EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

Volume 32 (2013), Number 2

Interactive Facades
Analysis and Synthesis of Semi-Regular Facades

Sawsan AlHalawani[†] Yong-Liang Yang[†] Han Liu[†] Niloy J. Mitra[⋆]

[†]KAUST [⋆]University College London

Abstract

Urban facades regularly contain interesting variations due to allowed deformations of repeated elements (e.g., windows in different open or close positions) posing challenges to state-of-the-art facade analysis algorithms. We propose a semi-automatic framework to recover both repetition patterns of the elements and their individual deformation parameters to produce a factored facade representation. Such a representation enables a range of applications including interactive facade images, improved multi-view stereo reconstruction, facade-level change detection, and novel image editing possibilities.

1. Introduction

Building facades are integral to most urban environments, both real-world and virtual. Such facades are often created with repeated elements (e.g., windows) placed in regular (e.g., along 2D grids) or semi-regular arrangements. However, facades get modified over time as window elements are opened or closed, secondary elements like curtains, flower plots are added, etc. Such geometric variations can lead to large appearance changes, often effectively obscuring the underlying simplicity of the facades.

In recent years, significant progress has been made towards simplified and accurate acquisition of facades, either using image-based methods or directly using 3D scanning setups (e.g., LiDAR scans). Such raw data can then be analyzed to reveal underlying repetition patterns facilitating compact storage, non-local data completion, or inverse procedural modeling (see recent surveys [MPWC12, MWA*12] and references therein). These methods, however, are designed to detect (near) identical repeated facade elements arranged on regular grids and can fail in scenes with structured variations like varying blind positions on windows or semi-regular grids (see Figure 1).

In this work, starting from any single facade image, we propose a semi-automatic framework to jointly extract element-level repetitions while factoring out allowed variations (e.g., opening or closing of windows) to obtain a decomposition that brings the facade elements to a canonical configuration essentially extracting a *factored representa-*

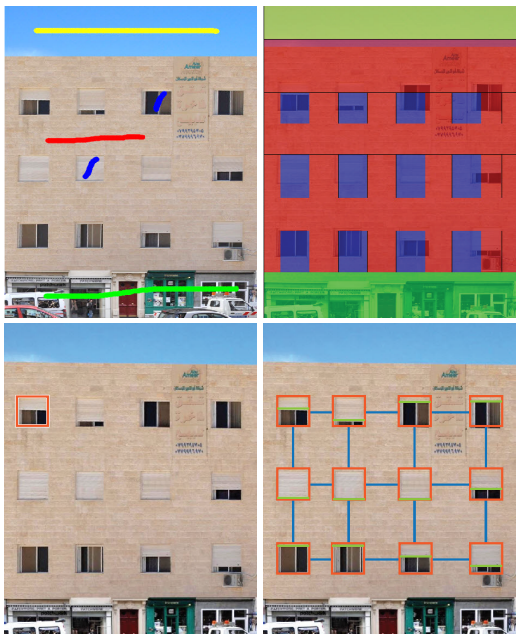

Figure 1: *Analysis of facade images is often complicated by variations of repeated elements (e.g., windows) from canonical configurations. For example (top), state-of-the-art facade analysis [TKS*11] can fail due to element variations, while our algorithm (bottom) correctly extracts both the underlying repetition pattern and the individual blind positions.*

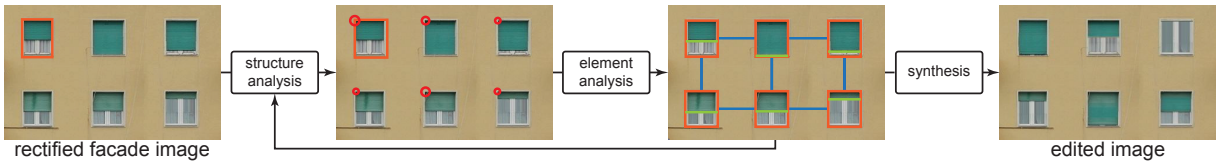

Figure 2: *Starting from a single rectified facade image with a single marked window (in orange), we solve for regularity maximizing repetition structure (red circles show local estimates of window locations before global coupling) while recovering the individual deformation parameters of the repeated window elements. The resultant factorization enables novel synthesis possibilities while preserving extracted facade-syntax.*

tion. Additionally, the decomposition provides valuable information about the allowed variation modes, which can then be used towards new editing and synthesis possibilities. Our key observation is that element segmentation, repetition (and structure) detection, and solving for the individual element configurations when performed jointly lead to increased accuracy and robustness. Although a full joint optimization is complex, in presence of simple user hints (e.g., marked rectangle in Figure 1-bottom-left) and simple priors about how windows can typically be manipulated (e.g., windows can slide up/down, window shutters can open outwards), we propose an iterative solution. Beyond new analysis possibilities leading to significantly compact facade encodings (typically 5-10x compression ratio), the factored representation (see Figure 10) reveals the functional workings of the facades and allows facade-syntax preserving manipulations leading to interactive facade images. (Note that the proposed analysis and synthesis have implicit cultural assumptions about window types and styles.)

We demonstrate our framework on a range of synthetic and real-world images with varying amount of element variations and clutter and use the extracted factored representations to make the input facades *interactive*. We demonstrate novel synthesis and manipulation possibilities, both on the input images and on synthesized geometries, towards the goal of meaningful facade manipulations (see supplementary video and interactive facade viewer demo).

In summary, our main contributions are: (i) introducing factored facade representation to specifically encode window-level low degree of freedom movements; (ii) extracting such encodings from input facade images; and (iii) using the encodings to enable a range of novel interaction to bring static (single) facade images to life.

