Data-driven Floor Plan Understanding in Rural Residential Buildings via Deep Recognition

Zhengda Lu[a,b], Teng Wang[a,b], Jianwei Guo[b,a], Weiliang Meng[b,c,a], Jun Xiao[a,*], Wei Zhang[d], Xiaopeng Zhang[b,a]

[a] *University of Chinese Academy of Sciences, Beijing 100049, China*
[b] *NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China*
[c] *The State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China*
[d] *China Architecture Design and Research Group, Beijing 100044, China*

Abstract

Automatic understanding of floor plan images is a key component of various applications. Due to the style diversity of rural housing design, the latest learning-based approaches cannot achieve satisfactory recognition results. In this paper, we present a new framework for parsing floor plans of rural residence that combines semantic neural networks with a post-processed room segmentation. First, we take case studies from typical residential buildings in China's rural areas and provide a novel image dataset, called *RuralHomeData*, containing 800 rural residence floor plans with accurate man-made annotations. Based on the dataset, we propose a new deep learning-based recognition framework using a joint neural network to predict the geometric elements and text information on the floor plan simultaneously. Our insight is that walls and openings (doors and windows) are the basic elements corresponding to the room boundary that a closed 1D loop must form a certain room. Then the semantic information (e.g., the room function) of room regions can be obtained through text detection and identification. Furthermore, we use the MIQP algorithm to divide the area containing multiple room type texts into multiple room areas. Finally, the input floor plan can be transformed into a room layout graph with room attributes

*Corresponding author
Email address: xiaojun@ucas.ac.cn (Jun Xiao)

and adjacent relationships. The proposed algorithm has been tested on both urban and rural datasets, and the experimental results demonstrate our efficiency and robustness compared with the state-of-the-art methods.

Keywords: Floor plan understanding, rural residence, neural networks.

1. Introduction

A floor plan is the most fundamental architectural diagram used to show the layout of rooms in a building by the spatial configurations between its elements (*e.g.*, windows, doors, walls) and room type texts. In the past decades, floor plan understanding remains an active research topic in the field of pattern recognition and document analysis.

Specifically, given an input floor plan image, not only the individual floor plan elements as well as their geometric properties (such as wall length, window size) are detected, but also the meaningful semantic units associated with high-level information (such as room function) can be identified. Early works focus on analyzing floor plans based on low-level image processing, such as line detection through Hough transform [1], graphical symbol recognition by a bag-of-words model [2]. However, the performance of these methods is largely limited by the representation power of the hand-crafted features.

Recently, several data-driven techniques [3, 4, 5] based on the convolutional neural networks (CNNs) have achieved promising results. However, the public datasets [6, 3, 7, 8] they used are collected from the apartments of urban residence, where the complexity of the floor plans is limited, *e.g.*, the types of rooms are relatively few and the graphic elements are regular straight lines. By contrast, the architecture of rural residences is more complex, and their structure of rooms is related to the local environment, production mode, and residence lifestyle. Therefore, buildings in different rural areas have different geometric structural layouts and functional space rooms as shown in the figure 1 (top) for the two rural residence floorplans from different provinces in China. In general, these learning methods cannot achieve satisfactory prediction results on the ru-

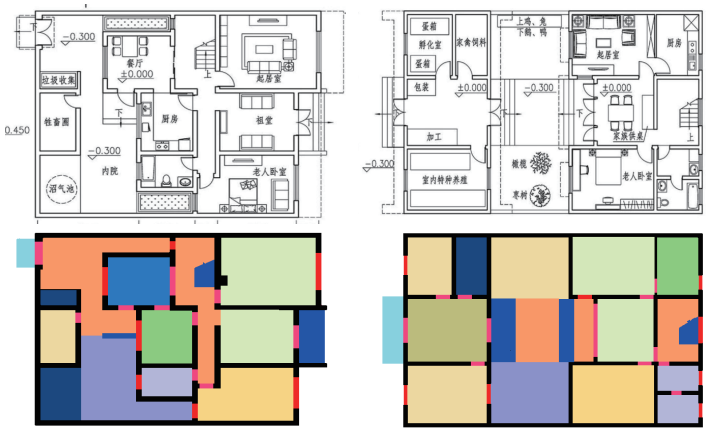


Figure 1: Top: the buildings in different rural areas have different structural layouts and functional space regions. Bottom: our parsing results by elements recognition and room segmentation.

ral dataset due to various irregular geometric elements (including arcs, diagonal walls) and a large number of room types.

In this paper, we propose a new deep learning-based recognition framework with an emphasis on better-understanding floor plans of rural residence. To this end, we have first built a new dataset containing 800 real rural residential floor plans from the China region, in which the annotation ground truth of each floor plan is manually labeled. Compared to previous work [4, 5], we not only labeled the geometric elements and the room semantic information but also labeled the room type text information. Because there are many room types in the rural floor plan, the room prediction accuracy is far beyond satisfactory. Our algorithm is based on two intuitive observations: first, walls and openings (doors and windows) are the graphical elements corresponding to the room boundary, and a closed 1D loop forms a certain room, where an adjacency between two rooms means they are connected by sharing walls or openings; second, the semantic information (e.g., the room function) of room regions can be obtained through room type text detection and identification. In our approach, we first propose a joint deep neural network to address the graphical elements recognition and

text detection of floor plans simultaneously. Then we recover the geometric and semantic structure of a floor plan by optimizing elements prediction results and using a MIQP-based room parametrization and segmentation method. Finally, we construct a room layout graph with room attributes and adjacent relationships from the input floor plan. In summary, our main contributions include:

- We design a joint neural network that simultaneously performs two tasks: recognizing basic graphical elements and detecting the room type text. Followed by room segmentation and layout generation, our framework is *simple and effective* to handle a variety of floor plan images. We also provide a splitting module to handle the input with arbitrarily large size.
- To effectively train our networks, we provide a novel and high-quality image dataset, called RuralHomeData, containing 800 real-world floor plans of rural residence with man-made detailed annotations. To the best of our knowledge, we are the first to analyze the floor plans of rural residence.

