High-Level Bottom-Up Cues for Top-Down Parsing of Facade Images

David Ok Mateusz Koziński Renaud Marlet Nikos Paragios
Université Paris-Est, LIGM (UMR CNRS), Center for Visual Computing
École des Ponts ParisTech, 6-8 av. Blaise Pascal, 77455 Marne-la-Vallée, France

Abstract

We address the problem of parsing images of building facades. The goal is to segment images, assigning to the resulting regions semantic labels that correspond to the basic architectural elements. We assume a top-down parsing framework based on a 2D shape grammar that encodes a prior knowledge on the possible composition of facades. The algorithm explores the space of feasible solutions by generating the possible configurations of the facade and comparing it to the input data by means of a local, pixel- or patch-based classifier. We propose new bottom-up cues for the algorithm, both for evaluation of a candidate parse and for guiding the exploration of the space of feasible solutions. The method that we propose benefits from detection-based information and leverages on the similar appearance of elements that repeat in a given facade. Experiments performed on standard datasets show that this use of more discriminative bottom-up cues improves the convergence in comparison to state-of-the-art algorithms, and gives better results in terms of precision and recall, as well as computation time and performance deviation.

1. Introduction

Image segmentation remains a generic, and in a large part, unsolved problem in computer vision. For many instances of this problem we are not restricted to using the information contained in the image alone. We may also resort to prior knowledge of the likely compositions of objects present in the scenes. Shape grammars are a concept that encodes such prior knowledge and is used for image segmentation. In the framework of shape grammars, segmentation amounts to assigning semantic labels to image regions and is known under the name of *image parsing* [24].

Applications involving images of highly structured scenes are likely to benefit most from the development of image grammars. One such field, which has been drawing increasing attention recently, is facade parsing [2, 10, 11, 3, 19, 18]. The goal of facade parsing is to automatically provide a hierarchical decomposition of a building facade into its constituent elements, given an image of the facade. Facade parsing has a variety of applications, including urban planning, thermal performance assessment of existing structures and reconstruction of models of existing buildings for games and simulators.

Amongst the most successful solutions to the problem of facade parsing are the top-down parsers proposed by Teboul et al. [19, 18]. They draw from the idea of split grammars for architectural modeling [10], where a variety of building models can be generated from a single grammar. The process is analogous to string or sentence derivation in formal and natural language processing. The goal of parsing here is to perform the derivation in such a manner that the resulting model corresponds to the input image. The top-down parsers have proven to be efficient in this task.

However, the robustness of this approach is dependent on the quality of the bottom-up information used for comparing the candidate models with the input image. Such information can be degraded because of challenging lighting conditions, facade appearance variation, or occlusions. This sensitivity is partly due to the fact that the underlying merit function is based on low-level, pixel- or patch-based information, as proposed in [18]. Besides, because of the high complexity of the problem space and due to the randomized nature of the approach, a good data-driven exploration of the solution space is crucial to simultaneously limit significant performance deviation in the parsing, and achieve fast convergence rate. But again, exploration presented in [18] is only based on pixels or patches.

To address the above issues, we propose a modified algorithm for top-down facade parsing that benefits from higher level and more robust abstractions, as provided by object detectors and geometric primitives. Namely, we integrate an object detector into the existing framework of [18], in this case a window detector. For this, we use a robust pattern search method, exploiting the fact that architectural elements present in a particular facade frequently share similar appearance. Additionally, to improve the convergence properties of the method, we guide the parser with the window detections and line segment cues. As a result, the parser not only better locates facade elements but also prunes the solution space much more selectively.

2. Related Work

The idea of representing the image contents in a hierarchical and semantized manner can be traced back to the work of Kanade and Ohta [12, 13]. However, the practical applications of grammars to image interpretation or segmentation are attributed to more recent works [4, 22, 5, 1].

In many works, the hierarchical and regular structure of man-made objects is explored to improve segmentation or detection results [22, 5, 1]. We focus on flexible grammars that allow the user to encode specific knowledge of the domain in the form of production rules that constrain the space of feasible solutions. The grammar-based image interpretation paradigm is thoroughly reviewed in the work of Zhu and Mumford [24]. A good example of this approach is the rectangle-based grammar of Han and Zhu [4], in which the prior knowledge is represented by means of an and/or graph. The terminal symbols are rectangles and the production rules combine them into rows, columns or grids, and allow for rectangle nesting. This case illustrates one of the difficulty of the problem: the number of terminals in the solution is unknown. The greedy algorithm presented in the paper copes well with this difficulty. However, since there is no semantic interpretation associated with the rectangles, there is no sensible way of deciding which of any two candidate parse trees is better.

