Drawing-based Procedural Modeling of Chinese Architectures

Fei Hou, Yue Qi, *Member, IEEE* and Hong Qin, *Member, IEEE*

Abstract—This paper presents a novel modeling framework to build 3D models of Chinese architectures from elevation drawing. Our algorithm integrates the capability of automatic drawing recognition with powerful procedural modeling to extract production rules from elevation drawing. First, different from the previous symbol-based floor plan recognition, based on the repetitive pattern trees, small horizontal repetitive regions of the elevation drawing are clustered in a bottom-up manner to form architectural components with maximum repetition, which collectively serve as building blocks for 3D model generation. Second, to discover the global architectural structure and its components' interdependencies, the components are structured into a shape tree in a top-down subdivision manner and recognized hierarchically at each level of the shape tree based on Markov Random Fields (MRF). Third, shape grammar rules can be derived to construct the 3D semantic model and its possible variations with the help of a 3D component repository. The salient contribution lies in the novel integration of procedural modeling with elevation drawing, with a unique application to Chinese architectures.

Index Terms—Procedural modeling, Elevation drawing segmentation, Elevation drawing recognition, Chinese architecture.

1 Introduction

Chinese architectures, which are becoming increasingly significant in the fields of urban simulation, restoration of ancient civilization, cultural heritage preservation, modern city planning, and digital entertainment, have their unique structures with complex decoration and appearance. A vast majority of ancient Chinese architectures may not exist any more nowadays due to historical evolution that spans across several thousand years. Yet, drawings are readily available for not only extant architectures but also non-existing ones. More than 20,000 archives of *Yang Shi Lei* (the family in charge of the imperial architecture construction during the Qing Dynasty) have been collected into the National Library of China so far. These historical documents, some of which are drawings, have been playing a critical role in the research and development of Chinese architectures. In order to help architects, city planners, Chinese architectural researchers, and even movie makers construct 3D models, we design a prototype system to build 3D models of Chinese architectures from drawing procedurally.

This research is critically relevant to drawing recognition and procedural modeling. Grammar-based procedural modeling technology has shown its power in architectural modeling [1] [2]. The grammar rules are always encoded according to the semantic interpretation of architectures by way of top-down splitting and they are proven to be efficient in large-scale outdoor scene modeling. In contrast, the text-based systems rely on domain expertise to design grammar rules manually. There has been much work to convert 2D drawings into 3D models [3]. Nonetheless, existing work is mainly focused on modeling interior structure from floor plan based on symbol recognition. In this paper, we aim to build the high-fidelity complex façade with unique outdoor appearance for Chinese architectures from elevation drawing.

Chinese architectures always tend to emphasize breadth rather than height. The architectural components, such as tiles, windows, doors, and columns, are repetitive along the horizontal direction. Strongly inspired by this observation on repetition, we encode the repetitive pattern of a horizontal repetitive region group in a Repetitive Pattern Tree (RPT), and cluster region groups with compatible repetitive patterns together in a bottom-up manner in order to segment the elevation drawing into semantically-sound architectural components. After segmentation, we subdivide the elevation drawing vertically and horizontally into components in a top-down manner to derive a shape tree of the architecture. In particular, we construct a Markov Random Field (MRF) of component groups from every level of the shape tree to recognize components hierarchically in order to get the semantic shape tree. With user-specified simple parameters, the building depth is inferred. Shape grammar rules are derived based on the semantic shape tree and architectural knowledge. Finally, the rule-driven deformable models are generated for 3D Chinese architectures. The main contributions of this paper include:

- Bottom-up clustering and image segmentation: in-

- *F. Hou is with the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, 100191, China. E-mail: houfei@vrlab.buaa.edu.cn*
- *Y. Qi is with the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, 100191, China. E-mail: qy@vrlab.buaa.edu.cn*
- *H. Qin is with the Department of Computer Science, Stony Brook University (SUNY Stony Brook), Stony Brook, NY, 11794-4400. E-mail: qin@cs.sunysb.edu*

spired by the repetition of architectural components, we are able to cluster pixels into meaningful regions, followed by region grouping and component grouping, to segment the drawing into semantically-correct components in a bottom-up manner. Potential repetitive patterns of every region group are encoded in the structure of a repetitive pattern tree for clustering.

• Top-down subdivision and component recognition: we then subdivide the drawing horizontally and vertically into components in a top-down manner to derive a shape tree of the architecture. We propose a heuristic method to construct Markov Random Field of component groups at every level of the shape tree to recognize components hierarchically by minimizing their energy using belief propagation.

• Shape grammar rule derivation and model generation: shape grammar rules are derived from the semantic shape tree based on priors and the building depths are inferred based on user-specified simple parameters. Rule-driven deformable models are generated from the derived shape rules, and semantically-correct 3D Chinese architecture is obtained with shape variation.

2 Related Work

Our work relates to the areas of drawing recognition, architectural modeling, and procedural modeling. We briefly review related work in the following categories.

Image-based architectural modeling. Architectural modeling from images has been extensive studied [4] [5] [6]. Xiao [7] [8] built street-side 3D buildings from street-view image sequences. Sinha [9] presented an interactive system to build architectures to be a piecewise planar model from unordered photo collections. Jiang [10] calibrated camera from single image based on symmetry to build symmetric architectures. However, image-based modeling is difficult to handle details and it is also expensive for large-scale modeling.

Procedural architectural modeling. Procedural modeling [11] [12] is an efficient way aiming at automatic generation of 3D models. Stiny [13] introduced the basic idea of shape grammar for architectural design, but it is too complex for direct application in computer graphics. Recently, based on shape grammars, Wonka [1] generated façade details using split grammar, which splits the façade hierarchically in a top-down manner. Extending split grammar by introducing the notation of mass model and context sensitive grammar, CGA shape grammar [2] is capable of generating massive urban models. The procedural modeling fulfills the requirement of automatic generation of models and presents a semantic interpretation of the model. But the text-based procedural modeling systems appear to be only useful for expert users. To facilitate automatic rule derivation, Lipp [14] introduced a visual editing system with direct local control for procedural architectures. Aliaga [15] introduced the style grammar, which was derived from user-subdivided input images, to create new buildings with the same style in a fast and intuitive manner. Müller [16] presented a framework to subdivide a façade image into elements in a top-down manner to form a shape tree from which shape grammar rules are derived automatically to generate models. These methods are either based on intense interaction or only capable of expressing simple façade layout. Whiting [17] introduced physical constraints into procedural modeling to construct structurally-stable buildings.

Drawing-based architectural modeling. Drawing-based modeling aims to convert 2D drawings to 3D buildings. Much work has been limited to low-level recognition, such as vectorization [18], text extraction [19], and symbol recognition [20] [21]. High-level recognition is much more complicated. Lewis [22] presented a system to build 3D polygonal models from floor plans. Lu [23] designed a system to construct detailed interior structure from computer-generated construction drawings for buildings. But they all failed to handle raster images of drawings. Dosch [24] introduced a framework to generate buildings from drawing images. After vectorizing the raster image to sets of polylines and arcs, they recognized the predefined architectural symbols and extruded 3D models for each floor separately. These methods take as input floor plans to primarily build the interior structure of architectures. They intended to recognize the predefined symbols based on vectorization and template matching in order to understand the drawings. Different from the vectorization-based methods, our algorithm does not have to undergo vectorization at the beginning. We have to analyze the elevation drawing to build the complex façades of Chinese architectures.

Architectural façade recognition. Chen [25] interpreted the architectural sketch using maximum likelihood based on shape and location features of the strokes. Prior image-based work also attempted to recognize architectural components, such as windows and doors based on Bayesian model [6] or conditional random field [26]. Compared with images, drawings contain fewer data with only lines and curves. To the best of our knowledge, prior work made no attempt to recognize elevation drawings. We develop a modeling framework to recognize the façade in graphical models via energy minimization. The energy minimization or inference on a graphical model (e.g., MRF) has been extensively studied [27] [28] [29]. The belief propagation [27] [30] on tree-structured graphs is an exact solution. In contrast, the loopy belief propagation [31] [29] and graph cut [32] [33] on general graphs are an approximated solution for general cases. These methods have been extensively studied in low-level vision [34] [32] [33], image completion [35], image-based architectural modeling [7] [8], etc.

Symmetric and repetitive pattern analysis. Man-made architectures always exhibit symmetric and repetitive characteristics to a certain extent. Many algorithms have been designed to discover symmetries in 3D models [36] [37] [38] [39] [40]. However, these analytic methods for 3D models are not suitable for drawings. Liu [41] detected the periodic pattern of images based on frieze and wallpaper groups, but it can only handle repetitive patterns of the entire image. Loy [42] and Cornelius [43] detected symmetries based on feature descriptors (such as SIFT) in images, but the image descriptors can hardly work for the drawings.

3 Brief History and Key Elements of Chinese Architectures

Chinese architectures, as an important component in both ancient civilization and the historical evolution of world architecture, have their unique structures that are mainly regulated by two classic books of *YingZao FaShi* (i.e., building standards) published in 1103 A.D. (Song Dynasty) and *GongCheng ZuoFa ZeLi* (i.e., structural regulations) published in 1733 A.D. (Qing Dynasty). The books, however, were written in old terminologies and expressions. In the 1930s, the pioneering scholar LIANG, Sicheng [44] [45], who regarded them as grammar books of Chinese architectures, studied them and translated the *GongCheng ZuoFa ZeLi* into drawings [46]. Li [47] explicitly perceived that the system of *YingZao FaShi* was parametric and rule-based, and tried to interpret it using shape grammar.

