Blucher Design Proceedings
Dezembro de 2014, Volume 1, Número 7
www.proceedings.blucher.com.br/evento/sigradi2013

Embedding Garifuna Shape Grammars in a Parametric Design Software

Dulce Andino
National Cheng Kung University, Taiwan
dma_22@hotmail.com

Sheng-Fen Chien
National Cheng Kung University, Taiwan
schien@mail.ncku.edu.tw

Abstract

The Garifuna are a group of people that live on the northern coast of Honduras and the coast of Belize. They have a very distinct and vibrant culture. Minority cultures are currently absorbed by mainstreamed cultures and the Garifuna ethnicity is directly influenced by this phenomenon. In this research it is of special concern to encapsulate Garifuna vernacular architecture by means of shape grammars. The research provides a clear documentation of the grammars implemented in Grasshopper, as well as discusses about the issues of embedding shape grammars in the Rhino/Grasshopper environment.

Keywords: Garifuna; shape grammar; Parametric shape grammars; Grasshopper.

Introduction

Minority cultures are currently being absorbed by mainstreamed cultures. Consequently, in some cases this fact leads to cultural identity issues, and observable among these issues is the discontinuation of a particular vernacular architectural style. The Garifuna ethnicity is directly influenced by this phenomenon, henceforth the urgency of decoding the style of Garifuna architecture. The importance of vernacular architecture is a matter of cultural identity (Rudosfsky, 1964). In this research it is of special concern to at least try to encapsulate vernacular construction knowledge and patterns. The shape grammar theory enables the possibility of achieving this in a very accurate documentation. The investigation focuses on digitally capturing the vernacular architectural style of the Garifuna people by means of a parametric shape grammar in Grasshopper software.

The Garifuna

The Garifuna are a group of people that live on the northern coast of Honduras, on the Pacific coast in Guatemala, and the coast of Belize. The Garifuna have a very distinct and vibrant culture. They are descendant from Carib and Arawakan Amerindian tribes, as well as escaped African slaves. In 1635 two Spanish ship loaded with black slaves sunk off the coast of San Vicente. The Africans quickly mixed with the natives to avoid being sent back to their owners as slaves, the Garifuna culture emerged from this mixture. Through a series of disputes and battles between the French and English, the Garifuna were expatriated to the northern coast of Honduras in 1797 (Gonzalez, 1997).

Garifuna settlements are located in the tropical coastal region. The sea works as a delimitation frame in their villages (Salinas, 1991). Their houses are located behind the coconut tree palms, along a main street that goes right through the village. The houses are oriented towards the street, they do not have any private yards, and this provides a solid communication between the dwellers in each house.

The Garifuna Grammars

The Shape Grammar methodology (Stiny, 1980) is utilized to capture the style of Garifuna architecture. The Shape Grammar Theory has aided contemporary researchers to untangle the mysteries of architectural styles or an architect's style, either from the past well as present day (Koning & Eizenberg, 1981). There are different methods to conceive shape grammars. Shape grammars are entirely flexible to behave or conform to each specific requirement (Knight, 1999). One of the methods to create shape grammars is in a two-dimensional approach (Stiny & Mitchell, 1978, 1980). Flemming (1981) established that a two dimensional grammar deals with the blueprint definition, while a three dimensional stage builds up the walls, positions the roof, places openings and so on.

In this research two sets of grammars are developed: the Garifuna settlement grammar, and the Garifuna hut grammar. The Garifuna settlement grammar is developed to generate the settlement patterns observed in Garifuna colony. The Garifuna hut grammar deals with the form and construction of a typical Garifuna hut.

The settlement grammar (Figure 1) has one basic shape and a symbol. There are four rules. Rule 1 is an addition rule to create a neighboring hut. Rule 2 is the addition rule for kitchen. Rule 3 is a reorientation rule that adjusts the label, and removes label. And rule 5 is a termination rule. The initial shape is a labeled shape, representing the very first hut in a new settlement.