The use of grammar-based facade parsing has been inspired by the successful application of split grammars for generating virtual urban environments [10]. The key to success is to encode in the grammar basic constraints on the generated objects: the principles of adjacency, non-overlap and snaplines. A number of research works has been aimed at applying the grammar principles for retrieving building models from images [19, 18, 8, 15]. In their work, Teboul et al. present an application of a 2D binary split grammar for parsing rectified facade images [19]. The method can accommodate several classes of terminal symbols and has been shown to be robust to partial occlusions and relatively flexible to variable facade appearance [18]. However, the algorithm suffers from a number of shortcomings, including the use of bottom-up gradient cues. The work of Yang et al. [23] focuses on the application of rank-1 matrix approximation for facade parsing. A binary classifier of window color is applied to the facade image. The image is divided into rectangular regions. The algorithm attempts to fit an irregular grid of windows to each of these regions. This is performed by approximating the output of the classifier by a rank-1 matrix. The main drawback of the algorithm is the constraint of two-class (window and wall) facades and the lack of flexibility in defining the grammar. Mathias et al. propose to use the grammar of [10] and generate the building while estimating the attributes of the applied grammar rules from the input images and a 3D point cloud. While the general idea seems attractive, the algorithm has not been shown to perform well with more than two classes of terminal symbols and accommodates only a small subset of rules of the original grammar [8].

3. Grammar-based Parsing

A shape grammar [16, 24] is a formalism to represent a structured collection of shapes. The symbols of the grammar are basic shapes, and the production rules transform one configuration of basic shapes into another. The split grammars [19, 10] are context-free shape grammars, where the production rules split the non-terminal basic shape on the left-hand side of the production rule along one dimension at a time. This simplification decreases the dimensionality of the space of parameters of a single production rule while preserving the expressive power of the grammar. This makes split grammars particularly suitable for modeling building facades.

The following part of this section gives a brief overview of 2D split grammars for facade modeling. The reader is referred to [19, 18] for more details.

3.1. Split grammars

The grammar dealt with in this paper operates on rectangles as basic shapes. Each production rule splits a rectangle along the horizontal or vertical dimension into a number of new rectangles. In the case of building grammars, the terminal basic shapes represent architectural elements, like windows and wall tiles, and the production rules encode the possible spatial compositions of these elements. Generating from the grammar, one obtains schematic images of building facades [19, 10].

Formally, a 2D split grammar $\mathcal{G}$ is a context-free grammar $(\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ where $\mathcal{N}$ is a finite set of non-terminal basic shapes $\{N_1, \ldots, N_m\}$, $\mathcal{T}$ is a finite set of terminal basic shapes $\{t_1, \ldots, t_n\}$, $\mathcal{R}$ is a finite set of rules $\{r_1, \ldots, r_l\}$ and $S \in \mathcal{N}$ is the starting shape (axiom).

Terminal and non-terminal symbols of the grammar are called *basic shapes*. They have a semantic type (e.g., window, balcony or floor) from a finite subset $\mathcal{C}$, and a bounding box. The vector of attributes of a basic shape of type c at position (x, y) with width w and height h is denoted as (c, x, y, w, h).

A rule $r : A \to B_1 B_2 \ldots B_k$ splits a single non-terminal basic shape A along a selected dimension into a sequence of basic shapes $(B_i)_{1 \leq i \leq k}$. For example, a vertical split rule decomposes a basic shape into multiple chunks of basic shapes along the y axis. The grammar can be transformed into Chomsky Normal Form, so that each rule applies at most one split to the processed basic shape. This reduces the number of continuous attributes of a production rule to one.

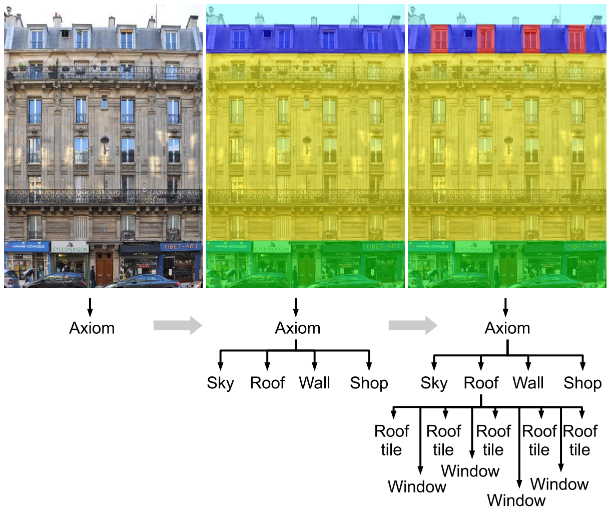


Figure 1: Top-down construction of a derivation tree. Top: the input image with the overlaid symbols. Bottom: the derivation tree under construction.

