Notation-invariant Patch-based Wall Detector in Architectural Floor Plans

Lluís-Pere de las Heras, Joan Mas, Gemma Sánchez, and Ernest Valveny

Computer Vision Center - Universitat Autònoma de Barcelona
Campus UAB, 08193 Bellatera, Barcelona, Spain
{lpheras,joan.mas,gemma.sanchez,ernest.valveny}@cvc.uab.es
http://www.cvc.uab.cat

Abstract. Architectural floor plans exhibit a large variability in notation. Therefore, segmenting and identifying the elements of any kind of plan becomes a challenging task for approaches based on grouping structural primitives obtained by vectorization. Recently, a patch-based segmentation method working at pixel level and relying on the construction of a visual vocabulary has been proposed in [1], showing its adaptability to different notations by automatically learning the visual appearance of the elements in each different notation. This paper presents an evolution of that previous work, after analyzing and testing several alternatives for each of the different steps of the method: Firstly, an automatic plan-size normalization process is done. Secondly we evaluate different features to obtain the description of every patch. Thirdly, we train an SVM classifier to obtain the category of every patch instead of constructing a visual vocabulary. These variations of the method have been tested for wall detection on two datasets of architectural floor plans with different notations. After studying in deep each of the steps in the process pipeline, we are able to find the best system configuration, which highly outperforms the results on wall segmentation obtained by the original paper.

Keywords: graphics recognition; floor plan interpretation; patch-based segmentation

1 Introduction

Floor plan interpretation is an active research topic inside the graphical document analysis field. One of the main reasons is that most of the architectural projects involve the reutilization or modification of previous designs. Therefore, automatic floor plan interpretation becomes an actual need to be able to reuse existing designs and retrieve any kind of information of interest. In this direction, several works exist as those proposed by Dosch *et al.* [2] — for printed floor plans — and Juchmes *et al.* [3] — for sketched floor plans — which aim to construct the 3D representation of the buildings modeled in the floor plans. The authors also proposed in [4] a complete interpretation system using a syntactic model to interpret structurally, hierarchically and semantically this kind of documents.

However, floor plan interpretation is still a non-solved problem. The non-existence of a standard notation creates a large variability in building models. Thus, building entities as walls, doors, rooms, dimensions, areas, etc. are modeled differently in distinct plans. On top of that, existing floor plan interpretation approaches are based on vectorizing the images in order to extract the basic linear components. Interpretation is done by applying a set of rules that permit to group this basic components into high-level entities (walls, doors, etc.). Thus, these methods need to completely reformulate the segmentation process to deal with every different notation. This is the case of the recent approaches presented by Macé *et al.* in [5] for room segmentation in floor plans, and Ahmed *et al.* in [6] for a complete interpretation of floor plans. Both approaches assume a priori knowledge of the graphical structure of the walls.

With the aim of solving this problem, the authors proposed in [1] a wall segmentation approach capable to deal with plans having completely different notations. This technique, which is a bag-of-patches approach, is based on recent works on patch-based image segmentation and object localization [7, 8]. A grid of patches is defined using three different topologies over the learning images. Then, feature vectors are extracted from every patch and clustered into a codebook. After that, and using the ground-truth information, a probability of belonging to each class of objects is assigned to every word. In the testing phase, each patch is assigned to the nearest word in the dictionary, inheriting the class probabilities of the word. In order to test it for wall segmentation, this approach is tested in floor plans with different resolutions and wall notations. As the visual appearance of every class of objects under each different notation is automatically learned by the codebook and the probability distribution of patches, the method can be easily adapted to work with several notations by just providing a set of learning images.

In this paper we study the impact of introducing some modifications to the original patch-based detector for walls. Firstly, as floor plans could be found with different sizes and resolutions, an unsupervised image size normalization is applied over all the images in the dataset. Secondly, we evaluate different descriptors extracted from the patches (image pixels, PCA and Blurred Shape Model [9]) in order to analyze their suitability for different datasets and compare their influence in the global performance of the system. Thirdly, instead of clustering these feature vectors to build a codebook of patches, we train a Support Vector Machine classifier that permits to directly classify every patch into one floor plan object class. These modifications are introduced to the original method and have been tested over the same two datasets used in [1] in order to evaluate their benefits and disadvantages. All in all, as a result of this study, the best system configuration is found, which considerably outperforms the results obtained by the original approach.

