Artificial Intelligence for Engineering Design, Analysis and Manufacturing

cambridge.org/aie

Parametric, grammatical, and perceptual iterations on structural design synthesis

Rizal Muslimin

School of Architecture, Design and Planning, The University of Sydney, Sydney, Australia

Abstract

This paper presents a computational design method to analyze and synthesize representation mechanisms in structural design. The role of shape grammar schemas in analyzing parametric and grammatical structural analysis is discussed, and a set of schema to generate novel structural topology is provided.

Research Article

Cite this article: Muslimin R (2018). Parametric, grammatical, and perceptual iterations on structural design synthesis. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **32**, 269–281. https://doi.org/10.1017/S0890060417000580

Received: 30 November 2016
Revised: 14 August 2017
Accepted: 20 August 2017

Key words:
Computational design; creativity and ideation; design cognition; design methods; shape grammar; structure

Author for correspondence:
Rizal Muslimin,
E-mail: rizal.muslimin@sydney.edu.au

Introduction

Software subroutines can be viewed as a frozen theory of how design intention is computed under certain logic. Therefore, issues on CAD operation for analyzing structural design might occur in the discrepancy between the way the user and the subroutines represent the design intention. In *Vitruvius Redux*, Mitchell outlines the limit of Durand's enumerative object-oriented iteration and its digital embodiment in BIM software. In this software, objects and combinatorial searches allow for an effective design modeling process; however, the provided objects (e.g. roof, wall, and door tools) are constrained by well-known typologies and may not fully align with the designer intention. Reflecting on several structural design cases, Mitchell called for re-promoting the parametric-grammatical approach with a shape grammar embedding mechanism for its impartiality to typological constraints (Mitchell, 1998, 2001). This paper sheds lights on representational issues in structural design from where Mitchell left off.

In physical media, structural problems are represented with a set of physical variables (e.g. material, loading, and supporting elements). Physical behavior on one scale is assumed to resemble that of another scale as the load-stress distribution is considered proportional across scales. For instance, experimental models developed by Gaudi, Otto and Isler (e.g. soap film, chains, and fabrics) served as physical schemas to simulate the behavior of shapes on a larger scale, such as a wide-span roof or a bridge (Collins, 1963; Otto, 1973; Chilton & Isler, 2000). By modifying the loads, supports, and material, the physical model could provide immediate feedback in an attempt to find desirable forms.

In digital media, structural analysis tools represent physical shapes into an assemblage of finite vertices, edges, and faces, along with their mechanical descriptions (e.g. a force's magnitude and the vertices' degree of freedom). The shapes' physical behavior is then simulated through virtual equilibrium modeling, either numerically, for instance with the finite element method, or graphically, such as in graphic statics (Maxwell, 1864; Rankine, 1864; Cremona, 1885; Przemieniecki, 1985; Bathe, 1996). Through this process, structural software, including programs such as Adina, Ansys, and Formfinder; add-ons, such as Karamba3D, RhinoVAULT, SOFiSTiK, and Revit Structure; and web-based apps, such as StructureFIT and eQUILIBRIUM, allow users to virtually analyze, simulate, and optimize an object's structural behavior (Bathe & Larsson, 1986; Azhar et al., 2008; Rippmann et al., 2012; Mueller & Ochsendorf, 2013; Preisinger & Heimrath, 2014).

Operating in virtual space, the structural software provides more freedom and flexibility to manipulate and simulate the design's structural behavior without the laborious tasks often required in a physical simulation (e.g. preparing physical models and testing equipments). Despite these advantages, it seems that association with existing physical precedents remains strongly bounded, as similar structural topologies keep recurring from the software. One popular example is the mesh-relaxation strategy used to simulate an object's deformation (e.g. membrane) under a certain load. This inclination toward a particular topology recalls Mitchell's concern, in that the objects provided in a software could lead to similar outputs across different projects. However, most structural software does not provide topological objects other than the basic structural elements and mechanical laws to run the structural simulations. Except for a few apps equipped with templates for learning (e.g. tower, bridge, and dome templates), structural software mostly lets the user model the structural problem and constraint the topology themselves.

Nevertheless, the analytical method underlying the software's subroutines does have specific ways of representing the design model. Some methods ask for a static model as an

© Cambridge University Press 2018

CAMBRIDGE UNIVERSITY PRESS

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

input, while others allow for a parametric model. This condition implies that topological similarities from structural software could be contributed partly by the software's subroutines and partly by the user's perception in representing the design problem to the software.

This study aims to investigate the way in which a structural problem is perceived and represented in order to identify their consequences and opportunities in design. In this case, we use shape grammar schema to contrast two widely used analytical methods: Finite-Element Analysis (FEM) and Graphic Statics, as well as their grammatical versions: Shape Annealing and Grammar-based Graphic Statics (Shea et al., 1997; Lee et al., 2015). Findings from this study are formulated into a new method for generating novel structures with less topological constraints.

This paper consists of four sections: first, we explain shape-grammar rule schemas and group them based on the typical modeling procedures in structural software. Second, using the rule schemas, we analyze parametric and grammatical representations of FEM and Graphic Statics. Third, we identify constraints and opportunities arising from such analysis and propose a new method for reasoning and representing shapes in structural design. Fourth, we demonstrate examples of how this new method could generate novel structures with less dependency on existing topologies. We conclude by discussing the role of shape-grammar rule schemas in analyzing and synthesizing representation mechanism in structural design strategy.

This research responds to the current state-of-the-art of shape-grammar rule schemas on investigating structural design problems from both users and software perspectives, including some of the latest developments in grammatical structure. This agenda is well suited to the visual calculation mechanism in shape grammar, as expressed in $C' = [C - t(a)] + t(b)$. Computing a shape rule $a \rightarrow b$ demands an intense visual recognition process to match the predicate shape (a) with the initial shape C via transformation $t(a)$, before applying the new shape (b) using transformation $t(b)$ to produce a new shape C'.[1]

Rule schemas

To rationalize various modeling procedures in structural analysis, we schematize their representational mechanisms with shape grammar's rule schemas. As proposed by Stiny in *Shape* (2006): "*A rule schema $x \rightarrow y$ is a pair of variable x and y that takes shapes in algebra U_{ij} as values. The values are given in an assignment g that satisfies a predicate (condition). Whenever g is used, a rule $g(x) \rightarrow g(y)$ is defined. $g(x) \rightarrow g(y)$ then applies as usual (as shape rule $a \rightarrow b$)*". Rule schemas go hand in hand with the algebra of shapes U_{ij} to define a shape based on its space (j) and spatial dimension (i) (Fig. 1b). The subscripts i and j can be 0, 1, 2, or 3; for instance, U_{12} is an algebra for a line in a two-dimensional (2D) space. Algebra which has $i = 0$ are points in space (U_{0j}), hence they are symbolic and can be directly translated into the discrete operation in a computer, whereas shapes in $i > 0$ could be perceived as a discrete and continuous shape in a user's perception, and thus they are more user-friendly. Label (V) distinguishes a shape's orientation and belonging by marking it with symbols or parts of the shape (Stiny, 1992). Shape U_{ij} works

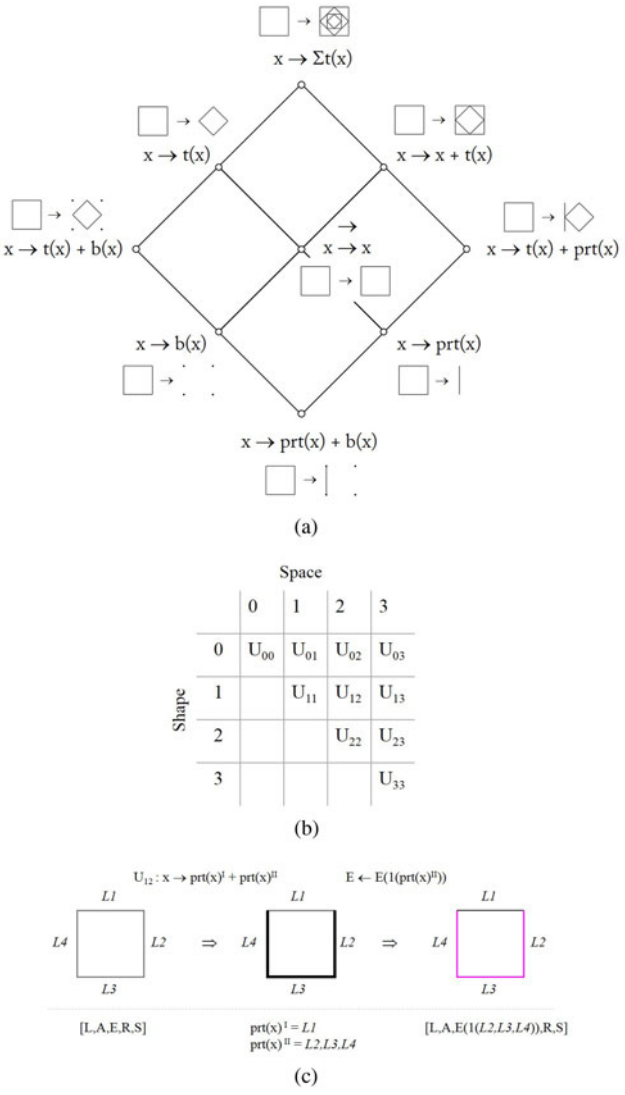


Fig. 1. (a) A lattice showing rule schemas' relationship with example shape rule (b) Algebra of Shape; (c) an example of descriptive schema iteration on registering physical description on a selected shape.

with an algebra of label V_{ij}, for instance, a label V_{12} on a shape U_{22} identify an orientation of a 2D shape with a 1D label, such as a line-label embedded in a square's edges.

