RULE-BASED GENERATION OF INTERWOVEN ASSEMBLIES IN ARCHITECTURAL DESIGN: A COMPUTATIONAL APPROACH INTEGRATING ATTRIBUTE GRAMMAR

KEVIN HARSONO[1], SHEN-GUAN SHIH[2], YEYINT AUNG[3], FELICIA WAGIRI[4] and TSUNG-WEI CHENG[5]
[1,2,3,4,5]*Department of Architecture, National Taiwan University of Science and Technology, Taipei, Taiwan.*
[1]*d11113803@mail.ntust.edu.tw, 0000-0003-1336-3380*
[2]*sgshih@mail.ntust.edu.tw, 0000-0001-7108-9960*
[3]*d11013803@mail.ntust.edu.tw, 0009-0008-3716-1891*
[4]*felicia_wagiri@hotmail.com, 0000-0001-8414-2599*
[5]*mike861104@gmail.com, 0009-0001-6620-2073*

Abstract. This paper explores the potential of procedural modelling in generating design and assembly sequences for interwoven panel systems. The primary objective is to establish a computational framework that enables architects and designers to conceptualize and construct interwoven assemblies through rule-based generation, employing attribute grammar. The research delves into how procedural modelling, can effectively generate design and assembly sequences for interwoven panels in architectural design. The methodology unfolds in two parts: firstly, the framework for generative design for individual interwoven panel; and secondly, the definition of production rules for the assemblies, accompanied by practical applications of attribute grammar. The results indicate that using attribute grammar is an effective way to handle rule-based generated assemblies, incorporating attributes as semantics to manage transformations and block placement. This approach is expected to aid architects and designers by providing a robust computational toolkit for the creation of rule-based generation and assembly of interwoven panels. This research is expected to contribute to enhancing the development of complex geometric designs within architectural practice.

Keywords. Interwoven Structures, Procedural Modelling, Architecture Design, Computational-aided Design Tools, Attribute Grammar.

1. Introduction

Complex geometric structures, including interwoven designs, have undergone significant development over the years. However, their full potential in architectural design remains largely unexplored. In architectural design, interwoven assemblies involve the intricate weaving of various elements within buildings. Interwoven

ACCELERATED DESIGN, Proceedings of the 29th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2024, Volume 1, 363-372.

assemblies are a type of interlocking system of blocks that enhance structural stability. These components also promote modular design, where rule-based generation can effectively represent their configuration and topological relationships. The ability to create panels from these interwoven assemblies allows designers to engage in procedural design. Despite dedicated research efforts on interwoven assemblies (Casado et al., 2021; Muslimin, 2010, 2011), architects have predominantly drawn inspiration from manual techniques and materials rooted in traditional basketry to craft elements such as furniture, facades, walls, and roofs. However, the widespread application of interwoven principles remains primarily within traditional practices. In the field of computer science, there has been extensive research on rule-based generation based on grammar techniques. However, there is a notable absence of systematic discussion in architectural research regarding the systematic and automated generation of designs and assembly sequences in real-world construction, particularly in the context of interwoven systems.

This research proposes a systematic framework for generating interwoven assemblies. The study is structured into two main parts. The first part focuses on the generation and design of the interwoven panel. The goal is to create a set of simplified blocks that can serve as components in more intricate assemblies within the interwoven assemblies. This phase involves the initial generation of foundational building blocks for the interwoven assemblies. In the second part, the research will extend to generating rules and grammar for the blocks created in the first phase. Additionally, it defines rules for hierarchical assemblies, providing a structured organization for assemblies based on different levels. This step aims to establish a systematic set of rules and grammar governing the construction of both individual blocks and their assembly into more complex structures.

In this paper, a computational framework designed for creating and assembling interwoven masonry systems is proposed. This study attempts to maximize the possibilities of interwoven masonry systems in architectural design, with a focus on their integration into the architectural building component. This work aims to revolutionize how architects and designers approach the concept of interwoven assemblies by merging computational design tools, procedural methodology, and utilization of attribute grammar and parsing techniques.

2. Interwoven in Architecture Realm

Previous research (Muslimin, 2010) has asserted that employing an interwoven configuration plays a crucial role in balancing stress factors like tension, bending, and shear on surfaces and cells. Moreover, the interweaving of neighbouring cells in both axes within the interwoven configuration serves to prevent yarn deformation caused by vertical loads. This underscores the significance of the interwoven concept not only as a cultural and artistic expression but also as a practical and functional solution in addressing structural stresses.

