Carnegie Mellon University
Research Showcase @ CMU

School of Architecture | College of Fine Arts

4-2008

# Determining the interior layout of buildings describable by shape grammars

Kui Yue
*Carnegie Mellon University*

Casey Hickerson
*Carnegie Mellon University*

Ramesh Krishnamurti
*Carnegie Mellon University*, ramesh@cmu.edu

Follow this and additional works at: http://repository.cmu.edu/architecture

Part of the Architecture Commons

Recommended Citation
CAADRIA 2008 (eds. W Nakapan, E Mahaek, K Teeraparbwong, and P Nilkaew), 117- 124.

This Conference Proceeding is brought to you for free and open access by the College of Fine Arts at Research Showcase @ CMU. It has been accepted for inclusion in School of Architecture by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.

# DETERMINING THE INTERIOR LAYOUT OF BUILDINGS DESCRIBABLE BY SHAPE GRAMMARS

KUI YUE, CASEY HICKERSON
*Carnegie Mellon University, Pittsburgh, Pennsylvania, USA*
*kyue@andrew.cmu.edu, caseyh@cmu.edu*

AND

RAMESH KRISHNAMURTI
*ramesh@cmu.edu*

**Abstract.** This paper describes our current research results on the general algorithm and data structures supporting the determination of the interior layout for buildings comprising rectangular spaces and describable by shape grammars.

**Keywords.** Shape grammars; interior layout; constraints satisfaction; graph-like structure.

## 1. Introduction

We were posed the following problem, namely, of determining the interior layout of a building from its exterior and other (type, environmental, site, cultural etc.) features. In varying incarnations, the problem has practical use; for instance, assessing the environmental impact of demolition and salvage of building stock requires one to estimate the amount of renewable materials (Lund and Yost, 1997). Automating the process of interior layout determination might greatly assist this process.[1] Although it is not especially hard for a human to roughly divine building layouts from familiar features, programming a machine to do so is much more difficult. Employing knowledge about building styles might make the problem tractable. This paper presents a first attempt solution, indeed, a naïve solution in the artificial intelligence sense of the word, using exterior features as input.

We may safely assume that many buildings follow a pattern book; that is, they vary according to well-defined configurational patterns as well as certain established sets of regulations and dimensions. In this respect, a shape grammar (Stiny, 2006) is a remarkable mechanism for encapsulating spatial and topological aspects of building styles and for generating such designs — examples include Queen Anne houses (Flemming, 1987), Frank Lloyd Wright's Prairie houses (Koning and Eizenberg, 1981), Alvaro Siza's

[1] We are grateful to Brad Guy, for bringing this to our attention.

houses at Malagueria (Duarte, 2005; Duarte, 2005) and the bungalows of Buffalo (Downing and Flemming, 1981).

The challenge is to use a base of general knowledge about buildings in a given style and specific knowledge of the exterior features of building to determine its interior features. Formally, we seek an algorithm to determine the interior layout of a building given three pieces of information: i) the footprint of each story; ii) a reasonably complete set of exterior features, e.g., windows, chimneys and surrounding buildings; and iii) a shape grammar that describes the building. We do so through two test cases: Queen Anne houses in Pittsburgh as a control environment with a known shape grammar, and rowhouses in Baltimore (Hayward and Belfoure, 2005), as a test environment, for which no known shape grammar exists.

This leads to an approach based on the fact that, when applied exhaustively, a shape grammar can generate, as a tree, the entire layout space of a style. See Figure 1. The approach begins with an initial layout estimation employing building feature constraints on the feature input. From this estimation, spatial and topological constraints are further extracted. The constraints are then used to prune the layout tree. The layouts that remain correspond to the desired outcomes.

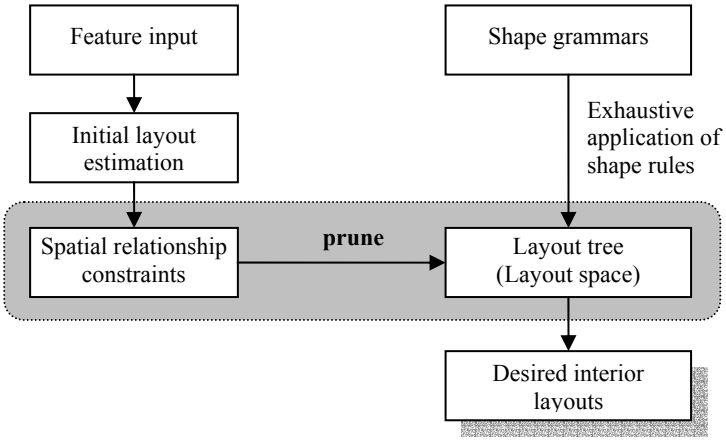



*Figure 1.* General approach to layout prediction

In the sequel, we detail the two test cases illustrating: the initial layout estimation using feature constraints; a graph-like data structure to support a general shape grammars interpreter for buildings with rectangular spaces; and the desired results from applying the above general approach.