The component names are illustrated in Fig. 1. As shown in the left figure, Chinese architectures exhibit bilateral symmetric and repetitive features prominently along the horizontal direction, such as the tiles, bracket sets, columns, windows, and doors. The bay, which is always odd and bilateral symmetric, is a basic building block indicating the area between columns. Vertically, the architecture can be roughly subdivided into levels of platform, timber framing and wall, roof, etc. The timber framing is used to frame the building, which determines the global structure and dimension of the building, whereas the windows, doors, and walls between the columns are serving the purpose of separation. As illustrated in the side sectional drawing [44] (also see Fig. 1 right), the roof is borne by a timber skeleton which consists of columns, beams, purlins, and rafters, where the cross beams are borne by the columns. The purlins, which are horizontal members bearing the rafters, are positioned along shoulders of the beams. The rafters are short, stretching down only from purlin to purlin. The horizontal projections of rafters are equally long, named as *bujia*, except at eaves where they are extended, named as *chuyan*, nevertheless, the height of the raise of purlins (i.e., the heights of the small columns on the beams) divided by the *bujia*, named as *jujia*, increases as the purlins rise.

4 Algorithmic Overview

As shown in Fig. 2, the entire algorithm can be mainly divided into three stages: 1) drawing segmentation, 2) shape tree derivation and component recognition, and

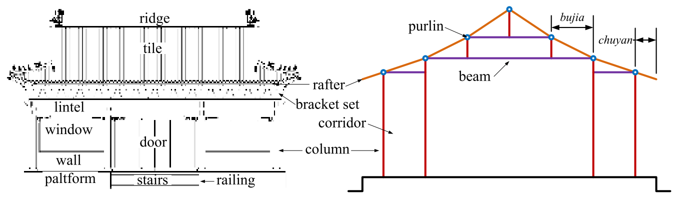

Fig. 1. Left: façade sketch map of Chinese architecture. The components are labeled in the drawing. Right: side sectional drawing sketch maps of Chinese architecture. The columns, beams, purlins, and rafters are shown in different colors.

3) rule derivation and model generation. We give a brief description of each individual stage below:

Drawing segmentation (Section 5): this stage describes how to segment the drawing into meaningful components. Inspired by the repetitive pattern of every type of components, different from previous vectorization-based methods, we segment the drawing into different groups of components, based on the bottom-up clustering of regions. The components in a group are repeated horizontally.

Semantic shape tree derivation (Section 6): in this stage, we split the drawing in a top-down manner to structure the components in a shape tree and recognize the components in the shape tree hierarchically based on MRF.

Rule derivation and model generation (Section 7): in this stage, rules to generate the models are derived from the semantic shape tree with architectural knowledge. The building depth is inferred based on their width differences and the rule-driven deformable architectural models are generated from the rules with the help of a 3D component repository.

5 Drawing Segmentation

In this stage, we segment the drawing into meaningful components in a bottom-up region-clustering procedure. Inspired by the repetition of components, as shown in Fig. 3, given the input drawing (Fig. 3(a)), we first cluster pixels into regions and region groups (Fig. 3(b), and Section 5.1). The region groups exhibiting different repetitive patterns should be combined to form components. In Section 5.2 we introduce the concept of repetitive pattern tree that encodes the potential repetitive patterns contained in a region group. Second, based on the repetitive pattern tree, the region groups with compatible repetitive patterns are clustered to vote for potential repetitive patterns of meaningful components (Fig. 3(c), and Section 5.3). Finally, based on the repetitive patterns, the region groups are clustered to form meaningful component groups (Fig. 3(d), and Section 5.4).

5.1 Region and Region Group

Noise region filling. A region is a closed area surrounded by black pixels in the drawing, which is the

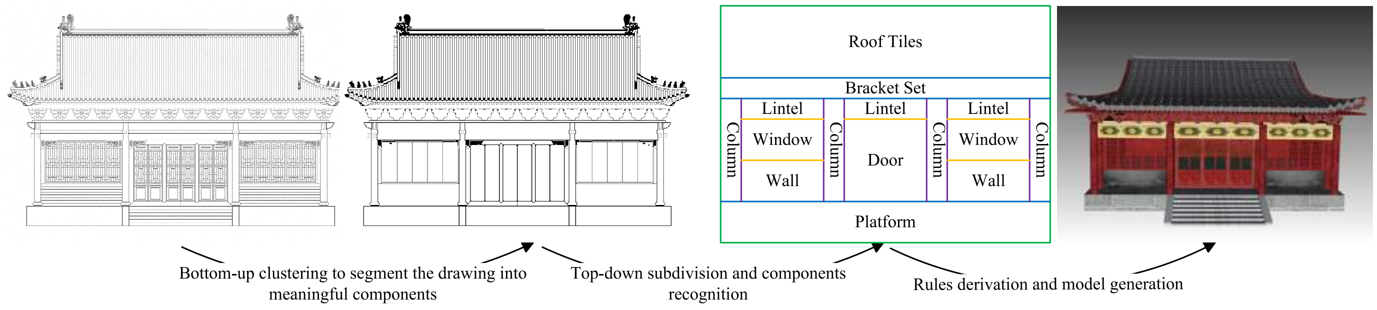
Roof Tiles
Bracket Set
Lintel
Window
Wall
Column
Door
Platform
Bottom-up clustering to segment the drawing into meaningful components
Top-down subdivision and components recognition
Rules derivation and model generation

Fig. 2. The pipeline of the drawing-driven procedural modeling. The regions are clustered in a bottom-up manner in order to segment the drawing into meaningful components. The drawing is subdivided by different groups of colored segments in a top-down manner into a shape tree and the components are then recognized. The rules are derived from the shape tree and the model is generated.

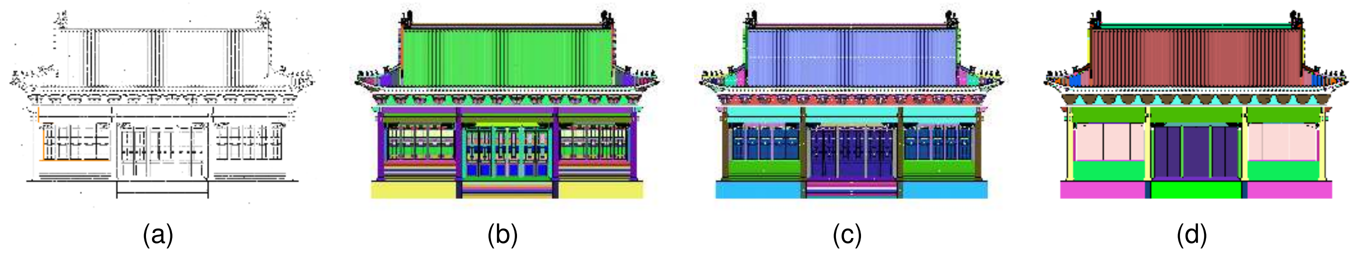

Fig. 3. The drawing segmentation procedure. Every group is colored differently for distinction. (a) The input drawing with user-drawn orange segments used to prevent over-aggressive grouping. (b) The drawing is first segmented into region groups. (c) The region groups with compatible repetitive patterns are clustered to vote for repetitive patterns. The region groups in the same equivalence classes are painted in the same color and the pattern centers are painted with white circles. (d) Finally, based on the repetitive patterns, the region groups are clustered to form meaningful component groups.

basic element of a component. It can simply be detected by the flood fill technique, however, in practice it is always corrupted due to noise. To combat noise, the drawing is pre-processed by the morphological open/close operation to close small areas and gaps, and later the drawing is filled by flood fill to detect connected regions which may also consist of multiple desired regions in one connected region due to gaps. Every connected region, which is processed independently during the next operation, is placed in a big enough sub-image (Fig. 4(a)), where pixels in the region are white and the others are black. We dilate white pixels (which are followed by the erosion operation) and pixels changed from black to white are marked as inner pixels (Fig. 4(b)) (i.e., these pixels may belong to some edges that may divide a connected region). Later, we dilate inner pixels to close potential gaps followed by the thinning operation to extract their skeletons (Fig. 4(c)) and the skeleton pixels are set to black in the original sub-image. At last, the sub-image is divided by flood fill again to detect desired regions (Fig. 4(d)). Some improper fillings do not affect the final results, such as some improper regions of the bracket sets in Fig. 3(b), which will be solved in the following steps. If the noise results in serious improper regions, we have to spend several minutes to slightly repair the drawing before processing.

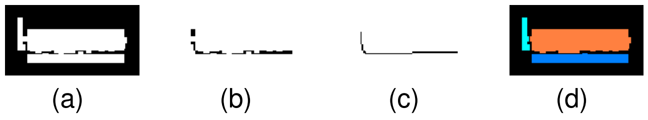

Fig. 4. (a) A connected small region which is placed in a big enough sub-image. (b) The extracted inner pixels with potential gaps. (c) The skeleton of the dilated inner pixels. (d) The detected 3 regions after gap closing.

Region group generation. The horizontally-repeated regions are clustered into region groups (Fig. 3(b)), in which the region locations in the vertical direction are mutually overlapped and the region shapes are similar. To build the region groups, regions are first clustered by their vertical locations, and later the regions with the same height are clustered based on their areas (i.e., the number of pixels) and bounding boxes to form the region groups, which are the basic repeated elements facilitating the following clustering operations.

