Text Recognition and Classification in Floor plan Images

Jason Ravagli, Zahra Ziran, Simone Marinai
Dipartimento di Ingegneria dell'Informazione (DINFO)
University of Florence - Florence, Italy
jason.ravagli@stud.unifi.it, zahra.ziran@unifi.it, simone.marinai@unifi.it

***Abstract*—In this paper we present a method for text recognition in floor plan images. In particular, we are concerned about locating, reading, and categorizing text inside floor plan images to obtain information about the building. Furthermore, the aim of this paper is to compare traditional text detection methods, based on image processing techniques, with recent approaches relying on convolutional neural networks. To improve results we combined several methods outperforming the original ones. Text regions are also classified in four semantic classes according to their purpose. Two datasets with different features, including quality and size, were considered in the experiments performed.**

***Keywords*-Floor plan analysis, text recognition in graphics, convolutional neural networks**

I. Introduction

Text recognition in graphical documents is an important task in document image analysis. In particular, textual information in engineering drawings, like floor plans, is momentous for further analysis, specifically when the semantics of rooms should be detected. From a broader point of view, reading text in images is a relevant topic in computer vision with several applications in automatic digitization of documents, real-time multilanguage translation, augmented reality, support to blind people, etc.

The work presented in this paper is related to a broader project whose aim is to allow visually impaired users to access graphical information [10]. By considering the recognized text it is possible to provide to visually impaired users useful information about rooms functions and size. Hence, by extracting and gathering primary information obtained from the floor plan, we can gain information about the entire building.

Text detection methods are usually based on connected components (CCs) [14], or sliding windows [8]. Fletcher and Kasturi [6] proposed one of the first methods based on the analysis of connected components and their mutual position. One weakness of this method is that it does not cope with text touching graphics. Tombre et al. [13] proposed an improved approach able to separate text touching graphical parts. Connected component methods divide pixels into characters, then group these into words. For example, Epshtein et al. [5] address text detection in natural images considering characters as CCs of the Stroke Width (SWT) transform and then propose an image operator to find the value of the stroke width. The SWT is used to group pixels into letter candidates. Li and Lu [9] adopt a stroke width based text detection approach and propose unique contrast-enhanced Maximally Stable Extremal Region (MSER) to extract character candidates. Gonzalez et al. [5] efficiently combine MSER and a locally adaptive thresholding. These approaches do not use neural networks for text detection and consider only image processing techniques.

The field of scene text detection has entered into a new era recently and deep neural network based algorithms [7], [8] have become the mainstream. EAST [16] is one CNN-based text detection method that is trained for direct forecast of the existence of text instances in full images. The model is a fully-convolutional neural network that outputs dense per-pixel predictions of words or text lines. The model omits intermediate steps such as candidate proposal, text region formation, and word partitioning. The post-processing only includes thresholding and NMS on predicted geometric shapes. Several approaches are used for text detection but most authors employ a bottom-up pipeline [3] [15], [7]. These methods commonly explore low-level features to distinguish text candidates. However, they are not robust because identify individual strokes or characters without context information. One solution to realize the accurate text localization and to address these problems is the CTPN method proposed by Tian et al. [12] to extract strong deep features for detecting text information directly in convolutional maps and developing an anchor mechanism that jointly predicts location and text/non text score for each proposal. They also propose an in-network RNN layer that connects sequential text proposals to explore meaningful context information.

In the rest of the paper we first describe the proposed method for detecting, recognizing, and classifying text in floorplans images and then analyze the experiments performed

II. The proposed method

The overall processing pipeline is summarized in Fig. 1. Given a floor plan image in input, three files are generated as output. An XML file containing the information about text regions; the input image with annotations about results (regions of text, recognized text and its type); the input image cleaned up from the detected text. The system is

