Symmetrization of Facade Layouts

Haiyong Jiang[a], Dong-Ming Yan[a,*], Weiming Dong[a], Fuzhang Wu[a], Liangliang Nan[b], Xiaopeng Zhang[a]

[a] *NLPR-LIAMA, Institute of Automation, Chinese Academy of Sciences, Beijing, China*
[b] *Visual Computing Center, King Abdullah University of Science & Technology, Thuwal, Saudi Arabia*

Abstract

We present an automatic approach for symmetrizing urban facade layouts. Our method can generate a symmetric layout through minimally modifying the original input layout. Based on the principles of symmetry in urban design, we formulate facade layout symmetrization as an optimization problem. Our method further enhances the regularity of the final layout by redistributing and aligning elements in the layout. We demonstrate that the proposed solution can effectively generate symmetric facade layouts.

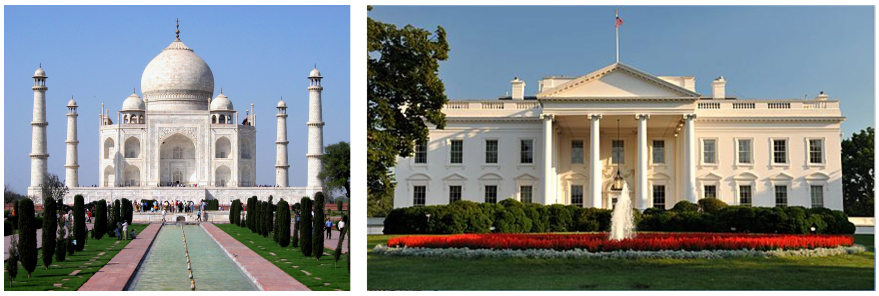
Keywords: Facade layout, Symmetrization, Energy minimization

1. Introduction

Symmetry refers to a sense of harmonious and beautiful proportion and balance [1]. It is ubiquitous in nature, science, and arts. In this work, we are interested in the symmetry exhibited in urban facade design.

In consideration of physical balance, aesthetic attractiveness, and construction costs, facade designs usually make extensive use of symmetry. These traits are presented in many kinds of architectures across different cultures and time periods (Fig. 1 shows two such examples). In computer graphics, symmetry has become a key ingredient for facade modeling [2, 3, 4], high level structural analysis [5], and manipulation [6]. However, when digitizing existing symmetric facades (e.g., extraction of facade layouts), errors are unavoidably introduced during the digitization process. Thus, it required to restore the symmetry faithfully to better represent the underlining layouts. By doing so, the visual quality of the facades' appearance could be improved, and the underlining description of the facades could also be greatly simplified. Other applications, such as changing an existing facade design (i.e., generating a new symmetric design from the original asymmetric design), generating facade variations from an existing layout,

*Corresponding author
Email addresses: yandongming@gmail.com (Dong-Ming Yan), weiming.dong@ia.ac.cn (Weiming Dong), liangliang.nan@gmail.com (Liangliang Nan), Xiaopeng.Zhang@ia.ac.cn (Xiaopeng Zhang)



(a) (b)

Figure 1: Two examples of symmetric facade design. (a) Taj Mahal; (b) White House.

etc., could also benefit from symmetrization. In this paper, a layout of a facade refers to a two-dimensional arrangement of the facade's elements (e.g., windows, doors and ornaments).

Extracting and symmetrizing facade layouts is a challenging problem. Directly manipulating a facade image is difficult due to the lack of a reliable automatic, accurate, and robust facade segmentation tool. In our work, we are not interested in extracting layouts from facade images. Instead, we are more interested in the symmetrization problem itself. Therefore, we simplify the data acquisition process by using the labeled rectangles on the facade images as input to our system. These rectangles can be seen as the initial layouts of the facades. In literature, quite a few techniques have been proposed to address symmetry detection [7, 8, 9], and symmetrization [10] of geometric models. These works are based on the observation that sufficient samples can be easily obtained to recover the symmetry of the models. Unfortunately, existing approaches neither are applicable, nor can be extended for symmetrization of urban facade layouts due to the sparsity of facades' content. Symmetry is simple in terms of transformation types (e.g., reflectional, translational, rotational, helical, scale, and the combination of the above). However, the complexity of relationships between facade elements makes this problem non-trivial. To the best of our knowledge, there is no work addressing the problem of automatic symmetrization of digitized facade layouts.

We present a method for automatic symmetrization of irregular facade layouts. Since there is no closed-form solution for the symmetrization problem. We search for a symmetric layout by minimizing the designed objective function. Specifically, we employ the simulated annealing algorithm to get an approximate optimal solution. Then, we improve the visual appearance of the facade layout by redistribution and alignment operations in order to make the layout more regular. Our results demonstrate our algorithm is effective in symmetrizing facade layouts.

In summary, the main contributions of this paper are as follows:

- An intuitive approach for symmetric structure detection in facade layouts.
- A new optimization framework for facade layout symmetrization based on an effective objective function and several editing operators.

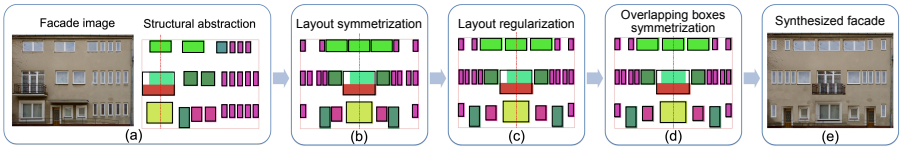


Figure 2: Overview of our framework. The boxes with the identical label are marked with the same color, and the black boundary box is used to identify a $unit$.

- A novel method to improve the regularity of the layout based on redistribution and alignment of facade elements.

2. Related Work

We briefly review the most related approaches to our work. The reader is referred to a recent survey [11] for more details on urban modeling.