2. Related work

Hand-crafted floor plan parser. Early works on floor plan understanding rely on a bunch of low-level image processing and strong heuristics. Generally, a floor plan contains graphic elements and text information. Therefore, traditional methods. [9, 10, 11, 1, 12, 13, 2, 14, 15] usually divide a floor plan into a text image and a graphic images and then extract key graphic elements from the graphic image to generate a 3D model. For example, Dosch et al. [9] analyze the architectural drawings by feature extraction and detect the basic architectural entities for the 3D building reconstruction. Or et al. [10] solve a slightly simplified problem that considers only walls, doors, and windows and convert a floor plan into the corresponding 3D description. Macé et al. [11] propose to segment the architectural floor plans into rooms based on wall and door detection by Hough transform. Ahmed et al. [1] apply a novel preprocessing method to separate texts from graphics and extract walls according to

line thickness, and the door and windows are located finally. de las Heras et al. [12] present an automatic wall segmentation method by assuming that the wall is a repetitive element and usually modeled by straight parallel lines. This method performs well on high-resolution images with different graphical styles. In their other works, they use the patch-based segmentation approach to detect walls [13], or the basic building blocks using a statistical approach where the rooms are located by the structural pattern recognition techniques [2].

At the same time, there are also some methods that consider text information to guide room type detection. Ahmed et al. [14] retrieve the meaningful room labeling by OCR and split floor plans into rooms vertically or horizontally according to the distribution of labels. Ravagli et al. [15] take the suitable pre and post-processing steps to improve the performance of text extraction, classification, and recognition in floor plan images. Besides, text information also plays an important role in many text-enriched tasks such as VQA [16], image captioning [17], text recommendation [18] and cross-domain retrieval [19], etc. Due to the complexity of these tasks, they usually need a large scale of training data, e.g., the ST-VQA [16] includes 23,038 scene images and 31,791 questions. However, in contrast to the above tasks, the number of room-type texts in floor plans is relatively limited, and we can effectively identify room type texts through OCR technology. Our proposed image splitting module further indirectly expands the training data. **R2 A2**

However, those low-level methods based on hand-crafted features are error-prone, and their generality is limited in the real world as they need to manually tune the specific parameters when dealing with different datasets.

Deep floor plan analysis. With the rapid development of deep learning technology, the models based on deep convolutional neural networks achieve state-of-the-art results in automatic floor plan analysis. Dodge et al. [20] propose to use a fully convolutional network (FCN) for wall segmentation, the Faster R-CNN for object detection, finally an optical character recognition API for estimating room sizes. Liu et al. [3] adopt a learning-based approach to extract a set of junctions and apply integer programming to encode high-level constraints. They

extracted the constraints in recovering the floor plan data from a raster image. However, this method cannot handle irregular layouts due to the assumption of the Manhattan world and uniform wall thickness. Later, Huang and Zheng [21] apply pix2pixHD in recognizing and generating architectural drawings, marking rooms with different colors, and then generating apartment plans through two convolutional neural networks. Jang et al. [22] propose an automatic method to extract the wall from a floor plan according to the indoor spatial information standard. Yamasaki et al. [4] also utilize FCN to segment floor plan images, then the semantic segmentation forms a graph model to measure the structural similarity. A recent work of [5] designs a deep multi-task neural network to predict room-boundary elements and rooms with types. However, the kernels in the context module are not efficient for the learning of spatial relations. And the pixel labels of the rooms and other elements are noisy, thus the performance is beyond satisfactory.

There are also some recent methods that focus on geometric structure and image semantic understanding. For example, Cheng et al. [23] propose the DARNet to extract polygon-based contours as building boundaries for automatic building segmentation. Some researchers focus on image understanding in different scenarios, such as multi-view image matching based on consistent affinity graph learning [24] and fashion compatibility prediction based on low-rank hypergraph regularizer multiple-representation learning [25].

Floor plan generation. Related to floor plan analysis, there are many other works with a special focus on designing and generating floor plans, i.e., determining the position and size of several rooms [26, 8]. Besides, reconstructing 3D room layouts from RGB images [27, 28, 29, 30], or RGBD streams [31] have recently gained a lot of attention from researchers. A detailed survey is out of the scope of this paper, we refer readers to the above references for an in-depth review of this area.

