Weaving Architectural Façades

Exploring algorithmic stripe-based design patterns

Inês Caetano [1], António Leitão [2]

[1,2] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa
[1] ines.caetano@tecnico.ulisboa.pt
[2] antonio.menezes.leitao@tecnico.ulisboa.pt

Abstract. With the recent technological developments, particularly, the integration of computational design approaches in architecture, the traditional art techniques became increasingly important in the field. This includes weaving techniques, which have a promising application in architectural screens and façade designs. Nevertheless, the adoption of weaving as a design strategy still has many unexplored areas, particularly those related to Algorithmic Design (AD).
This paper addresses the creation of weave-based façade patterns by presenting a Generative System (GS) that aids architects that intend to use AD in the design of façades inspired on traditional weaving techniques. This GS proves to reduce the time and effort spent with the programming task, while supporting the exploration of a wider solution space. Moreover, in addition to enabling the integration of user-generated weaving patterns, the GS also provides rationalization algorithms to assess the construction feasibility of the obtained solutions.

Keywords: Algorithmic Design, Façade Design, Weaving Patterns, Algorithmic Framework, Rationalization Processes.

1 Introduction

Architecture, society, and its culture, have always been tightly connected, influencing each other and accompanying the cultural and technological trends. The Arts and Crafts movement is one good example: combined with recent computation-based approaches, the different traditional art techniques are becoming increasingly important in architecture, e.g., glass-blowing methods [1], bamboo structures [2], folding and knotting [3], wreath designs [4], ceramics [5], wall paper design [6], and weaving techniques [7]–[11]. Regarding the latter, it has a visible promising application in the development of geometrical patterns for architectural screens and façades and, in fact, architects have explored the tectonic expressivity of weaving to achieve both aesthetical and performance design goals. This trend has been also

supported by the latest computation-based design approaches, including Algorithmic Design (AD), which (1) promoted the generation and manufacture of façade designs, (2) facilitated the modeling of more complex solutions, and (3) empowered architects' creativity, namely in terms of design expressiveness.

Despite previous efforts, weaving design strategies are still largely unexplored, particularly those related to AD. In this paper, we address the creation of weaving façade designs by using a novel AD approach suitable for architects with programming experience. The proposed approach provides specific support to: (1) devise algorithms that emulate weaving techniques, (2) combine the different envisioned algorithms to efficiently express weaving patterns, (3) embed optimization algorithms and rationalization techniques to assess the fabrication feasibility and refine the fabrication process of the produced weaving designs, and (4) apply the approach to algorithmically describe building envelopes.

2 Weaving Patterns in Architecture

In weaving techniques, stripe-shaped elements are often bent and interweaved, creating 2D or 3D structures that take advantage of the strength and stiffness of the material used. Although these techniques were mainly applied in the Art and Crafts universe to produce baskets, screens, tapestries, among others, they promise to have a wide applicability in architectural design, mainly in the development of building façades.

In architecture, we can find some literature on the use of visual programming tools to create weaving patterns [7]–[9]. Still, the proposed approaches are focused on the conceptualization phase, leaving the detailing, fabrication, and construction phases to be handled by other means. Moreover, they have limitations regarding the efficient exploration of weaving patterns, namely due to the lack of scalability and expressiveness of current visual programming languages [12].

Tan and Lee [13] study the use of textiles to create flexible molds of perforated curved cladding panels for customization, but do not address the specificity of the weaving structures. Gokmen [14] explores and implements in Grasshopper the geometrical rules of the *Shoowa Kuba* textile patterns to show their applicability in the digital design domain. Although the author uses one of these patterns to create a conceptual wrinkled façade design, he does not explain (1) the process of its application, (2) how to use different *Shoowa Kuba* patterns in developing façade designs, and (3) how flexible is the implementation to accommodate design changes. Davis [15] extends the traditional textile craft techniques by (1) coding three types of textiles structures for 3D printing, and (2) analyzing the relationship between the 3D printed textile's geometry and behavior, i.e., its capability to elongate, compress, twist, etc. However, despite demonstrating that using AD in testing different 3D textiles improves designers' relationship with the geometry, the scope of the research neither approaches the generation of large-scale weaving patterns, nor their application on building façades. The textile-based Thread Pavilion [16] explores the boundaries of scaled applications of knitted surfaces, resulting from the combination

of architecture, textiles, sportswear and engineering disciplines. Huang et al. [17] propose a new weaving structure system, composed of long elastic members that are weaved together for the construction of continuous curving lightweight structures. Still, such structures are suitable for exhibition and leisure spaces but not for the creation of building façades. Yan et al. [18] present a weaving structure system to construct different organic geometries, still it only focuses on generating the mesh geometry to which the weaving structure is going to be applied.

Considering the current scenario, we propose a Generative System (GS) to create weave-based façade designs. This GS aims to support architects that intend to use AD in the exploration of different façade designs inspired by traditional weaving patterns. Using this GS not only reduces the time and effort spent with the programming task, but also supports the exploration of a wider solution space. Furthermore, it allows the integration of user-generated weaving patterns, making them available for future application cases. Finally, in addition to the available geometric algorithms, the proposed GS provides some optimization algorithms, as well as rationalization strategies assessing the solutions' construction feasibility.

3 Weaving Patterns Generative System

Generally, most geometric patterns on façades result from the repetition of a certain shape or element, which can be kept unchanged along the façade's domain or can be transformed regarding its shape, size, etc. In some cases, the distribution of these elements along the façade results from its discretization in two directions, which results in the repetition of the elements along both dimensions of the façade (see Fig. 1A). In others, their distribution derives from the façade discretization in a single direction, which means that the elements are repeated in only one of the dimensions, being therefore continuous in the other (see Fig. 1B). In this paper, we address the algorithmic generation of the second case. To this end, we start by focusing on the mathematical representation of this type of geometric patterns with respect to:

1. the way each element's shape is created;
2. the type of geometric transformations supported;
3. the positioning of each element on the surface.

Fig. 1. A. Lisbon Aquarium extension by Campos Costa Arquitetos (@aasarchitecture); B. Tróia Design Hotel by Promontório Architects (https://www.pinterest.co.uk).

Considering that most weaving patterns are composed of stripe-shaped elements, i.e., elements that are continuous in one of the surface dimensions but discrete in the other, the proposed GS focuses on the mathematical development of such elements to then create weaving patterns. In the following two sections, we explain the mathematical strategy behind (1) the creation of different stripe-shaped elements and (2) the application of geometric transformations to produce multiple weaving patterns.

3.1 Stripe-shaped Elements

Currently, we can find several examples of façade designs composed by elements that are discrete in one of the surface's dimensions but are continuous in the other. In this research, we address the generation of geometric patterns for façades based on such elements, presenting a GS whose algorithmic structure was designed to consider two main tasks: (1) the creation of stripe-shaped elements, which we named *striped* elements; (2) the application of geometric transformations, especially the interweaving of elements. The resulting geometry derives from a combination of algorithms addressing different geometric features.

