DESIGN PREFERENCE ELICITATION: EXPLORATION AND LEARNING

ABSTRACT

We study design preference elicitation, namely discovery of an individual's design preferences, through human-computer interactions. In each interaction, the computer presents a set of designs to the human subject who is then asked to pick preferred designs from the set. The computer learns from this feedback in a cumulative fashion and creates new sets of designs to query the subject. Under the hypothesis that human responses are deterministic, we investigate two interaction algorithms, namely, evolutionary and statistical learning-based, for converging the elicitation process to near-optimally preferred designs. We apply the process to visual preferences for three-dimensional automobile exterior shapes. Evolutionary methods can be useful for design exploration, but learning-based methods have a stronger theoretical foundation and are more successful in eliciting subject preferences efficiently.

Keywords: Design Preference Elicitation, Genetic Algorithm, Statistical Learning, Active Learning

INTRODUCTION

User preference plays an intricate and yet essential role in engineering design. There is no universal acknowledgement on how preferences should be measured, modelled and elicited. Systematic elicitation of preferences can augment insights with rigorous optimization processes and reveal potential trade-offs between engineering and marketing objectives [1]. Standard econometric practice models for preferences using utility theory and model (parameter) estimation require extensive market or survey data. Such models have been adopted and extended in recent engineering design research [2-4], mainly to demonstrate how we can solve a design problem while incorporating consumer preference considerations. There are some limitations to adopting these models. The data acquired for preference model estimation can handle rather limited design attributes (or variables) since increasing the number of attributes raises the prediction error of the preference model when data size stays the same. Further, the questionnaires used for acquiring data from user surveys, either for discrete choice or choice-based conjoint approaches, tend to be tedious and thus bias the models. Researchers have suggested ways to enhance the data collection efficiency (e.g., Hoyle et al. on D-efficient sampling [5], Abernethy et al. on adaptive sampling [6]). These works have mathematical merit but limited practical appeal as a human-computer interaction mechanism. In the present work, we are interested in eliciting individual design preferences rather than aggregate ones. Indeed, such preferences may be elicited not just from users but also from any other stakeholder in the design process: designers, marketers, or producers. To pursue the goal of efficient preference elicitation with an appealing interaction experience, we investigate two algorithmic approaches for generating design choices for the subject: evolutionary and statistical learning-based. Our findings show that statistical learning-based methods can limit the number of repetitive iterations while evolutionary methods can be more useful for design concept exploration. We demonstrate these findings in experiments where subjects express preferences on automobile exterior shape designs. These experiments are limited and further validation will be necessary.

The paper is organized as follows. Section 2 elaborates the problem and reviews the relevant literature. Section 3 introduces briefly the online 3D parametric modelling and interaction program we developed for this study. We then delve into the preference elicitation algorithms using the evolutionary and statistical learning methods in Sections 4 and 5, respectively. Section 6 concludes the paper with discussion on some drawbacks of the proposed algorithms and provides directions for future improvement.

PROBLEM FORMULATION AND LITERATURE REVIEW

The following notation and assumptions will be used throughout the paper. Let $\mathcal{D} \subset \Re^p$ be a continuous design space where any $\mathbf{x} \in \mathcal{D}$ represents a design with p variables. Assume that an individual's preference is a function defined on $\mathcal{D}$, i.e., $f(\cdot): \mathcal{D} \rightarrow \Re$. The objective of preference elicitation is then to find an $\mathbf{x}$ that returns 'near-optimal' $f(\mathbf{x})$. This is an optimization problem with an objective function unknown to the analyst but known to the subject. 'Near optimal' means that the termination point will be in the vicinity of the optimum according to some termination criterion, but the precise optimum may not be found. The subject-computer interaction is designed in the fashion of a numerical search algorithm, where in each iteration the subject is presented with a set of designs and asked to evaluate them. The computer learns from this feedback and presents new designs based on its accumulated knowledge about this subject's preferences. The termination criterion is that the subject can no longer discriminate the presented design with respect to his or her preference.

Below we look first into research addressing this problem using evolutionary computation. This type of study usually assumes the existence of a preference measure $f(\mathbf{x})$ derived from user interaction. We then introduce a statistical learning method called active learning [7] and show in a later section that searching based on active learning will be effective even when the preference measure does not exist explicitly.