2. Related Work

We focus only on the works most relevant to our problem, while referring the readers to the recent survey [VAW*10] for a detailed exposition on facade and urban modeling.

Image-based modeling. Debevec et al. [DTM96] in their seminal paper propose an interactive image-based modeling technique to exploit characteristics of architectural objects coupling an image-based stereo algorithm with manually specified 3D model constraints. Subsequently, structure-from-motion (SfM) has been used to recover urban facades from unorganized photo collections (see [SSS06, FCSS09] and references therein) using photogrammetric reconstruction and image-based modeling techniques. Such systems produce massive collections of low-level unorganized textured points, which are not suited for low-memory footprint navigation or mobile interactions (e.g., Google Streetview). Furthermore, the methods rely on image-space similarity, which are often lost as window element configurations are changed, or repetition patterns are semi-regular.

Procedural modeling. Wonka et al. [WWSR03] use split grammars and an attribute matching system to synthesize buildings with varying styles. Later, researchers have explored auto-correlation based analysis of rectified images combined with shape grammars towards urban reconstruction [MZWG07]. The approach proposes an interesting mix of user interaction and image analysis for rule-based procedural modeling. Since these methods only support local operations, recently Musialski et al. [MWW12] propose a coherence-based interactive system to support non-local coupling. The method, however, fails in presence of allowed variations across similar elements, as is our focus. Note that the extracted factored facade representations obtained using our system can directly be used to produce accurate 3D geometries, which can then be procedurally edited and refined.

3D geometry synthesis. Multiple data sources (e.g., photographs, LiDAR scans, aerial images, GIS data) have been combined to improve the quality of 3D models [FJZ05, LZS*11]. Alternately, working with incomplete LiDAR scans, Zheng et al. [ZSW*10] use model scale repetitions to create a consolidated point cloud. Although the resultant point clouds have high resolution, the algorithms rely on multiple sources, have moderate to high memory foot-prints, and are not directly suited for creating realistic model variations. Ceylan et al. [CML*12] propose a 2D-3D coupled optimization using symmetric line arrangements towards accurate 3D reconstructions. However, they heavily rely on the presence of near symmetry across repeated elements, an assumption commonly violated in our inputs.

Facade annotations. Our work is inspired by recent facade analysis efforts of Teboul et al. [TSKP10] who perform supervised learning using shape grammar priors to interpret

building facades using random walks on the learned models. The method has been extended using recursive binary split grammar and reinforcement learning [TKS*11] to parse facades into element level masks (e.g., windows, doors) using training information. Wu et al. [WWF*10] analyze the translational grid patterns in the images to enable image resizing by manipulating the semantic grid cells. None of these efforts, however, explicitly account for intrinsic variations across elements (see Figure 1). In a related attempt, Alsisan et al. [AM12] swap window elements to maximize image-space repetitions and hence reduce memory footprint towards abstraction and efficient transmission. They, however, do not handle allowable deformations and hence inherit the typical problems of other regularity detection approaches.

3. Overview

Our goal is to decouple repetitions (regular or semi-regular) from element-level variations in the context of facade images. Figure 2 gives an overview of our algorithm. Starting from single input facade images, in the analysis phase, we extract repeated facade elements (i.e., windows) along with their variations. While image-space variations among repeated elements arise from a range of factors (e.g., geometric variations of windows being open/closed; customization due to the addition of flower pots, blinds; appearance variations due to shadow, reflection, etc.), in this work we primarily focus on extracting the geometric variations, which are necessary to enable realistic synthesis. We make two key observations: (i) errors in the local analysis for repetition detection can be rectified using a non-local repetition maximizing optimization, and (ii) having simple deformation priors for 3D window elements leads to robust extraction of the corresponding deformation parameters. The user marks a single window element and our proposed framework extracts the repetition pattern, per window deformation parameters, and corresponding structured image completion results for the window elements (see Section 4). Subsequently, we use the extracted factored representation to enable facade-syntax preserving interaction and synthesis possibilities.

4. Facade Analysis

In this section, we describe the facade analysis phase wherein the goal is to detect regular and semi-regular pattern(s), extract the repeated elements, and per-element deformation parameters.

Preprocessing. We first rectify the input image using vanishing lines. On the rectified image, say I, we extract Canny edges and discard the small edges (using a fixed threshold of 2% of element size). Note that at this local-analysis stage, edges arise from window frames, room interiors, blind features, facade textures, etc.

4.1. Facade Structure Analysis

State-of-the-art facade regularity detection methods assume the repetition elements to be similar. However, in our target examples, image-level repetition is often lost due to per element variations (e.g., opening/closing of blinds). This results in various forms of errors in automatic regularity detection (see Figure 3). Instead, we allow the user to mark one single element of interest W_s (implicitly identifying scale of interest) while the analysis framework recovers a repetition maximizing regularity, as described next.

Figure 3: *State-of-the-art algorithms (e.g., [WFP11]) designed to detect facade regularities often fail due to varying configurations of the repeated elements: regular grids may get split into multiple grids; detected repetitions may have offset ambiguity; or only small partial grids may be detected. In our interactive framework, we expect the user to indicate one window, while our framework extracts the (semi-regular) repetition pattern for window locations.*

Repetition detection. We use the selected dominant window mask W_s as a reference to search and extract a large set of candidate repeated elements from the rectified facade image. We search using a combined intensity and edge consistency scores as line features are typical in building facades. Both terms, however, can lead to spurious entries: in case of shadow or different interior elements (e.g., flower pots, curtains), intensity similarity produces errors; while inconsistent edges and varying window element configurations lead to spurious edge similarity. We later recover from such errors using a non-locally coupled extraction (see Figure 4).

