Automation in Construction 94 (2018) 405–416

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Customization and generation of floor plans based on graph transformations

Xiao-Yu Wang[a,*], Yin Yang[b], Kang Zhang[a]

[a] The University of Texas at Dallas, 800 W Campbell Rd., Richardson, TX, USA
[b] University of New Mexico, Albuquerque, NM, USA

ARTICLE INFO

Keywords:
Floor planning
Modeling
Rectangular dual
CAD

ABSTRACT

This paper introduces an approach for automatic generation of rectangular floor plans based on existing legacy floor plans with the capability of further improvement and customization. Our approach first derives a dual graph from the given input file specifying a floor plan. It then either automatically reproduces varied floor plans retaining the connectivity of the original plan or performs transformation rules to manipulate spatial relations among rooms, and to generate modified floor plans corresponding to specific requirements. Our approach introduces constraints, such as the maximum width-height ratio, to support the flexibility for various design requirements. A graphical user interface is provided for users to perform the automatic generation process. An experiment has been conducted to validate the feasibility of our approach and time taken in generating floor plans. It shows that our method is able to generate highly-customized floor plans in reasonable time.

1. Introduction

In the last few decades, researchers have developed various approaches to generate interior building layouts in styles similar to existing famous or historic design paradigms. Stiny and Mitchell [1] proposed a shape grammar method for the room arrangement of the Palladian style. They later proposed another shape grammar to generate plans following the style of Mughul gardens [2]. Koning and Eizenberge [3] developed a set of rules to reproduce the Frank Lloyd Wright's prairie style houses. Recently, we have witnessed the era of information explosion. Numerous floor plan resources are now digitized and achieved on the computers, which allows graphics and vision techniques to be applied to extract the spatial relation of rooms in the floor plan. For example, Fan et al. [4] presented a structure completion method from facade layout images, Liu and Zlatanova [5] extracted adjacency relations from CityGML, and Lin et al. [6] obtained obstacle data from the Industry Foundation Classes (IFC) data for pathfinding. Moreover, there are also works that aim to generate floor plans automatically or semi-automatically. Bhasker and Sahni [7,8] used a linear algorithm to check if there are rectangular duals and, if so, generates rectangular duals for any *n*-vertex planar triangulated graphs. Other researchers focus on the placement of components, e.g. Kahng [9] provided a way that places components into a space. However, his approach generates new spaces, which could change the connectivity of the original graph.

Clearly, it is highly desired to be able to reproduce well-known legacy floor plans, adjust or modify them to suit modern lifestyles while retaining the room adjacency, i.e., one of the most representative features for a given building style. To this end, we propose a Graph Approach to Design Generation or GADG, which generates floor plans based on the room adjacency from an input (as shown in Fig. 1). An adjacency between two rooms means they are connected by sharing a door or a wall. Input files could be floor plans or semantic rich languages specifying necessary requirements. With the extracted information, users are able to modify the corresponding dual graphs by applying transformation rules. To enhance usability and operability, we introduce two parametric transformation rules, the addition rule and subtraction rule, for modifying rooms. The rules are applied directly to the graph to add/remove vertices instead of rooms. Then, GADG performs an algorithm to remove isolated vertices from the graph. Since our approach only generates rectangular rooms from properly triangulated planar graphs, a validation algorithm is performed to check the existence of rectangular floor plans for the graph. Finally, with user-specified constraints, such as setting the maximum aspect ratio for each generated room and moving internal rooms to the boundary, GADG reproduces a set of new plans automatically by applying a rectangular dual finding algorithm so that the original connectivity information is preserved. With these generated plans, designers can come up with new design ideas or directly take the generated floor plans as prototypes.

GADG is featured by providing a fully automated process to generate controllable and tractable floor plans with identical connectivity. It is able to incorporate various constraints during the generation

* Corresponding author.
E-mail addresses: xwang@utdallas.edu (X.-Y. Wang), yangy@unm.edu (Y. Yang), kzhang@utdallas.edu (K. Zhang).

https://doi.org/10.1016/j.autcon.2018.07.017
Received 14 December 2017; Received in revised form 14 July 2018; Accepted 18 July 2018

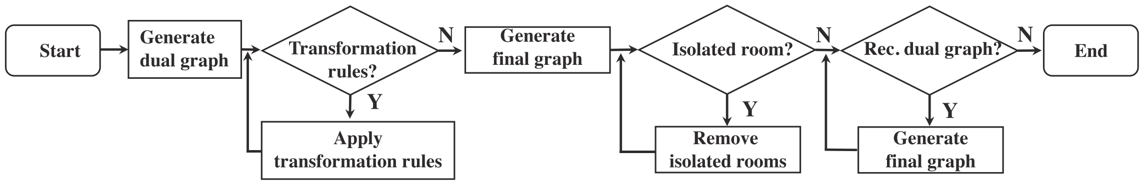

Fig. 1. The overall flow chart of the proposed floor plan generation pipeline.