Facade modeling.. Grammars are widely used in facade modeling, e.g., CGA shape grammars [12]. In specific cases, the grammars can be extracted from a facade image [13]. Structural analysis is another type of the facade layout modeling [2, 5, 14]. These approaches concentrate on the high-level semantic information. The split operation [15] and layering operation [5] are chosen to generate the hierarchical segmentation of the existing facade layout. During this procedure, some consensuses of the facade, such as the symmetry and regularity, are used to formulate the constraints to guide the segmentation and synthesis [2]. However, our work aims at generating new symmetric and regular facade layouts from the existing irregular inputs.

Layout design.. Layout design can be formulated as an optimization problem with specific criteria. For example, furniture layout design is resolved by optimizing the cost function with the functional and visual constraints [16]. The prior knowledge of facade layout design can be specified as cost terms [16, 17], or guided by the user's interaction [18]. The potential structural relations of the existing layout can also be inferred by the Bayesian network from the real-world data [19]. Bao et al. [18] divide the constraints into hard and soft ones. Ma et. al. [20] use configuration spaces and a graph-decomposition based layout strategy to steer the solution toward feasible layouts. However, the layout constraints usually involves non-differentiable variables. Thus stochastic optimization is used to generate a pleasant layout. Jiang et. al. [21] propose to detect constraints in the layout by an integer programming approach. The Metropolis criteria is preferred for its simplicity and effectiveness [19]. So we apply *Simulated Annealing* (SA) to our optimization problem.

Symmetry analysis.. Symmetry and regularity of object layouts can make the shape captured by human more easily. Any symmetry in a design reduces the amount of information necessary to specify shapes, thus it minimizes the entropy of the object layout [22]. A compact representation of the Euclidean symmetry can be extracted by

matching local shape signatures and clustering [7], while the symmetrization of the geometric objects can be achieved by an optimization process [10]. In our work, we only consider the reflectional symmetry of a set of boxes, which is a discrete optimization problem. In our scope, this is not involved in the previous works.

3. Overview

3.1. Definitions

Structural abstraction. Given a facade image $\mathbf{I}$, we can abstract the elements $\{e_1, \cdots, e_n\}$ in the facade layout $\mathbf{L}$ by a set of bounding boxes $\{b_1, \cdots, b_n\}$ with labels $\{l_1 \cdots l_n\}$. Here, we have $b_i = \{x_i, y_i, w_i, h_i\}$ with (x_i, y_i) as the bottom left corner and (w_i, h_i) as the size.

Element cells. In a facade image, elements with the fixed relation should be seen as an entirety. For example the window and its ledge are designed to be together, thus overlapping in most cases. In our paper, we consider elements with a fixed relation if they are overlapping, which will be gathered as a *cell*. A *cell* g_i is represented by the same information as an element, namely its bounding box $b_i^{'}$ and its label $l_i^{'}$. Assuming $g_i = \{e_1, \cdots, e_m\}$, we calculate $b^{'}$ as the bounding box of $\{b_1, \cdots, b_m\}$, and we have $l_i^{'}$ as the hash value of $\{l_1, \cdots, l_m\}$ to identify different element cells. In our work, we achieve symmetry of the facade layout $\mathbf{L}$ by symmetrizing the $cells$. Every operation applied on a $cell$ is transferred to the elements in it. For a layout $\mathbf{L}$, we want to optimize the original layout $\mathbf{L}^o$ to obtain a symmetrized and regularized layout $\mathbf{L}^t$.

Symmetry. In our paper, symmetry is referred to the reflective symmetry. Therefore, we can define the symmetry relation of two cells c_i, c_j with respect to the axis $x = x_s$ as $x_i + x_j + \frac{w_i + w_j}{2} = 2 \cdot x_s$. Here, we assume $w_i = w_j$ if they have the same label, and x_s is the symmetry axis of the facade. To measure the goodness of symmetry, we define the symmetry ratio of two cells g_i, g_j as $\frac{abs(x_i + x_j + \frac{w_i + w_j}{2} - 2 \cdot x_s)}{w_i}$. The symmetry is only valid for cells with the same label.

3.2. Pipeline overview

There are three main steps in our system, i.e., layout symmetrization (Sec. 4), layout regularization (Sec. 5) and overlapping boxes symmetrization (Sec. 5.2.3). The pipeline of our system is shown in Fig. 2. We briefly describe each step as follows:

- **Layout symmetrization.** We first calculate the symmetry axis, and determine the corresponding symmetric region. Next, we detect the existing symmetry structure in the original layout. At last, we compute the symmetric layout via energy minimization.

- **Layout regularization.** In addition to symmetry, most architectures also exhibit apparent layout regularity [22]. Therefore, the redistribution and alignment operations are applied to make the layout more regular and visually pleasant. Moreover, these two operations should preserve the symmetry and overlapping

relationships produced in the previous step. The redistribution operation translates elements in order to achieve an approximately even boundary distance in the horizontal direction, while the alignment operation aligns the elements to different rows or columns according to their current positions.

Overlapping boxes symmetrization.. In the last step, we symmetrize the overlapping boxes in every cell to enhance the symmetry of the whole layout.

4. Layout Symmetrization

Given an extracted facade layout, our purpose is to symmetrize the layout while respecting the original layout. The following challenges make the problem non-trivial. First, a facade may be only partially symmetric (See Fig. 3 for an example), and symmetrizing the whole layout will cause great changes. Second, facade layouts follow complex overlapping relation, for example the balcony should be arranged together with a window. Third, rearranging elements in a layout is with combinational complexity.

This part is organized as follows. At first, we determine the symmetric region and the symmetric axis (see Sec. 4.1). Secondly, we explore the hard constraints that our result should obey (see Sec. 4.2). Then we detect the potential symmetric pairs (see Sec. 4.3). Finally, we propose our solution to this symmetrization problem (see Sec. 4.4).

4.1. Symmetry axis identification

We assume that the symmetry axis is vertical, which is reasonable for almost all architectural facades. We firstly find the dominant symmetric axis of the layout, which is evaluated by the position with the maximal symmetry, and the symmetry score can be calculated by integral symmetry [5]. Then we use this symmetric axis and the bounding box to decide the region r_s that supports the symmetric axis as shown in Fig. 3. The other region is defined as $sidebar$.

