights.

In the system proposed in this paper, we chose a standard fully-connected feed-forward network [Bishop 1995] as implemented in the Shark library [Shark] with one input, one hidden and one output layer to predict the quality. The input layer contains 32 neurons, the hidden layer 4 neurons and the output layer 1 neuron, the activation functions of each neuron are sigmoidal. As stated above, the technical performance of each system is calculated by the distance between the top layer and the bottom layer geometry following eq. (1). The task of the network is to predict the technical performance, i.e., to return a quality value without calculating the mean distance explicitely. Therefore, the process falls into two steps. In a first step, a set of data samples is collected in a data container which is used to train the network. In a second step, the trained network can be used to estimate the quality.

Since the number of input neurons strongly influences the required number of training data, we chose a reduced set of 16 points on the top layer geometry, basically each 25^{th} point of the 400 points which define the rectangle as depicted in Figure 6.a). For each of the 16 points of each data sample the difference of the current top layer geometry to the initial top layer geometry in x- and in y-direction is calculated, thus, each data sample in the database consists of 32 input parameters. Additionally, for each data sample an associated performance value is available. For the training of the network we split the database in a set of training data and a set of validation data and learned the network's weights by backpropagation [Bishop 1995] on the collected data.

Figure 6.c) illustrates the feedback of the results of a trained neural network to the user. If the user selects a control point for possible variation, virtual control point modifications around the current position are carried out. Based on these modifications the 16 chosen points on the geometry are updated and used for estimating the performance value by the neural net. The results are visualized by a color map to the user, bright colors indicate promising control point movement directions whereas darker regions should be avoided from a technical viewpoint. Furthermore, if the user points on one of the evaluated squares, the predicted technical performance value is displayed providing additional information about the gradient of the quality change.

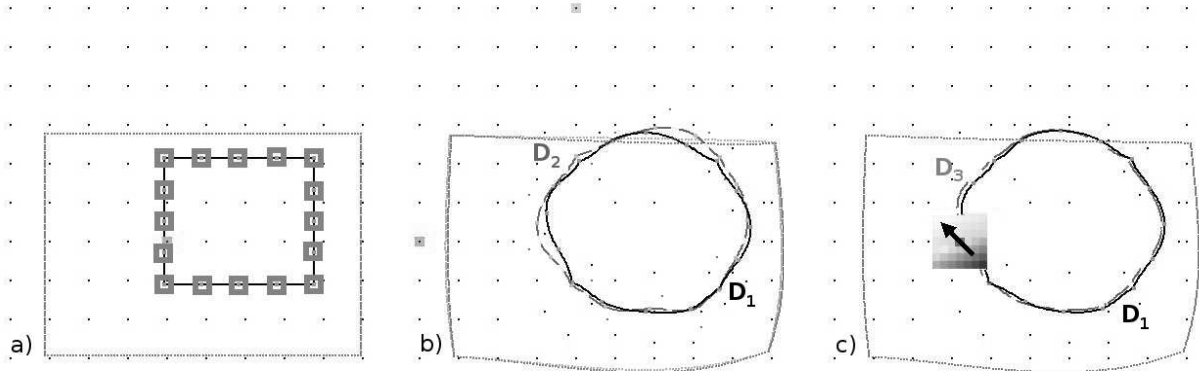

Figure 6. a) Initial experiment set-up (f=2.12), b) user's manually created circle-like shape D_1 (f_{D1}=1.975) and user's manually optimized shape D_2 (f_{D2}=1.931), c) shape D_1 (f_{D1}=1.975) and user's optimized shape using neural network assistance D_3 (f_{D3}=1.949)

6. Experiment: Metamodel assisted design development

In several experiments, the difference between manual design development with and without metamodel assistance is compared. A set of 36 single experiments has been carried out to study and illustrate the differences. The workflow of each single experiment is as follows: At the beginning the human user sees the GUI visualizing the initial top layer geometry, a black rectangle with random height and width placed at a random position, as depicted in Figure 6.a). In order to understand the

set-up of the experiments, the initial bottom layer geometry, a grey rectangle with random height and width placed at a random position, is also visible in Figure 6.a), However, note that during all experiments the gray shape is *not* visible to the user, so the user has initially no explicit knowledge on possible design optima. The bottom layer geometry is always hidden to keep the user visually unclear about where to move the control points to improve the technical performance.