Table 1
Features of different floor plan generation approaches.

Features	[3]	[10]	[11]	[12]	[13]	[7]	[14]	[15]	[16]	[17]	[4]	Our approach
Giving a floor plan to guide the generation	–	–	–	–	–	–	–	–	–	X	–	X
Generating floor plans conforming to a style	X	(–)	(–)	(–)	(–)	(–)	X	(–)	(–)	X	(–)	X
Using constraints to control generated plans	–	–	–	–	–	–	X	–	–	X	X	X
Supporting a fully automatic generation process	–	(X)	–	–	X	X	X	X	X	–	–	X
Using graphs to consider room adjacencies	–	X	X	X	X	X	–	X	X	(–)	–	X
2D/3D	3D	2D	2D	2D	2D	2D	3D	3D	2D	2D/3D	3D	2D

X: considered. (X): implicitly considered. –: not considered. (–): not mentioned.

process to generate floor plans under a wide range of design requirements. Table 1 shows a comparison of features between existing floor plan generation approaches and our approach.

2. Related work

Many researchers have proposed approaches for automatic generation of floor plans. Placing rooms into given planes is known as the space allocation problem [17]. Some researchers attempted to solve this problem by exhaustively generating all possible results, e.g., Galle et al. [10] proposed an algorithm that generates every possible building plan for any given number of rooms. The complexity of this algorithm increases exponentially with the increasing number of rooms. Others focus on generating floor plans with heuristic placements. Shekhawat [18] developed an algorithm that utilizes the given spaces inside a rectangle to find satisfactory results.

Some approaches generate floor plans based on a given room adjacency graph. Roth et al. [11] proposed an approach to turn a graph to a rectangular plan with the adjacency preserved. Their approach, however, is unable to automatically generate designs. Kozminski [13] developed an $O(n^2)$ algorithm to generate rectangular duals for given triangulated graphs. Bhasker and Sahni [7,8], then, improved this approach by providing a linear time algorithm for verifying and generating rectangular floor plans for triangulated graphs. The reproduced floor plans retain the same adjacency relations as the input graphs. This approach generates floor plans according to n-vertex planar triangulated graphs, which means that the results generated by this algorithm retain the same connectivity. Since input graphs are user-specified, without considering existing floor plans and different requirements, it is difficult to apply this approach to real-world designs.

An algorithm of Martin [15] generates floor plans by making and manipulating graphs in three main steps. The first step creates a graph to represent the floor plan. Each vertex in the graph represents a room in the floor plan. It then distributes rooms over the footprint according to the generated graph. Finally, the approach expands rooms using the Monte Carlo method to allow the rooms to grow or shrink.

Marson and Musse [16] proposed an algorithm for automatic generation of floor plans based on tree-maps provided to the program. In their approach, the initial shapes are subdivided into floor plans according to a given tree-map. Every child room in the tree-map is generated from the space of its parent. Therefore, the results maintain the hierarchical relations, while the adjacency relations between rooms are not guaranteed.

The shape grammar [19] has been widely used to generate and analyze building layouts. It is able to model spatial relations of geometries in paintings and sculptures. Shapes expand automatically with given rules in a shape grammar, providing the possibility for programs to generate different layouts by applying rules. Harada et al. [20] developed an interactive system with the shape grammar for generating building layouts.

Wonka et al. [14] proposed the split grammar, a parametric set grammar based on the concept of shapes, for automatic generation of building designs. To increase the efficiency and automation of rule derivations, a shape grammar called CGA shape [21] is proposed, which extends the split grammar by introducing the component split and the mass modeling functionality. The same authors also proposed a set of algorithms that derive high-quality 3D models from single facade images with arbitrary resolutions [22]. Wu et al. [23] later proposed an algorithm to automatically derive split grammars from facade layouts.

Instead of using a shape grammar, Merrel et al. [17] generated three-dimensional buildings with internal structures based on a given set of high-level requirements. Lin et al. [24] proposed a hierarchical approach for generating various 3D models that resemble an input piece with the same style. These approaches set a solid foundation for automatic building generation, but are unable to control the positions and adjacency of internal rooms.

In summary, none of the aforementioned approaches has been proposed to perform the complete design process for designers to generate a set of floor plans according to the required adjacency relations from the input specification. Some approaches require redundant control vertices added to the input graph or undesirable rooms would be created. Others may generate floor plans according to shape grammars, but cannot guarantee the resulting style and shapes of rooms.

3. Preliminaries

Considering a floor plan as a planar graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where each edge represents a wall, and each vertex represents an intersection of