## 2. Constraints resolution for initial layout estimation

Building features constrain one another. For example, a window normally belongs to a unique room (space), no wall falls within a window; the ratio of a room is usually ≥ ½, to be appropriate for use; and the minimum width of a space ≥ 2', to be functional. Other constraints require specific knowledge. For example, in Queen Anne houses, a front door is always on the front side of the building and opens into a hall way; however, this is not the case for Baltimore rowhouses.

For our purpose, the height of each story is determined by window and door geometries. Moreover, once the height of a story is estimated, the dimension of various forms of staircases can be estimated through common dimension of treads and risers (24" ≤ tread + 2 * riser ≤ 25", in USA), and the width of staircase is another constant (usually 3') in a given context.

In everyday circumstances, we take advantage of such building feature constraints. For instance, one might guess that a relatively small window on an upper floor of a house belongs to a toilet or to the landing on a staircase.

## 2.1 CONSTRAINTS SATISFACTION AND QUEEN ANNE HOUSES

For Queen Anne houses, we explore layout determination as a form of constraint satisfaction (Russell and Norvig, 2002). A constraint satisfaction problem (CSP) has three components: i) a set of variables $X = \{x_1,\ldots, x_n\}$, ii) a set of possible domain values, $D_i$, for each $x_i$, and iii) a set of constraints to restrict the values variables can simultaneously take. Different acceleration techniques are used to eliminate impossible values, thereby solving the CSP.

The CSP algorithm for layout derivation starts by generating rooms with conservative dimensions to accord with the given features. The algorithm then manipulates rooms as variables according to constraints established by common building properties. Two kinds of manipulations are considered: expanding room dimensions, and merging two rooms into one.

Figure 2 shows the results of applying the algorithm to the first floor of a Queen Anne house. Input features are locations in plan of the windows, doors, and chimneys, together with possible symmetry axes inferred from the facades. Step 1 extends the axes of exterior walls inward (assuming a wall thickness of 1') to form wall hotspots, to enforce a tendency of interior rooms to be aligned with one another. Step 2 uses the fact that larger public rooms on the first floor have fireplaces, which correspond to the chimneys. By projection, if the chimney falls within the interior of the footprint, then there are possible two rooms that share the chimney, with a fireplace each. If the chimney is on an exterior wall, only one room can use the chimney. Such rooms are assigned with an initial dimension of 8' × 8'. Step 3 adjusts rooms that are too close (1' threshold) to align with the nearest axis. Step 4 uses the fact that rooms do not intersect with other rooms and doors. Rooms are extended to include such features to resolve any conflict. Step 5 uses the fact that the minimum distance between two walls has to be large enough to be a useful space (usually > 3'). In step 6, rooms generated from the chimneys are stable. Step 7 specifies rooms (5' × 5') to unassigned windows and doors. Note that a room may be left-, center-, or right aligned with a window or door. Two largely overlaying rooms are merged as one. Also, the narrow space remaining between *RM1* and *RM5* is assigned to *RM5* according to the symmetry axis, *SYM-4*. Step 8 shows the final result; though incomplete, but it is close to the actual condition. At this stage, there are ambiguities that cannot be resolved without prior knowledge, which is left for the shape rules to handle.

Theoretically, the CSP algorithm should work for a variety of building types. However, it does not perform well on Baltimore rowhouses even though there is relatively little morphological variation when compared to the Queen Anne houses. Rather than add more constraints to specifically cater for rowhouses, we consider a simpler alternative approach.

*Figure 2.* Layout derivation of Queen Anne houses by CSP

## 2.2 SPACE SUBDIVISION TREE AND BALTIMORE ROWHOUSES

Procedurally, the first floor layout of the rowhouse can be determined by a decision tree (Figure 3) — essentially, as a subdivisive process.