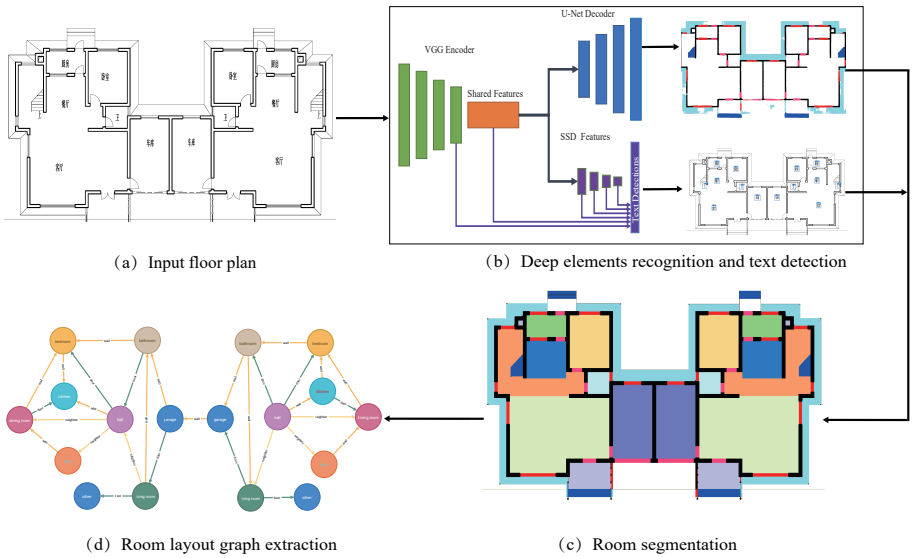


Figure 2: Overview of our framework. Given an input floor plan image (a), our deep recognition network detects the graphical elements and text locations simultaneously (b). After optimizing the prediction results, we segment the input image into different room regions (c), as well as generating a room graph layout (d) to encode the geometric and semantic information.

3. Overview

In this paper, we propose to recover the building topology with accurate room semantic information by recognizing various kinds of floor plan elements and room type texts, then constructing a room layout graph model for compactly encoding the input floor plan.

Fig. 2 illustrates the overall process of our algorithm. Given an input floor plan image, we start by recognizing the floor plan graphical and semantic information, where we have two main tasks in the network: one for predicting the room-boundary pixels of graphical elements with five labels (*i.e.*,wall, door, window, stair, and slope), and the other one for detecting the room type texts. Next, we perform a room segmentation step to extract the room semantic by optimizing the floor plan elements and get the room type by combining the text detection result. The rooms with more than one function label are split

by using the method of MIQP [26]. Finally, the room attributes and adjacent relationships of the floor plan are transformed into a room layout graph used for 3D model reconstruction.

4. Methodology

4.1. Dataset

As all of the current public floor plan datasets are sampled from the apartments of urban residence, we have provided a novel dataset for analyzing floor plans of rural residence in Chinese regions. Compared with the previous dataset, there exist a large number of different types of rooms, hollow walls, and curved windows in rural floor plans.

To create the ground-truth annotation, we collect 926 rural residence floor plans converted from original CAD files and ask humans to annotate both the semantic and geometric information for all elements that appear in the corresponding floor plan. We use Photoshop to manually label the basic architectural elements in the images including *walls, doors, windows, stairs, slopes, and texts.* Meanwhile, we use different colors to annotate room regions in each floor plan. Unlike [5], we marked different labels to separate some connected room regions, *e.g.*, living room and dining room, although they locate next to one another without walls separating them. We also perform a second check to make sure that all the semantic and geometric annotations satisfy the building standards. After dropping floor plans without semantic annotations, we collect 800 valid floor plans with different resolutions. In total, we marked 21 types of rooms including Bedroom, Living room, Kitchen, Bathroom, Restaurant, Storeroom, Stairs, Corridor, Balcony, Open space, Garage, Laundry, Study room, Railing, Clearing, Garden, Steps, Eaves, Slope, Roof, and Others. On average, each image contains 7 types, and the walls in each image are hollow, and there are 670 floor plans with curved or inclined windows. For the train-test split ratio, we randomly split it into 700 images for training and 100 images for testing.

4.2. Deep Floor Plan Recognition

Since the topological structure of a floor plan is defined by the wall and opening adjacencies while the semantic information of space regions is given by the texts, we first detect such basic graphical elements and the room type text.

Network architecture. Both deep object recognition and text detection have been well-studied in the computer vision community. However, for our task, it is redundant and time-cost to train two independent neural networks to handle graphical elements and texts respectively. To overcome this issue, we propose a new joint deep neural network with a carefully designed loss to address the graphical elements recognition and text detection simultaneously.

Our network architecture is based on a convolutional network that takes a fixed-size (512×512) image as the input and produces the mask of our defined architectural elements and the bounding boxes of room type texts. As illustrated in Fig. 2, we use the VGG-16 encoder [32] to get a common feature map and extract latent features of the input image. Then the common feature is shared for the following two branches: one uses a U-Net [33] decoder to get upsampling features and predicts the mask of our defined room-boundary elements, and the other uses the SSD [34] method to obtain additional SSD features and get the bounding boxes of the texts.

First, we use a U-Net decoder to predict the mask of architectural elements. To expand the range of feature perception and improve the prediction accuracy, we directly combine the original features and the deconvolution features by the element-wise product unlike the concatenation in original U-Net architecture, as shown in Fig. 3. Meanwhile, we propose to use a 1×1 convolutional layer to share the information of all features and use a 3×3 convolutional layer to expand the receptive field.

On the other hand, we use a fast single shot detector (SSD) [34], as the detector for room type texts. SSD is a method designed for detecting objects in images, whose key feature is predicting category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps. Our SSD network uses the features extracted from the VGG encoder

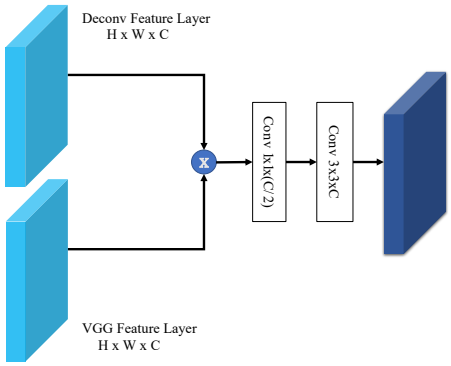


Figure 3: The combination module we used in U-Net.

layers to share the features with the above U-Net decoder. In addition to the original SSD features layers, we also extract the text locations and probability from multiple feature layers includes VGG encoder layers and additional SSD layers.

