Symmetry-aware Façade Parsing with Occlusions

Andrea Cohen[1], Martin R. Oswald[1], Yanxi Liu[2], Marc Pollefeys[1,3]

[1]ETH Zürich, Switzerland [2]Pennsylvania State University, USA [3]Microsoft, USA

Abstract

Symmetries and repetitions are common and valuable features in urban scenes. We propose to leverage such regularity information in an efficient optimization scheme in order to segment a rectified image of a façade into semantic categories. Our method retrieves a parsing which respects common architectural constraints as well as detected repetitive structures and edge information. Additionally, the use of symmetry information allows us to efficiently deal with large occluded areas and to recover plausible façade images with a minimum of occlusions. Our approach yields state-of-the-art accuracy on datasets with challenging occlusions. Competitive works either fully fail to deal with large occlusions or they are an order of magnitude slower than our approach.

1. Introduction

Façade parsing is the process of segmenting rectified images of façades into semantic categories corresponding to architectural structures like windows, doors or balconies. It is an important task for visual reconstruction and scene understanding. For example, knowing the position and size of the windows reduces wrong feature matches while knowledge about the repetitive structure and its symmetries is beneficial for structure from motion. Façade parsing is also a key step for applications such as procedural modeling of buildings (*e.g.* for video games, virtual reality), architectural design, city surveillance, or accurate large scale 3D reconstruction of geometry and texture for entire cities. Therefore, the task of parsing rectified images of façades as the one illustrated in Fig. 1 has hence gained increasing attention in recent years [4, 20, 12, 1, 17].

After image rectification, the first step of most façade parsing approaches consists of a per-pixel labeling via semantic classifiers. However, such per-pixel labels are usually imperfect since building façades typically exhibit a set of structural rules that should be respected in order to ensure semantic plausibility of the classification. Another characteristic of this task that could be taken into account is the fact that most man-made structures often present symmetries and repetitive structures. Therefore, we aim to leverage all these information sources to obtain semantic segmentations of building façade such as the one illustrated in Fig. 1(a) into structured regions (Fig. 1(b)) that respect the regularities of the scene (as those shown in Fig. 2).

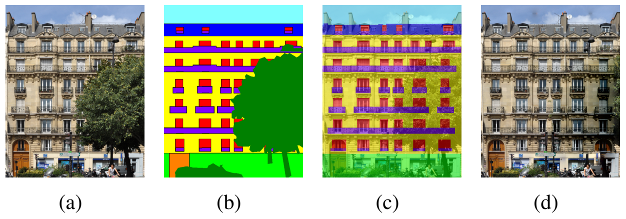
Figure 1. Our approach analyzes repeating structures to recover plausible information in occluded areas. The image of an occluded façade (a) gets semantically labeled (c) and the occluded area inpainted (d). Ground truth labels are shown for reference in (b).

Inspired by the work of [1], we present a greedy dynamic programming-based algorithm that imposes very few but common constraints on the parsing, while also respecting detected symmetries and repetitions, as well as image edges. We will first show how to extract symmetries and lattices on façade images and how to use these detections to cope with large occluded areas. Then, these regularities are used as soft constraints during the parsing which allows for stronger vertical alignment. With the detected symmetries and repetitions at hand, we are often able to recover large occluded image portions via inpainting with repetitive content. Finally, we illustrate the efficiency and efficacy of our method on 4 standard datasets.

Contributions. We propose an efficient method for semantic façade parsing that leverages symmetry and regularity information in order to deal with occluded parts in the images as well as softly enforce vertical alignment of architectural structures. Our approach yields state-of-the-art results on common benchmark datasets. Existing façade parsing methods are either not able to deal with occlusions or are by an order of magnitude slower than the proposed approach.

2. Related Work

Segmenting a rectified façade image into structured regions has been the focus of many works in the last years.

Façade Parsing. Teboul *et al.* [21, 20] solve the parsing problem by the use of split grammars. The parsing is obtained by sampling from the space defined by a specific grammar. Split grammars are simple and powerful, since they allow to express many parsing rules. However, they are not flexible enough to allow for deviations from these set of rules, and they also have to be user defined. This sampling-based approach also has no optimality guarantees and cannot be relied to produce repeatable results while also being very slow ($\sim$ 30 minutes processing time per image). Riemenschneider *et al.* [17] extended the use of grammars in order to take into account symmetry in their set of rules. This method also faces the problem of high complexity and very slow running times and therefore relies on image subsampling, therefore losing a lot of information that comes with higher image resolution.

Martinovic *et al.* [12] proposed an approach based on Recursive Neural Networks that aims to incorporate the architectural constraints on top of a general classifier's results. They introduce a three-layer approach that relies on a regularization scheme posed as a pairwise multi-label Markov Random Field (MRF). However, the resulting labeling is tied to the super-pixels used for the classifier, which do not always respect the structural constraints of a man-made scene. The third layer corrects this problem to some extent by applying some local corrections, but the obtained segmentation does not always respect the basic architectural rules. In a later work [11], a bayesian grammar is learned from labeled ground truth images and shown to work for parsing of façades with similar structure.

