An Interpolation/Extrapolation Process For Creative Designing

John S. Gero and Vladimir Kazakov
Key Centre of Design Computing and Cognition
Department of Architectural and Design Science,
University of Sydney, NSW, 2006, Australia
email: {john,kaz}@arch.usyd.edu.au

Key words: Creative design, computational exploration, design combination

Abstract: This paper introduces a new computational operation that provides support for creative designing by adaptively exploring design state spaces. This modification is based on the re-interpretation of the crossover operation of genetic algorithms as an interpolation and its generalization to extrapolation. Examples of the results of the application of the process are presented.

1. INTRODUCTION

Creative designing can be defined as that class of design activity when all the variables which define the structure and behaviour are not known a priori nor necessarily are all the processes needed to produce them. The implication of this conceptualisation of creative designing is that the focus is on processes for the introduction of new variables into the design and their integration into the existing variable structure. It is suggested that this is one basis for the production of potentially creative designs.

For a given set of variables and processes operating within a bounded context any model will construct a bounded state space. Creative designing can be represented in such a state space by a change in the state space. Non-creative or routine designing does not change the state space, it simply searches within it.

Exploration is the label we give to the class of processes that changes the design state space. However, in designing we work with three state spaces: function state space, behaviour state space and structure state space. It is customary to consider only the structure state space as the space for change. The behaviour state space is dependant on the structure state space. It is possible to have two classes of state space exploration. In the first class the state space maintains the same dimensions, ie. the same defining variables, but changes over time either additively or substitutively as in Figure 1.

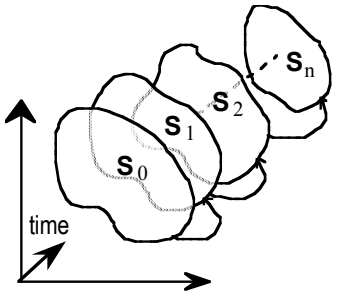

Figure 1 The change in the design state space over time. Here the state spaces (S_0, , , , S_n) keep the same dimensions since their defining variables are unchanged.

In the second class the dimensions of the state space also change with time with the change of defining variables as in Figure 2. In most current creative design systems the behaviour state space is characterised by the first class whilst the structure state space is characterised by the second class.

In this paper a process capable of playing a role in designing potentially creative designs will be introduced. A number of such processes have been developed which fall into one of the following categories: combination, mutation, analogy, first principles and emergence (Gero, 1994). We wish to commence with a process, which is able to be used in routine designing but which, under suitable conditions can be modified to produce non-routine designs. In another words, we would like to construct such a computational process that can behave either as a search or as an exploratory process, depending on the designs that are used to initiate it.

2. COMBINATION AS DESIGN TOOL

Combination is one of the well-known computational processes, which can be used to produce designs (including non-routine designs). One standard computational mechanism for combination is the crossover operation of genetic algorithms, where the genetic representation (genotype) of one design is combined with that of another. This results in a design which is different to either of its parents but which exhibits some characteristics of both. We take this genetic crossover as our starting point. However, crossover does not result in potentially creative designs since all the designs, which can be produced are already encapsulated in the genome. The basic

assumption behind the genetic crossover construction is that the genetic representation is fixed and static, implicitly defines all the designs that can be described/generated using it. In terms of the concepts in Figure 1, this means that all design state spaces are the same during design process, which is based on genetic crossover combination process ($S_0 = S_1 = \ = S_n$), Figure 2.

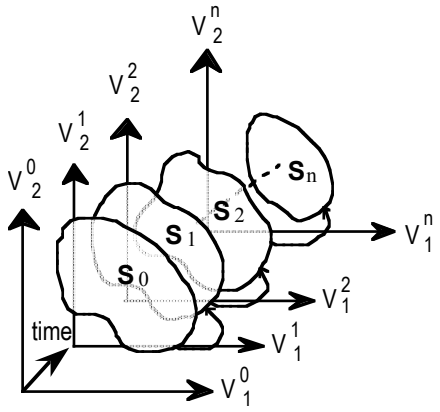

Figure 2. The change in the design state space over time. Here the state space is characterised by changing dimensions since its defining variable change over time.

