IJWR

L-Atur Generative Design L-Systems Based Web Application: New Challenges and Novel Application Horizons

Rashin Rahnamoun [a*], Romina Rahnamoun [b]

[a] Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran; rahnamounrashin@gmail.com
[b] Faculty of Visual Arts, University of Art, Tehran, Iran; rahnemoonromina385@gmail.com

ABSTRACT

L-Atur is a web application with the essence of human and machine collaboration that assists users who don't have coding or design skills to generate their customized products. We noticed that our previous version only supported small-scale designs properly, and the generated shapes didn't provide enough visual complexity for large-scale results. In this paper, we improved the supported grammatical properties in order to expand design variations. We offered random and specified color choices, variable line thickness, and multi-rule L-systems grammars as the new options for the current version of L-Atur. We also provided an in-depth analysis of how these new suggested properties can have fundamental impacts on the generated designs visually. We also discussed grammatical evolution as a solution to improve grammatical diversity by using machine-generated ones. As a result, the current version can cover broadened design aspects ranging from small-scale products to large ones with diverse levels of complexity and a high level of visual novelty.

Keywords— L-systems, Human-Machine Collaboration, Grammatical Evolution, , Computational Aesthetics.

1. Introduction

L-Atur [1] (L is borrowed from Lindenmayer and Atur means fire in middle Persian language) is a web application that integrates generative design and Lindenmayer systems (L-systems) based functionalities with human and machine cooperation. As a web application, our goal is to help individuals lacking coding or design skills personalize and design their unique products, which are computer-generated. Currently, there is no web application available at an affordable price that offers a high variety of products with diverse designs and also includes colorful generative shapes for non-designer users. However, L-Atur provides all of these features. Our method is focused on L-systems because, by using grammar theory, they generate innovative and fascinating geometric patterns [2]. L-Atur's web application earlier version was modified for creating small items, particularly accessories.

We must keep in mind that L-Atur is intended to be adjusted for a wide range of design probabilities. For resolving all these issues, we proposed a new logical diagram of L-Atur's web application. We also observed that our current system has a number of significant issues, including a dearth of grammar. Since humans wrote the prior version of our dataset, there weren't as many grammars to choose from. Although it worked as intended for small-scale items and users had the option to alter the parameters to their liking, there were two major problems. One of them was the lack of inclusion of colors in the design procedure, as many products cannot be made available without incorporating colors into the design. The other one, which was much more fundamental, is the fact that we needed to improve our supported grammars. We wanted to let our users to utilize broadened grammatical options, hence it makes much more diverse design possibilities.

In this paper, we offer a solution to provide greater design possibilities by expanding the supported grammars. We added several grammatical properties, which led to dramatic visual differences in the final results. These key grammatical properties are variable line thickness, color, and an expanded number of rules, which results in an adjustable complexity level for different practical usages. We also had an in-depth analysis of the provided results to showcase the effect of the referred parameters in the final design. We took one step further and discussed about generative grammars, we believe that using computationally generated grammars based on grammatical evolution is the right approach [3]. The initial stage of the new design process is based on machine generations. Then, we would provide a number of factors so that our consumers could create outcomes based on their personal preferences.

We should be aware that discussing computational aesthetics for our fitness function is necessary because the bulk of computationally created outputs won't seem pleasing [4]. Users will be able to modify the results depending on their individual choices because beauty is subjective. It is essential to note that since all of the designs are geometric,

http://dx.doi.org/10.22133/ijwr.2023.401502.1162

Citation. R. Rahnamoun, R. Rahnamoun, " L-Atur Generative Design L-Systems Based Web Application: New Challenges and Novel Application Horizons," *International Journal of Web Research*, vol.6, no.1,pp.39-47 , 2023, doi: http://dx.doi.org/10.22133/ijwr.2023.401502.1162.

Coressponding Author
Article History: Received:15 May 2023; Revised: 23 June 2023 ; Accepted: 10 July 2023.

we can explain aesthetic preferences from a formalistic perspective. There are some calculative qualities existing in the final shapes that can be appealing or unappealing to the majority of our users.

Finally, we did a comparative study of relevant startups and web applications to showcase how our current changes benefits L-Atur compared to currently available tools.

2. Related Works

In the fashion industry, attempts have been made to make computational design physically and three dimensionally replicable [5]. Thanks to the advances in 3D printing and laser cutting, designers are now able to create their designs freely available for public consumption at affordable prices [6].

L-systems, the new type of grammar Lindenmayer proposed in 1968 [7], at first were used to simulate plant growth. This system has attracted computer scientists, who are drawn to it and discovered fresh ways to use computer graphics and realistic plant images [8]. New applications have recently been used in fields like architecture [9], clothing design [5], and agriculture [10].

A major challenge in the use of these methods is to develop a proper grammar for L-systems. Using algorithms to generate grammars, certain researchers try to overcome this problem. In [11], Genetic Algorithms (GA) are used to generate context-free L-systems. Other researchers [12] try to consider L-systems in a competitive environment.