Kozinski *et al.* [5] present a binary linear program to enforce horizontal and vertical alignment of façade elements. However, this global alignment is very restrictive since it is not guaranteed that a scene will present this kind of regularity. They try to cope with this problem by adding exceptions which are very complex to solve, resulting in a running time of 4 minutes per image. In subsequent work [4], the authors cope with this lack of flexibility by returning to the use of split grammars in order to encode two-dimensional alignment of elements as well as occlusions. The task is formulated as a MAP-MRF problem over a 4-connected pixel grid and is solved efficiently in about 30 seconds per image. However, the use of a grammar (in the form of hierarchical adjacency patterns) requires careful definition by the user. We argue that most of the alignments can be discovered automatically from data by symmetry and regularity detection, and then automatically enforced in the parsing without user interaction for each dataset.

Only remotely related, [14] proposes a system for multi-view façade image editing for removing occluding objects. In a semi-automatic process a clean façade image is computed by selecting and blending image parts from different views.

Symmetries. The detection and use of symmetries and repeating structures is a very challenging and important problem in urban scene analysis. See [7] and [13] for a broad overview. Zhang *et al.* [26] focus on irregular façades and strive for a high-level understanding of façade structures by introducing a hierarchical description of façade elements with multiple layers. These structured elements are found as symmetric parts of the façade via symmetry maximization at all levels of the hierarchy. Musialski *et al.* [15] aim to repair partially occluded façade images, e.g. by street signs, cables, traffic lights, etc. After the detection of reflective and translational symmetries, they are used to identify and subsequently fill occlusions. The algorithm does not consider any semantics and requires a user provided coarse region mask of the symmetric image part.

Wu *et al.* [24] detect repetitive structures on façade images via the analysis of feature point repetitions and local reflective symmetries. Previously detected candidates of repetitive image parts are then evaluated by their matching quality to either reject, or identify and refine different types of repetitions. Xiao *et al.* [25] present a semi-automatic image-based façade modeling pipeline that aims for 3D model of a façade or entire building using mostly planar structures within an orthogonal alignment. In this work reflective symmetries are exploited to simplify the top-down subdivision process of façade parts, but it does not handle occlusions or compute semantic labels. Related to our symmetry-based inpainting approach is the work by Liu *et al.* [8] who also leverage symmetries to remove obstructions in images.

Relation to our Work. Our work is inspired by the algorithm introduced by Cohen *et al.* [1]. They propose a greedy sequential application of dynamic programs that can very efficiently recover a structured parsing of a façade. Even though the imposed structural constraints are hard-coded, these are very few, as they consist mainly of rectangular regions for each detected element, the location of doors are expected in the bottom of the image, while the roof, chimney and sky classes should appear on the top, with sky being above roof which is above the main facade. However, this algorithm does not take into account column alignment or any other sort of higher order clique. In this work, we propose to detect a-priori the symmetries and regularities of the buildings. On a second stage, we regularize the results of a general purpose classifier in order to respect the symmetry constraints. These results can readily be used by the simple parser of [1] to implicitly encode alignment on different dimensions. We also introduce the concept of binary costs in the parsing in order to further improve results.

3. Approach

As a common pre-processing step we first rectify all input images since it is an inexpensive method which drastically simplifies all further processing. Note that this step in no way limits the applicability of the algorithm and it is shared by all façade parsing methods. Given an image of a rectified fronto-parallel façade, the goal is to assign a semantic label corresponding to an architectural element to each pixel, under certain constraints. The set of possible labels, depending on the dataset, is usually a subset of the label set $\mathcal{L} = \{sky, chimney, roof, window, balcony, wall, door, shop\}$.

As introduced in [1], we will follow a set of very general constraints that are present in most urban buildings which we will refer to as $\mathcal{C}$. This set consists of the following constraints: all building elements (door, windows, roof, balconies) have a rectangular form, all window/balcony combinations lying on the same floor have the same height, balconies are required to be located below windows, the shops and doors are located on the bottom, chimneys originate from the top of the roof and the façade and the roof cover the entire width of the image. We would also like to incorporate column alignment, *i.e.* alignment of windows across different floors, as well as horizontal symmetry (windows located at equal distance of a vertical symmetry axes should be similar). Non-architectural elements such as vegetation, car, road, *etc*., have no constraints and are taken directly as the maximum label per-pixel from a general purpose classifier.

Finding the best possible parsing $\mathbf{z}$ comes down to finding the set of labels $\mathbf{z}_i$ for each pixel i such that the following score is maximized under the constraints previously described:

$$S(\mathbf{z}) = \sum_{i \in \{1,\dots,N\}^2} s_i(\mathbf{z}_i) \quad (1)$$

Where the score s_i of pixel i for label $\mathbf{z}_i$ is the likelihood output of a pre-trained classifier.

Overview. Our approach consists of the following steps: on a first stage, we detect a global symmetry, as well as repetitions (in the form of lattices) on the original façade images, as explained in Section 3.1. In Section 3.2, we show how to use the detected regularities in order to improve the per-pixel classification, as well as large occlusion handling. Finally, in Sec. 3.3 we apply a variation on the simple façade parsing from [1] in order to obtain a structured parsing of the façade that implicitly takes into account both occlusions and vertical alignment and symmetries. We also show, as an additional application of the regularity detection, how to do façade inpainting for large occluded areas (Section 3.4).