In a first phase, the user was asked to deform the rectangle to a circle, or circle-like geometry, by moving control points. This phase mimics aesthetic design modifications, hence, it was not required to minimize the technical performance but to generate a shape according to the users liking (or to the specifications give, i.e., a circle). Nevertheless, the user received already a quality value for each modification s/he carried out which is displayed in the GUI as absolute quality value according to eq. (2). Thus, during this phase the target for the user was to get a feeling for the interaction between control point modifications and geometry deformations as well as which modification increased or decreased the quality. The result of this phase which consisted of 50 steps was a circle-like geometry as it is shown in Figure 6.b) and described as the "user's aesthetic design" D_1 with an associated quality value f_{D1}.

During the second phase, the user was asked to decrease the quality value while still trying to keep the D_1 shape (i.e., stay circular) for 15 steps. Thus, the user could decide on her/his own on the range of control point movements, i.e. s/he could realize smaller or larger shape changes. The result is described as the "user's optimized design" D_2 as illustrated in Figure 6.b) with a corresponding quality value f_{D2}.

At the beginning of the third phase the geometry has been reseted to D_1. A neural network has been trained on the 50 data samples which have been generated in the first phase. During the third phase the user was asked to manipulate the control points based on the neural network predictions. After selecting a control point for movement, the neural network calculated a quality estimation map for possible movements directions. The quality predictions have been graphically visualized as depicted in Figure 6.c) and the user moved the control point according to the most promising directions. The resulting shape D_3 with an associated quality value f_{D3} is illustrated in Figure 6.c).

To compare both ways of design development for each experiment the performance differences between D_2 and D_1 as well as D_3 and D_1 have been recorded as $f_{D2,D1} = f_{D1} - f_{D2}$ and $f_{D3,D1} = f_{D1} - f_{D3}$ respectively. Furthermore, since the quality change has to be evaluated with respect to the shape change, the geometric distances between D_2 and D_1 as well as D_3 and D_1, called $d_{D2,D1}$ and $d_{D3,D1}$, are measured. These geometric distances are calculated by the mean distance analogue to eq. (2) where the bottom layer geometry is replaced by D_1.

So far, we have calculated the performance and shape changes for each experiment. Nevertheless, to compare all experiments it is necessary to scale both parameters. Thus, we added an additional fourth step after each experiment has finished. In this step, the geometry has been reseted to D_1 again and an evolutionary shape optimization has been carried out to find a shape D_4 as depicted in Figure 7.b) which corresponds to the closest technical optimum, i.e. a shape D_4 with the highest technical quality corresponding to the least distance to D_1. By D_4 it is possible to calculate the maximum distance decrease $f_{D4,D1} = f_{D1} - f_{D4}$ which the user could achieve manually and its corresponding shape distance $d_{D4,D1}$. Based on these results for each experiment the quality difference f^* and the shape change d^* have been scaled following eq. (3):

$$f^*_{D2,D1} = 1 - \frac{f_{D2,D1}}{f_{D4,D1}}, d^*_{D2,D1} = \frac{d_{D2,D1}}{d_{D4,D1}}, f^*_{D3,D1} = 1 - \frac{f_{D3,D1}}{f_{D4,D1}}, d^*_{D3,D1} = \frac{d_{D3,D1}}{d_{D4,D1}} \quad (3)$$

The final results are depicted in Figure 7.a). The plot in the upper right corner contains the results of two evolutionary optimization runs marked by grey triangles and grey diamonds which have been carried out to find the optimal design D_4 for two different initial designs D_1. Both series were scaled according to eq. (3). As it can be seen one series lies directly upon the other illustrating that by scaling each experimental result following eq. (3) a good comparison of all single experiments is possible.