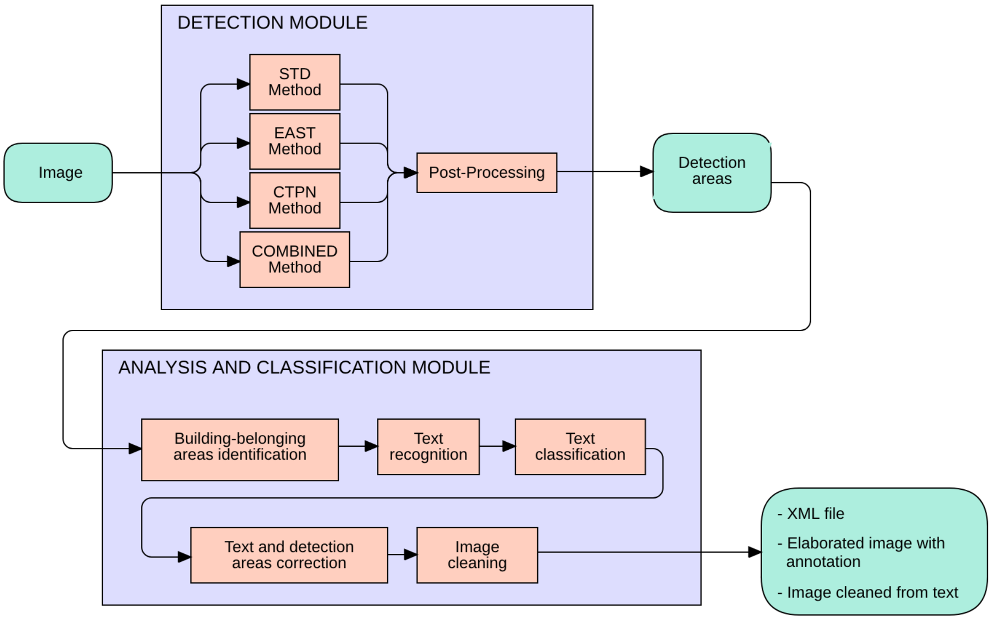

DETECTION MODULE
STD Method
EAST Method
CTPN Method
COMBINED Method
Post-Processing
Image
Detection areas
ANALYSIS AND CLASSIFICATION MODULE
Building-belonging areas identification
Text recognition
Text classification
Text and detection areas correction
Image cleaning
- XML file
- Elaborated image with annotation
- Image cleaned from text

Figure 1. System architecture.

composed by two main modules: text detection and text analysis and classification.

A. Text detection module

The text detection module takes an image as input and returns a list of rectangles around detected text regions as output. Four different detection procedures can be used: STD, CTPN, EAST, and COMBINED.

STD is implemented starting from one library based on MSER and SWT [11], [2], EAST is based on its OpenCV implementation, and for CTPN we used the library in [1]. COMBINED combines the results of the other three. Starting from an input image, the libraries in their original form calculate a set of points identifying the text areas (EAST and CTPN) or modify the input image to highlight the located text (STD). In order to be integrated in our project, the libraries have been modified to produce a list of structured data (rectangles identifying text). Moreover, we added functionalities to each detection method to improve the performance of the libraries as we will discuss later. Regardless of the detection procedure used, the module includes a final post-processing phase where rectangles are refined to better locate text within them and facilitate its processing by the analysis module.

1) STD detection method: The original library [2] is used in combination with Tesseract to discard regions containing non-readable text. The library detects single words of text returning a rectangle for each of them. However, in our work near, aligned and semantically correlated words need to be grouped into an unique text object. A suitable post-processing is carried out to merge detection areas by comparing their mutual distance and alignment.

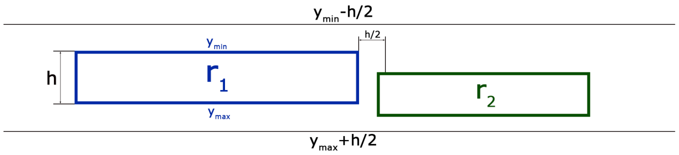
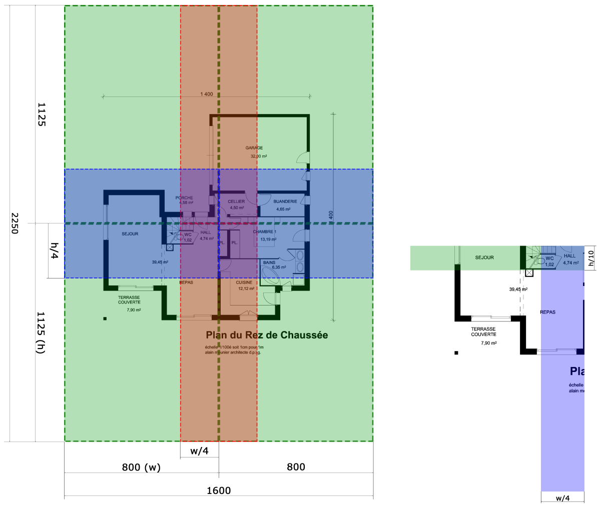


Figure 2. Example of rectangles respecting alignment and nearness conditions and therefore merged because they identify the same text object.