The interplay between shape rules and rule schemas was discussed by Economou and Kotsopoulos (2015); whereby shape rule, for example, □ → ◇, provides straightforward visual calculations for a specific shape. Rule schema, for example, $x \rightarrow t(x)$, offers an abstractions of the rule mechanism that is applicable to different shape rule. They contrasted these differences by exemplifying the way in which different schemas distinguish various ways of seeing a shape, and how one schema translated into different shape rules. Additionally, Economou and Kotsopoulos (2015) also provide annotation for schema development, such as data registration, for example, $S_1(L_1, L_2, L_3)$ and function notation, that is, $x \rightarrow f(x)$. Rule schema's capacity in encapsulating different shape rules could help to interpret how a user might perceive a structural design from different modeling procedures and vice versa, that is, how one modeling procedure represents different designs.

[1]The account to perception and pertaining theories by William James (1981), George A. Miller (1983), and D.A. Schon (1983), and the way in which they parallel to shape grammar calculation was addressed by Stiny in Shape (2006).

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

Groups of schemas

The shape-grammar rule schemas have been used in several contexts where the schemas are grouped to suit different agendas. For example, in assisting students to communicate their ways of thinking in studio projects, Özkar (2011) classified the schemas into groups for pedagogical purposes, such as a schema for putting two shapes together $x \rightarrow x + t(x)$ and a schema for a design exercise $x \rightarrow \Sigma t(x)$. Muslimin (2012) grouped the schema based on Gestalt Law principles to re-describe how our visual perception construct a whole from the parts; for instance, a schema for visual constancy $x \rightarrow t^{-1}(x)$, a schema for visual figuration $x \rightarrow b^{-1}(x)$, and a schema for visual simplification $x \rightarrow \text{prt}^{-1}(x)$.

For our agenda, we grouped the schemas based on general structural modeling procedures, where the shape needs to be recognized and described with physical attributes before being geometrically or mechanically transformed (Bathe, 1996). The lattice in Figure 1a shows the way the following schemas are structured for our cases (see the introduction of schema lattice in Stiny, 2011).

- *Recognition Schema* (also known as identity schema) $x \rightarrow x$ defines a shape that is being visually focused without any mechanical meaning (Stiny, 1993; Krstic, 2001, 2005), while a schema $\rightarrow$ (without any shape on the left and right sides of the arrow) means do nothing with a shape (Stiny, 1996). The latter allows a shape to be idle in a parallel computation. This type of schemas allows users to embed shapes ambiguously and explore different ways of seeing the same shape before further modification with other schemas. They are centered in the lattice as the basis for the user's visual calculation in each step.
- *Transformation Schema* let the user transforms the shape according to binary operations with schema $x \rightarrow x + t(x)$, $x \rightarrow \text{div}(x)$ or $x \rightarrow x' + x''$, and $x \rightarrow y$ (e.g. addition, subdivision, subtraction, and substitution), and/or formal transformations with schema $x \rightarrow t(x)$ (e.g. translation, rotation, reflection, and scaling). The former changes the topology by generatively changing the shape during the computation, while the latter constrains the shape topology throughout the computation process. Schema $x \rightarrow \Sigma t(x)$, at the top of the lattice, repeats the transformation generatively or recursively, such as in fractal geometry.
- *Reduction Schema* allows the user to retrieve a shape from a different or the same spatial dimension. Boundary schema $x \rightarrow b(x)$ retrieves a shape's boundary from a lower dimension or a shape's content from a higher dimension $x \rightarrow b^{-1}(x)$. Part schema, $x \rightarrow \text{prt}(x)$, retrieves the sub-shapes, or the parent-shape with $x \rightarrow \text{prt}^{-1}(x)$ from the same spatial dimension. Schema $x \rightarrow \text{prt}(x)^{\text{I}} + \text{prt}(x)^{\text{II}}$ recognize two different parts of a shape. Boundary and part schemas are connected in the lattice. When $t(x)$ and $\text{prt}(x)$ map a shape to itself (e.g. with 360° rotation, or with $\text{prt}(x)$ fully occupying the parent shape), $t(x)$ or $\text{prt}(x)$ is equal to x. Under such a condition, the schema $x \rightarrow t(x) + b(x)$ can be simplified into $x \rightarrow x + b(x)$, and the schema $x \rightarrow t(x) + \text{prt}(x)$ into $x \rightarrow x + \text{prt}(x)$.
- *Descriptive Schema* registers the shape's attributes with a string function: $a \leftarrow a[D_a(x)]$, where a symbolizes shape's physical properties, D_a is the property's value, and x is a list of shapes to be attributed with a and D_a. The left arrow $\leftarrow$ specifies property's registration. The shape physical properties are stored in a list $(a_1, a_2, a_3, \ldots, a_n)$, where a_n is a pair of the property's value and the attributed shape(s) (see more about description function in Stiny, 1981). To empty the list, a function $a(:) \leftarrow$- a removes all data for property a.

For this case, our descriptive schema adapts parameters from a simple bar problem in FEM analysis, that is, the bar's length (L), sectional area (A), external force (R), supporting condition (S), and material stiffness (E) (Bathe, 1996). Thus, the description list is (L, A, E, R, S) which is equipped with descriptive functions as follows:

a_n	$a \leftarrow a[D_a(x)]$	D_a
L	$L \leftarrow L[D_L(x)]$	D_L = length scale factor
A	$A \leftarrow A[D_A(x)]$	D_A = sectional-area scale factor
E	$E \leftarrow E[D_E(x)]$	D_E = degree of material elasticity
R	$R \leftarrow R[D_R(x)]$	D_R = force's vector
S	$S \leftarrow S[D_S(x)]$	D_S = supports' degree of freedom

The description list can be updated with new data. For example in Figure 1c, applying a function $E \leftarrow E[1(L2, L3, L4)]$ will register rigid material into sub-shapes $L2$, $L3$, $L4$ and update the list into $\{L, A, E[1(L2, L3, L4)], R, S\}$ (in this paper, we simplify the material elasticity's value into $D_E = 1$ means rigid; $D_E = 0$ means flexible).

With descriptive schema, the data and computation history can be recorded for further use, such as for CAD programming or documenting a user's creative process during the computation. This could be a useful companion, as shape grammar computation is memoryless, in the sense that a shape can be perceived as if a user sees the shape for the first time, and attributes in the previous steps can be ignored in the next step.

Parametric and grammatical structure revisited

With the group of rule schemas, in the following sections, we interpret the classical FEM and graphic static representations and their grammatical versions, that is, shape annealing and grammar-based graphic statics. To contrast the differences between them, the same initial shape in algebra U_{12}, , is used to see how the shape is recognized, described, and transformed under different methods (Fig. 2).

Parametric iteration in the classical FEM and graphic statics

Classical FEM

In a simple bar problem in Figure 2a, a physical shape is represented as an assemblage of lines (U_{12}) before being applied with the mechanical attributes. The process of how the user follows the framework in FEM can be represented with schema $x \rightarrow x + b(x)$ to return the line's endpoints. Descriptive schemas then register their physical attributes to describe that the scaling factors for all shapes' lengths and sectional areas are 1 (i.e. no scaling), stiffness for all shapes is 1 (rigid), the force vector applied to a point P1 is −1 (downward), and the support degree of freedoms for points P2 and P3 are 0 (fixed support):

With this registration, the shape's physical behavior is mathematically obtained by equating the internal and external forces based on the principle of virtual equilibrium $dP = LR/EA$, which transforms the points (P) into new positions (dP), represented with the schema $x \rightarrow t(x)$. Note that, in this method, the shapes