Training objectives. Technically, combining floor plan recognition and text detection seems simple, but it is not so straightforward, because the correlation between two tasks is relatively low, and it is easy to cause one branch to deteriorate during training. Therefore, we define two specific losses for each decoder branch in our network: $\mathcal{L}_{recognition}$ for predicting element masks of room-boundary pixels and $\mathcal{L}_{text}$ for locating the bounding boxes of texts. The total loss $\mathcal{L}$ for network training is a weighted sum of $\mathcal{L}_{recognition}$ and $\mathcal{L}_{text}$ with a trade-off factor λ as defined:

$$\mathcal{L} = \lambda \mathcal{L}_{recognition} + \mathcal{L}_{text}. \tag{1}$$

We use $\lambda = 0.025$ in our training step.

While detecting the basic graphical elements, we observe that the number of room-boundary pixels is too small compared to the whole image, which is a classical class imbalance problem. To overcome this issue, we define our element

type loss as:

$$\mathcal{L}_{recognition} = \sum \Big(g_c \alpha (1-p_c)^{\gamma} \log p_c + (1-g_c)(1-\alpha) {p_c}^{\gamma} \log(1-p_c) \Big) \tag{2}$$

where p_c is the pixel-level predicted confidence of class c and g_c is the ground truth of class c. α and γ are the super-parameter setting, we set $\alpha = 0.75$ and $\gamma = 2.0$ in our training stage.

To accurately locate the bounding boxes of texts, we use the following loss function defined in SSD:

$$\mathcal{L}_{text} = \frac{1}{N} \Big(\sum_{i \in Pos}^{N} \sum_{m \in \{cx,cy,w,h\}} x_{ij}^{p} smooth_{L_1}(l_i^m - \hat{g}_j^m) - \sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \Big). \tag{3}$$

In this function, x_{ij}^p means that the i-th default box is matched with the j-th ground-truth box of category p. $(\mathbf{c}_x, \mathbf{c}_y)$ is the center point of the default box (d) and w, h is its width and height. We uses a smooth L_1 loss between the predicted box (l) and the ground-truth box (g). The Pos and Neg indicate the positive and negative samples, where we use hard negative mining like SSD and the ratio between the positives and negatives is 1:1. Other terms (more details can be referenced in [34]) are defined as:

$$\begin{aligned} \hat{g}_j^{cx} &= \frac{g_j^{cx} - d_i^{cx}}{d_i^w}, \hat{g}_j^{cy} = \frac{g_j^{cy} - d_i^{cx}}{d_i^h}, \\ \hat{g}_j^{w} &= \log(\frac{g_j^w}{d_i^w}), \hat{g}_j^{h} = \log(\frac{g_j^h}{d_i^h}), \end{aligned} \tag{4}$$

and

$$\hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}, \tag{5}$$

where c_i^p is the confidence of the predicted text.

Training and implementation. Our joint neural network is implemented based on TensorFlow. For training, we optimize the loss functions using the ADAM optimizer with a fixed learning rate 1×10^{-3}. We use a batch size of 8

in our training step. We also use a pre-trained SSD model trained on VOC and COCO datasets to initialize the weights of VGG and additional SSD layers. We train the network for 30000 steps and takes about 4 hours.

Note that the size of our RuralHomeData is 2560×1600, but the input of our network is 512×512. It will cause serious loss of information if we use the original image directly or downsample it, *e.g.*the disappearance of some wall lines. In this paper, we split the image into multiple blocks (see Fig. 4) before feeding the image into our network. However, this approach will bring a new problem that the same text may exist in two blocks. To overcome this issue, we divide the image by building a grid with a cell size of 256 pixels, then we construct image blocks with a size of 512×512 by combining four neighboring cells. For example, the green square is shown in Fig. 4(a) is an image block. Since the overlaps (the dark-colored areas) between adjacent blocks will be detected more than one time, there will be some redundant text bounding boxes (see Fig 4(b)) and the elements masks may also be different in different blocks. Therefore, we combine these detection results as follows:

- For graphical elements masks, we first get the probability value for each image block from our U-Net decoder, then sum the values of corresponding pixels in the original input image to get the final probability. Finally, we use the 'argmax' function to obtain the classification result from the final probability.

