3-D Modeling Using Collaborative Evolution

Juan C. Quiroz
Computer Science and Engineering
University of Nevada, Reno
Reno, NV
1-775-813-4434
quiroz@cse.unr.edu

Amit Banerjee
Science, Engineering and Technology
Penn State University, Harrisburg
Middletown, PA
1-717-948-6661
aub25@psu.edu

Sushil J. Louis
Computer Science and Engineering
University of Nevada, Reno
Reno, NV
1-775-784-4315
sushil@cse.unr.edu

ABSTRACT
We present an implementation of a framework for creative design based on collaborative interactive genetic algorithms. The hypothesis is that creative designs can be produced if the design search space can be continuously expanded not just by modifying the values of variables but also by adding new ones. The framework presented herein transforms well-formed 3D models by evolving vertex programs, and allows designers to collaborate with peers by injecting solutions from their peers into their own populations. 3D models were created individually and collaboratively, and an evaluation of the individually versus collaboratively evolved 3D models showed that the majority of the evaluation participants rated the collaboratively evolved models as more creative.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms
Algorithms, Design, Human Factors

Keywords
Creative design, collaboration, interactive genetic algorithm

1. INTRODUCTION
In previous work [1], we presented a collaborative peer-to-peer methodology that allows users to exploit and guide evolutionary computation to breed new ideas quickly. In our methodology, users individually guide interactive genetic algorithms (IGAs) with the ability to case inject solutions from peers into their own evolving populations. The methodology was tested on a floorplanning problem, which concerned the art and science of laying out relationships between rooms and spaces within an enclosed area subject to both objective and subjective design constraints. The purpose was to support the hypothesis that collaboration in design results in more "creative" solutions when compared to a similar methodology implemented without collaboration. However, the nature of graphical visualization of 2D floorplans limited the ability of the participants to identify the creative aspect of the design. In addition, the implementation addressed the question of whether collaboration was enough to introduce a creative potential without explicitly expanding the search space by adding variables.

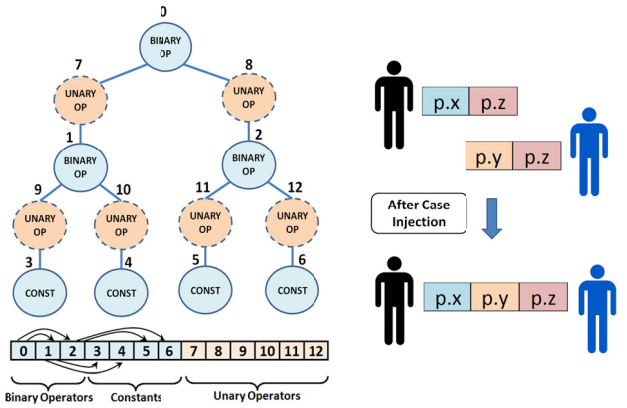
GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.
ACM 978-1-4503-1178-6/12/07.

Creativity has the potential to occur when new variables are added during the evolutionary search [2]. In this paper, we present a modified implementation that expands the search space during interactive evolutionary sessions by adding variables to introduce the potential to generate creative solutions. We apply the modified implementation to the problem of 3D modeling. Advanced 3D modeling software is expensive and its complexity results in steep learning curves for users. Even for experienced 3D artists, the process of coming up with a creative 3D model can be time consuming. Our implementation enables users to create variations of a well-formed 3D model, allowing a user to use an existing model as the seed to quickly create and explore a large number of variations of the 3D model by interacting with an IGA.

In particular, 3D modeling provides a problem context where the value of collaboration and its effects on creativity can be more easily assessed as compared to the previously tested 2D floorplanning problem. In this paper, we present a novel representation used to evolve 3D models, explain how case injection expands the search space during an evolutionary run, and present some preliminary results of our collaborative IGA methodology.

2. METHODOLOGY
We explore variations of well-formed 3D models with an IGA by allowing users to evolve functions for a Cg vertex program [3]. Vertex programs have great flexibility in that local transformations can be performed on a 3D model on a per-vertex basis by using complex transformation equations on each vertex.