Moreover, shape grammar [13] has many similarities with L-systems. In shape grammar, a set of language rules can be applied to generate shapes, and in this manner, they are derived from Chomsky grammar theory. In a variety of applications, particularly in architectural designs and indoor environments [14], shape grammars are applied.

Grammatical evolution (GE) is a subfield of evolutionary computing that uses context-free grammars to generate strings [3]. GE used to create a grammar from an image [15] or a string. GE is used in many applications. For example, architectural design [16] and fractal design [17]. Finding a fitness function for GE is a crucial work, and the construction of such a challenge is discussed in various applications. For example, sorting networks [18], optimization of structures [19] and architectural design [16].

There have been many researchers who have tried to find a mathematical explanation of the aesthetic parameters of geometric shapes and patterns. Since many of the outputs' visual characteristics are shared with fractals, it's important to mention that many researchers have provided explanations of the beauty of fractal patterns as well [20]. The existing preference for moderately low dimensions of fractal patterns in natural scenes is mentioned by Hagerhall [21]. Birkhoff [4] tried to explain aesthetic value based on the ratio of order and complexity. His description of aesthetics was completely mathematical, unlike the majority of psychological explanations.

3. Methodology

For covering a wide range of products, especially colorful and large-scale complex ones, the previous L-Atur's web application architecture was not appropriate. As a result, a new logical model for L-Atur's web application was prepared, as illustrated in Figure 1. The process is simple for the user without any coding or designing skills; the user just changes the parameters of the grammar. Grammars are generated by Grammatical Evolution (GE). For small-scale design, these generated grammars were used as input in the previous L-Atur architecture; and for large-scale and colorful design, a new model is used for more complex grammars with several rules, in which if the user wants color, a colorful shape generation can also be included.

3.1. L-Systems

L-systems are parallel rewriting rule systems that were developed to simulate plant growth [7], but the basic concept has been extended to a wide range of applications. Traditional grammars use a sequential rewriting method to generate grammar sentences, but L-systems use a parallel approach. This means that at each step of the rewriting process, every grammar symbol will be replaced by rewriting symbols specified in grammar. As a result, L-systems can generate strings that are not producible by any context-free Sequential grammars [2].

L-system grammars are classified into two types: context-free L-systems and context-sensitive L-systems [8]. The rewriting rules in context-free L-systems are independent of the context in which predecessors appear. Symbols will thus be replaced with their predecessors, regardless of their position in the string. Context-free L-systems are used as a base grammar in this paper to generate artifacts. Rewriting rules in Context-sensitive L-systems, on the other hand, are dependent on their predecessors. As a result, a specific symbol or group of symbols may be replaced with different successors, depending on the position of the symbol(s) in the string [23].

One of the most serious practical issues in L-systems is that the generated strings from the grammar are identical. If we generate a forest of plants from a grammar, the plants that result are all identical. Stochastic L-systems add probabilities to the rewriting rules, allowing the system to generate different strings from a given grammar because each rule has a different probability of being executed. As a result, the generated strings will have more variations. Another type of L-system is parameterized L-systems, which use parameters in rewriting rules, which means that each parameter can change the final generated string. Against deterministic grammars, the final string will be completely dynamic [2].

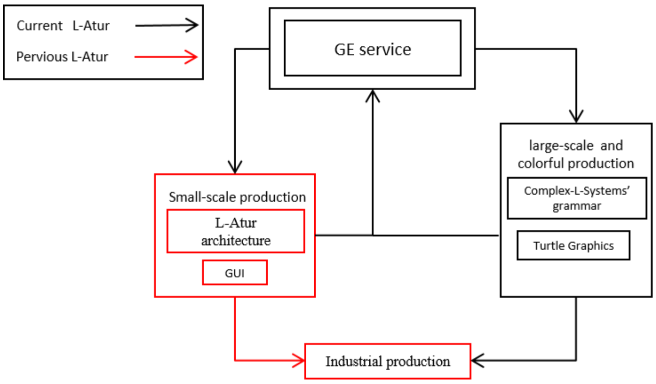

Figure. 1. Logical diagram of the new version of L-Atur's web application

A OL-systems is a triplet G = (∑, γ, P) where ∑ is the alphabets of the system, $\gamma \in \sum^+$ is a nonempty symbol(s) called an axiom and $P \subset \sum \times \sum^+$ is a finite set of rewrite rules. Each rewrite rule represented as $\alpha \rightarrow \beta$ that α is known as the predecessor ($\alpha \in \sum$) and β is also called successor of this rewriting rule ($\beta \in \sum^*$). OL-systems are called deterministic (DOL) if and only if there is exactly one $\beta \in \sum^*$ for each $\alpha \in \sum$ [22].

Consider we are starting from an axiom γ and a rewrite rule $\alpha \rightarrow \beta$, and suppose that γ can be expanded by this rule. So, a new derivation is generated in each parallel iteration, and after n iteration, we have a derivation of length n. This derivation sequence can be represented by:

$$\lambda_1 \Rightarrow \lambda_2 \Rightarrow \cdots \Rightarrow \lambda_n$$

Such that $\lambda_1 = \gamma$.