The generation process starts by applying a production rule to the axiom and continues applying production rules to the non-terminal basic shapes until there are only terminals in the derived configuration. The application of a rule requires the selection of the rule and the determination of its attributes, i.e., the number and positions of the splits. Generation is thus a sequence of decisions. It constructs a *derivation tree*. The root of the tree is the axiom S and all the nodes correspond to basic shapes, with terminal basic shapes at the leaves. An operation on a non-terminal node is performed by attaching to it the children nodes. Figure 1 illustrates the first two steps of a derivation process for an exemplary split grammar.

The idea of parsing is to construct a derivation tree corresponding to a given configuration of terminal shapes. It can be performed in a top-down or bottom-up fashion. In the first case, the derivation tree is constructed starting from the axiom; in the second case, the leaves of the tree are instantiated first.

In many practical facade parsing applications, the input data consists of a rectified facade image. The goal is to segment the image into a configuration of semantically meaningful regions that is allowed by the grammar. The grammar presented in [18] represents the Haussmannian architecture of the XIXth century buildings in Paris. The set of terminals includes sky, roof, shop, door, window, wall and balcony areas. As in [18], choosing a relevant split grammar is an issue we do not address. We assume the split grammar is known and written beforehand.

3.2. Reinforcement Learning for Top-Down Parsing

To assess the quality of a derivation tree, a pixelwise merit function is computed first. The merit function $m(x, y, c) \in [0, 1]$ estimates the likelihood that a pixel at (x, y) is of semantic type c. In [18], a random forest (RF) classifier is used to estimate m. The parser's goal is to find a derivation tree T that maximizes the *cumulated reward*, defined over all the terminals and the input image as: $\sum_{t_i} M(t_i)$. $M(t)$ is the reward for a single terminal $t = (c, x, y, w, h)$ and cumulates the reward over all pixels covered by the terminal:

$$M(t) = \sum_{x'=x}^{x+w} \sum_{y'=y}^{y+h} m(x', y', c)\ . \tag{1}$$

The task is difficult because the effect of decisions taken on the non-terminal nodes is not known until the terminal nodes are instantiated. In [18], the problem is formulated in terms of a Markov Decision Process. The parser acts as an agent constructing a derivation tree. Such tree has a state $s = (T, N)$ which consists of the tree T under construction and the next non-terminal node N to be processed.

The parser learns a policy function $\pi(s, a)$, which is the probability of choosing production rule attribute a in state s. By repeatedly simulating derivation trees with the current policy function $\pi(s, a)$, the parser updates at each iteration $\pi(s, a)$ accordingly with the history of cumulated rewards, which are determined when derivation trees are complete.

Denoting the current best optimal attribute by a^* and the prior distribution of rule attributes by $P(a|s)$, the policy function takes the form

$$\pi(s, a) = (1 - \varepsilon)\delta(a, a^*) + \varepsilon P(a|s), \tag{2}$$

where δ is the Kronecker delta and ε is a parameter of the algorithm. Specifically, the policy function either chooses with probability $1-\varepsilon$ the best rule attribute a^*, learnt "from experience", or draws another rule attribute a from the prior distribution $P(a|s)$ with probability ε.

After a number of iterations, $\pi(s, a)$ converges to the optimal policy function, which in turn can generate a derivation tree yielding the highest reward. In order to speed up the convergence, a data-driven version of the algorithm is used, where $P(a|s)$ is determined from the bottom-up cues. The prior distribution $P(a|s)$ for split locations is generated by marginalizing the horizontal and vertical gradient magnitudes along the y and x directions of the image.

3.3. Improved bottom-up cues for facade parsing

The algorithm proposed by [18] only utilizes local, pixel- or patch-based information. In particular, this is the case of the merit function, which lacks robustness. Such a limitation is significant in the case of buildings, because of possibly high variations of facade color and lighting conditions of image acquisition. Within the framework, it is also not possible to benefit from the fact that in a single image, elements of the same type may share similar appearance; e.g., the pixel classifier in [18] looks for any kind of window at any position. It does not reinforce the specific similarity of windows and window surroundings. Besides, the random nature of the Q-learning algorithm used in [18] results in significant result variations from run to run. In the following part of this paper we propose modifications to the algorithm to address these drawbacks.

Instead of constraining the bottom-up information to a local, low-level merit function, we propose to also use an object detector and a robust pattern search method. Our algorithm may thus exploit the repetition of specific instances of architectural elements within the facade. To better guide the parsing, we also design discriminative distributions of parsing actions using the object detections and line segment cues. As a result, the parser not only better locates elements but also prunes the solution space much more selectively.

In the rest of the paper, we describe our repetitive pattern search, our improved merit function, the new distribution for split positions and provide experimental results.