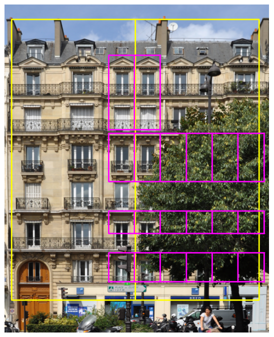
Figure 2. Detected global symmetry in yellow, and detected lattice (taking occlusion label into account) in pink.

3.1. Symmetry and Regularity Detection

In this section, we will explain the methods used to extract two different types of regularity on a given rectified façade image: global bilateral symmetry and translational repetitions *i.e.* lattices.

Global Symmetry Detection. In order to detect global reflective symmetries on the façade, we use a variation on the method described by Loy *et al*. [10].

On a first stage, a set of key-points are extracted on the image using SIFT [9], as well as a descriptor for each key-point. Next, mirrored descriptors are computed for each key-point. These symmetric descriptors are obtained by mirroring the image around the y axis and recomputing the descriptors for the mirrored key-point position. As opposed to the original method, we are only interested in finding symmetries with vertical axis, which are the most common in urban scenes, therefore only y-mirrored descriptors are used. Next, pairs of matching key-points are found by looking for their closest neighbor in feature space. Notice that the matching is done between original descriptor and mirrored descriptor. Only matches whose descriptor scale is close enough (up to a threshold) are taken into account. Each pair of matching points defines a potential symmetry axis. Since we are only interested in y-aligned axes, we only keep the pairs that fall under this category. Each pair is assigned a symmetry magnitude or score depending on their difference of scale as well as relative orientation and location. Next, all the symmetries defined by the remaining pairs are accumulated in a voting space in order to determine the dominant y aligned symmetries. The Hough transform for lines is used to find these symmetries where each matching pair casts a vote that is weighted by their symmetry magnitude. The maxima of the resulting Hough space are taken as the dominant symmetry axes. The bounding box of the population of pairs that voted for each axis is taken as the spatial extend of each symmetry. We choose the symmetry with the largest bounding box out of the top 10 dominant symmetries as the global symmetry that we will use in the subsequent steps of our method. The result of this detection is, therefore, a y aligned symmetry axis (defined by its x coordinate) as well as a bounding box for its area of influence. An example of the results obtained by this method can be seen on Figure 2.

Lattice Detection. We are interested in extracting axis-aligned lattices such that each lattice cell represents a repetitive element in the façade. In order to achieve this goal, we use a variation of the robust and efficient algorithm introduced by Wu *et al.* [24]. Since we are looking for lattices in fronto-parallel images, we do not need to find and refine vanishing points as described in their work.

We first extract upright SIFT features on the image, which are then matched along the horizontal and vertical direction. Matching pairs define possible repetitions intervals for which a histogram can be computed. Next, local maxima are extracted from this histogram to get a set of repetition intervals. Very small repetition intervals are skipped in order to detect big repetitive structures, as opposed to, *e.g.*, each individual window pane. The bounding box of features that voted for a specific interval can be seen as rough regions for the repetition. These candidate repetitions are then evaluated using a measurement that takes into account descriptor distance between corresponding pixels in the repeating regions, as well as “uniqueness” of the descriptors. A descriptor is unique if it is different enough from its neighbors, thus favoring repeating elements or motifs that are well textured and structured as opposed to large homogeneous or noisy areas. The repeating regions that have enough matching pixels (those whose similarity distance is smaller than a threshold) are kept and refined. We refer the reader to [24] for a more detailed description of this algorithm.

We perform an additional lattice extension step that is meant to take into account occluded areas. The aim is to extend the lattice both horizontally and vertically, as long as there are enough matching pixels in the new added lattice cells. We take the measurement described above, and a lattice column or row is added as long as each cell in the candidate column/row has enough matching pixels to a cell in the original lattice. However, for datasets with occluding labels (such as vegetation), we count occluded pixels as inliers. Therefore, if a cell is mostly occluded and is adjacent to the lattice, it will most likely be included in the extended lattice, unless there are enough unoccluded pixels that do not match the motif. Therefore, we can assume that, unless there is evidence on the contrary, the lattice structure is repeated behind the occlusion. The obtained grid for a partially occluded image can be seen on Figure 2. This additional steps allows us to hallucinate the façade structure behind large occluded areas during the parsing (Section 3.3). It is also a key component for large region inpainting as described in Section 3.4.

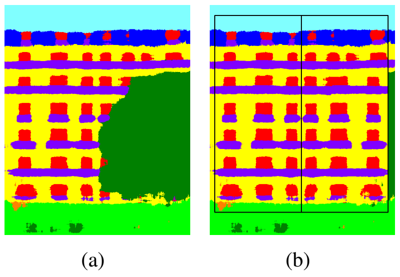
Figure 3. (a) shows the maximum label per pixel obtained by the classifier while (b) shows the labels corrected using the detected global symmetry.

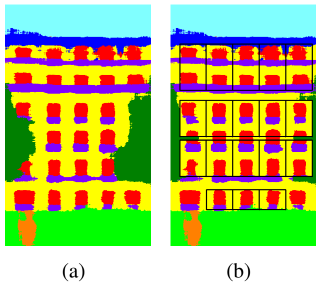
Figure 4. (a) shows the maximum label per pixel obtained by the classifier while (b) shows the labels corrected using the detected lattices.

