MUTATING 3D GENERATIVE FORM WITH CO-EVOLVE APPROACH

PI-YUAN TANG
Graduate Institute of Architecture, National Chiao Tung University, Hsinchu, Taiwan.
yuanc@arch.nctu.edu.tw

AND

TENG-WEN CHANG
Graduate school of Computational Design, National Yunlin University of Science and Technology, Yunlin, Taiwan.
tengwen@yuntech.edu.tw

Abstract. The generative design system is developed with the purpose of generating a large number of design plans. This system can let designers unceasingly explore new design plans. The present generative design system has only one kind of generative mechanism. Therefore, it is not feasible for a designer to explore new plans. In this paper, we propose a Co-Generative Form System (COgenForm). COgenForm is a 3D form exploration system that invokes two sest of generative mechanisms. This system uses co-evolutional characteristics to build the steps and framework of the co-generative design process. It includes two groups of generative mechanisms which will evolve with each other. Using this system, the form exploration and generation process can be more dynamic and with more alternatives.

1. Introduction

The generative design system offers an important mechanism for combining both design generation and computation mechanism. However, most generative design systems require either well-defined design problems or logically sound design knowledge. This limitation with its computational strength, allows the generative mechanism to be explored with logical expression and operation such as SEED (Flemming and Woodbury, 1995; Chang and Woodbury, 2004). In addition, the generative design system can also take advantage of current artificial intelligence trend such as genetic algorithms and then generate form and shape anonymously, such as GENR8 (Hembreg, et al., 2001) and Agency-GP (O'Reilly et al., 2001) with evolutionary agent-based mechanism.

One thing is certain : that the generative mechanism will be the key for generating design as well the representation (or exploration metaphor) for the generated design. Different generative mechanisms and representations will generate different design alternatives. Among those, the two main generative mechanisms often mentioned are evolutionary-based mechanisms and symbol-based mechanisms.

Much of the current research trend regarding generative design system focuses only on one generative mechanism. This makes the development simplified and computable. However, the trade-off is to sacrifice the feasibility and inter-relation between different generative mechanisms. If there is a mechanism that can incorporate more than one generative mechanism, the system generated or the design generated by the system might be closer to the design process.

Therefore, in this paper, a mechanism that can incorporate two different generative mechanisms is proposed and a generative system for form-exploration is proposed and implemented.

2. Co-evolve and generative design

For generative design process model, a mechanism called *co-evolution* is adapted for integration between two diverse evolutionary steps. Maher (Maher and Poon, 1997) described the co-evolution as a cyclic process that sketches the influence between two species in the natural world. In recent years, the concept of co-evolution has been applied in different domains, such as co-evolutionary learning (Sklar, et al., 1998) and game developing work. The model of co-evolution design is comprised of three characteristics: (i) two design problems extend in a parallel way; (ii) design problems can affect each other via an active way; (iii) one design problem will keep changing while changing another’s answers.

Generally speaking, generative design systems are comprised of four elements: (i) the design representation (ii) a generation engine (iii) an expression engine (iv) evaluation and selection mechanism. The generation engine is the main part of the mechanism of evolving, and new design instance will be generated within the evolving steps. In this research, we combine these two mechanisms (generative design system and co-evolution model) into the generative engine and make them affect each other through the features of the co-evolution process. The process that combines these two mechanisms is called *co-generation*.

The strategies applied for combining generative design system and co-evolution model are: (i) there are two groups of generative mechanisms in the co-generative process; (ii) the representation of two mechanisms should be the same or interchangeable; (iii) under the same representation, the factors or rules must be different; (iv) one of the generative factors or rules will change another; (v) the use of evolutionary concepts to combine two groups of generative mechanisms, and the generative factors and rules-like genes. These will influence each other like crossover and mutation. We show the co-generative process in Figure 1.

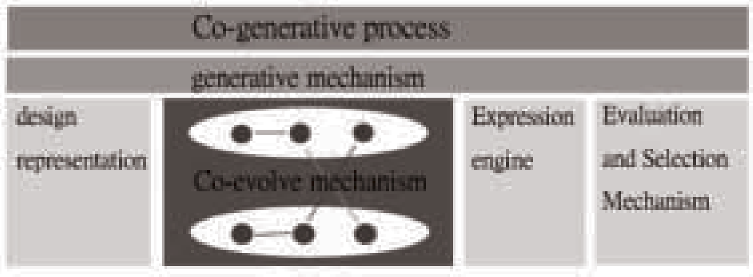

Figure 1. The framework of co-generative process.

3. An implementation framework

For testing the computability of our co-generative process, we implemented a design system for the purpose of 3D form exploration. The implementation steps of a design system using the co-generative design process are described as follow.

3.1. THE STEPS

Three steps that are applied in our preliminary implementation are: (i) selecting two three-dimensional form generative mechanisms, (ii) developing the inter-relationship between these two mechanisms, and (iii) the selection of operators based on the mechanisms and representation chosen. The details of each step are described below.

First, two groups of generative mechanisms will be chosen according to the principle that their representation must be the same or interchangeable. In addition, each can generate form by itself or by the other. In this research, two groups of generative mechanisms set up are L-System mechanism and Rotate.

Second, the co-generative design process must be able to let two groups of generative mechanisms interact with each other. We used the evolutionary concepts to combine these mechanisms. Generative rules or factors such as gene are defined and they will interact with each other through mutation as shown in Figure 2.

Finally, user-controlling strategies in the generative process are divided into two parts: (i) controlling the timeline, and (ii) adjusting the rules and parameters of design. The user can also decide when to process the mutation among generative processes. Both the generative mechanisms will keep generating alternatives according to the set rules until the user stops the timeline. Once stopped, the user can import other generative mechanism to change the rules or the generative form. Thereafter, the natural selection will be decided according to the user's satisfaction of the results.