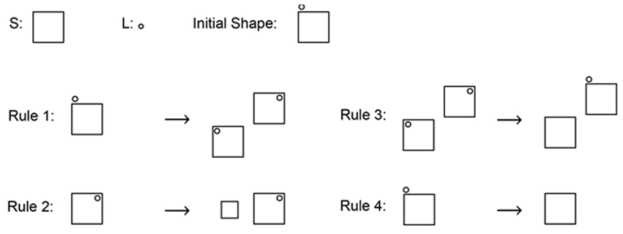

Figure 1: Garifuna Settlement Grammar.

Andino, Dulce; Chien, Sheng-Fen; "Embedding Garifuna Shape Grammars in a Parametric Design Software", p. 202-206 . **In: Proceedings of the XVII Conference of the Iberoamerican Society of Digital Graphics: Knowledge-based Design [=Blucher Design Proceedings, v.1, n.7]**. São Paulo: Blucher, 2014. ISSN 2318-6968, DOI 10.5151/despro-sigradi2013-0037

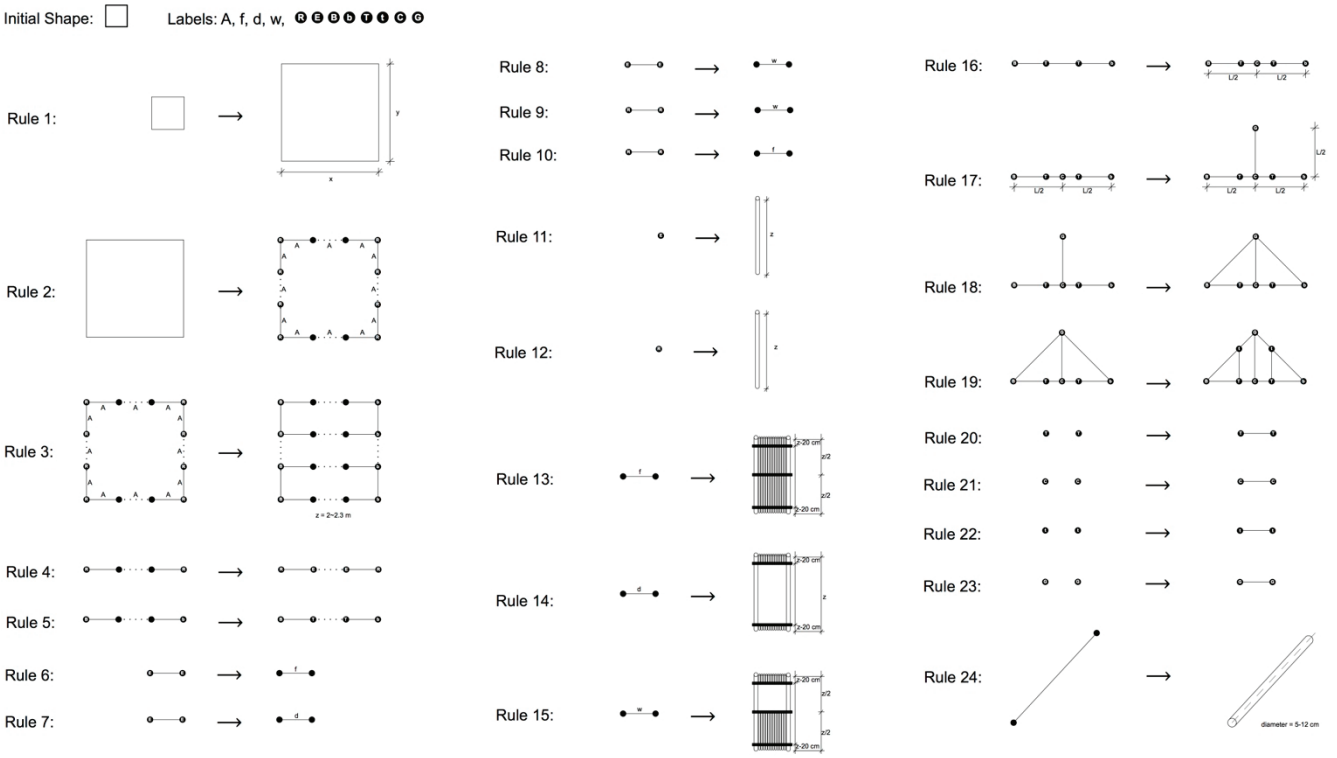

Figure 2: Garifuna Hut Grammar.