5.2 Repetitive Pattern Tree

In Fig. 5(a), two region groups that are colored exhibit different repetitive patterns, but they should be combined to form the doors. We notice that the region group

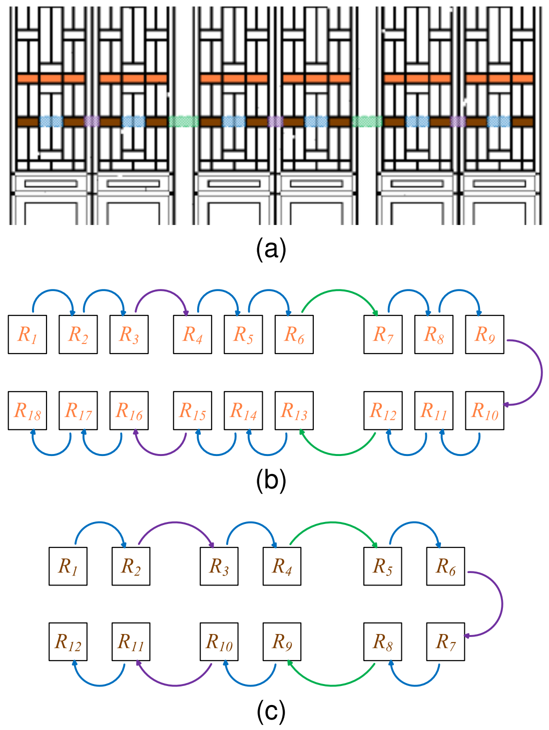
(a)
R1 R2 R3 R4 R5 R6 R7 R8 R9
R18 R17 R16 R15 R14 R13 R12 R11 R10
(b)
R1 R2 R3 R4 R5 R6
R12 R11 R10 R9 R8 R7
(c)

Fig. 5. (a) Two region groups are painted in orange and brown, respectively. The regions between the brown region groups are categorized and painted in dashed lines with different colors. (b) and (c) are chains of the two region groups and the edges are categorized and painted in different colors.

exhibits hierarchical repetition. For every region group, we define a Repetitive Pattern Tree (RPT) to encode all the potential repetitive patterns contained in the region group. The RPTs of the two region groups are shown in Fig. 6 in which the leaves represent the regions and every level represents a potential repetition.

A region group forms a chain (e.g., Fig. 5(b) and Fig. 5(c)) in which each vertex represents a region and each edge connects adjacent regions. In order to evaluate possible repetitions within a tree, we sort the edges into different categories to identify whether the areas in the drawing between adjacent regions (e.g., the dashed areas in Fig. 5(a)) are identical. To compare the areas, for every dashed area, we compute its horizontal length and a histogram to document the area ratios of different region groups covered by it. Whether two edges are identical is compared by their lengths or histograms. Fig. 5(b) and Fig. 5(c) show a chain of the two region groups in Fig. 5(a). The edge categories are represented by different colors.

The RPT is constructed by subdividing the chain recursively. The subdivision of a chain is the collection of all the connected subchains after removing some categories of edges. A repetitive subdivision is a subdivision in which all the subchains are the same, i.e., they have the same number of edges along with the same sequences of edge categories. To analyze repetitive patterns implied in a region group is to compute all the repetitive subdivisions of the chain. The repetitive subdivisions can be represented hierarchically in the RPT and constructed by removing some categories of edges iteratively.

Theorem 1: Let $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ be the n categories of edges of a chain and $|\mathcal{E}_i|$ be the number of edges in $\mathcal{E}_i$. If $|\mathcal{E}_i| \leq |\mathcal{E}_j|$, $\mathcal{E}_j$ can not be removed before $\mathcal{E}_i$ is removed.

Proof: Suppose $\mathcal{E}_j$ is removed before $\mathcal{E}_i$ is removed, then the chain is divided into no less than $|\mathcal{E}_j| + 1$ subchains after removing $\mathcal{E}_j$. Since $|\mathcal{E}_i| \leq |\mathcal{E}_j| < |\mathcal{E}_j| + 1$, the subchains can not be repetitive. This immediately gives rise to the contradiction. □

Repetitive Pattern Tree (RPT) construction. The RPT represents the potential repetitions hierarchically in which each level represents a potential repetitive subdivision. Based on Theorem 1, in the process of constructing the RPT of a region group, we know if $|\mathcal{E}_i| < |\mathcal{E}_j|$, $\mathcal{E}_i$ is removed before $\mathcal{E}_j$, and if $|\mathcal{E}_i| = |\mathcal{E}_j|$, they are removed at the same time. In each iteration of constructing the RPT, we remove the least category of edges and check whether the remaining subchains are exactly the same. If they are identical, the subchains form a level of the RPT (i.e., a potential repetition), or the next iteration starts. Until no edges being left, all the individual regions form the leaves of the tree. Let us consider the two chains in Fig. 5(b) and Fig. 5(c) as an example. In three iterations, the green, blue and purple edges are removed successively, forming three levels of the RPTs. At the end no edges are being left. The RPTs of the two region groups are shown in Fig. 6.

5.3 Repetitive Pattern Voting

This step is to extract the repetitive patterns of meaningful components based on extracting the common patterns of the RPTs as shown in Fig. 3(c). We first describe how to extract the common repetitive pattern of two RPTs and later describe the multiple RPTs voting.

Pair RPTs matching. The pair RPT matching is to extract the common repetitive pattern of the two RPTs. A repetitive pattern $\mathcal{P}$ is a set of centers $\{c_1, c_2, \ldots, c_n\}$ that define the recurrence positions of the repetitive elements. Each level of the RPT represents a repetitive pattern where each node denotes a repetitive element and the center of a node is the average center of all the regions in the subtree of the node. Let $\mathcal{P}_1 = \{c_1, c_2, \ldots, c_r\}$ be the pattern of a certain level of RPT_1 and $\mathcal{P}_2 = \{c'_1, c'_2, \ldots, c'_s\}$ be the pattern of a certain level of RPT_2. The repetitive patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ are of equivalence iff. $r = s$ and $\forall i \forall j (c_i - c'_i = c_j - c'_j), i = 1, \ldots, r,\ j = 1, \ldots, s$. If all the repetitive patterns of level 0 to level n of RPT_1 and RPT_2 are of equivalence and the patterns of level $n + 1$ are not equivalent, then the RPT_1 and RPT_2 are called compatible at level n and denoted by $level(RPT_1, RPT_2) = n$. In Fig. 6, the patterns of the two RPTs are of equivalence at level 0, level 1, and level 2. So their common pattern is the pattern of their compatible level 2, which indicates the repetitive pattern of the six doors in Fig. 5(a).

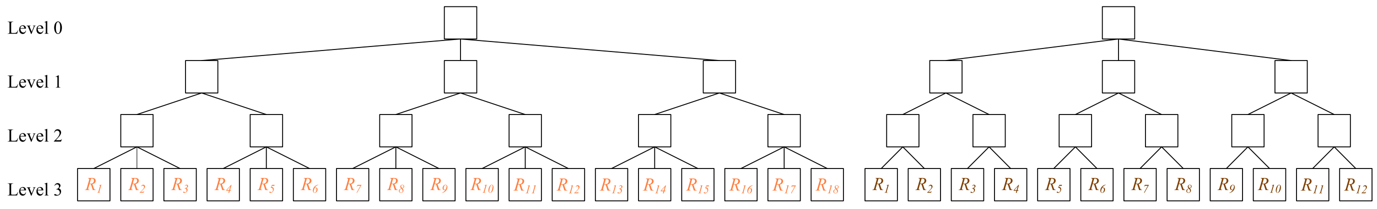

Fig. 6. The two RPTs of region groups in Fig. 5. Every node refers to all the regions (leaf nodes) in its subtree. Left: the RPT of the brown region group of Fig. 5(a). Right: the RPT of the orange region group of Fig. 5(a).

Theorem 2: If two RPTs are compatible at a level, they are called compatible. The compatible relation between RPTs is an equivalence relation.

Proof: It is clear that the compatible relation is reflexive and symmetric. If $level(RPT_1, RPT_2) = n$, $level(RPT_2, RPT_3) = m$, and $m < n$, then $level(RPT_1, RPT_3) \geq m$. So it is transitive and the compatible relation is thus an equivalence relation. □

Therefore, given multiple RPTs, the RPTs can be clustered into equivalence classes. The compatible level of an equivalence class is defined as the minimum compatible level among all pairs of the RPTs in the class. Since the relation is equivalent, the compatible level of a class equals to the minimum compatible level between a certain RPT and all the other RPTs in the class, which reduces the comparison complexity. The common pattern of an equivalence class is the pattern of its compatible level.

Multiple RPTs voting. This step is to cluster multiple RPTs to evaluate the equivalence classes of region groups and their common patterns. We build a region group adjacent graph G_{adj} to assist the procedure, where every vertex is a region group and two vertices are adjacent iff. the two region groups are adjacent in the drawing. In each iteration, arbitrary node is selected as a seed to iteratively grow to cluster adjacent ones with equivalent repetitive pattern in a breadth-first manner until no one will be clustered. The goal of this process is to vote for the common pattern of the connected compatible vertices. The clustered vertices are removed from G_{adj} and later restart a new iteration for a new repetitive pattern. The confidence of a repetitive pattern is the number of regions voting for it and the pattern is supposed to be a pattern of a component group in Section 5.4. The algorithm framework is shown in Fig. 8.