The first floor is typically decomposed into two or three rectangular blocks: a block containing a parlor towards the front, a block containing a kitchen towards the rear, and an optional, smaller central block that connects the two. In a three-block rowhouse, the central block contains a pantry or a stair while the front and rear blocks are divided into one or two rooms. The kitchen is always the rear-most space while the parlor is the front-most space. The dining room usually appears in the front block behind the parlor

or in the rear bock forward of the kitchen. The two cases can be distinguished by comparing the depths of the front ($h_2$) and rear ($h_1$) blocks.

Two-block rowhouses are more involved. Depending on the depth ($h$) of the front block, it can contain a single room, or be divided into a parlor and dining room possibly separated by a staircase. If the front block comprises two rooms, the staircase can occupy an enclosed space or it can be open to one or both rooms. If the front block comprises a single room, the staircase may have two possible arrangements, either in the front block, or in the rear block. These configurations cannot be determined by the decision tree, which needs further refinement using shape rules.

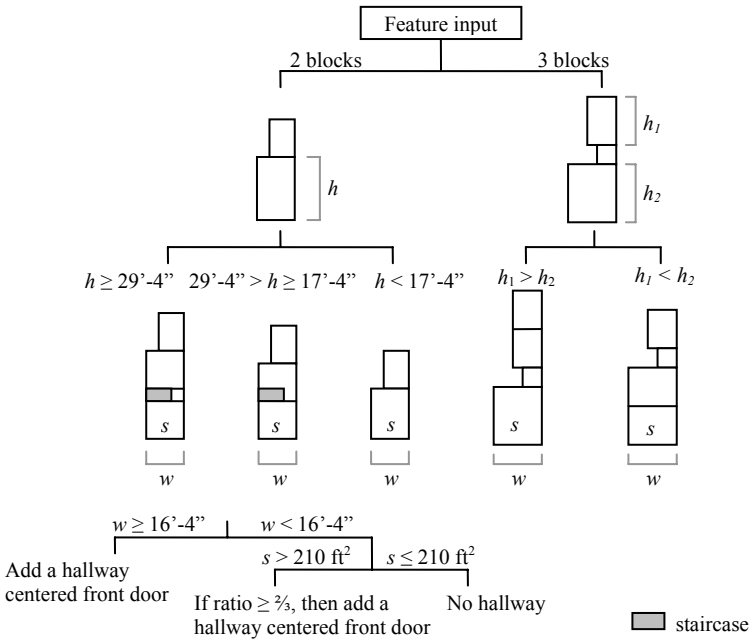



*Figure 3.* Space subdivision tree of a rowhouse.

Regardless of whether the layout has two or three blocks, the front door enters into the front-most room or a dedicated hallway. This is determined from the width ($w$) and area ($s$) of the front-most room.

## 3. Improving the initial layout estimation using shape rules

To obtain the desired layouts, the initial estimation has to been refined by shape rules. This, essentially, requires a shape grammar interpreter. A single general interpreter for different building types is essential, as it is impractical to implement individual interpreters for each grammar. Implementing a general interpreter is still a valid topic of research (Chau et al., 2004). The challenge lies in adequately handling emergence, parametric rules, and curved forms for subshape detection.

However, this does not pose an obstacle for most common building types. Consider, for example, the Queen Anne house and the Prairie house (Koning and Eizenberg, 1981) grammars. These are parametric grammars with no emergent shapes. Markers drive shape rule application, and configurations are rectangular or can be approximated as such. Moreover, parameterization

is often limited to the height or width of a room, or to the ratio of a room split. Shape rules typically tend to relate to adding a room, spitting a room, or refinements such as adding windows, doors, etc.

The interpreter needs to translate shape rules into pieces of 'code.' Traditionally, the design of a shape grammar focuses simply on succinctly describing the underlying building type, with little consideration on how the grammar can be computationally implemented. For example, a description of the form "If the back or sides are wide enough, rule 2 can be used…" is inherently counter-computable. A general interpreter requires the rules to be quantitatively described, especially, the conditions under which they apply.

### 3.1 DATA STRUCTURE FOR LAYOUTS WITH RECTANGULAR SPACES

A rectangular space is specified by a set of walls in such a way that the space is considered rectangular by the human vision system. In Figure 4a, among other variations, a space can be specified by four walls jointed to one another, four disjoint walls, three walls, or framed by four corners.