To make the GS more flexible, it relies on the use of anonymous functions, i.e., functions that do not have a name, and Higher-Order Functions (HOF) [19], i.e., functions that receive functions as arguments or return functions as result. Therefore, to mathematically describe each continuous element, the GS provides a function (*stripe*) that receives a set of other functions as input, each one representing a different geometric feature: (1) the stripe's central axis (S_{axis}), (2) the stripe's section shape ($S_{section}$), and (3) the transformation to apply ($S_{transformation}$). Equation 1 represents this HOF.

$$stripe(S_{axis}, S_{section}, S_{transformation}) \quad (1)$$

As a simple example, imagine a façade surface on which we want to apply stripe-shaped elements. We start by sampling the surface at $n \times m$ points, obtaining a matrix of points, as represented in Fig. 2 (top). On this matrix, we can then create either horizontally oriented stripe-shaped elements, if we provide the *stripe* function with the matrix' rows ($striped_{\rightarrow}$), or vertically oriented ones, if we use instead the matrix' columns ($striped_{\uparrow}$). In both scenarios, the algorithms receive the following set of parameters: the surface on which the stripes are to be generated (*surf*), the number of stripes to be created (n_{stripes}), and the function responsible for producing each stripe (*stripe*). Equation 2 conceptually represents the $striped_{\uparrow}$ function and Fig. 2A-C outlines its application in generating a set of vertical stripes.

$$striped_{\uparrow}(surf, n_{stripes}, stripe) \tag{2}$$

Note that the stripes obtained in this example are regular, i.e., they share the same curvature as the surface because no transformation was applied to them. In the following section we explore the application of transformations to produce different weaving patterns.

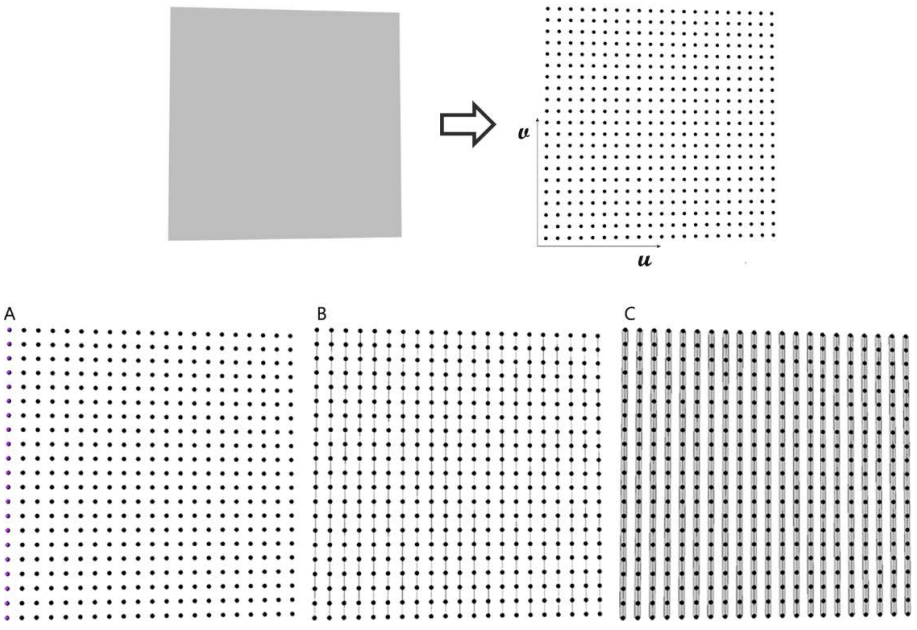

Fig. 2. Top: the surface discretization into a $n \times m$ matrix of points. Bottom: the generation of vertical stripes – A. selection of a matrix' column; B. creation of a stripe axis at each column; C. generation of the stripes.

3.2 Geometric Transformations

The proposed GS aims at facilitating the creation of weave-based façade patterns by using an AD approach. Considering that weaving patterns are composed of stripe-shaped elements that are strategically bent to not intersect each other, the GS must address the *bending* transformation along with the *weaving* one. For both transformations, the GS provides a set of algorithms that can be combined with the *stripe* algorithm, thus affecting how each stripe-shaped element is generated.

Regarding the *bending* transformation, its algorithmic implementation is quite simple because it is not affected by the geometric transformations applied to the other façade elements. In practice, it simply controls the undulating movement of each stripe according to the values given to its parameters. Contrarily, the implementation process of the *weaving* transformation is more complex. Although it is based on multiple *bending* transformations, the values given for each stripe *bending* transformation are dependent on the other stripes values so as to ensure that they do not intersect each other.

Considering this, we start by approaching the *bending* transformation in the next section. We describe its algorithmic implementation and structure and we explain its practical application with simple examples. Then, the following section follows the same structure but focusing on the *weaving* transformation.

Bending

As mentioned previously, the *stripe* function supports a set of transformation algorithms ($S_{transformation}$). It is frequently the case where these transformations can be decomposed into a series of more elementary transformations (Equation 3).

$$stripe(S_{axis}, S_{section}, S_{transformation}) = stripe(S_{axis}, S_{section}, T_{bend}(M_{bend}, P_{bend})) \quad (3)$$

As the name suggests, the T_{bend} algorithm controls the wavelike movement (*bending*) of each stripe and it receives two inputs:

1. a row vector with one or more types of undulating movements;

$$M_{bend} = [f1_{bend}, f2_{bend}, \dots, fn_{bend}], f_i \in \mathbb{R} \rightarrow \mathbb{R}^n$$

2. a column vector dictating their order of application.

$$P_{bend} = \begin{bmatrix} i_1 \\ i_2 \\ \dots \\ i_n \end{bmatrix}, i_j \in \mathbb{N}$$

In practice, M_{bend} receives a set of algorithms, each one representing a type of wavelike movement, while P_{bend} accepts a set of integers representing the types of movements available in M_{bend}. Given that the f_{1bend} algorithm has index 1 in M_{bend}, it therefore matches that value in P_{bend}. This also happens with the second algorithm, f_{2bend}, thus corresponding to the number 2 in P_{bend}, and with the third algorithm, this time matching the number 3, and so on. As an example, imagine we use the following vectors:

$$M_{bend} = [f_1, f_2, f_3] \qquad \text{and} \qquad P_{bend} = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 3 \end{bmatrix}$$

The result is a continuous pattern composed of three types of stripes, each one resulting from one of the transformations in M_{bend}, whose application follows the sequence $\{f_1, f_2, f_1, f_3, f_1, f_2, \dots\}$.

