Abstract

Existing floor plan retrieval methods match residential floor plans based on room function and adjacency, ignoring the shape of the interior rooms. Inspired by shape grammars, we incorporate interior layout into the similarity metric using a tree structure that represents both the layout hierarchy and room shapes of the floor plan. We create parse trees from floor plans and evaluate their similarity using an appropriately defined tree edit distance. We evaluate the method on a public dataset of 11,250 vector graphic representations of Japanese homes. A user study shows that our method retrieves layouts preferential to those obtained using the exterior outline or room adjacency matrix.

1 Introduction

Floor plan retrieval engines identify recommendations for home buyers from consumer databases or act as a means for architectural inspiration. The task is frequently framed as a graph matching problem due to the innate representation of floor plans as graphs and the effectiveness of graphs as a pattern recognition tool [2]. Graphs representing features of the floor plan topology or room function have been compared using a variety of graph matching techniques [1, 6, 9, 15, 16, 17, 18, 19, 20, 25, 26]. However, many existing techniques do not explicitly account for the shape of the layout, i.e. how the rooms are arranged spatially, which is desirable for finding similar residences in databases. Incorporating features such as the room footprint [1] gives consideration to the shape of the floor plan exterior, but it does not weight shapes in the interior room layout. A challenge in incorporating the interior layout is that no clear metric permits its comparison. Many metrics for generic shape matching have been proposed [24]. However, homes exhibit a hierarchy of functional zones, which are collections of single rooms [9]. This hierarchical nature is not accounted for when applying a shape similarity metric.

Inspired by shape grammars [21, 30], we approach the floor plan retrieval problem by representing floor plans as trees that can be efficiently compared using the tree edit distance [22]. This distance is the minimum-cost sequence of node edit operations to transform one tree into another, and it can be computed in polynomial time using the Zhang-Shasha algorithm [29]. We developed a cost function for tree edits that incorporates the layout hierarchy. Trees are constructed from vector graphic representations of floor plans, which can be obtained using the method in [7, 10].

Figure 1: Floor plan matching. We define a distance between floor plan images based on the room layout. Instead of working with the input (top) or segmented images (middle), we use a graph representation (bottom), where edges represent walls and nodes represent intersections. Nodes are colored by degree.

The retrieval method is shape-based and does not consider the functionality of different rooms. Methods that consider different room functionalities could be incorporated as a filter or as an additional weighted cost in the layout similarity. We applied our method to 11,250 Japanese homes (see Figure 1 for examples) selected from the dataset made available by Liu et al. [10]. The novelty of this work is that we incorporate shape hierarchy into shape similarity using a tree edit distance matching metric. We thus provide a solution for floor plan retrieval by introducing a shape matching method for hierarchical structures.

2 Prior Work

Existing approaches to floor plan matching compare functionality and room adjacency and do not explicitly account for room shape in the similarity metric. Spectral graph matching was applied to floor plan matching in [6, 17]. Schaffranek compared office layouts using room function and a weighted adjacency matrix dependent on the physi-
3.1 Floor Plan Parsing

The goal is to convert a floor plan to a binary parse tree that represent steps of a split grammar. Since floor plans are not well represented by independent splits along coordinate axes, we employ a non-binary split grammar [27]. Binary split grammars as presented by Teboul et al. [23] are only capable of dividing shapes into rectangles. A non-binary split grammar permits representation of a more diverse set of room shapes, since shapes after a split can be any pair of basic shapes from the vocabulary. Our parsing problem remains tractable since the floor plan images to be parsed are binary images that are considerably more simple compared to the RGB facade images studied in [23].

We define the split grammar as follows. The basic shapes are in the set $B$ of shapes formed from the union of three rectangles $\{R_1, R_2, R_3\}$. The rectangles are connected and axis-aligned. Basic shapes have no holes, and are labeled as terminal or non-terminal. At each split, a non-terminal shape is selected and the split rule is applied. A shape is terminal if it contains exactly one room. Examples of valid splits are shown in Figure 2. We found the shape representation to be a reasonable trade-off in terms of accuracy and simplicity, where the majority of shapes in the dataset are accurately captured, while curved shapes lead to an approximation error. The $N$ leaves of the tree represent single rooms. Internal nodes are combinations of adjacent single rooms and have a shape that is the union of its children. Conversely, the two children of an internal node are the result of an applied split. The root node contains all rooms and its shape is the footprint of the floor plan. Nodes are labeled with a feature vector discussed in Section 3.2.

We build the parse tree using a greedy, bottom-up algorithm to obtain near equal area splits close to the root node while respecting room adjacency. The tree parsing process begins with leaf nodes which are initially disjoint. At each iteration, the area of all remaining pairs of disjoint trees (room combinations) that satisfy room adjacency are summed. The pair with the smallest sum-area is combined. This is repeated until all rooms have been combined. Parse trees obtained using this method are shown in Figure 3.
Fig. 4: Examples of room shapes inscribed using rectangles. Dotted outlines represent rectangles used to calculate the affine transforms. A non-rectilinear shape is not completely parameterized by three rectangles.