4. Detection of repeated objects

There are numerous methods for object detection, in particular based on learning techniques, such as the cascade classifier [20]. However, these methods are not perfect and still make errors, i.e., miss objects, hallucinate objects, or do not localize them properly. One or several parameters may usually be tuned to favor precision or recall. Moreover, they are trained to recognize a wide variety of object instances. They do not exploit the fact that, in some circumstances, only similar object instances appear in the image, e.g., windows on a given single facade. A hypothesis of these methods is that all detected objects are independent one from another (as long as they do not overlap).

We propose to use such a detector only to find few but very likely object occurrences, tuning the detector for precision. We then use these accurate detections as problem-specific models and rely on a robust pattern search procedure to look for similar instances of these models in the image. For more robustness, to improve recall, this procedure is repeated recursively on the new detections (resulting from the pattern search) until there are no more detections.

In the general case, other similar objects can be seen from different view angles. Assuming the visible part of the considered model object is mostly planar, other instances appear as a projective transformation of this model. In practice, when working on rectified images, mostly translation and rotation are to be expected. Depending on the application, scale could also vary and the expected transformation could then be a similarity. In the particular case of facade windows, there is no rotation, but the window sizes may vary. Indeed, windows on the same facade often have two or three different widths, depending on the size or use of the corresponding room. Although stretched horizontally, all windows on the facade however “look alike”. Also for older structures, including Haussmanian buildings, bottom floors have higher ceilings and higher windows than top floors, to compensate for the lesser illumination. But here again the window appearance is only stretched, vertically. To accomodate these different situations, our pattern search procedure looks for affine transformations of the model, that satisfy additional, problem-specific constraints. In particular, for facade windows on a rectified image, the score of an affinity matrix $A = \begin{bmatrix} a_1 & a_2 & t_x \\ a_3 & a_4 & t_y \end{bmatrix}$ is defined as $s(A) = \exp(-(a_2^2 + a_3^2))$, to discourage choosing affinities with strong shear components and large rotation angle, which can be generated due to inaccurately localized keypoints. Thus, affinities with a score below threshold $s_{\min}$ are discarded.

To improve robustness, our pattern search procedure is based on feature correspondence rather than pixel similarity (e.g., with SSD or NCC). Features located in the model subimage indeed provide a robust model abstraction and are a good indicator of the presence of other pattern instances in the rest of the image. Given the extent P of the pattern in image I, matches to consider are only pairs of features (x, y) such that x is in the model subimage P and y in the rest of the image $I \setminus P$.

In this situation, the transformation can be estimated from the feature point positions. Methods as RANSAC or one of its numerous variants [14] are well suited to determine such a transformation. However, several transformations are to be sought here, one for each instance of the model. RANSAC thus has to be iterated after a first instance is found, to find other ones, or a variant of RANSAC looking simultaneously for several models has to be used [25]. In our case, the model instances are expected to be clearly separated and a simple sequential RANSAC is enough. Note that pattern repetition intrinsically introduces match ambiguities between features associated to repeated elements. Depending on the amount of repetition, the number of match outliers can be huge. In our exemples with windows, the outlier contamination rate can reach 98% of about 500,000 feature matches. That would normally be beyond what most RANSAC variants can handle, but the fact that we are only looking here for (specific) affinities reduces the search space, as these affinities are defined by just a triple of matches. Furthermore, to prevent the exploration of a vast amount of triples than cannot lead to matching the model, we pick a first match (x_1, y_1) and draw the other two matches $(x_i, y_i)_{2 \leq i \leq 3}$ of a triple looking for y_i in the neighborhood of y_1, at a distance less than the diameter of P. Depending on the feature detector, this distance may take into account a scale factor defined by the relative scale of x and y. Moreover, to further speed up model estimation, matches are drawn based on the distance of their descriptor.

Finally, for more robustness, we sligthly increase the area of the model P, by constant factor, to include some environmental information. After patterns are localized, they are shrunk with the same factor to recover the estimated boundary of the actual model instances.

5. Enhanced merit function

We propose a new, more robust and more accurate merit function, which combines the local, low-level (pixel- or patch-based) information with standalone, high-level object detection. We interpret a detector for a class d as a pixel classifier $w_d(x, y)$, the value of which is 1 if (x, y) is a pixel belonging to the detected object and 0 otherwise. Not all semantic categories can have a sensible detector in practice. For instance, a classifier can be trained to detect windows or doors, but it is much harder to practically and reliably detect walls or roofs. Moreover, although the semantic types of terminals have no intersection in this kind of grammar, actual detectors can locate objects that overlap several semantic types of the grammar. For example, a general window model for detection can encompass both window-only areas and cast iron balconies in front of windows in the grammar. We thus make a difference between the semantic classes $\mathcal{C}$ of the grammar and the semantic classes $\mathcal{D}$ of the detectors, and define $c(d)$ as the classes of $\mathcal{C}$ that have an intersection with class $d \in \mathcal{D}$. In case we have several detectors, for classes in D, we define $c(D) = \bigcup_{d \in D} c(d)$.