Interactive Evolutionary Computation (IEC)

IEC is a branch of evolutionary computation where the fitness of designs is assigned by the user. Notable contributions include work by Sims (1991) for interactive genetic algorithm (IGA) [8], Johanson and Poli (1998) for interactive genetic programming (IGP) [9], and Tokui and Iba (2000) for combined interactive GA and GP [10]. Kelly and Papalambros introduced an IGA to elicit user preference on car silhouette design [11]. A comprehensive review of IEC by Takagi (2001) pointed out that a major obstacle for IEC to be practical is human fatigue, meaning that population size, number of generations and eventually the size of the design space limit IEC performance [12]. Although remedies to this obstacle have been studied [12], an essential difficulty with IEC is that asking the user to assign real-valued or ranking fitness iteratively is not a natural way to express preferences and thus causes fatigue. Choice-based user feedback is more appealing. Econometric studies are more in favour of choice-based analyses than rating or ranking-based ones. We believe the same applies to the preference elicitation problem here. However, choice responses are binary, i.e., 1 for designs picked as "preferred" and 0 for "not preferred", and thus will not work as fitness values as required in IEC.

Statistical Learning and Active Learning

In order to utilize choice responses during an interaction, we resort to a statistical learning-based method. The idea is to use the binary choices data, i.e., 'preferred' and 'not preferred', as classification labels for each design and to create a decision boundary from each observation. Iterative interactions help to refine these boundaries, which approximate the separation between preferred and not preferred designs from the user.

Like for evolutionary methods, the key question here is how the next generation of designs can be generated from user feedback so that the elicited preferences become more refined. We examined the classification problem previously in the context of black-box optimization [13], and realized that the same problem occurs in active learning. Therefore, we review some of the most relevant developments in active learning below and we adopt a modified version of the SIMPLE algorithm developed by Tong and Koller [7] in the later demonstration.

Formally, let a classification problem be finding a set of decision boundaries for observations $\{\mathbf{x}_j\}_{j=1}^n$ with labels $\{y_j\}_{j=1}^n$. When the size of observations is large, querying labels can be costly. Active learning refers to methods where the algorithm starts by querying labels for a subset of $\{\mathbf{x}_j\}_{j=1}^n$ and, from that, finds new query samples using the current classifier. This approach potentially helps to reduce the number of queries and still 'learn' sufficiently well [7]. Formulating the classification task using a support vector machine (Vapnik, 1998 [14]), Tong and Koller proved that efficient queries can be made by cutting the space of the classifier coefficients into equal halves. Research in computer

science has examined the balance between exploration (querying without using knowledge gained) and exploitation (querying with all knowledge gained, like by Tong and Koller) to achieve more stable performance on different problems (Osugi, 2005 [15] and Baram 2004 [16]). Research in marketing science has looked also into a similar problem. Toubia and Hauser (2003) improved the sampling efficiency of Choice-Based Conjoint (CBC) analysis by adapting questions based on user feedback [17]. They showed that, under a linear utility assumption, asking a user to choose among designs is equivalent to cutting the utility space. Since utility estimation can be improved by reducing the volume of the utility space, a good question to ask is one that cuts the remaining utility space into equal halves, ensuring the utility space to be halved after each iteration. Toubia and Simester (2003) showed that compared to traditional methods (e.g., CBC with random or efficient fixed samples and adaptive conjoint analysis), their adaptive CBC resulted in better estimations of the utility coefficients especially when the number of questions is limited [18]. Not surprisingly, the use of active learning to enhance adaptive CBC became evident. Chapelle and Harchaoui (2005) showed that, under the assumption of linear utility and discrete attribute levels, estimating the utility coefficients can be considered exactly as an active learning problem, with utility coefficients analogous to classifier coefficients [19]. The same view is shared by Abernethy et al. (2007) where the authors further argued that robustness of utility estimation is enhanced by introducing the concept of complexity control from SVM [6].

The difference between our method here and those on statistical learning-based conjoint analysis [6, 17-21] is that we do not search for the optimal utility estimates; rather, we search for users' preferred designs directly from the design space.

ONLINE 3D PARAMETRIC MODELLING AND INTERACTION

Parametric Modelling

The automobile exterior three-dimensional (3D) parametric model developed in this study uses 151 control points and 32 Bezier surfaces. The control points are then controlled by 33 variables ranging from 0 to 1. Therefore every point in the 33-dimensional unit box represents a design. ***Figure 1*** demonstrates some random designs generated by the program. A parametric model can represent only limited geometric styles. In future studies, a set of different modeling codes will need to be combined to represent a richer range of designs.