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

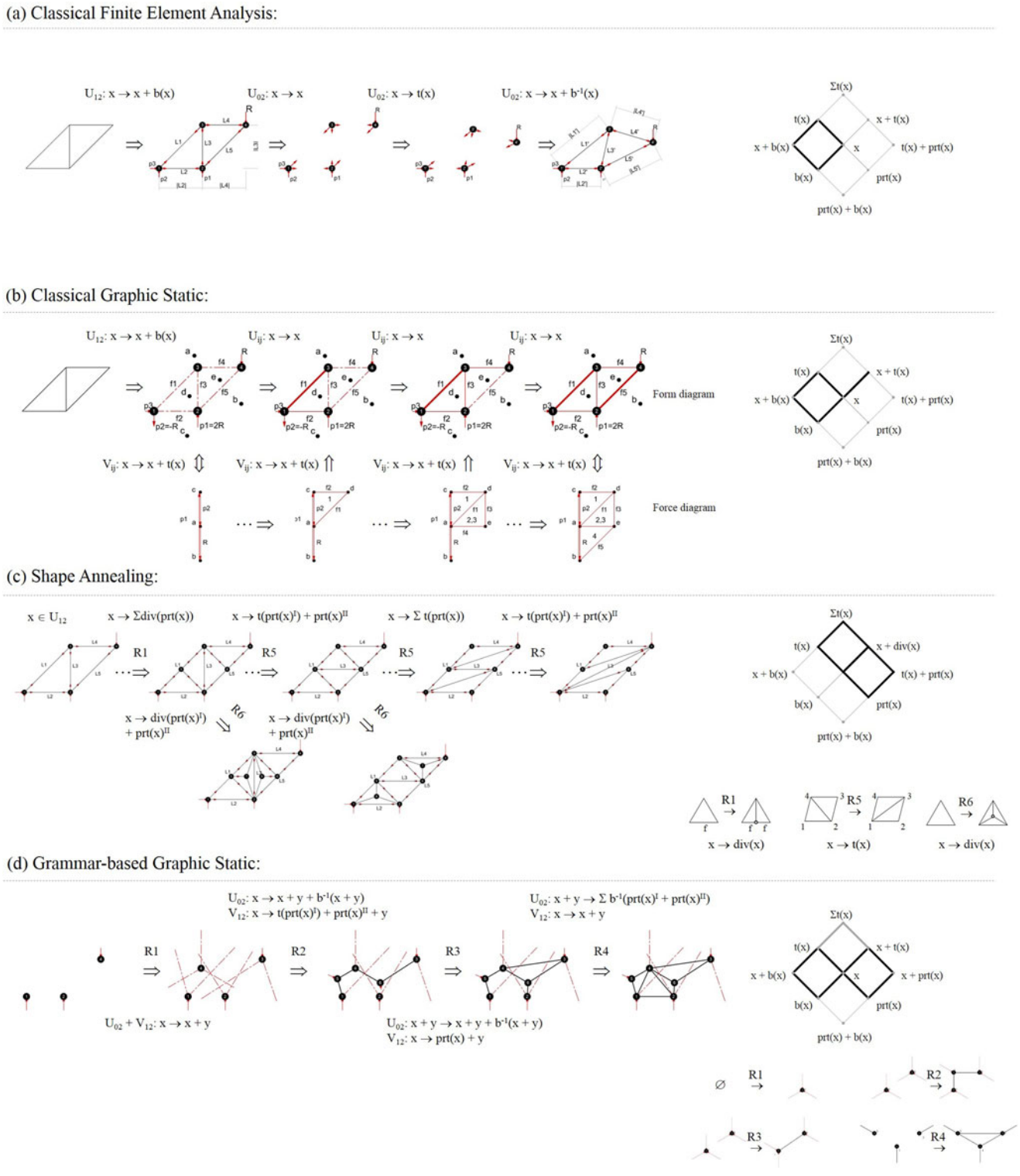


Fig. 2. Parametric and Grammatical Structural Synthesis, with the lattice on the right indicating the dominant schemas used in each operation: (a) A Classical Finite Element Analysis procedure on representing a design into an assemblage of discrete lines and vertices. The physical behavior is mathematically simulated in a zero-dimensional space before visualized back into a higher spatial dimension; (b) In Classical Graphic Static, a shape is identified as an assemblage of point, line, and polygon along with the applied forces in a form diagram (first row) and then force diagram is developed accordingly to represent and graphically calculate the force acting on a shape's (second row); (c) Hypothetical Shape Annealing iteration using three of the original rules (Shea & Cagan, 1997). The initial shape is labeled and embedded with mechanical attributes. (d) Hypothetical Grammatical Graphic Static iteration. Instead of using a predefined design, it started with initial supporting and loading condition on several points. Using a set of rules (Lee et al., 2015), the grammar develops a series of forces lines to be chosen as a predicate shape for the subsequent steps.

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

physical behavior is analyzed in the algebra of U_{02} (symbolic). Based on the new positions, schema $x \rightarrow x + b^{-1}(x)$ returns the lines to visualize the simulation results showing the initial line's length (L) changed into $L + \Delta L$.

Classical graphic statics

Graphic statics represent the shape in two corresponding diagrams: form diagram, representing the shape and the applied forces, and force diagram, representing the forces' relationship that is proportional to the forces' orientation and magnitude in the form diagram (Fig. 2b). The form diagram simplifies the actual shape into vertices and lines, recognize the force-line acting on shapes, and label them with descriptive schema, similar to the FEM descriptive process. However, instead of analyzing the shape in the algebra U_{oj}, classical graphic statics looks at how the forces are graphically arranged in the force diagram. We consider the force in the form diagram as a label. As such, the schema $x \rightarrow x + t(x)$ copies all labels V_{ij} from form diagram into force diagram, such that:

- all forces-line acting on points U_{02} are registered as line-label V_{12}.
- all forces-line acting on lines U_{12} are registered as line-label V_{12}.
- all polygons U_{22} between the acting forces-line are registered as point-label V_{02}.
- all points U_{02} are registered as polygon-label V_{22}.

In the force diagram, the line-label's length is considered to be proportional to the force magnitude in the form diagram (e.g. 1 kN = 1 cm). The computation order follows a certain direction, for example, compute the external forces first in a clockwise direction, before the internal forces. The external loading and supporting forces are in equilibrium ($\Sigma F_h = 0$; $\Sigma F_v = 0$; $\Sigma M = 0$), which is achieved by graphically compute the known external forces' magnitude to develop a reciprocal force diagram. The force diagram is established when all point-labels, line-labels, and polygon-labels are defined, and all forces acting in the form diagram are proportional to the force diagram within a reciprocal configuration. As the form and force diagram are parametrically linked, changes in one diagram affects another simultaneously, for example, a shorter line in the force diagram means a lower force in the form diagram, or a long line in the force diagram means a thicker line (i.e. A+ΔA) in the form diagram (see digital application in StructureFIT and eQUILIBRIUM).

The rule schemas provide an opportunity to interpret analytical improvement, such as analyzing a 3D structure instead of a 2D one on a 3D graphic static based on Rankine theorem (See Akbarzadeh et al., 2015). Instead of representing force lines into vectors in a force diagram, schema $x \rightarrow b^{-1}(x)$ can represent how the force diagram may shift from a 2D into a 3D representation. For example, in a 3D truss in Akbarzadeh et al. (2015), an inverse-boundary schema $b^{-1}(V_{12})$ turns line-labels into plane-labels (V_{23}) perpendicular to the bar, where the plane's area in the force diagram is proportional to the bar's force magnitude in the form diagram.

Grammatical iteration on shape annealing and grammar-based graphic statics

As seen in the previous section, a parametric iteration in FEM and graphic statics transform the shape with schema $x \rightarrow t(x)$ and $x \rightarrow b(x)$; therefore, the topology remains the same. In this section, we use rule schemas to represent how shape annealing and grammar-based graphic statics incorporate shape rules to liberate the topological constraints.

Grammar-based FEM in shape annealing

The shape annealing method integrates shape rules with the FEM, along with a simulated annealing algorithm to generate various structural designs that satisfy certain objectives and constraints (Shea & Cagan, 1997, 1999; Shea et al., 1997). Most of its rules can be represented with the subdivision schema $x \rightarrow \mathrm{div}(x)$, while others have addition, transformation, and erasing schemas: $x \rightarrow t(x)$; $x \rightarrow x + t(x)$; $x \rightarrow$, including their inversed version: $a \rightarrow b$ and $b \rightarrow a$ (Suppapitnarm et al., 2004). While deriving various topological variants, the rules are able to preserve a certain design language. For instance, the subdivision rule on a closed polygon would typically generate the shape outside-in, thus maintaining the initial shape's outlines (Fig. 2c). Additionally, the label and the physical description are mostly predefined with shape rules (e.g. whether a point's degree of freedom is fixed or free).

Figure 2c shows hypothetical shape annealing iteration using the same initial shape as in the previous section. After the shape is grammatically transformed in one step, it is evaluated through multicriteria optimization before the next step. If the result does not satisfy certain objectives, then another transformation rule will be applied, and so on. As shape annealing rules are equipped with $x \rightarrow x + t(x)$ and $x \rightarrow x$ to match a shape in different ways and to transform the design generatively, the computation will likely produce various structural topologies under a predefined design language.

Grammar-based graphic statics

Grammatical graphic statics incorporate parametric shape rules into the classical graphic statics method (Lee et al., 2015). Similar to shape annealing, shape rules in grammatical graphic statics are bounded to applied load and reaction forces (Fig. 2d). The rules ensure that the action-reaction force in each iterative step will be close to equilibrium state (i.e. have a reciprocal configuration in the force diagram).

While the form diagram in classical graphic statics is known, the form of grammatical graphic statics is unknown, thus becoming the goal of the rules generation. In this sense, given loading and supports positions, this method explores what kind of shape can be generated by the rules? The ideation process behind this premise can be interpreted by applying an inversed-schema of the classical graphic statics schemas (Fig. 3).

- Schema $(x + y) \rightarrow (x + y) + t(x + y)$ copies the labels into a force diagram for structural analysis and synthesis. In the force diagram, the line-label is transformed with schema $x + y \rightarrow x + t(y)$ into a nearly equalized reciprocal configuration by leaving one line-label undrawn, hence, leaving the options for users to decide how the force lines will be finalized in the form diagram. As line-label of x and y are intersected at a point in force diagram, it implies their force-lines share a polygon in the form diagram. This condition suggests a new force line is required in the form diagram to close the unfinished polygon-label in the force diagram.
- After the user finalized the form with $x + y \rightarrow x + y + z + b^{-1}(x + z) + b^{-1}(y + z)$, that is, by defining the new intersection point and the new force-line acting on the point, schema $x \rightarrow x + b^{-1}[\mathrm{prt}(x)]$ then registers the new force-line into the force