The improved merit function m_+ gives confidence to the high-level detectors over the underlying, low-level merit: in case of a detection at a given pixel, it zeroes the merit of undetected classes (and the merit is renormalized). More formally, let $D_{x,y} = \{d \in D \mid w_d(x, y) = 1\}$ be the set of detected classes at pixel (x, y). We define $m_+(x, y, c) = m(x, y, c)$ if $D_{x,y} = \emptyset$, i.e., it is unchanged where there are no detection. Otherwise, if $D_{x,y} \neq \emptyset$, then $m_+(x, y, c)$ is defined as

$$\begin{cases} \dfrac{m(x,y,c)}{\sum_{c' \in c(D_{x,y})} m(x,y,c')} & \text{if } c \in c(D_{x,y}) \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

In our experiments we trained a general window detector that also localizes windows with a cast iron balcony in the foreground. We thus have $D = \{\text{whole-window}\}$ and $c(D) = \{\text{window}, \text{balcony}\}$. Figure 2 illustrates the improved merit function. We display $m^*(x, y, c)$ = $\arg\max_c m(x, y, c)$ with different colors for different classes (and likewise for m^*_+) and an image illustrating $w_{\text{whole-window}}$ with patches of the original image in places where whole windows have been detected.

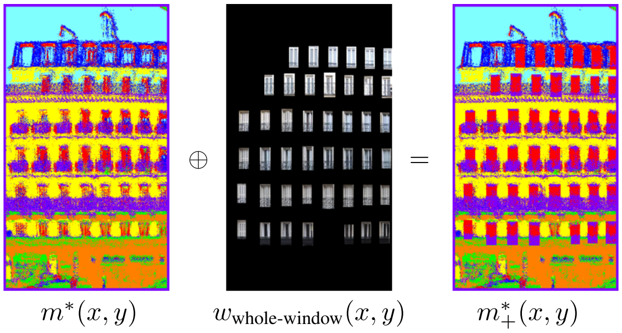


Figure 2: Classification based on the local merit function(left) vs the higer-level merit function (right). The result of the window detection is presented in the middle.

6. Enhanced distribution of split positions

Our last contribution is the design of more discriminative distributions of parsing actions $P(a|s)$ for the policy function $\pi(s, a)$. The most critical parsing action is the choice of the split position that decomposes a basic shape in the optimal manner. We consider two distributions for the split positions: one for horizontal splits and one for vertical splits. In [18], these distributions are obtained by accumulating gradients in the image along the x and y axes. However, these marginal distributions are noisy because of the harmful accumulation of gradients not corresponding to objects of interest, but resulting from shadows, texture or small architectural details. We propose another approach, based on marginalizing the distribution of line segments detected in the image. These higher-level abstractions are better split indicators.

We first detect line segments L in the image. (In our experiments we use LSD [21].) Let $v(y)$ be the distribution of vertical split positions. We denote by $[a_l, b_l]$ the projection of a segment $l \in L$ on the vertical axis, and by θ_l its angle with respect to horizontality. The value of the distribution at height y is computed as follows:

$$v(y) = C \sum_{l \in L} \mathbb{1}_{y \in [a_l, b_l]} \exp(-\frac{\tan^2 \theta_l}{2\sigma^2}) \,, \tag{4}$$

where σ is a parameter of the distribution and C is a normalization constant. The definition is symmetrical for horizontal splits. For our experiments we set $\sigma = 0.06$, which roughly leads to a segment contribution of $\frac{1}{3}$ for a segment with a $5°$ angle, whereas a perfectly axis-aligned segment contributes for 1. To reduce computation time, line segments with an angle beyond a threshold (around $10°$ for the given σ value) can be discarded right after detection.

In the same manner, we build a normalized histogram of the contours of the detected objects. The two distributions are summed and the resulting histogram is normalized, yielding the final distribution of applicable split positions. The major benefit of this approach is that the exploration

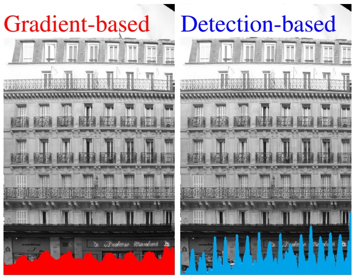


Figure 3: The gradient-based vs the detection-based distribution of split positions.

of the solution space is significantly pruned and the splits are attracted to optimal positions. The parser avoids being stuck in local minimum and the temporal standard deviation of the energy decreases over time faster than for the original algorithm (see Figures 3 and 5).