We define the symmetric region and symmetric axis as follow:

- If there is no identically labelled elements in the $sidebar$ with elements in region r_s, then we treat the $sidebar$ as an independent region from the symmetric region. Simply symmetrizing the layout will break the $sidebar$, therefore we choose to copy them to achieve the global reflectional symmetry. Fig. 3 (top) shows an example of the latter case.

- Otherwise, we merge the $sidebar$ and region r_s as the symmetric region. In other words, the symmetric region is the whole input layout. See Fig. 3 (bottom) for an example.

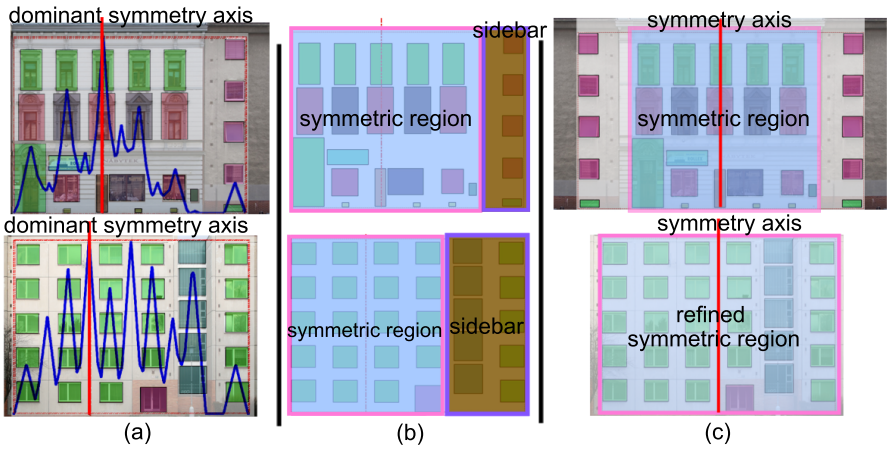


Figure 3: Illustration of the symmetry axis detection and the $sidebar$ complement. (a) structural abstraction and symmetry measure [5], (b) the detected symmetric region and the *sidebar*, (c) the final symmetric region after *sidebar* complement.

4.2. Hard constraints

Elements in the layout have very complex positional relations, among which the overlapping relation plays an important role. In our implementation, we group overlapping elements together to keep the relation, and impose the later optimization stage on the $cells$, so that their original relationships will not be wrecked. Furthermore, the overlapping constraints can be achieved by forbidding the overlap between cells. At the same time, every cell should lie in the symmetric region. Our optimization scheme and the rearranged layout ensure the above two assumptions.

4.3. Symmetric pairs detection

In most cases, only partial layout is asymmetric. Thus, focusing on the asymmetric elements will greatly reduce the computation cost and the complexity of the algorithm. Thus we need to firstly detect and symmetrize the existing or potentially symmetric pairs in the layout for the later usage. In our algorithm, we regard a pair of elements as symmetry if the symmetry ratio is greater than a user specified threshold, which is empirically set to 0.8 in our implementation. This can be used to detect the existing symmetric pair in the original layout. Besides, we further find more potentially symmetric pairs by using the Gale-Shapley algorithm [23]. We define the matching measure as:

$$match(i,j) = \begin{cases} abs(\dfrac{x_i + x_j + \frac{w_i + w_j}{2}}{2} - x_s), \text{if } SLR(i,j) \\ +\infty, \text{otherwise} \end{cases} \tag{1}$$

where i, j are the indices of the $cells$ and $SLR(i,j)$ is 1 if cell i, j with the same label and approximately in the same row with ϵ tolerance. Then we apply the matching measure to the stable matching algorithm. The output matching pairs are the potentially

symmetric pairs, because they can minimize the moving distance to make the pairs symmetric.

Afterwards, we enhance the symmetry of the layout by symmetrizing the existing and potentially symmetric pairs. This is reasonable, since the translation is the operation that has the minimum cost in the final energy formulation. Thus it has the same effect as the direct symmetrization but reduces the computational cost of the optimization. The symmetrization of two $cells$ (i, j) can be realized by moving each $cell$ with the distance $(x_s - \frac{x_i + x_j + \frac{w_i + w_j}{2}}{2})$ [10]. If the above step violates the original overlapping relationships, we discard the symmetrization modification of these pairs. These operations can decrease the burden of the following optimizations for other complicated cases.

4.4. Optimization

The layout symmetrization are reduced into an optimization problem [19]. The difficulty lies in the diversity of the new rearrangement. In this section, we present an effective formulation to tackle this problem. Note that the objective terms are calculated based on the elements, while the editing operations employ the $cell$ as a basic module.

4.4.1. Objectives for constrained optimization

We describe the cost terms that evaluate the quality of the proposed rearrangements with respect to the original layout. Our objectives are formulated based on the observation that a good rearrangement demands the minimal modifications of the original layout.

Asymmetry.. The most important term is the asymmetry term f_s. We use it to measure the asymmetry of the layout. The definition is given as follow:

$$f_s = \frac{\sum_{i \in S^t_{asym}} area(i)}{\sum_{i \in S^t} area(i)} \tag{2}$$

where $area(i)$ is the area of the i-th element, S^t_{asym} the index set of asymmetric elements in the rearranged layout, and S^t the index set of all elements in the rearranged layout.

Completeness.. In order to keep the diversity of element labels, we define the completeness f_c as the changes in the label set:

$$f_c = \frac{(N^o_l - N^t_l)}{N^o_l}, \tag{3}$$

where N^o_l denotes the number of labels in the original layout, N^t_l the number of labels in the target layout.

Label number.. Except for the changes in the label types, we also want to minimize the changes in the number of used labels. We define f_{lnc} to evaluate it.

$$f_{lnc} = \sum_{i \in S^o} \frac{\delta(l_i^o, l_i^t)}{\sum_{j \in S^o} \delta(l_j^o, l_i^o)} \tag{4}$$

where S^o is the index set of the labels in the original layout, and l^o, l^t represents the label in original and target layout, respectively. The function $\delta(\cdot, \cdot)$ equals 1 if two values are same, otherwise 0.