We compute the intensity score $w_{intensity}$ using the Normalized Cross Correlation (NCC) [BH01] as it is resilient to intensity variations due to lighting and shadow fluctuations.

We compute the edge score for a candidate patch W_t based on the matches with the edges from the user provided template W_s. We mark a pair of line segments (l_i, l_j) such that $l_i \in W_s, l_j \in W_t$ to be matching (i.e., match$(l_i, l_j) = 1$) if their directions agree within a $\pm 20^\circ$ margin AND the distance between the midpoints of the line segments is smaller than a threshold δ (10% of element size in our implementation). We define the total edge score as the sum of such matches:

$$w_{edge} := \sum_{l_i \in W_s, l_j \in W_t} \text{match}(l_i, l_j)/|W_s||W_t|. \qquad (1)$$

Note that since we ignore the small edges in the edge extraction step, we did not find it necessary to weigh the scores by the edge lengths, but only normalized by the edge counts. Finally, we define the combined score as: $w_t(W_s, W_t) :=$

Figure 4: *Extracted repetition grid in (rectified) facade images under varying regularity, occlusion, and difficult illumination conditions. In each example, the user prescribed one template rectangle W_s (top-left windows in each example). Subsequently, blind/shutter positions (not shown here) are extracted as explained in Section 4.2.*

$w_{intensity}(W_s, W_t) + \lambda w_{edge}(W_s, W_t)$, with $\lambda = 0.5$ in our tests. Note that while the user can adjust λ to assign different preference for the intensity versus edge terms, we kept the weights fixed in our tests.

Having defined an error metric to compare template W_s with image patches W_t, we find candidate matching image patches. As common on urban facades, we assume most windows to be horizontally or vertically aligned. We first sweep along a vertical line from the template and for each point on the line we extract the locations where W_s locally have maximal similarity (using a 3-5 pixel margin). Effectively, we extract the local maxima in the correlation profile along the horizontal direction. In the vertical sweep, we suppress similar solutions to keep only local maxima solutions (see also [MZWG07]). For robustness and efficiency, we use a 3-level Gaussian image pyramid for searching. Thus, we obtain a set of candidate 2D locations for the marked template W_s, each with a confidence score based on the similarity values. However, the candidate solutions can overlap and are not yet non-locally coupled (see Figure 5).

Repetition structure optimization. Due to occlusions and illumination variations, we detect wrong suggestions in the candidate locations. We use an optimization to prune such wrong ones and extract the (near-) regular structure of the facade (see [ACM12] for another use of repetition maximizing optimization). Essentially, we look for consistent splitting lines, although the resultant grid need not be fully occupied. Note that we allow irregularly-spaced partially-filled grid of window elements, unlike other methods that solve for full 2-parameter grids [PMW*08]. Due to the nature of the problem, we independently solve for the x and y splits. Note that we implicitly assume the windows to be semi-regularly arranged.

Let the set of candidate locations be denoted by $P := \{p_1, \ldots p_n\}$ with their associated confidence weights $W := \{w_1, \ldots w_n\}$. Note that these denote the candidate points projected to x or y direction (since we solve for each direction independently). Our goal is to select a subset of k splits that best explains the current observations, while maximizing regularity. Let, S be such a candidate set that contains k points, i.e., the original template position W_s and $k-1$ other points from P. A set of points is considered to be *valid* only if the stencils at the corresponding positions do not overlap. We use simulated annealing to optimize for a good split, i.e., we start with a random assignment S consisting of k points from P and keep the valid solution with the best fitting energy, which we define next.

In order to measure the fitness of a valid set, we compute its energy as a combination of a data term and a non-local

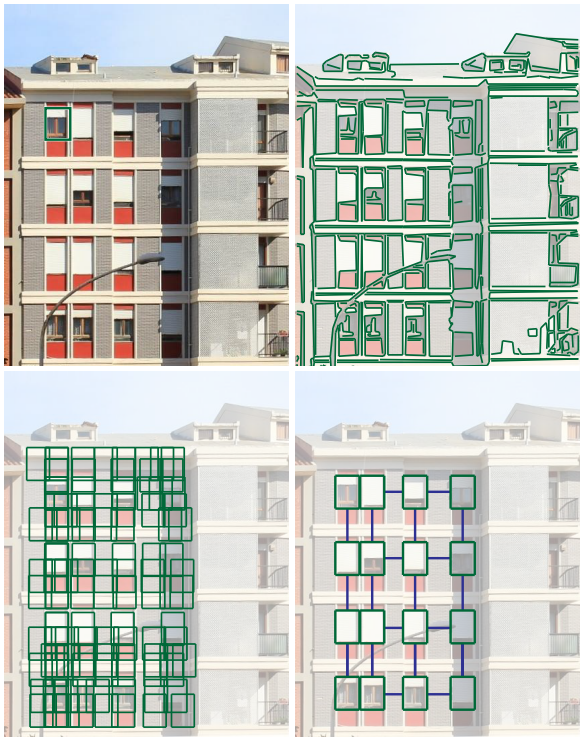

Figure 5: *(Top-left) Starting from an input (rectified) image and an element proxy W_s (shown in green), we extract image edges (top-right) and identify potential matches using a combined NCC and edge similarity score (bottom-left). Subsequently, we select a subset of matches in a repetition maximizing optimization (bottom-right). Note that we also handle semi-regularly arranged elements (see also Figure 6).*

regularity term as follows:

$$\min_S E(S) := E_{data}(S) + \lambda E_{reg}(S) \quad (2)$$

where λ determines the relative contributions among the data and regularity terms ($\lambda = 0.3$ in our tests). We now describe the two terms.