7. Experimental validation

Our experimental validation is mainly based on the *ECP CVPR 2010* and *ECP Benchmark 2011* datasets [17], that picture rectified Haussmanian buildings annoted with 7 semantic classes: sky, roof, wall, window, balcony, shop, door. We also used the *eTRIMS* [6] dataset, which displays many different architectural and building styles, with annotations for windows. As *eTRIMS* images are not rectified, we first performed the rectification (by hand). We also corrected and normalized the annotations, as the ground truth windows were sometimes erroneous or not delineated similarly as in the ECP datasets. All datasets are publicly available. Before presenting parsing results, we first assess the quality of our window detector alone.

7.1. Window detection

In our implementation, the initial model regions are obtained by the Viola-Jones cascade classifier (CC) [20]. We trained it to detect windows on the 20 training images of the ECP CVPR 2010 dataset. We also added negative examples of windows to minimize false detections. The classifier is parameterized by a detection score threshold τ, to balance precision and recall. We note CC(τ) the corresponding detector. Using τ, we may search only for reliable windows and then use them as models to find windows that were missed or wrongly localized by the classifier. For feature detection and matching we use Harris-Affine [9], MSER [9] and DoG [7] detectors, and the SIFT [7] descriptor.

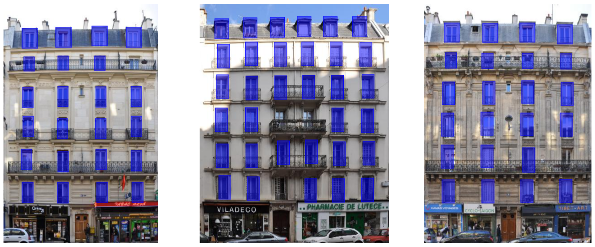
We compare the results of the cascade classifier alone for various detection thresholds (CC(τ)) against our pattern search procedure (CC(τ')+PS). We also compare with the grammar-based method of Teboul et al. (RL) [18], with two different grammars: a binary grammar (windows, not window) that assumes an irregular grid layout (RL(bin)), and a

Methods	$\overline{\textbf{TPR}}$ (%)	$\overline{\textbf{TNR}}$ (%)
CC ($\tau = 5$)	77	85.5
CC ($\tau = 10$)	70	81
CC ($\tau = 20$)	64	94
CC ($\tau = 30$)	56.5	95.5
CC ($\tau = 20$) + PS	76	94
CC ($\tau = 30$) + PS	71	95
RL (bin)	51.5	64.5
RL (haussm)	67	93.5

Table 1: Average performance of window detection on *ECP CVPR 2010* and *ECP Benchmark 2011*.

Methods	$\overline{\textbf{TPR}}$ (%)	$\overline{\textbf{TNR}}$ (%)
CC ($\tau = 5$)	46	96
CC ($\tau = 5$) + PS	60	93
RL (bin)	27	77

Table 2: Average performance of window detection on the rectified *eTRIMS* dataset.



TPR(%)	**TNR**(%)	**TPR**(%)	**TNR**(%)	**TPR**(%)	**TNR**(%)
86	96	77	96	81	97

Figure 4: Three window detection results (best viewed using magnification). Top row: results of our pattern search. Bottom row: corresponding detection rates.

Haussmannian grammar with 7 classes (RL(haussm)). The methods are compared in terms of mean true positive rate ($\overline{\text{TPR}}$) and mean true negative rate ($\overline{\text{TNR}}$).

Tables 1 and 2 summarize our results, and detection examples are shown in Figure 4. In all datasets, for a $\overline{\text{TNR}}$ more than 90%, our pattern search algorithm (CC+PS) outperforms other methods in terms of $\overline{\text{TPR}}$. It significantly improves the initial $\overline{\text{TPR}}$ of CC, by 12–15 points. It only slightly decreases $\overline{\text{TNR}}$ with respect to CC on rectified eTRIMS, by 3 points.