3.2. Turtle Graphics

As explained in the previous section, L-systems can generate strings of derived length of n. Then we a need a method to convert these strings to a graphic shapes and act as a function from strings to images. Let us consider P is a 2D matrix which representing an image. 3D matrices are used to represent colored images, but only black and white images are used in various applications in the current version of L-Atur. Now we need a function F: S → P that maps the input string S to the output P matrix. Turtle Graphics is just that function [22].

Turtle Graphics work on a 2D Cartesian coordinates and each state of a turtle can be represented by a triplet (x, y, δ). The position of turtle which is determined by (x, y), and angle δ is referred to as its heading and denotes the turtle's facing direction.

The basic commands listed below can draw an image from a starting position of (x_0, y_0, δ_0) if you take into account that l is the step size and θ is the turtle's rotational degree [22]:

- F: Move forward with a step size l, so (x, y, δ) location changed to (x', y', δ'). As a result, a line segment between two mentioned points, is drawn.
- f: Move forward with a step size l without drawing a line
- +: Turn left by angle θ, so the turtle change to state (x, y, δ- θ)
- -: Turn right by angle θ, so the turtle change to state (x, y, δ+ θ).

Let S be a string, (x_0, y_0, δ_0) be the initial position of the turtle and l, θ are constants parameters that described above, Turtle Graphics can generate an image from a string that is generated by the L-systems grammar.

Consider the following simple L-system grammar in Equ (1):

γ : F

P : F → F + F + F F

θ : 60 ° (1)

Figure 2 shows the result for n = 3. What if we used more complex grammar with more than just rewriting rules? Consider the following grammar of Equ (2), which is more complicated than Equ (1):

γ : xyxy

P :{

F → + F F F + F + F + F + F

x→ x F x + F x + F x F y

y→ + F x + F x F y - F y - F y

}

θ : 135 ° (2)

Figure 3 shows the output for n = 3 and obviously it is a more complicated image than Figure 2.

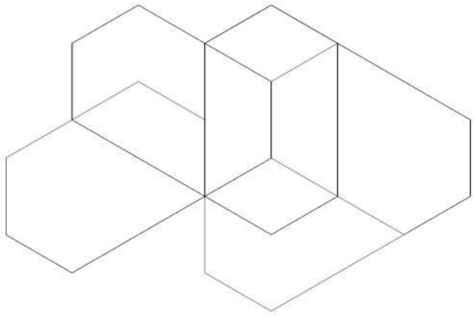

Figure. 2. A simple grammar with one rewrite rule as described in Equ (1)

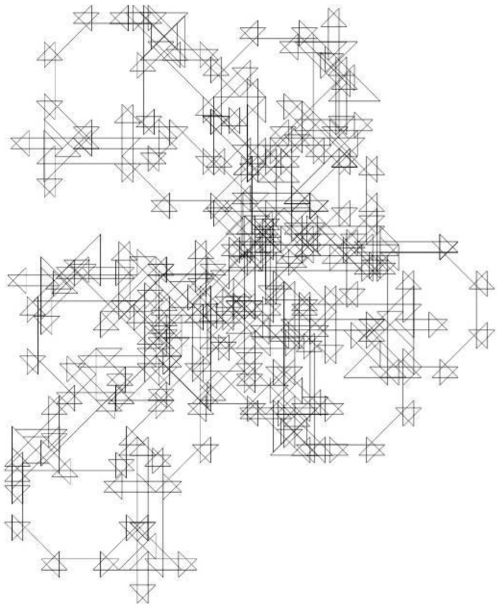

Figure. 3. A complicated grammar with multiple rewrite rules as described in Equ (2)

3.3. Evolutionary Computation

Evolutionary Computation (EC) inspired from search in biological evolution. Based on the concept of Darwinian natural selection, these group of algorithms try to simulate evolution in computers. In EC, algorithm begin by generating random population of individuals. In each iteration of the algorithm, selection mechanism choose fittest individuals and use them to reproduction process. In reproduction phase, genetic operators such as crossover and mutation is used to create new offspring. This process continues until a termination condition is satisfied [24].

EC is a broad field and contained different subfields. The most important of this subfields is Genetic Algorithms (GA) [25]. GA is used in wide range of applications from engineering to finance and specially applied in optimization problems. Evolutionary Strategies is also another famous subfield of EC for designing optimization algorithms. Moreover, we can mention Swarm Intelligence and Artificial Life in this area.

In EC, genotype and phenotype are essential concepts to understand this process. Genotype can be represented by a string especially in binary form, but phenotype is a representation of the genotype in real world. So we need a mapping function from coded genotype to a real sample in example as phenotype. Genotype is expressed by a general string form and can be used in wide range of applications, but a mapping function is essential to convert it to real sample in specific application.

3.4. Grammatical Evolution

Genetic programming (GP) is a subfield of evolutionary computing designed to generate programs. Automatic programming (AP) has a long history, from GP and inductive logic programming to advanced deep learning methods. The ultimate goal of AP is to enable programmers to write their code in a high-level abstract form. GP starts with a population of parse trees created from the grammar of the target language. Based on an evolutionary paradigm, this population evolves through generation by genetic operators (mutation and crossover).