The hut grammar (Figure 2) is to produce the actual construction of a traditional Garifuna hut. The hut grammar assumes that a settlement has already existed. Therefore, the initial shape should be a rectangle representing a footprint for a hut. The grammar is divided into three parts: frame generation, skin infill, and the roof frame generation. The frame generation grammar takes the footprint of a hut, adds columns, and allocates walls and openings, and establishes the grid that the roof base will be built upon. The skin infill grammar deals with the infill rules within a frame, and finally the roof frame generation grammar refers to the gable's roof positioning and construction. To consider the materials and sizes of the actual construction of a hut, the hut grammar is implemented as a parametric grammar.

Garifuna Hut Grammar in Grasshopper

Implementation strategy

The shape grammar is implemented using Rhinoceros' (Rhino) plug-in, Grasshopper. By merging different components into clusters, the left hand side (LHS) and the right hand side (RHS) of the rules in the grammar enable the rules to be executed precisely. For LHS pattern matching, a predefined geometry component is utilized to select desired geometry. For example, to match on point elements in a cluster of shapes, the cluster of shapes is first obtained from a "Geometry" component and then passed to a "Point" component; the Point component outputs all point elements (with warnings if there are elements of different type). This is a kind of an "automatic pattern matching" mechanism provided by Grasshopper (Figure 3). To take advantages of this feature, all shape rules are implemented in separate Grasshopper documents.

Furthermore, we have limited ourselves from using scripting or customized components in Grasshopper so that we may explore the extent of its possibilities. The application of shape rules is performed manually where RHS outputs of each rule are baked into Rhino to be ready for use in the next rule application.

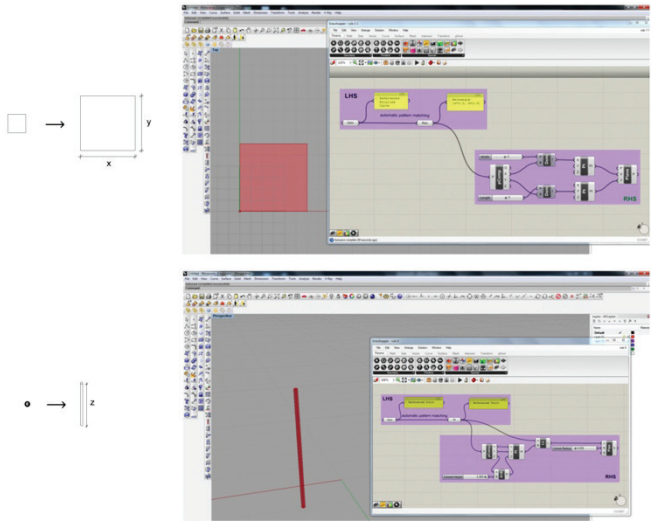

Figure 3: Two illustrations of the implementation strategy.

Rules

Rule 1 states the parameters of the hut's width and height. To achieve this in Grasshopper, the LHS is established as a rectangle. This geometry is then connected to another group of components that together allow the RHS transformation of the shape. The RHS utilizes an exploded B-rep component that works with an edge at a time to scale the length and the width of the shape (hut area). Once the desired dimension is obtained, and then one must bake the geometry into Rhino (Figure 4).

Rule 2 parameterizes the labels for the column positioning (Figure 5). The LHS is a surface geometry acquired through Rule 1. The RHS consists of an exploded B-rep component to work with edges of the surface independently. Each edge is divided into segments of equal length. At each division point, a tag component is implemented as the label, which is required to work as a LHS for the upcoming Rule 3. The points and the tags have to be baked in Rhino.

