Knowledge-Based Repair for Knowledge-Lean Techniques in Non-Routine Design

Michael A. Rosenman and Nicholas Preema

Abstract— This paper presents a methodology for producing good design solutions more efficiently. The methodology is based on augmenting a conventional evolutionary design approach with a method for improving suboptimal design solutions with a domain-specific knowledge-rich approach. This approach is based conceptually on the practice of plastic surgery, i.e. making minor adjustments to an entity, based on some desired qualities, i.e. specified fitness function. Additionally, the modifications made to the phenotype may require the re-engineering of the genotype to accord with the modified phenotype if the entity is to be used further in evolutionary operations. A method for genotype re-engineering is proposed in the domain of cellular growth generation.

Keywords— evolutionary design, genetic algorithms, plastic surgery, genetic re-engineering.

I. Introduction

NON-ROUTINE design tasks are characterized by the lack of knowledge available for their immediate solution [3]. Thus knowledge-lean approaches, such as evolutionary computation methods, are well suited to the task of non-routine design [14] [1], [2], [7]. Evolutionary computation methods are able to arrive at reasonable solutions fairly quickly to begin with but then need many generations to make subsequent small improvements, [3], [11]. A great deal of effort can be expended to make a small (but maybe critical) improvement. In general, there is no guarantee that such an improvement will be found. In addition, in non-routine design, it is not always possible to perfectly specify the fitness function such that optimal solutions will be found since the design task is not well-known. This paper argues that, even in such conditions, it is possible to obtain reasonable solutions within the bounds given and then, using these resulting solutions as a guide, make improvements to obtain better solutions. For example, Fig. 1 (a) shows a configuration of 12 cells that may arise after a number of generations.

To produce an improvement such that the protrusion is removed and the indentation is filled (leading to a square in this case), Fig. 1 (b), may take a great deal effort from the evolutionary computation process. Where the number of cells is large, (>=100), the ability of such a system to remove such irregularities will be inefficient. While we can see that removing the protrusion and filling in the indentation would lead to a good solution, the evolutionary system based on random genetic operations (crossover and mutation) on the genotype may not be able to produce the required solution within a feasible timescale.

(a) an almost 'perfect' solution (b) an improved solution

Fig 1. Improvement of a design solution.

Plastic surgery is a practice whereby features of an entity (generally a human) are altered to improve the appearance of that entity. This may be for cosmetic purpose or for more serious reasons. In all cases, the effect is on the entity itself, i.e. the phenotype, and there is no change to the genotype (DNA). Since evaluation is done on the phenotype, any improvement to the phenotype gives the entity a better chance of survival or attaining its goal, e.g. attaining self esteem, attracting other entities, etc. Plastic surgery is generally done to correct minor features; an entity has been generated in some way, but is defective in some features and minor corrections are made (to the phenotype) to improve it. It can be seen that specialized knowledge is required to modify the phenotype. Different domains require different knowledge. The phenotype and the defects must be recognized and the means for modification determined and implemented.

As in the human example, any modification to the phenotype (design solution) is not transmitted to the genotype. Any 'children' may carry the defective genes and reproduce the same defects. However, in design, if the modified design solution is the final solution required, and no more processing is to take place, then this does not matter as the genotype was just the means to the end and is no longer of any interest. However, if the modified design solution is only a part solution and is required to take part in further evaluation, a problem exists since all evolutionary operations are carried out on the genotype. In that case, its genotype must be re-engineered to match the modified phenotype. This difficult to implement since, in general, there is no known connection between the phenotype and the genes in the genotype.

Manuscript received February 14, 2008.

M.A. Rosenman is with the Key Centre of Design Computing and Cognition, The University of Sydney, Australia (phone: +61 (2) 9351; fax: 303-555-5555; e-mail: mike@arch.usyd.edu.au).

N. Preema, is also with the Key Centre of Design Computing and Cognition, The University of Sydney, Australia.

Although there are some examples of genetic engineering [10], in general, there is no universal knowledge on how to modify the genotype to produce required characteristics of the phenotype.

A conventional evolutionary method may be used, in situations where the form required is not known a priori, to produce possible solutions which are reasonably good but need some improvement. Once the solutions are produced, any suitable solutions can be improved by making small modifications to the phenotype. If the resulting solution is required to take part in further evolutionary operations, genetic-re-engineering is required as described above. This paper will put forward a method for re-engineering the genotype in a given design representation.