In our derivations we will follow the following steps. We commence with a standard genetic crossover. Then we re-cast it as a particular type of interpolation operation. The next step is a generalization of this particular version of interpolation to a general form of interpolation. Finally, we further generalize this operator by replacing interpolation with a more general type of operation – extrapolation. As a result we construct a computational combination operator from the genetic crossover operator, that is capable of generating designs, which cannot be generated using a crossover based combination. Since each of these generalization steps includes genetic based crossover as a particular case, it is possible that both initial genetic crossover-based and generalized interpolation/extrapolation combination operators give the same results as the genetic crossover. But in the general case, they will produce different combinations from the same initial designs. In terms of state space evolution, this means that if designs to be combined belong to the state space S_0 and design-combinations produced by genetic crossover-based combination belong to the state space S_1 then the design-combinations that are produced by interpolation/extrapolation combination belong to its super-space $\mathbf{S_1}$: $S_1 \subset \mathbf{S_1}$. $\mathbf{S_1}$ can include additional dimensions compared to S_1.

3. GENETIC CROSSOVER–BASED COMBINATION

In genetic algorithms (GAs) all designs are represented in two forms: in an encoded form of genetic strings (*genotypes*) and in the decoded form of *phenotypes* – the design structures. The genetic crossover operation is the major computational engine

of GAs. As a rule, when a fixed-length genetic representation is used, the crossover operation mixes two parental genotypes in component-wise fashion, when each gene of the resulting design comes from one of the parents, Figure 3.

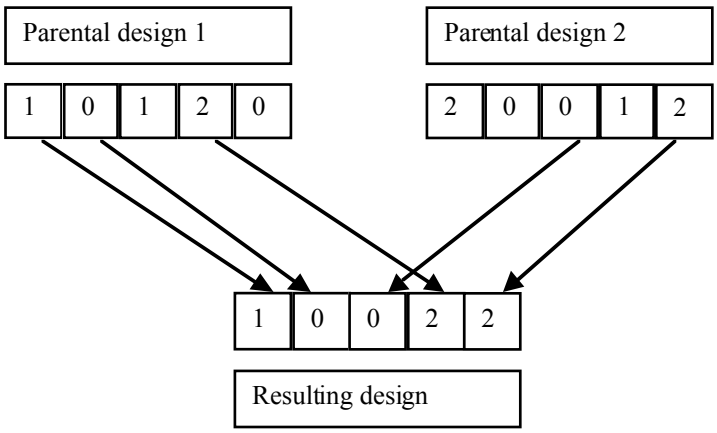

Figure 3. The standard genetic crossover function

This operation of mixing genetic components can be re-interpreted as the operation of choosing a point on a line segment between two points representing both parental designs in the space of genotypes, Figure 4.

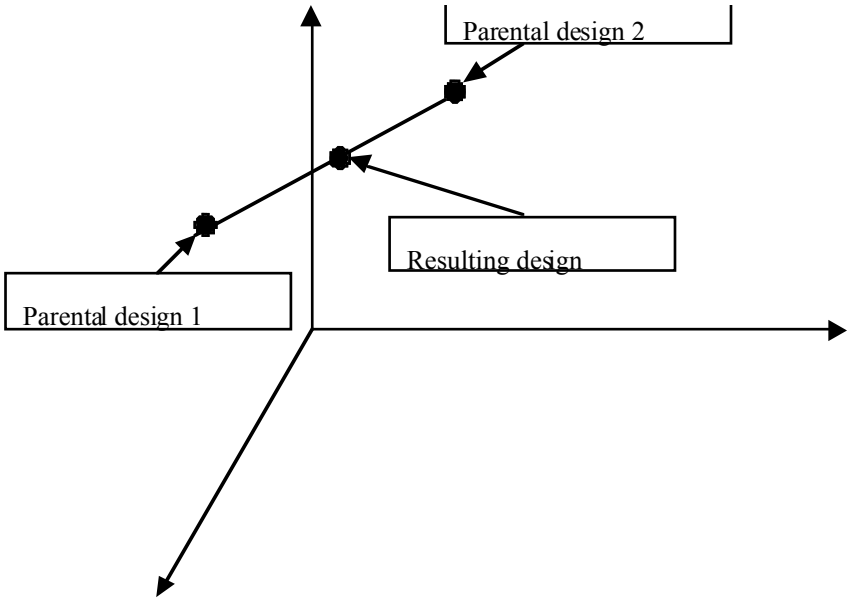

Figure 4. Genetic crossover as interpolation in the space of genotypes.