Box area percentage.. Since we want the changes of the architectural area to be minimal, we define the related objective term f_{lcba} as follows.

$$f_{lcba} = \frac{abs(\sum_{i \in S^o} area(i) - \sum_{j \in S^t} area(j))}{\sum_{i \in S^o} area(i)} \tag{5}$$

where the symbols have the same meaning as above.

Relative position.. The relative position term f_{rp} measures the transitive distances between the corresponding elements in the original layout and the proposed rearrangement. This value cannot be calculated directly, because the elements in the layout are disturbed or removed during the optimization. Thus, we use the stable matching algorithm [23] to find the corresponding pairs between the original layout and the rearranged layout. For elements in the same row with the same label, we use their distance as the matching score, otherwise they are regarded as not matching. Then we can use the stable matching pairs to calculate f_{rp}:

$$f_{rp} = \sum_{\substack{i \in S^o, j \in S^t \\ (i,j) \in M_{stable}}} \frac{abs(x_i^o - x_j^t)}{w_b}, \tag{6}$$

where S^o, S^t have the same meaning as above, M_{stable} is the map of indices from the original layout to the rearranged layout, and w_b is the width of the symmetric region. In the results, we show the effects of these terms.

Cost Function.. The cost function $C(x)$ is a combination of the above objective terms.

$$C(x) = w_s \cdot f_s + w_c \cdot f_c + w_{lnc} \cdot f_{lnc} + w_{lcba} \cdot f_{lcba} + w_{rp} \cdot f_{rp} \tag{7}$$

where $w_s, w_c, w_{lnc}, w_{lcba}$ *and* w_{rp} specify the tradeoffs between different terms.

4.4.2. Editing Operators

To effectively explore the space of the symmetric layout, our operators are designated to augment the symmetry of the current layout. We find the following operators are efficient in the generation of candidate layouts. As the architecture has obvious storey information, the translation of cells should be confined in the horizontal direction.

Move cells. In our algorithm, we translate the cell position by the mixture Gaussian distribution,

$$p(x_i) = r_1 \cdot N(x_j - x_i, (x_j - x_i)^2) + r_2 \cdot N(x_{j'} - x_i, (x_{j'} - x_i)^2) \tag{8}$$

where $j, j^{'}$ are the two nearest cells to the cell i in the left and right sides in the original layout. The other items are defined as $r_1 = abs(\frac{x_j - x_i}{x_j - x_{j'}})$, $r_2 = abs(\frac{x_j - x_i}{x_j - x_{j'}})$. The cells are translated with this distribution for the aim of minimal movements.

In each iteration, we randomly translate each cell with a probability that is determined by the total distance cost of elements of the specified cell. For the purpose of the maximal symmetry, we move the symmetric cells in pairs in order to keep their symmetry. Thus this operation can be applied in each iteration free of the disarrangement of the symmetric pairs.

Swap cells. In order to rapidly explore the layout space, we introduce the swapping operation to interchange a cell from a position to another in the same row. We do not directly swap two cells, for the small probability of successful permutations, which is caused by the different sizes of cells. For the same purpose, we push the cells to left and right respectively according to their positions relative to the swapping position, therefore more spaces for the swapping cell can be made. To maximize the symmetry, we choose the swapping position from the Multinomial distribution with equal probability $\frac{1}{n}$. The corresponding random variables are constituted by the position of the symmetry axis, positions and mirror positions relative to the symmetry axis of cells, which are in the same row and have the same label as the swapping cell.

The probability of the swapping operation is determined by the f_s in Sec. 4.4.1. In each iteration, we only choose a cell in the asymmetric set for swapping with a probability proportional to its area.

Remove or add an asymmetric cell. Considering about the complexity of the layout, simple movement and swapping may fail to symmetrize the layout. In this case, we need to change the elements in the original layout, thus we add the removal and adding operation for the asymmetric cells.

We use this operation with a probability determined by the ratio of asymmetric cells. In each removal operation, we only randomly select an asymmetric cell with the probability which is inversely proportional to its area. In opposite to the removal operation, we add the reflectional cell of an asymmetric cell to make the corresponding cells symmetric.

Add a removed cell. To make the operation complete, we have to be capable of adding the removal cell. When we delete a cell, we record it for the possible adding. To make the rearranged layout easier to accept in the optimization procedure, we add cell as symmetric pairs. Cells can also be added to the symmetry axis or replace cells in the symmetry axis.

The frequency of this operation is based on the percentage of the removed boxes in the original layout. In this operation, the adding cell is selected from the removal cell set in proportion to its area.

Remove a symmetric pair. With the above operations, we can achieve the symmetrization. However, the results incline to increase the number of elements, and there is no reverse operation that can decrease it. Therefore we add this operation for a balance.

The probability of this operation is set according to f_{lnc}, while the selection of a symmetric cell is proportional to the number changes of its corresponding labels.

4.4.3. Symmetrization

Since the space of the rearranged layout is discrete and combinatorial, the optimal solution must be searched in the global space. However, the combinatorial exploration of the solution space makes the pursue of the optimum impossible. The technique of the constraint satisfaction programming enumeration algorithm or branch and bound [24] can not work in this case. As a strict global optimum is not necessary for our problem, we opt to a stochastic search method to seek an approximate optimum.

In our approach, we employ the simulated annealing algorithm to solve this problem, because of the efficiency of Metropolis criteria [19]. Our objective function is defined in Eq. 7, and the editing operators proposed in Sec. 4.4.2 are used to generate a new layout. Acceptance probability of a new layout is defined as

$$a(x^{\star}|x) = \min\left(1, \exp\left(-\frac{C(x^{\star}) - C(x)}{T_n}\right)\right), \tag{9}$$

where $C(x)$ is the objective function, and T_n the annealing temperature. We schedule the temperature with $T_n = \alpha T_{n-1}$, where $T_0 = 150, \alpha = 0.995$ are selected by experience. Our algorithm stops when both the expected temperature and the symmetry of the layout are satisfied.