Grammatical Evolution (GE) is a type of GP that uses a gene as a string concept. In GE, a genotype of string is randomly generated to make the initial population. Genotype-phenotype mapping occurs based on a Backus–Naur Form or Backus Normal Form (BNF) notation in grammar which is translated by bit-string genotype to the phenotype described by the grammar. Like other evolutionary algorithms, the search continues from the current generation to the next generation.

3.5. A Review on L-Atur Architecture

L-Atur's main goal is to produce designs for customers at reasonable prices. Users can work with the web application to select the design of their products. The primary goal of this start-up is to facilitate interaction between customers and the system.

This startup’s web application uses L-systems to generate strings, which are then transformed into pictures by turtle graphics. L-Atur has a set of parameters with which users can interact to find the final proposed design.

Finally, when the user has finished interacting with the system and has found the final design, this high-level L-system code is sent to the server side, where it is converted to a low-level code via a sophisticated process, and this is the input of the third phase.

This low-level code will be prepared for use as input for the laser cutting machine during the third phase. The user design will be converted into a physical accessory during the cutting process. The finished physical product will be given to the customers at the end of this process. Figure 4 describes the L-Atur architecture in detail, and [1] discusses implementation specifics.

3.6. Complex L-Systems’ Grammar

The earlier L-Atur approach was best suited for small-scale designs, particularly accessories, which were the focus of our main production. The potential of L-systems helps us imagine a wider range of design applications for L-Atur. It fundamentally wasn't a good fit for our strategy of pairing human and computer cooperation, so we had to make some adjustments to our web-based application. We discovered that while the current approach creates appropriate designs for small-scale and fragile objects, it falls short in other contexts, such as large-scale designs. Of course, there were certain technological constraints on producing new goods that needed to be taken into account as well.

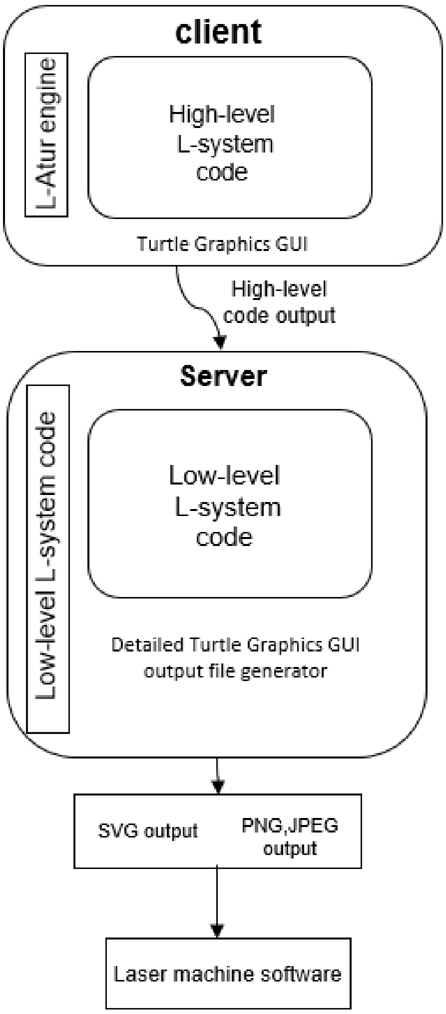

Figure. 4. General schema of L-Atur

We must keep in mind that the majority of our customers lack design skill, thus it is crucial that we take into account our primary motivation to increase the likelihood of visually pleasing designs while maintaining high levels of customizability. We need a more advanced version of the L-Systems since the created goods are not only aesthetic items; their functionality in real life is also crucial.

For a better understanding, pay attention to the grammars mentioned in Equ (3) and Equ (4):

γ : f+f+f+f

P : f → ff+f+f+f+f+ff

θ : 30° (3)

γ : [f]--[f]--[f]-[f]+ff

P : f → <{{{fff}}[f][--f][+fff]

θ : 12° (4)

Figure 5 shows the result of grammar in Equ (3) with n = 5 and Figure 6 shows Equ (4) with n = 5 but in a colorful example. L-systems grammar in Equ (4) uses the "<" symbol as a decrement of color. For saving state (position and direction), "["is used and"]" restores the state. Now, consider the following more complex grammar:

γ : xyyxyx+xyxyxyx

P :{

x→ fx+fx+fxfy-fy+fy+fy

y→ +f+fxfy-fy-fyx

f → f+f+f

}

θ : 120° (5)

Figure 7 shows Equ (5) with n = 5. However, to manufacture Figure 7, the complexity of the example makes it impossible for creation, and the design won't turn out appealing for a small-scale object. Although if we have any desire to reproduce the referred model as a wall decoration, it can turn out intriguing for visual investigation.

From a technical standpoint, our previous designs were based on L-systems, which were written with only one rule (grammars in Equ (3) and (4)). With a more extensive scope of utilizations, we should grow the quantity of rules to make a more prominent possibility of visual intricacy (grammar in Equ (5)). We should provide separated sections for various products because they can bring out a variety of visual complexities and technical limitations. Improving the user experience is a crucial component, and if we ignore such essential features, it can lead to their frustration.

