PL-SYSTEM: VISUAL REPRESENTATION OF PATTERN LANGUAGE USING L-SYSTEM

JUNAH YU¹ and DEEDEE MIN²
¹,²Department of Interior Architecture Design, Hanyang University.
¹abcdmi0115@hanyang.ac.kr, 0000-0002-6765-4813
²mindeed@hanyang.ac.kr, 0000-0002-6651-5133

Abstract. Pattern Language provides simple and conveniently formatted solutions to complex design problems ranging from urban planning to interior design for community-led inclusive designs. Despite the intention, the concept has been more widely adopted by computer science professionals. One possible reason is the lack of visualization, making it difficult to be used interactively by the non-professionals. To overcome the issue, we aim to integrate the patterns from Pattern Language into the L-system to visualize the paper architecture into geometric forms. Specifically, we implement Pattern L-System (PL-System), which generates diagrammatic floor plans using design rules based on the pattern languages. We first made analogical comparisons of the concepts and grammar structure between Pattern Language and the L-system. Next, we defined a geometric interpreter for drawing diagrammatic floor plans using turtle graphics, which consist of geometrical rules for putting shapes together. Then, we selected three patterns and reinterpreted them for visualization using strings of turtle graphics letters that determine the turtle’s movements for the geometric representation of walls, columns, and doors. From this research process, we learned that Modular L-system opens up the possibility for the visualization of the patterns in Pattern Language.

Keywords. Pattern Language; L-System; Diagrammatic Floor Plan; Turtle Graphics; Geometric Forms; SDG 11.

1. Introduction

Christopher Alexander’s Pattern Language Theory (1977) was originally created for design purposes. Pattern Language contains rules for how human beings interact with built forms reflecting different lifestyles, customs, and behaviours (Salingaros, 2014). This characteristic of pattern language allows it to be applicable for comprehensive design language. In fact, research was conducted that evaluates the applications of patterns from Pattern Language for universal design, including accessible, green, and public space (Lee, 2015; Iwańczak & Lewicka, 2020).

While Pattern Language is useful conceptually in architectural design, they present a lack of usage in architectural design and planning in practice mostly because they are
semantic and abstract, lacking in objectivity. From these characteristics, two issues arise when incorporating Pattern Language in design practice. The first issue is about the ambiguity of network structure patterns connections and the interrelationship between the patterns. In order to resolve this issue, a sequential approach has been proposed (Porter et al., 2005; Ribeiro, 2020). The second issue is about the lack of visual aid. As pointed out by Ribeiro (2020), visual aids such as photos or illustrations are essential when adopting Pattern Languages into a design process. However, this problem is left largely unsolved. Therefore, in this study, we tackle the visualization problem to increase the applicability of Pattern Language into design practice for all-embracing community designs.

To achieve this aim, we incorporate the concept of L-system to Pattern Language. In particular, we focus on creating a floor plan diagram using a Modular L-System with rules based on the patterns from Pattern Language. In order to achieve this goal, we first made analogical comparisons of the concept, characteristics, and grammar structure between Pattern Language and the L-system. By observing the two concepts theoretically, we were able to identify that they have different grammatical structures, which serves as a limitation when coupling the two. While the L-system is a parallel rewriting system and recursive, Pattern Language is serial and has a network structure. To overcome this limitation, we incorporated the concept of the Modular L-system proposed by Cieslak et al. (2011) as a strategy to develop an L-system using separate modules to represent different aspects. After, we defined a geometric interpreter for drawing floor plan diagrams using turtle graphics. Turtle graphics consist of geometrical rules for putting matter together. Turtle graphics can perform a geometric interpretation of the new L-string in each derivation step upon request. Next, in order to incorporate Pattern Language into the system, we selected three patterns applicable for rules from Pattern Language. The patterns were reinterpreted and represented using turtle graphics letters. At this stage, we were able to create modules that would become the new definition of the pattern using Turtle graphics letters. Finally, we implemented the PL-System using Processing language. From this research, we aim to answer whether L-System is applicable for opening up the possibility of extending the use of Pattern Language in the architecture field by visualizing semantic patterns into geometric forms facilitating the use for comprehensive community design by the non-professionals.

2. Literature Review

2.1. PATTERN LANGUAGE

Christopher Alexander has been a leading pioneer of academic research on the architectural and urban design since the early 1960s (Galle, 2020). Alexander addressed the problem of capturing recurring problems and their solutions in the context of civil architecture in the 1970s. In his work, Alexander struggled with the need to document and share architectural knowledge, which could be easily applied by his fellow architects. The theory of Pattern Language is a design theory for deriving 253 patterns from physical environments with design problems and solutions. According to the theory, because good spaces have regular and timeless patterns, professionals or ordinary people can design new spaces by modifying or combining
the patterns. Unlike the original intention of the Pattern Language, it has been applied to mostly computer science fields, including user interface configuration and evaluation of web services (Pautasso et al., 2016).