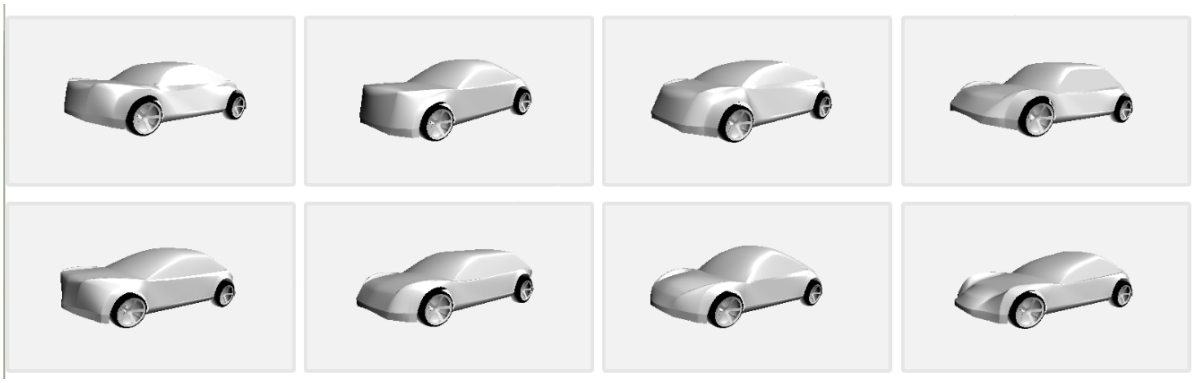

Figure 1 A variety of random designs created from the 3D parametric modeling program. The program utilizes the WebGL library so that the user-computer interaction can happen directly online.

Online Interaction

The parametric modeling program was implemented using WebGL [22] that provides application programming interfaces (APIs) for 3D visualization in web browsers. Using the AJAX platform [23], the client-side user inputs are transferred back to the server-side where the accumulated user data are processed and the algorithm creates new samples. Our current implementation ensures rapid response from the server side, while relatively high-end computers are needed to process the visualization code in real-time on the client-side.

GENETIC ALGORITHM APPROACH

Generational and Steady-State GAs

A generational GA refers to the case where information about previous generations will be obsolete when a new generation comes into existence. Thus the mechanism of creating a new generation depends only on the fitness of the current generation. In contrast, a steady-state GA keeps record of all generations created so far and of their fitness. The new generation in this case is created using the accumulated knowledge. Assuming that the preference of a user is fixed, i.e., the preference function is deterministic, a steady-state GA is favoured since it utilizes more information than the generational one. However, there is a drawback. Consider when real-valued fitness is assigned. Comparison of fitness cross generations is less meaningful since the user assigns fitness only by comparing designs within a generation, and the accumulated fitness may not reflect reality.

Fitness Calculation

In order to circumvent the drawback of steady-state GA but utilize its advantages, we design a fitness calculation scheme as follows: In each iteration, all existing designs, including the new generation, are shown to the user. The user is asked to select preferred designs. A design that has been picked in previous iterations can be picked again. We then define the fitness as the normalized count of a design being selected. The advantages of this scheme are: (1) the user can update preferences throughout the interaction, and (2) binary choice is a more natural way for users to express their preferences than rankings or ratings. The disadvantage of the scheme, however, is that it assumes a higher count to correspond to a more preferred design. This may not be the case. As an example, design *A* that appears early can be picked repeatedly because it is a design relatively better than the others present; once a better design *B* comes into play, the user may drop *A* and pick *B* which will have only one count. This is akin to situations described in voting processes (see, e.g., Saari [24]). More sophisticated schemes can be developed to address this issue, but this learning-based method is a simple way to address binary choice user feedback.

Parent Selection, Crossover and Mutation Schemes

The parent selection, crossover and mutation schemes are standard in our implementation. Although there exists a variety of parent selection schemes differing in computational cost, the difference is trivial for this study since only four new designs are generated each time due to the limited screen space available. A multi-armed roulette wheel is thus used for parent selection and *Figure **2*** shows its procedure.

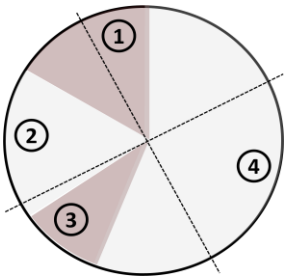

Figure 2 An illustration of a multi-armed roulette wheel parent selection. The pie is sliced into four segments representing fitness of the four candidates. A multi-armed roulette wheel is rotated randomly to the shown position. Thus chromosomes 1 and 2 are picked once and chromosome 4 is picked twice. The resulting parents are the set (1, 2, 4, 4).