The distance, which determines what is a line segment here, is defined as a sum of the Hamming distances plus some penalty for the number of genes from different parents that are located side by side in the resulting genotype. The form of this penalty determines what version of crossover operator is used (one-point, two-point, uniform, etc.). In this paper we will assume that the one-point crossover is used. The mapping of this line segment in genetic space onto the space of actual designs produces some other type of trajectory that is, in the general case, much more complex, Figure 5.

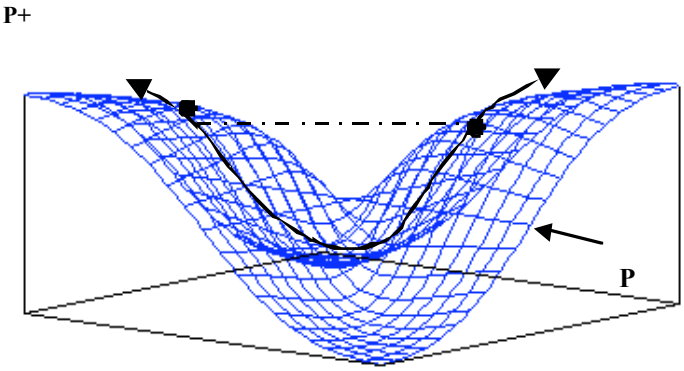

Figure 5. The illustration of the crossover-induced interpolation in P (the original structure space) and direct interpolation in enlarged space P+. The enlarged space P+ represents the complete three-dimensional space and the set P represents the surface in it. The solid line represents an interpolation in P (the mapping of the line segment in Figure 4 in genotypic space), whilst the dotted line represents an interpolation in P+

Because a genetic crossover has been constructed to produce genetic combination, its operations in genetic space can be easily understood, interpreted and described. But in design structure space (space of phenotypes) these are much more difficult tasks. Even the problem of expressing genetic crossover completely in terms of phenotypic space becomes a formidable one. The only exception is an extreme case when the genetic description becomes homomorphic with the phenotypic one. This case is illustrated in Figure 6, where a simple shape grammar is used to build a shape by adding elementary cells to one of 4 positions adjoining the currently labeled cell.

Ultimately, because we want to extend our structure space, we want to generalize our genetic representation by relaxing some of the constraints that are built into the genome. We will do that partial relaxation by replacing the original specialized "rigid" genetic representation, capable of representing only the designs from the fixed class, with a non-specialized representation that is homomorphic to the structure space representation. Once this is done, we will follow the above-described plan: derive the expression for genetic crossover in this representation, re-represent this expression first in an equivalent form and then construct various generalizations of it.

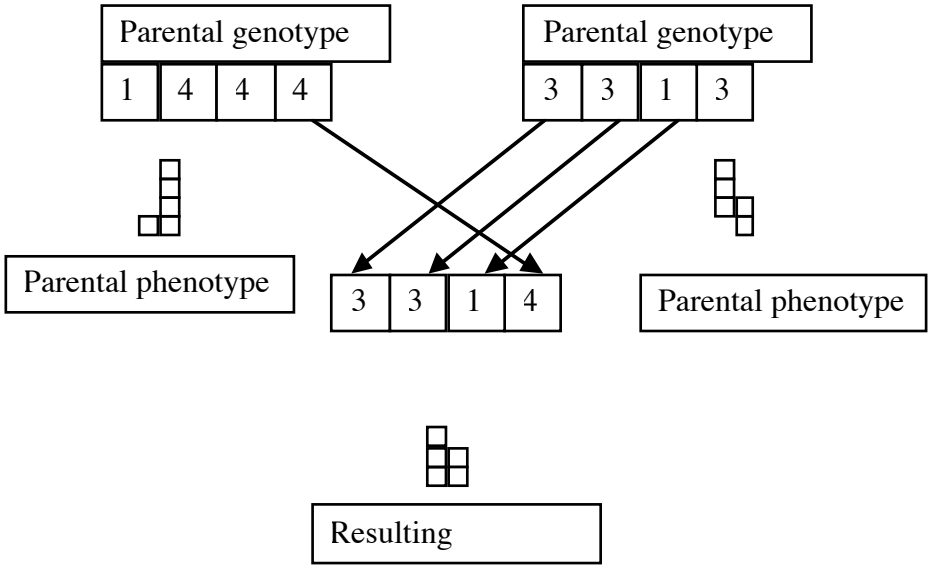

Figure 6. The case when both genetic and structure spaces are homomorphic and genetic crossover can be easily expressed in structure space terms.