Transitioning into architecture, the term "interwoven" assumes a specific design connotation. In architectural contexts, it refers to a technique that intricately intertwines or locks different elements to shape a unified and visually compelling structure, exemplified by structures like the Imai Hospital Daycare Center in Japan designed by Shigeru Ban. Upon deeper examination, the significant characteristic of interwoven

assemblies lies in their repetition, allowing for modularization into smaller, more manageable parts. This inherent quality facilitates procedural design and prompts consideration of the rules and procedures involved in generating and defining these complex structures.

3. Interwoven Block Design

To prepare the design of the interwoven building block, a comprehensive understanding of its components is imperative. As clarified in Section 2, the interwoven assemblies can be derived from smaller modular panels, allowing for the dissection of each part based on its function. Through the literature review and analysis of case studies, it becomes obvious that developing a modularized interwoven structure for architectural building components involves the integration of two fundamental elements. The first element, the horizontal component, is embodied by sturdy columns that furnish structural support and stability. On the other hand, the vertical component comprises interwoven panels strategically integrated with the columns. This not only enhances aesthetic appeal but also provides functional advantages such as environmental control or privacy. The right side of Figure 1 illustrates the initial prototyping for interwoven assemblies.

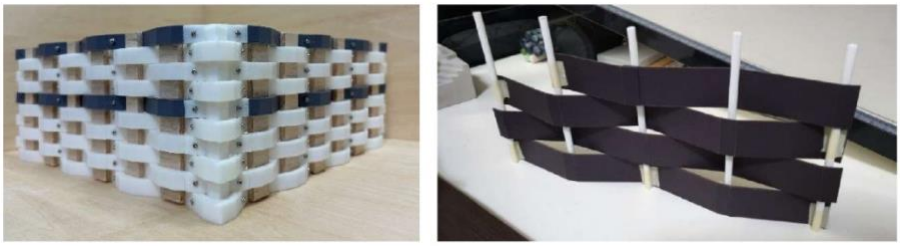

Figure 1. Left: The 3D printed model of the interwoven panel. Right: Prototype of interwoven block.

From this experiment, we establish a design framework for producing interwoven panels. The parameters involved include the profile of the weaving section and the distance between columns. The weaving angle is dependent upon the distance between the columns, while the profile dictates the height of the column panels. In this case, the formation of modular interwoven panels produces five distinct types, each assigned a specific function. Broadly categorized, these panels can be classified into two main components, shown in Figure 2, which are weaving and column panels. The weaving part further diverges into two categories. First, there is the Straight Weave (SW), shown in red colour, regarding panels characterized by a straightforward weaving pattern. Second, the corner weave category introduces two types: Inner Corner (IC), shown in green colour, indicating panels weaving through the inner part of the column, and Outer Corner (OC), shown in blue colour, signifying panels weaving through the outer part of the column. Parallelly, the column part presents two primary types: Corner Column (CC), shown in white colour, representing columns with a turning configuration, and Straight Column (SC), shown in yellow colour, where panels exhibit a straightforward column configuration.

The parameters for designing the interwoven assemblies include the distance

between the vertical part and the profile of the weaving part. The weaving part's profile is rectangular, measuring 8x12 cm, with a column distance of 48 cm. The column height is maintained at twice the profile height for modularity, specifically 24 cm. The interlock between panels is meticulously designed to preserve the connection position, ensuring the maintenance of XY displacement. To establish connections between panels, a bolt join method is employed.

The modular approach guarantees the generation of a varied collection of panels, each customized for specific functionalities. This versatility is essential in constructing interwoven structures, facilitating nuanced and purposeful designs. To validate the functionality of the assemblies, a scaled model is presented using 3D printing, as illustrated on the left side of Figure 1.

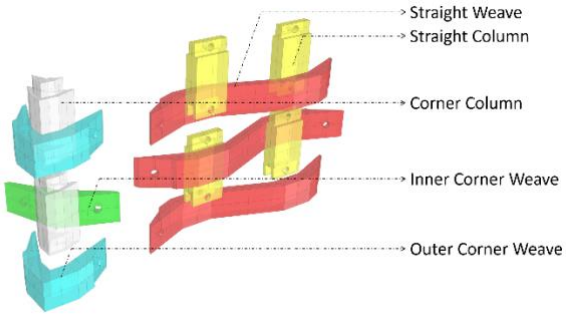

Figure 2. Block design and assemblies.