2.2. L-SYSTEM

In 1968, biologist Aristid Lindenmayer proposed a string-rewriting system that can model simplified plants and their growth process. This theory is now known as L-Systems (Hansmeyer, 2003). An L-System is a recursive and formal grammar system that rewrites collections of characters into new strings based on a certain set of rules. By iterating a number of times over the string, the character arrangement changes. Figure 1 illustrates the rewriting of a string by a set of rules. With the L-System, minimal inputs can create a complex output.

![Figure 1. Rewriting string of L-system](image)

2.2.1. Geometrical Interpretation of Strings: Turtle Graphics

In 1986, in order to model higher plants, Prusinkiewicz (1986) focused on the geometrical interpretation based on a LOGO-style turtle. In turtle graphics, a fictive turtle follows movement commands that correspond to individual alphabets and symbols of a string. The turtle’s path is thus a visual interpretation of the string. F in turtle graphics means to move forward a step while drawing a line, + means to turn right, and – means to turn left at assigned angles. Figure 2 shows an example of the geometric interpretation of the L-System using turtle graphics. The initial string, rule, and rotation angle parameters produce a form that resembles a Sierpinski Triangle:

- Initial string: F
- Rule: F→F+F−F−F+F
- Rotation angle: 60°

![Figure 2. Sierpinski triangle](image)
2.2.2. Modular L-System

Cieslak et al. (2011) proposed a strategy to develop a modular L-system based on the use of separate modules to represent different parts. As not every alphabet of the L-system needs to correspond to a turtle command, new characters can be used that are simply replaced by groups of other symbols and alphabets (Hansmeyer, 2003). For example, with the rewriting rule of \( C \rightarrow F+F+F+F \) where \( F \) implies a movement forward and + implies a right turn, a new character \( C \) could be defined to create a square, as illustrated in Figure 3.

![Figure 3. Example of modular form](image)

2.3. PATTERN LANGUAGE AND L-SYSTEM

In Section 2, we reviewed the characteristics of the Pattern Language and the L-System. Pattern Language is useful for interpreting complex socio-spatial considerations through a simple building-block format, which makes this content accessible to non-professionals (Dawes & Ostwald, 2017). Meanwhile, using the L-system with a set of rules, simple initial objects can be successively replaced, resulting in a more complex final object (Hansmeyer, 2003). Through this process, L-systems can generate two-dimensional and three-dimensional geometries efficiently that otherwise would have taken an extended amount of time (Serrato, 2005). As discussed earlier, these two concepts have different grammatical structures—one being serial with a network structure and the other one being parallel with recursive structure—serving as a limitation when coupling Pattern Language and the L-system. Despite this limitation, the use of the L-system should be considered, given that, it is highly scalable and portable, consisting of scripts that can be parsed and easily rewritten. By expanding the logic to include independent parameters, we can gain a high degree of control over the produced form (Hansmeyer, 2003).

3. System Architecture

3.1. SYSTEM FLOW

The implemented PL-System largely consists of three modules. One is the turtle graphics module for visualization, another is the pattern definition module for the extraction of patterns from Pattern Language, and the third module is the rule production module for the definition of the drawing rules according to the patterns extracted. These three modules are linked into an iterative system flow, as shown in Figure 4. First, the system initializes the parameters of turtle graphics with start length, start thickness, and rotation angle. Next, the system creates patterns using the inputs by
the user, which are based on the patterns from Pattern Language. This process of adding patterns can be iterative as long as the user desires. When the patterns are set, the rule production module is set up. Using the rule production module, rewriting of the rules are iterated until the user’s needs.

![Figure 4. The System architecture of the PL-System](image)

### 3.2. Initializing Turtle Graphics

As mentioned above, the system initializes the parameters of turtle graphics using the user inputs of start length, start thickness, and rotation angle. Each of the inputs is used to define the turtle commands, as summarized in Table 1. For example, the wall element is represented using the character w, and its command is to move forward while drawing a line with the user-defined parametric length and thickness. Or, for the corner element, the character + commands the turtle to turn right at a user-defined angle. Using these turtle commands, the patterns from Pattern Language can be visually represented.

<table>
<thead>
<tr>
<th>Elements</th>
<th>Characters</th>
<th>Commands</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1 Turtle commands
3.3. PATTERN DEFINITION

Among the patterns from Pattern Language, we chose three patterns specifically related to the layout of a floor plan consisting of walls, columns, and doors. Then from the semantic information provided by Pattern Language, we redefined the patterns using the characters of the turtle graphics, which provides pattern modulation. Then the modulated pattern can be instantiated as a pattern module and be stored in the system, and be used later in the rewriting process of PL-System.