4. GENETIC CROSSOVER AS A PARTICULAR CASE OF INTERPOLATION

As an example, in this paper we consider phenotypes (design structures) which are 2-d or 3-d shapes or objects. These designs are represented using an F-representation (Pashko et al., 1995) as real valued functions $F(\mathbf{x})$ such that $F(\mathbf{x})>0$ is inside the object, $F(\mathbf{x})=0$ is on its boundary and $F(\mathbf{x}) <0$ is outside of the object. Here $\mathbf{x}$ is a 2-d or 3-d vector with a defined feasible bounded region D: $\mathbf{x} \in D$. Assume that our design structures are represented using these F-representations and let us try to express genetic crossover in terms of this representation. Since the crossover yields a combination of two input designs, both designs here are represented as functions and a well known combination operation in space of functions is interpolation, it is natural to try to attempt to construct such expressions for genetic crossover as a function interpolation. Let us consider two design structures $F_1(\mathbf{x})$ and $F_2(\mathbf{x})$ and the transformation:

$$F_c(t,\mathbf{x}')= t\, v(\mathbf{x}')\, F_1(\mathbf{x}') + (1-t)\, w(\mathbf{x}')\, F_2(\mathbf{x}'),\ \mathbf{x}'(t,\mathbf{x}): D \to D, \qquad (1)$$

where t is a scalar which ranges from 0 to 1, $v(\mathbf{x})$ and $w(\mathbf{x})$ are non-negative scalar functions (called modulating functions) of $\mathbf{x}$, and $\mathbf{x}'(t,\mathbf{x})$ is a coordinate transformation which determines a homomorphism from (D,t) to D for any $t \in [0,1]$ and depends continuously on t. Usually the mapping $\mathbf{x}'(t,\mathbf{x})$ is chosen to provide a correspondence between the positions $i_j(t)$ of the similar features (points, line-segments, etc.) in the two shapes: $\mathbf{x}'(t,i_j(0))=i_j(t)$. The mapping $\mathbf{x}'(t,\mathbf{x})$ can be

constructed using the algorithm proposed by Fujimura and Makarov (1997). The only condition which is necessary for this formula to define an interpolation between $F_1(\mathbf{x})$ and $F_2(\mathbf{x})$ (that is, $F_c(0,\mathbf{x})=F_1(\mathbf{x})$ and $F_c(1,\mathbf{x'(1,x)})=F_2(\mathbf{x})$) is the positivity of functions $v(\mathbf{x})$ and $w(\mathbf{x})$ for all $\mathbf{x} \in D$. Hence, the goal is to find such positive functions $v^c(\mathbf{x})$ and $w^c(\mathbf{x})$ (we shall call them crossover modulating functions). Such functions must satisfy the requirement that the interpolation path $F_c(t,\mathbf{x})$ fits the mapping of the genetic interpolation $\mathbf{g_c}(t)$ induced by a crossover in genotypic space onto phenotypic space. With such functions, the GA search can be formulated completely in terms of phenotypic space without any explicit references to genotypic space (except implicit information built into functions $v^c(\mathbf{x})$ and $w^c(\mathbf{x})$). Whether or not this can be done (that is, whether or not $v^c(\mathbf{x})$ and $w^c(\mathbf{x})$ exist) needs to be investigated for each particular system. Note that for this problem phenotype interpolation turns out to be the problem of interpolation of real functions.

Now, that we have re-interpreted GA crossover as an interpolation operation in both genotypic and phenotypic spaces, we are in a position to produce a generalization of GA crossover which yields exploration, that is, enlargement of the phenotypic space, which is to be subsequently searched. In the next section we will do that and establish when this generalization will be productive, that is, when it could yield this enlargement.

