VISUAL DESIGN USING GENETIC PROGRAMMING

Süleyman Sevinç, Gözde Bal, Ender Bulgun
Department of Computer Engineering
Dokuz Eylul University
Izmir, Turkey
Suleyman.sevinc@deu.edu.tr,bgozde@hotmail.com,ender.bulgun@deu.edu.tr

Keywords: *Genetic Programming, Genetic Algorithms, Evolutionary Computing, Design.*

This paper presents a methodology to generate new designs by using evolutionary algorithms, that is, genetic programming (GP). A design session starts with a number of initial design patterns. New appropriate representations emerge as crossover and mutation operations are applied to these initial patterns. Emerging designs are assessed and classified based on fitness values. Common LISP is used to do processing and to simulate evolutionary processes.

I.INTRODUCTION

Designers used to generate new designs using traditional tools, such as pencil and marker. Introduction of computers has changed the process. The arrival of desktop publishing and graphic art software applications introduced a generation of designers to computer image manipulation and creation that had previously been manually executed. Computer graphic based design enabled designers to instantly see the effects of layout or typographic changes, and to simulate the effects of traditional media without requiring a lot of space [3].

Nowadays, computers are accepted as inevitable tools in the visual design. Many different tools and resources made available by use of computers trigger creativity and ease the creative process. Today, many designers use the computer while generating new design images. Designers can explore multiple ideas more quickly and efficiently than what was achieved by traditional methods such as hand rendering or paste up on paper.

It is common practice now that designers and engineers benefit from CAD (Computer-aided Design) tools in order to help design process. They move the designer through the creative process more efficiently. CAD tools are very good in translating a design idea into a visual form. But we feel the need to add more layers in order to emulate human creativity computationally.

In this paper, we exploit the principles of evolution to generate new design images by mimicking nature. For this we use Genetic Programming (PG) and Genetic Algorithm (GA) approaches. [2, 5] Basic operations of GP and GA (crossover and mutation operations) are specifically developed for this project in order to get more accurate and better designs.

In the GA approach, a design is considered as two dimensional images. Every pixel on the image has localization (x-coordinate, y-coordinate) and colour information. Chromosomes include the “genetic” information about image are represented as bit strings containing of pixel information and each has a fitness value. In order to generate new design patterns we take advantages of Darwinian principal of reproduction and survival of the fittest. [1, 7] In the GP approach, we use Common LISP programming language [9] in order to evolve new design images from the existing ones. Genetic information about designs is kept within the instructions of the programs which generate specific design patterns. With the help of LISP, computer programs can be manipulated, altered, evaluated or even created as data without extensive parsing or manipulation of binary machine code. Therefore, we are able to perform crossover and other genetic operations not on design patterns themselves but on the programs which contain the genetic information for design patterns.

Users could get involved in every step of the design process. When each generation is created, the user can select the appropriate ones by overwriting the calculated fitness val-

ues (subjective choice). The quality or "fitness" of these solutions is then determined. In some case, this can be done algorithmically; however, a human can judge subjectively. [6] After filtering, genetic operations (crossover, mutation, and elitism) are applied on the population. Thus, by assessing generations, it allows to get more relevant and efficient design patterns satisfying user requirements. In addition using one of the techniques presented below, further designs may be selected for evaluation. Determination of how many times the process will work is left up to the user until appropriate and effective design patterns are reached.

In Genetic Programming approach, a program produces a potentially large number of possible solutions. To reduce the search space, fundamental operations of genetic programming (crossover and mutation) are applied iteratively. A termination criterion is also determined by the user. Therefore, a run is terminated when criteria are satisfied. As a result of evaluation new design patterns are created. (Figure 1)