2.1 Representation
A bit encoded binary tree is used as the representation of the vertex programs. Binary trees can be stored efficiently in an array, which can then be readily mapped to a bit string. For each vertex in a model, the vertex program can modify the x, y and/or z coordinates of the vertex. A node in a binary tree has two child nodes; this imposes a limitation that operations involving two operands can only be represented. Such operations include addition, subtraction, multiplication and division. However, limited pretests indicated that unary operators, such as trigonometric functions, are also useful in creating interesting transformations of 3D models. A second array is therefore used to store unary operators applied to each child node in the binary tree. The representation is shown in Figure 1(a). The first array (representing the binary operations) and the second array (representing unary operations) are combined into a single bit string that is operated on by the IGA with standard crossover and mutation operators. To ensure that the size of the binary tree does not get too large, the type of nodes that can be used was limited; e.g. the leaf nodes are always a constant, such as the x, y, z coordinate of the current vertex or a randomly generated number.



(a) (b)

Figure 1: (a) Two-tier binary string representation, (b) Expanding design variable space through collaboration.

2.2 Fitness Function

The transformed models are rendered using the Object-Oriented Graphics Rendering Engine (OGRE 3D). The IGA maintains a large population of individuals, but only the individuals with the highest fitnesses are displayed to the user for evaluation. The user is asked to select the solution he/she likes best, and the fitness of every other individual in the population is set based on the hamming distance to the user selected solution.

2.3 Collaboration

Figure 1(b) illustrates an example of how case injection is used to expand the search space during IGA sessions. A first user evolves vertex programs that modify the x and z coordinates of each vertex in the 3D model. A second user evolves vertex programs that modify the y and z coordinates of each vertex in the 3D model. Through collaboration, the first user can expand his/her search space of vertex programs from one that that modify only the x and z coordinates to programs that modify the x, y, and z coordinates.

When a solution is injected from the peer set, the IGA replaces the bottom 10% of the population with the injected solution. When a user first injects a solution from the peer set, the genomes of half of the population are expanded with a random bit string (resulting in the portion of the vertex program that modifies the new coordinate being randomly initialized). For the other half of the population, the genomes are expanded by copying, from the peer injected solution, the portion of the vertex program that modifies the new coordinate.

3. RESULTS

A group of participants (n = 20), consisting of students from the Computer Science department at the University of Nevada at Reno, evolved transformations of 3D models individually and collaboratively. The primary requirement given to this group was to create special effects for a video game. An evaluation group of participants (n = 16) was asked to evaluate the solutions generated by the first group of participants with an online survey.

During an individual IGA session, each user only sees his/her evolved solutions. During the collaborative IGA sessions, the user interface screen of the IGA is divided into halves – right and left. The 3D models displayed on the left half of the screen are selected from the user's population and the models on the right half of the screen are selected from peers. Users guide evolution by picking the solution they like best from their own population, and by having the option to inject multiple solutions from the peer set at any one time.

Figure 2(a) shows the well-formed and original, blue female 3D model without any transformations. The model in Figure 2(b) was generated without collaboration and the mode in Figure 2(c) was generated collaboratively. Both individual and collaborative evolution resulted in numerous interesting solutions. But in general, models evolved collaboratively were found to have more dramatic effects, e.g. in Figure 2(c) the evolved vertex program made the female model expand upwards, like the tail of a comet. The collaboratively evolved models also tended to have a less polished look (looking more like a work in progress). Each of the design participants concluded their evolution by selecting a model that most appealed to them.

(a) (b) (c)

Figure 2: Variations of the Female 3D Model developed, (a) original 3D model, (a) without collaboration, (b) with collaboration

The evaluation group was then asked to rate these selected models on a 7-point Likert scale on items that gauged the creativity, novelty and surprising quality of the solutions, and their usefulness in a video game (with and without minor tweaks). Participants in the online survey were shown a row of individually created models followed by a row of collaboratively created models and asked to rate which row they "liked" more and which row was more creative. More than 80% of the online participants rated the row with collaboratively evolved models as more creative.

4. CONCLUSION

Collaboration allows designers to come up with solutions that may not have been possible without adding variables to the search space. We achieve this by evolving vertex programs to create transformations of 3D models. The presented collaborative IGA framework enables users to quickly explore creative solutions by guiding evolution with their personal feedback and by incorporating feedback from peers by borrowing evolved solutions from peers.

5. REFERENCES

[1] Quiroz, J., Louis, S., Banerjee, A., and Dascalu, S. (2009). Towards creative design using collaborative interactive genetic algorithms. *CEC 2009*, 1849–1856.

[2] Gero, J. S., and Kazakov, V. (2000). Adaptive enlargement of state spaces in evolutionary designing. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 14, 31-38.

[3] NVIDIA. (2009). Cg - the language for High-Performance realtime graphics. http://developer.nvidia.com/page/cg main.html.