The four parental chromosomes are paired deterministically in the pattern of (1, 3) and (2, 4). We utilize two random parameters to perform the crossover for pairs: the first parameter is used to split one chromosome a into two pieces:

$$a = [a_1, a_2].$$

Once each chromosome in a pair is split, we use the second parameter α to calculate the children a' and b':

$$a' = [\alpha \cdot a_1 + (1-\alpha) \cdot b_1, \alpha \cdot b_2 + (1-\alpha) \cdot a_2],$$
$$b' = [\alpha \cdot b_1 + (1-\alpha) \cdot a_1, \alpha \cdot a_2 + (1-\alpha) \cdot b_2].$$

The split and recombination parameter allows fusion of design features from parents as can be observed in ***Figure 3***.

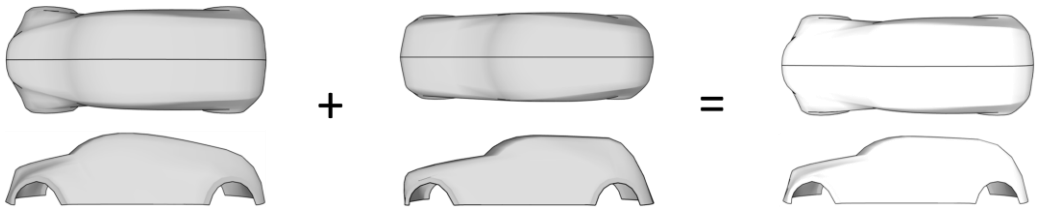

Figure 3 An example of crossover. Here the two grayed designs are parents and the white one is a child. The child acquires the hood design of the first parent and the silhouette design of the second parent. Due to the linear combination parameter ***α****, these two features of the child are not exactly the same as those from the parents.*

Mutation takes place for every design variable in the form:

$$x_i' = \max\{\min\{x_i + \delta_i, U_i\}, L_i\},$$

where x_i' is the ith mutated variable, U_i and L_i are the upper and lower boundaries of variables. Further, the random mutation δ_i is controlled by:

$$\delta_i = \delta\big(-1 + 2U(0,1)\big)\left(\exp\left(c \cdot \frac{U(0,1)}{\sqrt{n}}\right) - 1\right).$$

Here $U(0,1)$ is a realization of the standard normal distribution. Normal distribution is used because two thirds of the realization will lay within one deviation; therefore the mutation is usually small, but significant mutation is also allowed with non-zero chance. The exponential term is used to force mutation shrinkage. The two parameters δ and c are determined using two criteria: 1) The probability of having mutation greater than 0.1 at the 20th iteration should be less than 5% (all variables are between 0 and 1), ensuring algorithm convergence; and 2) the probability of having mutation greater than 0.8 at the first iteration should be greater than 5%, allowing a significant mutation to explore the designs space. ***Figure 4*** shows these two probabilities as functions of δ and c, according to which the choice $\delta = 0.4$ and $c = 1.5$ will satisfy the requirements.

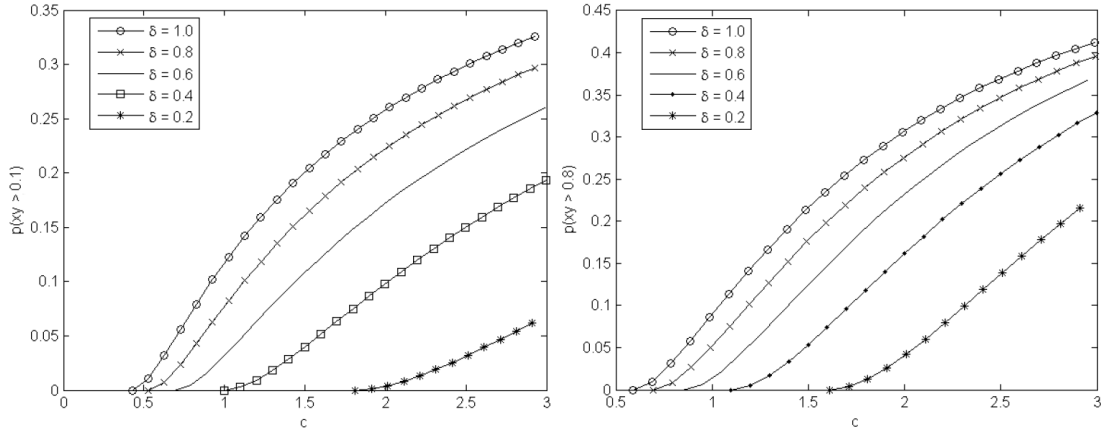

Figure 4 The two criteria as functions of ***δ*** *and* ***c****:* ***P****(mutation greater than 0.1 at the 20th iteration) on the left and* ***P****(mutation greater than 0.8 at the first iteration) on the right.*

Genetic Algorithm Drawbacks

The setup discussed above can be useful when we explore the design space interactively. However, besides the issue with fitness calculation and the ambiguity of setting up the parameters, it is also possible the GA will not converge if there are multiple preferred designs. In reality, this is often the case, e.g., a user may prefer both a sporty-looking vehicle and a safe-looking one although the two are very different in terms of the underlying design variables.