To address different types of *bending* movements, the GS includes algorithms to handle multiple wavelike effects, which receive as parameters an undulation amplitude ($U_{amplitude}$), phase (U_{phase}), and frequency ($U_{frequency}$). In case the $U_{amplitude}$, U_{phase}, and $U_{frequency}$ are fixed values, the wavelike effect is constant throughout the surface domain. On the other hand, in case one or more of these parameters is changeable, the wavelike movement therefore changes along the façade surface: e.g., when one of the parameters (1) increases or decreases according to the horizontal, vertical, or both surface directions, or (2) varies according to a certain mathematical rule. Fig. 1B is an example of a continuous variation of both U_{phase} and $U_{frequency}$ parameters along the building's height: observing closely, we realize that the initial curvature of each balcony changes from floor to floor, as does its frequency.

To better understand the use of these algorithms, we develop, in a step-by-step process, a conceptual example resulting from their application. Consider a straight surface horizontally divided in n stripes (Fig. 3A-B) where we want to apply a *bending* transformation affecting only alternated stripes. This means that, on each pair of stripes, the first one must remain unchanged, while the second one suffers a transformation. To treat both the same way, we introduce the identity transformation function (id) that does not change its argument, i.e., $id(x) = x$. In mathematical terms, we need the T_{bend} algorithm to receive:

1. a row vector containing two types of *bending* transformations, one for the odd stripes and one for the even ones:

$$M_{bend} = [id, f_1]$$

2. a column vector establishing the alternating sequence by which the *bending* movements are going to be applied:

$$P_{bend} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

In practice, as we are dealing with an alternating sequence, it means the shape of the odd stripes, i.e., the first, third, fifth, etc., are affected by the identity transformation

algorithm (id), whereas the even ones are affected by the f_1 algorithm (see Equation 4). Since an id function returns the value given as its argument, in this case, its use will return the original stripe shape, creating no *bending* transformation as a result. This means that the stripes to which this algorithm was applied were kept unchanged. In contrast, the stripes affected by the algorithm f_1 become undulated (see Fig. 3C).

$$f_1\,(t) = U_{amplitude}(t) \times \sin\big(U_{phase}(t) + t \times U_{frequency}(t)\big) \tag{4}$$

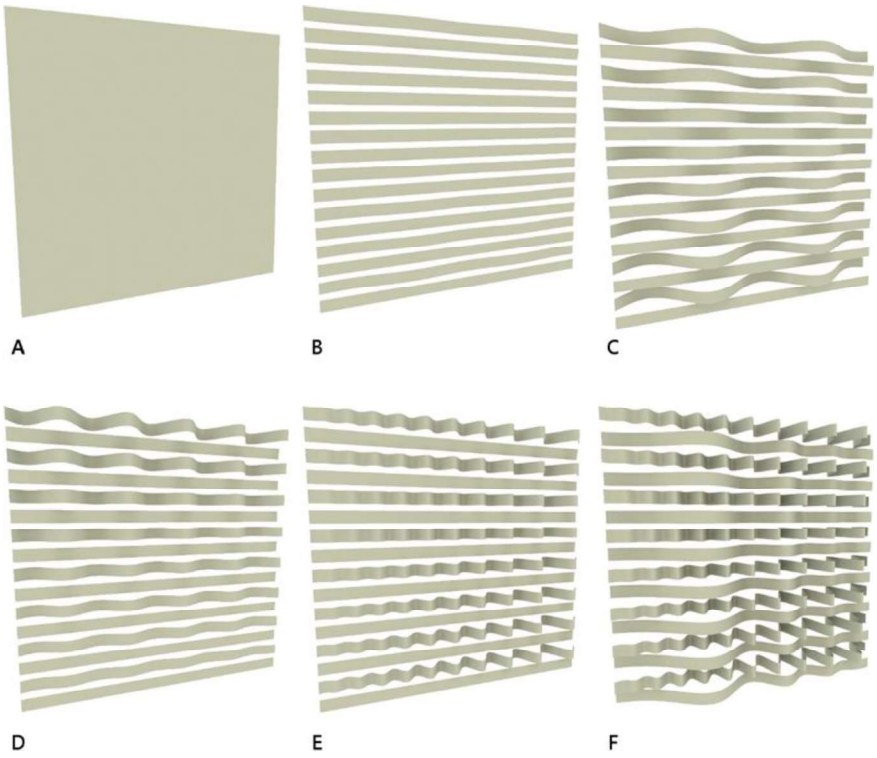

Fig. 3. The application of both *stripe* and T_{bend} functions: A. The initial surface; B. Creation of n horizontal stripes; C. T_{bend} algorithm with one transformation; D. Amplitude increasing with the vertical direction; E. Amplitude increasing with the horizontal direction; F. T_{bend} algorithm with two transformations.

Now, imagine that the f_1 algorithm is combined with another algorithm that controls the *bending* amplitude value according to the position along the façade domain. Such algorithm is implemented in the GS as $f_{\nearrow}(t) = t \times f_{intensity}$, in which $f_{intensity}$ is a factor varying between 0 and 1. To make the amplitude value vary along the façade's height, we need to combine the $f_{\nearrow}$ algorithm with the c_y algorithm, which selects the *ordinate* value of a given coordinate: $c_y(p) = p.y$. The result is a new algorithm, the $f_{y\nearrow}$, which

is implemented as follows: $f_{y\nearrow}(p) = f_{\nearrow} \circ c_y = f_{\nearrow}(c_y(p))$. In practice, the result of combining both $f_|$ and $f_{y\nearrow}$ algorithms is an alternation between straight stripes and bended stripes, whose undulation amplitude increases with the façade height (see Fig. 3D). Conversely, to increase the *bending* amplitude with the surface length, as shown in Fig. 3E, we need to combine the $f_{\nearrow}$ algorithm with the c_x algorithm instead, which is implemented in the GS as $c_x(p) = p.x$, thus obtaining the $f_{x\nearrow}$ algorithm.

Finally, in case we exchange the *id* algorithm for another *bending* transformation, the stripes that previously were straight, now become undulated according to the received parameters. Fig. 3F and Equation 5 illustrate this last example.

$$T_{bend}\left([f_1, f_2], \begin{bmatrix}1\\2\end{bmatrix}\right) = [f_1, f_2, f_1, f_2, \dots] \tag{5}$$

Weaving

This section focuses on the implementation of *weaving* transformations in the GS. Also known as woven structures, these transformations are based on stripes going over and under each other without knotting [20]. Currently, we can find several buildings whose façade designs are inspired by the traditional weaving techniques (Fig. 4).

Fig. 4. A. Eemsmond Building in Deflzijl, 2006 (photo by @hollandcomposites); B. Yong He Yuan residential buildings, 2014 (photo by ©NEDELEC CO. LTD); C. The Triton Building at Regent's Place, 2013 (photo by ©TateHindle Ltd); D. Cincinnati Children's Hospital Medical Center, 2009 (source: http://cambridgearchitectural.com).