Table 1: Features used in tree edit distance cost.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width and height of $R_2$</td>
<td>2</td>
</tr>
<tr>
<td>Affine transformations $(s_2, t_2, l_2, a_2)$ that map $R_1 \rightarrow R_2$ and $R_1 \rightarrow R_3$</td>
<td>8</td>
</tr>
<tr>
<td>Area of $R_2 \cap R_1$ and $R_1 \cup R_2$</td>
<td>2</td>
</tr>
<tr>
<td>Area not covered by $R_1 \cup R_2 \cup R_3$</td>
<td>1</td>
</tr>
</tbody>
</table>

3.2 Labeling

Each tree node is labeled with a feature vector obtained from simplifying the room shape. Rooms are simplified to a shape from $B$ by inscribing up to three overlapping rectangles in the room boundary. $R_1$ is assigned the largest inscribed rectangle in the room boundary. The largest inscribed rectangle can be solved for a generic polygon by rendering the room footprint and representing it as a Cartesian tree [13]. $R_2$ and $R_3$ are inscribed in the remaining area, and extended to maximize overlap with $R_1$ while remaining within the polygon bounds. Examples of rectangles inscribed using this method are shown in Figure 4. $R_1$, $R_2$, and $R_3$ are used to obtain a translation-invariant feature vector $v$ with 13 values described in Table 1.

3.3 Matching

We use the tree edit distance [22] as a similarity metric between query and candidate parse trees. To compute tree edit distance, the edit operations (relabel, add, delete) are assigned a cost. The trees must also be ordered for tree edit distance to have polynomial complexity [29, 28]. Tree nodes are ordered first by the number of children, then by the room area if the number of children are equal. The cost to relabel node $A$ to $B$ is defined as:

$$C(A, B) = ||\text{area}(A) v_A - \text{area}(B) v_B||_1,$$

where, $\text{area}(\cdot)$ is the areas covered by the parameterization of the shape at the node, and $||\ast||_1$ is the mean absolute difference. The edit cost is symmetrical and satisfies the triangle inequality [29]. For deletion and insertion, $v_B = 0$ and $\text{area}(B) = 0$. Intuitively, $||\ast||_1$ penalizes differing shapes and the number of inscribed rectangles. The area factors make shapes closer to the root and overall footprint more costly to modify due to their greater area.

4 Experiments

Dataset. 11,250 floorplans were selected from the dataset by Liu et al. The cost function and feature vector was designed to provide good matches on 1250 floor plans. The remaining 10,000 were used as a test set for the preference study. The study was conducted using an online form, which participants completed independently. Method names were replaced by letters and the order randomized to avoid bias. Example floor plan retrievals shown in Section 5 are sampled i.i.d. from the set of 1250 floor plans.

Baseline methods. The tree retrieval method was compared to footprint matching [1] and graph matching using the Laplacians of the room adjacency matrix as in [17]. For footprint matching, the distance of the exterior from the centroid of each floor plan was calculated in pixels with rays drawn at increments of $5^\circ$. The matching floor plan minimizes the Euclidean distance between the query and candidate feature vector. For graph matching, the room layout is represented as an undirected weighted graph. Edge weights are the reciprocal of the distance between room centroids if rooms are adjacent and zero otherwise. The $l_2$-distance of the spectrum of the normalized Laplacian $N$ was used as the similarity metric [17]. The spectrum is zero padded to the maximum room count to account for different numbers of rooms.

4.1 Layout Preference Study

Twenty-five random floor plans were chosen from the test set. Floor plans that appeared to have missing rooms were removed from the test. Rooms were rendered with a single color, ignoring the room function in order to emphasize the internal shape. For each query, the top four matches obtained using each matching method were presented simultaneously, similar to Figure 5 (with labels removed). Thirty-six participants were asked to select which group was most similar to the query based on layout, room shape and relative sizes of rooms. They were asked to consider the query layout under rotation and reflection. A ‘no preference’ option was provided for the case that the participant felt no groups were similar to the query.

5 Results

5.1 Comparison of Matching Methods

Figure 5 shows results for query plans from the development set where floor plans with identical footprints exist but have vastly different interior layouts. Spectral matching finds layouts with similar room clustering, but does not respect the footprint of the floor plan. The spectral matching approach is preferential towards floor plans with the same number of rooms. In Figure 5(a), the shape matching preserves the division of the large room and balcony combined with an opposite half with a simple layout and few rooms. Since many plans with the query footprint exist, the footprint match chooses floor plans with a different
interior layout. In Figure 5(b), there is a distinctive ‘T’ feature (indicated by blue arrows) that appears in all matches obtained by the proposed shape-based method. In the footprint match, this feature incidentally appears as indicated by the orange arrow. Figure 6 shows matches obtained for floor plans without identical footprints. The shape matching method discovers floor plans with similar footprints that better adhere to the division and complexity of the query.

5.2 Preference Study on Test Set

Parsing floor plans in the test set took approximately 10 hours, but only needs to be calculated once. We computed the tree edit distance against 10,000 candidates, and the average response for each category, and responses for each of the twenty-five cases are shown in Figure 7. The most preferred method of the average participant was the shape-based method (53% ± 11%). Inter-user variability was relatively high with shape-based preference ranging from a minimum of 16% to a maximum of 76%. Users with less preference for the shape-based method were more attentive to footprint and clustering rather than individual room shape. Cases where the spectral method was selected the majority of the time had simple footprints and small numbers of rooms.

6 Discussion

We have presented a method for incorporating layout shape in the floor plan retrieval process. This was achieved by parsing floor plans into a tree structure that represents a split grammar and comparing the parse trees with a shape-based tree edit distance. A preference survey indicated that our retrieval method identifies layouts with higher similarity to the query than layouts obtained using previously proposed retrieval methods.
References