The rest of the paper is organized as follows. In section 2 we describe all the steps of the proposed approach. Section 3 is devoted to explain the experimental setup and in section 4 we show the results of the application of the method. Finally, section 5 concludes the paper.

2 Methodology

The pipeline of the system is shown in figure 1. First, some standard image processing techniques are applied. Then, a grid is placed over the image and some features are extracted for every patch of the grid. In the learning step, a ground-truth of patch descriptors is used to train a classifier. Finally, in the testing, input image pixel categorization will rely on patch classification. All these steps are described in the remainder of this section.

2.1 Image preprocessing

In [1], all the dataset images are first binarized by applying the well-known approach proposed by Otsu in [10]. Then, textual information is removed using the text-graphic separation algorithm presented in [11]. In addition to that, in this paper we propose a new pre-processing step to normalize images in terms of resolution and line thickness.

Images in a given dataset can be at different resolutions and therefore, the line thickness can vary from one image to another. This would result in a larger variability in the visual appearance of the regular patches. To avoid it, an automatic line-thickness normalization is applied to all documents. This process consists in creating a histogram accounting for the length of the sequences of consecutive black pixels in the horizontal direction for each document. The histogram maxima corresponds to the thickness of the thinnest lines in the document. Then, all the images are resized using a bilinear interpolation method in order to achieve the same line width. Hence, the thickness of the walls becomes similar for all plans, and thus, the relative size of patches is similar for each floor plan.

2.2 Grid creation

As introduced before, a rigid grid is placed over the images where every cell defines a regular patch. Each patch allows to capture local redundancy of neighboring pixels which later can be modeled by different description approaches. We have used two different rigid grid topologies – those that performed better in [1] – forming squared patches over the images: *Non-overlapped regular grid* and *Overlapped regular grid*.

Non-overlapped regular grid: This grid is composed of squared non-overlapped patches directly defined over the image. The main advantage of this topology is its simplicity and its cheap computation cost. However, since each pixel of the image belongs to only one patch, final pixel class assignment will be only affected by its patch label, while sometimes one patch can contain pixels from different categories. Moreover final assignment of pixel category will strongly depend on how patches fall into the image.

Overlapped regular grid: In order to avoid the strong dependence on the grid location over the image, we have also defined a squared patched grid, but with overlapping. In this grid, each pixel belongs to several patches according to the parameter ϕ_{ov}, which specifies in pixels, the separation between patch neighbor centers. Therefore, final class assignment of a pixel is weighted up between the class probabilities of all its patches. This process is explained in section 2.5. The main advantage of this topology is that images are defined by more patches, thus object boundaries would be better segmented. On the other hand, for the same reason, pixel-level classification is more costly with respect to a non-overlapped grid.

2.3 Feature extraction

Once the desired grid is created, a patch-descriptor is calculated to represent every patch that contains at least one black pixel. Hence, since white patches are considered as *background*, they are ruled out in the learning step due to computational reasons. We have used three patch-descriptors to analyze the impact of feature extraction in the global performance of the system.

- **PID: Pixel Intensity Descriptor:** This simple descriptor is formed by concatenating the raw pixels of the patch in a row-wise manner.
- **PCA: Principle Component Analysis:** PCA is calculated over the row-wise vectors of all patches. The 95% of the discriminative information of the patches is maintained meanwhile the dimensionality is highly reduced.
- **BSM: Blurred Shape Model:** BSM is a shape descriptor introduced by Escalera et al. in [9] that has been successfully applied to different graphics recognition applications. The patch is divided in $n \times n$ equal-sized subregions ($BSMreg$) where each subregion receives votes from the points in it, and also from the points in the neighboring subregions. Each point contributes with a weight according to the distance between the point and the subregion centroid. The final description is a vector formed by concatenating the number of weighted votes received by each subregion.