All possible pairs of rectangles (r_1, r_2) are considered for merging by checking their alignment (Fig. 2). Let h be the height of the taller rectangle (r_1), y_{min} the coordinate of the upper edge of r_1 and y_{max} the position of the lower edge of r_1. r_1 and r_2 are aligned if r_2 is entirely contained in the interval $[y_{min} - h/2,\ y_{max} + h/2]$. We allow an offset of $h/2$ at the top and bottom to deal with ascenders and descenders in r_2. If two rectangles are aligned we then check their horizontal distance: r_1 and r_2 are near if their distance is lower than $h/2$.

Pairs of near and aligned rectangles are then merged into the minimum enclosing rectangle defined by (x_{min}, y_{min}), (x_{max}, y_{max}) that corresponds to the minimum and maximum x and y coordinates occupied by the two rectangles.

2) CTPN detection method: The CTPN library [1] can only process images with a maximum size of 1200 pixels. For larger images the library automatically scales the dimensions before performing the detection, thus introducing a loss of information. When the image is much larger the



(a) Split into sub-images (b) Transition areas

Figure 3. (a) Large images split into sub-images (green area). Red and blue areas are the horizontal and vertical transition sub-images, respectively. (b) The horizontal transition area (blue) and the vertical one (green) of a sub-image.

loss of information causes a very poor detection. For this reason large images are first split in sub-images, which are processed separately by CTPN. The original image is split into equal parts both in height and in width so that the dimensions of each sub-image does not exceed the threshold (Fig. 3(a)).

Unfortunately, the image split gives rise to new problems that need to be addressed. Text objects lying across of images will be detected as two separated objects; in other cases the text area might be wrongly computed by the library, which detects erroneously text too close to the image borders. To address these problems, additional sub-images are generated, centered in the transition areas, on which the detection will be performed as well (Fig. 3(a)).

After executing CTPN in all the sub-images (eight in Fig. 3(a)) we have a list of rectangles returned by CTPN. Some rectangles can overlap because the same text is recognized in overlapped regions. These redundant detections are addressed in the post-processing. In addition, there are also rectangles identifying parts of the same text object recognized in different sub-images. These rectangles are addressed as follows. For each sub-image (with size (h, w)) its vertical transition areas are defined the two vertical rectangles along the vertical border of the sub-image with width equal to $w/4$ (blue rectangle in Fig. 3(b)). The horizontal transition areas are identified in a similar way considering the horizontal border and height equal to $h/10$ (green rectangle in Fig. 3(b)). The algorithm merges aligned rectangles lying on adjacent transition areas (that share an edge but belong to different sub-images). The definition of aligned rectangles and the features of the merged ones are the same used when merging side by side rectangles in the STD detection method. As we mentioned above, this analysis of the transition areas is needed because CTPN struggles to accurately identify text very close to the image borders.

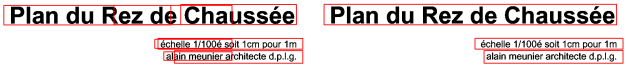

Plan du Rez de Chaussée
échelle 1/100é soit 1cm pour 1m
alain meunier architecte d.p.l.g.
Plan du Rez de Chaussée
échelle 1/100é soit 1cm pour 1m
alain meunier architecte d.p.l.g.

Figure 4. Text detection before (left) and after (right) the post-processing.

3) EAST detection method: EAST accepts as input only images having width and height multiples of 32. We therefore add a suitable padding of white pixels to those images that do not respect this constraint. EAST detects single words of text like STD. Therefore, also in this case, after the detection we perform the algorithm for merging near and aligned rectangles described above.