It is clear that the algorithm votes for the repetitive patterns with maximum repeated elements, e.g., in Fig. 7 the windows, walls and columns are over grouped due to compatible repetitive patterns. To prevent over-aggressive grouping, the user should draw some vertical or horizontal separating segments interactively. If two region groups are separated, they will not be clustered. Fig. 3 shows the orange separating segments, correct repetitive patterns and components.

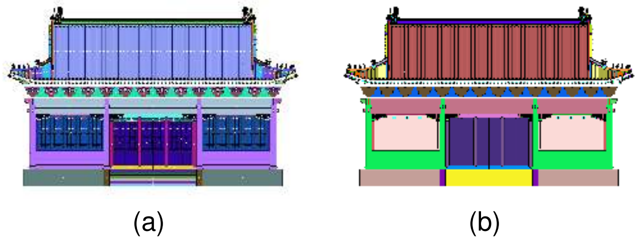

Fig. 7. (a) and (b) show the incorrect repetitive patterns and component groups respectively without separating segments. The windows, walls and columns are over grouped.

```
1:  build region group adjacency graph $G_{adj}$
2:  set repetitive pattern list $\mathcal{P}_{list} \leftarrow \emptyset$
3:  while $G_{adj}$ is not empty do
4:      select arbitrary vertex $\mathcal{R}$ as a seed
5:      breadth-first search for non-separated vertices
        equivalent to $\mathcal{R}$ starting from $\mathcal{R}$
6:      insert the new pattern $\mathcal{P}$ to $\mathcal{P}_{list}$
7:      remove all equivalent vertices from $G_{adj}$
8:  end while

9:  rebuild region group adjacency graph $G_{adj}$
10: while $G_{adj}$ is not empty do
11:     $\mathcal{P} \leftarrow$ the maximum confidence pattern in $\mathcal{P}_{list}$
12:     remove $\mathcal{P}$ from $\mathcal{P}_{list}$
13:     breadth-first search for non-separated and approximately equivalent vertices starting from the
        equivalent vertices
14:     combine the (approximated) equivalent region groups to form a new component group and re-
        move them from $G_{adj}$
15:     update the associated region groups and repetitive
        patterns
16: end while
```

Fig. 8. The algorithm framework for segmenting the drawing. Line 1 to Line 8 is repetitive pattern voting (Section 5.3) and Line 9 to Line 16 is region group clustering (Section 5.4).

5.4 Region Group Clustering

Based on the voted repetitive patterns, which imply the repetitive patterns of horizontally repeated meaningful components (i.e. component groups), this step is to further cluster the region groups into component groups as shown in Fig. 3(d). To complete the lost elements due to occlusions or improper clusterings in the previous

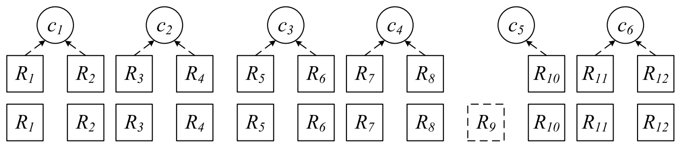
c_1
c_2
c_3
c_4
c_5
c_6
R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_{10} R_{11} R_{12}
R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_9 R_{10} R_{11} R_{12}

Fig. 9. Region group and repetitive pattern matching. The repetitive pattern $\mathcal{P}$ is $\{c_1, c_2, c_3, c_4, c_5, c_6\}$ and the region group $\mathcal{R}$ is $\{R_1, \dots, R_8, R_{10}, \dots, R_{12}\}$. The matching between regions and pattern centers are indicated by the dashed arrows. We get $\mathcal{R}' = \{R_1, \dots, R_8, R_9, R_{10}, \dots, R_{12}\}$ by adding R_9.

steps, adjacent region groups with approximated repetitive patterns should be clustered into one component group. We first describe how a region group matches a repetitive pattern and later describe the clustering procedure.

Region group and repetitive pattern matching. Given a region group $\mathcal{R} = \{R_1, R_2, \dots, R_n\}$ and a repetitive pattern $\mathcal{P} = \{c_1, c_2, \dots, c_m\}$, every region $\mathcal{R}_i$ is mapped to a center $c_k \in \mathcal{P}$ that is closest to the center of R_i (Fig. 9). After mapping, the $\mathcal{R}$ is subdivided into a set of subgroups according to their mapped centers. We compute the maximum subgroup whose average center equals to its mapped center (just like the subgroup R_1, R_2 in Fig. 9) and translate it to all the other centers of pattern $\mathcal{P}$ forming a new region group $\mathcal{R}'$, which is the estimated region group matched with pattern $\mathcal{P}$. In Fig. 9 the missing region R_9 is added into $\mathcal{R}'$. We use $miss$ and $redundancy$ to measure the degree of equivalency of $\mathcal{R}$ and $\mathcal{P}$:

$$miss = \frac{|\overline{\mathcal{R}} \cap \mathcal{R}'|}{|\mathcal{R}'|} \quad redundancy = \frac{|\mathcal{R} \cap \overline{\mathcal{R}'}|}{|\mathcal{R}'|}$$

In our experiments, if $miss$ and $redundancy$ are all below 0.4, we consider $\mathcal{P}$ and the repetitive pattern of $\mathcal{R}$ are approximately equivalent. So the $miss$ is 0.0833 and $redundancy$ is 0 in Fig. 9.

Component group clustering. To cluster the region groups into component groups, we first rebuild the graph G_{adj}. In each iteration, the most confident repetitive pattern is chosen as the pattern to cluster region groups to form a component group in which every repetitive pattern center corresponds to a component. The region groups in the equivalence class of the pattern are first clustered to the component group and then used as seeds to iteratively grow to cluster adjacent region groups in a breadth-first manner. If an adjacent region group is approximately equivalent to the pattern and not separated by the separating segments, it is clustered and the redundant regions are left out to form a new region group and a new pattern. Until no adjacent region groups can be merged, the confidences of repetitive patterns (whose region groups are merged) should be updated and the repetitive patterns with no region groups being left out should be deleted. After that, the remaining repetitive pattern with the maximum confidence starts the next iteration. The algorithm framework is shown in Fig. 8. As shown in Fig. 3(d), the region groups have been clustered into component groups. Obviously, the occluded parts, such as window corners, can be completed according to other components of the same group. From the above discussions, we know that the separating segments should be drawn at the places where the patterns of adjacent component groups are (approximately) compatible, which implies they can be merged. The output of this step is the drawing segmented into component groups.

6 Shape Tree Derivation and Component Recognition

This stage is to organize the already-segmented components hierarchically into a shape tree and recognize the component groups based on the shape tree. In Section 6.1, in order to derive a semantic shape tree, we split the drawing horizontally and vertically in a top-down manner to structure the architectural components hierarchically into a shape tree. This process will also aid the tasks of component recognition and shape grammar derivation. The component placement is interdependent, e.g., the roof is on top of columns and windows are between columns. Instead of recognizing every component independently, in Section 6.2, we propose a method to recognize components hierarchically by minimizing an energy on MRF, where the dependent relations are derived from the shape tree.

6.1 Shape Tree Derivation

After the bottom-up clustering into components, we split the façade into components in a top-down manner in this step. Some small, low repetitive, oblique or non-rectangle components, such as the oblique hip, are first filtered out. We split the façade recursively by alternating between vertical and horizontal directions until we arrive at single component. In each splitting, the components are sorted by their lengths along the splitting direction in the decreasing order and they are clustered into subdivided groups. Two components are clustered together if the location in the splitting direction of one component is covered by another one prior to it (i.e., longer than it). This process organizes the components into a structured shape tree, facilitating the component recognition and rules derivation. Fig. 10 shows four subdivision steps of the façade in Fig. 2 by vertical splitting and horizontal splitting and the derived shape tree is shown in Fig. 11.

6.2 Component Recognition

The components should be recognized simultaneously rather than individually to satisfy the structural constraints of Chinese architectures. The interdependencies are represented in a MRF $\mathcal{G}(\mathcal{V}, \mathcal{E})$ whose vertices are component groups to be recognized. A simple method

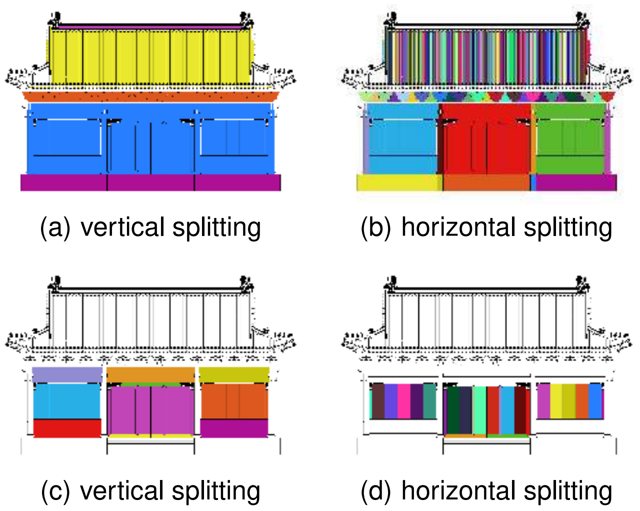

Fig. 10. (a-d) are the subdivisions corresponding to four levels of the shape tree respectively, in which every subdivided part is painted in different colors.

to form the graph is to connect adjacent component groups in the drawing, but the graph with cycles is too complicated to be solved exactly. Alternatively, to avoid cycles, we develop a heuristic method to decompose the graph $\mathcal{G}$ into tree-structured graphs $\mathcal{G}^{(n)}(\mathcal{V}^{(n)}, \mathcal{E}^{(n)})$ (Section 6.2.1) corresponding to each level n, $\mathcal{T}^{(n)}$, of the shape tree $\mathcal{T}$ and recognize the component groups (Section 6.2.2) iteratively at each level in a top-down manner.