II. Evolutionary Design

While knowledge-lean methods, such as evolutionary design, are good for discovering possible reasonable solutions where little knowledge is known a priori regarding the form of the solution, they are generally computationally expensive and may not be able to make the necessary improvements in a reasonable time with reasonable resources. Additionally, in an environment where there exists little a priori knowledge, it is not always possible to perfectly specify the requirements, i.e. formulate a 'perfect' fitness function. On the other hand, while knowledge-rich approaches can solve problems where the problem is well defined and the knowledge and methods required are also known, they operate in specific problem areas and, even within those areas, have little capacity for producing innovative solutions.

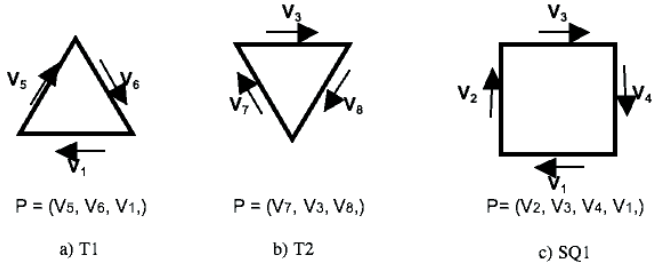
This paper presents an approach combines knowledge-lean and knowledge-rich approaches to increase the efficiency of producing good design solutions in a non-routine design problem environment. The conventional evolutionary computation approach generates reasonably good solutions within given initial specifications and the proposed plastic surgery makes small modifications as necessary based on local knowledge of the problem once the solutions are evident.

A. A Design Representation

In general, design can be defined as the derivation of structure (form) that will satisfy a given set of requirements [18]. In its simplest mode, the construction of form can be thought of as the set of decisions for locating a set of cells of substances, where a substance may be physical (composed of a physical material) or virtual (e.g. composed of graphic entities or pixels). The construction of a spatial entity may be considered as the allocation of a physical substance composed of a 'space' material, i.e. a number of space cells. So, in summary, design can be see as the generation of form which can be produced by cellular growth. In an evolutionary design approach, a gene selects a module of substance and allocates it to some location. In the approach of Rosenman [14], [15], a gene locates a module of substance relative to another module. A gene, GN, is thus (M_1, M_2, L_{12}) where M_1 and M_2 are two modules of some substance and L_{12} is the operator for locating module M_2 relative to module M_1. A module, M_i, may be a single unit cell or a set of unit cells already grouped and, in general, M_1 and M_2 need not be composed of the same substance. Nor, in general, does a grouping of units necessarily need be constructed of units of the same substance. The instructions for a complete design solution, i.e. a genotype, G, is a sequence of genes where G = (GN_1, … GN_m) and GN_i, i = 1,m is a gene.

In the approach, based on the joining of polygons representing units of space, the allocation operation is founded on the joining of polygons through their free edges represented as vectors. Fig. 2 shows the representation of two triangles, T1 and T2, and a square, SQ1, as closed vector loops. The vector V1 is a vector of length 1 unit and angle 180^o, the vector V2 is a vector of length 1 unit and angle 90^o, V5 is a vector of length 1 unit and angle 60^o, etc. The phenotype, P, of each polygon is given as the loop of vectors. Since this is a loop, the start point of the loop is immaterial, although in the examples it is given as the lowest-leftmost point.

Polygons may be joined by conjoining counteractive (equal and opposite) vectors [13].



When a number of squares are joined randomly the resulting shapes (polyminos) are not likely to show much regularity, especially if the number of squares is large. A shape with many protrusions and indentations will have many changes of direction on its boundary. Fig. 4 shows 40 random generations of 16-unit polyminos.