3.2. Pixel Scores Modification and Occlusion Handling

The detected symmetries and lattices are used in order to modify the per-pixel likelihoods in order to improve classification accuracy.

The global symmetry detected in Section 3.1 is very valuable in order to recover from missing detections in the presence of large occluded areas, as symmetric areas might be occluded only on one side of the image. If the set of labels taken into account during per-pixel classification includes occlusion classes such as vegetation, then we can consider those pixels labeled as occluded as unknown. Therefore, we can search for all the pixels inside the area of the detected global symmetry and check for its symmetric counterpart. If its reflected pixel is not occluded, then one can simply replace the class likelihoods at that pixel with those of its non-occluded reflected counterpart. Figure 3 shows the maximum scoring label per-pixel for such an image, and how the modified classification looks like. This step allows us to get rid of an important part of occlusions, as most façades exhibit global symmetry and occlusion classes are not necessarily symmetric.

Repetitive Structures (Lattices). As described in Section 3.1, repetitive structures are detected as axis-aligned lattices or grids, which are extended over areas labeled as occluded as long as there is no evidence against the repetition. In order to further be able to infer what lies behind an occluded area, one can rely on the information of the lattice

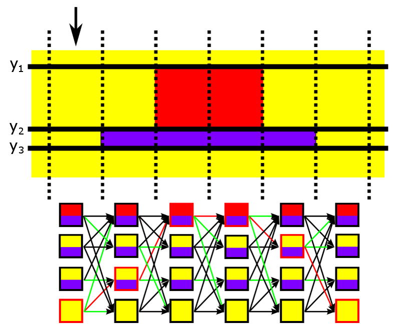

Figure 5. Illustration of how DP2 works: there are 4 possible states per column. The possible transitions from a chosen state to the next are highlighted in green, while the highlighted red ones are the chosen ones that maximize the total score sum over the row.

cells that are not occluded. Therefore, a median cell is computed from the non-occluded cells of a given grid, and their median class likelihoods are copied over the corresponding pixels that are labeled as occluded in all cells of the lattice. The result of this step is illustrated in Figure 4.

Additionally, in order to implicitly enforce alignment across repetitive elements, the class likelihoods of the non-occluded pixels of the lattice will be averaged with those of the median cell. Since the detection is subject to noise and errors, we perform a weighted average. More specifically: for a given pixel x_i, and its corresponding pixel in the median cell x_i^m, its class likelihoods are computed as:

$$\forall l \in \mathcal{L}: \quad s_l(x_i)^{\text{new}} := (1-\lambda)s_l(x_i) + \lambda s_l(x_i^m) \ , \quad (2)$$

where $l \in \mathcal{L}$ represents all the labels from the label set. In our experiments, we set λ to 0.3.

3.3. Façade Parsing

In order to maximize the score given Eq. (1) subject to the architectural constraints $\mathcal{C}$, we apply the dynamic programming-based algorithm presented in [1] which we briefly describe in this section, as well as a variation that takes into account edge information and column alignment.

The basic algorithm consists of a series of steps: first, all pixels are initially assigned the label for wall. The next and most important step involves finding the best scoring floors recursively, where each floor consists of window/balcony combinations of the same height. The detection of the single best-scoring floor within a height range is achieved through successive calls to an efficient dynamic program whose complexity is linear in the width of the image. The third step involves a variation of this dynamic program that finds the best combination of door and shop with the condition that the shop needs to start at the bottom of the image (note that this is easily modified for the case where there are no shop labels). The last step consists of an exhaustive search on the top part of the image for the best combination for the labels roof, sky and chimney.

Single Floor Parsing. Given the vertical boundaries of a floor (window and balcony heights), which we will refer to as y_1, y_2 and y_3, [1] introduces a linear time dynamic program that finds the horizontal boundaries of the windows and balconies that maximize the sum of scores across this floor. This dynamic program, referred to as **DP2** in [1], is a version of Viterbi's algorithm [23], with 3 basic states: presence of balcony, presence of balcony with window on top, or absence of both elements. For each one-pixel wide column bounded by y_1, y_2 and y_3, **DP2** explores which state is best (the highest scoring with respect to the pixel-wise likelihoods) depending on the previous column. In practice, the minimum number of states for a plausible parsing is 4, since two different states are needed to represent the presence of balcony (without window on top) such that no balcony can start and finish without having had a window appear on top at some point. The transition between these states are limited so that these conditions are met. **DP2** is illustrated in Fig. 5.

In order to automatically find the best scoring floor, one would need to run **DP2** on all possible height triplets $y_1 < y_2 < y_3$. Given N, the width/height of the image, this would run in $\mathcal{O}(N^4)$. However, [1] proposes an algorithm that relies on an efficient computation of upper bounds that allows for very few calls of **DP2**, therefore running in almost cubic time. Thus, the best scoring floors are found automatically in a very efficient way without the need of a-prioir knowledge of the height of the floors or the number of floors. The description of this algorithm is outside the scope of this paper, as we did not modify it.

Addition of Pairwise Costs. In addition to the per pixel class likelihoods, additional information can be extracted from an image, such as vertical and horizontal edges (x-gradient and y-gradient). In an optimization/maximization setting, these edges can be seen as pairwise-costs (or scores in the maximization case) where a transition between classes should be preferred at the locations where the gradient magnitude is high. In our rectified setting, we will only care about vertical and horizontal image gradients. As explained previously, a window/balcony row can be detected by calling **DP2** for different (y_1, y_2, y_3) combinations in order to find the best vertical parsing of such a row. In order to explain how to add the edge costs to this stage, we first explain **DP2** in more detail.