Rule 3 declares that the labeled points for column positioning are to be projected in the z-axis and state a new set of labels for the gable roof grid. However, when the attempt of making the tag the LHS of the rule fails, the first limitation of the software is encountered. At this point, we realized that the tag although visible in Rhino cannot be identified as geometry in Grasshopper. Therefore, the LHS is defined by a point instead of a labeled point (Figure 6). The user has to pick the correct points manually for LHS.

Rules 4 and 5 set the beam positioning parameter and beam labels. The LHS of the rule is defined by geometries of points; these points have visible labeled tags in Rhino. However, since labeled points cannot be pattern-matched, these two rules are identical in Grasshopper. Once the user defines the points, the data is transferred to the RHS of the rule. The RHS draws the beam and positions another set of labeled tags, both elements are to be baked in Rhino (Figure 7).

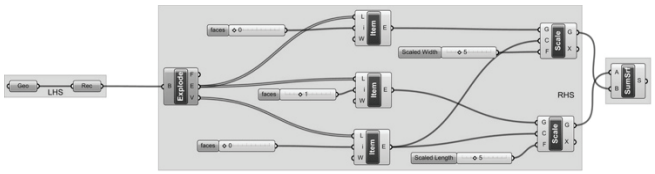

Figure 4: Rule 1.

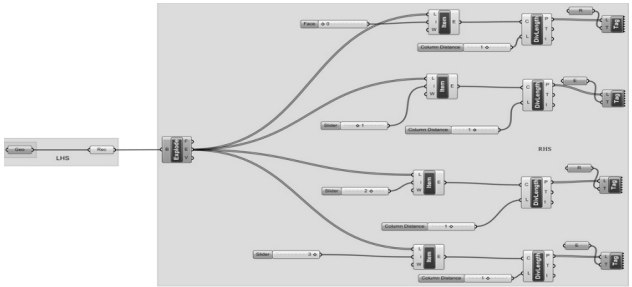

Figure 5: Rule 2.

Rules 6 to 10 set labels for doors, windows, and wall panels. However, since the resulting labeled points cannot be pattern-matched, these rules are omitted.

Rules 11 and 12 parameterize the labeled points into columns. Without the distinction of labels, they are identical in Grasshopper. The LHS takes a point, which is transformed into a column in the RHS of the rule. The RHS of the rule consists of a vector component (pComp), to define the column length in the z-axis. Then it draws the column (Figure 8).

Rules 13 to 15 parameterize the profiles of doors, windows, and walls in a hut. The LHS sets two points (columns) as geometries that the user has to pick if they are one next to the other. The RHS then determines if it will become a door, a window, or a wall panel (Figure 9). Each profile is defined by the loft component.

Rule 16 sets a label for the central end stud positioning (Figure 10 top). The LHS is defined by two labeled points, the end points of the beam given after the execution of Rule 5. The RHS of this rule, divides the beam in two equal parts, draws a point and sets a labeled tag "C" to be baked in Rhino. Rule 17 assigns the parameters for the central end stud and labels at the top for further rules to be properly executed (Figure 10 bottom). The LHS consists of a labeled point geometry provided by baking Rule 16. The RHS takes the point and moves it drawing the central end stud along with it in the z-axis with a span equal to half the length of the beam, to create a 45-degree angle in the following rule.

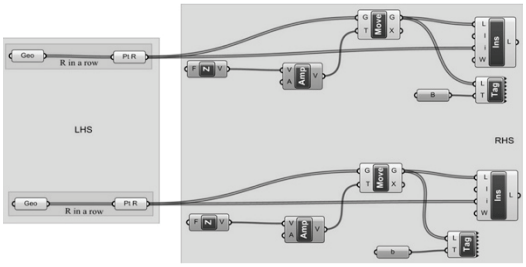

Figure 6: Rule 3.

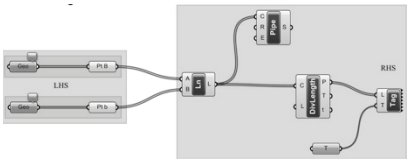

Figure 7: Rule 4 (Rule 5).