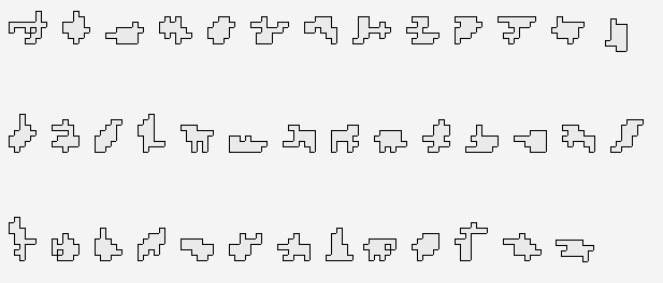


Figure 3. 40 random generations of 16-unit polyminos

For the example in Fig. 3, using 16 cells, the evolutionary program will evolve the shapes to find L- and T-shapes as well as rectangles and the 4 x 4 square. However, when using a small-scale cell to allow for small discriminations in the dimensions, the number of total cells will be very large. For example if the scale of the cells in Fig. 3 were reduced by a factor of 10, allowing for increments in length of say 10cms

rather than 1m, the total number of cells would be 1600. For a large number of cells, an evolutionary process, will, after a number of generations, give some indication of possible satisfactory shapes but will, usually, not be able to perfectly smooth out all the protrusions and indentations. One reason for this may be that the fitness function is not perfectly specified since such knowledge may be beyond the current knowledge of the design situation. For example, a fitness function based on minimizing the perimeter to area ratio will find that, with a large number of cells, the number of cells at the perimeter is small compared to the number of interior cells which are fairly well compacted. Thus most of the solutions will show a fairly high score for that fitness function. The process will not be able to make any significant improvement in any reasonable time. Fig. 4 shows a shape of 85 cells after evolution over a number of generations. The perimeter to area ratio shows a fitness of 85.8% compared to the ideal of a square of area 85 (perimeter = 36.88).

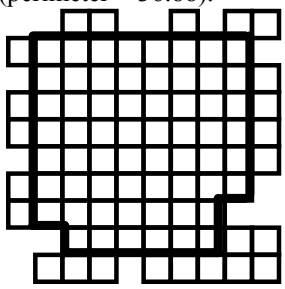


Figure 4. A polymino of 85 units

The reason for the high fitness, even though the perimeter is not very smooth, is that the central part of the shape (outlined in bold), which contains the majority of the units, is quite compact. Continuing the evolutionary process usually leads to convergence on one or other of the 'better' solutions to date before any major improvement is attained.

III. Plastic Surgery in Evolutionary Design

In an evolutionary system, selection acts with respect to the phenotype. Those members whose phenotypes are judged to be well-suited to their environment will have a better chance of survival and of propagating their genes [7]. Thus any improvement in the phenotype, regardless of any change in the genotype, will improve that member's chance of survival and propagation. Of course, this improvement will not be transmitted to the member's descendants. In a design domain the fitness of the design is what counts, how it got to be that way is secondary.

The Merriam-Webster dictionary [9] states that plastic surgery is:

": surgery concerned with the repair, restoration, or improvement of lost, injured, defective, or misshapen body parts"

Plastic surgery is aimed at improving the organism's survival in its environment (whatever survival may mean).

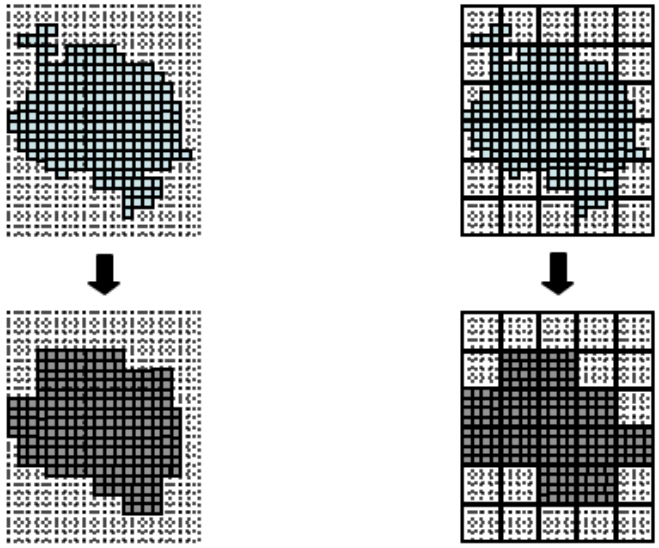
Plastic surgery is proposed here as a solution to improving a phenotype (design solution) generated through an evolutionary computation method. It is proposed as a general concept where the issues are as follows:

- design solutions are produced which are reasonable but could be improved by relatively small modifications. This requires particular domain knowledge.
- if the design solutions are the end product of the process then the modified phenotype is the final solution and it is no longer necessary to consider the genotype. However, if the 'solution' is part of an on-going process, e.g. a component of a hierarchical composition, it may be necessary to re-engineer the genotype to match it to the new phenotype so as to enable it to be operated on by the genetic operators.
- the modifications should be limited to relatively small remedial improvements. It is not meant to carry out major reconstructions of the phenotype as this leads to too large a departure from the solutions found.