Observing the structure of weaving patterns, we realize that it consists in undulating stripes arranged in perpendicular directions. As a first approach, we could create weaving patterns by (1) producing a set of n vertical stripes and a set of m horizontal stripes, and (2) applying a *bending* transformation to both sets of stripes. However, this would require us to solve the intersections between vertical and horizontal stripes each time a new design variation was generated.

Even though dealing with the intersection between stripes seems to be a simple process, such task ends up having to be repeated every time the design is changed, which is a current reality in the typical architectural design process. Furthermore, to avoid the stripes' intersection, distinct design changes might require distinct modifications to the weaving pattern:

1. when changing the number of stripes, the frequency must be increased;
2. when experimenting with different weaving types, both phase and frequency must vary accordingly;
3. when modifying the surface dimensions or curvature, the amplitude, phase, and frequency must be adjusted.

To overcome these limitations, while facilitating the exploration of weave-based façade patterns, the GS provides a set of functionalities addressing different *weaving* transformations. Equation 6 represents the T_{weaved} function, which receives as parameters (1) the surface on which the weaving pattern will be applied (*surf*), (2) the number of vertical and horizontal stripes composing the weaving pattern (n_{stripes} and m_{stripes}), (3) the amplitude of the weaving movement (U_{amp}), (4) the type of weaving pattern (M_{weaved}), and (5) the shape of the stripes section ($S_{section}$).

$$T_{weaved}(surf,\ n_{stripes},\ m_{stripes},\ U_{amp},\ M_{weaved},\ S_{section}) \tag{6}$$

At this stage, one of the main challenges was to establish the best strategy to mathematically represent different weaving types in the M_{weaved} matrix. After studying several possibilities, we decided to follow the approach of Grünbaum and Shephard [21] to represent fabrics: using a 2D squared regular mesh to represent the different weaving patterns, where each squared-unit can be colored black, meaning the stripe along the vertical direction passes over the stripe in the horizontal direction, or colored white, meaning the opposite (see Fig. 5).

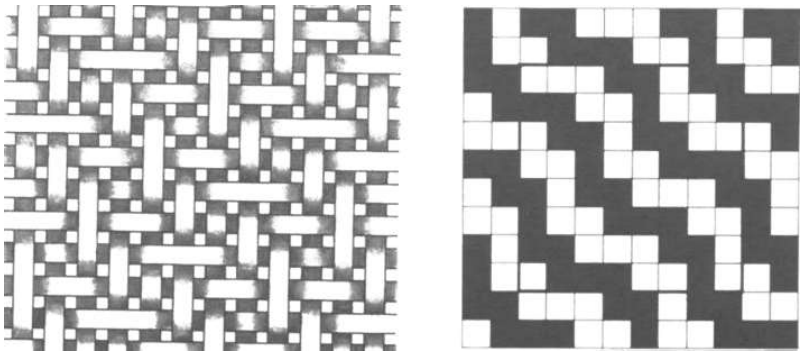

Fig. 5. Left: a sketch of the fabric; Right: the conceptual representation of the fabric [22].

In mathematical terms, the M_{weaved} parameter corresponds to a matrix illustrating the different types of weaving movements based on two values, one representing the white color and the other the black color. Using this strategy, we can therefore inform

the T_{weaved} algorithm about the weaving movement that each stripe should follow. Fig. 6 illustrates a weave-based pattern together with its corresponding M_{weaved} matrix.

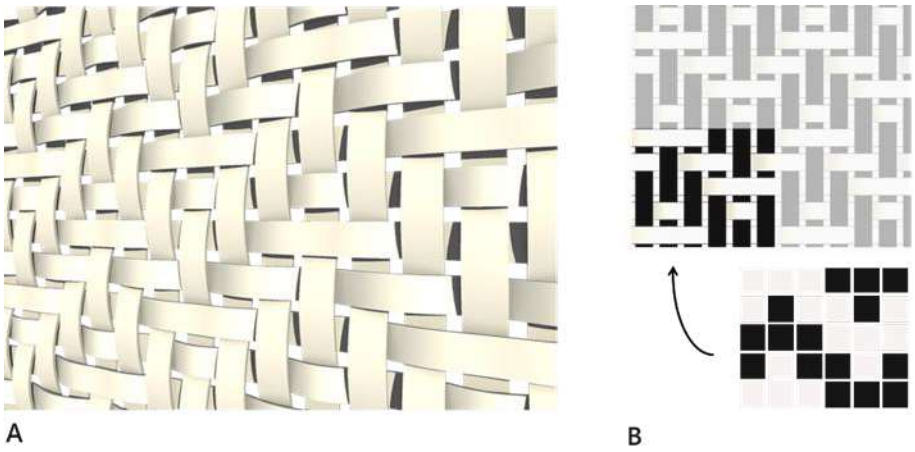

Fig. 6. A weaving movement created using the T_{weaved} algorithm: A. the obtained model in Rhinoceros 3D; B. the weaving pattern conceptual representation.

4 Evaluation

As the aim of the proposed GS is to facilitate the algorithmic development of weaving façade patterns, it therefore provides several types of weaving movements already predefined. To improve the solutions obtained, as well as to ensure their construction viability, the GS also integrates some optimization and rationalization algorithms. In this section, we evaluate the GS by applying these different types of algorithms in several case studies.

4.1 Weaving Patterns Creation

To use the GS, the user can simply select the weaving type to use, which corresponds to a M_{weaved} matrix, and, then, combine it with the T_{weaved} algorithm. As a result, this not only accelerates the design exploration of weaving patterns, but also facilitates the generation and evaluation of a wider range of weave-based solutions. Fig. 7 shows a set of weaving patterns resulting from different M_{weaved} matrices.

Additionally, to augment the visual expression of the weaving patterns explored, the T_{weaved} algorithm allows the use of different colors/materials for both vertical and horizontal stripes. To this end, the T_{weaved} algorithm was designed to support an additional parameter containing this information (M_{colors}), which corresponds to a pair of matrices: a column vector with the colors/materials of the horizontal stripes and another one regarding the vertical stripes. As an example, the models in Fig. 8 result from two different M_{colors}:

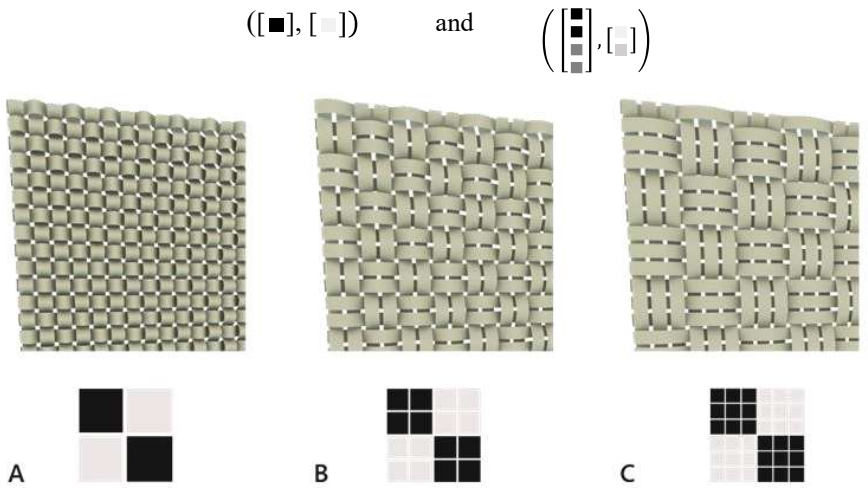

Fig. 7. Three weaving patterns available in the GS: A. chess, B. double-chess, and C. triple-chess.