STATISTICAL LEARNING APPROACH

We investigate a statistical learning approach with the motivation that an IGA by nature cannot handle binary choice data properly and will not converge to multiple preferred designs. Since binary data are natural for classification, we attempt to represent regions of preferred designs using decision boundaries, and use an iterative process to refine these boundaries. This iterative process is inspired by the active learning technique reviewed earlier. Below we discuss modifications made to the active learning algorithm for it to work properly in this study.

Interaction Using Active Learning

First, we describe the interaction procedure: In the first iteration, the computer generates a set of l designs using a Latin-Hypercube design. The user evaluates designs and labels them as preferred or not preferred. Let the number of preferred design be m. The computer then creates the decision boundaries based on the user feedback using support vector machine. In the fashion of the SIMPLE active learning algorithm proposed by Tong and Koller, $l - m$ new designs are created on the current decision boundaries and presented to the user, along with the m preferred designs from the previous turn. Thus the best designs so far are always compared with other new designs. *Figure **5*** summaries this procedure.

input : Design space $\mathcal{D}$, max_iteration k, max_sample l
output: Decision boundary $g(\mathbf{x}) = 0$

// Initialize data set
$\mathbf{X} = \emptyset$; $\mathbf{y} = \emptyset$;

// Initialize a Latin-Hypercube sample set
$\{\mathbf{x}_j\}_{j=1}^{l} =$ `LQScatter` $(\mathcal{D}, l)$;

// Request user feedback
$\{y_j\}_{j=1}^{l} =$ `Label` $(\{\mathbf{x}_j\}_{j=1}^{l})$;

// Record the initial observation
$\mathbf{X} \leftarrow \{\mathbf{x}_j\}_{j=1}^{l}$; $\mathbf{y} \leftarrow \{y_j\}_{j=1}^{l}$;

for $i = 2$ **to** k **do**
  // Start active learning if observe two classes, otherwise continue to explore $\mathcal{D}$
  if $\{y_j\}_{j=1}^{l}$ *has ones and zeros* **then**
    // Train cumulated data to get the decision function
    $g(\cdot) \leftarrow$ `Train` $(\mathbf{X}, \mathbf{y}, svm\_options)$;
    // Scatter $l - m$ new sample points on $g(\cdot) = 0$, where m is the number of preferred designs from the previous iteration
    $\{\mathbf{x}_j\}_{j=1}^{l-m} =$ `Scatter` $(g(\cdot), \mathcal{D}, l - m)$;
  else
    $\{\mathbf{x}_j\}_{j=1}^{l} =$ `Scatter` $(\emptyset, \mathcal{D}, l)$;
  end
  // Request user feedback. New samples are combined wit the previous preferred ones
  $\{y_j\}_{j=1}^{l} =$ `Label` $(\{\mathbf{x}_j\}_{j=1}^{l})$;
  // Cumulate observations, labels of the previous preferred designs are updated
  $\mathbf{X} \leftarrow \{\mathbf{x}_j\}_{j=1}^{l}$; $\mathbf{y} \leftarrow \{y_j\}_{j=1}^{l}$;
end

Figure 5 Pseudo code for the statistical learning-based preference elicitation algorithm.

Two major differences exist between the preference elicitation study and the active learning study: 1) in active learning, labels are deterministic, meaning that once a label is revealed, it will not change further. This does not hold in preference elicitation. Indeed a preferred design can become not preferred once a better design is presented. We will demonstrate empirically that by preserving the best designs in the comparison set allows active learning to work for preference elicitation; 2) the size of samples in active learning is usually finite, thus one can enumerate all unlabeled samples to find the one closest to the decision boundary. In preference elicitation, however, we have the entire design space to explore, in which case forcing new samples to be on the decision boundary requires a Newton-Raphson iteration as shown in ***Figure 6***.

input : Design space $\mathcal{D}$, decision function $g(\cdot)$, sample point $\mathbf{x}$
output: $\mathbf{x}'$ such that $g(\mathbf{x}') = 0$

$tol = 1e-3$; *// Set tolerance*
$i = 1$; *// Set iteration count*
$k_{newton} = 1000$; *// Maximum iteration*