Given a triplet of vertical coordinates, (y_1, y_2, y_3), and n, the width of the image to be parsed, the input to **DP2** is the following:

- The sum of balcony minus wall scores for all pixel-wide columns delimited by (y_2, y_3), referred to as $B = \{b_1, b_2, \ldots, b_n\}$.
- The sum of window minus wall scores for all pixel-wide columns delimited by (y_1, y_2), referred to as $W = \{w_1, w_2, \ldots, w_n\}$.

- The set of states a column can take, referred to a $S = \{s_1, s_2, s_3, s_4\}$. s_1 is the absence of elements (the column will be left with a wall label), s_2 represents a balcony state (before having seen a window), s_3 is a window and balcony state, and s_4 is again, just balcony (seen after a window and balcony state).
- The set of possible transitions between states: $Tr = \{s_1 \to (s_1, s_2, s_3),\ s_2 \to (s_2, s_3),\ s_3 \to (s_1, s_3, s_4),\ s_4 \to (s_4, s_1)\}$.

The first step of **DP2** is a forward pass through the image x-coordinates in which two tables T_1 and T_2 of size $(|S|, n)$ are constructed. For each x-coordinate, and for each possible state s, the tables are built as:

$$\begin{aligned} T_1(s,x) &= \max_k \left(T_1(k, x-1) + b_x|w_x+b_x|0\right) : k \to s \in Tr \\ T_2(s,x) &= \arg\max_k \left(T_1(k, x-1) + b_x|w_x|0\right) \end{aligned} \tag{3}$$

Here the vertical bars represent or-operations whose meaning is explained in the following. The next step is a backwards pass that recovers the best scoring states per column by doing back-tracking on T_2.

In Eq. (3), each entry of the table $T_1(s,x)$ is filled after examining the scores of all states from the previous x-coordinate that allow to transition to state s. The addition of either b_x, $b_x + w_x$ or 0 depends on the state s. If s is a balcony state, b_x is added, otherwise, if it's a window/balcony state, $b_x + w_x$ is added, which is the score obtained by labeling column x with the corresponding labels. It is during this forward pass that edge information can be added depending on the chosen transitions. As an example, consider that for a given coordinate x, we are constructing the entry $T_1(s_3, x)$, which means that we have to choose the maximum among the following quantities:

$$T_1(s_3,x) = \max \begin{cases} T_1(s_1, x-1) + b_x + w_x + \alpha(g_y(x, y_1) \\ +g_y(x, y_2) + g_y(x, y_3) + \sum_{y=y_1}^{y_3} g_x(x, y)), \\ T_1(s_2, x-1) + b_x + w_x + \alpha(g_y(x, y_1) \\ +g_y(x, y_2) + g_y(x, y_3) + \sum_{y=y_1}^{y_2} g_x(x, y)), \\ T_1(s_3, x-1) + b_x + w_x + \alpha(g_y(x, y_1) \\ +g_y(x, y_2) + g_y(x, y_3)) \end{cases} \tag{4}$$

Where $g_x(x,y)$ is the magnitude of the x-gradient at pixel (x,y) and $g_y(x,y)$ the y-gradient. Notice that all three possible transitions share the unary score given by $b_x + w_x$ as well as the vertical gradients taken into account when adding a column of balcony/window. However, the first possible transition involves going from a wall label to a window/balcony label, therefore adding the x-gradient for the whole column. The second transition, which involves changing a balcony-only label to a balcony/wall one, takes into account only the x-gradient between y_1 and y_2. All gradients are added using a weight α that needs to be tuned so that the class likelihoods and the gradient magnitudes are on a similar scale. These transitions can be extended similarly to all other 3 states in this simple and logical manner, without increasing the complexity nor the running time of the algorithm.

3.4. Façade Occlusion Repairing

The previously identified symmetries are helpful to identify and repair occlusions of the façade images [15]. In contrast to [15] which uses pixel-wise independent occlusion reasoning and inpainting which can lead to ghosting artifacts, we employ an approach that is more robust to image misalignments and distortions. Furthermore, we are able to exploit multiple types of symmetries to recover occluded façade areas: We first make use of the global bilateral symmetries and subsequently use the translational repetitions to inpaint as many occluded areas as possible.

Global Symmetry Occlusion Inpainting. The global inpainting is especially useful to recover large occluded areas, which are mostly due to vegetation. We therefore use the global reflection to copy pixels from one side to the other when only one side is labeled as *vegetation*. In order to avoid artifacts due to copying image parts, we perform image blending as described in the following.

Image Blending. In order to obtain a visual appealing image composition when copying image parts, we use Poisson image editing [16] to blend the copied patch with the original image. That is, we compute the newly composed image I_c from the original image I_o and the image patch I_p as the minimizer of the following energy:

$$I_c = \arg\min_{I_c} \int_{\Omega_p} \|\nabla I_p - \nabla I_c\|_2^2 \, dx \quad \text{with } I_c\big|_{\partial\Omega_p} = I_o\big|_{\partial\Omega_p} . \tag{5}$$

Here, Ω_p denotes the image domain of the patch and $\partial\Omega_p$ its boundary. Visually explained, the patch I_p is copied into the composed image I_c in a way that image values along the patch boundary are identical to the original image I_o and the image gradients of composed image I_c shall be as close as possible to the ones of the patch I_p. The composed image can thus be computed by solving the resulting poisson equation with a standard linear solver.