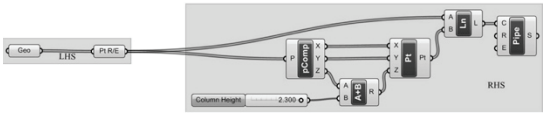

Figure 8: Rule 11 (Rule 12).

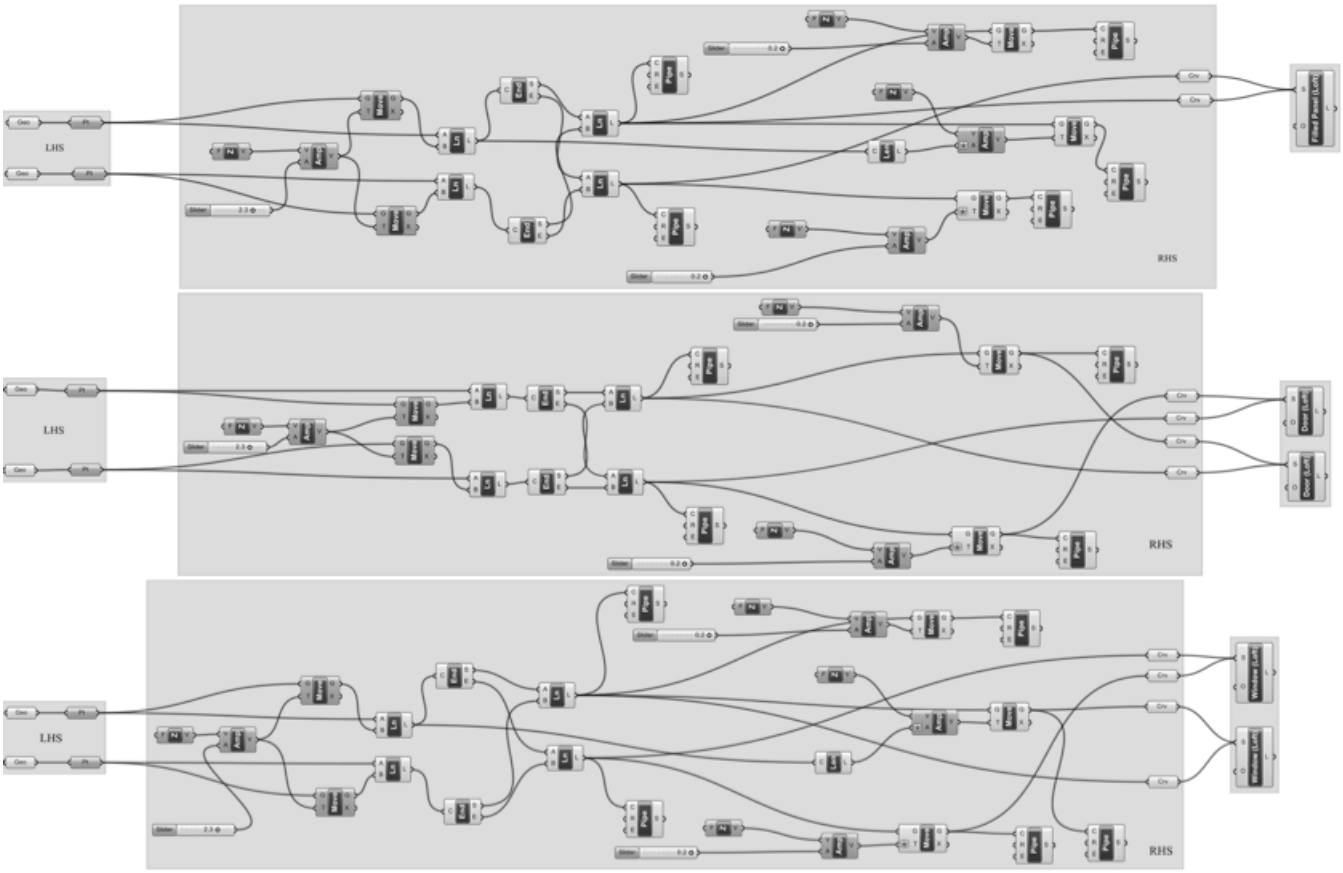

Figure 9: Rules 13 (top), 14 (middle) & 15 (bottom).