7.2. Facade parsing

To evaluate our approach, we ran the modified shape grammar parser on the test set of the ECP Benchmark 2011

	Gradient-Based Action Distribution								*Detection-Based Action Distribution*								*Type*
Plain RF merit	**60**	30	4	0	4	2	0		**72**	20	3	0	2	2	0	+12	*window*
	6	**81**	10	0	2	0	1		5	**84**	8	0	2	0	1	+3	*wall*
	16	31	**50**	0	2	0	1		18	26	**53**	0	2	0	1	+3	*balcony*
	0	1	1	**51**	0	0	48		0	2	0	**45**	0	0	53	−6	*door*
	13	3	0	0	**74**	10	0		8	2	0	0	**81**	9	0	+7	*roof*
	3	0	0	0	6	**91**	0		3	0	0	0	5	**92**	0	+1	*sky*
	0	7	3	8	0	0	**82**		0	8	2	8	0	0	**81**	−1	*shop*
Detection-enhanced merit	**71**	16	8	0	2	2	0	+11	**85**	8	5	0	1	2	0	+25	*window*
	9	**73**	15	0	2	0	0	−8	8	**76**	13	0	2	0	0	−5	*wall*
	15	27	**56**	0	2	0	1	+6	19	19	**60**	0	1	0	0	+10	*balcony*
	0	2	0	**55**	0	0	43	+4	0	2	0	**52**	0	0	45	+1	*door*
	16	2	0	0	**71**	10	0	−3	12	2	1	0	**77**	7	0	+3	*roof*
	4	0	0	0	6	**90**	0	−1	4	0	0	0	5	**91**	0	0	*sky*
	0	7	4	7	0	0	**81**	−1	0	7	3	7	0	0	**82**	0	*shop*

Table 3: The average confusion matrices. Top-left: the original algorithm. Top-right: the algorithm with the new action distribution. Bottom-left: the algorithm based on the modified merit function. Bottom-right: our final algorithm with both modifications combined.

dataset[1] (104 images). The window detector we used is CC($\tau = 20$)+PS, that experimentally performs best (see Table 1). We compare this parser against the original one, presented in [18]. In each case we run the parsers once. The results are evaluated with use of the ground truth annotations accompanying the dataset.

We present the results of the comparison in the form of the confusion matrices. The detection rate of building elements corresponds to the diagonal entry of the matrix (see [18] for details). Table 3 shows the efficiency of our two contributions separately. Consistent improvement of the results over the whole range of classes is visible already even for the partial contributions of the refined merit function and the new distribution of split positions separately. The improvement is amplified when we combine the two modifications into our final algorithm (bottom-right matrix). In particular, the window detection improves from 60% to 85% while most other rates are improved or preserved. Our algorithm also shows better convergence properties than the original one. In Figure 5 we show that the proposed algorithm converges faster, attains better values of the reward function and is less prone to deviate from the optimal solution. A few actual results are illustrated in Figure 6.

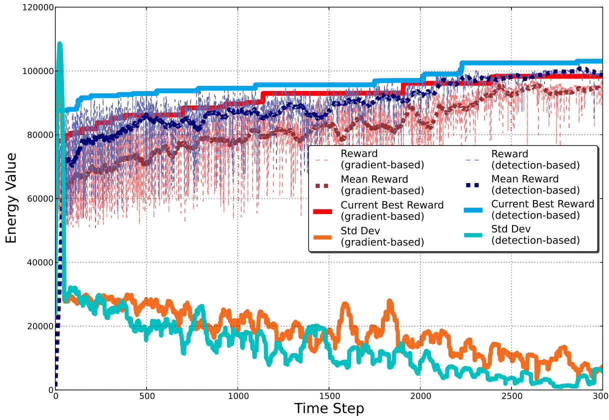


Figure 5: The convergence of our algorithm with respect to [18]. The plot shows the evolution of the reward, the current best reward and the standard deviation of the reward over time for a single run of the parser. The 'mean reward' is the plot of the reward function smoothed over time, to eliminate the high-frequency variation.

8. Conclusion

We have presented a method to enhance top-down facade parsing. It is based on the parser presented in [18] and carries two significant contributions with respect to the original version: the use of robust and adaptive object detectors to better estimate the merit function used by the parser, and the use of detected objects as well as line segments to better determine split positions, which effectively guides the reinforcement learning algorithm and speeds up its convergence. The significant performance improvements that we observe experimentally demonstrate the importance of high-level bottom-up cues in top-down parsing.

We note that our algorithm for repeated pattern detection can be used as a standalone algorithm for detecting multiple object instances in images. The algorithm significantly improves detection performance in cases where a particular type of object repeats over the image. In such situation, the algorithm can be viewed as an adaptive detector that adjusts to the specific appearance of the repeated object.

[1] This dataset must *not* be confused with *the ECP CVPR 2010 dataset* which consists of 10 test images only. Hence the numbers differ from what is reported in [18].