4. Grammar and Rule-Based Generation

The concept of incorporating grammar into design draws parallels between visual and natural languages. In natural language, grammar plays a vital role in facilitating communication by providing a structured framework for organizing and conveying meaning. Designers can represent complex visual information with the assistance of formal grammar defined by specific rules (Bruton, 1997). Rule-based generation involves computational tools and a predefined set of rules to generate architectural designs. These rules, affiliated with grammar in computer programming languages, provide a structured framework for design creation. By defining rules, designers can systematically generate designs that adhere to predetermined standards. Grammar serves as a systematic set of principles, functioning as an architect's toolkit to articulate the syntax and structure of the design language.

In this context, Context-Free Grammar (CFG) plays a prevalent role in rule-based generation research due to its simplicity and efficiency in interpretation for parsing and language generation. CFG necessitates being context-free and composed of a single non-terminal (Bikaun et al., 2022). However, CFG can only deal with strings of grammar rules, but not necessarily executable. Addressing the challenges posed by CFG, Attribute Grammar (AG) serves as an extension to CFG. By assigning attributes to symbols and defining how to compute these attributes, attribute grammar incorporates additional information (Prasad, 2009) into more meaningful and executable grammar.

Ultimately, to translate grammar into semantics, parsing is essential. This implies that to convert syntactic rules into meaningful and executable assembly sequences, the input must undergo analysis and interpretation by a parser. In architectural design, parsing involves dissecting input into components such as nouns (representing objects), verbs (depicting methods), and associated attributes or options. Guided by established grammar, this process takes a token string as input and transforms it into the corresponding representation (Teboul, 2021). In this study, the SLR parser is utilized. The parser ensures adherence to language grammar rules, thereby providing a logical and organized framework for the design investigation.

5. Attribute Grammar for interwoven Assemblies

In this study, the attribute grammar (AG) for a syntax-directed definition (SDD) is denoted as AG with a triple (G, A, AR), where G represents the context-free grammar for the language, A is associated with each grammar having attributes, and AR is semantic linked with production rules. Specifically, for interwoven assemblies, Token$\langle s, TS(s)\rangle$ and Token$\langle a, ID(a)\rangle$ are addressed to signify the terminal symbols with transformation action and the semantic of block placing action, respectively.

Take the production rule $X \rightarrow s\ a\ s\ a\ s\ a$ as an example; it implies moving straight and placing the block three times. Here, s represents TS or transformation of the block, and a signifies the $ID(a)$, the semantic aspect, indicating the placement. Since CFG alone cannot handle the semantics, the property $ID()$ is treated as an attribute inside the production rule in attribute grammar. Thus, the appropriate representation of the grammar with syntax and semantics is:

$$X ::= s\ s\ s\ \{X.val = TS(s)\ ID(a)\ TS(s)\ ID(a)\ TS(s)\ ID(a)\} \quad (1)$$

In this context, there are two primary rules governing interwoven assemblies: S, utilized for straight assemblies, and T, employed for turning configurations. Bellow CFG and AG grammar rule set S' show a language pattern with translation $TS(t)$, rotation $TS(u), TS(v)$ and placing block $ID(a)$:

$$\begin{aligned}
S' &::= S && \{S'.val = S.val\} \\
S &::= E\ S \mid \epsilon && \{S.Val = E.Val\ S.Val \mid ID(a)\} \\
E &::= S \mid T && \{E.Val = S.Val \mid T.Val\} \\
S &::= t && \{S.val = ID(a)\ TS(t)\} \\
T &::= U \mid V && \{T.Val = U.Val \mid V.Val\} \\
U &::= u && \{U.val = TS(u)\ ID(a)\ TS(t)\} \\
V &::= v && \{V.val = TS(v)\ ID(a)\ TS(t)\}
\end{aligned} \quad (2)$$

To establish the production rule of the interwoven assembly system, it is crucial to define the transformation between blocks. The terminal $\langle t, TS(t)\rangle$is made to define the transformation of the blocks. This step is essential because the grammar can only be formulated when the assemblies are explained in a geometric manner. The configurations of these interwoven structures find geometric explanations, particularly concerning rotational and translational transformations.