- For text detection, we get the text bounding box and its probability value from our SSD network. Then we apply a non-maximum suppression (NMS) algorithm on all results in the whole input image. Since the redundant text bounding boxes may be a small part in the whole boxes and the IoU is too low, so we use the following function to calculate the Jaccard which measure the similarity of two text bounding boxes B_1 and B_2:

$$jaccard = \frac{area(B_1 \cap B_2)}{min(area(B_1), area(B_2))}.$$

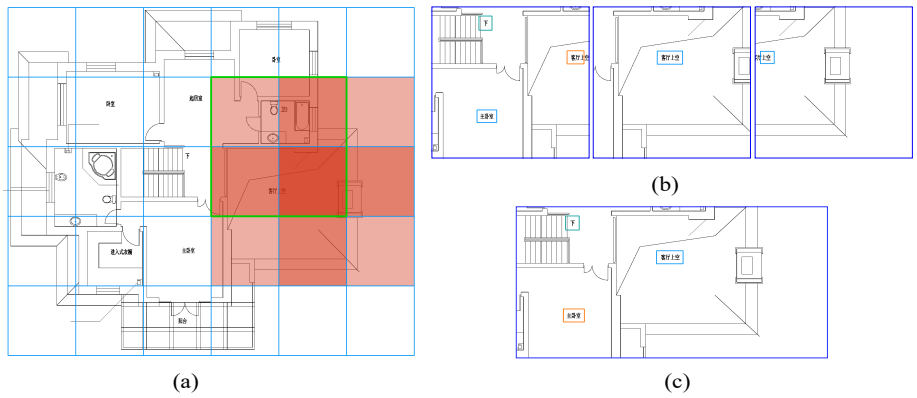
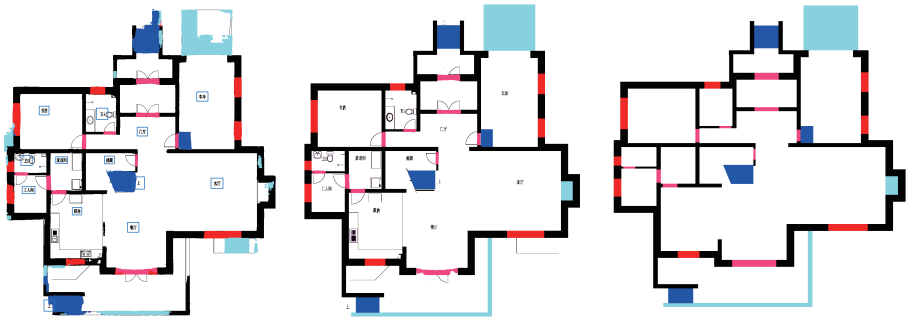


Figure 4: (a) Illustration of image blocks; (b) three individual image blocks with text detection; (c) final combination result from (b).

Fig 4 (c) shows the combination results from three image blocks in Fig 4 (b).

4.3. Room Segmentation

Room-boundary elements optimization. After feeding a floor plan into our network, we obtain the prediction results of its walls, doors, windows, stairs, slopes, and the bounding boxes of text symbols. Since our network is based on per-pixel prediction, the output is not regular and smooth, and some noises are existed near the boundaries (see Fig 5 (a)). Before completing the room segmentation, we optimize the graphical elements prediction results. Since the labels for basic elements (such as walls, windows, stairs, and slopes) are marked based on the element lines in the original input floor plan, we use the same way to optimize these types of pixels. First, we map the pixel prediction results to the binary image without assigning values to the pixels with a value of 255 to divide these results as they may lie on the element lines. And we find the connected components in the current image by clustering adjacent pixels that share the same label. To this end, we remove the noisy components in which the pixel number is less than a threshold $\mathbb{N}\,(\mathbb{N} = 50)$. And we perform region growing to assign unlabeled pixels to the label of their adjacent pixels. If the



(a) (b) (c)

Figure 5: (a) Initial floor plan prediction results; (b) the expanded result for the elements of wall, window, stair, and slope; (c) final result after elements optimization.

Intersect over Union (IOU) between the newly expanded component and its original component exceeds 50%, we accept this expanded component as the new prediction result. Since these components do not contain the original boundary of element lines, we further expand the wall prediction results to the adjacent lines.

Afterward, we iteratively visit the adjacent pixels of the basic elements and assign the label of the corresponding class whose number of neighboring pixels exceeding 10 to the same class label. As shown in Fig 5 (b), all connected elements of the four categories are expanded. Although the walls are generally composed of straight lines in standard CAD drawings, in our input floor plan, some extra pixels exist in the boundary of the intersection between the wall line and other elements lines due to the vector-to-raster conversion and image compression. Therefore, we find the rectangle set with the most pixels in the wall components and the rectangle width is more than 1, and then we delete the remaining wall element vertices as noise. Finally, because the label of the door is a standard rectangle connecting the surrounding walls, we traverse the corners of adjacent walls and find the smallest rectangle that can surround the connecting elements of the door as the optimization result. The final element optimization result is illustrated in Fig 5(c).

MIQP based room parametrization and segmentation. We combine the

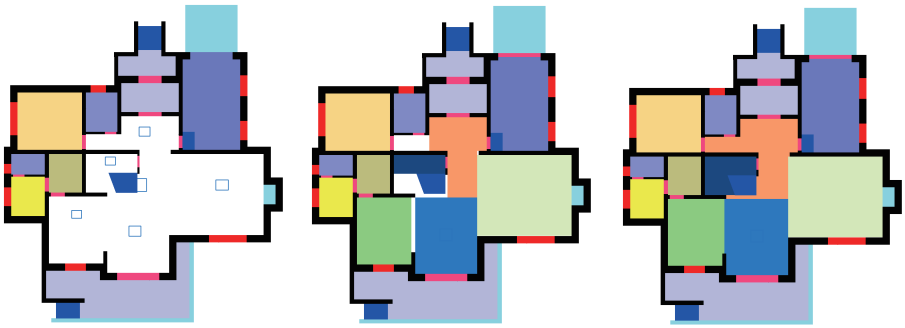


Figure 6: (a) The white area need to be further divided; (b) room segmentation result obtained by MIQP; (c) final room segmentation result.