Motif Computation and Occlusion Detection. Lets assume we found a translational repetition and a certain part of the image is repeated K times. Hence, we identified K similar looking image parts, called *motifs*, $\{I_m^i\}_{i=1}^K$ in our input image $I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3_{\geq 0}$. Due to noise and occlusions these motifs might not directly be suited to inpaint occluded parts. We therefore look for a motif that is free of occlusions. Such a motif can be generated by computing the pixel-wise median of all motifs, i.e. compute

$$I_{\text{med}}(x) = \text{med}\left(\{I_m^j(x)\}_{j=1}^K\right) \tag{6}$$

for every pixel x, in which the function $\text{med}(S)$ returns the median element of the set S.

Unfortunately, this approach easily leads to ghosting, thickening or thinning effects of image details due to imperfect alignment of the detected symmetry or distortions of the input image. We therefore identify the motif that is closest to the pixel-wise median motif:

$$\hat{I}_m = \underset{I_m^i \in \{I_m^1,\ldots,I_m^K\}}{\arg\min} \int_\Omega \left\|I_m^i(x) - I_{\text{med}}(x)\right\|_2^2 \, dx \ . \tag{7}$$

For the case that none of the motifs is free of occlusions we identify remaining occlusions by thresholding the difference from the mean and compute the set of occluded pixels as $\Gamma = \{x \in \Omega \mid \|\hat{I}_m(x) - I_{\text{med}}(x)\|_2^2 > \sigma\}$. The final motif image I_m is then computed as the combination of the two previously computed images:

$$I_m = \hat{I}_m\big|_{\Omega\setminus\Gamma} \cup I_{\text{med}}\big|_\Gamma \ . \tag{8}$$

This way, we obtain a motif which is free of occlusions as much as possible while minimizing ghosting artifacts. The computed motif I_m is then copied to fill any occlusions using the same Poisson blending as described above.

4. Experiments

We evaluate our proposed parsing approach quantitatively on four standard façade datasets: ECP dataset [19], eTrims dataset [3], Graz50 dataset [17] and ArtDeco [2]. The overall accuracy that we reach is on par with the best performing state-of-the art methods for each dataset, while being an order of magnitude faster in most cases. We would like to remind the reader that the main contribution of this paper is the handling of large occluded areas while keeping the speed and accuracy of the fastest and most accurate methods, as opposed to tuning up the performance of the parsing on the classical datasets. We also show a few qualitative results of the façade inpainting.

In order to allow for a fair comparison with the state-of-the-art façade parsing methods, we use the same per-pixel likelihoods as in [5, 4, 1]. For ECP, Graz50 and ArtDeco, the likelihoods are obtained using the multi-feature extension [6] of the TextonBoost algorithm [18] with multi-class boosting [22]. For the eTrims dataset, a Recursive Neural Network is used for pixel scores computation, as explained in [12]. Experiments are performed using five-fold cross validation setup as in previous methods, with 80% of the images being used for training and the rest for testing. We did not seek to improve overall classification accuracy by using more modern or better classification methods, since the goal of this section is to demonstrate the performance and structural correctness of our parsing scheme given the same basic unary potentials.

Method	[12]	[5]	[4]	[1]	Ours
Time [s]	110	240	30	**2.4**	3.1

Table 1. Average run-time per image for different approaches. Note, that [1] cannot deal with occlusions or inter-floor alignment.

For most datasets, we compare w.r.t. to previous approaches, mainly Kozinski *et al.* [4] and Cohen *et al.* [1]. For runtime comparisons, we refer the reader to Table 1. Our method is an order of magnitude faster than competitive approaches which can deal with occlusions. [1] is slightly faster than ours, but cannot handle occlusions.

ECP dataset. The ECP database [19] consists of 104 rectified images of façades with the labels: {*sky, chimney, roof, window, balcony, wall, door, shop*}. In Tab. 2, we compare to the mentioned state-of-the-art methods. As seen on the table, we slightly improve labeling accuracy w.r.t. [4] which is the state-of-the-art, while performing almost as fast as the fastest algorithm [1]. We improve upon [1] on both window and balcony labels, as expected when adding the regularity information. We also use the *chimney* label and show that the obtained accuracy is still better than the state-of-the-art for these sets of label. The results for the extended set of labels are presented in the last two columns of the table.

ArtDeco dataset. The ArtDeco dataset [2] consists of 80 images of rectified façades of very similar style. However, a large portion of the dataset presents large occluded areas, mostly due to vegetation. This dataset is ideal for testing the inference of the architectural elements in the presence of large occlusions, as it provides hand-annotated ground truth for the labels behind the vegetation. As presented by [4], we show results both for the segmentation of only the visual elements, as well as for the segmentation of the occluded façade structure, which we infer from symmetry and regularity. The results of the first task are presented in the first three columns of Table 4. For the second and more interesting task, we achieve very similar results as [4] while being an order of magnitude faster. This dataset validates the main challenge of our method: keeping the speed of the fastest method ([1]) and the accuracy of the best performing one ([4]). The results are presented in the last columns of Tab. 4. Note that the use of symmetry and regularity increases window and balcony detection accuracy by over 20% w.r.t. [1] while keeping the same speed. We also show some qualitative results on Fig. 6.