The designation of the terminal $\langle a, ID(a)\rangle$ acts as a semantic descriptor for the interwoven panel block placement. This flexibility allows for dynamic adjustments to the block based on predefined geometries or groups of geometries. The initial configuration of interwoven block IDs introduces two distinct types: $\langle corner\rangle$ and $\langle straight\rangle$, as shown in Figure 3. The $\langle straight\rangle$ type functions as a straight arrangement, incorporating 4 types of components: 2 panels of Straight Weave (SW) and 2 panels of Straight Column (SC). On the other hand, the $\langle corner\rangle$ type serves as a corner or turn function, comprising 3 types of components: Inner Turn Weave (IC), Outer Turn Weave (OC), and Corner Column (CC). Here, the height of the assemblies is determined by the H, as a branch of the sentence.

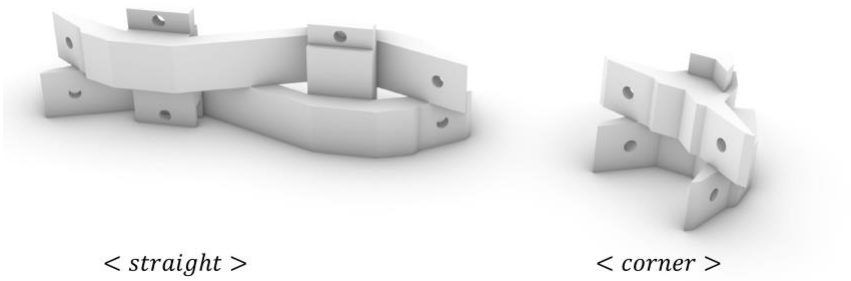

Figure 3. Block arrangement.

To address branching, L-systems are employed to manage the hierarchy of the sentence structure. In L-systems, branching is achieved by using the push ('[') operation and the pop (']') operation following the rule "first in last out", storing an action onto the top of a stack and retrieving it out of the stack, respectively. The concept was derived from turtle graphics and push-down automata (Prusinkiewiczy et al., 1996; Sipser, 2012). For example, if and only if n is a natural number, it could represent the height is H' token concatenating for n times and ID(i) places a block, the appropriate way to express this in production rules is:

$$A ::= [\, n\, H'\,] \quad \{A.Val = H'.Val^{N(n)}\}$$
$$H' ::= h \qquad \{H' = TS(h)\, ID(i)\} \tag{3}$$

The transformation of this interwoven configuration will have three distinct forms of a natural number n with the non-terminal token in the right of rules: h, s, and t. Each attachment type serves a specific purpose, h is employed for the height, which means it will determine the number of interwoven blocks placed vertically. This s is employed for straight assemblies, and t represents turn actions. The transformation associated with these terminals is illustrated in Figure 5. Each type corresponds to a specific block type, along with a corresponding transformation, which is rotation $R_{(x,y,z)}$ and translation $T_{(x,y,z)}$ as proposed by previous research (Shih, 2018).

The rule regulating the interwoven panels demands that the turn is executed by flipping it. However, prescribing a rule to be flipped can introduce ambiguity and inefficiency. To address this challenge, the rule set for this interwoven panel is streamlined to only two options: go straight $\langle S\rangle$ and turn $\langle T\rangle$. To ascertain the direction,

the parser matches terminal patterns with the transformation matrix derived from the segments of the input polyline, determining whether it aligns with the given token. To obtain the semantic patterns for matching the height configuration, the attributes are modified with $[\langle H\rangle.Val^{N(n)}]$. Figure 4 illustrates the transformation matrix for each $TS()$. Additionally, the $\langle T\rangle$ offers two choices depending on the direction, whether it involves a left turn or a right turn, this means the $\langle T\rangle$ can be either $\langle U\rangle$ or $\langle V\rangle$, corresponding to the matching direction transforming actions. This approach enhances clarity and efficiency in the rule-based generation of interwoven structures.