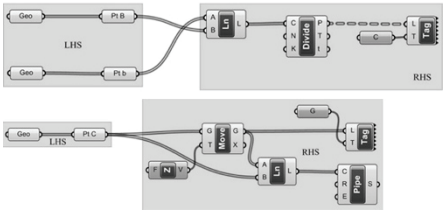

Figure 10: Rules 16 (top) & 17 (bottom).

Rule 18 draws the 45-degree angle rafters (Figure 11 top). The LHS is represented by point geometries. The points used in this rule are; the two labeled end points of the beam (Rule 5), and the labeled point at the top of the central end stud (Rule 16). The RHS basically draws the rafters from three points, similarly as drawing a triangle from three points.

Rule 19 defines the remainder end studs along the beam (Figure 11 bottom). The LHS is determined by taking two geometries: a point and a line. The labeled point is obtained from Rule 3, and the line is a 45-degree angle rafter. The RHS takes these geometries and finds their intersection point, and then labels the intersecting point with a tag. It also draws the end stud between the rafter and the labeled point on the beam.

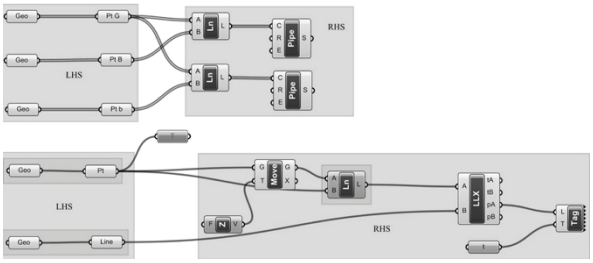

Figure 11: Rules 18 (top) & 19 (bottom).

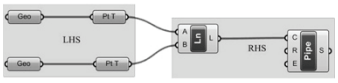

Figure 12: Rule 20 (Rule 21, 22, 23).

Rules 20 to 23 draw linear elements between labeled points. Because all of them have the same logic in Grasshopper; they are identical (Figure 12). The LHS is defined by the geometry of two identical labeled points. Then the RHS of these rules draws a joist in Rule 20, another joist in Rule 21, a truss brace in Rule 22, and the roof's ridge in Rule 23.

A Garifuna hut

The Grasshopper rules are verified through reproducing traditional Garifuna huts of various types. Figure 13 illustrates the derivation process of rule applications to create an original hut (Salinas, 1991).

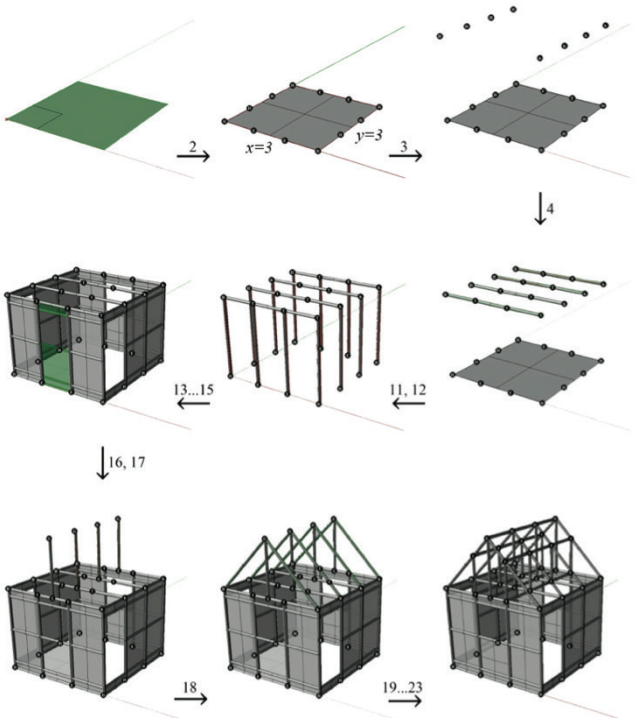

Figure 13: Design derivations of an original hut.