6.2.1 The $\mathcal{G}^{(n)}$ Construction

To construct $\mathcal{G}^{(n)}$, for every node $v_i \in \mathcal{T}^{(n)}$, we have to choose a representative component group from its subtree to be a vertex of $\mathcal{G}^{(n)}$ corresponding to v_i. The algorithm is shown in Fig. 13. If a component group in the subtree has been recognized in previous levels, the recognized one is assigned to the vertex. Otherwise, we compute the component groups corresponding to the least-depth leaves in the subtree and denote them by $shallow(v_i)$. For every component group in $shallow(v_i)$, we count its repeated times that is the number of nodes $v_i \in \mathcal{T}^{(n)}$ whose subtrees contain some components of the component group and the most repeated ones are selected. Finally, the largest one of the most repeated component groups is assigned to be the component group corresponding to v_i. This method tends to choose the longest along the splitting direction, the most repeated and the largest (i.e., the most correlated) component groups.

The immediate siblings of $\mathcal{T}$ are adjacent to each other in the drawing, so they are interdependent. In $\mathcal{G}^{(n)}$ the vertices corresponding to immediate siblings of $\mathcal{T}$ are connected. Next, the vertices representing the same component group are merged together. Every edge is associated with a property of top-down or left-right to indicate the relative location of the connected component groups. Since $\mathcal{T}^{(1)}$ corresponds to the first vertical splitting of the architecture, two special vertices, Top and Base, are added to $\mathcal{G}^{(1)}$ as constraints shown in the first sub-figure of Fig. 12.

Consider the shape tree in Fig. 11 and its corresponding graphs in Fig. 12 as an example. In Fig. 11, the representative component groups for non-leaf nodes are shown in purple. In $\mathcal{T}^{(1)}$, the column is selected since it is the only least-depth component. In $\mathcal{T}^{(2)}$, the lintel is selected since it is the most repeated one in the least-depth nodes. In Fig. 12, the blue vertices are known component groups already recognized in previous levels. In $\mathcal{T}^{(3)}$, the chains are merged since they have lintel nodes in common. The graphs are all acyclic and the most correlated component groups are recognized simultaneously. This hierarchical recognition decomposes the simultaneous recognition into different levels of recognition and links different levels by the known vertices.

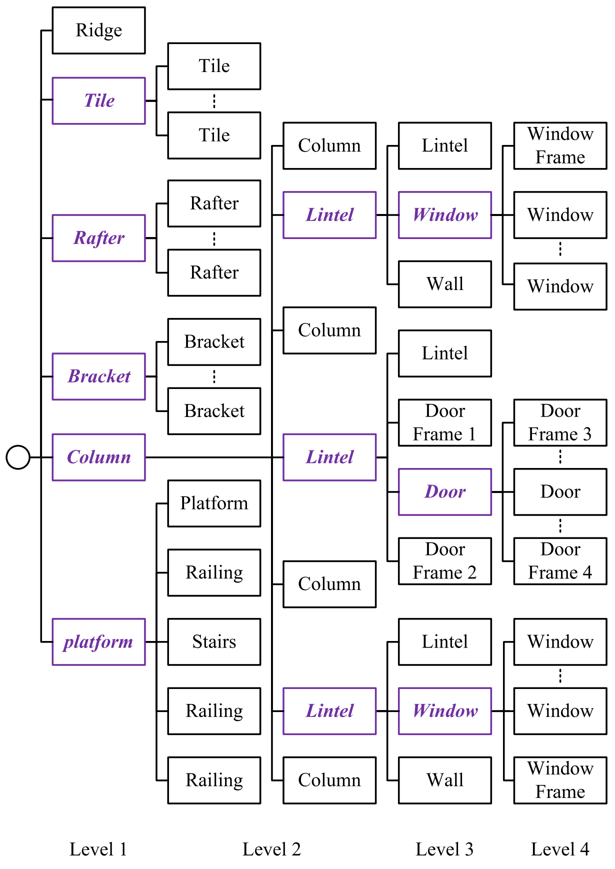

Fig. 11. The shape tree of the façade in Fig. 2. The purple nodes are non-leaf nodes and the italic purple component groups, which are chosen from their subtrees, are recognized in the levels where they are residing.

6.2.2 Energy Minimization

To label component groups, where the possible labels are column, door, window, wall, bracket set and tile, etc. as shown in Fig. 1, we define the following energy on

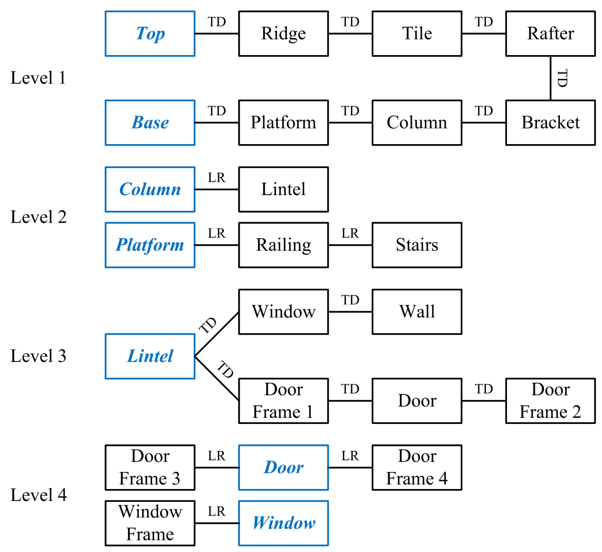

Fig. 12. Some graphs correspond to different levels of the shape tree. The blue vertices with italic font are known component groups and the black vertices are known vertices. The TD and LR denote top-down and left-right location relationships, respectively.

$\mathcal{G}^{(n)}(\mathcal{V}^{(n)}, \mathcal{E}^{(n)})$,

$$E(L) = \sum_{i \in \mathcal{V}^{(n)}} E_{sim}(l_i) + \sum_{(i,j) \in \mathcal{E}^{(n)}} E_{coh}(l_i, l_j),$$

where l_i is the label of the i-th component and $E_{sim}(l_i)$, $E_{coh}(l_i, l_j)$ are similarity energy and coherence energy, respectively.

$E_{sim}(l_i)$ describes the shape similarity of the i-th component to its labeled type l_i, and $E_{sim}(l_i)$ is evaluated based on the features of aspect ratio of bounding box, horizontal duty cycle, repeated frequency, etc. We have predefined the potential feature values or feature value scopes for every component type. For example, the ratio of width to height of lintel is relatively high while the column is relatively low. In the horizontal direction, the duty cycle of tiles is high, while the column is low. The repeated frequency of tiles is high while the lintel is not high. Given a component, the similarity energy to a certain component type is simply evaluated from the consistency between the feature values of the component and the component type.

$E_{coh}(l_i, l_j)$ describes the location coherence of adjacent pair of component groups with labels l_i and l_j indicating the interdependencies, e.g., the bracket set should be on top of the column and the lintel should be between the column. The wall always hints that it is window rather than door above it. $E_{coh}(l_i, l_j)$ is evaluated based on the relative positions of adjacent pairs of component groups (i.e., top-down or left-right).

The labels $L = \{l_1, l_2, \ldots, l_n\}$ are solved by minimizing the energy $E(L)$. If $\mathcal{G}^{(n)}$ is a single chain, the minimization can be solved with dynamic programming. However, due to vertex merging, $\mathcal{G}^{(n)}$ is sometimes a

```
1: initialize the repeated counter of every component group to zero
2: for all $v_i \in \mathcal{T}^{(n)}$ do
3:     $shallow(v_i) \leftarrow$ the component groups corresponding to the leaves of least-depth in the subtree of $v_i$
4:     the repeated counters of all the component groups in the subtree of $v_i$ increase one
5: end for
6: for all $v_i \in \mathcal{T}^{(n)}$ do
7:     if a component group in the subtree of $v_i$ is recognized then
8:         insert the recognized one to $\mathcal{G}^{(n)}$
9:     else
10:        find the most repeated component groups in $shallow(v_i)$
11:        insert the largest one of the most repeated component groups to $\mathcal{G}^{(n)}$
12:    end if
13: end for
14: connect vertices of $\mathcal{G}^{(n)}$ corresponding to immediate siblings in $\mathcal{T}$
15: merge common vertices in $\mathcal{G}^{(n)}$
```

Fig. 13. The algorithm of constructing the graph $\mathcal{G}^{(n)}$.

tree-structured graph. The exact minimization of $E(L)$ is solved using belief propagation, which is a powerful tool in computer vision (please refer to [30] [48] for details about belief propagation). During minimization, the labels of known vertices are clamped to not only infer the unknowns but also break down possible cycles.

Due to different heights, the tiles may not be combined into one component group as shown in Fig. 3(d). After recognition, the tile components are expanded horizontally by merging the adjacent shorter tiles to form the complete roof, which is pivotal to recognize the roof style and extract roof parameters.

7 Rule Derivation and Model Generation

In this step, the shape grammar rules are derived from the semantic shape tree using CGA shape grammar [2] with prior knowledge of Chinese architectures. 3D models are then generated from the rules with the help of a 3D component repository. The components in the repository should be parameterized to control their dimensions to match with the drawing dimensions. Most components can be simply parameterized by their widths and heights. However, the simple parameters are not appropriate for the complicated roof. We shall present the parameters for the roof generation and the building depth determination.