Graz50 dataset. For this dataset, the overall accuracy achieved is slightly below the one achieved by [4]. However, it is important to note that we achieve the best results for all classes except *sky*, which skews the overall accuracy.

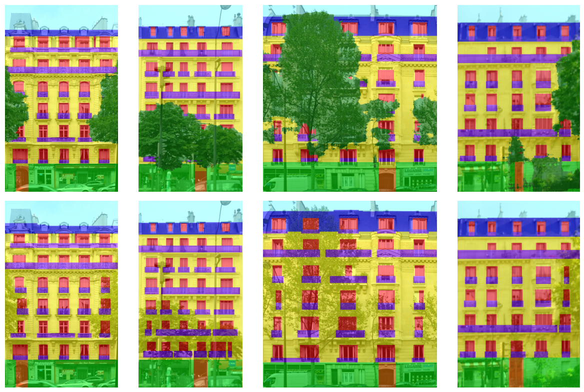
Figure 6. Results on the ArtDeco dataset, with and without vegetation (top and bottom respectively).

Class	[12]	[5]	[4]	[1]	Ours	[1]^c	Ours^c
Window	75	85	**87**	85	**87**	85	**87**
Wall	88	**91**	90	90	**91**	90	**91**
Balcony	70	90	91	91	**92**	91	**92**
Door	67	74	**79**	**79**	**79**	**79**	**79**
Roof	74	**91**	**91**	90	**91**	**91**	**91**
Sky	**97**	96	**97**	**97**	**97**	**94**	**94**
Shop	93	95	**97**	94	96	94	**96**
Chimney	-	-	-	-	-	85	**88**
total acc.	84.2	90.8	91.3	90.8	**91.8**	90.3	**91.4**

Table 2. Results on the ECP dataset [19]. (^{c} = with chimney)

Class	[17]	[5]	[4]	[1]	Ours
Sky	91	**93**	**93**	88	88
Window	60	82	84	76	**85**
Door	41	50	60	52	**64**
Wall	84	**96**	**96**	95	**96**
total acc.	78.0	91.8	**92.5**	89.6	91.6

Table 3. Results on the Graz50 dataset.

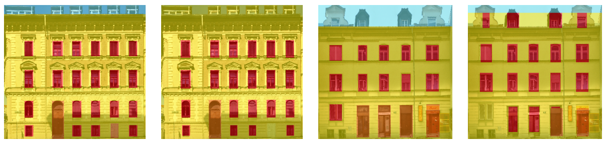
Figure 7. Example of ambiguities in the ground truth (left) for the sky class on the Graz50 dataset in comparison to our results (right).

We believe this is due to some ambiguities in the ground truth labels as can be seen in an example in Fig. 7. Again, a significant increase in window accuracy can be seen w.r.t. [1]. The results are presented on Table 3.

The class and overall accuracies for the eTrims datasets are presented in Table 5. Notice that for this dataset our method achieves very similar results as [1], since this dataset presents very little symmetry and grid regularity compared to the other datasets presented.

Qualitative Evaluation for Inpainting. We show how using both the global symmetry and the repeating elements helps clean up the façade images from the ArtDeco dataset. A few examples for inpainting are shown in Figure 8.

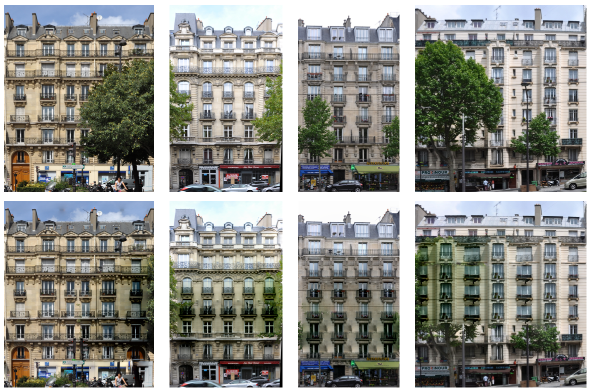
Figure 8. Inpaintint results for the ArtDeco dataset, with and without vegetation (top and bottom respectively).

Class	[1]^v	[4]^v	Ours^v	[1]	[4]	Ours
Roof	84	82	**85**	**86**	81	85
Shop	**97**	**97**	**97**	**97**	**97**	**97**
Balcony	85	**87**	86	65	82	**83**
Sky	94	**97**	95	90	**98**	90
Window	**82**	**82**	**82**	57	**82**	78
Door	56	57	**65**	58	57	**65**
Wall	**88**	**88**	**88**	**94**	89	90
Vegetation	**90**	**90**	**90**	-	-	-
total acc.	88.6	88.8	**88.9**	85.3	**88.8**	88.3

Table 4. Results on the ArtDeco dataset. (^{v} = with vegetation)

Class	[12]	[4]	[1]	Ours
Building	87	**92**	91	91
Car	69	**70**	**70**	**70**
Door	19	**20**	18	18
Pavement	**34**	33	33	33
Road	56	**57**	56	**57**
Sky	94	96	**97**	**97**
Vegetation	88	**91**	90	90
Window	**79**	70	71	72
total acc.	81.6	83.5	83.8	**83.8**

Table 5. Results on the eTRIMS dataset.