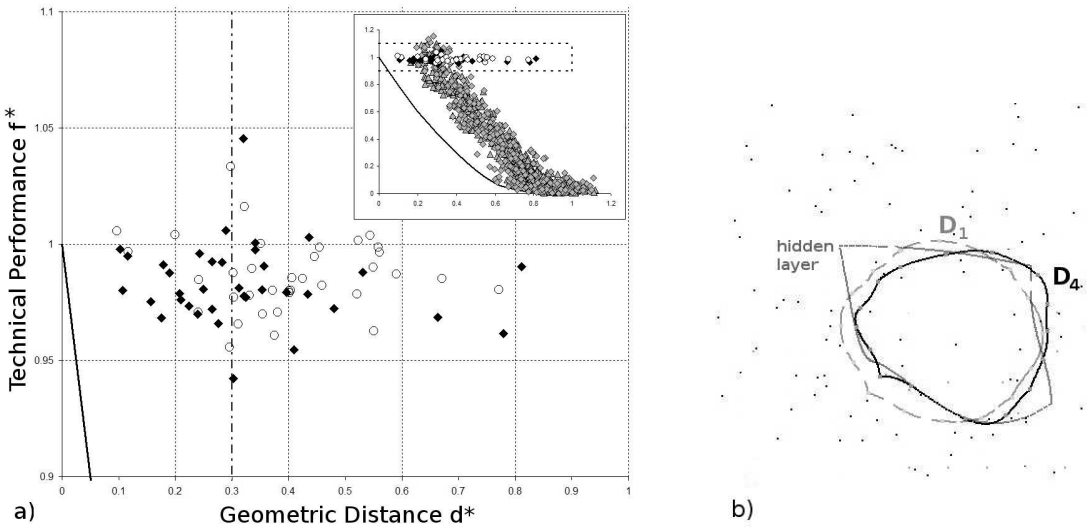

Figure 7. a) Summary of experimental results, b) Computationally optimized shape D_4 (f_{D4}=0.35)

For reasons of a better understanding we also added an estimated Pareto front (not calculated) shown as a black curve. The Pareto front represents all designs which are non-dominated by other ones, i.e. each design on the Pareto front is the best one with respect to the trade-off of distance decrease and least design variation (from the circle). Such solutions are called Pareto optimal solutions. Naturally, it is rather difficult to achieve a Pareto optimal solution by manual design modifications since these solutions require correlated movements of a large number of control points.

The 36 experimental results are in the area ranging between performance values of 0.9 and 1.1. Therefore, we enlarged this part of the plot for better visibility. The white circles mark the results based on manual design modifications without using a neural network (phase 2) whereas the black diamonds mark the results of manual design modifications including neural network support (phase 3). The geometric distance expresses the degree of shape variation of D_2 and D_3. As it can be seen, most of the design changes are in a range of 0.1 to 0.6. The technical performance improvements are up to approximately 5%. For the example given in Figures 6 and 7.b), the geometric distances are $d_{D2,D1} = 0.160$, $d_{D3,D1} = 0.079$ and $d_{D4,D1} = 0.420$, hence, $d^*_{D2,D1} = 0.381$ and $d^*_{D3,D1} = 0.189$, and the scaled quality values following eq. (3) are $f^*_{D2,D1} = 0.973$ and $f^*_{D3,D1} = 0.987$.

As depicted in Figure 7.a) the resulting designs D_2 and D_3 are sorted into two groups according to their geometric distances. The first group R_1 contains designs which are characterized by rather small shape variations with respect to D_1, i.e. $d^* \leq 0.3$, and the second one, R_2, contains designs with rather large modifications, i.e. $d^* > 0.3$. For R_1, the designs modified using the neural network outperform the ones which have been modified without using the neural network. The mean performance over all $D_{3,R1}$ is 0.983 with a standard deviation of 0.012, whereas over all $D_{2,R1}$ it is 0.993 with a standard deviation of 0.025. In R_2, the mean performance over all $D_{3,R2}$ is 0.984 with a standard deviation of 0.023, whereas over all $D_{2,R2}$ it is 0.985 with a standard deviation of 0.013. I.e. in R_2 both ways for design development are similar in the mean performance value, nevertheless the quality variation for $D_{2,R2}$ is smaller.