Data term (E_{data}): This term measures the confidence with which a set S covers the candidate set of points P as:

$$E_{data}(S) := \sum_{i=1}^{k} 1/(w_i \cdot g_i) \quad (3)$$

where, w_i denotes $w_s(q_i, W_s)$ and g_i measures how well the point $q_i \in S$ agrees with the other points $p_j \in P$ using a Gaussian falloff as:

$$g_i(S_i) := \sum_{p_j \in P} \exp(-\|q_i - p_j\|^2 / 2\sigma^2) \quad (4)$$

where σ represents the width or height of the template W_s depending on the dimension of the optimization.

Regularity term (E_{reg}): A typical facade has a near-uniform distribution of distances among its neighboring windows. Therefore, we introduce this term to measure the regularity of the points q_i among the set S. We find the spacing distribution among $q_i \in S$ as $\{d_1, d_2, d_3, ..., d_{k-1}\}$ after sorting them by x (or y) values. Then, we measure the regularity by computing the regularity of the selection as follows:

$$E_{reg}(S) := \sum_{i=1}^{k-1} (d_i - \mu)^2 + \sum_{i=1}^{k-2} (d_{i+1} - d_i)^2, \quad (5)$$

where μ denotes the average spacing. Note that the first term captures deviation from mean separation, while the second term penalizes variations in neighboring deviations.

In order to find the splitting lines that specify the grid containing the windows, we minimize the energy given in Equation 2 using a simulated annealing based sampling (SA) [KGV83]. We start with $E \leftarrow \infty$. Then, in the SA step, we accept the new solution with energy $E(S)$ if $E_{new} \leq E(S)$; else we accept the new solution with probability of $\exp(-(E_{new} - E(S))/t)$, where t is the temperature; else we reject the new layout and retain the old one. If we accept a solution, we set $E \leftarrow E_{new}$. In the annealing schedule, we reduce temperature t and continue with the iterations. We stop if either the maximum number of steps (50-100 in our tests) has been reached, or when $E < threshold$. (Note that, more advanced stochastic optimization strategies like reversible jump MCMC can also be used, but we did not find it necessary.) Starting from $k = n$ we solve for decreasing values of k and keep the first local minima for $E(S)/k$. This identifies a set $S^\star$ whose points specify the positions of the splitting lines, solved for x and y independently. We mark each such grid location as active if we have a $p_i \in P$ located nearby. Figures 4 and 6 show results under varying grid spacing, different occlusion, and challenging illumination conditions.

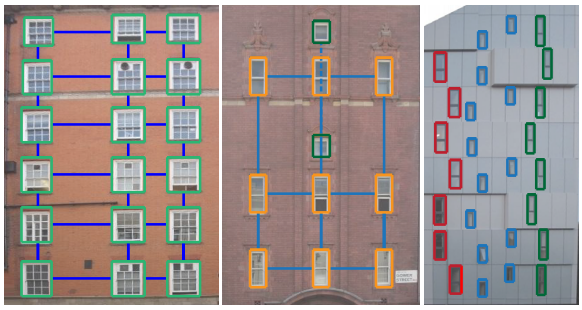

Figure 6: *Our facade structure analysis can also detect semi-regular structures. Note that in each case one element of each type was marked by the user.*

Selecting elements. Having extracted the dominant vertical and horizontal split lines, we finally mark the grid locations that are active (i.e., have a matching element) based on the scores computed previously as $w_t(W_s, W_t)$. Specifically, we mark any the grid locations as active if it has a matching score larger than 50% of $\max_{W_t} w_t(W_s, W_t)$ (see Figure 6-right).

4.2. Facade Element Analysis

In this section, we analyze each repeated element, i.e., a single window, to recover its deformation parameter. We make the following observations about windows that typically have low degrees of freedom: (i) windows (blinds) slide to open vertically or horizontally, i.e., the changes are approximately on the facade plane and hence can be analyzed directly on the rectified images (see Section 4.2.1); or (ii) windows (shutters) rotate to open, and hence need to be analyzed using some rough 3D information and back projected on the image to verify the solution (see Section 4.2.2). Note that in cultural settings where these observations are not satisfied, then our analysis method can fail. Further, windows can broadly be classified as opaque (e.g. blinds, wooden shutters) or transparent (e.g., glass on a wooden/metal frame). In the case of opaque elements, the opaque segments (e.g., blinds) share similarity across repetitions; while the rest of the windows (i.e., open elements) share little similarity across the repetition structure since they represent interiors of different rooms. Based on these observations, we now describe how we extract the window deformation parameters either automatically, or based on user hints (e.g., roughly marking a frame of a glass window).

4.2.1. Sliding window analysis

Each window can have its independent configuration of blind. We explain how we extract up/down sliding, while left/right sliding can be treated similarly. Such window configurations can simply be encoded as t that determines the status of the blind: $t = 0$ denotes open window, $t = 1$ denotes closed window, and $t \in (0, 1)$ denotes partial cover.