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

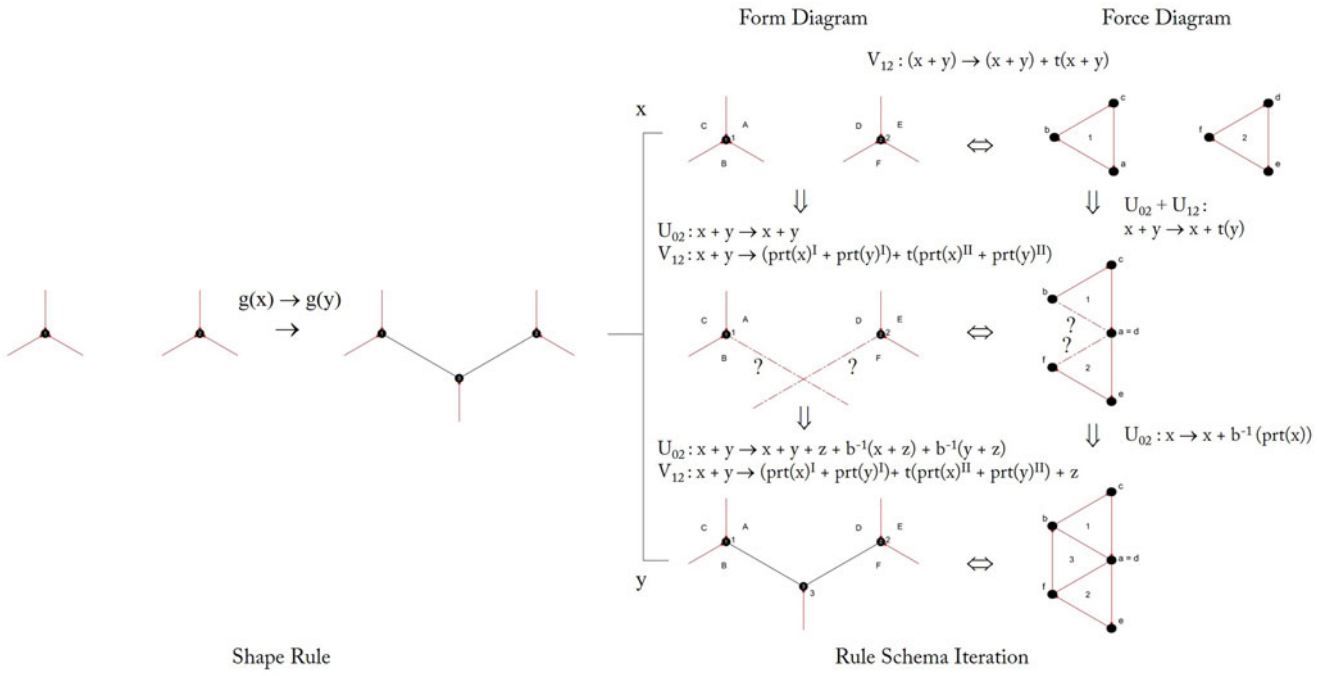


Fig. 3. Interpretative representation of the rule-synthesis for rule R2 in the Grammatical Graphic Static. The initial force and form diagram show two points applied with forces as the initial condition (top-right). Two force diagrams are connected and share a point in a force diagram, indicating that their force-lines also share polygon in a form diagram (middle-right). New lines and force are established in the form diagram, which closes the unfinished polygon in the force diagram (bottom-right). The initial condition and the outcome are assigned to a new shape rule (left).

diagram. As the new line corresponds to the equalized configuration in the force diagram, the structure will likely be stable and correspond with other forces in the form diagram.

- The initial shape and the final form are then formalized into a parametric shape rule $a' \rightarrow b'$ using assignment $g(x) \rightarrow g(x) + g(y)$, with x value is the initial condition of two separated points, each of which applied with force-lines, and y value is the addition of a new third point applied with a new force and connected to the initial two points through a new line.

Each time the rules are applied, new force lines will be created to be matched to the subsequent iteration. Parameters such as the length between points, and angles between lines in the rules allow for incremental variations. As such, grammatical graphic statics offer two matching processes: [1] which force-line to match with, and [2] which rules to apply for the chosen force-lines.

Perceptual structure

Findings from the four cases are contrasted in the lattice in the right side of Figure 2. As discussed earlier, the transformation schemas were intensively used in a different manner. While classical graphic statics and FEM use parametric transformation $x \rightarrow t(x)$, where label and description are manually registered, shape annealing and grammar-based graphic statics use generative transformation $x \rightarrow x + t(x)$, with labels and descriptions predefined in the rules. The study also shows that recognition schema $x \rightarrow x$ was more active in the grammatical method rather than in parametric method. The shape's algebra for analysis is mostly consistent (e.g. U_{0j} for FEM and U_{12} for Graphic Static). Except for grammar-based graphic static, the use of reduction schema is minimal as it is applied only for reducing the shape into different algebra for graphical or numerical analysis.

Based on these findings, we outline a method that intensifies the role of user perception and investigate the potential use of the underutilized schemas in the existing methods. Perceptual iteration, relating to the user's ability to sense potential shapes visually and to extend possible physical manifestation of what they see, is driven by intensifying recognition, reduction, and descriptive schemas. In our proposal, the groups of schemas are loosely tied to one another to provide room for visual and physical ambiguity. In this sense, users would have more flexibility to make pivotal decisions that would impact on the design topology, such as which shape to focus on, which spatial dimension to operate with, and which physical description to embed.

Figure 4 shows how allowing spatial and mechanical ambiguity could rapidly extend the design topology in early stages. The initial shape is recognized as two lines in algebra U_{1j}, and the boundary schema $x \rightarrow b(x)$ turns it into points (Fig. 4a, b). Perceptual iteration begins by applying schema $x \rightarrow x$ and $x \rightarrow \text{prt}(x)$ to recognize the points in a different order {e.g. (p_1, p_2, p_3, p_4) or $[(p_3, p_1)\ (p_4, p_2)]$}, and then the inverse boundary schema $x \rightarrow b^{-1}(x)$ is applied based on this order (Fig. 4a–c). As a result, the schema not only returns the initial shape from the points but also yields other linear and planar compositions in the higher dimensions (see an alternating example of seeing and doing in Knight & Stiny, 2015). Ambiguity continues to catalyze design variations when registering the physical description. With a different visual attention to the part of shapes, applying different stiffness with schema $E \leftarrow E[1(x)]$ or $E \leftarrow E[0(x)]$ leads to different structural configurations (Fig. 4d, e).

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

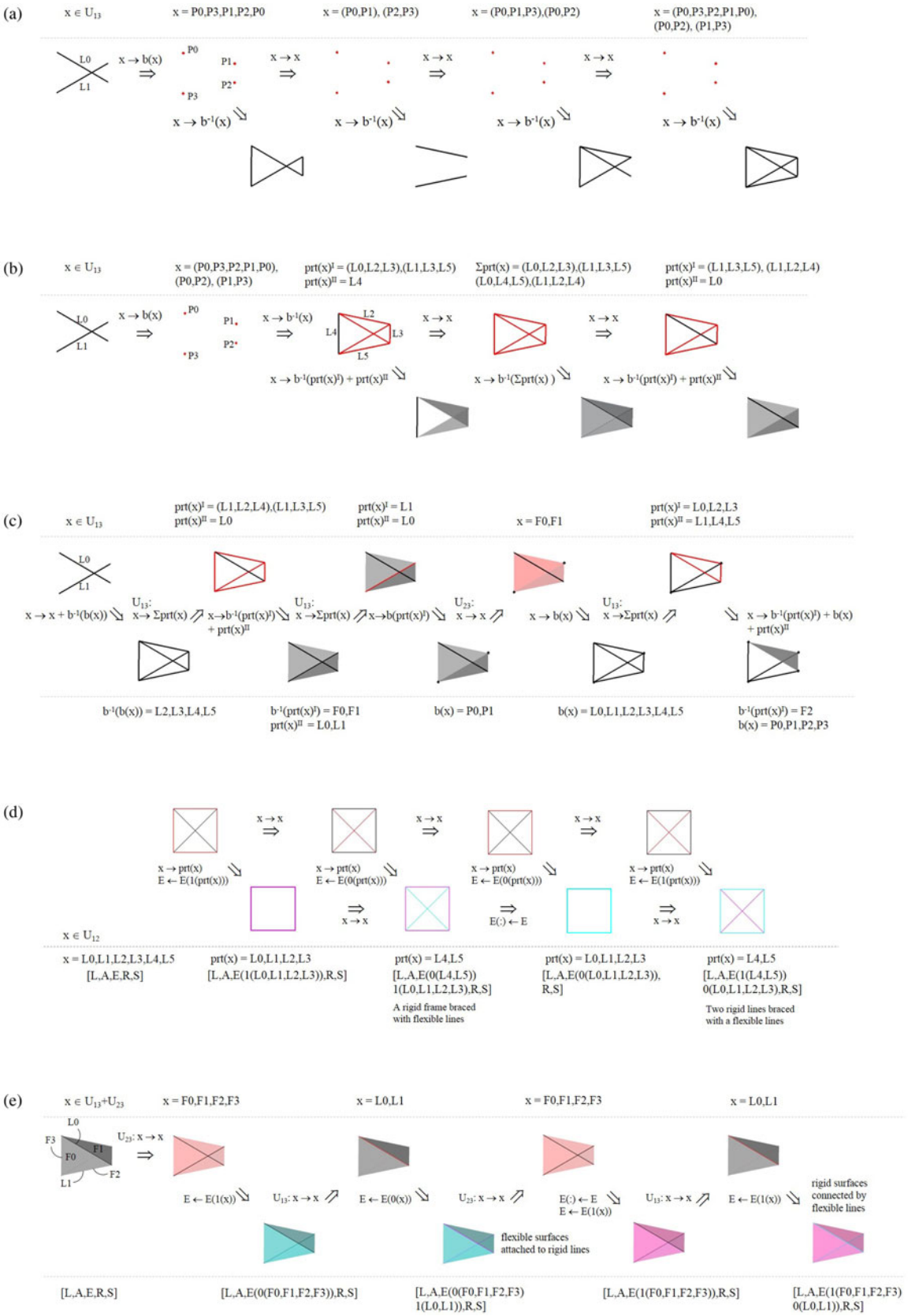


Fig. 4. (a–c): Recognition and boundary schema allows for different ways of perceiving the shape, which results in various configurations when followed by the inverse-boundary schema. (d, e) Recognition schema followed by Descriptive schema on applying the material property generates different structural configuration.