5. Layout Regularization

As the symmetrization procedure does not follow the regularity constraints of the layout, we redistribute cells and align elements to increase the regularity. During this process, the layout will still hold the symmetry constraints by only dealing with half part of the symmetric cells, and the other part is disposed by mirror movement.

5.1. Redistributing the space

In order to distribute cells with a proper distance, we equalize the space of nearby cells. This operation further increases the regularity of the layout. We achieve a uniform spacing by extending the cost function in *Boundary Voronoi Tesselation* (BVT) [25].

At first, We find the left and right neighbors, $N_l(i), N_r(i)$, of cell i. For a cell that close to the boundary of the layout, there only exists left or right neighbor. Then our target is to minimize the following cost:

$$E(S^t) = \sum_{i \in S^t} (d(i, N_l(i))^2 + d(N_r(i), i)^2) + \sum_{i \in S_{LB}} d_{2l}(i)^2 + \sum_{i \in S_{RB}} d_{2r}(i)^2, \tag{10}$$

where S^t is the index set of all cells, S_{LB}, S_{RB} represent the cells that are close to the left/right boundary of the layout, respectively. The function $d_{2l}(\cdot)$ and $d_{2r}(\cdot)$ compute

the distance of a cell to the left and right boundary, respectively. The distance function $d(\cdot,\cdot)$ is defined as:

$$d(i,j) = x_i + \frac{w_i}{2} - x_j + \frac{w_j}{2} - (w_i + w_j)/2, \tag{11}$$

where $x_i > x_j$ should hold.

Minimizing Eq.10 yields the optimal solution. Because Eq. 10 is quadratic, the minimum can be achieved by using the gradient descent method. Hence, we update the position of each cell with:

$$x_i^{'} = \begin{cases} 0.5 \cdot (x_j + x_l), \text{if } i \in S_{NLB} \\ 0.5 \cdot (x_j + x_r), \text{if} \in S_{NRB} \\ 0.5 \cdot (x_j + x_{j'}), \text{otherwise} \end{cases} \tag{12}$$

where j, j' are the indices of the cells that are nearest to cell i in the left and right sides, x_l, x_r stand for left and right boundary, respectively. We execute the above update for 80 iterations to get the solution.

The experiments show that our method distributes boxes with extent in the horizontal direction as shown in Fig. 4(c).

5.2. Center alignment

In this part, we align cells with their center x/y position to improve the regularity. The alignment method proposed in [26] inspired us, but it is not suitable for our problem. Hence, we design our own strategy for the alignment as described in the following parts.

5.2.1. Horizontal alignment

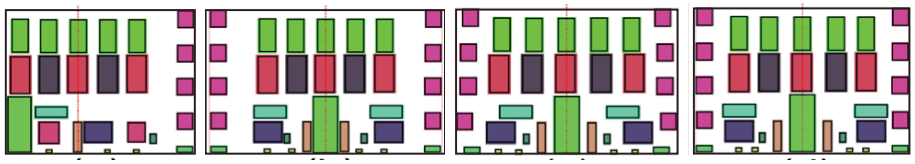
We separate the cells into clusters according to their vertical center with ϵ precision, then we align the cells in each cluster separately. This simple threshold way can achieve a good alignment in the horizontal direction because of obvious storey arrangement in the facade layouts.

5.2.2. Vertical alignment

Aligning cells in the vertical direction is a little complex, because it's hard to decide which cells should be aligned. Thus, we need to find the existing alignment cells in the layout, then align cells to these alignment positions.

Clustering cells.. A cell is more likely to align with the cell that is closer to it. We propose to use the greedy strategy to cluster cells. Initially, we calculate the distance matrix for any two cells. Then cells are clustered hierarchically. During this step, we need to check if the cluster step will cause the overlapping of cells, in which case we will discard this clustering pair to avoid conflicts in the later alignment.

Aligning cells in different clusters.. Once we have the aligning clusters, we compute the average of cell's centers weighted by their area as the vertical alignment position. Then cells are translated to the alignment position. As shown in Fig. 4(d), our method can attain the desirable alignment of cells.

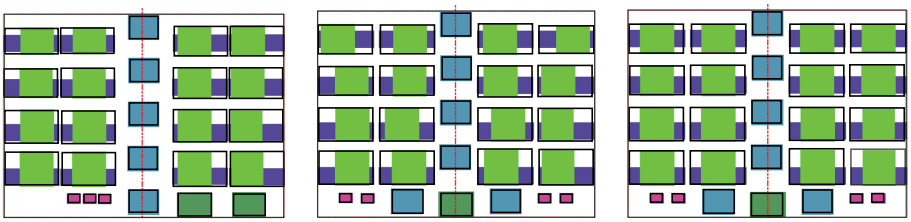


(a) (b) (c) (d)

Figure 4: The effects of redistribution and alignment method. (a) structural abstraction, (b) after symmetrization, (c) after redistribution, (d) after alignment.

5.2.3. Intra-cell symmetry enhancement

Every cell is seen as an entirety in the layout symmetrization and regularization. To improve the symmetry, we utilize the method proposed in Sec. 4.3 to symmetrize the overlapping elements in each cell. We use their bounding boxes as the symmetric region and the center of the bounding box as the symmetry axis. With this additional symmetry relationship, the symmetry of the whole set can be augmented. The effect of symmetrizing the overlapping boxes is shown in Fig. 5.



(a) (b) (c)

Figure 5: The effect of overlapping elements' symmetrization. (a) structural abstraction, (b) after layout symmetrization and regularization, (c) after intra-group symmetrization.

6. Results

Test data.. Our experimental results are evaluated on the publicly available dataset shared by [5], which contains facade images, as well as the corresponding labelled boxes as the structural abstraction. We also provide a marking tool for user to mark out this structural abstraction. In Fig. 6, we present some sampled results, which demonstrate the effectiveness of our algorithm.