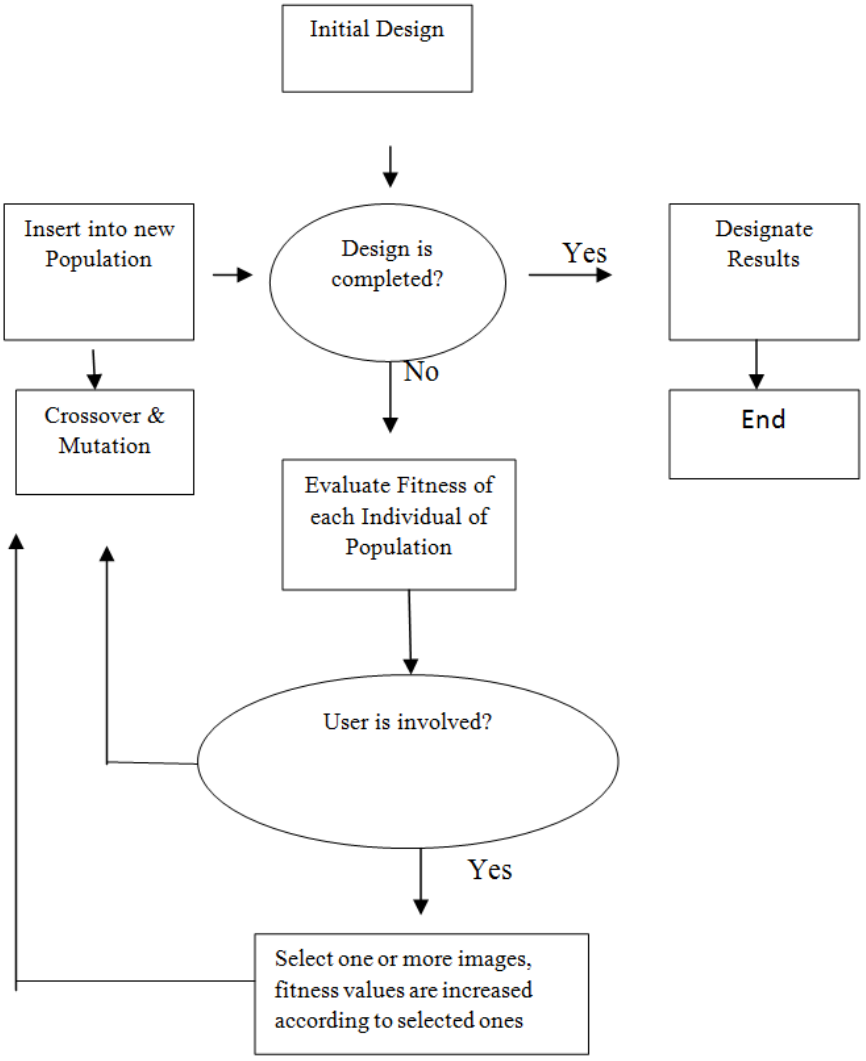

Figure 1: Visual design with GA&GP

II. CHROMOSOME REPRESENTATION

Two methods are used to represent the genetic information of design patterns: bit strings and program code.

1. Bit Strings

In this representation, genetic information for each design pattern is considered to be a bit string (chromosome). Each chromosome has localisation coordinates and colour information. A set of chromosomes form the initial population, as simulation of evolution progresses, a fitness value is evaluated for each surviving design pattern.

2. Program Code

Common LISP programming language is used in order to implement Genetic Programming, because of its capabilities for expressing and evaluating the compositions of functions and terminals. LISP is a flexible and beneficial language, due to its wide range of possible representations. [9]

Initial population is a random composition of functions, such as rotating by an angle, flipping or morphing. Tree representations are used to express LISP programs and to perform genetic operators.

III. CROSSOVER

In Genetic Algorithm approach, either one parent design image or two parent design images (mating) could be used to generate new generations of design patterns. The latter technique is named as multi-parent approach. By finding similarities between parents, overlapping, eliminating features and applying

evolutionary processes, new design patterns emerge. Fitness formulations of various forms, e.g. closed areas are assigned higher fitness values are used to select the most successful designs within each generation.

The genetic process of reproduction between two computer programs is used to create new offspring computer programs which are then selected in proportion to their evaluated fitness. [4] The parental [programs are typically of different sizes and shapes. The offspring programs are composed of sub expressions (sub trees, subprograms, subroutines, building blocks) inherited from their parents. These offspring programs are typically different sizes and shapes from their parents, but they preserve the essential features of their parents. [2, 5]

In Genetic Programming approach, one tree might be inserted randomly into the other, or sub trees might be exchanged. [9]

IV. MUTATION

Mutation consists of only one input design pattern. That is, it can be applied separately on offsprings one by one. By rotating by an angle, flipping or morphing (thicker lines, fatter areas, thinner, longer and shorter functions) are the features of this technique.

In addition, in the Genetic Programming approach a change might be made to a sub tree: changing a leaf node from a constant to a variable, inserting or deleting an internal operator node, or changing a node from one function to another. [2]

V. CONCLUSION

This research aims at improving general design activities by using Genetic Programming (GP) and Genetic Algorithms (GA). In a typical session a user starts with an initial population of design patterns. At each interaction, designs with lower fitness values are eliminated and the remaining designs are applied genetic operators of crossover and mutation to form the next generation of design patterns. The user (the designer) of the process has manual control over each step. Design session ends when the user requirements are met.

The work on the development of new crossover and mutation operations specific for general design work has been continuing. We search and experiment with these newly defined operators. We intend to report our results in a future publication.

References:

1. A source of information about the field of genetic programming and the field of genetic and evolutionary computation. (n.d.) Retrieved February 09, 2011, from http://www. Gebetic-programming.org/

2. Goldberg, D.E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Addison- Wesley Longman Publishing.

3. Graphic Design. (n.d.) Retrieved February 07, 2011, from Wikipedia: http:// en. Wikipedia. Org/ wiki/ Graphic design

4. Köppel, M. &Nickolay, B. (1998, May). Design of image exploring agent using genetic programming. Berlin, Germany.

5. Koza, J.R. (1998). Genetic Programming on the Programming Computers by Means of Natural Selection. Cambridge, Massachusetts, London, England: MIT Press.

6. Lewis, M. Evolutionary Visual Art and Design. In M. Lewis. USA: ACCAD.

7. Muni, D.P., Pal< N.R., & Das, J. (2006). Texture Generation for Fashion Design Using Genetic Programming. ICARCV. IEEE.

8. Visualization _ Computer Graphics. (n.d.) Retrieved February 05, 2011, from Wikipedia: http:// en/Wikipedia.org/wiki/Visualization _ (computer _ graphics)

9. Winston, P.H., &Horn, B.K. (1981). LISP. Addison – Westley.