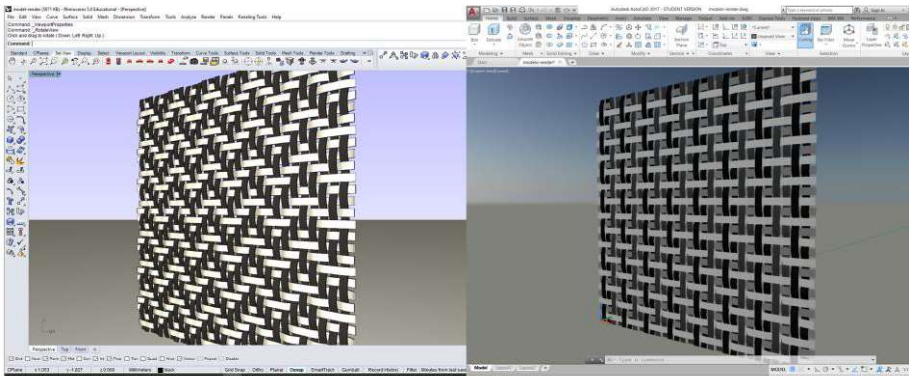

Fig. 8. The same weaving pattern resulting from two different M_{colors} matrices and generated in two different CAD tools: Rhinoceros 3D and AutoCAD.

Note that the left-side model was generated in *Rhino* and the right-side one in *AutoCAD*. This, therefore, demonstrates the independence of the GS regarding particular CAD tools. In fact, one advantage of using a mathematical description is that it can be easily implemented in different AD systems. Regarding Fig. 9, the set of examples demonstrate that the use of different colors/materials increases the visual expression of the different weaving patterns available in the GS.

Another advantage of the GS is the automatic adaptation of the weaving pattern being explored to the different design changes. Fig. 10 proves this ability by presenting a set of design solutions resulting from the same algorithm: the same weaving pattern is used with a varying number of stripes or applied on surfaces of different geometries.

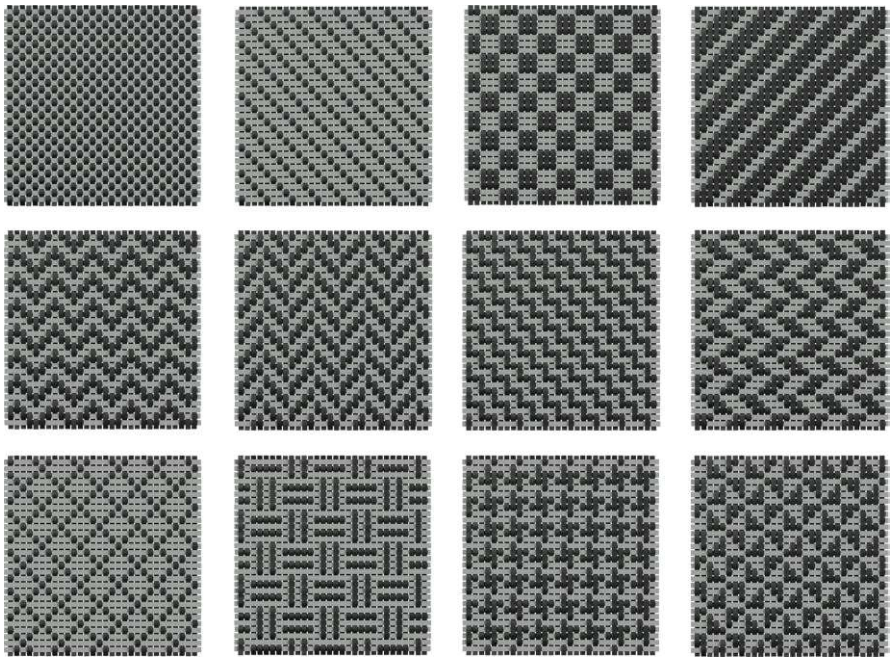

Fig. 9. Different weaving patterns available in the GS.

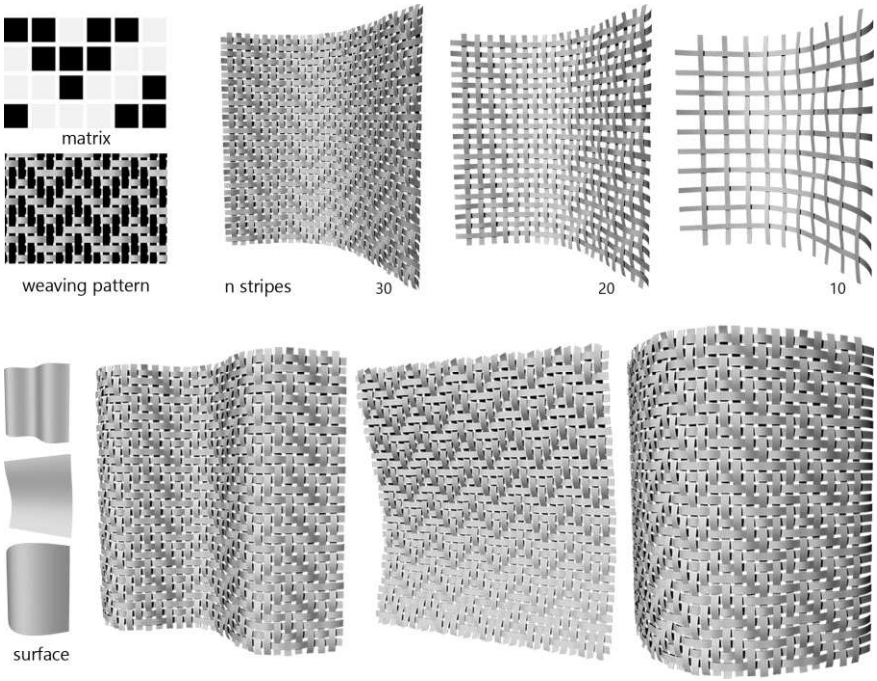

Fig. 10. Weaving patterns resulting from the same algorithm.