Discussions and Conclusion

Our experience of implementing the rules of the Garifuna grammars has allowed us to gain insights of the fits and misfits of embedding shape grammars in Rhino/Grasshopper. A shape grammar system should allow designers to (1) create and modify the shape grammar, (2) compile the grammar, and (3) explore the language of designs defined by the grammar (Tapia, 1999). Below, we discuss issues of implementing shape grammar in Rhino/Grasshopper according to the three phases.

Key issue in the creating and modifying of the shape grammar is representation of shapes and rules. The representation of rules in Grasshopper is straightforward: a cluster of components performing LHS and another cluster of components performing RHS (e.g., Figure 12). The representation of shapes to allow for subshape detection and emergent shapes, however, cannot be easily achieved in Grasshopper without additional scripting. The scripting or textual programming usually involves breaking down geometric elements into lines and vertices (points) to recompose into new shapes (e.g., Celani & Vaz, 2012; Grasl & Economou, 2011).

For compiling the grammar, the key issues are pattern matching for LHS and shape substitution for RHS. Grasshopper provides convenient pattern matching of typified geometry. However, for matching onto subshapes or emergent shapes, scripting is needed (as described in the preceding paragraph). For labeled shapes, the tag component is convenient for label creation but the labels are not geometric elements therefore cannot be used for LHS pattern matching. So performing LHS pattern matching in Grasshopper is limited to predefined shapes. In terms of RHS shape substitution, removing or replacing geometries in Rhino may cause the Grasshopper system to fall into an execution loop. Therefore, shape substitution is not likely to achieve in Rhino/Grasshopper environment.

To explore the language of designs defined by the grammar, the convenient RHS pattern matching (described in the preceding paragraph) can potentially perform multiple activations of rules. Essentially having the effect of parallel rule firing in a rule-based system. To control the design exploration process, the designer may activate/deactivate (load/unload) certain rules in the grammar or adjust the shape database (input shapes) for each rule. These two design exploration schemes, rule activation management and input shapes management, are those used in our research.

In conclusion, we found several limitations when embedding shape grammars in Rhino/Grasshopper. However, designers may still utilize the system to implement shape rules and explore designs. Nevertheless, to gain the full power of shape grammar systems, designers at least need the ability of scripting (beyond visual programming) to achieve.

References

Celani, G., & Vaz, C. (2012). CAD scripting and visual programming languages for implementing computational design concepts: a comparison from a pedagogical point of view. *International Journal of Architectural Computing*, 10(1), 121-138.

Flemming, U. (1981). The Secret of the Casa Giuliani Frigerio. Environment and Planning B, 8, 87-96.

Gonzalez, N. L. S. (1997). The Garifuna of Central America. In S. M. Wilson (Ed.), The Indigenous People of the Caribbean. (pp. 197-205). Gainsville: University Press of Florida.

Grasl, T., & Economou, A. (2011). GRAPE: using graph grammars to implement shape grammars. Proceedings of SimAUD 2011, Boston.

Knight, T. (1999). Shape grammars: six types. Environment and Planning B, 26, 15-32.

Koning, H., & Eizenberg, J. (1981). The language of the prairie: Frank Lloyd Wright's prairie houses. Environment and Planning B, 8, 295-323.

Rudosfsky, B. (1964). Architecture without architects. New York: Doubleday.

Salinas, I. M. (1991). Arquitectura de los grupos etnicos de Honduras (Vol. 1). Tegucigalpa, Honduras: Editorial Guaymuras.

Stiny, G. (1980). Introduction to shape and shape grammars. Environment and Planning B, 7(3), 343-351.

Stiny, G., & Mitchell, W. J. (1978). The Palladian grammar. Environment and Planning B, 5(1), 5-18.

Stiny, G., & Mitchell, W. J. (1980). The grammar of paradise: on the generation of Mughul gardens. Environment and Planning B, 7(2), 209-226.

Tapia, M. (1999). A visual implementation of a shape grammar system. Environment and Planning B, 26(1), 59-73.