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

Case study

In the following case study, we demonstrate perceptual iteration and discuss its extended features, such as function, inverse-function, and recursion.

Figure 5 shows how perceptual iteration generates different structural topologies by changing the shape's description and the shape's algebra through reduction, recognition, and descriptive schemas. Consider the initial shape at the top of Figure 5 as two overlapping lines retrieved from a weaving configuration using the part schema. The description list (E, S) has an initial data $\{E[1(L0, L1)], S\}$ which reads as "lines $L0$ and $L1$ are rigid." Note that, in this example, we only show the properties that are modified, that is, material stiffness (E) and support's degree of freedom (S), and leave the other unmodified properties, L, A, and R, hidden.

Through steps 1–3, the iteration is as follows: Schema $x \rightarrow x + b^{-1}[b(x)]$ retrieves new lines from the existing line endpoints ($L2$, $L3$, $L4$). Schema $E \leftarrow E[0(x)]$ registers a flexible material to the new lines and appends the description into $\{E[0(L2, L3, L4), 1(L0, L1)], S\}$. These three steps change the original configuration, where the two rigid lines are now tensioned by three new flexible lines. By applying schema $x \rightarrow \Sigma t(x)$ to this module, we then have a new design that resembles a tensegrity structure.

The computation from steps 3 to 5 started with schema $x \rightarrow x + b^{-1}(x)$ to retrieve two faces from existing line, thus x value for algebra U_{23} is $F0$ and $F1$. Here, the previous material attributes are ignored with schema $E(:) \leftarrow E$ empties the material data, thus returning it back to (E, S). The schema $E \leftarrow E[1(x)]$ register rigid material to the faces, and schema $S \leftarrow S[1(x)]$ turn all lines into a hinge which rotates around one axis. The list now consists of $\{E[1(F0, F1)], S[1(\Sigma Ln)]\}$, which changes the tensegrity module into two faces connected with all lines as a hinge support. Applying $x \rightarrow \Sigma t(x)$ to this module will return a folding configuration (see

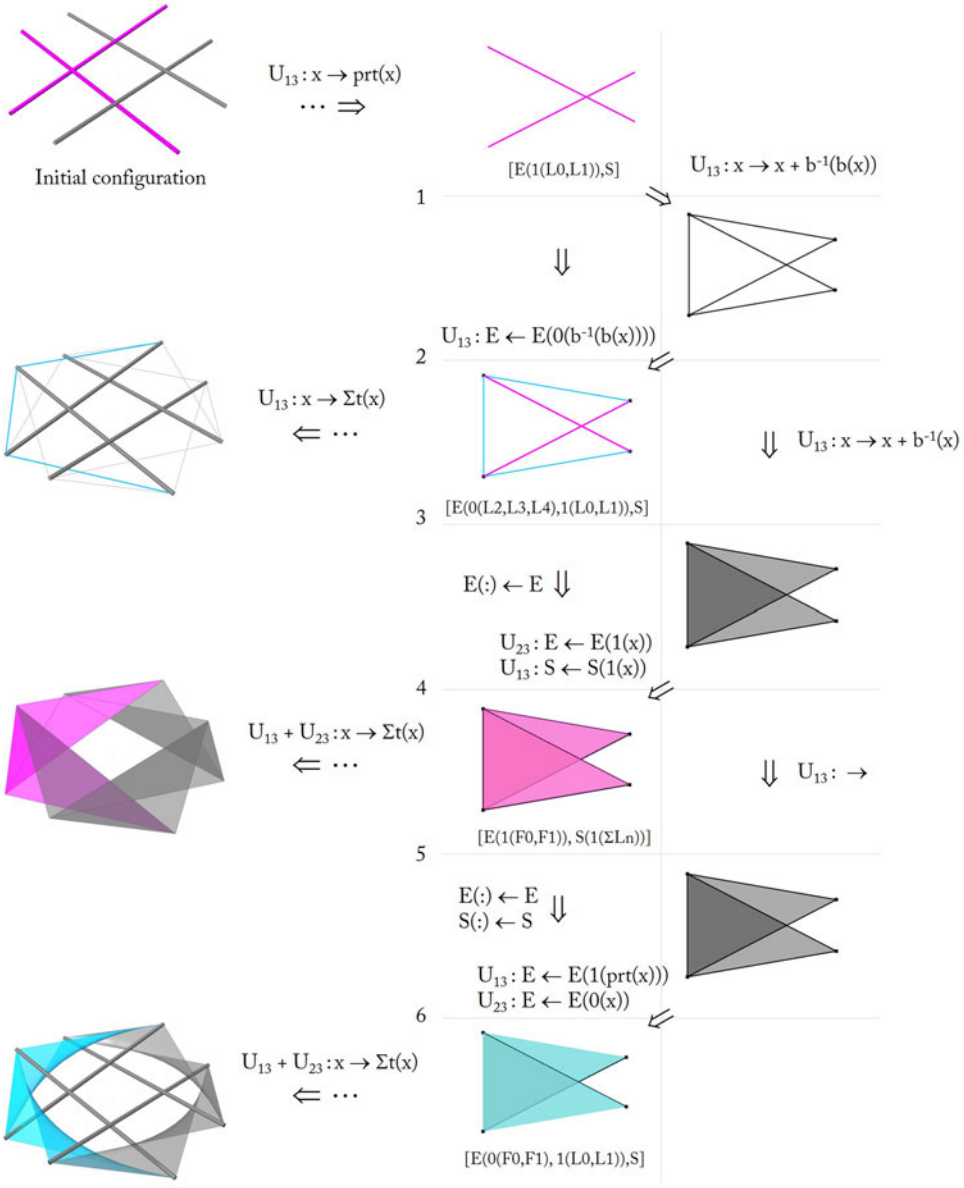


Fig. 5. Perceptual iteration on shifting one's perception on the traditional craft by alternating across different algebra with boundary and part schemas, and registering different physical description to the shape with descriptive schemas.

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

folding example in Jackson, 2011, and extended iteration in Muslimin, 2016, 2017)

Function schema

As the data and iterative steps in Figure 5 are explicit and recorded, the schema iteration can be compiled into functions to replicate the new topology for initial description: $\{E[1(\Sigma \mathrm{Ln})], S\}$, and predicate: x is a set of interwoven lines in 3D space ($x \in U_{13}$). As listed below, function $f_1(x)$ turns weaving into tensegrity and $f_2(x)$ converts weaving into folding.

Function	Compiled schema
$f_1(x)$: $x \Rightarrow \mathrm{prt}(x) \Rightarrow \mathrm{prt}(x) + b^{-1}\{b[\mathrm{prt}(x)]\}$ $E \leftarrow E\{0[b^{-1}(b(\mathrm{prt}(x)))]\}$ $y = \mathrm{prt}(x) + b^{-1}\{b[\mathrm{prt}(x)]\}$ $x \Rightarrow \Sigma t(y)$	$x \rightarrow \Sigma t\{\mathrm{prt}(x) + b^{-1}[b(\mathrm{prt}(x))]\}$ $E \leftarrow E\{0[b^{-1}(b(\mathrm{prt}(x)))]\}$
$f_2(x)$: $x \Rightarrow \mathrm{prt}(x) \Rightarrow \mathrm{prt}(x) + b^{-1}\{b[\mathrm{prt}(x)]\}$ $y = \mathrm{prt}(x) + b^{-1}[b(x)]$ $y \Rightarrow y' + y'' \Rightarrow b^{-1}(y' + y'') + (y' + y'')$ $E \leftarrow E\{1[b^{-1}(y' + y'')]\}$ $S \leftarrow S(1(y' + y''))$ $z = b^{-1}(y + y'') + (y' + y'')$ $z \Rightarrow \Sigma t(z)$	$y = b^{-1}[b(x)] + \mathrm{prt}(x)$ $y \rightarrow \Sigma t(b^{-1}(y' + y'') + (y' + y''))$ $E \leftarrow E\{1[b^{-1}(y' + y'')]\}$ $S \leftarrow S[1(y' + y'')]$

The double-arrow ⇒ in the function indicates a schema iteration, and the single arrow → designates a compiled schema. Note that the x value for the schema iteration in Figure 5, is the output from the last step. Hence the left-hand-variable is always reset to x. Inside the function, however, the variable's notation [e.g. $b(x)$ or $\mathrm{prt}(x)$] are preserved and nested in the subsequent steps to express how the visual calculation is progressing from the predicate. A new variable is introduced when the last output in the iteration is considered as a new shape (y).

As variables inside the function are modifiable, these functions serve as an open source to develop other possible topologies for inventive use. For instance, if hinges in $f_2(x)$ are substituted into a fixed joint, $S[0(y' + y'')]$ instead of $S[1(y' + y'')]$, the function will produce a rigid folded-plate structure (Engel, 1968; Weinand, 2009). Alternatively, instead of applying the rigid material to the faces in $f_2(x)$, we can apply a flexible material to the faces using $E \leftarrow E\{0[b^{-1}(y' + y'')]\}$. As seen in the output in step 6, with the same rule schema iteration, this tweak creates a new function $f_3(x)$, to develop a textile-tensegrity configuration from weaving (Peña Villamil et al., 2011).