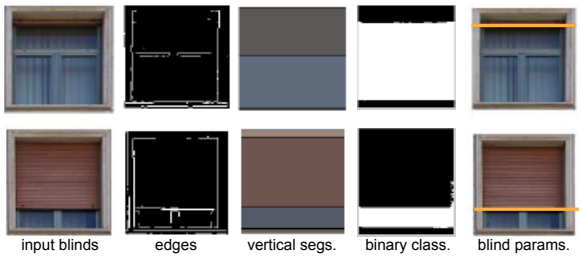

Figure 7: *Stages of sliding window parameter estimation. Starting from window images (first column), we perform Canny edge detection to extract dominant horizontal edges (second column) to vertically divide the windows into top and bottom segments (third column). We then look across the extracted top (and bottom) segments across multiple windows to determine consistent blind (and interior) colors and use them to reclassify regions for each window (fourth column). Finally, we use the information to extract the respective blind parameter for each window (last column) as a single parameter $t \in [0,1]$ capturing the extent to which the corresponding window is open. Here, we show analysis results for top-down sliding window types.*

For opaque blinds, t is the ratio between the area of the blind and the area of the window (see Figures 7 and 8).

Window blind area estimation. We start by detecting edges on each window element using Canny edge detector and enumerating the number of edge pixels per row of the image. Then, we mark a set of candidate horizontal splitting lines by picking the rows where the number of pixels are locally maximum. We use non-maximal suppression (using 3 pixels) to remove nearby candidates. These splitting lines divide one window into several vertically aligned sub-regions. For each sub-region, we compute its average color and use it to fill in the sub-region. Finally, we obtain a mask image of the vertical segmentation for each window.

In the mask image, each splitting line is associated with two colors, say top color c_{top} and bottom color c_{bottom}. Suppose c_{top} is the blind color and c_{bottom} is the window color, we compute a sum of fitness values over all windows and select the pair of colors with best fitness for the blind color and window color, respectively. We compute the fitness value for one mask as follows: for all possible splitting lines, find the line where the average color of the upper part of the window is as close to the top color and also the bottom part is as close to the bottom color, the lowest difference is used as the fitness value. Essentially, our approach exploits that: variance in colors across opaque blinds among the repeated elements is low compared to variation of colors across the open parts (i.e., interiors), which is distinctive for the different windows.

Figure 8: *Based on repeated windows of the same type but different estimated deformation/opening parameters, we synthesize a new window position by appropriately treating the background (i.e., window specific layer) and the blind, which is repeated across the windows. In case of sliding transparent (e.g., glass) window frames, we expect the user to roughly indicate a mask for the frame.*

Based on the extracted blind color c_b and interior color c_i estimates, we classify each pixel with color c_p as blind if $|c_b - c_p| \leq |c_i - c_p|$, else as interior. The classification results in a binary image where black denotes blind and white denotes interior. Finally, we determine the parameter t by finding the horizontal splitting line where the sum of black pixels above and white pixels below reach the maxima (see Figures 7 and 8).

4.2.2. Window shutter analysis

In this case, let the dihedral (rotation) angle be θ between one shutter and the background window frame, then half of the window is fully covered (i.e., closed) if $\theta = 0$, fully exposed if $\theta = \pi$, and partly shaded if $\theta \in (0, \pi)$. Here we use θ_l and θ_r to denote the rotation angles for the left and the right windows, respectively (facing the facade plane).

Window shutter estimation. Since the shutter rotations are off the facade plane, we assume access to rough 3D information. We obtain such rough 3D data using the method proposed by Wu et al. [WFP11]. Although a full grid detection using this method can fail due to different element positions, a coarse geometry is often sufficient to roughly estimate the 3D facade plane. (However, if even the rough estimation fails then the user has to intervene.) We use RANSAC-based plane fitting to extract the dominant plane from the rough 3D pointset. We then lift the extracted 2D grid obtained in the facade image analysis to get initial window placements in 3D. We estimate the size of the window and its two shutters in world coordinates and abstract its shape as three rectangles (in 3D), two rectangles for the window shutters and one for the window frame, by projecting the user labeled window onto the facade plane. The abstracted window is denoted as $W_a^{(3)}$. Note that we assume that the user labeled a fully opened window as the reference window W_s. For each of the detected repetitive window W_t, we perform a line-search (in the space of angles) based 2D-3D matching to find the corresponding θ_l and θ_r (see [CML*12] for joint 2D-3D optimization).

First, we translate $W_a^{(3)}$ so that its center is aligned with that of the W_t's 3D counterpart. Then, we rotate the left shutter by an angle $\hat{\theta}$ and apply the projective transformation from 3D→2D image. The 3D rectangle representing the left shutter will be mapped to a quad region $\mathcal{R}$ on the image.

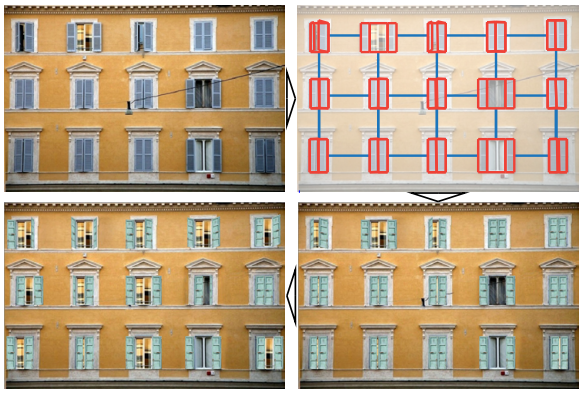

Figure 9: *Starting from an input image (top-left), we detect the repetition windows along with their rotation parameters (top-right). We can then synthetically replace the window textures (bottom-right) and subsequently virtually change window rotation parameters (bottom-left).*

Suppose the average color of $\mathcal{R}$ is $c_{\mathcal{R}}$ and the average color of the left shutter in W_s is c_{W_s}, the matching score is defined as $area(\mathcal{R})/(|c_{\mathcal{R}} - c_{W_s}| + 1)$. We also use an edge similarity between the image edges and the projected 3D window frame edges. We simply perform a line search over θ and retain the angle with maximal matching score. Figure 9 shows typical detection results with such rotating windows.