3.2. IMPLEMENTATION

A co-generative 3D form system called *COgenForm* is implemented according to the aforementioned analysis. COgenForm is comprised of four elements: (i) Two groups of generative mechanisms (L-System mechanism and Rotation mechanism); (ii) Co-generative interaction; (iii) Selection operator unit; (iv) User interface. This version of COgenForm is implemented using MEL/Maya. MEL (Maya Embedded Language) was the main programming language of COgenForm.

4. An example

A housing design is used as the application of COgenForm. The requirements of this housing design are (i) a 3-bedroom apartment located at Taichung central park; (ii) participants need to discover diverse form strategy to fulfill the function and site requirements. We provide the system COgenForm which lets a designer with architecture design background test this project. The purpose of this housing design is to explore more alternatives and more possibilities with two parallel logic form generative mechanisms. The results are shown in Figure 2 which desides two generative processes A and B. Process A uses L-system and Process B uses rotation features as the mechanism to generate free form of housing design. The top left window shows the generation 22 of a sequence of alternatives using L-systems, and the right is generation 55. The timeline and parameters of generation process are shown at the left side of the windows.

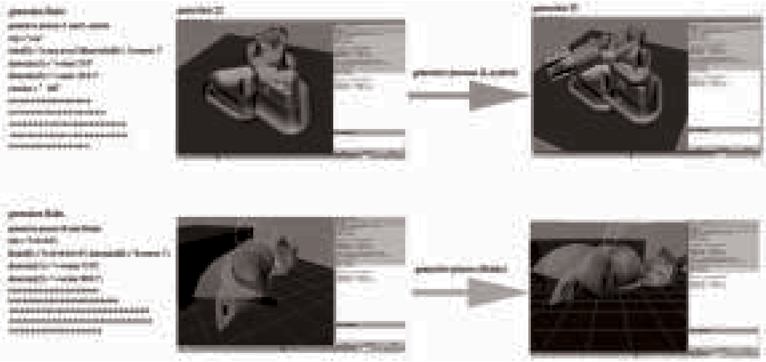

Figure 2. Two generative process that are used L-system and rotate.

In Figure 3, we show how to combine the two generative processes with mutation and user control. This situation occurs once a designer decides to mutate the design mechanisms. The Generation 82 of process A is then generated by mutating both mechanism *a* and mechanism *b* based on its parent node.

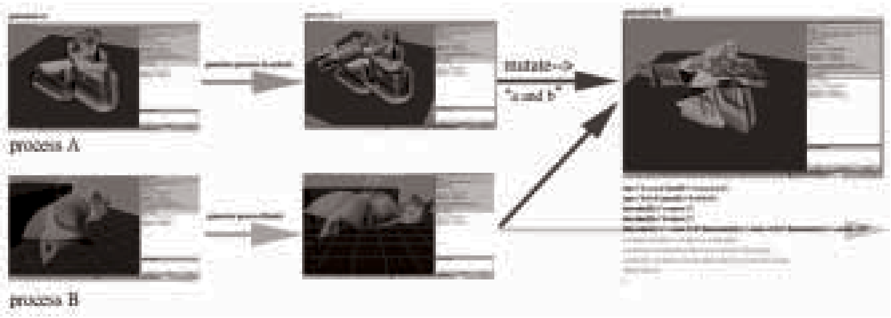

Figure 3. The co-generative process with mutation operations.

5. Conclusion

In this paper we present a “co-generative form” system called COgenForm. Via the feature of co-evolving, the generative engine of COgenForm includes two groups of generative mechanisms. Through the influences and stimulation between each other. COgenForm enriches the varieties and possibilities of the children generation and its processes, as shown in the foregoing example. The inspiring results that inherit both mechanisms are shown in the display.

By directly manipulating these mechanisms, designers can discover more alternatives or detours from the original fixed generation patterns. The computability of applying co-evolution metaphor on to design generation is clear and useful. In additions constraints by representation and its mechanisms are unleashed in the paper.

In the future, more generative mechanisms that fulfill the four requirements of co-generative process can be implemented in COgenForm. Further, the interfaces of COgenForm for the users can be evaluated and argued through the refinement of the system. Hopefully, it can make the form generated by COgenForm more inspiring and interesting.

Acknowledgement

This research thanks Reau-Wen Tsai and Ting-Han (Daniel) Chen for the stimulating discussion. The initial concept was thanks to Shang-Yuan Chen’s critical discussion.

References

Chang, T-W. & Woodbury, R.F. 2004, Highly Structured Design Spaces with Geometry, in H Orbak (ed.), *The eCAADe 2004 22nd conference*, Copenhagen, Denmark, pp: 248–254.

Flemming, U. & Woodbury, R.F. 1995, Software environment to support early phases in building design (SEED): overview, *Journal of Architecture Engineering*, vol. 1, no. 4, pp. 147–152.

Hembreg, M., O'Reilly, U. M. & Nordin, P. 2001, A Design Tool for Surface Generation, Submitted as a late-breaking paper, Genetic and Evolutionary Computation Conference, *GECCO 2001*.

O'Reilly, U. M., Testa, P., Greenword, S. & Hembreg, M. 2001, Agency GP: Agent-Based Genetic Programming for Surface Design, Submitted as a late-breaking paper, Genetic and Evolutionary Computation Conference, *GECCO 2001*.

Maher, M. L. & Poon, J. 1997, Co-evolution and Emergence in Design, *Artificial Intelligence in Engineering*, vol. 11, no. 3, 319–327.

Sklar, E., Blair, A. D. & Pollack, J. B. 1998, in G. Ayala (ed), *Workshop on Current Trends and Applications of Artificial Intelligence in Education: Fourth World Congress on Expert Systems*, pp. 98–105.