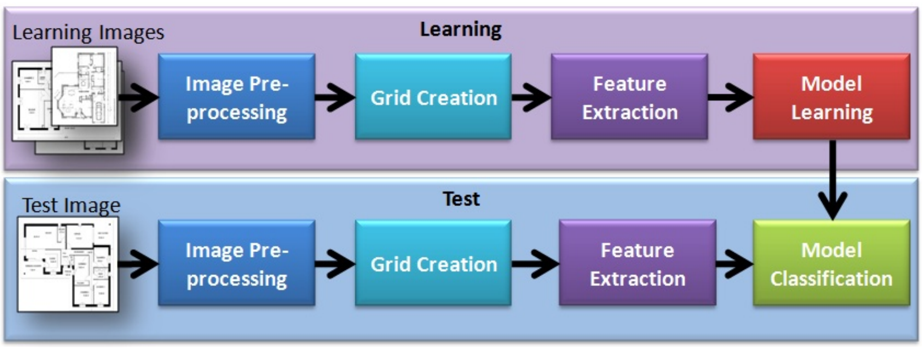

Fig. 1: Process pipeline

2.4 Model learning

In [1], a vocabulary of visual words is created by clustering similar patch-descriptors. Later, the likelihood of each word belonging to every class is learned from the training-set. Thus, each visual-word in the codebook has a probability of belonging to each of the classes. In the testing, each patch-descriptor of the input image is compared with all the words in the codebook and hard-assigned to the closest one, inheriting its class probabilities defined for such word.

Contrarily, to enhance the system velocity in testing time, which is a critical issue in [1], here we train support vectors to discriminate between classes. This process starts by choosing a previously specified number N of labeled instances, selected randomly from the two classes of objects to segment, $C = \{Wall, Background\}$, $N/2$ patches for each class. Then, a support vector machine (SVM) using the LIBSVM [12] implementation with a Gaussian RBF Kernel is trained on the labeled patches. The Radial Basis Function used is defined as

$$K(pd_i, pd_j) = e^{-\gamma \|pd_i - pd_j\|^2}, \tag{1}$$

where pd_i and pd_j are patch-descriptors and $\gamma \in \mathbb{R}_+$ is a the RBF width parameter selected by cross-validation.

2.5 Model classification

The earliest steps in the final classification process – from patch to pixel classification – are equal to those in the learning phase, as it can be seen in figure 1.

Firstly, every test image is preprocessed as explained in section 2.1. Secondly, features are extracted from patches in the manner it is described in sections 2.2 and 2.3. Thirdly, each patch-descriptor is classified using the SVM model trained in the learning phase. Finally, as in [1], final pixel classification will depend on the grid topology used to describe the input images.

In the case of the non-overlapped grid, every pixel in the image is contained in a single patch. Thus, pixels are directly categorized with the same label than their respective patches have obtained by SVM classification.

Distinctively, when an overlapped grid is used, pixels belong to several patches. Then, the classification of each pixel would depend on the several patches that contain that pixel, which allows to add contextual information in this process. Therefore, not only a classification label for each patch is needed, but also a confidence score for each class. In order to obtain a degree of classification confidence for each input patch, the probability estimation implemented in LIBSVM is used. In such a way, we can assign to every pixel a definite number of classification probabilities per object category $P(c_i|pd)$, one for each patch pd that the pixel px belongs to. Then, the final classification of a pixel can be seen as a combination of classifiers problem. Adapting the Mean Rule presented in the theoretical framework for combining classifiers proposed by Kittler et al. in [13], every pixel is finally classified to class c_i according to:

$$C(px) = \arg\max_i mean(P(c_i|pd)), \forall pd \mid px \in pd. \tag{2}$$

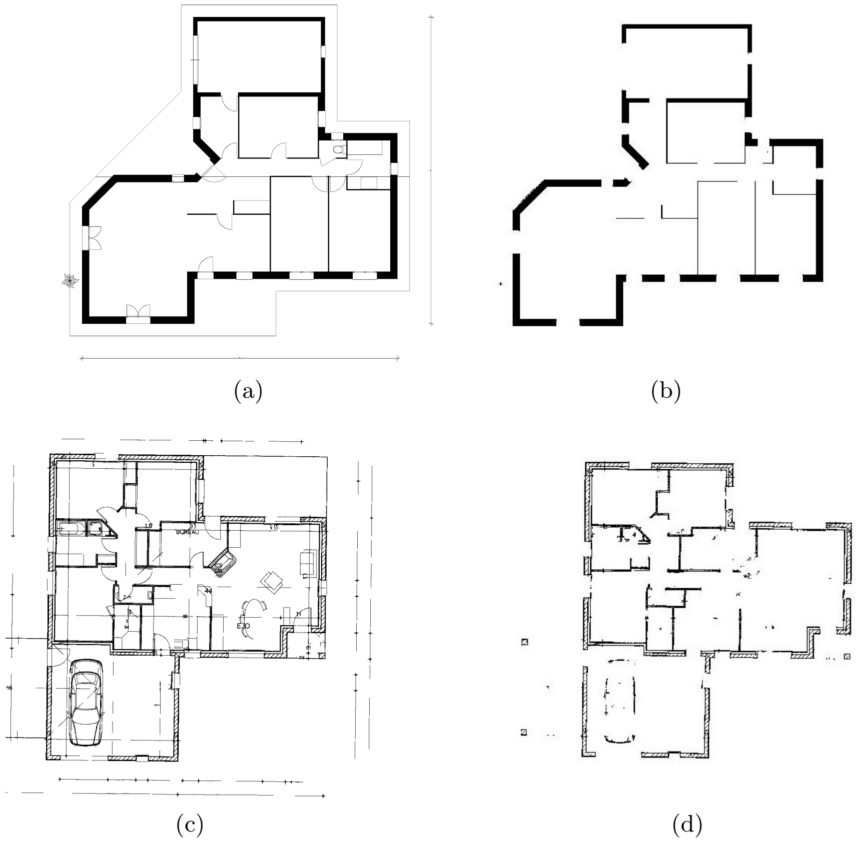

Fig. 2: Qualitative results for wall segmentation in both datasets. (a) and (c) Plan examples of *Dataset-1* and *Dataset-2* respectively. (b) and (d) walls segmented using the best system configuration for each dataset.

3 Experiments

Even though wall detection is a fundamental process in floor plan interpretation, it can not be found in the literature any work that gives a quantitative evaluation of this task. Moreover, the lack of public datasets in architectural drawings provoke that the performance of our approach could only be compared with our previous work in [1].

In this section we describe the experimental setup used to evaluate the results obtained by our approach. Firstly, the datasets used for wall segmentation are presented. Finally, the evaluation protocol followed to evaluate the performance of our system is explicated.

3.1 Floor plan Dataset

In order to evaluate our approach, we use the two datasets used in [1]. Both collections were specifically created to perform wall segmentation and they contain plans with complete different graphical notations and resolutions. They were manually labeled for the classes *Wall* and *Background*. These datasets will be made publicly available soon. Actually, they have already been used to perform floor plan interpretation in [6].

- **Dataset-1** is a collection of 90 real architectural floor plan drawings of high resolution, see figure 2a. Both, interior and exterior walls are modeled by black lines of different thickness. The dataset is split in two subsets: the validation-set and test-set. The former contains 30 plans and is used for parameter validation, e.g. patch-size or number of learning samples, using a 5-fold cross-validation. The test-set contains the rest of the plans, and using a 10-fold cross-validation is used to evaluate our system.
- **Dataset-2** contains 10 real floor plans documents at low-resolution, see figure 2c. The notation for walls varies whether they are exterior or interior. Exterior walls are modeled with hatched lines meanwhile interiors are modeled with dotted lines. Our intention is to confirm whether the system is capable to segment walls in plans with a completely different graphical convention. Due to the small amount of plans in this dataset – only 10 – all the documents have been used for training and testing following a Leave-One Out strategy. Moreover, these plans contain images at different resolutions which allows us to evaluate the performance of the system by introducing the wall thickness normalization methodology explained in section 2.1.