Again, in the case of window frames with transparent glass, we allow the user to manually specify a frame-mask using rough strokes. This mark is then used as a template to search and solve for sliding parameters or rotation angles as appropriate. In such cases, the edge similarity is given higher weight over color matching.

Iterative refinement. After detecting the window parameters, we synthetically bring all the windows to an arbitrary but canonical position (see Section 5), once again perform facade structure analysis, and iterate. We typically perform 2-3 iterations in our tests. Although the refinements are small, the process does help to improve the estimates of the element positions and their configurations (see Figure 2).

5. Factored Facades

We now describe how we use the extracted repetition pattern and window configurations to plausibly fill in missing information in background layers, eventually leading to a factored facade representation.

Blind synthesis. We use the set of window elements I_i with the corresponding blind parameters t_i to synthesize new windows with arbitrary blind configurations (see Figure 8). The synthesis involves two tasks: filling in the missing regions for the background and filling in the missing regions for the windows (if they are in partially open positions). We note that while the window backgrounds typically capture building interiors and differ even among repeated windows; the windows (i.e., blinds) are repeated and missing regions for one window can be filled in using information from other repeated (more) closed windows. Intuitively, since we know the opening parameters of the repeated windows, we implicitly have correspondence information across them. We exploit this implicit correspondence to solve a graph-cut based formulation for image completion and consolidation using information across the repeated window blinds.

We first move each of the window elements I_i to the closed position by appropriately moving (i.e., translating) the detected blind from position t_i to $t_i' = 1$. Note that we keep track of the empty regions using masks, i.e., for closing a window from t_i we have a mask region spanning $1 - t_i$ from the top (for a blind lowered down). Now, to perform image completion for window I_i, we use missing regions from the corresponding parts, if available, from the other images I_j for $j \neq i$ while minimizing seam lines with the current blind region. We use a standard multi-label graph formulation to perform this image synthesis [ADA*04].

Shutter synthesis. In this stage, we have a sampling of (repeated) window texture for a discrete set of angles, i.e., $\theta_1, \theta_2, \ldots$ with the goal to synthesize textures for intermediate values. Since we have a simple proxy geometry as the window rectangle in 3D, we directly bring the adjacent windows (in space of rotations) to target configuration θ and interpolate. While more advanced view-dependent textures can be used, we did not pursue such alternatives. In case of frames with transparent glass, we expect the user to provide a rough frame (using GrabCut [RKB04]).

Background layer. For each individual image element, we first remove the front blinds, shutters, or window frames. We used PatchMatch-based [BSFG09] image completion using structuring guidelines, where we prefer information propagation vertically since windows have a reflective symmetry.

Factored facade representation. Finally, for an input facade image, we encode the following information (see Figure 10): (i) the rectifying transform; (ii) the repetition grid along with which of the elements are occupied; (iii) the opening parameters for each window along with synthesized open/closed positions; (iv) the background layer.

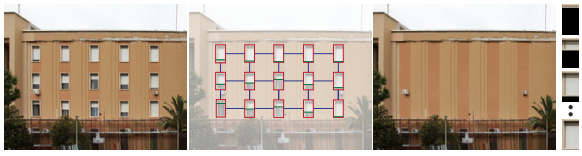

Figure 10: *Factored facade representation extracted from an input image. (Left-to-right) Input image, extracted repetition pattern with the individual window positions, the synthesized background layer, and intermediate blind positions (without synthesized interiors) (please zoom for details).*

6. Evaluation and Applications

We evaluated our framework on a large number of images including publicly available benchmark facade images (see supplementary materials for a wide range of facade analysis results). The results show that our framework is simple, requires marginal user hints, and yet is effective for analysis of facade syntax at a level very difficult to achieve with previous methods. The obtained factored facade representations are useful for several applications including interactive facade images, multi-view stereo, procedural modeling, and facade-syntax preserving synthesis.

Interactive facade images. We have implemented a basic user interface to interactively synthesize new facade images based on the recovered factored facade representations (see demo and video). The main functionalities are:

- *Repetition restructuring.* We allow the user to interactively change the grid structure of the repetitions and reposition a specified element. During the relocation, all the elements in the same row/column are updated while preserving alignment (see also [WWF*10]).
- *Window blind synthesis.* The user can interactively change the positioning of a window blind by dragging the bottom edge of the blind up and down in the original facade view.
- *Window shutter synthesis.* The orientation of the window shutters can be specified interactively.

Please note that although the interactive synthesis is performed on 2D facade images, we utilize the underlying 3D information. For example, when changing the shutter parameters, we rotate the shutters in 3D and then project it back to 2D. (In the default mode, shadow effects are turned off.) Such user interaction mimicks real-world behavior and makes the synthesis more realistic.