above-optimized results with the text locations and room prediction results to parameterize the input floor plan. We introduce the *room rectangle* that is defined as the room element of a floor plan. A room rectangle is described by a tuple $(x_{min}, y_{min}, x_{max}, y_{max}, l)$, where (x_{min}, y_{min}) and (x_{max}, y_{max}) denote the position of the top-left and bottom-right corner of the rectangle respectively, and label l denotes its room type/functionality. Since the doors and windows have been optimized to be rectangular shape adjacent to wall corners, the pixels bounded by the optimized walls, doors, and windows form a closed area $\mathbb{R}$. We first extract the rectangle surrounding each stair element as the room rectangle with the label 'stair'. Besides, we use an open-source OCR engine, called Tesseract[1], to recognize the text indicating a room type in each detected text box and map the text positions into $\mathbb{R}$. Then from the text locations, we apply region growing again to divide $\mathbb{R}$ into multiple areas containing texts. For the area containing one text, we identify it (including its corners and room type) as a room in the floor plan if it is rectangular; otherwise, the polygonal area is decomposed into a set of corners and uses the same text label.

For the area containing multiple texts (Fig 6(a)), we use the *mixed-integer*

[1]https://github.com/tesseract-ocr/tesseract

quadratic programming (MIQP) method to divide the polygonal area, $\mathbb{M}$, into multiple rooms. Specifically, we first find the smallest rectangle that contains $\mathbb{M}$ and parameterize the outside area that does not belong to $\mathbb{M}$ into rectangles. Then we use the internal constraints and non-overlapping constraints proposed in [26], and add a new text location constraint that ensures the texts are located in the room rectangles. The text location constraint is defined as:

$$\begin{cases} x_{min} \leqslant x_{text}, \\ x_{max} \geqslant x_{text}, \\ y_{min} \leqslant y_{text}, \\ y_{max} \geqslant y_{text} \end{cases} \tag{6}$$

where (x_{text}, y_{text}) denotes the text location. Then we define an energy function that prefers room rectangles cover the $\mathbb{M}$ as much as possible:

$$E_{cover}\left(M\right) = Area\left(M\right) - \sum_{i} \left(x_{max_i} - x_{min_i}\right) * \left(y_{max_i} - y_{min_i}\right). \tag{7}$$

By minimizing Eq. 7 under constraints, we can get an initial valid layout of rooms. However, $\mathbb{M}$ may not be completely covered by current room rectangles (see Fig 6 (b)), then we iteratively cluster the remaining areas to obtain final layouts. Then we divide the area adjacent to only one room rectangle in $\mathbb{M}$ into room rectangles and label them directly with the same room label. For the area which is adjacent to multiple rooms: if it is rectangular, we decompose it into two small rectangle rooms according to the adjacent boundary, and the label is the same as the adjacent room label; otherwise, we first divide the area from adjacent corners into multiple rectangles and then assign the room label with the neighboring room type. Finally, we can get the room segmentation result as illustrated in Fig 6 (c)) after a few seconds of post-processing.

4.4. Room Layout Graph Extraction

From the above room segmentation, we could transform the input floor plan into a room layout graph representation, in which each node is a specific room and each edge indicates the connection relationship between two rooms (the priority is neighbor, door, window, and wall).

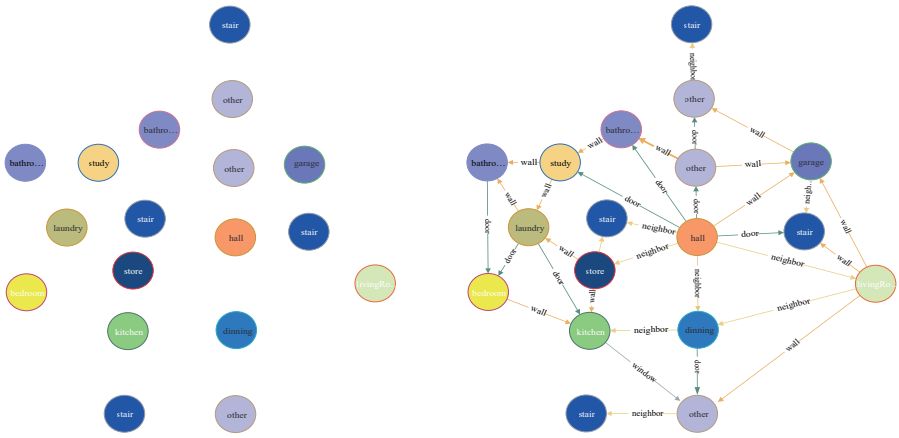


Figure 7: (a) The initial room layout graph with only nodes; (b) the final result with detailed attributes.

To construct the room layout graph, we directly take all rooms in the floor plan as the graph nodes as shown in Fig 7 (a), then detect the room attributes and connection relationships for the graph edges. First, we can obtain detailed attributes of each room based on the room area pixels, such as room types, room area, shape, aspect ratio, and orientation. We store them as node attributes. Then, we obtain a series of room boundaries from a set of room corners and expand their pixels outward to obtain the room connection relationship. While the label of the next pixel is a wall, door, or window, we will continue to expand until encounter other room pixels; Otherwise, we stop the expansion and build or update the edge in two nodes based on the original pixel label and edge priority. If the label is room, the connection relationship is neighbor, otherwise, the connection relationship is the same as the label, which is door, window, or wall. After all the expansions stopped, we got the final room layout graph that encodes all the detailed attributes of the entire input floor plan (see Fig 7 (b)).