As part of our large-scale designs, we are currently concentrating on the production of wall decorations, particularly decorative mirrors Figure 8. We faced some technical limitations in the laser cutting process, and since we want to offer affordable prices, we should have provided a suitable design solution for the available scale of sheets for our current market place. Extremely large-scale wall decors can be offered as multiple panels in the developed version of L-Atur inspired by traditional Persian tile making.

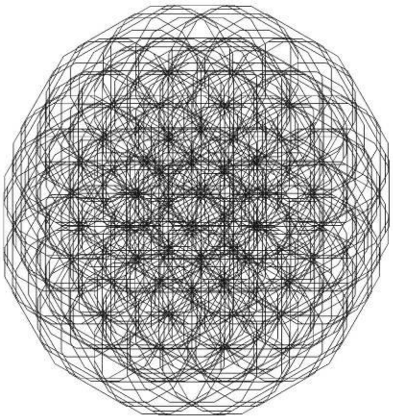

Figure. 5. A circle of circles grammar according to Equ (3) with n=5

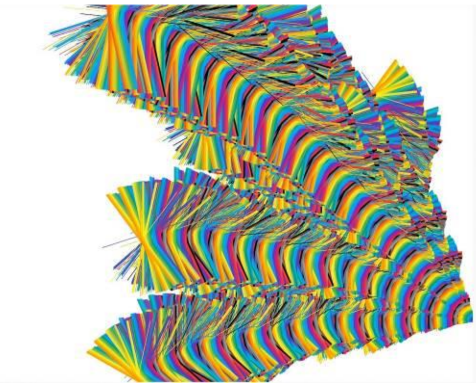

Figure. 6. An implementaion grammar in Equ (4) and n=5

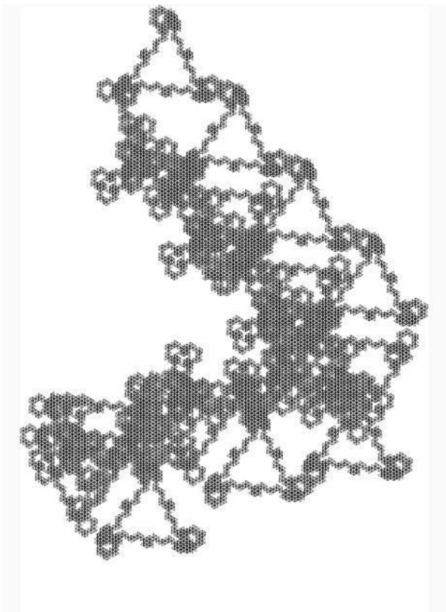

Figure. 7. A complex grammar according to Equ (5), n=5

Figure. 8. A decorative mirror wall instalation generated by L-Atur's web application

We noticed that we can choose from a wide range of colors and materials when it comes to laser cutting fabrics. Additionally, all of our options are offered at an affordable price. Table mats and runners provide us with a chance to offer moderately complex L-system-based designs. Textile design is also another option that we want to include in the next versions. Considering different color choices in L-systems enables us to provide interesting, colorful prints that have wide usage in the fashion industry, as shown in Figure 9.

Figure. 9. A prototype of a textile design generated with L-systems

4. Computational Aesthetics of L-Atur

As it was explained in section 5, we need to generate grammars with considerable amount of variety so that an evolutionary method (grammatical evolution) is needed. Grammatical evolution operates based on fitness function. There are two strategies to make fitness function possible; one relies on computer and human interaction to finally reach some thresholds of what is deemed aesthetically beautiful depending on user reactions. We evaluated this approach on a few candidates, but we found that it won't be useful for our purpose because it demands extreme patience from users. The alternative method, which brings us to machine aesthetics, is to provide certain aesthetic metrics to enable the machine to generate visually pleasing grammars.

It's essential to define calculative aesthetic preferences to provide a proper fitness function. A crucial component for us is to define the most significant characteristics of L-systems that affect the aesthetic preferences of our users and alter their judgments. Of course, it's challenging to find a clear definition of what qualities make a shape pleasant because of diverse individual definitions of beauty, which can be varied among different age groups, personal experiences, etc., but it's possible to find some characteristics that are considered by the majority of users to be aesthetically pleasing or unappealing.

Considering that L-systems are mainly associated with plant growth and branching modeling, the existing qualities are rooted in nature, so obviously that's considered pleasant for humans. That's why we can witness a wide usage of L-systems in design and architecture because it brings such visual fascination, especially if we want to put terms like biomimetic design under the spotlight.

Geometric shapes generated by L-systems have some mathematical qualities that can be helpful when it comes to defining beauty from a mathematical perspective. L-systems possess order and complexity as two considerable qualities, and some researchers have tried to offer a mathematical explanation of beauty based on these features. Birkhoff [4] took a mathematical approach to aesthetics by formalizing aesthetic qualities. In his book [4], he considers order and complexity as two main features to achieve an aesthetic measure. M, which is a ratio between the shape's order and complexity: M= f (OC), and f stands for the function of this ratio. There are different definitions of O (order) and C (complexity) in various classes of objects, and their aesthetic measure was calculated by Birkhoff for a wide range of art objects. Berlyne [26] tried to achieve such a goal by considering patterns instead of polygons. He defined physical complexity, which affects the numerosity of pattern content like the amount of material, the number of units, and the symmetry of the pattern.