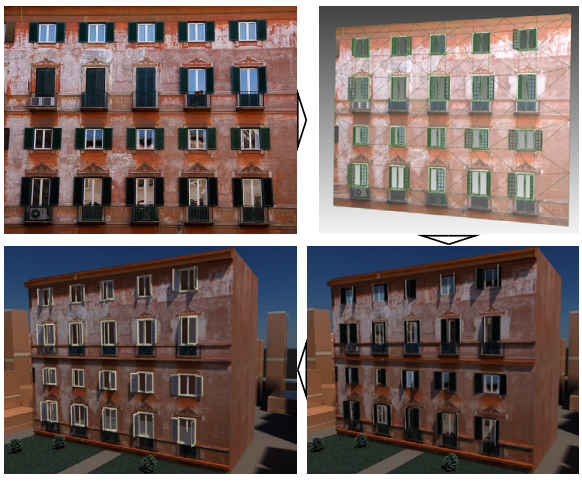

Figure 11: *Image-based 3D facade synthesis: input image (top-left), image-based synthesized 3D facade mesh (top-right), rendered synthesized model with different window rotations (bottom-right), and new window geometries positioned in 3D (bottom-left).*

Figure 12: *Starting from two input facade images with one window marked on each (top), we extract the repetition pattern of the respective windows along with their deformation parameters (both sliding windows). We then swap the windows across the two images to synthesize new facade images. Note that drastic rescaling of the windows can produce artifacts due to varying frame thickness (bottom-right).*

Procedural generation of 3D facades. We can procedurally generate a facade model as a 3D counterpart of the input 2D facade image while taking into account the element level deformations (see Figure 11). Other synthesis results as replacing the repetition elements with new elements can also be naturally realized.

Facade-syntax preserving synthesis. We can easily create novel facade images while preserving the facade-syntax using the extracted element parameters and synthesizing new element variations. Figure 12 shows an example of swapping window elements across two images.

Multi-view Stereo. Structure from motion (SfM) and multi-view stereo (MVS) are popular techniques to reconstruct 3D data from a series of 2D images and has been widely used in reconstruction of building facades. The quality of the reconstructed data, however, directly depends on the detection of feature correspondences across the images. Compared to wall regions which lack sufficient features, facade elements (e.g., windows) usually contain more image/geometry level details and are good candidates for detecting feature correspondences. As a result, the variations caused by changes in element parameters pose challenges for SfM based reconstructions. For example when we apply the patch-based MVS algorithm of Furukawa et al. [FP09] on synthetic scenes with facade element variations we obtain poor results (see Figure 13). Instead, after recovering the window deformation parameters for each image, we bring the windows to a canonical position in all the images and use them as input for the MVS algorithm. We observe that the reconstruction quality is significantly better. We have used the dense

Figure 13: *(Top) We start with 10 input images of a synthetic scene, but each window randomly positioned. We use the known (since synthetic) camera positions to extract dense reconstruction from the images. The result is poor for the windows as image-level correspondence is imperfect. Instead, we detect repetition grids on each image, and use the rough 3D points from the previous stage to help estimate the rotating window parameters. Subsequently, we synthetically position all the windows to a canonical position (as in top-left image) and run dense reconstruction. Not surprisingly, the recovered 3D scene has a much improved resolution.*

reconstructions for this comparison to emphasize the significance of the improvement. We expect similar improvements for SfM which is essential for recovering the transformations between the images in a multi-view setting.

Implementation details and performance. We implemented our algorithm using a mixture of C++ and MATLAB code running on an Intel Xeon X5680 @ 3.33 GHz (2 processors) 8 GB RAM and Windows 7 64-bit computer. We use EdgeLink [Kov] for edge detection and OpenCV [Ope] for basic image processing operations. Typically, the grid estimation and blind parameters produce about 90% accurate results while the shutter parameters estimation is around $80-85\%$ accurate. The facade repetition detection usually takes 5-10 seconds with 80% of the time being devoted to calling Matlab code; while image completion and consolidation runs in a few seconds. Subsequent edits are interactive (see demo).

User interactions. We support three types of user interactions in our system: (i) during facade structure analysis (Section 4.1), the user marks a single window element. In case of facades with multiple element types, the user marks one element for each type; (ii) during facade element analysis (Section 4.2), the user marks a frame-mask with rough strokes only when the window frame has transparent glass; and (iii) during interactive facade application, the user directly manipulates the factored facade representation (see supplementary video and demo application) to synthesize new facade images.

7. Conclusion

We presented a semi-automatic algorithm to analyze input facade images to extract the regularity pattern among the repeated window elements, while also identifying the corresponding window deformations (i.e., displacement parameters for sliding windows and angle parameters for rotating windows) to obtain a factored facade representation. We evaluated our framework on various challenging facade images, where state-of-the-art facade analysis algorithms fail, and demonstrated a range of novel facade manipulation possibilities, both on input images and on synthesized 3D geometry while preserving extracted facade-syntax.

Limitations. Our work has several limitations: (i) large-scale window element variations can result in failure in the initial repetition detection preventing the algorithm to proceed; (ii) significant light/shadow variations, or presence of outlier elements, can also lead to failure to abstract the window variations or even miss the repetition grids; and (iii) cultural differences across window types can result in failures.

In the future, we would like to investigate a more tightly coupled repetition detection and window variation estimation to address these issues. Also, the proxy geometries (in 3D) can be used towards more accurate appearance modeling effects under known light and camera settings (see also [ZCC*12]). Finally, the recovered window templates (with allowed variations) can be used to reveal interesting image parts, which cannot be explained by extracted window templates, leading to novel image understanding possibilities (see Figure 14 for an initial result).

Acknowledgement. We thank the anonymous reviewers for their useful suggestions, Suhib Alsisan for initial discussion on the topic and Charlotte Rakhit for video voiceover. Sawsan sincerely thanks Duygu Ceylan for her many good suggestions and constant encouragement. The work was partially supported by the Marie Curie Career Integration Grant 303541.