4.2 Weaving Patterns Optimization

The GS also supports the development of design solutions influenced by their own performance. To this end, each design solution must be subjected to an analysis process regarding a fitness criterium and, then, the results obtained must be used to inform the generation of the next design iteration. In practical terms, this scenario requires the combination of the algorithms that create different weaving patterns with a set of analysis and optimization algorithms.

In this section, we focus on the creation of weaving design solutions based on their natural lighting illumination performance. Thereupon, we added to the GS an algorithm to control the fraction of *opaque vs. transparent* areas of the weaving patterns being created. This algorithm allows the architect to establish the weaving pattern's percentage of *opaque* area (f_{opaque}) in addition to the previously mentioned parameters: *surf*, $n_{stripes}$, $m_{stripes}$, U_{amp}, and M_{weaved}. This results in a set of visually diverse design solutions that simultaneously respect the imposed percentage. Equation 7 conceptually describes the $\%_{opaque}$ algorithm, in which *surf*, $n_{stripes}$, $m_{stripes}$, U_{amp}, and M_{weaved}, are the same parameters as in Equation 6 and the f_{opaque} is the parameter that controls the fraction of opaque area.

$$\%_{\text{opaque}}(surf, n_{\text{stripes}}, m_{\text{stripes}}, U_{\text{amp}}, M_{\text{weaved}}, f_{\text{opaque}}) \tag{7}$$

To clarify this combination of geometric and optimization algorithms, we now develop a set of examples that explore different weaving patterns by considering different opaqueness values. We start with a straight façade on which we apply a weaving pattern matching the following matrix:

$$M_{weaved} = \begin{bmatrix} \blacksquare & \blacksquare & \square \\ \square & \blacksquare & \blacksquare \\ \blacksquare & \square & \blacksquare \end{bmatrix}$$

To evaluate the indoor illumination level that each variation promotes, we apply the $\%_{opaque}$ algorithm to generate a set of design solutions deriving from:

1. the chosen weaving type;
2. the assignment of different values to the $n_{stripes}$, $m_{stripes}$, and f_{opaque} parameters.

Then, the algorithm automatically adjusts the width of the stripes to match the given opaqueness values. Fig. 11 presents some of the solutions obtained: the values of both $n_{stripes}$ and $m_{stripes}$ parameters vary between 10 and 20, whereas the f_{opaque} parameter ranges between 0.15, 0.55, and 0.95.

Despite the simplicity of this example, the use of algorithms similar to the $\%_{opaque}$ promises to be quite useful for the optimization of weave-based façade solutions. Additionally, the application of such algorithms can be done during the entire design process, i.e., from conceptual design stages to more advanced and detailed ones. More algorithms of the same type are being developed to further complement the GS,

addressing different design analysis and optimization strategies, as well as performance criteria.

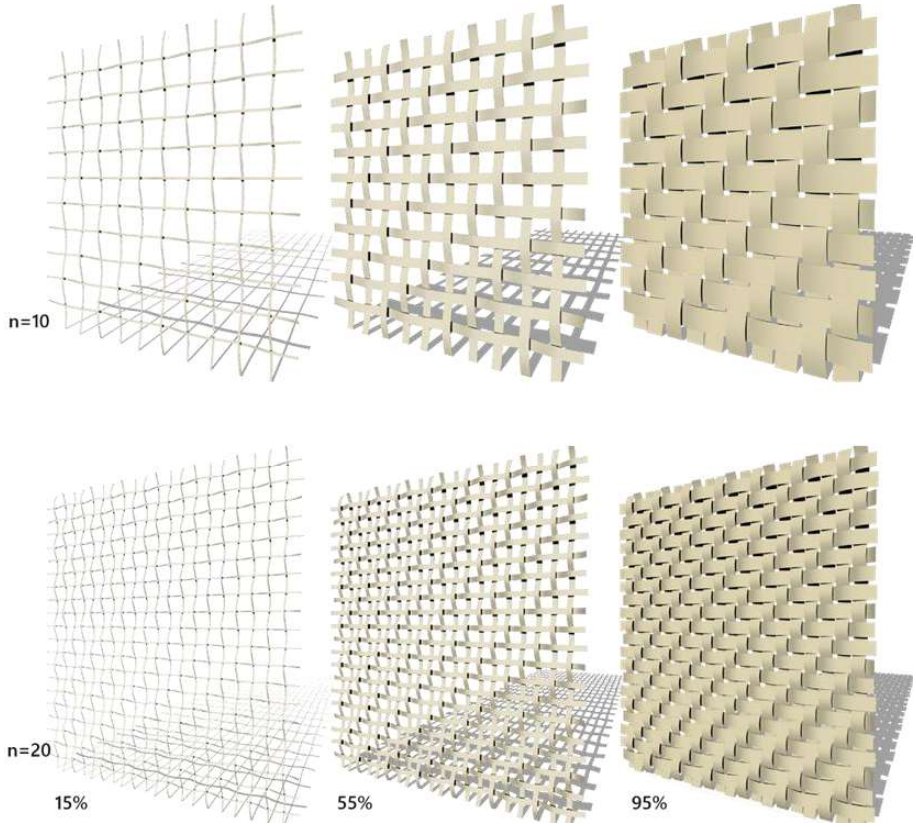

Fig. 11. A set of optimized weaving patterns: the top solutions have 10×10 stripes and the bottom ones have 20×20 stripes, and each column matches a percentage of opaque area - 15%, 55%, and 95%, correspondingly.

4.3 Weaving Patterns Rationalization

In addition to allowing the exploration of several design variations and their optimization regarding a certain fitness criterion, it is important to ensure that the obtained solutions are possible to manufacture. To this end, the GS also addresses the feasibility of the weaving patterns being explored, providing algorithms to:

1. discretize the different types of weaving patterns;
2. fabricate the façade panels of each weaving pattern;
3. rationalize the obtained solutions by making a balance between the intended design and the most affordable design solution.

Regarding the first two issues, the GS provides some algorithms capable of producing the façade module corresponding to the type of weaving pattern being developed. This means producing a 3D model of each module, already including the support to then connect with the other modules. In general, the dimension of each module matches the size of the matrix that represents each weaving type: e.g., a 2×2 matrix corresponds to a façade module composed by 2×2 stripes, a 4x4 matrix to a 4×4 stripes module, and so on. Fig. 12 illustrates the discretization process of a façade weaving pattern based on a *piedpull* pattern, which is described by a 4×4 matrix, with its corresponding façade module sized of 4×4 stripes.