5. Extended Fields in the Current Version

In the current version of L-Atur, we have expanded the supported grammatical rules, making it much more suitable for different products. Considering various material properties and applications, it was essential for us to provide a wider range of supported grammatical rules to generate shapes with extreme visual diversity. We have identified some fundamental elements that we decided to incorporate into our current version. These key grammatical properties include variable line thickness, color, and an expanded set of rules for increased visual complexity. Combining these options opens up a whole new realm of design possibilities. In the previous version of L-Atur, we did not consider these fundamental elements since we did not cover such an extensive scope of design.

Color: We have adjusted our current color palette based on the available materials for laser cutting. We offer opaque and transparent plexiglass, MDF, and fabrics, providing a considerable range of colors. Additionally, we support multimaterial designs, allowing users to choose different material properties in various colors. We have also added 2D image printing as a new option. With the expanded shape complexity in our current version, we can create more intricate designs that remain visually captivating as Figure

10. However, in the previous version, our designs were simpler and appeared dull as 2D patterns on flat surfaces Figure 11. Users have the flexibility to select their preferred colors based on their taste and specific preferences, such as key colors for interior design like Figure 12. Furthermore, we offer a random color option for more vibrant results as Figure 13.

Variable line thickness: In the current version, users have greater freedom to adjust line thickness. In the previous version, such an extensive choice of line thickness was not available, and the overall thickness remained the same. As evident in Figure 14, this element can lead to significant changes in the final designs.

Expanded number of rules: Users can now choose from a combination of different grammatical rules to create much more complex designs. In the previous version, which was limited to small-scale designs, we only offered a single rule for product generation. A simple comparison between Figure 11 and Figure 15 reveals the noticeable differences resulting from this expansion.

The combination of all these key elements improves design possibilities to an extraordinary range. Figure 16 showcase designs from the previous version, which clearly cannot match the level of interest achieved by the designs produced in our current version Figure 17 and Figure 18.

Figure. 10. A colorful visually complex notebook cover generated by the current L-Atur version .

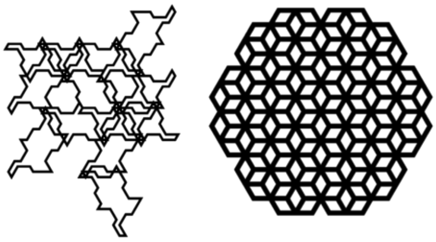

Figure. 11. Exampels of simple designs generated by the previous version and singular grammatical rule .

Figure. 12. A prototype of curtain design with specified color made by the new version

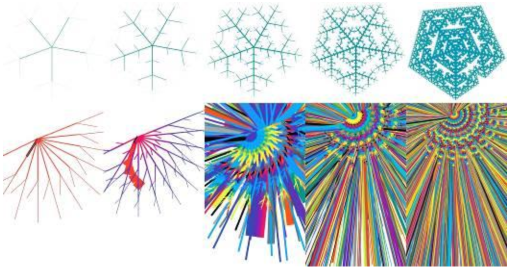

Figure. 13. Complex patterns generated by random color option and specified color choice.

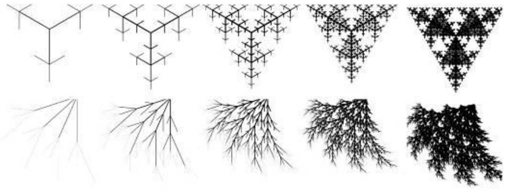

Figure. 14. A step by step shape generation highly affected by variable line thickness option.

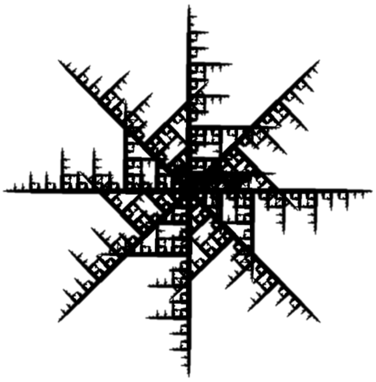

Figure. 15. A shape generated by several grammatical rules .

6. Comparative Study of Results

Because of different methodologies, there are not actually the same web applications or startups to compare with L-Atur. The main reason is that the available tools require design skills, and these softwares follow their designer aspect, which results in similar output shapes with not that much variation. And also, these tools are designed for specific products, like a necklace or bracelet [27, 28], and do not cover a wide range of products. Because they do not use laser-cutting machine technology in production, their products are expensive. The generation of colors not included in all of them. The comparison results are in Table 1.

Figure. 16. Prototypes made by the previous version. As it is obvious such simple shapes are only suitable for small scale designs.

Figure. 17. Mug design made with a combination of current extended options.