3.2 Evaluation Protocol

The protocol chosen for evaluating our system is completely the same used in [1]. We evaluate our method at pixel level but only considering in the score those pixels which are black in the original binary image, as only black pixels convey relevant information for segmentation. All the results in the experiments are expressed using the Jaccard Index *JI*. This index takes values between 0 and 1 and the higher it is, the better segmentation is performed.

$$JI = \frac{TruePos}{(TruePos + FalsePos + FalseNeg)}. \tag{3}$$

4 Results and discussion

In essence, our system is influenced by two general parameters: the grid topology (GT), and the method used to describe patches (D). In the first case, table 1 shows that overlapping grid behaves better than the non-ovelapped. The main reason is that using overlapping patches, pixels which belong to wall boundaries are better represented and therefore, more respected. Contrarily, using

Table 1: System behavior regarding grid topology in dataset-2.

Dataset	**GT**	PS	D	LS	ϕ_o	JI
Dataset-2	Non-overlapped	15×15	PCA	1000	-	0.7316
Dataset-2	Overlapped	15×15	PCA	1000	3	**0.7981**

Table 2: System behavior regarding descriptor in dataset-2.

Dataset	GT	PS	**D**	LS	JI
Dataset-2	Non-overlapped	15×15	PID	7500	0.7206
Dataset-2	Non-overlapped	15×15	PCA	7500	0.7316
Dataset-2	Non-overlapped	15×15	BSM_8	7500	**0.7441**

a non-overlapped grid, some boundaries are lost because pixels in these areas can easily fall into patches which mostly represent background. In the case of patch-descriptors, table 2 shows the performance of the system using different approaches to describe patches. BSM, that can be seen as a local blurring of the image in each patch, describes walls better, and also it can characterize better high intra-class variability, as it is the case for walls in dataset-2. In addition to that, as it can be seen in table 3, the proposed image normalization process improves the global performance of the system because it reduces the large variability in patch appearance when plans have different resolutions. Lastly,it is worth saying that the number of subregions selected while using BSM descriptor (BSMreg), the size of the patches (PS), the number of overlapped pixels (ϕ_o) in the overlapped-grid, and the SVM learning samples (LS) have been learned experimentally in the system validation process.

Up to this point, we have proved that the best system configuration includes the image-size normalization, an overlapping grid and BSM as patch-descriptor. The next step is to analyze the behavior of the system –in its best configuration– when different classification strategies are used. With this aim, table 4 shows the best system performances using SVM classifier (SVM-WD) and vocabulary based classifier ([1]+iNorm+BSM). Both methods are compared with the baseline approach proposed in [1].

According to the results, SVM-WD performs very similar to the baseline method in both datasets, and closely to the [1]+iNorm+BSM in dataset-1. Moreover, SVM-WD is three times faster in testing-time than using a vocabulary-based classifier. This yields to consider the use of SVM as a good alternative

Table 3: System behavior regarding image normalization in dataset-2.

Dataset	GT	PS	**D**	LS	JI
Dataset-2	Non-overlapped	15×15	BSM_8	7500	0.7357
Normalized Dataset-2	Non-overlapped	15×15	BSM_8	7500	**0.7441**

Table 4: Best wall-segmentation results for (SVM-WD) and [1] $+iNorm+BSM$ using the best system configurations. Both systems are compared with the baseline approach ([1]). DS is the codebook size used in [1].

Method	Dataset	GT	PS	D	LS	DS	ϕ_o	***JI***
[1]	*Dataset-1*	Overlapped	8×8	PCA	-	100	4	**0.9673**
	Dataset-2	Overlapped	20×20	PCA	-	2000	5	**0.8241**
SVM-WD	*Dataset-1*	Overlapped	15×15	BSM_8	50000	-	3	**0.9667**
	Dataset-2	Overlapped	15×15	BSM_8	7500	-	3	**0.8233**
[1]+$iNorm + BSM$	*Dataset-1*	Overlapped	10×10	BSM_8	-	100	5	**0.9714**
	Dataset-2	Overlapped	18×18	BSM_{16}	-	2000	3	**0.8612**

when datasets obey to the same characteristics as dataset-1. On the other hand, [1]+iNorm+BSM highly outperforms the SVM-WD for the challenging dataset-2 (from 0.82 to 0.86) and slightly improve the classification in dataset-1. This concludes that a vocabulary based classifier must be used when performance is more critical than time, as it is usually the case of floor plan interpretation methods.