Concluding, if small modifications are applied, i.e. $D_{2,R1}$ and $D_{3,R1}$ are rather similar to D_1, the performance of designs developed with neural network support is higher than for the ones which have been developed without neural network support. This can be explained by the fact that the neural network has been trained on few data with only small variations and as a consequence the prediction quality of the network is high if the applied variations are rather local. In contrast, for larger modifications the performance of $D_{2,R2}$ and $D_{3,R2}$ is similar but the quality variation of $D_{2,R2}$ is smaller. In this range R_2, the human user often performed better without the model assistance because if s/he detected promising directions that improved the performance it was very likely that these directions were followed strategically, i.e. several neighbouring control points have been moved in the same directions to increase the effect. At the same time, since the modifications are larger and the collected

data did not include design regions which were not within a close distance to D_1, the metamodel gets more inaccurate decreasing the prediction performance. It is also visible in Figure 7.a) that the human user tends to carry out larger modification when s/he does not rely on the neural network. Summarizing, especially for the case that a manually optimized design should be similar to a certain design idea D_1 while still improving the technical performance, the neural network support has been very helpful.

7. Conclusions

The strict principle of "form follows function" has never been followed in automotive design (consider the fate of the Chrysler Airflow). The reason is simple: uniquely brand identifying characteristics seem more important for customer satisfaction than technical function like aerodynamics. However, times are changing: the collective consciousness towards ecological efficiency has dramatically increased. At the same time, safety requirements (European New Car Assessment Programme) have become more challenging. Therefore, there is the need to bring the technical performance of design back into the focus without losing the individual aesthetic brand styling. There is a need for form and function to meet on par. In this paper, we have outlined a conceptual framework on how this can be achieved, on how technical information can be fed back to the designer taking the current constraints of time consuming quality estimations into account. We have outlined a much simplified system of such a conceptual framework.

To evaluate the benefits of the proposed system, a software application has been implemented which includes the possibilities of interactive and intuitive shape modifications with neural network support. Using first experimental results the potential benefit of such a system has been demonstrated. The inclusion of the neural network prediction capability in the manual design optimization process has been advantageous especially for local shape variations. Such designs provided a fair trade-off between technical performance increase and small shape variation.

For real world domains as the automotive field, the utilization of metamodels to integrate a large amount of domain knowledge which is already available, e.g. from former CFD simulations or wind tunnel experiments, is very promising. A high quality metamodel together with an intuitive user interface could provide a fast feedback within seconds on the influence of possible shape variations and, hence, stimulate the creative and decision making process of the designer.

References

Bishop, C. M., "Neural Networks for Pattern Recognition", Oxford University Press, 1995.

Coquillart, S., "Extended Free-Form Deformation: A Sculpturing Tool for 3D Geometric Modeling", Computer Graphics, 24(4), 1990, pp. 187-196.

Fogel, D. B., "Evolutionary Computation: toward a new philosophy of machine learning", IEEE Press, 1995.

Hansen, N., Ostermeier, A., "Completely Derandomized Self-adaptation in Evolution Strategies", Evolutionary Computation, Vol. 9, No. 2, 2001, pp. 159-196.

Jin, Y., Olhofer, M., Sendhoff, B., "A Framework for Evolutionary Optimisation With Approximate Fitness Functions", IEEE Transactions on Evolutionary Computation, Vol. 6, No. 5, 2002, pp. 481-494.

Menzel, S., Sendhoff, B., "Representing the Change – Free Form Deformation for Evolutionary Design Optimization", Evolutionary Computation in Practice, T. Yu et al. (Ed.), Springer-Verlag, 2008, pp. 63-86.

Sederberg, T.W., Parry, S.R., "Free-Form Deformation of Solid Geometric Models", Computer Graphics, 20(4), 1986, pp. 151-160.

Shark – a modular C++ library for the design and optimization of adaptive systems. Open Source Software: http://sourceforge.net/projects/shark-project

Dr. Stefan Menzel
Honda Research Institute Europe GmbH
Carl-Legien-Strasse 30, D-63073 Offenbach/Main, Germany
Phone: +49(0)69 890 11 758, Fax: +49(0)69 890 11 749
e-mail: stefan.menzel@honda-ri.de
URL: www.honda-ri.de