4) Combined detection method: after applying all three methods and obtaining three distinct lists of detected rectangles the combined method performs a voting process to discard false positives. Each rectangle r of a list is compared with all the rectangles of the other two lists. If there is almost one rectangle coinciding with r, then r is added to the final detection list, otherwise it is discarded. Furthermore, each time two rectangles are found to be coincident, they are merged to refine the detection area. In this case, two rectangles are coincident if their IoU (Intersection over Union) is greater than 50%.

5) Post-processing: The post-processing performs two steps to refine the detection areas (Fig. 4). In the first step, the rectangle list is cleaned to eliminate redundant detections (mostly generated by the analysis on intersection areas between different sub-images in CTPN). For each pair of rectangles, we calculate their IoU: if it is greater than 40% they are merged. Otherwise we check if one is contained for 70% in the other, and in this case the smaller rectangle is discarded as the two rectangles are detecting the same object.

In the second step aligned and overlapped rectangles are merged. These are generated by text objects detected as two split objects for several reasons: splitting into sub-images, presence of noise or other objects superimposed to the text, or to the incorrect evaluation by the library. The concept of alignment is the same presented in STD and EAST detection methods.

B. Text analysis and classification module

This module gets the floor plan image and the rectangles calculated by the detection module and computes additional information for each text object: the text type; the recognized text; a list of rectangles for all the characters in the area. The module also returns a floor plan with the text erased.

1) Detection of areas belonging to buildings: we calculate the image CCs and identify a CC corresponding to the

Dataset	Images	Text Objects	Description	Size (m^2)	Size (ft^2)	Generic Text
CVC-FP	90	2779	1261	925	0	593
Flo2Plan	64	1632	793	0	376	453

Table I
IMAGES AND NUMBER OF LABELED TEXT OBJECTS IN EACH DATASET.

building if the CC is not contained in any other CC and its area is larger than 10% of the whole image. The convex hull is then calculated from each non-discarded CC.

2) Text recognition: each text area is cropped and passed to Tesseract for recognition, computing also the position of words and single characters. After recognition, we check the validity of the text discarding empty strings or not containing letters or numbers.

3) Text classification: each text box is classified as follows. If the text object does not intersect with a building (the CC's convex hull computed in step 1) it is classified as generic text. Otherwise, we examine the recognized text: if there are more letters than numbers then it is classified as room description, otherwise, it is classified as room size choosing room size measured in m^2 or room size measured in ft^2 according to the suffix of the recognized text (e.g. a room size measured in ft^2 ends with single apex, double apex or ”ft”).

4) Text correction and detection area refinement: the text recognition may not be perfect (e.g. because of noise in the area or poor image quality). We correct the text for the objects classified as room description using some heuristics. Each word is compared with a dictionary of valid words and then replaced with the closest according to the edit distance. In case of words with the same edit distance, the word with the highest frequency is chosen. Text corresponding to a room size is corrected by removing from the string non valid characters (digits, punctuation marks, letters, and symbols for units of measure). Since we are not able to make assumptions about the correctness of their content, no correction is made on generic text objects. The area of each text object is refined considering the rectangles of single characters. In the last step the rectangles surrounding each character are filled with white.

III. EXPERIMENTAL RESULTS

We first present the datasets used in the experiments, explaining their features and peculiarities. Then, the detection and classification results are shown separately. Finally, a qualitative analysis of the results is carried out.

For the experiments, we used two datasets (Table I) CVC-FP and Flo2Plan, respectively. CVC-FP [4] contains 90 high quality floor plan images with an uniform style and a well-readable text. Flo2Plan is a subset of the dataset proposed in [17] and contains 64 floor plans downloaded from Internet with very different styles and resolution. Both datasets have been manually labeled to provide information about text objects.

A. Quantitative results

In these tests we compared the performance of the proposed methods for text detection and text classification. We ignored the accuracy of the read text, because it mainly depends on Tesseract and on the coverage of the dictionary used to a lesser extent.

1) Performance indices and detection baseline: To evaluate the detection performance we compute the Precision (P), Recall (R) and F_1 Score indices. To compute these values, the detection of a text object is considered correct if its IoU with the ground truth is larger than 65%.