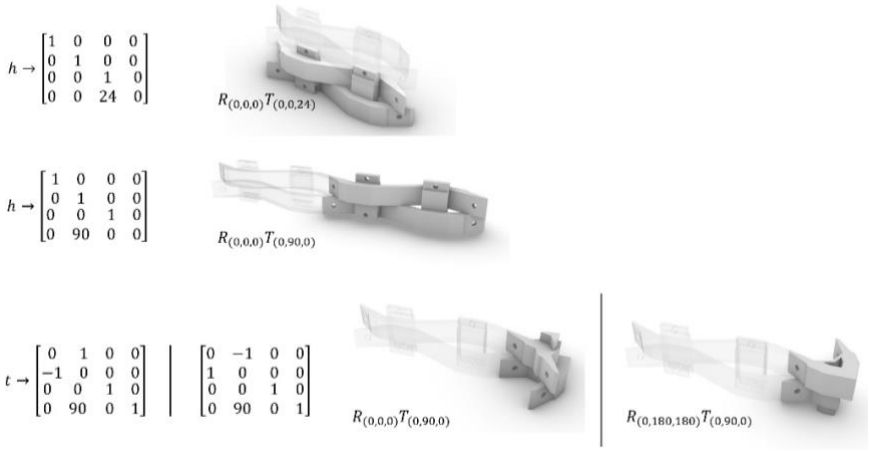

Figure 4. The transformation matrix for the non-terminal of the interwoven panel.

Therefore, after defining the terminal tokens for transformation $TS()$ and the $ID()$ of each block, the production rules can be specified. The syntax and semantic structure of the production rules are represented as follows:

$$
\begin{aligned}
&\langle Main\rangle ::= \langle Stmt\rangle && \{\langle Main\rangle.Val = \langle Stmt\rangle.Val\}\\
&\langle Stmt\rangle ::= \langle E\rangle\langle Stmt\rangle \mid \epsilon && \{\langle Stmt\rangle.Val = \langle E\rangle.Val\ \langle Stmt\rangle.Val \mid \epsilon\}\\
&\langle E\rangle ::= \langle corner\rangle \mid \langle straigth\rangle && \{\langle E\rangle.Val = \langle corner\rangle.Val \mid \langle straigth\rangle.Val\}\\
&\langle straight\rangle ::= \langle S\rangle && \{\langle straight\rangle.Val = \left[\langle H\rangle.Val^{N(n)}\right]\langle S\rangle.Val\}\\
&\langle corner\rangle ::= \langle T\rangle && \{\langle straight\rangle.Val = \left[\langle H\rangle.Val^{N(n)}\right]\langle S\rangle.Val\}\\
&\langle H\rangle ::= h && \{\langle H\rangle.Val = TS(h)ID(i)\}\\
&\langle S\rangle ::= s \mid \epsilon && \{\langle S\rangle.Val = ID(i)TS(s) \mid ID(i)\}\\
&\langle T\rangle ::= \langle U\rangle \mid \langle V\rangle && \{\langle T\rangle.Val = \langle U\rangle.Val \mid \langle U\rangle.Val\}\\
&\langle U\rangle ::= u && \{\langle U\rangle.Val = TS(u)ID(i)TS(s)\}\\
&\langle V\rangle ::= v && \{\langle V\rangle.Val = TS(v)ID(i)TS(s)\}
\end{aligned}
\tag{4}
$$

To employ attribute grammar in a parser, it's necessary to build a parse tree. This tree is subsequently navigated in a specific sequence through Syntax-Directed

Translation (SDT). The creation of the parse tree relies on the token string and the production rules outlined in the preceding section, illustrated in Figure 5. At every node of the parse tree, the SDT rules is implemented, utilizing attributes linked to the grammar symbols to calculate semantic information (Aho et al., 2007).

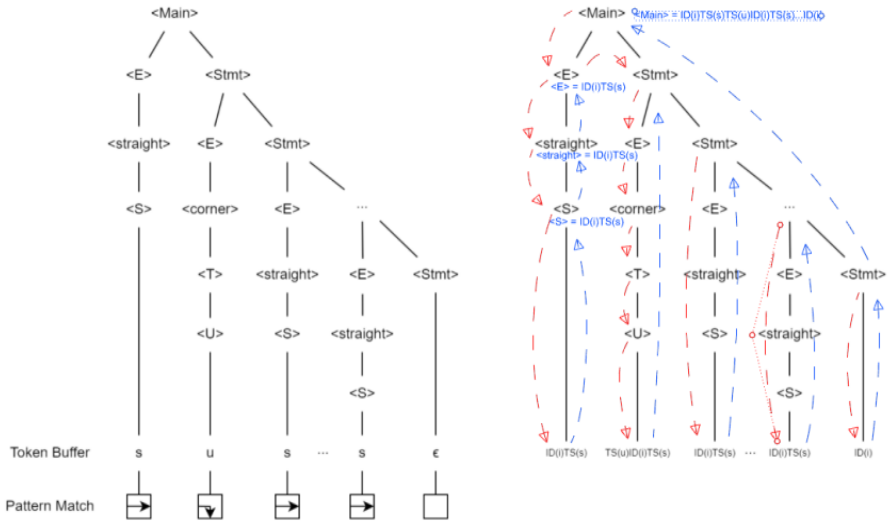

Figure 5. Left: Parse tree. Right: SDT.