7.1 Roof Parameters and Building Depth

The roof structure is discussed in Section 3 whose parameters are a bit complicated. The roof width and height are

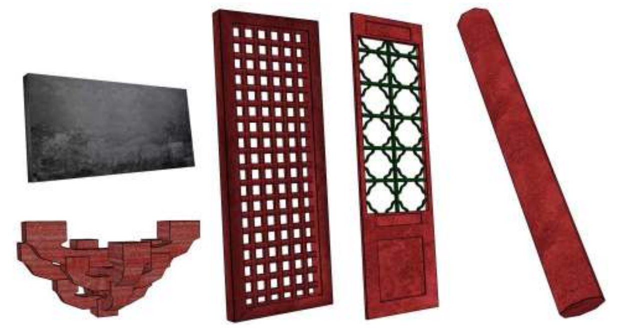
Fig. 14. A subset of 3D component models in the component repository.

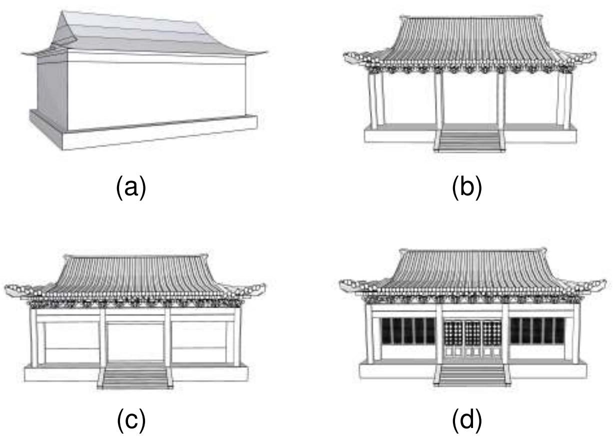

Fig. 15. The diagrammatic sketch of the model generation process. (a-d) correspond to the four levels of the shape tree, respectively, by splitting the façade while alternating between vertical and horizontal directions.

read from the drawing. The user specifies the parameters of *chuyan* and *jujia* indicating the slopes of the rafters. To infer the roof depth, the *bujia* can then be solved from the parameters of roof height, *chuyan* and *jujia*. The roof depth is $(purlin - 1) \times bujia$, where $purlin$ is the number of purlins. The roof can be constructed from these parameters.

The elevation drawing describing façade details lacks of depth information which is critical to the 3D shape of buildings. The width differences between different building levels (i.e., the nodes in the 1st level of the shape tree) equal to their depth differences. So the depths of different building levels, which are concentric, are inferred based on their width differences from the roof level. For buildings with corridor, the depth of corridor is set to *bujia*.

7.2 Shape Grammar Rule Derivation

In this step, we encode the semantic shape tree as shape grammar with prior knowledge of Chinese architectures and construct 3D models from the rules. Since the drawings provide little information about component details, as shown in Fig. 14, the buildings are generated based on a component repository which is a collection of 3D models of architectural components stored in different categories, including window, door, column, bracket set, etc. The final 3D models are comprised of the components from the repository.

We encode the building with the CGA shape grammar [2]. The rules (of splitting the building into hierarchical scopes) encode the building semantically in a top-down manner with subdivision split and repeat split rules corresponding to every subdivision of the shape tree. When a scope is split into just one component, it is replaced with its 3D instance in the component repository, otherwise, it continues the splitting process.

We consider a 4-level shape tree in Fig. 11 as an example to explain the rules encoded by subdivision and repeat rules. The rules are encoded corresponding to the four levels of subdivision detailed as follows: (1) With the depth data inferred as above, the rules first construct 3D mass models consisting of platform, building room, bracket set, rafter, roof and ridge from bottom to top using subdivision split. Since we assume the corridor exists, the building room is divided into two concentric volumetric shapes representing the outer column and the inner wall shapes, respectively (Fig. 15(a)); (2) In order to encode façade details, by making use of component split, the 3D mass models are split into 2D faces. The faces of roof, rafter and bracket set are then split and replaced with 3D models of tiles, rafters, and bracket sets, respectively, by the repeat split rules to fill as many elements as what the space could afford. The bays are the basic building blocks, which are bilateral symmetric and repeated horizontally. Except the center and side ones being fixed, the inner wall face is split into alternately repeated bays and columns (to be adaptive to any façade width). If just three bays are in the drawing, the ones on the side are repeated. Such a method ensures the bays are bilateral symmetric for any dimension. The outer columns are imposed to correspond to the inner ones. The stairs are arranged in the middle of the platform (Fig. 15(b)); (3) The bays are subdivided vertically into lintels, walls, doorframes and scopes of windows and doors (Fig. 15(c)); (4) Finally, the window and door scopes are subdivided horizontally and replaced with window, door, and door frame models, respectively (Fig. 15(d)). After arranging the façade details, the side and back faces are split into rafters and bracket sets similar to the façade and walls under the bracket sets, because the side and back faces of most Chinese architectures are simple walls. Alternatively, they can also be split into windows, doors, or even corridor similar to the façade. The textured model is shown in Fig. 18(a).

8 Results and Applications

A two-story building as input is shown in Fig. 16(a), in which the orange segments are sketched out to prevent over-aggressive merging. During the bottom-up cluster procedure, the regions are clustered to form meaningful

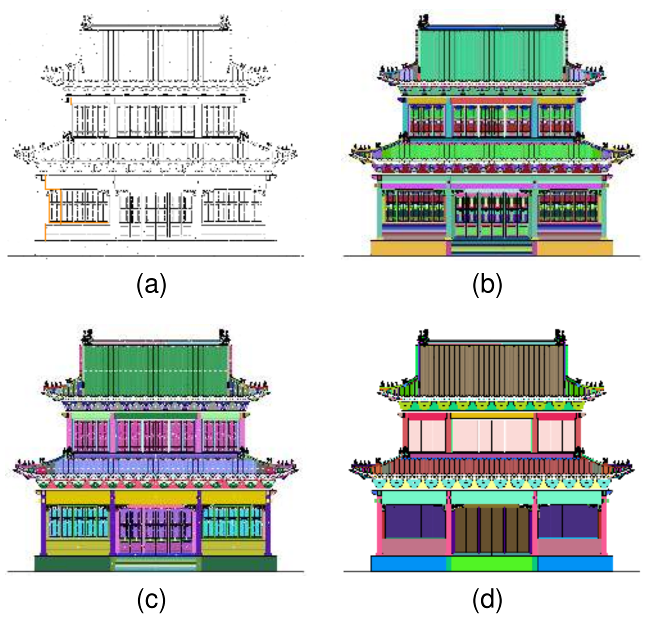

Fig. 16. Segmentation of a two-story architecture. (a) The original drawing with orange-sketched separating segments. (b) The region groups. (c) The patterns formed by different clusters of region groups. (d) The component groups further expand from the patterns.

components in order to segment the drawing. After noise handling and region filling, the regions are clustered according to their heights and shapes to form region groups as shown in Fig. 16(b). Then we compute their RPTs and the compatible region groups are clustered to form repetitive patterns as shown in Fig. 16(c), where the white circles indicate repetitive pattern centers. Finally, the regions are clustered based on the patterns to form components as shown in Fig. 16(d).

Next, during the top-down subdivision procedure, the entire building is subdivided hierarchically to form a shape tree. Some of the subdivisions are shown in Fig 17. The components recognized at the first level from bottom to top are platform, column, bracket set, rafter, roof, ridge, column, bracket set, rafter, roof and ridge. In the second level, the second floor is subdivided. The lintels are recognized while being constrained by the columns already recognized at the first level. And at the third level, the windows are recognized while being constrained by the lintels that were recognized at the second level. The other components are recognized analogously. After the recognition process, the shape grammar rules are derived from the semantic shape tree. The generated building with corridor is shown in Fig. 18(b). The component models are retrieved from the 3D component repository.

Figure 19(a) shows a 5-bay building with gable roof. The clustered component groups are shown in Fig. 19(b). The generated building is shown in Fig. 18(c). Figure 20(a) shows another building where the doors are not centered in the façade and the clustered component

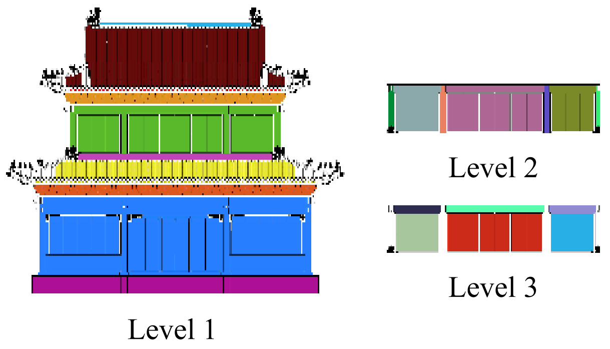

Fig. 17. The first subdivision of the façade, and the second and third subdivisions of the second floor.

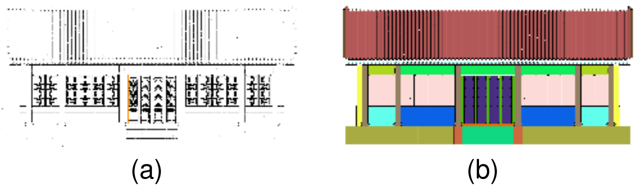

Fig. 19. A 5-bay building with the gable roof.