In detail, we selected the patterns 196, 216, and 212, which are Corner Doors, Box Columns, and Columns At The Corners, respectively. For the redefinition process, we extracted the related problems and their corresponding solutions. Table 2 represents our redefinition results:

<table>
<thead>
<tr>
<th>Patterns</th>
<th>Problems</th>
<th>Solutions</th>
<th>Redefinition</th>
<th>Pattern Modules</th>
</tr>
</thead>
</table>
| 196. Corner Doors | “If the doors create a pattern of movement which destroys the places in the room, the room will never allow people to be comfortable” (Alexander, 1977, p.904). | “In most rooms, especially small ones, put the doors as near the corners of the room as possible” (Alexander, 1977, p.905). | Place door(d) next to corner (+ or -) | A(1) = f+
A(2) = f-
A(3) = +f
A(4) = -f |
| 216. Box Columns  | “Columns feel uncomfortable unless they are reasonably thick and solid…[and] must be easy to connect to foundations, beams and walls” (Alexander, 1977, p.1013). | “A column which has all these features is a box column” (Alexander, 1977, p.1015). | Place box shape(r) columns | B = r |
| 212. Columns at The Corners | “The necessities of the drawing itself change the plan, make it more rigid, turn it into the kind of plan which can be drawn and can be measured” (Alexander, 1977, p.990). | “On your rough building plan, draw a dot to represent a column at the corner of every room…” (Alexander, 1977, p.993). | Place column(B) next every corner (+ or -) | C(1) = B+
C(2) = B-
C(3) = +B
C(4) = -B |
3.4. RULE PRODUCTION

After the pattern definition, rules can be produced for each of the pattern module IDs and the parameters initialized in turtle graphics (start length, start thickness, and rotation angle). The rules at this stage of the system aim to diagram floor plans by mainly converting the input parameters that describe the floor plan morphology into L-system parameters. For the rule production, three distinct inputs are required, which are the initial string, one or more substitution rules, and the number of times to perform the rewriting operation.

The rewriting process is divided into two parts. In the first rewriting process, the final sentence is returned through the rewriting process by applying the rewriting number, axiom, and ruleset input by the user. The system performs the rewriting process by setting the returned final sentence as an axiom of the new rewriting process and pattern modules as rulesets. During the second rewiring process, the system visualizes the final sentence derived through two rewriting processes on the display area of the floor plan through turtle interpretation.

4. System Design

4.1. USER-SYSTEM INTERACTION

As shown in Figure 7, the PL-System receives the turtle parameters, pattern module, and the production rules from the user to perform rewriting. Then after the calculation, the graphical interpretation of the patterns along with the axiom and the rewriting rules are displayed to the user.

![Figure 7. User-system interaction](image)

4.2. GUI (GRAPHICAL USER INTERFACE)

In order for the abovementioned user-system interaction to occur with high understandability, a well-designed GUI is necessary. Therefore, as illustrated in Figure 8, the developed PL-System enables users to control the system through three different distinctive inputs and outputs for each of the modules: the pattern module, production
rule, and turtle graphics. For example, for the pattern module (Area A), pattern ID and the pattern components are received from the user. Then, these two inputs are combined, saved, and represented as a list of pattern modules (string) in Area E. As for the production rule input (Area B), the number of generations, axiom (string), and rulesets is received. Then the geometric interpretation of axiom and the rewriting rule is displayed in Areas F and G, respectively. The advantage is that the user can visualize the geometric shapes for the axiom and rewrite rules in real-time. Finally, for the turtle graphics input (Area C), the starting length, thickness, and rotation angle are received from the user. Integrating the data received from Area A and B, the system performs the rewriting and visualizes the floor plan accordingly in Area D.

Figure 8. GUI layout
4.3. SYSTEM DEMONSTRATION

To demonstrate how our PL-System can graphically visualize a user-defined pattern based on the production rule derived from the redefinition of Pattern Language, we used all the pattern modules mentioned in Table 2 and set the generation number as two. Then, the axiom was defined as \((w+w)\), which commands to draw a wall at a given length, turn at a given angle, and draw another wall at the same length as shown in Figure 9 inside the axiom display Area F. Then, the axiom can be elaborated into a form illustrated in the display Area of D using the user-given production rule and turtle parameters where the start length is 100, thickness is 20, and the rotation angle is 90°.

![Figure 9. Demonstration of PL-System](image)

5. Discussion

This paper proposes the possibility of integrating the L-System for the graphical interpretation of the patterns from A Pattern Language. The fundamental advantage of the L-System is that minimal inputs can create a spatially complex output. By expanding the logic to include independent parameters, we can gain a high degree of control over the produced form. According to the original intention of the Pattern Language Theory, users should be able to transform the given patterns for their own use. Along with this flexibility, Pattern Language has high scalability, applicable for participatory and inclusive sustainable design for small to large scale design projects. However, because the patterns are semantic, the non-professionals have difficulty transferring the knowledge into practice. In response, we proposed a PL-System that is able to redefine patterns with characters and create a diagrammatic floor plan.
composed of walls, columns, and doors based on the generative concept of the L-System.

The PL-System proposed and developed in this paper meets the theoretical requirement of Pattern Language, which is the transformability with a high degree of control over the produced form but at the same time, deriving many patterns under certain parameters. The use of the PL-System in this paper is tested using the simple drawing of architectural elements such as walls and columns. By visualizing the patterns into geometric forms and using the recursive functions, we expect that the system will aid non-architects to understand the relationship between architectural elements and spaces visually when designing a public space for sustainable community. Further research will focus on the verification of the usefulness and usability of the system through user tests with both professionals and non-professionals.

References