4.1 Construction of generalized crossover using a general interpolation

Since we have already derived an equivalent formulation of genetic crossover as interpolation, equation (1), using the crossover modulating functions $v^c(\mathbf{x})$ and $w^c(\mathbf{x})$, we can now use the same formula with some other modulating functions that are more general than the crossover modulating functions. Essentially, this means that instead of choosing between various intermediate designs that have a fixed, given genetic structure (and therefore belong to a fixed given class of structures), we choose between intermediate designs that share only the ability to be represented (described) using the same framework. This is clearly a much weaker requirement.

4.2 Further generalization of crossover by replacing interpolation with extrapolation

The next obvious step in the generalization of genetic crossover is an attempt to extend the process of building combinations between designs, by starting from one design then building combinations that gradually transform it into the second design and allowing this process to continue beyond this second design. Mathematically, this means the replacement of interpolation with extrapolation. Intuitively extrapolation (if restricted to the states which are close to the parental points) seems to keep only marginally less information from these points but is capable of going outside the current space. Hence, we can further increase the potential exploratory power of our generalized genetic crossover in this way by completely replacing the crossover operation with the interpolating formula, equation (1), where the original

range of t values [0,1,…,n], is extended, for example, over [-k+1,-k,…,-1,0,1,...,n,n+1,…] to encompass extrapolation. Geometrically this case corresponds to the one when the extensions of the crossover-based interpolating path between two parental points on the surface **P** leave this surface, the heavy line in Figure 5.

5. CHOOSING THE FORM OF PHENOTYPIC INTERPOLATION

When we constructed the computational operations which could form the basis of our algorithm, we assumed that the non-crossover based interpolating modulating functions v(**x**) and w(**x**) in our example are somehow chosen. Their choice is critically important for the algorithm to be productive. Three strategies can be employed for finding such "good" interpolating operators. In the first approach ("blind extension"), some general parametric forms of these operators are chosen a priori, ie v(e,**x**) and w(e,**x**), where e is the vector of parameters e ∈ E within the feasible range E. Then when a combination is constructed additional search is carried out with respect to different values of e.

The second approach to the choice of interpolating formula (that is, modulating functions in our example) is based on the following strategy. First, the set of properties required of the states in an enlarged space in order to be valuable are established. Which properties should be preserved from the original space in the enlarged space need to be decided upon. Second, an interpolating operator with this set of properties is constructed. Various shape transformation algorithms which preserve different properties of the end point designs have been invented which assist in this process (eg Beier and Neely, 1992, Wolberg, 1990). This process is still largely experimental.

In the third approach the same set of properties required from the states in the enlarged design space are used as a "viability" test that is applied to each new design generated by generalized crossover. Only those new points which pass it are considered as viable combinations and are subjected to further processing. Of course, this is a search where there is no guarantee that such points can be found or even exist.

6. EXAMPLE

As an example, we consider the problem of designing a cross-section of a beam. The structure space consists of cross-sections of the pre-defined shape with a fixed area, Figure 7. This shape is determined by 4 parameters (the width and height of the top and bottom rectangular flanges and the width and height of the middle rectangular web) with the same range [1,100]. The area of the cross-section is fixed and is set equal to 16. The fulfillment of this area constraint is guaranteed by scaling the

shape. The problem has a two-component behaviour (fitness) function **F** which consists of the moment of inertia ,**I,** and the section modulus, **Z**.

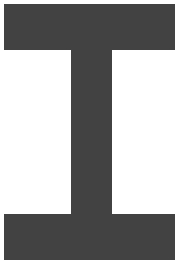

Figure 7 The cross-section template which defines the original structure space.

The standard genetic algorithm found the following Pareto-optimal designs in the original structure space, Figure 8. Because they all have the same genetic structure, they all have the same characteristic form.