Function	Compiled schema
$f_3(x)$: $x \Rightarrow \mathrm{prt}(x) \Rightarrow \mathrm{prt}(x) + b^{-1}\{b[\mathrm{prt}(x)]\}$ $y = \mathrm{prt}(x) + b^{-1}[b(x)]$ $y \Rightarrow y' + y'' \Rightarrow b^{-1}(y' + y'') + (y' + y'')$ $E \leftarrow E\{1[\mathrm{prt}(y' + y'')]\}$ $E \leftarrow E\{0[b^{-1}(y' + y'')]\}$ $z = b^{-1}(y + y'') + \mathrm{prt}(y' + y'')$ $z \Rightarrow \Sigma t(z)$	$y = b^{-1}[b(x)] + \mathrm{prt}(x)$ $y \rightarrow \Sigma t[b^{-1}(y' + y'') + \mathrm{prt}(y' + y'')]$ $E \leftarrow E\{1[\mathrm{prt}(y' + y'')]\}$ $E \leftarrow E\{0[b^{-}(y' + y'')]\}$

To convert rule schema into a shape rule, Stiny (2006) shows how assignment $g(x) \rightarrow g[t(x)]$ turns rule schema $x \rightarrow t(x)$ into shape rule $a \rightarrow b$ (Stiny, 2006; Economou & Kotsopoulos, 2015). In a similar manner, initial input from x, and outputs from $f_2(x)$ and $f_3(x)$ can be abbreviated into a shape rule: $a \rightarrow b + c$, with function assignment, $x \rightarrow f_n(x)$ and predicate: a is the overlapping composition of n lines (weaving), b is the rigid surface based on a's inversed boundaries (folding), and c is a combination between the flexible surface from on a's inversed boundaries and a's lines' composition (textile-tensegrity). This shape rule format is more practical and straightforward for quick application. The examples in Figure 6 show parametric-grammatical rule iterations along

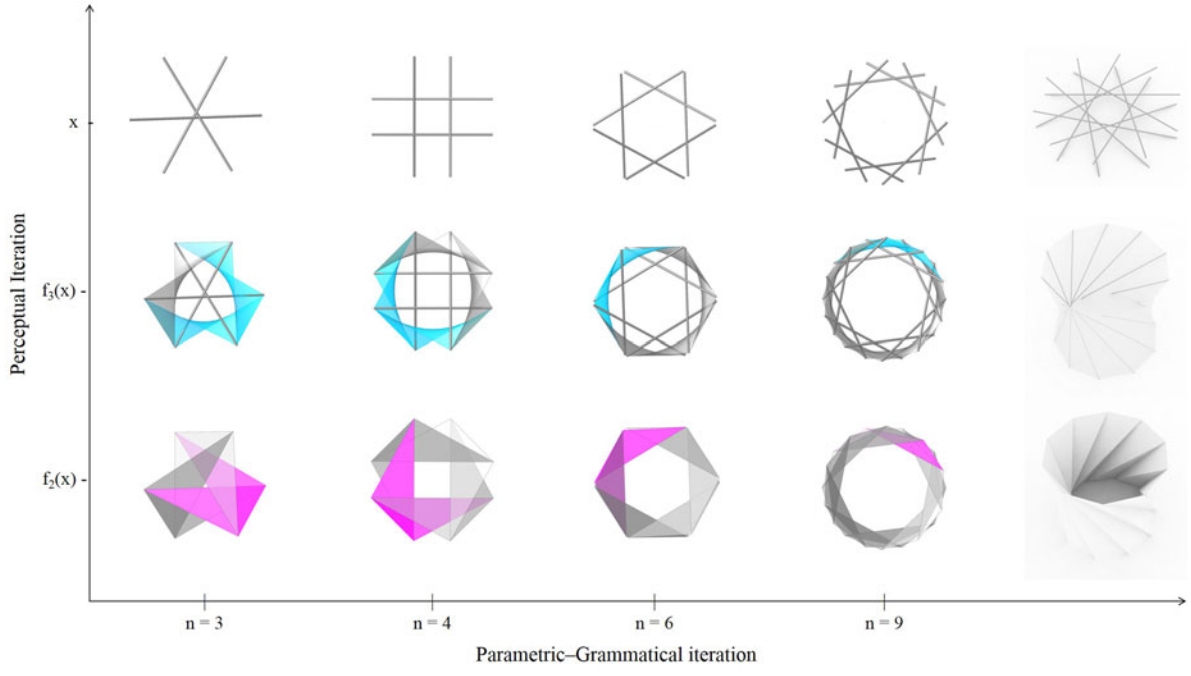


Fig. 6. The iteration of Weaving-Tensegrity-Folding Function. Modification on one craft affects another type of craft's simultaneously.

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

with perceptual iteration, where the modification in weaving (*a*) simultaneously alters the iteration in folding (*b*), and textile-tensegrity (*c*).

Inversed function

With inverse schema $A \rightarrow B$ and $B \rightarrow A$, the schema iteration within a function can be reversed to develop yet more functions. For example, function $f_4(x)$ below converts tensegrity to folding from steps 3 to 5 in Figure 5. The initial description is: {*E*[0(*L*2, *L*3, *L*4)], 1(*L*0, *L*1), *S*}.

Function		Inversed function
$f_4(x)$: x = axial-lines in tensegrity $x \Rightarrow x + b^{-1}(x' + x'')$ $E \leftarrow E\{1[b^{-1}(x' + x'')]\}$ $S \leftarrow S[1(x)]$ $S[1(x)] \leftarrow S[\mathrm{Cv}(x)]$ or $S[1(x)] \leftarrow S[\mathrm{Cm}(x)]$ $y = x + b^{-1}(x' + x'')$ $y \Rightarrow \Sigma t(y)$	inverse and substitute ⇒	$f_5(x)$: x = surface-modules in folding $x \Rightarrow b(x)$ $E(:) \leftarrow$ $S \leftarrow E$ $\mathrm{Cv} \leftarrow 1$ $\mathrm{Cm} \leftarrow 0$

The steps of retrieving a face from a line in $f_4(x)$ can be inversed from $x \Rightarrow x + b^{-1}(x)$ into $x \Rightarrow b(x)$. As a result, we have a new function $f_5(x)$ to convert the folding module into a tensegrity module. Figure 7 demonstrates this new function by using the Miura fold as the initial sample (Miura, 1985; Muslimin, 2016). A typical folding input is annotated with mountain creases (C_M) and valley creases (C_V), which indicates a series of hinge support (S) between faces on a 2D paper, such that $\{S[C_v(\mathrm{prt}(x)^I), C_m(\mathrm{prt}(x)^{II})]\}$; $x \in U_{12}$. Steps 1–2 show the steps in producing a new structural configuration where all valley creases become flexible line (e.g. cables) and all mountain creases become rigid line (e.g. bars). The descriptive function converts the creases into either rigid or flexible lines, which change the description in Figure 7 from:

$$[E(1(\Sigma F_n)), S(C_v(L1, L8, L9, L3, L4, L5, L6), C_m(L2, L11, L12, L10, L7))]$$
$$\text{into } [E(1(L1, L8, L9, L3, L4, L5, L6), 0(L2, L11, L12, L10, L7))].$$

Recursive schema

The schemas can continue to be applied in the subsequent steps in order to generate more structural topologies. Starting from step 2 in Figure 7, the iteration shows the boundary schema recursively applied to the new tensegrity modules. Applying the schema $x \rightarrow x + b^{-1}\{b[\mathrm{prt}(x)]\}$ returns new diagonal lines from the lines' end-points, and then, in step 4, another schema $x \rightarrow b^{-1}[\mathrm{prt}(x)^I] + \mathrm{prt}(x)^{II}$ returns new faces (the arrowheads). For each new shapes from steps 3 and 5, descriptive schema $E \leftarrow E[D_E(x)]$ alternates between registering rigid and flexible material to derive new structural modules.

Transformation schema $x \rightarrow \Sigma t(x)$ multiplies the 2D modules into 3D modules, from which inverse boundary schema $b^{-1}(x)$ can be reapplied to different parts of shape to create yet more new modules. The result is a series of space structures with a combination of rigid and flexible material that could respond to certain architectural requirements, such as enclosure (see left images in Fig. 7).

Structural simulation and physical validation

Figure 8 shows FEM analysis of four models from the above iteration. The transformation schema $x \rightarrow \Sigma t(x)$ applied to the four models is the same. The models were rotated within a triangular base and stacked vertically four times. All four models have three fixed supporting points at the base, and three vertical loading points at the top with the same magnitude. While the stiffness of the cable, rigid bar, rigid surface and flexible membrane are the same across the four models, the main difference is in the mechanical configuration at the modular level.

- Model A has vertical rigid surfaces, supported by flexible cables.
- Model B has vertical rigid surfaces, supported by flexible membranes.
- Model C has horizontal flexible membranes which support trusses of rigid and flexible bar (thick bars are rigid).
- Model D has horizontal flexible membranes attached to the vertical rigid bar.

All models were generated digitally, without initially considering their structural consequences. This leads to some peculiar model outputs, for example, the condition where the rigid surfaces in model B are supported only by flexible membranes that look structurally implausible. However, FEM analysis shows an opposite result. Out of the four, model B appears to have the smallest displacement of all (0.381 cm out of the total height 3 m) and model A has a larger displacement (3.41 cm), while model C and D have a very large displacement, and thus collapse.

To validate these results, physical models (models M and N) were built to represent the configurations for models A and B (Fig. 9). The transformation schema is the same, except that the modules were stacked three times instead of four times, and the base for model M is square and model N is a hexagon. Therefore, these physical models have more supporting points and are shorter than their virtual versions.

The physical validation shows that, although model M and N may look rather unusual, they are physically stable. The rigid surfaces in model M are attached with aluminum ball-chains, while the rigid surfaces in model N are attached with black fabrics. As the chain has small strains and the fabric has large strains, model *N*'s fabric is stretched slightly downwards due to its own surface weight, yet it continues to support the surfaces well.