We first performed the detection to the CVC-FP and the Flo2Plan datasets using the STD, EAST and CTPN libraries in their original version, without modifications. The results, together with the average execution time per image (on a Mac OSX PC with a 7th generation Intel i5 CPU without GPU), are shown in Table II. Obviously, exploiting a GPU the CTPN and EAST execution times would be considerably reduced. As we can see, the performance are very poor in general. For STD and EAST, this is mostly due to the fact that both libraries detect single words instead of grouping neighboring words on the same text line under a single text object, like CTPN does. Conversely CTPN shows significant limits in the analysis of poor quality images (Flo2Plan dataset) and large images (CVC-FP dataset) because of the image scaling and subsequent information loss.

2) Detection performance: In Table II we also report P, R, F_1 Score, and average execution time of the four detection methods. The execution time includes the time required for the classification and analysis module. The execution time grows linearly with the number of text objects in the image, therefore it can significantly vary from image to image.

Evaluating the results we must first notice that no method can detect vertical or curved text. With the clean and high resolution images of CVC-FP the CTPN method obtains very good results ($P = 85\%$ $R = 82\%$), significantly better than those obtained with the original library. The higher execution time is due to the sub-images splitting. Its performance however drop dramatically if applied to the Flo2Plan dataset. The noise, the compression and the poor resolution of these images strongly affect the analysis by the CTPN library. On the other hand on the Flo2Plan dataset the EAST method achieves good results ($P = 59\%$ $R = 65\%$). Furthermore it maintains low execution times. This is possible because EAST has been developed for real time detection in video streams. This result also confirms that text detection methods based on CNNs perform a better detection than STD not only for scene images but also for graphical documents.

	CVC-FP				Flo2Plan			
Original Library	P	R	F_1	**Time**	P	R	F_1	**Time (s)**
STD	14.60%	19.78%	16.80%	21.10	34.26%	23.72%	28.03%	39.21
EAST	17.82%	33.58%	23.28%	16.28	33.11%	43.17%	37.48%	2.34
CTPN	25.62%	5.24%	8.70%	4.52	25.85%	22.43%	24.02%	5.19
Modified Library	P	R	F_1	**Time**	P	R	F_1	**Time (s)**
STD	58.05%	60.00%	59.01%	35.04	56.34%	32.93%	41.57%	45.61
EAST	58.61%	64.90%	61.59%	31.94	61.06%	60.16%	60.61%	13.26
CTPN	84.85%	81.74%	83.27%	103.25	50.36%	39.10%	44.02%	12.27
COMBINED	86.11%	81.05%	83.50%	138.65	68.13%	37.45%	48.33%	52.01

Table II
DETECTION PERFORMANCE OF THE METHODS ON THE TWO DATASETS.

	CVC-FP			Flo2Plan		
Text type	P	R	F_1	P	R	F_1
Room description	97.54%	98.62%	98.08%	79.30%	84.57%	81.85%
Room size (m^2)	98.55%	97.79%	98.17%	-	-	-
Room size (ft^2)	-	-	-	28.13%	23.70%	25.73%
Generic Text	95.43%	93.46%	94.43%	28.30%	28.65%	28.47%

Table III
CLASSIFICATION PERFORMANCE ON THE TWO DATASETS.

The combined method shows good performance on both datasets, overcoming the other methods in term of Precision. The Recall is very low (37%) on the Flo2Plan dataset; this is an expected result considering the voting system it adopts which increases the number of false negatives on images where the other methods have some difficulties.

3) Classification performance: In Table III we report the performance for text classification when using the COMBINED method to perform the detection. Although in the CVC-FP dataset the classification module obtains good results, in the Flo2Plan dataset the performance are very poor. The qualitative analysis presented in the next section attempts to analyze some possible causes.

B. Qualitative Results

We show in Fig. 7 the results on two images from the two datasets. In Fig. 5 we compare CTPN and EAST applied to an image of the Flo2Plan dataset. As can be seen, when the text is not really sharp, the recognition by Tesseract is incorrect even with a correct detection. Images in Flo2Plan have different styles of rooms and generic annotations inside the buildings. As we can see in Fig. 6, the method incorrectly classifies the text as room description (blue boxes) in both cases. In Fig. 6(a) the room sizes have a format different from that observed in the other floorplans: letters prevail on digits and the method classifies the text object as room description. Instead the floor plan in Fig. 6(b) contains also names for the furniture (D/W is drier and washing machine), which are identified as room description instead of generic text.