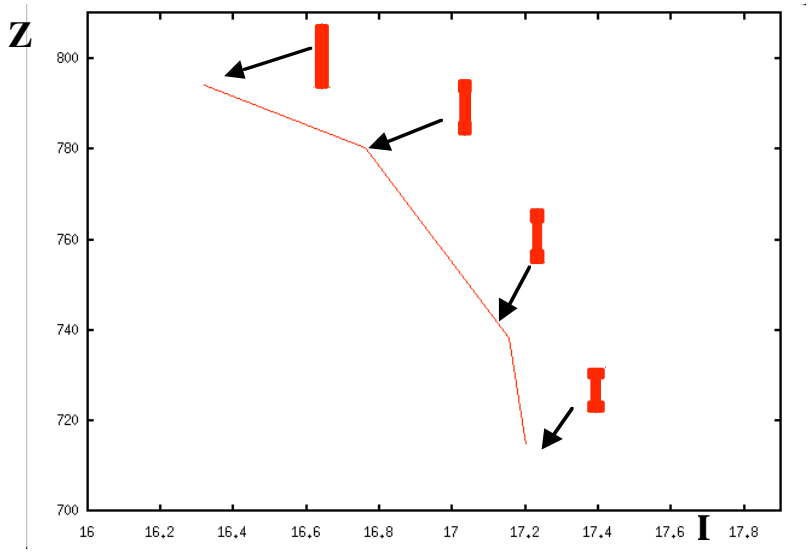

Figure 8 The Pareto set for the initial structure space with the corresponding shapes

Then these designs from the Pareto front are re-represented in F-representations, as real functions F(x), and their combinations produced using two types of interpolations – "linear" interpolation, when both modulating functions are always equal 1 in equation (1), and non-linear interpolation, when these modulating functions are chosen from a fixed class of (trigonometric) functions and "blind" search over modulating functions from this class is performed each time the combination is generated. The results are shown in Figures 9 and 10. It is quite clear

that even the use of the linear interpolation leads to the structure space being extended, and that non-linear interpolation leads somehow to a more powerful exploration process which becomes extremely potent when extrapolation is switched on, Figure 11.

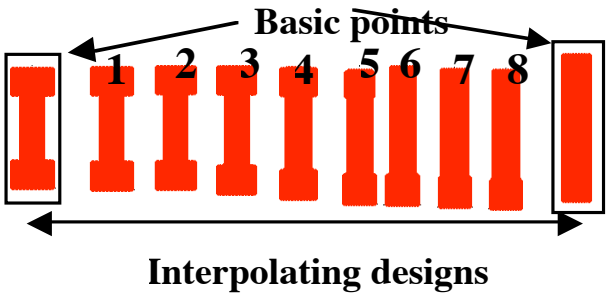

Figure 9. Combinations produced by the "linear" structure-structure interpolation

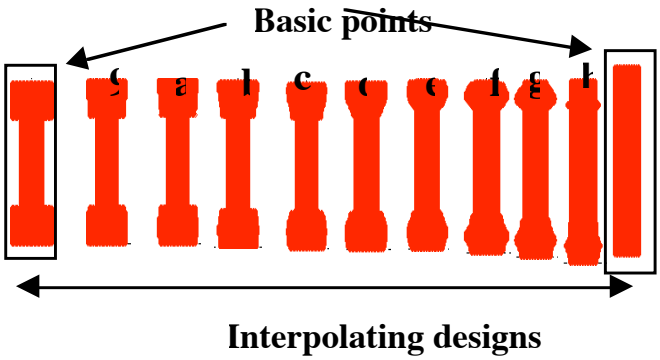

Figure 10. Combinations produced by the "non-linear" structure-structure interpolation

Figure 11. Unexpected combination produced by structure-structure extrapolation

Designs generated from both the linear and non-linear interpolations exhibit characteristics which are not inherited from their parents, as would be the case with genetic crossover. These novel characteristics are a demonstration of the capacity of this process to expand the state space of possible designs.

We can plot these novel designs on the Pareto-optimal front and we note that some of them dominate previous Pareto-optimal designs, Figure 12. From this example, we can see that this generalized combination produces both diversity and the potential to obtain designs which perform better than those produced by genetic combination alone.