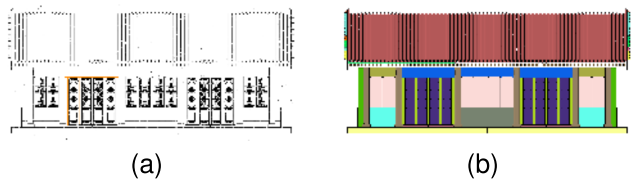

Fig. 20. A building in which the doors are not centered.

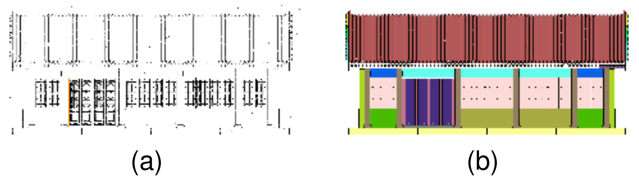

Fig. 21. A non-bilateral-symmetric building.

groups are shown in Fig. 20(b). The generated building is shown in Fig. 18(d). Figure 21 shows a non-bilateral-symmetric building. Even though such a building is conflicting with the convention of Chinese architectures, our algorithm can still proceed well, demonstrating its robustness. The generated building is shown in Fig. 18(e).

The rules are the semantic codes of the building, which drive the building formation and its variation with changing parameters. While the building is growing wider, the tiles, bracket sets and bays increase adaptively to fill the façade controlled by the repeat split rules as shown in Fig. 22, corresponding to the models in Fig. 18. Based on the shape rules, we construct a scene consisting of 3 courtyards with various widths of buildings as shown in Fig. 23, where the courtyard walls are built manually.

Limitations: Our framework presented in this paper

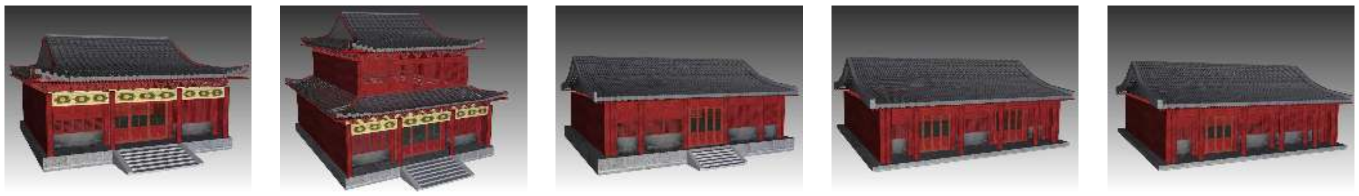

Fig. 18. The generated building models. (a) The one-story building. (b) The two-story building. (c) The gable-roof building. (d) The building without a centered entrance. (e) The non-bilateral-symmetric building.

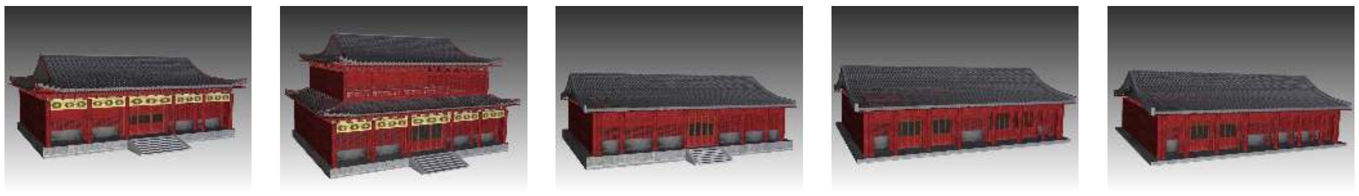

Fig. 22. The deformed building models correspond to the buildings in Fig. 18.

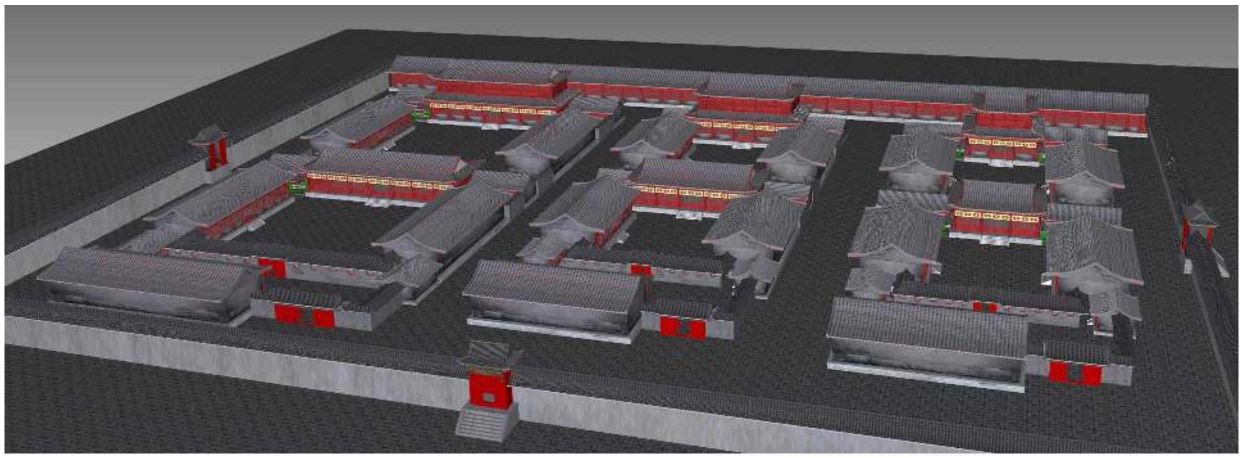

Fig. 23. A scene is built according to the derived rules, consisting of 3 courtyards with various widths of buildings, where the courtyard walls are built manually.

still needs user interaction during model generation. At present, we only concentrate on elevation drawing without paying attention to the sectional drawings or floor plans. The rule derivation is based on priors which may also confine the applicable scope. The noise in the drawing may result in corrupted regions. As shown in Fig. 24(a), in the narrow regions, outliers may expand and connect to the edges during dilation resulting in over-segmented regions. The parallel edges of narrow regions may even be connected during dilation if they are too close to each other. In Fig. 24(b), the gap in the red circle is failed to be closed, since it is at the terminal of the edge of the inner black pixels, which are unable to be stretched during the dilation and thinning procedure. If the number of corrupted regions is large, the correct repetitive patterns might be suppressed by the incorrect ones. As a result, this may engender incorrect segmentation (Fig. 25). In principle, the top-down subdivision aims to split the drawing along vertical and horizontal directions, so it is difficult to handle the non-rectangle components, such as the oblique hip. The building styles in this paper are somewhat limited. But, we believe that the façades of rotational symmetric buildings, such as pagoda with six or eight faces or the circular Heaven Temple, can be flattened to a plane under isometric mapping and our technology can be easily adapted to the 3D creation of these buildings. We only focus on horizontal repetitions in this paper, since Chinese architectures are only horizontally repeated. It is also applicably to vertically-repeated components (e.g., modern buildings) if we detect the region groups, repetitive patterns, and component groups vertically. The current technology is only applicable to 1D repetitions. It is not yet capable of analyzing 2D repetitions. During the model generation process, some flat components (e.g. doors) are flattened to arrange them easily.

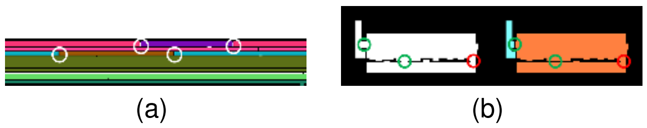

Fig. 24. (a) The narrow regions are over-segmented due to noisy points. The white circles mark the over-segmented edges. (b) The red-circled gap is failed to be closed and the green-circled gaps are closed well.

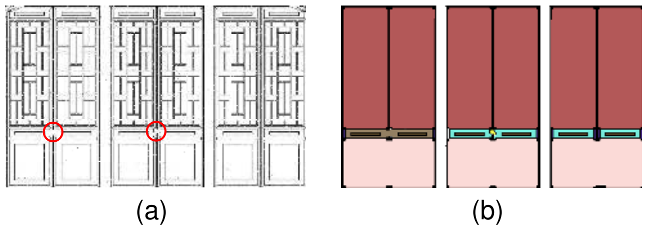

Fig. 25. The doors in (a) are improperly segmented because the gaps (red circles) are too large in the middle part resulting in doors being cut off as shown in (b) where each color represents a segmented component group.

9 Conclusion

We have developed a novel framework to extract rules and construct semantic models for Chinese architectures from elevation drawing. Our framework can generate rule-driven Chinese architectures with shape and dimension variations. The key idea is the integration of the bottom-up method for drawing segmentation and the top-down approach for shape tree construction. We design a bottom-up procedure to cluster regions into meaningful components based on their repetitive patterns encoded in the RPTs. This segmentation process is augmented by a top-down subdivision to organize components into a hierarchical shape tree. The components are then recognized by using MRF constructed by a heuristic method based on the shape tree. The shape grammar rules are derived from the semantic shape tree and 3D models are generated from the rules based on the component repository.

We envision that the elevation drawings can be integrated with other types of drawings to build not only the exterior but also the interior of 3D architectural models. The shape grammar rules can potentially facilitate semantic-driven model manipulation. It is possible to construct, deform, edit and manipulate models based on semantic information. In order to support the large-scale complex scene production and its real-time applications, different LOD models should be generated automatically with the help of the semantic shape tree structure. Our ongoing and near-future research tasks are concentrated on these topics.