Evaluation.. In our scope, automatically symmetrizing the facade layout is not involved. The grid structure can guarantee the regularity and symmetry at the same time [27]. However, this structure is too simple to represent the complex cases in the dataset. Zhang et. al. [5] propose an algorithm to extract a layer representation of various kinds of facade layouts. However, they only offer the symmetry results with

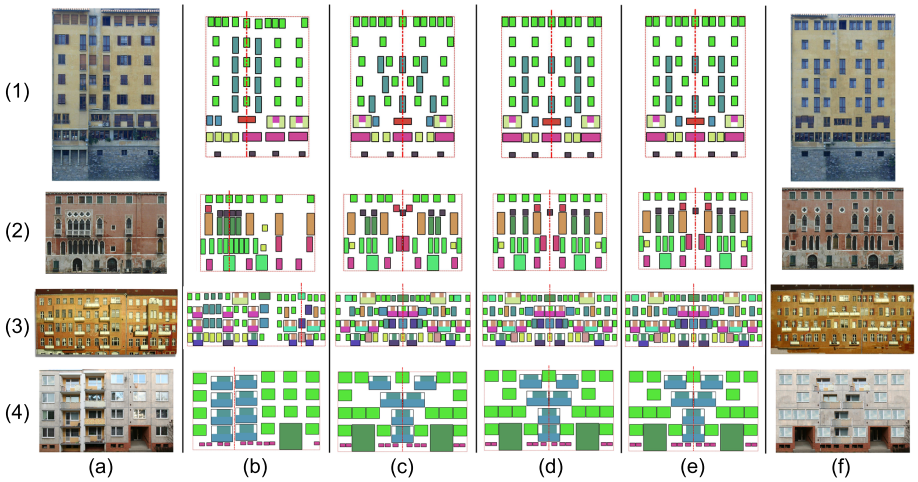


Figure 6: Populated examples from the final results. (a) the original facade image, (b) structural abstraction, (c) after layout symmetrization, (d) after layout regularization, (e) after intra-cell symmetrization, (f) the synthesized facade image.

the user interaction. Furthermore, the hierarchical structure always leads to too much constraints, thus easily failing to obtain a completely symmetry result.

Our system can ensure the symmetry of the cells in the final layout with all editing operators in Sec. 4.4.2. However, our arrangements can be customized for different user preferences. For example, some users may want to improve the symmetry but not to remove any boxes. Our algorithm is suitable for this kind of applications by simply forbidding the removal operations (See Fig. 7). Besides, it's easy for our algorithm to adjust the frequency of different operations to achieve different symmetric layouts (See Fig. 8). The statistics of sampled results are shown in Tab. 1. In our implementation, we make extensive use of the objective information to guide the operators' frequency.

Fig.	add	rm	add_rm	rm_sp	swap	energy
(a)	-	-	-	-	-	19375.3
(b)	0.05	0.80	0.10	0.1	0.1	2871.8
(c)	0.80	0.05	0.01	0.1	0.1	643.9
(d)	0.80	0.05	0.10	0.1	0.1	254.1
(e)	0.10	0.10	0.80	0.1	0.1	686.5
(f)	0.10	0.10	0.10	0.8	0.1	273.3
(g)	0.10	0.10	0.10	0.1	0.8	260.9

Table 1: Cost comparison for examples using varying frequency of different operations shown in Fig. 8.

The effects of different cost terms are demonstrated in Fig. 9. Note that we do not list the effect of excluding asymmetry because it is necessary for the layout symmetrization. The effect of f_{lcba} is not obvious as a result of the influence of f_c, f_{lnc} on the cost of the area percentage.

To accomplish symmetry of the final layout, w_s is set to a value much larger than

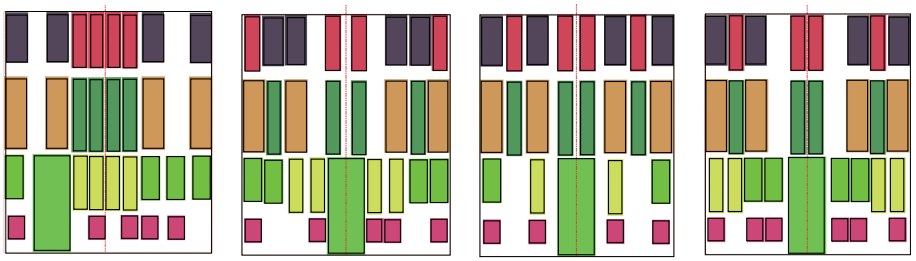


Figure 7: Effects of parts of the operations. (a) structural abstraction, (b) without removal operation, (c) without adding operation, (d) with all operations.

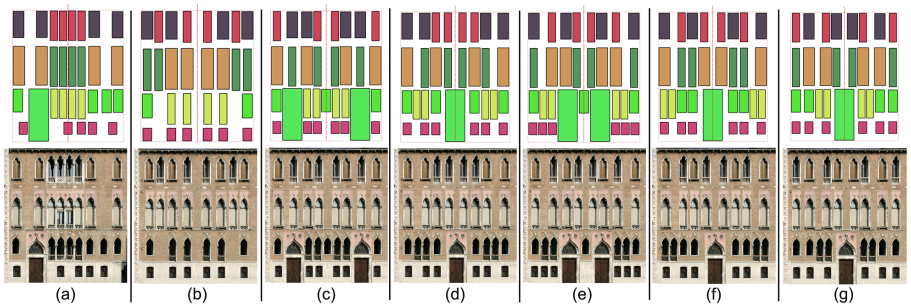


Figure 8: Illustration of the effect of different operations. (a) the original facade; (b∼g) results of using varying frequency of different operations listed in Table 1.

others. The remaining weights are assigned according to their relative importance, for example w_c should be larger than w_{lnc}. It is because the adding operation, which changes the number of elements holding a specified label, is preferred than the removal operation, which may reduce the diversity of the layout.