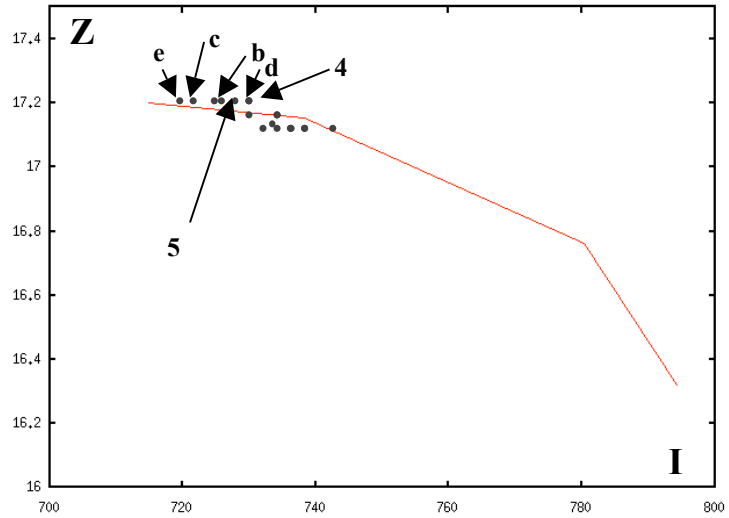

Figure 12. Improved design produced by generalized combination

7. DISCUSSION

One of the well-established notions related to creative designing processes is that an important means of characterising them is to determine whether they have the capacity to expand the state space of possible designs - exploration (Gero, 1994). Of the three state spaces used to describe designs (function space, behaviour space and structure space), only the structure space has the capacity to directly produce novel designs as it is expanded, although the other spaces can also be expanded and hence produce either novel interpretations or indirectly force the expansion of the structure space (Gero and Kazakov, 1998). There are two classes of expansion or modification operators applicable here. The first class contains those operators which rely largely on external knowledge which is applied to the values of the existing space and as a consequence expand it. The second class contains those operators which make use of emergent features in the design space and use those to expand the design space. The extrapolation operator described here belongs to the first class.

One of the difficulties in deriving operators that are capable of expanding a design space is the need for the operators to have the capacity to add variables to the range

of variables used to describe the space originally. There have been a number of approaches adopted previously. These include splitting a single variable into two variables (Aelion et al., 1992), importing variables from other design spaces using combination or analogy, and emerging new features and reverse engineering new variables to describe them. The approach adopted here distinguishes itself from these and other approaches in that it opens a range of possible new variables. It does so by recasting the structure space from being restricted to the surface defined by the variables and the original processes of genetic algorithms to being a larger potential space. Which parts of that larger space are used is dependent on the interpolation/extrapolation functions involved and the values of the variables used when the functions are applied. The range of applicable functions may well be unlimited. Each function potentially produces a different trajectory outside the original surface and each trajectory represents potentially different structure variables which are required to describe the resulting design.

As can be seen from the example the resulting designs are unpredictable in the sense that they are unexpected given only knowledge of the original designs and of the interpolation/extrapolation functions. In this sense the process matches well the meaning of exploration both in the technical sense used in this paper and in the natural language sense. The designs produced by the system demonstrate both the novelty and unexpectedness of what can be generated.

This paper can be seen as a generalization of a number of related approaches (see for example, Graf and Banzhaf, 1995) where cross-image interpolation has been used to replace a GA's crossover operator. Our results are more general because we use a more general type of phenotype-phenotype (cross-object and not cross-image) interpolation which can be reduced to cross-image interpolation in a particular case only.

Acknowledgments

This work is supported by a grant from the Australian Research Council. Computing resources are provided by the Key Centre of Design Computing.

References

Aelion V, Cagan J and Powers G, 1992, "Input Variable Expansion - An Algorithmic Design Generation Technique", *Research in Engineering Design,* 4 101-113

Beier T and Neely S, 1992, Feature-based image metamorphosis, *Computer Graphics* (Proc. of SIGGRAPH), 35-52

Fujimura K and Makarov M, 1997, "Homotopic shape deformation", *International Conference on Shape Modeling and Applications,* Aizu-Wakamatsu, 215-225

Gero J S, 1990, "Design prototypes: a knowledge representation schema for design", *AI Magazine,* 11 (4) 26-36

Gero J S, 1994, "Computational models of creative design processes", *in* T. Dartnall (ed.), *Artificial Intelligence and Creativity,* (Kluwer, Dordrecht), 269-281

Graf J and Banzhaf W, 1995, "Interactive evolution in civil engineering", *in* J. S. Gero, M.L.Maher and F. Sudweeks (eds), *Preprints Computational Models of Creative Design,* (Key Centre of Design Computing, University of Sydney, Sydney, Australia), 303-316

Pasko A A, Adzhiev V D, Sourin A I and Savchenko V V, 1995, "Function representation in geometric modeling: concepts, implementation and applications", *The Visual Computer*, 11 (8) 429-446

Wolberg G, 1990, *Digital Image Warping*, (IEEE Computer Society Press)