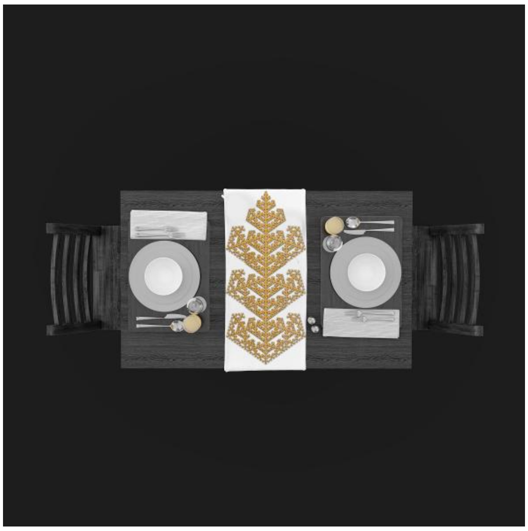

Figure. 18. A medium scale artificial leather table runner made with the current L-Atur.

7. Conclusions and Furthur Research

The practical limits of the present L-Atur are discussed in this paper. Since a wide range of products were intended to be covered, we realized we needed to create a developed version of the present web application. We explained about various adjustments that would provide a less constrained experience for our users. It should be remembered that while our consumers may not be experienced designers, they do have their own subjective preferences. They can experiment with a wide variety of designs thanks to L-Atur while staying away from the majority of the aesthetically unattractive outcomes that can be generated by the machine.

We discussed techniques that provide the items varying degrees of aesthetic complexity dependent on their functioning. Additionally, automated grammar generation would result in an infinite amount of possible visual exploration options. Such approach can make the machine's role more prominent as the designing assistant in startup scene.

A more detailed explanation of machine aesthetics can be really crucial in such applications. If we want to provide proper machine assistants in artistic fields for users who are not experts, we need to define measures of beauty. Generally saying it seems impossible to achieve such machine-based judgments. However, when it comes to geometric shapes, there is still some chance to define what seems appealing to the majority of humans. There are still some mathematical parameters that can clearly display what seems visually pleasing to the majority. But of course, we can't find an exact explanation of what makes a geometric shape beautiful. There are still personal preferences that vary among users, and we cannot ignore the subjective nature of beauty. There must be a chance of adjustment and exploration for the best results possible.

Our next plan is to investigate characteristics of geometric beauty based on different visual traditions. A strong geometric base is noticeable in traditional Persian crafts and architecture. It's possible to reproduce such patterns and objects based on grammar theory by using machines.

Ideals of beauty can be easily explained because existing visuals share the same characteristics. It is noticeable that producing these abstract patterns, known as part of Islamic art, was affected by technical limitations. Due to advancements in technology, we would be able to generate such aesthetic standards in a way that has long been impossible. The human-machine collaboration nature of L-Atur can change the mentioned patterns to a developed version of what they used to be.

Table 1. Comparison of generative design apps

App	*Product Variation*	*Shape Complexity*	*Color*	*Cost*
L-Atur	High	High	Yes	Low
Old L-Atur	Medium	Medium	No	Low
Fusilli	Low	Low	No	High
Radiolaria	Medium	Low	No	High
Hexatope	Low	Low	No	High

Declarations

Funding

This research did not receive any grant from funding agencies in the public, commercial, or non-profit sectors.

Authors' contributions

RaR: Study design, software design and implementation, supervision, acquisition of data, interpretation of the results, drafting the manuscript, revision of the manuscript;

RoR: Study design, interpretation of the results, drafting the manuscript, revision of the manuscript, visualization.

Conflict of interest

The authors declare that no conflicts of interest exist.

References

[1] R. Rahnamoun and R. Rahnamoun, "L-Atur, a Generative Design L-Systems Based Web Application with a Human-Machine Collaboration Approach," *2023 9th International Conference on Web Research (ICWR)*, IEEE, 2023, pp. 156-160. https://doi.org/10.1109/ICWR57742.2023.10139298.

[2] P. Prusinkiewicz and A. Lindenmayer, *The algorithmic beauty of plants*. New York, NY: Springer, 1996.

[3] C. Ryan, M. O'Neill, and J. J. Collins, Eds., *Handbook of grammatical evolution*. Cham, Switzerland: Springer Nature, 2018.

[4] G. D. Birkhoff, *Aesthetic Measure*. Cambridge, MA: Harvard University Press, 1933.

[5] Z. Xu and T. Huwan, "The application of clothing intelligent 3D display with uncertainty models technology in clothing marketing," *Complexity*, vol. 2022, pp. 1–10, 2022. https://doi.org/10.1155/2022/3486100.

[6] J. Louis-Rosenberg and J. Rosenkrantz, "Anti-entrepreneurs using computation to unscale production," *Archit. Des.*, vol. 90, no. 2, pp. 112–119, 2020. https://doi.org/10.1002/ad.2554.

[7] A. Lindenmayer, "Mathematical models for cellular interactions in development. II. Simple and branching filaments with two-sided inputs," *J. Theor. Biol.*, vol. 18, no. 3, pp. 300–315, 1968. https://doi.org/10.1016/0022-5193(68)90079-9.

[8] P. Prusinkiewicz and J. Hanan, Eds., *Lindenmayer Systems, Fractals, and Plants*. New York, NY: Springer, 2014.