While an example in the domain of the generation of smooth polygons will be used to demonstrate the concepts, this paper suggests that the general principles of plastic surgery and genetic re-engineering could be applied to all domains since all design is a function of locating elements in a certain configuration.

IV. Methodology

The implementation of plastic surgery consists of several transformation functions. There exist various smoothing algorithms mainly in image processing, where they are used to produce smoothed surfaces from polygonal or noisy surfaces [5], [6], [19]. Algorithms such as Potrace [12] transform bitmap images into vector graphics. Another process uses sampling for anti-aliassing in ray tracing [17].



(a) smoothing maintaining sampling

(b) smoothing using smaller unit size

Figure 5. Comparing plastic surgery to sampling

Sampling works by overlaying a grid of larger cells on the form. Each cell is analyzed to determine what percentage of the cell is occupied. Cells with 50% or more occupation would be filled in completely, while those with less than 50% occupation would be left empty. This process is shown in Fig. 5 where it is compared to the process which allows smaller increments of discrimination. The small-increment method is closer to the philosophy of making minor repairs rather than large-scale modifications and results in shapes closer to the original shapes than the sampling method which results in 'major reconstructions'.

Modifications can be carried out to various levels of refinement, i.e. with respect to the number of units to be treated. Fig. 6 shows the various examples (defects) which may require modification. These include protrusions, indentations and corners, ranging from one unit to several units. The number of units in each direction may depend on the scale, i.e. the total number of units in a shape. While Fig. 6 shows defects on one edge or corner only, the defects may occur on any of the four edge or corner directions (for polymino shapes).

Fig. 7 shows the rules for plastic surgery, i.e. modifying the phenotype (shape) according to the type of defect (protrusion, indentation or corner) and the number of units to be rectified in the two directions. Again, it should be noted that the defect may occur in any direction so that the depth and width of a defect are local to the particular direction.

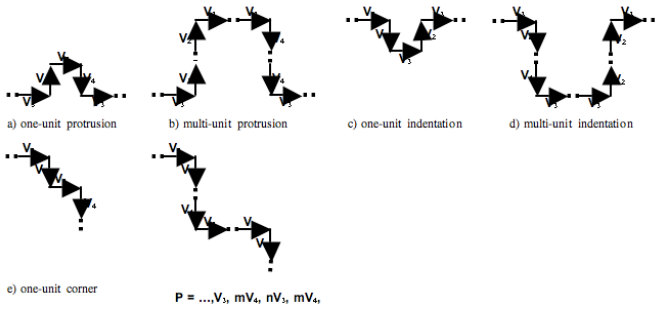
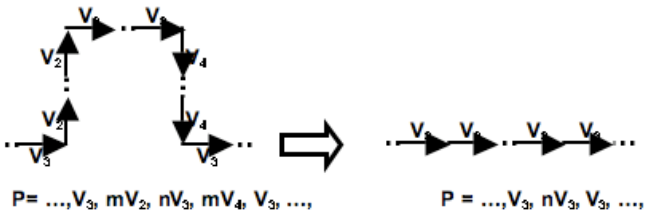


Figure 6. Cases for modification

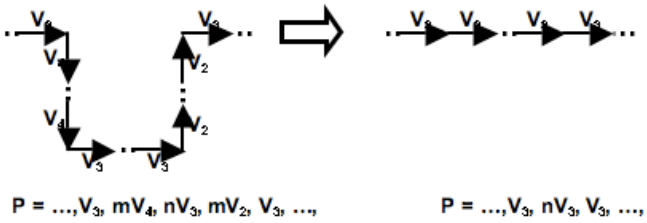
The level of refinement is set by setting the depth and width, in terms of number of units, for the plastic surgery to take effect. The degree of refinement and the order of implementation of the operations will determine the final result. Different parameters and sequences will produce different results. In the physical world it is not possible to try several alternatives, whereas in a computational process it is possible to try alternatives and select among them depending on the result. Fig. 8 shows two different sequences of operations on a shape of 50 units based on the following operations or rules:

Rule 1: Defect = protrusion max depth = n
max width = 1
Rule 2: Defect = indentation max depth = 1
max width = 3
Rule 3: Defect = corner max depth = 1
max width = 1

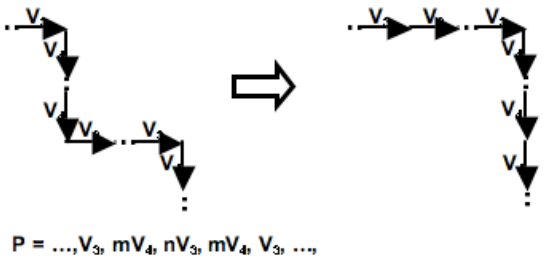
Rule 1 states that all protrusions of width 1 unit, no matter their depth, are to be deleted.



a) removal of a multi-unit protrusion



b) 'filling-in' of a multi-unit indentation



c) 'filling-in' of a multi-unit corner

Figure 7. Rules for plastic surgery of defects

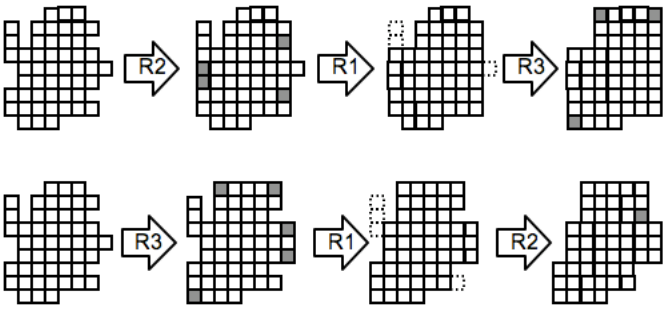


Figure 8. Two different sequences for plastic surgery

The shaded and dotted units show the units added or trimmed. The first solution has grown from 50 units to 54 units whereas the second solution has increased to 52 units. The size of the resulting solution depends on the number of units trimmed or filled. Since the number (size of the element) may be critical, some constraints may need be applied regarding the number of units trimmed or added or the number of elements trimmed may need to be balanced by the number

of elements added (and vice versa). In a very large number of units, the number of units adjusted may not makes a significant change to the size of the shape since the number of units on the perimeter is small compared to the total number of units.

A method for recognizing which shapes are suitable for plastic surgery is based on the measure of fitness of the shape as well as on a measure of the number of defective units with respect to the shape's perimeter.

V. Implementation Example

An example in the domain of room designs was implemented. Rooms need not necessarily have rectangular shapes nor do they necessarily have to have 'smooth' walls. They may have recesses but generally these need to be large enough to accommodate furniture such as bookshelves etc. So, in general, small protrusions and recesses in the perimeter are not acceptable. The aim is to generate shapes for a room of 18 m^2. A variation of 300 mm in each dimension was set to allow for a wide range of possible dimensions. This results in the arrangement of 200 square units of 300 mm x 300 mm.

The fitness functions used were those used in Rosenman [14], [15]. A function that tends to smooth the perimeter is that of minimizing the perimeter. The minimum perimeter of a polymino shape is ideally a square. Using this fitness function will tend to make shapes more compact, thus reducing the length of the perimeter. The aim of room design is not to necessarily produce square or rectangular shapes but to use the fitness function to drive the evolutionary process towards such shapes, generating other suitable shapes in the process. Another measure of the smoothness of the perimeter is that of minimizing the number of corners. The minimum number of corners of a polymino shape is 4. Obviously a square has both the minimum area and the minimum number of corners. This function has a tendency to prefer L-shapes over T-shape. Both these shapes will have the same perimeter to area ratio but the L-shape has six corners compared to eight for the T-shape.

For the first function, minimizing the perimeter to area function, the fitness is given by:

$f1 = (MaxP - P / MaxP - MinP) \times 100$ --------- (1)

where

f1 = fitness function wrt minimum perimeter to area

MaxP = maximum possible perimeter for a shape of n units

P = perimeter of generated shape

MinP = (ideal) minimum perimeter of a shape of n units

and

Min P $= 4\sqrt{n}$ (ideal square)

MaxP $= 2n + 2$ (e.g. shape of 1 unit width and n units length)

where

n = number of units

For the second function, that of minimizing the number of corners, the fitness is given by:

$f2 = (Max\ C - C / MaxC - 4) \times 100$ ---------- (2)

where

MaxC = maximum possible number of corners for a shape of n units

C = number of corners of generated shape

and

MaxC $= 2n$ (e.g. fully stepped shape)

Both functions use a ratio of the range of possible values to determine the normalized percentage fitness of the shape. The total fitness is given as:

$TF = (f1 + f2) / 2$ ---------- (3)

Different weightings could be used for each fitness function to influence the shape towards one or the other but for this example a simple weighting of 1 for each has been used for simplicity.