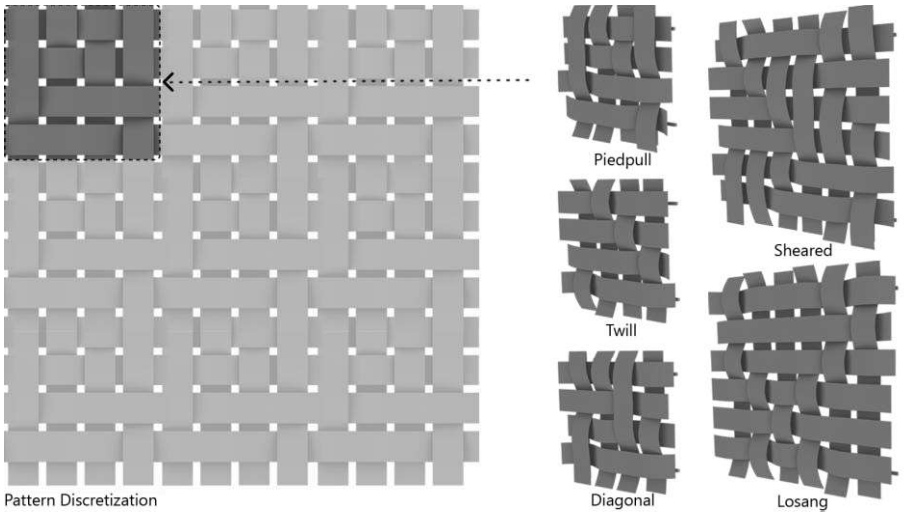

Fig. 12. Left: Discretization process of a façade weaving pattern based on a *piedpull* pattern; Right: a set of different façade modules resulting from different weaving patterns.

Regarding the third issue, the GS provides the algorithm R_{weaved}. This algorithm is responsible for making the balance between the user's design intent and the most affordable design solution. Since the number of stripes is the design parameter that has the strongest impact on the solution's affordability, the available algorithm focuses mainly on balancing this design property. To better explain this process, we develop a simple façade pattern composed by 10 horizontal and vertical stripes and we use a type of weaving pattern whose matrix has a 4×4 size. As the corresponding façade module has also a size of 4×4 stripes, the resulting pattern requires the use of 2.5×2.5 modules in both directions. This means manufacturing two types of pieces: an entire module and half of a module, which is unnecessarily expensive. In fact, if the pattern had 12×12 stripes instead of 10×10, it would allow us to use 3×3 modules and avoid the manufacturing of half modules. This example is illustrated in Fig. 13 (left side).

Besides the simplicity of this example, it demonstrates that a simple change in the number of stripes facilitates the manufacturing process. Fig. 13 (right side) illustrates the application of the R_{weaved} algorithm in a façade design inspired by the *piedpull*

weaving pattern with a varying number of stripes. Note that there is a range of values for which the algorithm produces the exact same solution, because it is considered the most affordable one, e.g., for the range between 10 to 13, the algorithm suggests a solution with 12 stripes, for the range between 14 and 17, it suggests a 16 stripes solution, and so on.

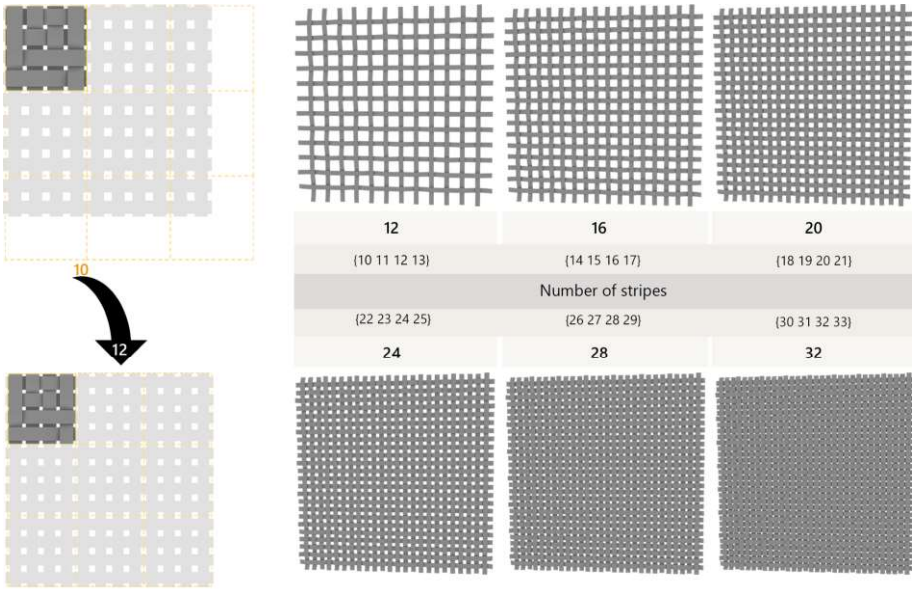

Fig. 13. On the left: Rationalization process of a weaving pattern. On the right: The application of the R_{weaved} algorithm – according to the number of stripes given, the algorithm returns the solution that is more affordable to fabricate.

Regarding the materiality of the modules, we are currently studying different alternatives.

The first one uses bended metal sheets [23][24] made of aluminum (1mm), steel (0.5 mm), or titanium (0.5 mm). For their manufacture, assembly, and fixation on a façade, we foresee two possible approaches: (1) a more traditional one, using squared support frames made of I-beam metal profiles to fix and give the necessary structural stability to the stripes, which already include docking systems that fit those on the façade structure; (2) a more digital one, entirely based on additive manufacturing, e.g., 3D printing, producing both the façade module and its supporting system at once.

The second alternative considers the use of ceramic [25] or concrete [26] in the production of these modules. To this end, the GS provides algorithms to automatically create a 3D model of each weaving pattern corresponding mold. The latter is then used to manufacture the required number of modules, which then follow a fixation strategy identical to the one used with metals.

Finally, the last alternative regards the use of polymers as a constructive material [27], a possibility whose implementation in the GS appears to be accessible via the use of additive manufacturing techniques.

As future work, we plan to develop prototypes to evaluate these approaches.

5 Conclusion

Architecture has always been influenced by society and its culture. By combining the Arts and Crafts movement with recent computation-based approaches, the different traditional art techniques became increasingly usable in architecture. Within these techniques, this paper focused on traditional weaving processes due to their promising application in the development of patterns for architectural screens and façades.

As the adoption of weaving patterns in architecture has been little explored, especially when considering Algorithmic Design (AD) approaches, this paper addressed the creation of weave-based façade designs by using an AD approach. The proposed approach was implemented in an algorithmic-based Generative System (GS) that provides algorithms to: (1) emulate different types of weaving patterns, (2) efficiently apply such patterns in the development of building façades, (3) optimize the obtained designs regarding a fitness criterion, and (4) embed rationalization techniques to both assess and refine the fabrication feasibility of the produced weaving designs.

In the paper, we described the GS algorithmic structure and we explained the application of the different types of algorithms. To this end, a set of simple examples were developed to evaluate the suitability of the GS for creating different weaving façade patterns. As a result, the set of examples ended up evidencing the GS flexibility by proving its ability to support the systematic application of design changes. Also, the examples demonstrated the GS ease of use, as the user takes advantage of a large variety of algorithms specially designed for the creation of different weaving façade patterns. Furthermore, they also proved that the use of the proposed GS reduces the time spent on the programming task, while facilitating the algorithmic resolution of the geometric problems that typically emerge during the development of weave-based patterns.