5. Experimental Results

In this section, we first give our data collection process and training details. Then we provide a complete comparison with state-of-the-art approaches with qualitative and quantitative evaluations on the test set. We further showcase several applications. All experiments were conducted on a server computer equipped with an Intel Xeon Gold 6226R processor clocked at 2.9GHz with 64 cores, 256 GB of RAM, and an NVIDIA GeForce RTX 2080Ti (11GB memory) graphics card. We implemented our floor plan elements recognition algorithm using TensorFlow. The room segmentation was implemented in Python.

5.1. Experimental Setup

Datasets. For performance evaluation and comparison, we carry out experiments on three datasets: 1) Our RuralHomeData containing floor plan images with various room types to assess the challenging task of parsing complex floor plans of rural residence. 2) two benchmark datasets commonly used to compare performance. For the latter, we first use the R2V dataset [3] consisting of 870 ground-truth floor plan images of urban residence, where 770 images are used as training data and the remaining 100 examples are served as test images. We then randomly collect and manually label 900 images from a large-scale dataset, CubiCasa [7], which consists of 5000 ground-truth Finnish floor plan images. We split it into 800 images for training and 100 images for testing.

Evaluation metrics. For quantitative evaluations, we adopt several measures that are commonly used in the related literature [20, 5], including *overall pixel accuracy*, *per-class pixel accuracy*, *mean accuracy*, and *mean Intersect over Union (IOU)*. They are formally defined as:

$$overall\_accu = \frac{\sum_i N_i}{\sum_i \hat{N}_i}, \tag{8}$$

$$class\_accu = \frac{N_i}{\hat{N}_i}, \tag{9}$$

$$mean\_accu = \frac{1}{C} \sum_{c \in classes} \frac{N_i}{\hat{N}_i}, \tag{10}$$

$$mean\_IOU = \frac{1}{C} \sum_{c \in classes} \frac{N_i}{\hat{N}'_i}, \tag{11}$$

where C is the number of classes, $\hat{N}_i$, N_i, and $\hat{N}'_i$ are the total number of the ground-truth pixels, the correctly-predicted pixels, and the union of ground-truth and predicted pixels for the i-th classes.

5.2. Evaluation and Comparison

As competitors, we select two representative learning approaches to comparatively evaluate floor plan recognition: Raster-to-Vector [3], DeepFloorPlan [5]. These works provide a plethora of comparison to other techniques and establish themselves as state-of-the-art methods, so we omit comparisons with the works that have already been compared: *e.g.*, Raster-to-Vector outperforms a traditional approach [1], while DeepFloorPlan compares itself with various recent networks aiming to edge detection [35] and general semantic image segmentation [36, 37].

Comparison on RuralHomeData. We first evaluate the algorithms using our proposed new dataset. For a fair comparison, we use the training images from the RuralHomeData to re-train DeepFloorPlan [5] and our network. We provide both the initial prediction results of our network and the room semantic results after room segmentation. Since there are more than 20 categories of rooms in the RuralHomeData, we convert the ground-truth images into the corresponding dataset for training DeepFloorPlan. Besides, the Raster-to-Vector network can only output 2D joint points and room rectangles in the image, thus we followed the procedure presented in [35] to convert their output into an image.

Fig. 8 shows a complete visual comparison of the experiments on 5 test images. The Raster-to-Vector network is mainly for rectangular floor plans, but the RuralHomeData contains a lot of non-rectangular elements such as curved

windows and walls. As shown in the top and second rows of Fig. 8 (e), we can see that the results of Raster-to-Vector miss many room regions even though most joint points can be detected while the input image contains arcs and diagonal lines. The reason is that it is based on the assumption of the Manhattan world thus it cannot process such irregular images.In the following three rows of Fig. 8 (e), Raster-to-Vector can detect most of the walls, doors, windows, and rooms information, but they often generate wrong rectangular results when some isolated elements such walls and stairs appear in the input image. Due to the too many room categories and the similarity of room structures in the RuralHomeData, the floor plan recognition results produced by DeepFloorPlan have too many errors and cannot be used even after post-processing (see Fig. 8 (f) and (g)), because this approach cannot generalize to floor plans with unseen room types. Therefore, RuralHomeData provides very challenging data for Raster-to-Vector and DeepFloorPlan to produce satisfactory recognition results. In comparison, our method can detect most of the graphic elements (wall, window, door, stair, and slopes) and the text bounding boxes, as shown in Fig. 8 (c). Furthermore, in the later room segmentation step, we optimized the prediction results of the graphical elements and they can form an enclosed area through post-processing. We can obtain the floor plan recognition results closest to the ground-truth compared to other methods by mapping the text detection results to the enclosed area (Fig. 8 (d)).

We also report numerical statistics about the performance of each method in Table 1. As can be seen from the quantitative results, our method achieves higher accuracies for most floor plan elements, and the postprocessing could further improve our performance. However, there is no identification text in some Hall areas as shown in the last two rows of Fig. 8 (a), so the room type in these areas will be marked as other or their adjacent room types, resulting in the prediction result not being the best.