In our implementation, we select weights $w_s = 86000$, $w_c = 8600$, $w_{lnc} = 1500$, $w_{lcba} = 1000$, $w_{rp} = 700$ for Eq. 7 empirically. These weights have a great influence on the final layout (see Fig. 9). Especially, the relative value of f_c, f_{lnc} in Eq. 7 will directly impact on the acceptance of adding and removal operations.

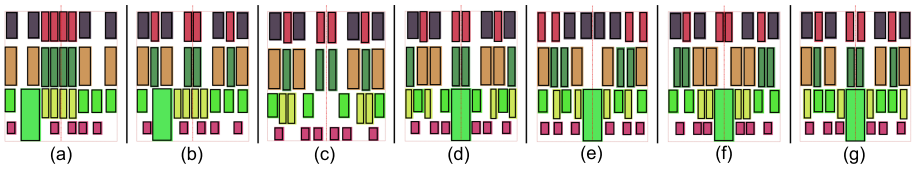
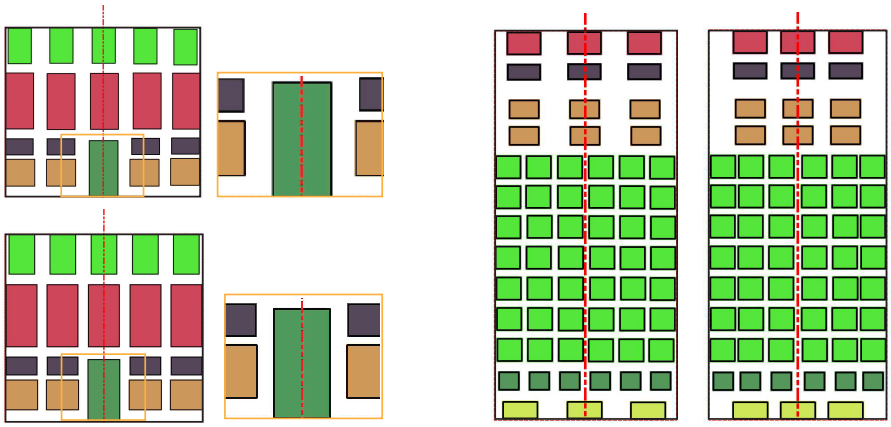


Figure 9: The effects of different objective terms. (a) structural abstraction, (b) the result after preprocess, (c) completeness excluded, (d) label number excluded, (e) relative position excluded, (f) box area excluded, (g) all terms included.



(a) (b)

Figure 10: The applications on the error correction of facade layouts. In (a), the top row is the input and bottom row the output. We zoom in the yellow region to highlights the difference. In (b), the left is the input and right is the output.

Application and extension.. Symmetry is an important attribute for the layout design. With our algorithm, we can achieve a symmetric facade layout, which will generate pleasing variations of a given facade layout (see Fig. 10 as an example). Our algorithm ensures global symmetry, but not restricted to. For users who only require partial symmetry, our system can easily satisfy this kind of demands by specifying the symmetric region (See Fig. 11). By weighting the cost terms, we can only also obtain different symmetric results.

Running time.. Our experiments were conducted on a PC with an Intel i7-3770 CPU and 16GB RAM. Our algorithm takes $2 \sim 3$ seconds for a moderate input facade layout with about 50 boxes, thus it can be easily applied to interactive applications. The detailed performance statistics for Fig. 6 are given in Table 2.

Fig.	asym.	comp.	#L	area.	dist.	#B	T(s)
(1)	0	0	0	0.01	0.10	55	2.49
(2)	0	0.11	0	0.02	0.14	40	1.72
(3)	0	0.06	0.11	0.03	0.12	82	7.70
(4)	0	0.00	0.25	0.02	0.08	42	2.90

Table 2: Performances and objective values in Eq. 7 of examples in Fig. 6.

Limitation and future work. Our work is a preliminary attempt for the layout regularization, and has several limitations. First, our algorithm only explores the overlapping relations, while being fully aware that other semantical relations should be learned so as to keep the architectural style. For example, the balcony and window are more likely to be center aligned and grouped together. Second, we do not inspect repetitions

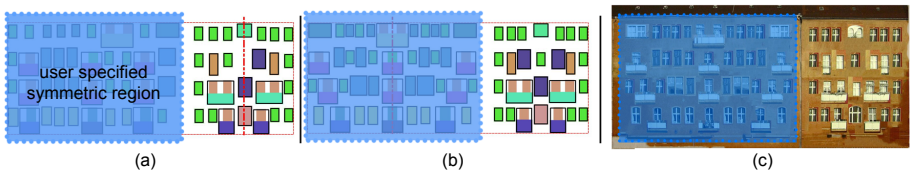


Figure 11: With user specified symmetric region, we can achieve partial symmetry. (a) structural abstraction and user specified symmetric region, (b) the partial symmetry result, (c) the synthesized facade image.

of elements in a facade layout, which are expected to be grouped as an entirety (see Fig.12 as an example). In our experiment, the appropriate scale of repetitions is hard to determine. At last, more efforts should be put on the texture synthesis in order to get more realistic results.

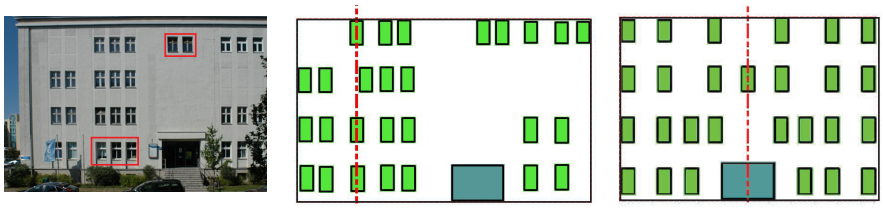


Figure 12: A failure case that does not maintain the original repetitive patterns. Left: input image, middle: input layout, right: output layout. In the left image, the red bounding box highlights some repetitive patterns, which is not preserved in the final result.