A graph-like data structure is used to record such variations. There is a boundary node for each corner of the rectangular space, as well as a node for each end of a wall. Nodes are connected by either a wall edge (solid line) or an empty edge (dotted line). A central node represents the room corresponding to the space, and connects to the four corners by diagonal edges (dashed lines). It is needed for manipulating boundary nodes, such as dividing a wall through node insertion, deleting a wall by changing its edge type to empty, and so on. Additionally, information about the room is recorded in the room node, e.g., a staircase within the space. Unlike traditional graph data structures, the angle at each corner is set to be a right angle. A node has at most eight neighbors. A set of such graph units can be combined to represent layouts comprising rectangular rooms (Figure 4b).

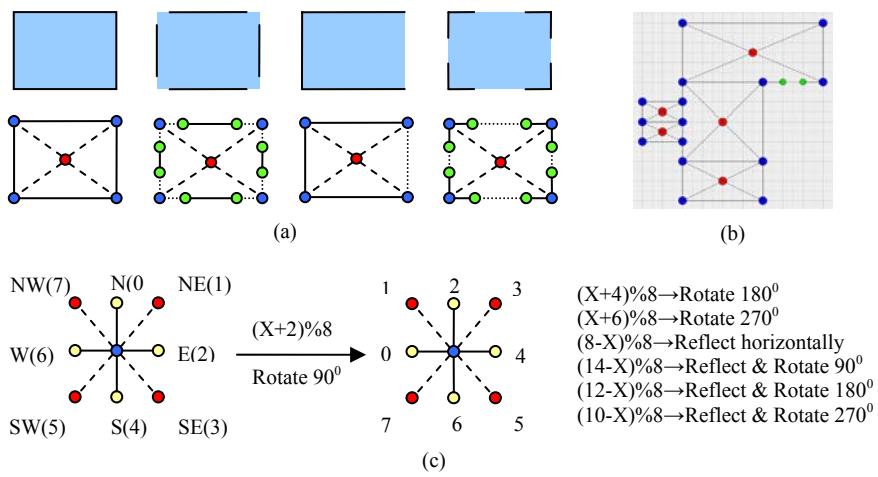



*Figure 4.* Graph-like data structure for rectangular spaces

It is necessary for the data structure to support geometric transformations. For layout of rectangular spaces, applicable transformations are translation, rotation, reflection, glide reflection, and scale (uniform and non-uniform). Moreover, rotations are multiples of 90˚ and reflections are about the either horizontal or vertical.

The transformations are easily implemented on the data structure through index manipulation. Each neighbor of a node is assigned an index from 0 to 7; indices are then transformed simply by modulo arithmetic (Figure 4c). For example, index+2 (modulo 8), rotates ccw neighbor vertices through 90˚. Other rotations and reflections are likewise achieved. By viewing the original neighbor relationship for each node with the transformed indices, we obtain the same transformation of the whole graph. Moreover, we need manipulate only the interior layout instead of the left side of a shape rule. This gives the same result, although simpler. Thus, we only need to consider how to apply shape rules in the case of translation, which is automatically applicable to the configuration under different possible transformations.

Application of shape rules is achieved by manipulating the data structure. Examples of common manipulations include finding a room with a given name, finding the north neighbor of a given room, finding the shared wall of two given rooms, etc.

## 4. Application of the general approach on the two test cases

The general approach has been successfully applied to the Baltimore rowhouses. As the initial layout estimation is the same as the first several rules of the rowhouse grammar (Figure 5), actual application of shape rules on the initial estimation becomes a direct refinement without significant tree pruning. Figure 6a shows the results from the implementation.