// Iterate when the absolute value of decision function is more than tol
while $i < k_{newton}$ *AND* $|g(\mathbf{x})| > tol$ **do**
  $\mathbf{dx} = \frac{-g(\mathbf{x})}{\nabla g(\mathbf{x})^T \nabla g(\mathbf{x})} \nabla g(\mathbf{x})$; *// Calculate step* $\mathbf{dx}$
  $\mathbf{x} += \mathbf{dx}$; *// Update* $\mathbf{x}$
  $\mathbf{x} =$`Project` $(\mathbf{x}, \mathcal{D})$; *// Constrain* $\mathbf{x}$ *in* $\mathcal{D}$
  $i += 1$; *// Update count*
end

Figure 6 Pseudo code for a Newton-Raphson iteration to project a random sample back to the decision boundary.

It should be noted that although random sampling and projection works in the current demonstration, it would be better for the new samples to be far away from any existing ones to enhance exploration. However, such a sampling scheme requires the global optimum of the following nonlinear program:

$$\max_{x} \quad \min_{i} \{\| x - x_i \|_2\}$$
$$\text{subject to} \quad g(x) = 0.$$

Here x_i is the ith labeled sample and $\|\cdot\|_2$ is a Euclidean distance. The objective and the constraint are highly nonlinear, and so a global optimum is hard to obtain within a time tolerable to the user. Thus random sampling is used, though it is less sound theoretically.

Demonstration

We demonstrate that assuming the user has a most-preferred target design in mind, the search algorithm can approximate this design during the interactions using binary choice information only. The target design we used here is a Nissan 350Z as shown in ***Figure 8***. This design is picked because such a model can be approximated with the current model implementation. Figure 7 shows the first twelve iterations. Each iteration contains 8 designs and user-preferred designs are highlighted.

We briefly explain how user decisions were made in these 12 iterations. As users, during the early stages, we focus mainly on silhouette design: we pick the more sporty-looking designs, i.e., curvy silhouette, lower hood orientation, and longer hood. With these sporty-looking designs presented, we then scrutinize the windshield angle: in ***Figure 8***, we prefer a front windshield angle larger than the rear one. This preference is then learned gradually by the computer as can be confirmed by the fact that within the last two iterations, most designs present acquire this feature. We further look for designs with a good approximation of the target hood style. Such an effort was not fully successful since the final design picked from the 12th iteration has a different headlight style from the target. However, considering the limited total observations made, the result is encouraging.