[9] P. Muller, P. Wonka, and S. Haegler, *Procedural modeling of buildings"*. ACM SIGGRAPH, pp 614–623, 2006. https://doi.org/10.1145/1179352.1141931.

[10] T. DeJong, "Simulating fruit tree growth, structure, and physiology using L-systems," *Crop Sci.*, vol. 62, no. 6, pp. 2091–2106, 2022. https://doi.org/10.1002/csc2.20847.

[11] J. Bernard and I. McQuillan, "Techniques for inferring context-free Lindenmayer systems with genetic algorithm," *Swarm Evol. Comput.*, vol. 64, pp. 100893, 2021. https://doi.org/10.1016/j.swevo.2021.100893.

[12] J. Talle and J. Kosinka, "Evolving L-Systems in a competitive environment," In *Computer Graphics International Conference*, Cham: Springer International Publishing, pp. 326-350. https://doi.org/10.1007/978-3-030-61864-3_28.

[13] S. Haakonsen, A. Rønnquist, and N. Labonnote, "Fifty years of shape grammars: A systematic mapping of its application in engineering and architecture", *International Journal of Architectural Computing*, vol 21, no 1, pp. 5-22, 2023. https://doi.org/10.1177/14780771221089882.

[14] H. Tran, K. Khoshelham, A. Kealy, and L. Díaz-Vilariño, "Shape grammar approach to 3D modeling of indoor environments using point clouds," *J. Comput. Civ. Eng.*, vol. 33, no. 1, p. 04018055, 2019. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000800.

[15] D. Beaumont and S. Stepney, "Grammatical Evolution of L-systems," in *2009 IEEE Congress on Evolutionary Computation*, IEEE, 2009, pp. 2446-2453. https://doi.org/10.1109/CEC.2009.4983247.

[16] U. M. O'Reilly and M. Hemberg, "Integrating generative growth and evolutionary computation for form exploration," *Genet. Program. Evolvable Mach.*, vol. 8, no. 2, pp. 163–186, 2007. https://doi.org/10.1007/s10710-007-9025-y.

[17] A. Ortega, A. A. Dalhoum, and M. Alfonseca, "Grammatical evolution to design fractal curves with a given dimension," *IBM J. Res. Dev.*, vol. 47, no. 4, pp. 483–493, 2003. https://doi.org/ 10.1147/rd.474.0483.

[18] M. Bidlo and M. Dobes, "Evolutionary development of growing generic sorting networks by means of rewriting systems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 232–244, 2019. https://doi.org/10.1109/TEVC.2019.2918212.

[19] B. R. Bielefeldt, G. W. Reich, P. S. Beran, and D. J. Hartl, "Development and validation of a genetic L-System programming framework for topology optimization of multifunctional structures," *Comput. Struct.*, vol. 218, pp. 152–169, 2019. https://doi.org/10.1016/j.compstruc.2019.02.005.

[20] R. P. Taylor and J. C. Sprott, "Biophilic fractals and the visual journey of organic screen-savers," *Nonlinear Dynamics Psychol. Life Sci.*, vol. 12, no. 1, pp. 117–129, 2008. https://sprott.physics.wisc.edu/pubs/paper311.pdf.

[21] C. M. Hagerhall and T. Purcell, R. Taylor, "Fractal dimension of landscape silhouette outlines as a predictor of landscape preference," *Journal of Environmental Psychology*, vol. 24, no. 2, pp. 247–255, 2004. https://doi.org/10.1016/j.jenvp.2003.12.004.

[22] P. Prusinkiewicz, "Graphical applications of L−systems," in *Proceedings of Graphics Interface*, 1986, vol. 86, no. 86, pp. 247-253. http://algorithmicbotany.org/papers/graphical.gi86.pdf.

[23] R. J. Morris, Ed., *Mathematical modelling in plant biology*, 1st ed. Cham, Switzerland: Springer International Publishing, 2018. https://doi.org/10.1007/978-3-319-99070-5.

[24] K. A. de Jong, *Evolutionary computation: A unified approach*. Mit Press, 2006.

[25] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimed. Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, 2021. https://doi.org/10.1007/s11042-020-10139-6

[26] D. E. Berlyne and J. B. Crozier, "Effects of complexity and prechoice stimulation on exploratory choice," *Percept. Psychophys.*, vol. 10, no. 4, pp. 242–246, 1971. https://doi.org/10.3758/BF03212813.

[27] L. Trautmann, "Product customization and generative design," *Multidiszciplináris tudományok*, vol. 11, no. 4, pp. 87–95, 2021, https://doi.org/10.35925/j.multi.2021.4.10.

[28] "Hexatope | Design your own unique jewellery," *hexatope.io*. https://hexatope.io/ (accessed Oct. 11, 2023).

Rashin Rahnamoun is currently studying bachelor of computer engineering at the faculty of computer science and engineering, Shahid Beheshti University, Tehran, Iran. Her research interests include complex networks and business analytics.

Romina Rahnamoun is currently studying bachelor of fine art (painting) at the faculty of visual arts, University of Art, Tehran, Iran. Her research interests include computational aesthetics, machine creativity and generative art.