5. Conclusion

We presented a novel and efficient approach for façade parsing with large occluded areas. The proposed method exploits symmetry and regularity information in multiple ways. Depending on their existence our approach detects and exploits global bilateral symmetries and translational repetitions, which allows our algorithm to enforce vertical alignment during the parsing when relevant. This information also significantly helps to tackle occluded areas and is useful to obtain plausible occlusion inpainted façade images. We demonstrated the effectiveness of our method in multiple qualitative and quantitative experimental evaluations and showed that our method is an order of magnitude faster than the most accurate competitors.

Acknowledgements. This work was funded by the EU Horizon 2020 programme (grant No. 637221) and Google Tango.

References

[1] A. Cohen, A. G. Schwing, and M. Pollefeys. Efficient structured parsing of facades using dynamic programming. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 1, 2, 3, 5, 7, 8

[2] R. Gadde, R. Marlet, and N. Paragios. Learning grammars for architecture-specific facade parsing. *International Journal of Computer Vision (IJCV)*, 117(3):290–316, 2016. 7

[3] F. Korc and W. Förstner. eTRIMS Image Database for Interpreting Images of Man-Made Scenes. http://www.ipb.uni-bonn.de/projects/etrims_db/, 2009. 7

[4] M. Kozinski, R. Gadde, S. Zagoruyko, G. Obozinski, and R. Marlet. A MRF shape prior for facade parsing with occlusions. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2820–2828, June 2015. 1, 2, 7, 8

[5] M. Kozinski, G. Obozinski, and R. Marlet. Beyond procedural facade parsing: Bidirectional alignment via linear programming. In *Proc. Asian Conference on Computer Vision (ACCV)*, pages 79–94, 2014. 2, 7, 8

[6] L. Ladicky, C. Russell, P. Kohli, and P. H. S. Torr. Associative hierarchical random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1056–1077, 2014. 7

[7] Y. Liu, H. Hel-Or, C. S. Kaplan, and L. V. Gool. Computational symmetry in computer vision and computer graphics. *Foundations and Trends in Computer Graphics and Vision*, 5(12):1–195, 2010. 2

[8] Y. Liu, A. R. Pope, V. Verma, and S. C. Hsu. Symmetry-based interpolation in images, 2014. US Patent 9,240,055. 2

[9] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. 3

[10] G. Loy and J. Eklundh. Detecting symmetry and symmetric constellations of features. In *Proc. European Conference on Computer Vision (ECCV)*, pages 508–521, 2006. 3

[11] A. Martinovic and L. J. V. Gool. Bayesian grammar learning for inverse procedural modeling. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 201–208, 2013. 2

[12] A. Martinovic, M. Mathias, J. Weissenberg, and L. van Gool. A Three-Layered Approach to Facade Parsing. In *Proc. European Conference on Computer Vision (ECCV)*, 2012. 1, 2, 7, 8

[13] N. J. Mitra, M. Pauly, M. Wand, and D. Ceylan. Symmetry in 3d geometry: Extraction and applications. *Comput. Graph. Forum*, 32(6):1–23, 2013. 2

[14] P. Musialski, C. Luksch, M. Schwärzler, M. Buchetics, S. Maierhofer, and W. Purgathofer. Interactive multi-view façade image editing. In *Proc. of the Vision Modeling and Visualization Conference (VMV)*, November 2010. 2

[15] P. Musialski, P. Wonka, M. Recheis, S. Maierhofer, and W. Purgathofer. Symmetry-based façade repair. In *Proc. of the Vision Modeling and Visualization Conference (VMV)*, 2009. 2, 6

[16] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. Graph.*, 22(3):313–318, 2003. 6

[17] H. Riemenschneider, U. Krispel, W. Thaller, M. Donoser, S. Havemann, D. Fellner, and H. Bischof. Irregular lattices for complex shape grammar facade parsing. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1, 2, 7, 8

[18] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation. In *Proc. European Conference on Computer Vision (ECCV)*, 2006. 7

[19] O. Teboul. Ecole Centrale Paris Facades Database. http://vision.mas.ecp.fr/Personnel/teboul/data.php, 2010. 7, 8

[20] O. Teboul, I. Kokinos, L. Simon, P. Koutsourakis, and N. Paragios. Shape Grammar Parsing via Reinforcement Learning. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 1, 2

[21] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios. Segmentation of Building Facades Using Procedural Shape Priors. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[22] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 762–769, Washington, DC, June 2004. 7

[23] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory*, 1967. 5

[24] C. Wu, J. Frahm, and M. Pollefeys. Detecting large repetitive structures with salient boundaries. In *Proc. European Conference on Computer Vision (ECCV)*, pages 142–155, 2010. 2, 4

[25] J. Xiao, T. Fang, P. Tan, P. Zhao, E. Ofek, and L. Quan. Image-based Facade Modeling. In *Proc. SIGGRAPH Asia*, 2008. 2

[26] H. Zhang, K. Xu, W. Jiang, J. Lin, D. Cohen-Or, and B. Chen. Layered analysis of irregular facades via symmetry maximization. *ACM Trans. Graph.*, 32(4):121:1–121:13, July 2013. 2