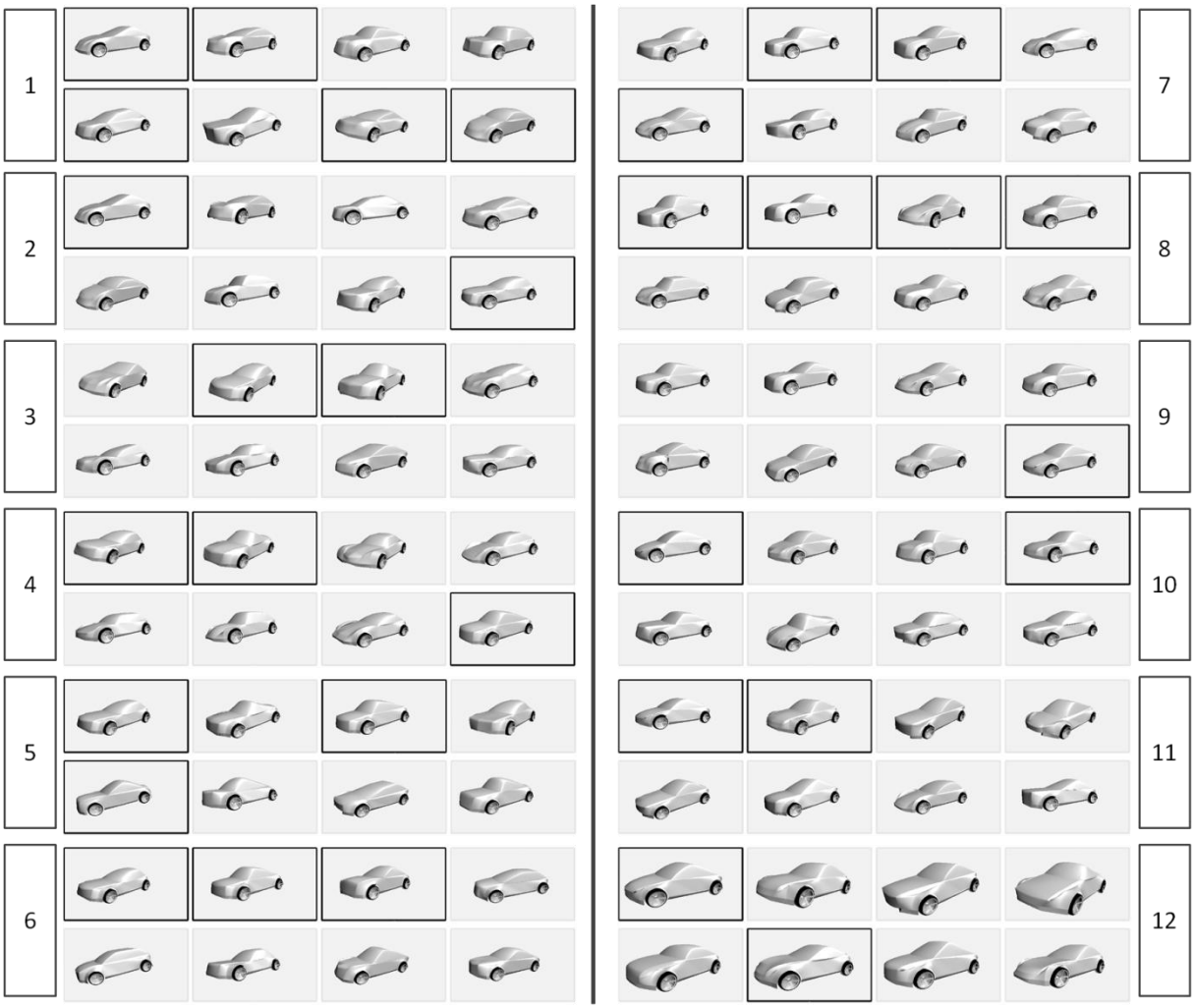

Figure 7. The first twelve iterations of a recorded interaction. The demonstration assumes that the user is looking for a roaster design resembling a Nissan 350Z. In each iteration, the computer shows 8 designs and the highlighted ones are those chosen by the user. The designs are of different perspective views since the user is allowed to rotate, pan and zoom each individual design.

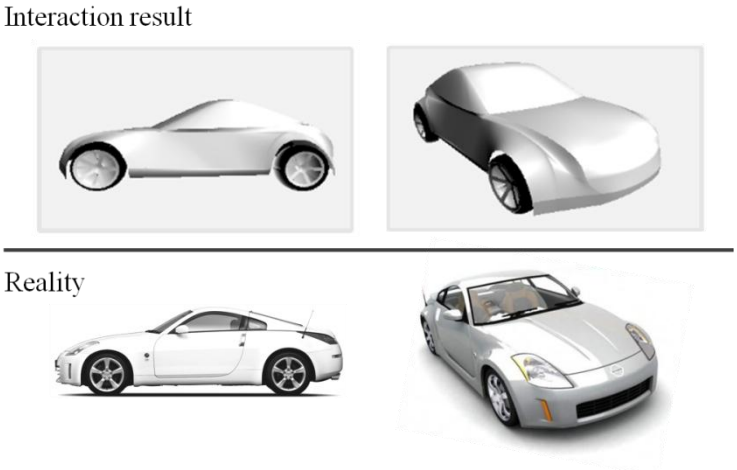

Figure 8 Comparison between the interaction results at the 12th iteration and a real target design.

CONCLUSION AND FUTURE WORK

The engineering design community has been long interested in preference elicitation. Successfully eliciting consumer preference in a certain market enables better analysis and design of the product, from a holistic rather than purely functional perspective. Traditional elicitation tools such as choice-based conjoint with utility models have drawbacks: studies show that user decision making in surveys does not always follow utility theory [23]; also, choice-based questionnaires can be long and tedious. Inspired by evolutionary computation and statistical learning, we investigated and implemented search algorithms to make the user-computer interaction more appealing while attempting to elicit the user preference at a lower time cost. The results show that while the interactive genetic algorithm can be used for design exploration, the learning algorithm is theoretically more preferred for binary choice data and can learn user preference effectively.

Further improvements possible for the proposed statistical learning-based algorithms include: (1) a more effective sampling scheme is needed to explore the design space better while constraining the new samples on the decision boundary; (2) considering that the search algorithm keeps the interaction records of each user and that the users may have some clustered preferences, the algorithm can be made more efficient when history records are used for the current search; some early simulations have proved this concept but a user-oriented algorithm has not yet been developed; (3) the car model we used in the demonstration has 31 variables in total; this large number of design variables is a major factor in slowing down algorithmic learning; segmenting the design into stages will make the interaction more efficient since users are likely to focus on only part of the design features (silhouette, hood, rear and so on.); how the design should be segmented requires expertise in that specific domain; (4) finally, the parametric car models require more details for users to perceive the designs better. The results presented here are largely theoretical.0020Experimenting with actual subjects is the natural next step in this entire line of research.