Acknowledgments

We would like to thank all the anonymous reviewers for their valuable comments. We thank Wei Wang, Ning Wang and Sijia Yu for developing the rules compiler, and constructing the component repository and the scene. This work has been supported by the National Natural Science Foundation of China (Grant No. 60773153 and 61073078), the National High-Tech Research & Development Program of China (Grant No. 2009AA012102), Beijing Natural Science Foundation (Grant No. 4102037). Hong Qin's research reported in this paper has been conducted in Stony Brook University. This paper has been supported in part by US NSF grants: IIS-1047715, IIS-1049448, IIS-0949467, and IIS-0710819. Yue Qi (Beihang University) was a research scholar at Stony Brook University in 2010, when his research work reported in this paper was conducted.

References

[1] P. Wonka, M. Wimmer, F. Sillion, and W. Ribarsky, "Instant architecture," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 669–677, 2003.

[2] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool, "Procedural modeling of buildings," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 614–623, 2006.

[3] X. Yin, P. Wonka, and A. Razdan, "Generating 3d building models from architectural drawings: A survey," *IEEE Comput. Graph. Appl.*, vol. 29, no. 1, pp. 20–30, 2009.

[4] P. E. Debevec, C. J. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach," in *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1996, pp. 11–20.

[5] B. M. Oh, M. Chen, J. Dorsey, and F. Durand, "Image-based modeling and photo editing," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2001, pp. 433–442.

[6] A. R. Dick, P. H. S. Torr, and R. Cipolla, "Modelling and interpretation of architecture from several images," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 111–134, 2004.

[7] J. Xiao, T. Fang, P. Tan, P. Zhao, E. Ofek, and L. Quan, "Image-based façade modeling," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 1–10, 2008.

[8] J. Xiao, T. Fang, P. Zhao, M. Lhuillier, and L. Quan, "Image-based street-side city modeling," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–12, 2009.

[9] S. N. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys, "Interactive 3d architectural modeling from unordered photo collections," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 1–10, 2008.

[10] N. Jiang, P. Tan, and L.-F. Cheong, "Symmetric architecture modeling with a single image," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–8, 2009.

[11] B. Watson, P. Müller, O. Veryovka, A. Fuller, P. Wonka, and C. Sexton, "Procedural urban modeling in practice," *IEEE Comput. Graph. Appl.*, vol. 28, no. 3, pp. 18–26, 2008.

[12] C. A. Vanegas, D. G. Aliaga, P. Wonka, P. Müller, P. Waddell, and B. Watson, "Modelling the appearance and behaviour of urban spaces," *Computer Graphics Forum*, vol. 29, pp. 25–42, 2010.

[13] G. N. Stiny, *Pictorial and formal aspects of shape and shape grammars*. Basel: Birkhauser Verlag, 1975.

[14] M. Lipp, P. Wonka, and M. Wimmer, "Interactive visual editing of grammars for procedural architecture," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–10, 2008.

[15] D. G. Aliaga, P. A. Rosen, and D. R. Bekins, "Style grammars for interactive visualization of architecture," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 4, pp. 786–797, 2007.

[16] P. Müller, G. Zeng, P. Wonka, and L. Van Gool, "Image-based procedural modeling of façades," *ACM Trans. Graph.*, vol. 26, no. 3, p. 85, 2007.

[17] E. Whiting, J. Ochsendorf, and F. Durand, "Procedural modeling of structurally-sound masonry buildings," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–9, 2009.

[18] X. Hilaire and K. Tombre, "Robust and accurate vectorization of line drawings," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 6, pp. 890–904, 2006.
[19] L. A. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 6, pp. 910–918, 1988.
[20] C. Ah-Soon and K. Tombre, "Architectural symbol recognition using a network of constraints," *Pattern Recogn. Lett.*, vol. 22, no. 2, pp. 231–248, 2001.
[21] S. Yang, "Symbol recognition via statistical integration of pixel-level constraint histograms: A new descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 2, pp. 278–281, 2005.
[22] R. Lewis and C. Séuin, "Generation of 3d building models from 2d architectural plans," *Computer-Aided Design*, vol. 30, no. 10, pp. 765 – 779, 1998.
[23] T. Lu, C. L. Tai, F. Su, and S. Cai, "A new recognition model for electronic architectural drawings." *Computer-Aided Design*, vol. 37, no. 10, pp. 1053–1069, 2005.
[24] P. Dosch, K. Tombre, C. Ah-Soon, and G. Masini, "A complete system for the analysis of architectural drawings," *IJDAR*, vol. 3, no. 2, pp. 102–116, 2000.
[25] X. Chen, S. B. Kang, Y.-Q. Xu, J. Dorsey, and H.-Y. Shum, "Sketching reality: Realistic interpretation of architectural designs," *ACM Trans. Graph.*, vol. 27, no. 2, pp. 1–15, 2008.
[26] A. Berg, F. Grabler, and J. Malik, "Parsing images of architectural scenes," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, Oct. 2007, pp. 1–8.
[27] J. Pearl, "Reverend bayes on inference engines: A distributed hierarchical approach," in *Proceedings of the American Association of Artificial Intelligence National Conference on AI*, Pittsburgh, PA, 1982, pp. 133–136.
[28] S. L. Lauritzen and D. J. Spiegelhalter, "Local computations with probabilities on graphical structures and their application to expert systems," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 50, pp. 157–224, 1988.
[29] J. Mooij and H. Kappen, "Sufficient conditions for convergence of the sumcproduct algorithm," *Information Theory, IEEE Transactions on*, vol. 53, no. 12, pp. 4422–4437, Dec. 2007.
[30] J. S. Yedidia, W. T. Freeman, and Y. Weiss, *Understanding belief propagation and its generalizations*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 239–269.
[31] Y. Weiss and W. Freeman, "On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs," *Information Theory, IEEE Transactions on*, vol. 47, no. 2, pp. 736–744, Feb 2001.
[32] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, 2001.
[33] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, 2004.
[34] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael, "Learning low-level vision," *Int. J. Comput. Vision*, vol. 40, no. 1, pp. 25–47, 2000.
[35] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image completion with structure propagation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 861–868, 2005.
[36] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser, "A planar-reflective symmetry transform for 3d shapes," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 549–559, 2006.
[37] N. J. Mitra, L. J. Guibas, and M. Pauly, "Partial and approximate symmetry detection for 3d geometry," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 560–568, 2006.
[38] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas, "Discovering structural regularity in 3d geometry," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–11, 2008.
[39] M. Bokeloh, A. Berner, M. Wand, H.-P. Seidel, and A. Schilling, "Symmetry detection using feature lines," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 697–706, 2009.
[40] K. Xu, H. Zhang, A. Tagliasacchi, L. Liu, G. Li, M. Meng, and Y. Xiong, "Partial intrinsic reflectional symmetry of 3d shapes," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–10, 2009.
[41] Y. Liu, R. T. Collins, and Y. Tsin, "A computational model for periodic pattern perception based on frieze and wallpaper groups," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 3, pp. 354–371, 2004.
[42] G. Loy and J.-O. Eklundh, "Detecting symmetry and symmetric constellations of features," in *ECCV (2)*, 2006, pp. 508–521.
[43] H. Cornelius and G. Loy, "Detecting bilateral symmetry in perspective," in *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, ser. CVPRW '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 191–191.
[44] S. Liang, *Qing Shi Ying Zao Ze Li (in Chinese)*. Tsinghua University Press, 2006, republication of the 1934 edition.
[45] ——, *A Pictorial History of Chinese Architecture: A Study of the Development of Its Structural System and the Evolution of Its Types*, W. Fairbank, Ed. MIT Press, 1984.
[46] ——, *Qing Gong Bu "Gong Cheng Zuo Fa Ze Li" Tu Jie (in Chinese)*. Tsinghua University Press, 2006.
[47] A. I.-k. Li, "A shape grammar for teaching the architectural style of the yingzao fashi," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.
[48] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

Fei Hou received his B.E. degree in computer science from Beijing Institute of Technology in 2004. He is currently a Ph.D. candidate in the State Key Laboratory of Virtual Reality Technology and Systems at Beihang University. His research interests include procedural modeling, image-based modeling and geometric modeling.

Yue Qi is a professor in the State Key Laboratory of Virtual Reality Technology and Systems at Beihang University, China. He received the BS degree and PhD degree in system engineering from National University of Defense Technology in 1991 and 2001 respectively, the MS degree in computer science from University of Science and Technology of China in 1995. His research interests include virtual reality, augmented reality and computer graphics, with an emphasis on geometry and appearance modeling. He is a member of the IEEE and the IEEE Computer Society.

Hong Qin is a full professor of Computer Science in Department of Computer Science at State University of New York at Stony Brook (Stony Brook University). He received his B.S. degree and his M.S. degree in Computer Science from Peking University, China. He received his Ph.D. (1995) degree in Computer Science from the University of Toronto. During his years at the University of Toronto (UofT), he received UofT Open Doctoral Fellowship. He was also a recipient of NSF CAREER Award from the National Science Foundation (NSF), Honda Initiation Award, and Alfred P. Sloan Research Fellow by the Sloan Foundation. Currently, he serves as an associate editor for The Visual Computer, Graphical Models, and Journal of Computer Science and Technology. His research interests include geometric and solid modeling, graphics, physics-based modeling and simulation, computer aided geometric design, human-computer interaction, visualization, and scientific computing. Detailed information about Dr. Hong Qin can be found from his website: http://www.cs.sunysb.edu/∼qin. He can be reached at qin@cs.sunysb.edu.