A C++ program for Windows was written to generate and evolve a population of polymino shapes using a genetic algorithm based on cell addition using the edge vector representation discussed previously and then to perform plastic surgery. The inputs to the generation and evolution are: the number of units, the number of members of the population and the maximum number of generations to be run. The genetic algorithm may terminate before the maximum number of generations is reached if it converges or remains stable. A run converges if the average fitness is within 5% of the best fitness and remains stable if there is no significant change in the best solution or average fitness over a specified number of generations. Simple one-point crossover was used with the best of the two populations (parent and child) kept to preserve the best solution. The remaining members of the new generation are selected using the roulette wheel method. The inputs to the plastic surgery are the width and length of the three repair cases (protrusion, indentation and corner) specifying the scale of the repair.

The program was run several times with the following parameters:

No. of units 200
Population 40
Max. no. of generations 60
Max. depth 1
Max width 3

Results were similar over a number of runs. Fig. 9 shows the results of one of these runs. Fig. 9. shows a typical growth in fitness over the 60 generations using the conventional evolutionary process. The average fitness of the population is 72.6%. As can be seen from the graph, the population has arrived at a fairly stable state and it could take a very large number of additional generations to produce any improvement

(if any is possible). After the application of plastic surgery, the average fitness jumps to 89.3%. This is a 23% improvement.

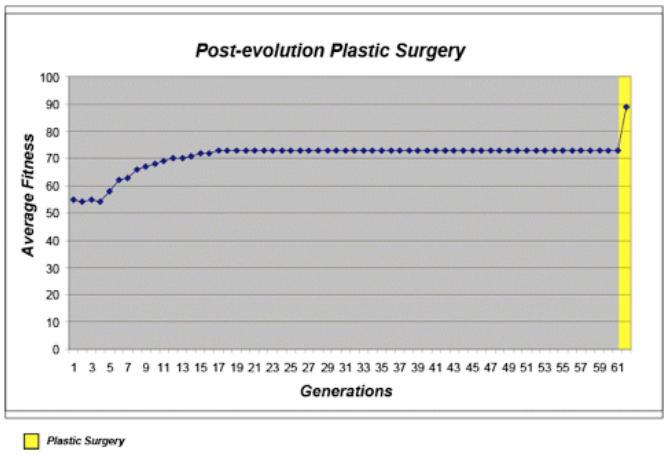


Figure 9. Effect of plastic surgery after the evolutionary process

Fig. 10. shows three of the shapes subjected to plastic surgery. It can be seen that these three members, previous to the plastic surgery, had a high fitness (95.6 to 96.6) even though their shapes are not all that good. The first shape is better than the other two but still has some small changes in direction in the upper left-hand part.

The relatively high fitness values are due to the fitness function used which, in part measures the compactness of the shape. Since a large proportion of the shapes is indeed compact, the fitness values are high and there is little pressure to improve them. In the previous work (Rosenman 1996a, b) where only relatively small number of units were used (maximum 25) this problem did not exist. It can be seen that while the application of the plastic surgery has improved the fitness values, its main contribution is in producing better shapes, i.e. shapes with fewer small protrusions, etc. No method was used to ensure that the size of the shape (room) remained the same and the first shape has increased to 208 units, the second to 206 and the third to 207, an increase of less than 5% in all cases. Note that while none of the shapes shown are rectangles, nevertheless they could be suitable as rooms in certain instances.