REFERENCES

[1] Michalek, J., Feinberg, F., and Papalambros, P., Linking Marketing and Engineering Product Design Decisions via Analytical Target Cascading, *Journal of Product Innovation Management*, 2005, 22(1), pp. 42–62.

[2] Wassenaar, H., Chen, W., Cheng, J., and Sudjianto, A., Enhancing discrete choice demand modelling for decision-based design, *Journal of Mechanical Design*, 2005, 127, p. 514.

[3] Kumar, D., Hoyle, C., Chen, W., Wang, N., Gomez-Levi, G., and Koppelman, F., Incorporating customer preferences and market trends in vehicle package design, *In Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2007, no. DETC2007-35520.

[4] Li, H., and Azarm, S., Product design selection under uncertainty and with competitive advantage, *Journal of Mechanical Design*, 2000, 122(4), pp. 411–418.

[5] Hoyle, C. and Chen, W. and Ankenman, B. and Wang, N., Optimal experimental design of human appraisals for modeling consumer preferences in engineering design, *Journal of mechanical design*, 2009, 131(7).

[6] Abernethy, J., Evgeniou, T., Toubia, O., and Vert, J., Eliciting consumer preferences using robust adaptive choice questionnaires, *IEEE Transactions on Knowledge and Data Engineering*, 2007, pp. 145–155.

[7] Tong, S., and Koller, D., Support vector machine active learning with applications to text classification, *The Journal of Machine Learning Research*, 2002, 2, pp. 45–66.

[8] Sims, K., Evolving Virtual Creature, *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994, pp. 15-22.

[9] Johanson, B. and Poli, R., Gp-music: An interactive genetic programming system for music generation with automated fitness raters, *Journal of Genetic Programming*, 1998, pp. 181-186.

[10] Tokui, N. and Iba, H., Music composition with interactive evolutionary computation, *Proceedings of the Third International Conference on Generative Art*, 2000, pp. 215-226.

[11] Kelly, J. and Papalambros, P.Y., Use of shape preference information in product design, *In Proceedings of International Conference on Engineering Design*, 2007, pp. 28-31.
[12] Takagi, H., Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation , *In Proceedings of the IEEE*, 2001, Vol. 89, No. 9, pp. 1275-1296.
[13] Ren, Y., and Papalambros, P.Y., Design preference elicitation, derivative-free optimization and support vector machine search, *In Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2010, no. DETC2010-28475.
[14] Vapnik, V., and Vapnik, V., *Statistical learning theory*, Vol. 2. 1998 (Wiley New York).
[15] Osugi, T., Kun, D., and Scott, S., Balancing exploration and exploitation: A new algorithm for active machine learning, 2005.
[16] Baram, Y., El-Yaniv, R., and Luz, K., Online choice of active learning algorithms, *The Journal of Machine Learning Research*, 2004, 5, pp. 255–291.
[17] Toubia, O., Hauser, J., and Simester, D., Polyhedral methods for adaptive choice-based conjoint analysis, *Journal of Marketing Research*, 2004, 41(1), pp. 116–131.
[18] Toubia, O., Simester, D., Hauser, J., and Dahan, E., Fast polyhedral adaptive conjoint estimation, *Marketing Science*, 2003, 22(3), pp. 273–303.
[19] Chapelle, O., and Harchaoui, Z., A machine learning approach to conjoint analysis, *Advances in neural information processing systems*, 2005, 17, pp. 257–264.
[20] Evgeniou, T., Boussios, C., and Zacharia, G., Generalized robust conjoint estimation, *Marketing Science*, 2005, 24(3), pp. 415–429.
[21] Toubia, O., Evgeniou, T., and Hauser, J., Optimization-Based and Machine-Learning Methods for Conjoint Analysis: Estimation and Question Design, *Conjoint Measurement*, 2007, pp. 231–258.
[22] http://www.khronos.org/webgl/, accessed on Jan. 08, 2011.
[23] http://en.wikipedia.org/wiki/Ajax_(programming), accessed on Jan. 08, 2011.
[24] Saari, D.G., Aggregation and multilevel design for systems: finding guidelines, Journal of Mechanical Design, 2010, 132(8).
[25] Hauser, J.R. and Toubia, O. and Evgeniou, T. and Befurt, R. and Dzyabura, D., Disjunctions of Conjunctions, Cognitive Simplicity, and Consideration Sets, Journal of Marketing Research, 2010, 47(3), pp. 485–496.