Conclusion

In this paper, we have illustrated and discussed different structural representation mechanisms using shape-grammar rule schemas. The schemas' versatility for analyzing existing methods, as well as synthesizing a new method, is threefold:

First, the rule schema explicitly represents procedures underlying the structural method and interprets users' perceptions through representing their design intentions. The schema representation indicates that an inclination toward a certain structural topology might be contributed by the modeling principles applied during the iteration process, that is, parametric and grammatical modeling. In parametric iterations, such as classical FEM and graphic statics, the design is optimized incrementally to achieve the equilibrium state while maintaining the shape's topology and the mechanical description. Grammatical iteration, on the other hand, alters the topologies using a set of generative rules. The

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

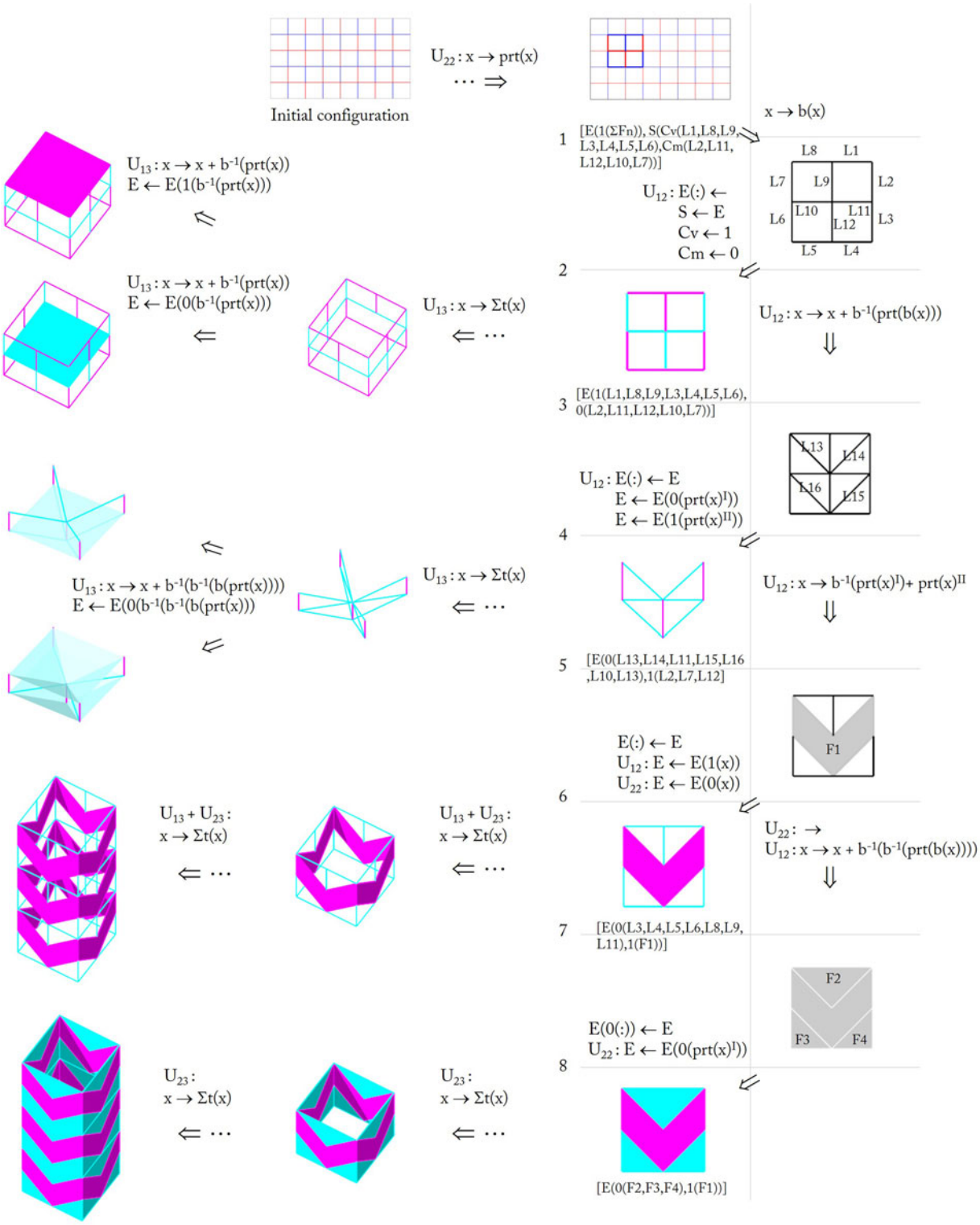


Fig. 7. Perceptual-iteration process from Miura Folding crease to its Tensegrity version. Miura modules were recursively applied with reduction and descriptive schema to develop new tensegrity modules (right). Transformation schema is applied to generate 3D structures from these modules (left).

rules might preserve a certain design language, as shown in the shape annealing method, or promote random designs, as seen in grammar-based graphic statics. Nevertheless, both methods work under predefined constraints, where parametric iteration constrains the shape relationship, grammatical iteration constrains the physical description.

Second, the rule schemas analysis suggests opportunities for new approaches in structural representation by highlighting some underutilized schemas. While the schemas in parametric and grammatical iterations mostly attached to physical description and operating in constant algebra, perceptual iteration loosened this attachment to foreground the role of user perception and ambiguity in deriving

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

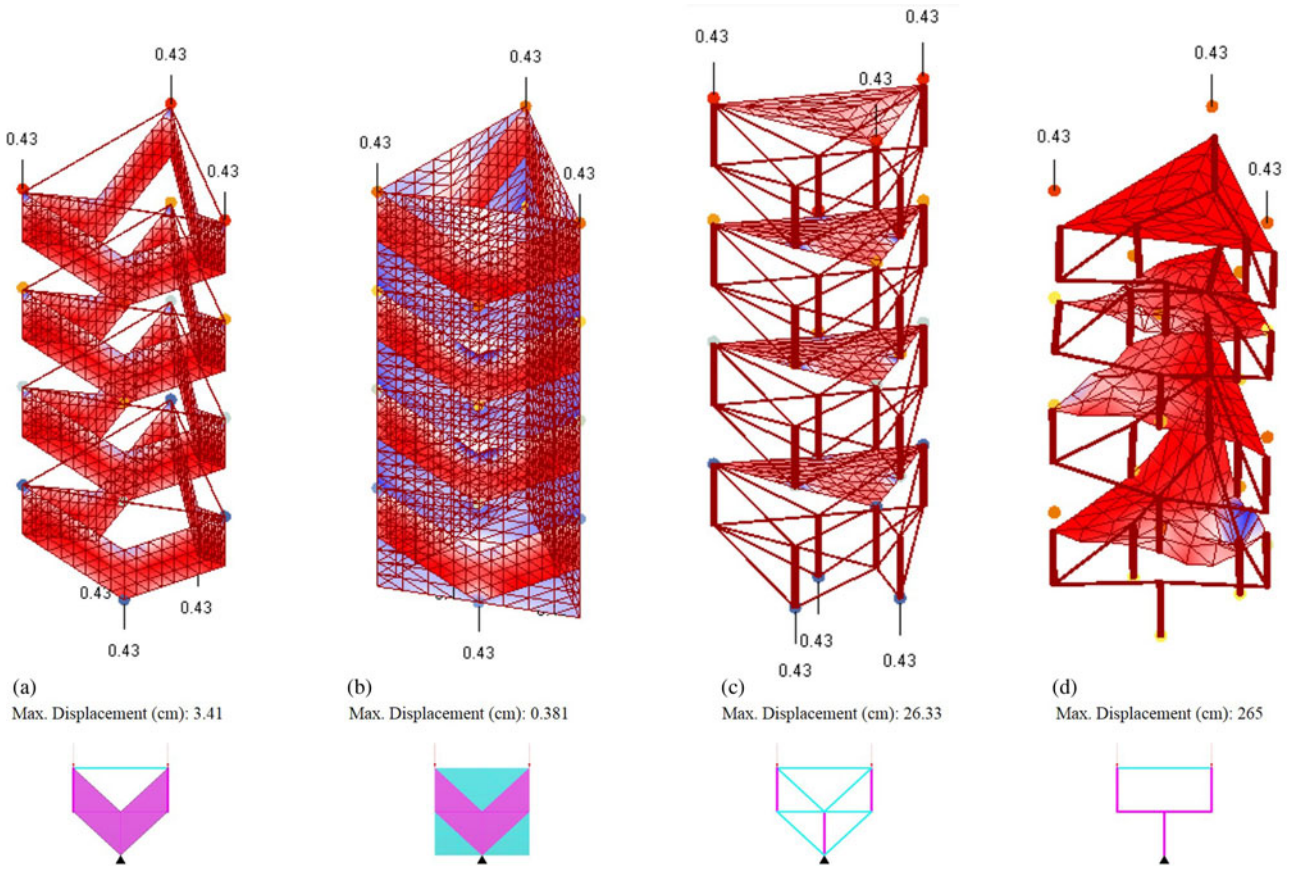


Fig. 8. Finite Element Analysis on assessing several design results. All models have a triangular footprint and receive the same external load. Model A (rigid wall with cable) and B (rigid wall with fabric) are stable while model C and D collapsed.