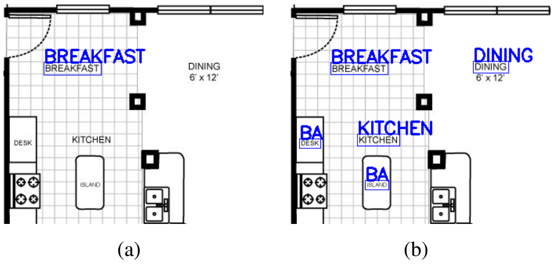


Figure 5. Comparison between the CTPN (a) and the EAST (b) on an image of the Flo2Plan dataset.

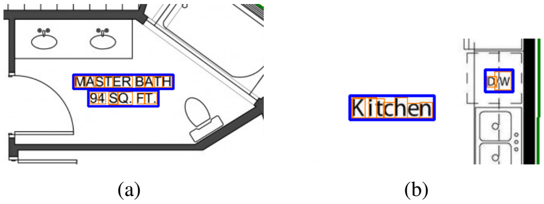


Figure 6. Example of wrong text classification in the Flo2Plan dataset.

IV. CONCLUSIONS

We address text extraction, classification, and recognition in floor plan images and compare traditional approaches and methods based on deep learning. Performance of the original methods are significantly improved thanks to suitable pre and post processing steps specifically implemented to address this task.

