DYNAMIC INTERFACE FOR AXIAL SPATIAL ARRANGEMENTS USING SOFT COMPUTING

Yoshihiro Kobayashi
Arizona State University
College of Architecture and Environmental Design
ykobaya@asu.edu

Meenakshi Sharma
Arizona State University
College of Architecture and Environmental Design
Meenakshi.Sharma@asu.edu

Abstract

This paper describes a computer application that can organize 3D objects along an axis and also redistribute those based on a set of constraints or existing patterns as selected by the user. In particular, the focus of the paper is the two dynamic Graphical User Interfaces (GUIs): the Constraint Arrangement (CA) and the Pattern Arrangement (PA). For these, we use the Soft Computing Techniques – Kohonen's Self Organizing Map (SOM) and Genetic Algorithm (GA). In both the cases, the application outputs an axial organization of predefined 3D objects which, either adhere to selected constraints or follow patterns set by prior design. The application is implemented, tested and its results are demonstrated using buildings systems.

1. Introduction

Computers have been successfully used as a means of exploring form and spatial arrangements in architectural design. This study demonstrates this by means of an application that can arrange and redistribute 3D objects along an 'Axis' based on their individual physical properties.

Most of the CAD and CG applications (AutoCAD, Maya, formZ, 3D Studio Max) have basic transform functions such as scaling, translating, rotating, etc. for 3-Dimensional Modeling. More advanced applications have more complex transform functions, in which the users need to specify a lot of parameters. They have an intimidating number of menus, modes and widgets for object transformations and scene manipulation [1], which apply to the selected objects in the scene. However, the existing approaches restrict the space of design alternatives, make it difficult to compare alternatives and require tedious user intervention. Also, a designer sometimes needs to rearrange the positions of 3D objects similar to prior designs.

Considering these needs of the designer and what the current applications offer, we propose to use *SOM* and *GA* as new approaches in this field. SOM has an advantage of being able to organize the alternative designs visually, whereas GA can optimize the results in a limited time. The SOM can be visualized as a sheet-like neural-network array, the cells (or nodes) of which become specifically tuned to various input signal patterns or classes of patterns in an orderly fashion [2]. The SOM was developed by Prof. Teuvo Kohonen in the early 1980s [3]. The SOM is widely used as a data mining and visualization method for complex data sets like image processing, image recognition, process control, economical analysis, and diagnostics in industry and in medicine [2].

GA is a randomized search and optimization technique based on natural biological evolution using a direct analogy of natural selection and genetic processes. GA is used for problems which could have multiple solutions and the optimum one is found amongst them. GA has been successfully used in shape design [4, 5], Pattern Matching and Pattern Recognition, amongst others.

2. Methodology

2.1. Constraint Arrangement using Self-Organizing Map

In CA, we use the physical properties (distance from axis, area and volume) of 3D objects as parameters to build equations manually for the constraints. The planar SOM consists of a regular grid of "neurons" as the processing units. Each cell on the SOM is associated with a constraint and the *result set* of the constraints are arranged topologically on the map, such that the related ones lie closer to each other compared to un-related ones. It makes it easier for the user to judge and select the appropriate constraint by the mouse actions. From the selected cell in the SOM, the constraint is applied to the 3D objects in the spatial arrangement which impacts the position of each of the object with respect to the axis.

To form the linear constraints, weights are applied to the three parameters to form a possible set. For the non-linear constraints, various combinations of these parameters are formed and functions applied to these.

A function f(1) is applied to get the new distance of object from axis. The input parameter in this case is old distance.

f(1): new distance = old distance * 0.5

The above function is the one of the constraints that result in the sample data for the SOM. For instance, if for a set of five 3D objects, the input set is {0.5, -0.8, 0.3, 0.75, -0.2}, the output after applying the above mentioned function would be {0.25, -0.4, 0.15, 0.375, -0.1}. See Table 1 for a partial list of constraints, with function ID 0 as the original distance. A set of 100 functions were manually defined and used for the application.

Table 1: Partial list of functions forming constraints.