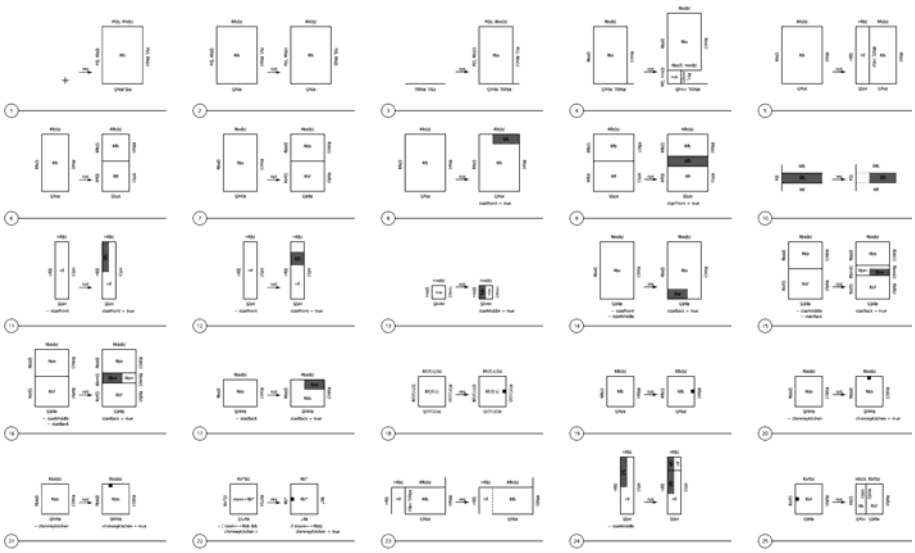
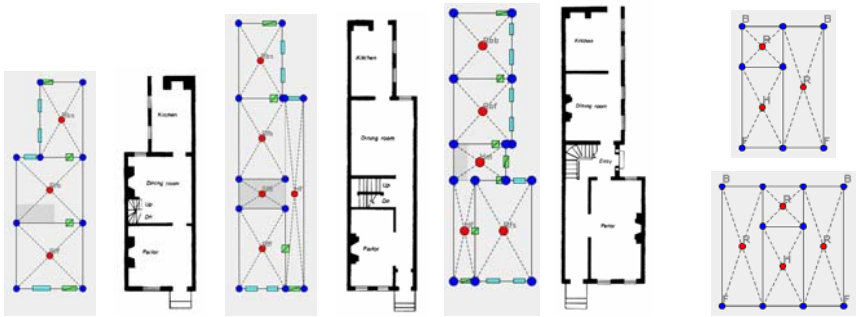


*Figure 5.* Some shape rules from the Baltimore rowhouse grammar (total 52 rules)

For the Queen Anne houses, we are presently reworking the rules given in Flemming (1987). By comparison to Baltimore rowhouse shape rules, Queen Anne shape rules are more complex as the manner in which the rules apply differ from the initial layout estimation approach. Still, it is necessary for the initial layout estimation prior to pruning the layout tree. We are still working on the pruning mechanism. Nonetheless, our current work shows the generality of the graph-like data structure. See Figure 6b.



(a) The Baltimore rowhouses (b) Queen Anne houses

*Figure 6.* Implementation results

## 5. Conclusion

This paper presents an approach for determining the interior layouts of buildings as well as a data structure for its implementation. More importantly, this work forms the basis of a framework for computation-friendly shape grammars, which will facilitate the implementation of a general shape grammar interpreter.

## Acknowledgements

The work reported in this paper is funded in part by CERL. Any opinions, findings, conclusions or recommendations presented in this paper are those of the authors and do not necessarily reflect the views of CERL.

## References

Chau, H. H., Chen, X., McKay, A. and Pennington, A.:2004, Evaluation of a 3D shape grammar implementation. *Design Computing & Cognition '04*, Boston, pp. 357-376.

Downing, F. and Flemming, U.:1981, The bungalows of Buffalo, *Environment and Planning B: Planning and Design*, **8**, pp. 269-293.

Duarte, J. P.:2005, A discursive grammar for customizing mass housing: the case of Siza's houses at Malagueira, *Automation in Construction* **14**(2), pp. 265-275.

Duarte, J. P.:2005, Towards the Mass Customization of Housing: the grammar of Siza's houses at Malagueira, *Environment and Planning B: Planning and Design*, **32**(3), pp. 347-380.

Flemming, U.:1987, More than the sum of parts: the grammar of Queen Anne houses, *Environment and Planning B: Planning and Design*, **14**, pp. 323-350.

Hayward, M. E. and Belfoure, C.:2005, *The Baltimore Rowhouse*, Princeton Architectural Press, New York.

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

Lund, E. and Yost, P.:1997, Deconstruction - Building Disassembly and Material Salvage: The Riverdale Case Study. NAHB Research Center, Inc., Upper Marlboro, MD.

Russell, S. and Norvig, P.:2002, *Artificial Intelligence: A Modern Approach*, Prentice Hall.

Stiny, G.:2006, *Shape: Talking about seeing and doing*, MIT Press, Cambridge.