6. Application of Grammar for Interwoven Assemblies

In this section, the practical application of grammar and parsing is demonstrated. The initial step involves inputting data, represented by curves in this case, into the parser as a token string. This input curve can represent a building plan, which architects can utilize for designing an interwoven panel wall.

Expanding on the rules delineated in the prior section for interwoven assemblies, the subsequent phase entails parsing the curve into interwoven assemblies. In this scenario, the height is configured to be H concatenating 8 times, approximately 2 meters. By defining H, we address the branching issue in the production rule. The applied grammar in this context can be observed as follows:

$$
\begin{aligned}
&\langle straight\rangle ::= \langle S\rangle && \{\langle straight\rangle.val = [\langle H'\rangle.Val^8]\,\langle S'\rangle.Val\,\}\\
&\langle corner\rangle ::= \langle T\rangle && \{\langle corner\rangle.val = [\langle H'\rangle.Val^8]\,\langle T'\rangle.Val\,\}\\
&\langle H'\rangle ::= h && \{\langle H'\rangle.val = TS(h)\;ID(i)\}\\
&\langle S\rangle ::= s \mid \epsilon && \{\langle S\rangle.val = ID(i)\;TS(s) \mid ID(i)\}\\
&\langle T\rangle ::= \langle U\rangle \mid \langle V\rangle && \{\langle T\rangle.Val = \langle U\rangle.Val \mid \langle U\rangle.Val\}\\
&\langle U\rangle ::= u && \{\langle U\rangle.Val = TS(u)ID(i)\;TS(s)\}\\
&\langle V\rangle ::= v && \{\langle V\rangle.Val = TS(v)ID(i)\;TS(s)\}
\end{aligned}
\tag{5}
$$

Figure 6 shows the application for the input curve. The curves serve as tokens for

parsing the grammar. Consequently, adjustments are made to the curve to ensure the proper spacing between blocks. The original curve, highlighted in black, is modified to align with the block's dimensions. The resulting adjusted curve, shown in green, emerges as a product of this process.

From the token buffer, the initial string is derived as

$$Token\ buffer\ \rightarrow\ \#s^4\ u\ s^{23}\ u\ s^4 u\ s^{12} v s^{12} \epsilon \#$$

Applying the production rule to the string results in

$$\langle Main \rangle ::= 4\ \langle straight \rangle\ \langle corner \rangle\ 23\ \langle straight \rangle\ \langle corner \rangle\ 4 \langle straight \rangle$$
$$\langle corner \rangle 12\ \langle stright \rangle \langle corner \rangle 12\ \langle straight \rangle\ \epsilon$$

Therefore, after traversing the syntax tree, we can gain the semantic:

$$\{\langle Main \rangle . Val = ([\ (TS(h)\ \ ID(i))^8]\ \ ID(i)\ \ TS(s))^4 \ldots ID(i)\} \quad (6)$$

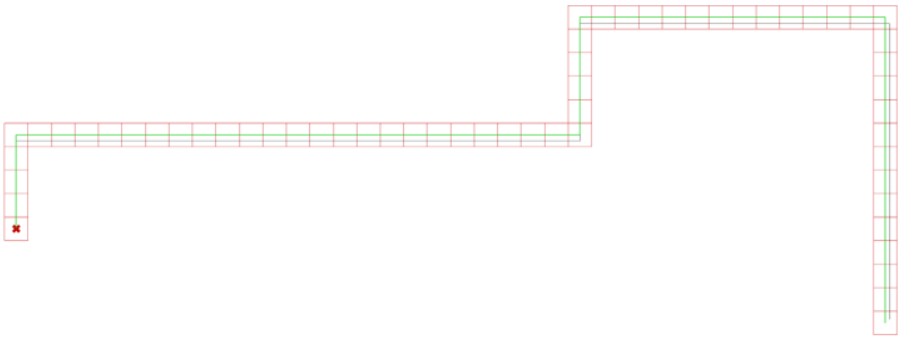

Figure 6. Curve adjustment.