ID	New Distance to Axis	ID	New Distance to Axis
0	dis	76	**pow(volume, 1.0/3.0)/dis**
4	0.66*dis	77	**dis+ sqrt(area)/dis**
25	0.075*area	88	**dis- sqrt(area)+ pow(volume, 1.0/3.0)**
54	dis+0.0005*volume	94	**sin(dis)+(0.001*volume)**
62	**0.2*dis+0.01*area-0.005* volume**	**99**	**area*0.01+ cos(volume/100.0)**

2.2. Pattern arrangement using genetic algorithm

The second GUI, PA is used to imbibe the spatial arrangement pattern of a prior design in the current design most appropriately. When the number, size and volume of spaces are completely different between a current design and prior design, it is difficult to find the optimal positions manually. PA is designed to solve this problem by using the technology of GA.

This application is based on a *four bit gene model*. This implies that for each 3D space, its x and y-coordinate of the center point is coded in four bits. Example:

x = 0100, y = 0110

The total number of possible combinations for x-coordinate and y-coordinate this way would be 2^4 x 2^4 that is 256 possible positions and these would be mapped on a 16 x 16 grid. A gene with *n* 3D objects is:

$$\text{Gene} = \{x_0, y_0, x_1, y_1, x_2, y_2 \ldots x_n, y_n\}$$

For four 3D objects, the typical Gene = {0100, 1010, 0110, 0001, 0011, 0101, 0000, 1011}

Using four bits for the GA gave best results in terms of time taken to optimize and the results found. There is a function introduced to calculate the occupancy for each position on the grid. For each cell on the grid, the following function is used, where f(i,j) is the function, l is the total number of objects, n(i,j) is the occupancy at the cell position (i,j).

$$f(i, j) = \sum_{k=0}^{l} n(i, j)$$

The minimization of a real function is described below, where $f(i, j)$ belongs to the instance and $f_0(i, j)$ to the pattern.

$$\min \sum_{m=0}^{i} \sum_{n=0}^{j} | f(i, j) - f_0(i, j)$$

The score is increased by one if ($f(i, j) > 0$ and) $f_0(i, j) > 0$) and the score is increased by one if ($f(i, j) = 0$ and $f_0(i, j) = 0$), else the score remains same. The result for 100 populations is checked in one generation and there are 1000 generations. The crossover rate used is 0.3 and the mutation rate is 0.05.

3. System framework

In CA:

- The input set is formed by a set of 3D objects.
- There are 100 pre-defined functions, to which more can be added by hard-coding.
- The user's task here is to specify the Axis in the 3D arrangement.
- The task of the CA Engine is to apply each function to the original objects to form spatial arrangements.

For each arrangement generated by the CA Engine, a set of numbers is formed by calculating the distance of each translated object to the axis. For 100 functions, 100 arrangements are formed and in turn, 100 such sets are calculated. Then the 100 sets are topologically assigned to the cells in the SOM space. The user then selects one of these cells and can view the resulting arrangement. Neighboring cells display similar patterns making it easy for the user to make selections. Based on the arrangement selected by the user, the CA Engine updates the data for the SOM space.

In PA:

- The input set is formed by a set of 3D objects.
- Here, the user's task is to specify the Axis in the 3D arrangement and as well as selecting a pattern from one of the predefined ones.
- The predefined patterns follow a 16x16 grid.
- The task done by the PA Engine results in an arrangement that best matches the selected pattern with respect to the positions occupied by the objects.

The new arrangement is drawn in the viewer panel. DXF files can be easily imported and exported from within this application that is programmed in Java. The data needed for CA and PA is described as XML files.

4. Case studies

Richards Medical Center, Philadelphia, Pennsylvania designed by Louis Kahn has been used as case study to demonstrate how axis-based arrangement using CA and PA works for practical purposes. The building is formed of different stacks of spaces attached to tall service towers with circulation along an 'Axis'. The building is a synthesis of twenty-two 3D spaces. The testing of the case study for the purpose of CA was done by implementing three functions to the original arrangement. There were three directions of the axis considered for each of these – Case 1, Case 2 and Case 3. The results are successfully

displayed in Figure 1. We observe that the cells close to each other have similar resulting arrangements and help the user visualize the results.