Figure 14: *Having extracted factored facades for window elements (top), the allowed frame movements can be used to explain and prune out edges from the original edge maps (middle). This can potentially reveal interesting parts not explained by the learned deformations of the windows, for example, here we flag regions of interest (flower pots) as unique compared to the window template (bottom).*

References

[ACM12] AIGER D., COHENOR D., MITRA N. J.: Repetition maximization based texture rectification. *Computer Graphics Forum (EUROGRAPHICS)* (2012). 4

[ADA*04] AGARWALA A., DONTCHEVA M., AGRAWALA M., DRUCKER S., COLBURN A., CURLESS B., SALESIN D., COHEN M.: Interactive digital photomontage. *ACM Trans. Graph. 23*, 3 (2004), 294–302. 7

[AM12] ALSISAN S., MITRA N. J.: Variation-factored encoding of facade images. In *EUROGRAPHICS Short Paper* (2012). 3

[BH01] BRIECHLE K., HANEBECK U. D.: Template matching using fast normalized cross correlation. In *Proc. of SPIE* (2001), vol. 4387. 3

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: PatchMatch: A randomized correspondence algorithm for structural image editing. *Proc. ACM SIGGRAPH 28*, 3 (Aug. 2009). 7

[CML*12] CEYLAN D., MITRA N. J., LI H., WEISE T., PAULY M.: Factored facade acquisition using symmetric line arrangements. *Computer Graphics Forum (EUROGRAPHICS)* (2012). 2, 6

[DTM96] DEBEVEC P. E., TAYLOR C. J., MALIK J.: Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *SIGGRAPH* (1996), pp. 11–20. 2

[FCSS09] FURUKAWA Y., CURLESS B., SEITZ S., SZELISKI R.: Manhattan-world stereo. In *CVPR* (2009). 2

[FJZ05] FRUEH C., JAIN S., ZAKHOR A.: Data processing algorithms for generating texture 3D building facade meshes from laser scans and camera images. *IJCV 61* (2005), 159–184. 2

[FP09] FURUKAWA Y., PONCE J.: Accurate, dense, and robust multiview stereopsis. *IEEE PAMI 32* (2009), 1362–1376. 8

[KGV83] KIRKPATRICK S., GELATT C. D., VECCHI M. P.: Optimization by simulated annealing. *Science* (1983), 617–680. 5

[Kov] KOVESI P.: Edge linking and line segment fitting. 9

[LZS*11] LI Y., ZHENG Q., SHARF A., COHEN-OR D., CHEN B., MITRA N. J.: 2d-3d fusion for layer decomposition of urban facades. In *ICCV* (Barcelona, Spain, November 2011). 2

[MPWC12] MITRA N. J., PAULY M., WAND M., CEYLAN D.: Symmetry in 3d geometry: Extraction and applications. In *EUROGRAPHICS State-of-the-art Report* (2012). 1

[MWA*12] MUSIALSKI P., WONKA P., ALIAGA D. G., WIMMER M., VAN GOOL L., PURGATHOFER W.: A Survey of Urban Reconstruction. In *EUROGRAPHICS 2012 State of the Art Reports* (2012), Eurographics Association, pp. 1–28. 1

[MWW12] MUSIALSKI P., WIMMER M., WONKA P.: Interactive coherence-based façade modeling. *CGF (EUROGRAPHICS)* (2012). 2

[MZWG07] MÜULLER P., ZENG G., WONKA P., GOOL. L. V.: Image-based procedural modeling of facades. In *SIGGRAPH* (2007). 2, 4

[Ope] : *The OpenCV Reference Manual*. 9

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L.: Discovering structural regularity in 3D geometry. *ACM TOG 27*, 3 (2008), 43:1–43:11. 4

[RKB04] ROTHER C., KOLMOGOROV V., BLAKE A.: "grabcut": interactive foreground extraction using iterated graph cuts. *ACM TOG 23*, 3 (Aug. 2004), 309–314. 7

[SSS06] SNAVELY N., SEITZ S. M., SZELISKI R.: Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings* (New York, NY, USA, 2006), ACM Press, pp. 835–846. 2

[TKS*11] TEBOUL O., KOKKINOS I., SIMON L., KOUTSOURAKIS P., PARAGIOS N.: Shape grammar parsing via reinforcement learning. In *CVPR* (Colorado Spring, USA, 2011). 1, 3

[TSKP10] TEBOUL O., SIMON L., KOUTSOURAKIS P., PARAGIOS N.: Segmentation of building facades using procedural shape prior. In *CVPR* (San Francisco, USA, 2010). 2

[VAW*10] VANEGAS C., ALIAGA D., WONKA P., MÜLLER P., WADDELL P., WATSON B.: Modeling the appearance and behavior of urban spaces. *Computer Graphics Forum 29*, 1 (2010), 25–42. 2

[WFP11] WU C., FRAHM J.-M., POLLEFEYS M.: Repetition-based dense single-view reconstruction. In *CVPR* (2011). 3, 6

[WWF*10] WU H., WANG Y.-S., FENG K.-C., WONG T.-T., LEE T.-Y., HENG P.-A.: Resizing by symmetry-summarization. *ACM TOG 29*, 6 (December 2010), 159:1–159:9. 3, 8

[WWSR03] WONKA P., WIMMER M., SILLION F., RIBARSKY W.: Instant architecture. *ACM TOG 22* (2003), 669–677. 2

[ZCC*12] ZHENG Y., CHEN X., CHENG M.-M., ZHOU K., HU S.-M., MITRA N. J.: Interactive images: Cuboid proxies for smart image manipulation. *Proc. ACM SIGGRAPH 31*, 4 (2012), 99:1–99:11. 9

[ZSW*10] ZHENG Q., SHARF A., WAN G., LI Y., MITRA N. J., COHEN-OR D., CHEN B.: Non-local scan consolidation for 3D urban scenes. *ACM TOG 29*, 4 (2010), 94:1–94:9. 2