This production rule is then incorporated into the parse tree, and the result from SDT is depicted in Figure 7, showcasing the interwoven panel wall.

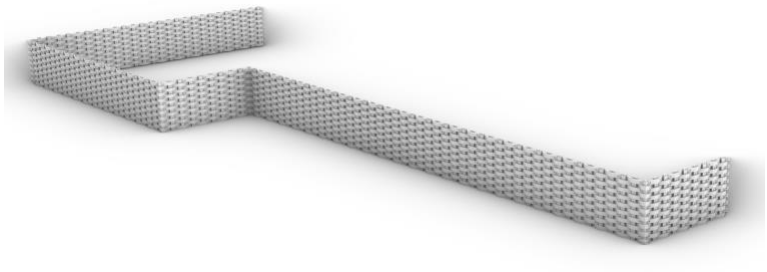

Figure 7. Result from SDT

7. Discussion

The utilization of this rule-based generation applies to architectural components, such

as landscape furniture, walls and room dividers. By employing grammar, the designer becomes capable of managing the complexity associated with architectural elements. Furthermore, comprehending the production rule and its grammar provides designers with increased flexibility in designing the interwoven block and the aggregation of it.

This study emphasizes the potential of rule-based generation in creating interwoven assembly systems. By taking curves as token input, the system can generate both the interwoven assemblies and the corresponding syntax representing these assemblies. Furthermore, the integration of attribute grammar enhances the parsing process, particularly concerning geometric information. Attribute grammar facilitates the formal inclusion of semantic information, utilizing attributes, with a specific focus on symbols representing geometric objects in this study.

However, this study has certain limitations. Firstly, it does not currently support branched token input. Another limitation is its restriction to working only with curves at perpendicular angles. This implies that the software lacks the capability to generate interwoven assemblies at angled intersections. Lastly, the scope of this research is limited to flat surfaces. In future research, the integration of L-systems will be explored to overcome the branch limitation. Additionally, the application of interwoven assemblies on curved surfaces will be a focus of further investigation.

Reference

Aho, A. V., Lam, M. S., Sethi, R., & Ullman, J. D. (2007). Compilers: Principles, techniques, & tools (2. ed., Pearson internat. ed). Pearson Addison-Wesley.

Bikaun, T., Stewart, M., & Hodkiewicz, M. (2022). Using Context-Free Grammar to Generate Synthetic Technical Short Texts. In H. Aziz, D. Corrêa, & T. French, in *AI 2022: Advances in Artificial Intelligence* (Vol. 13728, pp. 325–338). Springer International Publishing.

Bruton, D. (1997). Grammars and Art. In R. Junge, *CAAD futures 1997* (pp. 71–82). Springer Netherlands.

Casado, A., Sánchez, A., Marieta, C., & Leon, I. (2021). Use of Flat Interwoven Wooden Strips in Architecture and Construction. Simulation and Optimization Using 3D Digital Models. *Sustainability*, 13(11), 6383. https://doi.org/10.3390/su13116383

Muslimin, R. (2010). Interweaving Grammar: Reconfiguring Vernacular Structure through Parametric Shape Grammar. *International Journal of Architectural Computing*, 8(2), 93–110. https://doi.org/10.1260/1478-0771.8.2.93

Muslimin, R. (2011). One-piece weaving: Reconfiguring folding and knotting algorithm in computational design. In *International Conference on Computer-Aided Architectural Design Research in Asia, CAADRIA 2011* (pp 9–18).

Prasad, T. K. (2009). *Attribute Grammars*. Encyclopedia of Information Science and Technology, Second Edition. http://dx.doi.org/10.4018/978-1-60566-026-4.ch046

Prusinkiewiczy, P., Hammely, M., & Hananz, J. (1996). L-system: from the theory to visual model of plants. In *2nd CSIRO Symposium on Computational Challenges in Life Sciences (pp. 1-32)*.

Shih, S.-G. (2018). The Art and Mathematics of Self-Interlocking SL Blocks. In *Bridges 2018 Conference* (pp. 107–114).

Sipser, M. (2012). *Introduction to the theory of computation (3rd Ed)*. Course Technology Cengage Learning.

Teboul, O. (2021). *Shape Grammar Parsing: Application to Image-based Modeling*. Ecole Centrale de Paris.