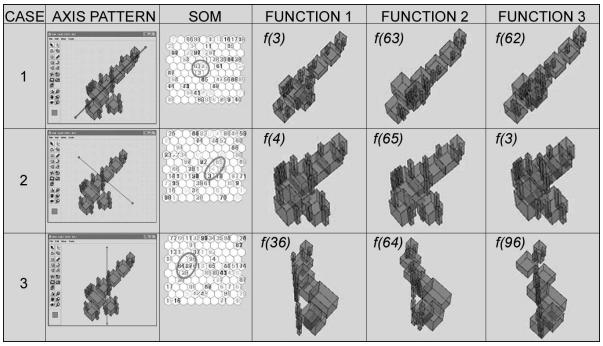

Figure 1: Arrangements resulting from use of CA for three tested cases.

In the case of PA, there are three patterns that are defined. Each of these patterns is applied to the original arrangement with three configurations for the axis direction. This also results in nine samples as shown in Figure 2. As seen, the application successfully matches the physical pattern of the current design to the prior design.

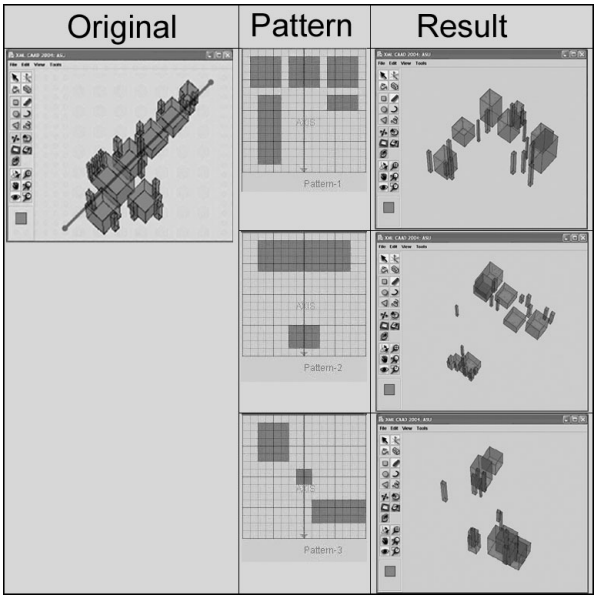

Figure 2: Arrangements resulting from use of PA using three different patterns.

5. Conclusions and future work

In conclusion, the application successfully presents and demonstrates two user-friendly GUIs, which help the designer explore axial arrangements dynamically with the help of mouse movements. While CA arranges the objects dynamically as the user drags the mouse, in PA, GA takes a considerable amount of time to calculate and display the output. Currently, one drawback is that while doing the PA, the grid size chosen is 16 x 16, which does not allow accuracy in the drawing. If the grid size is increased, GA takes a considerable amount of time to calculate and display the output. We noted that the technique behind this needs to be revisited and more effective fitness function should be applied.

Currently the application is designed for planar arrangements of 3D objects and limited to axis-based transformations. It would be possible to expand this to cater to any kind of transforming function. It would be further developed to be used for arrangement of 3D objects in 3D environments. This would imply development of a 3-Dimensional SOM as well. By introducing more parameters and constraints, the user would have more control over spatial arrangements and a larger search space for optimal design solutions.

Finally, the methodology allows the proposed GUI to be adapted for use in several other applications for digital spatial arrangements.

References

1. Smith, G., Salzman, T. and Stuerzlinger, W., 3D scene manipulation with 2D devices and constraints, Graphics Interface Proceedings, 2001, Ottawa, Ontario.
2. Honkela, T., Self-organizing maps in natural language processing, Helsinki University of Technology, Espoo, Finland, 1997.
3. Kohonen, T., Construction of similarity diagrams for phonemes by a self-organizing algorithm, Helsinki University of Technology, Espoo, Finland, 1981.
4. Gero, J.S., Louis, S.J. and Kundu, S., Evolutionary learning of novel grammars for design improvement, Artificial Intelligence in Engineering Design, Analysis and Manufacturing, 1994, 8, 83-94.
5. Watabe, H. and Okino, N., A study on genetic shape design, in: Forrest, S., eds., Proceedings of the Fifth International Conference on Genetic Algorithms, Morgan Kauffman Publishers, San Mateo, California, 1993, 445-450.