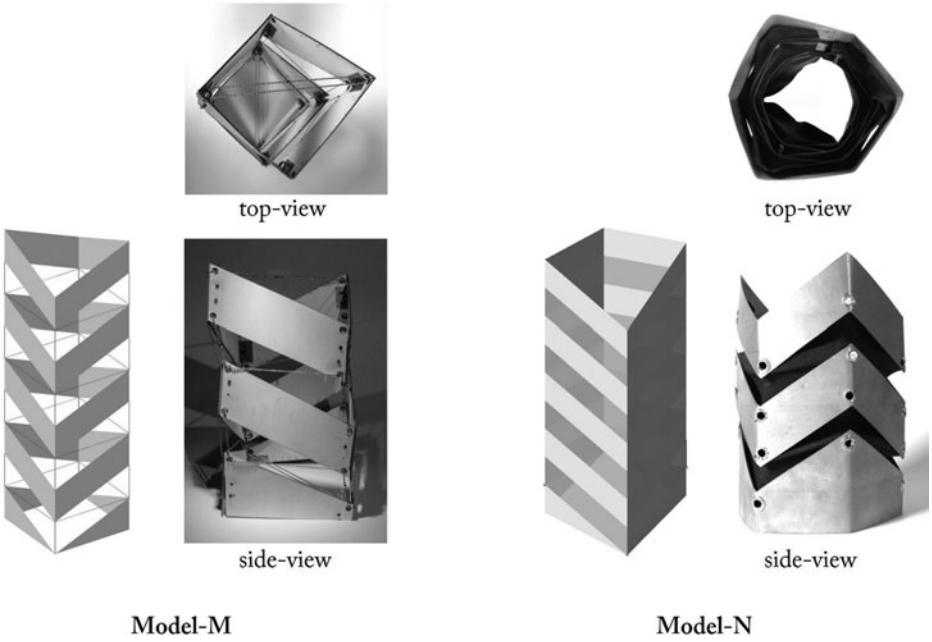


Fig. 9. Examples of physical validation from Miura Folding to Tensegrity iteration. Model M is based on model A consisted of a series of the rigid panel supported by chains (left), while in model N based on model B, the rigid panels are supported by black fabric (right).

new structural configurations. Perceptual iteration constantly challenges the user to reconsider which shape to see and what to do with it, structurally or spatially. As a result, the iteration rapidly produces outputs beyond the initial topology. In addition, the schema iterations were able to produce a new set of functions that could be repeated or modified to synthesize a new topology.

Finally, this method allowed us to study the ideation process of the existing structural typology. In the example of converting

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press

weaving into tensegrity, the iteration provided ways to interpret the creative moment when the inventor, Kenneth Snelson, discovered tensegrity from weaving. It also revealed how such an ideation process might link to other crafting techniques, such as folding, thus paving a new path across disciplinary boundaries to enrich the language of structural design.

Acknowledgment. The author would like to thank the reviewers for their valuable feedback, George Stiny for his inputs on the schemas, and Cristina Ramos for analyzing model A to D.

References

Akbarzadeh M, Van Mele T and Block P (2015) On the equilibrium of funicular polyhedral frames and convex polyhedral force diagrams. *Computer-Aided Design* **63**, 118–128.

Azhar S, Nadeem A, Mok JY and Leung BH (2008) Building information modeling (BIM): a new paradigm for visual interactive modeling and simulation for construction projects. In *Proceedings of the First International Conference on Construction in Developing Countries*, pp. 435–446.

Bathe K-J (1996) *Finite Element Procedures*. Englewood Cliffs, N.J: Prentice-Hall.

Bathe K-J and Larsson G (1986) The ADINA system in engineering practice. *Finite Elements in Analysis and Design* **2**(1–2), 41–60.

Chilton JC and Isler H (2000) *Heinz Isler*. London, UK: Thomas Telford.

Collins GR (1963) Antonio Gaudi: structure and form. *Perspecta* **8**, 63–90.

Cremona L (1885) *Elements of Projective Geometry*. Portsmouth, NH: Clarendon Press.

Economou A and Kotsopoulos S (2015) From shape rules to rule schemata and back. In Gero JS and Hanna S (eds). *Design Computing and Cognition'14*. Springer, pp. 383–399.

Engel H (1968) *Structure Systems*. New York: Praeger.

Jackson P (2011) *Folding Techniques for Designers: From Sheet to Form*. London, UK: Laurence King Publishing.

James W (1981) *The Principles of Psychology*. Cambridge, MA: Harvard University Press.

Knight T and Stiny G (2015) Making grammars: from computing with shapes to computing with things. *Design Studies* **41**, 8–28.

Krstic D (2001) Algebras and grammars for shapes and their boundaries. *Environment and Planning B: Planning and Design* **28**(1), 151–162.

Krstic D (2005) Shape decompositions and their algebras. *AIE EDAM* **19**(04), 261–276.

Lee J, Fivet C and Mueller C (2015) Modelling with forces: grammar-based graphic statics for diverse architectural structures. In Thomsen MR, Tamke M, Gengnagel C, Faircloth B and Scheurer F (eds). *Modelling Behaviour*. Design Modelling Symposium 2015, Copenhagen, Denmark, pp. 491–504.

Maxwell JC (1864) XLV. On reciprocal figures and diagrams of forces. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **27**(182), 250–261.

Miller GA (1983) Information theory in psychology. In Machlup F and Mansfield U (eds). *The Study of Information: Interdisciplinary Messages*. New York: John Wiley and Sons, pp. 495–496.

Mitchell WJ (1998) Articulate design of free-form structures. In Smith I (ed.). *Artificial Intelligence in Structural Engineering*. Berlin, Heidelberg: Springer, pp. 223–234.

Mitchell WJ (2001) Vitruvlus redux: formalized design synthesis in architecture. In Antonsson EK and Cagan J (eds). *Formal Engineering Design Synthesis*. New York: Cambridge University Press, pp. 1–19.

Miura K (1985) Method of packaging and deployment of large membranes in space. *Title The Institute of Space and Astronautical Science Report* **618**, 1.

Mueller C and Ochsendorf J (2013) From analysis to design: A new computational strategy for structural creativity. In *Proceedings of the Second International Workshop on Design in Civil and Environmental Engineering*. Mary Kathryn Thompson, pp. 46–56.

Muslimin R (2012) Recursive Embedding of Gestalt Laws and Shape Grammar in the Weaving Design Process. In *Presented at the 30th eCAADe, Prague, Czech Republic*, vol. **1**, pp. 443–449.

Muslimin R (2016) On visual-mechanical synthesis in shell structures. *Nexus Network Journal* **19**(3), 645–664.

Muslimin R (2017) Weaving, folding and the tension between them: a discourse on a structural ideation method. In *Presented at the CAAD Futures 2017, Istanbul, Turkey*, vol. **724**.

Otto F (1973) *Tensile Structures*. Cambridge, Mass: The MIT Press.

Özkar M (2011) Visual schemas: pragmatics of design learning in foundations studios. *Nexus Network Journal* **13**(1), 113–130.

Peña Villamil DM, Llorens Duran JID, Sastre Sastre R, Crespo Artiaga D and Tristancho Martínez J (2011) Application of tensegrity to tensile-textile constructions: form-finding and structural analysis. *Journal of the International Association for Shell and Spatial Structures* **52**(2), 67–81.

Preisinger C and Heimrath M (2014) Karamba – a toolkit for parametric structural design. *Structural Engineering International* **24**(2), 217–221.

Przemieniecki JS (1985) *Theory of Matrix Structural Analysis*. Mineola, NY: Courier Corporation.

Rankine WM (1864) XVII. Principle of the equilibrium of polyhedral frames. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **27**(180), 92–92.

Rippmann M, Lachauer L and Block P (2012) Interactive vault design. *International Journal of Space Structures* **27**(4), 219–230.

Schon DA (1983) *The Reflective Practitioner*. New York: Basic Books.

Shea K and Cagan J (1997) Innovative dome design: applying geodesic patterns with shape annealing. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing* **11**(05), 379–394.

Shea K and Cagan J (1999) The design of novel roof trusses with shape annealing: assessing the ability of a computational method in aiding structural designers with varying design intent. *Design Studies* **20**(1), 3–23.

Shea K, Cagan J and Fenves SJ (1997) A shape annealing approach to optimal truss design with dynamic grouping of members. *Journal of Mechanical Design* **119**(3), 388–394.

Stiny G (1981) A note on the description of designs. *Environment and Planning B: Planning and Design* **8**(3), 257–267. doi: 10.1068/b080257.

Stiny G (1992) Weights. *Environment and Planning B: Planning and Design* **19**(4), 413–430. doi: 10.1068/b190413.

Stiny G (1993) Boolean algebras for shapes and individuals. *Environment and Planning B: Planning and Design* **20**(3), 359–362. doi: 10.1068/b200359.

Stiny G (1996) Useless rules. *Environment and Planning B: Planning and Design* **23**(2), 235–237. doi: 10.1068/b230235.

Stiny G (2006) *Shape: Talking About Seeing and Doing*. The MIT Press. Available at http://dl.acm.org/citation.cfm?id=1795940

Stiny G (2011) What rule(s) should I use? *Nexus Network Journal*, **13**(1), 15–47. doi: 10.1007/s00004-011-0056-6.

Suppapitnarm A, Parks GT, Shea K and Clarkson PJ (2004) Conceptual design of bicycle frames by multiobjective shape annealing. *Engineering Optimization* **36**(2), 165–188.

Weinand Y (2009) Innovative timber constructions. *Journal of the International Association for Shell and Spatial Structures (Journal of the IASS)* **50**(2), 111–120.

Rizal Muslimin teaches at the University of Sydney. He holds a Ph.D. in Architecture: Design and Computation from Massachusetts Institute of Technology, United States. His computational design research centers at the intersection of craft, culture, and architecture. Dr. Muslimin has published scholarly articles in the field of computational craft, shape studies, and ethnocomputation in peer-reviewed journals and conferences. His research on computational craft had received best paper award and design awards.

https://doi.org/10.1017/S0890060417000580 Published online by Cambridge University Press