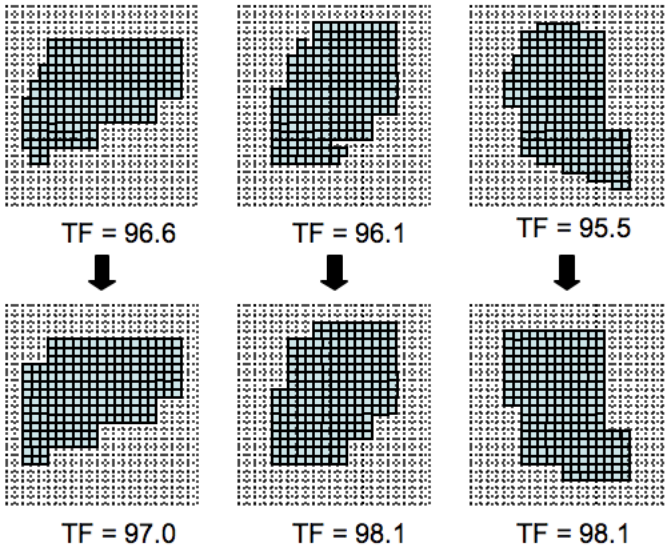


Figure 10. members from the run before and after plastic surgery

VI. Re-Engineering the Genotype

A phenotype may be generated in many ways, i.e. the same phenotype may have different genotypes. Fig. 11 shows just three examples of the same shape generated by adding the cells in different sequences. The bold lines are the edges joined. The genotype description shows just the edge joining part of the genes for simplicity as the module added is the same unit square cell.

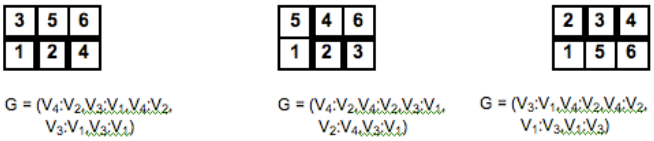


Figure 11. Three different ways of generating the same shape

A shape can be ordered according to different traversal strategies. Fig. 12 shows three such strategies. The genotype shown below each strategy is for the general case (n units) calculated according to the particular traversal carried out.

All three strategies are based on starting at the left-hand bottom corner. The first strategy is based on traversing the shape left-to-right as far as possible, moving up one unit and then traversing right-to-left as far as possible, moving up one unit and repeating the process until all units have been traversed. The second strategy is based on a single direction traversal. That is traversing left-to-right as far as possible then moving up one unit from the starting point and repeating the process. The third strategy is based on a spiral traversal. Note that traversal could be carried out either horizontally or vertically. Other geometries, those not based on orthogonal axes, may need different strategies. The genotype is calculated by simply noting whether the traversal was carried out in a left-to-right, right-to-left, upward or downward direction, equating to a $V_4{:}V_2$, $V_2{:}V_4$, $V_3{:}V_1$ or $V_1{:}V_3$ edge joining. While this is not an exhaustive presentation of all possible strategies, it shows that it may be possible to construct a genotype given a phenotype where the phenotype has been

constructed through a sequential series of allocation of cellular units.

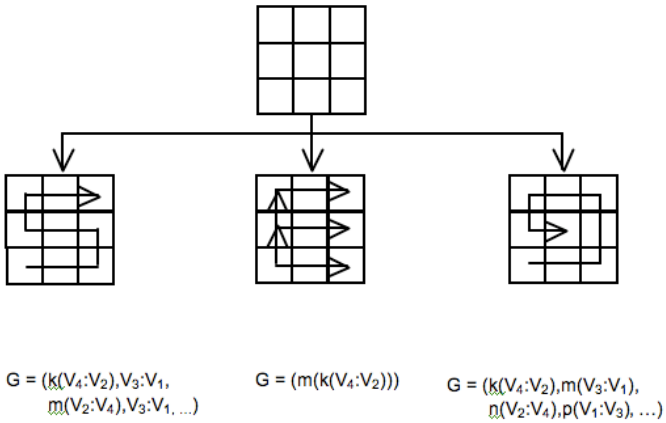


Figure 12. Three different strategies for generating a shape

VII. Conclusions

This paper has presented a method of generating form through cellular growth as a simplified model of design through the allocation of substance to satisfy a set of given requirements. It has argued that evolutionary design methods are suitable approaches for non-routine design generation since they are knowledge-lean and hence suitable for situations where there is little a priori knowledge available regarding any associations between the requirements and the form to be generated. A simple model of allocating substances through gene sequences was presented, where each gene carries the instruction for locating one module of substance relative to another module. An example using square cells was used for simplicity although the approach could be generalized to 3-D polyhedral shapes. However, it is argued that for complex objects with large number of cells, with fitness functions that may be imprecise, the solutions arrived after a reasonable effort may still need improvement.