Moreover, the GS ability to combine both geometric and simple optimization algorithms was evaluated. An example was developed, which uses optimization algorithms to adapt the geometric characteristics of the weaving pattern to meet a given fitness criterion. This is a promising research area that we plan to further develop by implementing additional optimization algorithms within the GS.

Finally, the potential of the GS for integrating rationalization strategies was also illustrated by (1) discretizing a set of façade weaving patterns, while creating the corresponding façade modules/panels, and (2) establishing a balance between the intended design and its fabrication viability. Nonetheless, it already denotes the relevance of such methods in designing weave-based façade patterns. We will also consider this research area in our future work, extending the algorithms available in

the GS to support the rationalization of curved surfaces, while considering other constructive features that have an impact on the manufacturing process.

Acknowledgements. This work was supported by national funds through *Fundação para a Ciência e a Tecnologia* (FCT) with references UID/CEC/50021/2019 and PTDC/ART-DAQ/31061/2017, and by the PhD grant under contract of FCT with reference SFRH/BD/128628/2017.

References

1. I. Koh, "Generative-glass: Prototyping Generative Architectural Systems with Artisan's Glass Blowing and Automated Digital Fabrication Techniques," in Rethinking Comprehensive Design: Speculative Counterculture, Proceedings of the 19th International Conference on Computer- Aided Architectural Design Research in Asia CAADRIA 2014, 2014, pp. 389–398.
2. J. M. Huang, "Integrating Computational Design and Traditional Crafts: A Reinvention of Bamboo Structures," in Protocols, Flows and Glitches, Proceedings of the 22nd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2017, 2017, pp. 437–445.
3. R. Muslimin, "One-Piece Weaving: Reconfiguring folding and knotting algorithm in computational design," in Circuit Bending, Breaking and Mending: Proceedings of the 16th International Conference on Computer-Aided Architectural Design Research in Asia, 2011, pp. 9–18.
4. S. Y. Özgan, "Recalling the Heritage of Ancient Wreath Designs: An Exploration Of The Computation In Making," in SIGRADI 2015 [Proceedings of the 19th Conference of the Iberoamerican Society of Digital Graphics], 2015, pp. 183–189.
5. J. E. Sabin, "Digital Ceramics: Crafts-based Media for Novel Material Expression & Information Mediation at the Architectural Scale," in ACADIA 2010: Life in:formation, 2010, pp. 174–181.
6. S. Gokmen and D. Baerlecken, "Digital Wallpaper: Tiles of Proliferation and Continuity," in Acadia 2014: Design Agency, 2014, pp. 301–310.
7. Z. Khabazi, Generative algorithms - Concepts and Experiments: Weaving. MORPHOGENESISM, 2010.
8. F. I. Harnomo and A. Indraprastha, "Computational Weaving Grammar of Traditional Woven Pattern," in Proceedings of the 8th ASCAAD Conference 2016, 2016, pp. 75–84.
9. M. Feorgiou, O. Georgiou, and T. Kwok, "Form Complexity – Rewind," in ACADIA 2014, 2014, pp. 229–236.
10. R. Muslimin, "Learning from weaving for digital fabrication in architecture," Leonardo, vol. 43, no. 4, pp. 340–349, 2010.
11. J. E. Sabin, "Matrix Architecture," in Inside Smartgeometry, Wiley-Blackwell, 2014, pp. 60–71.
12. A. Leitão, L. Santos, and J. Lopes, "Programming Languages for Generative Design: A Comparative Study," IJAC, vol. 10, no. 1, pp. 139–162, 2012.
13. Y. Y. Tan and T. L. Lee, "The flexible textile mesh," in Learning, Adapting and Prototyping, Proceedings of the 23rd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2018, Volume 2, 2018, vol. 2, pp. 349–358.

14. S. Gokmen, “The Ornaments of Shoowa Kuba: A digital re-interpretation of a textile art,” in SIGraDi 2012 [Proceedings of the 16th Iberoamerican Congress of Digital Graphics], 2012.
15. F. Davis, “3D Printed Textiles from Textile Code: Structural Form and Material Operations,” in SIGraDi 2012 [Proceedings of the 16th Iberoamerican Congress of Digital Graphics], 2012.
16. J. E. Sabin, “myThread Pavilion: Generative Fabrication in Knitting Processes,” in Telecommunications Policy, 2013, pp. 347–354.
17. W. X. Huang, D. Yan, P. Luo, and X. L. Li, “Digital design and construction of a weaving structure,” in the 8th International Conference on Fibre-Reinforced Polymer (FRP) Composites in Civil Engineering (CICE 2016), 2016.
18. D. Yan, W. Huang, and Z. Song, “Generation of Weaving Structure on Free-Form Surface Using a Remeshing Algorithm,” in Living Systems and Micro-Utopias: Towards Continuous Designing, Proceedings of the 21st International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA 2016, 2016, pp. 105–114.
19. A. Leitão and S. Proença, “On the Expressive Power of Programming Languages for Generative Design: The Case of Higher-Order Functions,” in Fusion - Proceedings of the 32nd eCAADe Conference - Volume 1, 2014, vol. 1, pp. 257–266.
20. I. Emery, The Primary Structures of Fabrics: An Illustrated Classification, 4th edition Thames & Hudson, 2009.
21. B. Grünbaum and G. Shephard, “Satins and Twills: An Introduction to the Geometry of Fabrics,” Math. Mag., vol. 53, no. 3, pp. 139–161, 1980.
22. B. Grünbaum and G. C. Shephard, “An extension to the catalogue of isonemal fabrics,” Discrete Math., vol. 60, pp. 155–192, 1986.
23. C. Schittich, Ed., “Materials in the building skin – from material to construction,” in Building Skins, Birkhäuser, 2006, pp. 60–69.
24. E. Sopeoglou, “Digital Fabrication Research for Integrated Design Innovation,” pp. 805–812, 2006.
25. D. Rosenwasser, S. Mantell, and J. Sabin, “Clay Non-Wovens,” in ACADIA 2017: DISCIPLINES & DISRUPTION [Proceedings of the 37th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA), 2017, pp. 502–511.
26. M. L. Yu, “Precast concrete spandrel,” in Skins, Envelopes, and Enclosures: Concepts for designing building exteriors, New York and London: Routledge: Taylor & Francis Group, 2014.
27. J. Knippers, F. Scheible, M. Oppe, and H. Jungjohann, “Bio-inspired Kinetic GFRP-façade for the Thematic Pavilion of the EXPO 2012 in Yeosu,” in IASS-APCS Symposium 2012: From Spatial Structures to Space Structures, 2012.