The most desirable pursuit would be to incorporate the semantic groups of architectural elements based on the statistical information. This make it possible to prevent the specific styles being wrecked. Investigations into the discovery of architectural styles are interesting aspects of facade analysis. Our algorithm is designed for facade layouts, but it can be easily generalized to the symmetrization problem of all kinds of layouts. In our work, we use the simulated annealing to optimize our problem. To make the algorithm more productive, we will prospect the Reversible Jump Markov Chain Monte Carlo algorithm in order to effectively search the candidate space.

7. Conclusions

We have presented a system for facade layout symmetrization. Our method optimizes structural abstraction extracted from facade images and emphasizes on the symmetry and minimal modifications of the original layouts. The regularity of the layout is also enhanced by convex optimization and clustering. We believe that this work will take a step towards developing new tools for high-level facade modeling.

8. Acknowledgements

We would like to thank Peter Wonka for many helpful discussions. This work was partially supported by the National Natural Science Foundation of China (Nos.

61331018, 61372168, 61272327 and 61572502), the China National 863 Program (No. 2015AA016402) and the KAUST Visual Computing Center.

References

[1] A. Zee, Fearful symmetry: the search for beauty in modern physics.

[2] F. Bao, M. Schwarz, P. Wonka, Procedural facade variations from a single layout, ACM Trans. on Graphics 32 (1) (2013) 8:1–8:13.

[3] D. Ceylan, N. J. Mitra, Y. Zheng, M. Pauly, Coupled structure-from-motion and 3d symmetry detection for urban facades, ACM Trans. on Graphics 33 (1) (2014) 2.

[4] P. Musialski, P. Wonka, M. Recheis, S. Maierhofer, W. Purgathofer, Symmetry-based façade repair., in: VMV, 2009, pp. 3–10.

[5] H. Zhang, K. Xu, W. Jiang, J. Lin, D. Cohen-Or, B. Chen, Layered analysis of irregular facades via symmetry maximization, ACM Trans. on Graphics (Proc. SIGGRAPH) 32 (4) (2013) 121:1–121:13.

[6] S. Al-Halawani, Y.-L. Yang, H. Liu, N. J. Mitra, Interactive facades analysis and synthesis of semi-regular facades, Computer Graphics Forum 32 (2pt2) (2013) 215–224.

[7] N. J. Mitra, L. J. Guibas, M. Pauly, Partial and approximate symmetry detection for 3D geometry, ACM Trans. on Graphics (Proc. SIGGRAPH) 25 (3) (2006) 560–568.

[8] N. J. Mitra, M. Pauly, M. Wand, D. Ceylan, Symmetry in 3d geometry: Extraction and applications, Computer Graphics Forum 32 (6) (2013) 1–23.

[9] K. Xu, H. Zhang, A. Tagliasacchi, L. Liu, G. Li, M. Meng, Y. Xiong, Partial intrinsic reflectional symmetry of 3d shapes, ACM Trans. on Graphics (Proc. SIGGRAPH) 28 (5) (2009) 138.

[10] N. J. Mitra, L. J. Guibas, M. Pauly, Symmetrization, ACM Trans. on Graphics (Proc. SIGGRAPH) 26 (3) (2007) 63:1–63:8.

[11] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Gool, W. Purgathofer, A survey of urban reconstruction 32 (6) (2013) 146–177.

[12] P. Müller, P. Wonka, S. Haegler, A. Ulmer, L. Van Gool, Procedural modeling of buildings, ACM Trans. on Graphics 25 (3) (2006) 614–623.

[13] P. Müller, G. Zeng, P. Wonka, L. Van Gool, Image-based procedural modeling of facades, ACM Trans. on Graphics 26 (3).

[14] F. Wu, D.-M. Yan, W. Dong, X. Zhang, P. Wonka, Inverse procedural modeling of facade layouts, ACM Trans. on Graphics (Proc. SIGGRAPH) 33 (4) (2014) 121:1–121:10.

[15] P. Wonka, M. Wimmer, F. Sillion, W. Ribarsky, Instant architecture, ACM Trans. on Graphics 22 (3) (2003) 669–677.

[16] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, V. Koltun, Interactive furniture layout using interior design guidelines, ACM Trans. on Graphics 30 (4) (2011) 87:1–87:10.

[17] Y.-L. Yang, Q.-X. Huang, Traygen: Arranging objects for exhibition and packaging, Computer Graphics Forum 32 (7) (2013) 187–195.

[18] F. Bao, D.-M. Yan, N. J. Mitra, P. Wonka, Generating and exploring good building layouts 32 (4) (2013) 122.

[19] P. Merrell, E. Schkufza, V. Koltun, Computer-generated residential building layouts, ACM Trans. on Graphics 29 (6) (2010) 181:1–181:12.

[20] C. Ma, N. Vining, S. Lefebvre, A. Sheffer, Game level layout from design specification, Computer Graphics Forum 33 (2) (2014) 95–104.

[21] H. Jiang, L. Nan, D.-M. Yan, W. Dong, X. Zhang, P. Wonka, Automatic constraint detection for 2d layout regularization, Visualization and Computer Graphics, IEEE Transactions on PP (99) (2015) 1–1.

[22] N. A. Salingaros, M. W. Mehaffy, A theory of architecture, UMBAU-VERLAG Harald Püschel, 2006.

[23] L. E. Dubins, D. A. Freedman, Machiavelli and the gale-shapley algorithm, The American Mathematical Monthly 88 (7) (1981) 485–494.

[24] J. Michalek, R. Choudhary, P. Papalambros, Architectural layout design optimization, Engineering optimization 34 (5) (2002) 461–484.

[25] B. Reinert, T. Ritschel, H.-P. Seidel, Interactive by-example design of artistic packing layouts, ACM Trans. on Graphics 31 (6).

[26] J. Lin, D. Cohen-Or, H. Zhang, C. Liang, A. Sharf, O. Deussen, B. Chen, Structure-preserving retargeting of irregular 3d architecture, ACM Trans. on Graphics 32 (4) (2013) 122.

[27] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, L. Guibas, Discovering structural regularity in 3D geometry, ACM Transactions on Graphics 27 (3) (2008) #43, 1–11.