The results of the implementation of the example show that plastic surgery is a useful method for efficiently improving design solutions where the evolutionary process has achieved stability. Plastic surgery is seen as a knowledge-based mutation of the form (phenotype). Though illustrated in the context of the 2D cellular formation of shapes and the smoothing of irregular perimeters, it is a general concept applicable to 3D forms and other applications. Other applications will use domain specific knowledge for their repair rules.

Future work will need to take into consideration the allocation of units of different substances and the repair of the whole. This will mean deciding not only what form needs to be repaired but what substance should be used.

References

[1] Bentley, P. J. (ed.), *Evolutionary Design by Computers*, Morgan Kaufman, San Francisco, CA, 1999.

[2] Bentley, P. J., "Natural design by computers", *Proc of the AAAI Symposium on Computational Synthesis*, Stanford University, Palo Alto, CA, (2003)..

[3] Coyne, R. D., Rosenman, M. A., Radford, A. D., and Balachandran, M. B. and Gero, J. S., *Knowledge Based Design*, Addison-Wesley, Reading, Mass., 1990.

[4] Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Reading, Mass., 1989.

[5] Hobby, J. D, Smoothing digitized contours, *Theoretical Foundations of Computer Graphics and CAD*, 1998, pp.777-793.

[6] Hoppe, H., "Progressive meshes in computer graphics", *SIGGRAPH'96 proceedings*, Aug 4-9, 1996, pp.99-108.

[7] Janssen, P., Frazer, J. and Ming-Xi, T,. "Evolutionary design systems and generative processes", *Applied Intelligence*, 2002, **16**:119-128.

[8] Koza, J. R., Jones, L. W., Keane, M. A., Streeter, M. W. and Al-Sakran, S. H, "Towards automated design of industrial-strength analog circuits by means of genetic programming", in *Genetic Programming Theory and Practice II*, U-M Oreilly, R. L. Riolo, G. Yu and W. Worzel (eds), Klewer Academic, Boston, 2004, Chapter 8, pp121-142.

[9] Merriam-Webster Online Dictionary, http://www.m-w.com/dictionary/Plastic+Surgery

[10] Old, R. W. and Primrose, S. B., *Principles of Gene Manipulation: An Introduction to Genetic Engineering (studies in Microbiology)*, Blackwell Science 5th ed., Oxford, UK, 1994.

[11] Parmee, I. C. and Denham, J., "The integration of adaptive search techniques with current engineering design practice", in *Proc.of Adaptive Computing in Engineering Design and Control '94*, University of Plymouth, Plymouth, 1994, pp1-13.

[12] Potrace, Transforming bitmaps into vector graphics, http://potrace.sourceforge.net , 2007.

[13] Rosenman, M. A,. "An edge vector representation for the construction of 2-dimensional shapes", *Environment and Planning B:Planning and Design*, 1995, **22**:191-212.

[14] Rosenman, M. A., "The generation of form using an evolutionary approach", in *Artificial Intelligence '96*, J. S. Gero and F. Sudweeks (eds), Kluwer Academic, Dordrecht, The Netherlands, 1996a, pp.643-662.

[15] Rosenman, M. A., "A growth model for form generation using a hierarchical evolutionary approach", *Microcomputers in Civil Engineering*, special issue on Evolutionary Systems in Design, 1996b, **11**(3):161-172.

[16] Rosenman, M. A., "A face vector representation for the construction of polyhedra", *Environment and Planning B: Planning and Design*, 1999, **26**:265-280.

[17] Rossignac, J. R. and Borrel, P., "Multi-resolution 3D approximations for rendering complex scenes", in *Geometric Modelling in Computer Graphics*, B. Falcidieno and T. L. Kunii (eds), Springer-Verlag, Genoa, Italy, 1992, pp455-465.

[18] Simon, H. A,. *The Sciences of the Artificial*, MIT Press, Cambridge, Mass., 1989.

[19] [Volino, P. and Magenat Thalman, T., "The SPHERIGON: a simple polygonal patch for smoothing quickly your polygonal meshes", *MIRAlab Copyright Information*, 1998, http://www.miralab.unige.ch/papers/50.pdf.