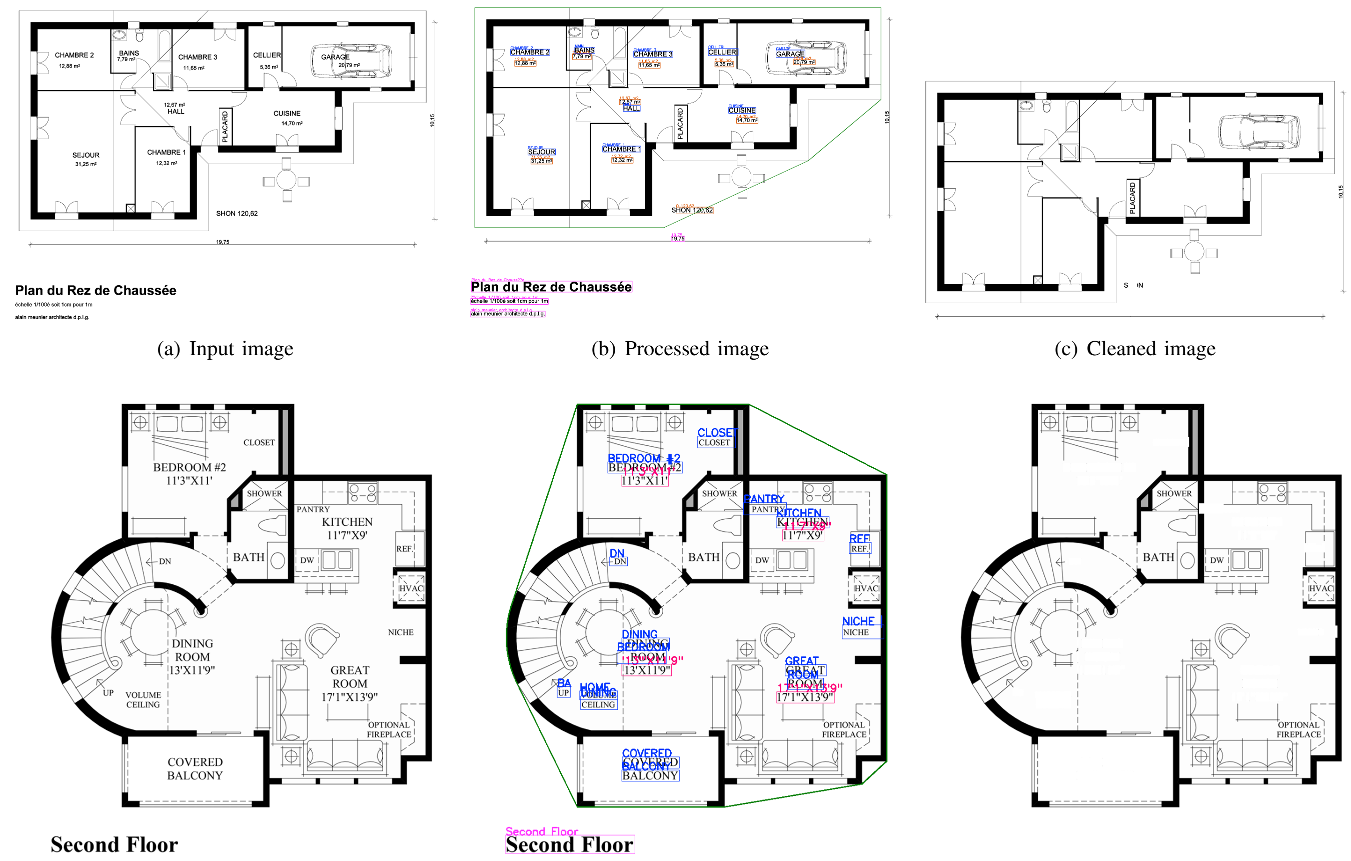
REFERENCES

[1] CTPN. https://github.com/eragonruan/text-detection-ctpn.

[2] Text detection with MSER and SWT. https://github.com/azmiozgen/text-detection.

[3] M. Buta, L. Neumann, and J. Matas. Fastext: Efficient unconstrained scene text detector. In *ICCV*, pages 1206–1214, 2015.

[4] Lluís-Pere de las Heras, Oriol Ramos Terrades, Sergi Robles Mestre, and Gemma Sánchez. CVC-FP and SGT: a new database for structural floor plan analysis and its groundtruthing tool. *IJDAR*, 18(1):15–30, 2015.



(a) Input image
(b) Processed image
(c) Cleaned image

(d) Input image
(e) Processed image
(f) Cleaned image

Figure 7. Processing of an image of the CVC-FP (top) and Flo2Plan (bottom) datasets. Blue: text classified as room description; Red: text classified as room size (m^2); Dark pink: text classified as room size (ft^2); Magenta: text classified as generic.

[5] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. In *CVPR 2010*, pages 2963–2970, 2010.

[6] L. A. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE TPAMI*, 10(6):910–918, Nov 1988.

[7] Weilin Huang, Yu Qiao, and Xiaoou Tang. Robust scene text detection with convolution neural network induced MSER trees. In *ECCV*, pages 497–511, 2014.

[8] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. In *ECCV*, pages 512–528, 2014.

[9] Yao Li and Huchuan Lu. Scene text detection via stroke width. In *Proc. ICPR*, pages 681–684, 2012.

[10] Anuradha Madugalla, Kim Marriott, and Simone Marinai. Partitioning open plan areas in floor plans. In *ICDAR*, pages 47–52, 2017.

[11] Azmi Can Ozgen, Mandana Fasounaki, and Hazim Kemal Ekenel. Text detection in natural and computer-generated images. In *26th Signal Processing and Communications Applications Conference*, pages 1–4, 2018.

[12] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *ECCV*, pages 56–72, 2016.

[13] Karl Tombre, Salvatore Tabbone, Loïc Pélissier, Bart Lamiroy, and Philippe Dosch. Text/graphics separation revisited. In Daniel Lopresti, Jianying Hu, and Ramanujan Kashi, editors, *DAS*, pages 200–211, 2002.

[14] C. Yi and Y. Tian. Text string detection from natural scenes by structure-based partition and grouping. *IEEE TIP*, 20(9):2594–2605, Sep. 2011.

[15] X. Yin, W. Pei, J. Zhang, and H. Hao. Multi-orientation scene text detection with adaptive clustering. *IEEE TPAMI*, 37(9):1930–1937, Sep. 2015.

[16] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. EAST: an efficient and accurate scene text detector. In *CVPR 2017*, pages 2642–2651, 2017.

[17] Zahra Ziran and Simone Marinai. Object detection in floor plan images. In *ANNPR*, volume 11081 of *LNCS*, pages 383–394. Springer, 2018.