5 Conclusion

This paper presents a notation-invariant method to detect and segment walls in floor plans. This approach, which is an evolution of the previous wall detector presented by the authors in [1], is a statistical patch-based detector that escapes from the traditional structural techniques based on vectorization. For that reason, our method only needs to be retrained for every new notation, instead of being reformulated as the majority of the state-of-the-art techniques do.

Three different alternatives from our previous work are analyzed. Firstly, since floor plans can be found at different resolutions, an unsupervised preprocess to normalize the size of all input images is applied. Secondly, after dividing the images into patches following two different strategies – squared-rigid grid and overlapped grid –, the influence that different patch-description techniques have into the global system performance is studied. The patch descriptors tested are PID, PCA and BSM, being BSM that one that better encapsulates the information from patches. Finally, patch classification is performed by a Support Vector Machine. Experiments on two datasets with different notations show that using these alternatives – an SVM classifier along with image normalization and BSM features – yields to a very similar accuracy to the baseline approach presented in [1], but being a big deal faster in testing time. In addition have also proved that joining the vocabulary-based classification used in [1] with the image normalization and BSM features as presented in this paper leads to the best configuration of the system, which highly outperforms the results of the original paper.

Acknowledgment

This work has been partially supported by the Spanish projects TIN2009-14633-C03-03, TIN2011-24631, a CONSOLIDER-INGENIO 2010(CSD2007-00018), an Eureka project TSI-020400-2011-50 and by a research grant of the Universitat Autonoma de Barcelona (471-02-1/2010).

References

[1] de las Heras, L. Mas, J., Sánchez, G., Valveny, E.: Wall Patch-Based Segmentation in Architectural Floor. In: Int. Conf. on Document Analysis and Recognition. 1270–1274 (2011)
[2] Dosch, P., Tombre, K., AhSoon, C., Masini, G.: A complete system for the analysis of architectural drawings. In: International Journal on Document Analysis and Recognition. vol 3, issue 2, 102–116 (2000)
[3] Juchmes, J., Leclercq, P., Azar, S.: A Multi-Agent system for the Interpretation of Architectural Sketches. In: Eurographics Workshop on Sketch-Based Interfaces and Modeling. 53–61 (2004)
[4] de las Heras, L., Sánchez, G.:And-Or Graph Grammar for Architectural Floor Plan Representation, Learning and Recognition. A Semantic, Structural and Hierarchical Model. In: Pattern Recognition and Image Analysis. 17–24 (2011)
[5] Macé, S., Locteau, H., Valveny, E., Tabbone, S.: A system to detect rooms in architectural floor plan images. In: Proc. IAPR Int. Work. on Document Analysis Systems. 167–174 (2010)
[6] Ahmed, S. Liwiki, M., Weber, M., Dengel, A.: Improved Automatic Analysis of Architectural Floor Plans. In: Int. Conf. on Document Analysis and Recognition. 864–869 (2011)
[7] Larlus, D., Verbeej, J., Jurie, F.: Category Level Object Segmentation by Combining Bag-of-Words Models with Dirichlet Processes and Random Fields. In: Int. Journal on Computer Vision. 238–253 (2010)
[8] Burns, T.J., Corso, J.J.: Robust unsupervised segmentation of degraded document images with topic models. In: IEEE Conf. on Computer Vision and Pattern Recognition. 1287–1294 (2009)
[9] Escalera, S., Fornés, A., Pujol, O., Radeva, P., Sánchez, G., Lladós, J.: Blurred Shape Model for Binary and Grey-level Symbol Recognition. In: Pattern Recognition Letters, vol. 30, 1424–1433 (2009)
[10] Otsu, N.: A threshold selection method from gray-level histogram. In: IEEE Trans. Syst. Man Cybern. vol 9, 62–66 (1979)
[11] Tombre, K., Tabbone, S., Pélissier, L., Lamiroy, B., Dosch, P.: Text/Graphics Separation Revisited. In Document Analysis Systems V. 615–620 (2002)
[12] Chang, C., Lin, CJ.: LIBSVM: A library for support vector machines. In: ACM Transactions on Intelligent Systems and Technology. vol. 2, 27 (2011)
[13] Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. In: IEEE Trans. Pattern Analysis and Machine Intelligence. vol. 20, 226–239 (1998)