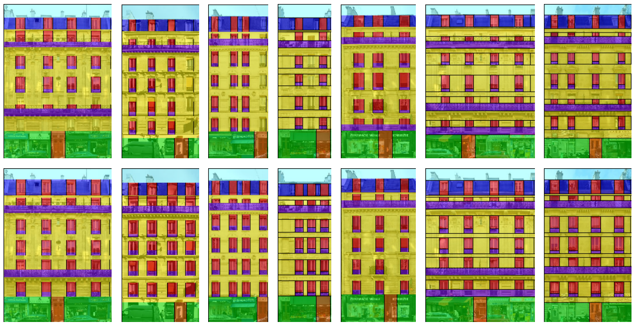


Figure 6: Examples of images for which our parser outperforms the original algorithm (best viewed using magnification). Top: results of original parser [18]. Bottom: our modified algorithm.

Acknowledgements

This work was carried out in IMAGINE, a joint research project between Ecole des Ponts ParisTech (ENPC) and the Scientific and Technical Centre for Building (CSTB). We thank Olivier Teboul for his great help and fruitful discussion.

References

[1] N. Ahuja and S. Todorovic. Connected Segmentation Tree - A Joint Representation of Region Layout and Hierarchy. In *CVPR*, 2008. 2

[2] F. Alegre and F. Dellaert. A probabilistic approach to the semantic interpretation of building facades. In *International Workshop on Vision Techniques Applied to the Rehabilitation of City Centres*. International Committee for Architectural Photogrammetry; (CIPA); Lisbon, October 2004. 1

[3] O. Barinova, V. Lempitsky, E. Tretiak, and P. Kohli. Geometric image parsing in man-made environments. In *11th European Conference on Computer vision: Part II*, ECCV'10, pages 57–70. Springer-Verlag, 2010. 1

[4] F. Han and S.-C. Zhu. Bottom-up/top-down image parsing with attribute graph grammar. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):59–73, 2009. 2

[5] Y. Jin and S. Geman. Context and hierarchy in a probabilistic image model. In *CVPR (2)*, pages 2145–2152, 2006. 2

[6] F. Korč and W. Förstner. eTRIMS Image Database for interpreting images of man-made scenes. Technical Report TR-IGG-P-2009-01, Dept. of Photogrammetry, University of Bonn, April 2009. 6

[7] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 6

[8] M. Mathias, A. Martinovic, J. Weissenberg, and L. V. Gool. Procedural 3D building reconstruction using shape grammars and detectors. In *3DIMPVT*, 2011. 2

[9] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65:2005, 2005. 6

[10] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool. Procedural modeling of buildings. *ACM Transations on Graphics*, 25(3):614–623, 2006. 1, 2

[11] P. Müller, G. Zeng, P. Wonka, and L. J. V. Gool. Image-based procedural modeling of facades. *ACM Transations on Graphics*, 26(3):85, 2007. 1

[12] Y. Ohta, T. Kanade, and T. Sakai. An analysis system for scenes containing objects with substructures. In *Proceedings of the Fourth International Joint Conference on Pattern Recognitions*, pages 752–754, 1978. 2

[13] Y. Ohta, T. Kanade, and T. Sakai. A production system for region analysis. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 684 – 686, 1979. 2

[14] R. Raguram, J.-M. Frahm, and M. Pollefeys. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In *ECCV (2)*, pages 500–513, 2008. 4

[15] H. Riemenschneider, U. Krispel, W. Thaller, M. Donoser, S. Havemann, D. Fellner, and H. Bischof. Irregular lattices for complex shape grammar facade parsing. In *CVPR*, 2012. 2

[16] G. N. Stiny. *Pictorial and formal aspects of shape and shape grammars and aesthetic systems*. PhD thesis, University of California, Los Angeles, 1975. AAI7526993. 2

[17] O. Teboul. École Centrale Paris datasets. http://vision.mas.ecp.fr/Personnel/teboul/data.php. 6

[18] O. Teboul, I. Kokkinos, L. Simon, P. Koutsourakis, and N. Paragios. Shape grammar parsing via reinforcement learning. In *CVPR*, pages 2273–2280, 2011. 1, 2, 3, 4, 5, 6, 7, 8

[19] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios. Segmentation of building facades using procedural shape priors. In *CVPR*, pages 3105–3112, 2010. 1, 2

[20] P. A. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004. 4, 6

[21] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:722–732, 2010. 5

[22] W. Wang, I. Pollak, T.-S. Wong, C. A. Bouman, and M. P. Harper. Hierarchical stochastic image grammars for classification and segmentation. *IEEE Transactions on Image Processing*, 15:3033–3052, 2006. 2

[23] C. Yang, T. Han, L. Quan, and C.-L. Tai. Parsing façade with rank-one approximation. In *CVPR*, pages 1720–1727. IEEE, 2012. 2

[24] S.-C. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Visions*, 2(4):259–362, 2006. 1, 2

[25] M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multiRANSAC algorithm and its application to detect planar homographies. In *ICIP (3)*, pages 153–156, 2005. 4