Comparison on R2V dataset. We also test our algorithm on the R2V dataset [3] to see how our pipeline can be generalized to floor plan images collected from various regions in Japan. We train our network on the R2V

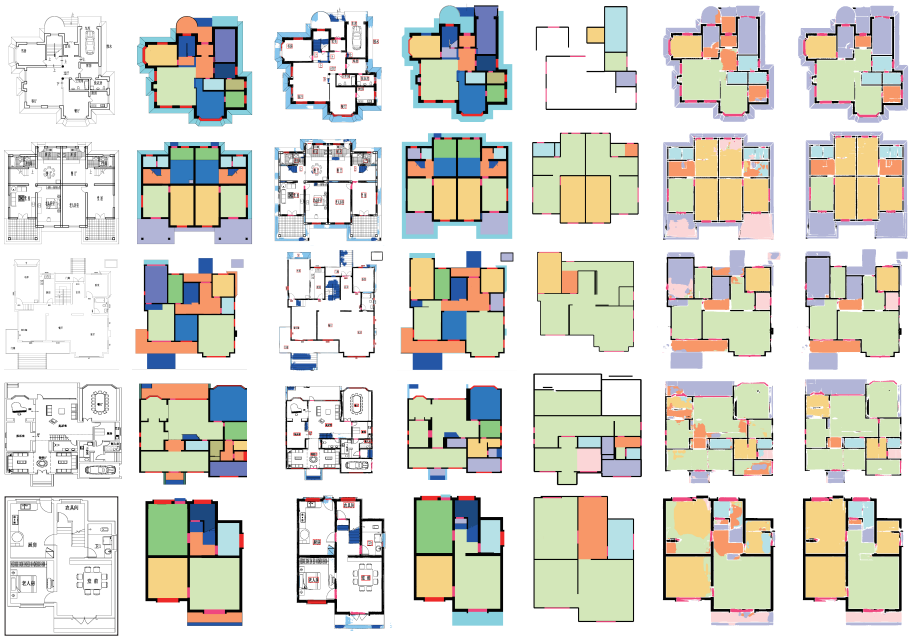


Figure 8: Visual comparison of floor plan recognition results on RuralHomeData. From left to right: input floor plan (a), ground-truth (b), our method (c), our method after room segmentation (d), Raster-to-Vector [3] (e), DeepFloorPlan [5] (f), and DeepFloorPlan with postprocessing (g).

Table 1: Comparison on our RuralHomeData. The best result of each measurement is marked in **bold** font. The symbol * indicates the methods with postprocessing. Note for our method before postprocessing ('Our'), we can only report the results on the elements of walls, doors, and windows.

Methods	*overall_accu*	*class_accu*								*mean_accu*	*mean_IoU*
		Wall	Door-and-window	Closet	Bathroom & etc.	Living room& etc.	Bedroom	Hall	Balcony		
Raster-to-Vector	0.61	0.73	0.56	0.52	0.56	0.64	0.58	0.34	0.39	0.55	0.48
DeepFloorPlan	0.63	0.88	0.63	0.76	0.60	0.69	0.64	**0.62**	0.53	0.64	0.52
DeepFloorPlan*	0.74	0.88	0.63	0.79	0.69	0.81	0.72	0.43	0.57	0.69	0.58
Our	0.85	0.90	0.73 (window) / 0.77 (door)	×	×	×	×	×	×	0.85	0.76
Our*	**0.87**	**0.96**	**0.80**	**0.89**	**0.79**	**0.93**	**0.94**	0.53	**0.70**	**0.81**	**0.76**

training data to compare to Raster-to-Vector and DeepFloorPlan.

As the Raster-to-Vector network only outputs the vector-graphics representation, we follow the annotations and procedure introduced in [3] to convert the

Table 2: Comparison on R2V dataset.

Methods	*overall_accu*	*class_accu*								*mean_accu*	*mean_IoU*
		Wall	Door-and-window	Closet	Bathroom & etc.	Living room& etc.	Bedroom	Hall	Balcony		
Raster-to-Vector	0.84	0.73	0.71	0.86	0.89	0.80	0.96	**0.87**	0.84	0.85	0.71
DeepFloorPlan	0.86	0.86	0.79	0.81	0.80	0.80	0.83	0.58	0.89	0.80	0.67
DeepFloorPlan*	0.86	0.86	0.79	0.83	0.83	0.85	0.84	0.75	0.79	0.83	0.67
Our	0.88	0.89	0.82	×	×	×	×	×	×	0.86	0.64
Our*	**0.92**	**0.90**	**0.84**	**0.90**	**0.90**	**0.94**	**0.98**	0.76	**0.94**	**0.90**	**0.82**

vector-graphics to the pixel images for comparison. For the DeepFloorPlan, we re-train their network using the official open-source code released by the authors. As this approach applies a postprocessing step to refine per-pixel prediction in room regions, we provide their initial prediction results and optimized results after postprocessing.

Fig. 9 shows the visual comparisons. As the Raster-to-Vector network predicts the junction locations to obtain the basic elements such as walls, doors, and windows, and finally finds the room area by an integer programming with various geometric and semantic constraints. Comparing to the ground-truth in Fig. 9 (b), the results of Raster-to-Vector lose some isolated wall lines and their room prediction is wrong in the area contains multiple corner points, as shown in Fig. 9 (e). The results of DeepFloor always contain noises at the junction of different classes due to the per-pixel prediction and their prediction result of the room type may be wrong. Compared with these methods, our initial door and window prediction results contain noise. After the post-processing by a simple heuristic algorithm, we eliminate the noise on the boundary. Our predictions for room type are more similar to the ground-truths as we use the text prediction results.

The numerical statistics about prediction accuracy and mean IOU for each method are reported in Table 2. Consistent with the results of RuralHomeData, our performance scores are clearly better than other methods except in the hall area.

Comparison on CubiCasa dataset. Finally, we perform the evaluation on CubiCasa [7] by retraining all of the algorithms on this dataset. Since the

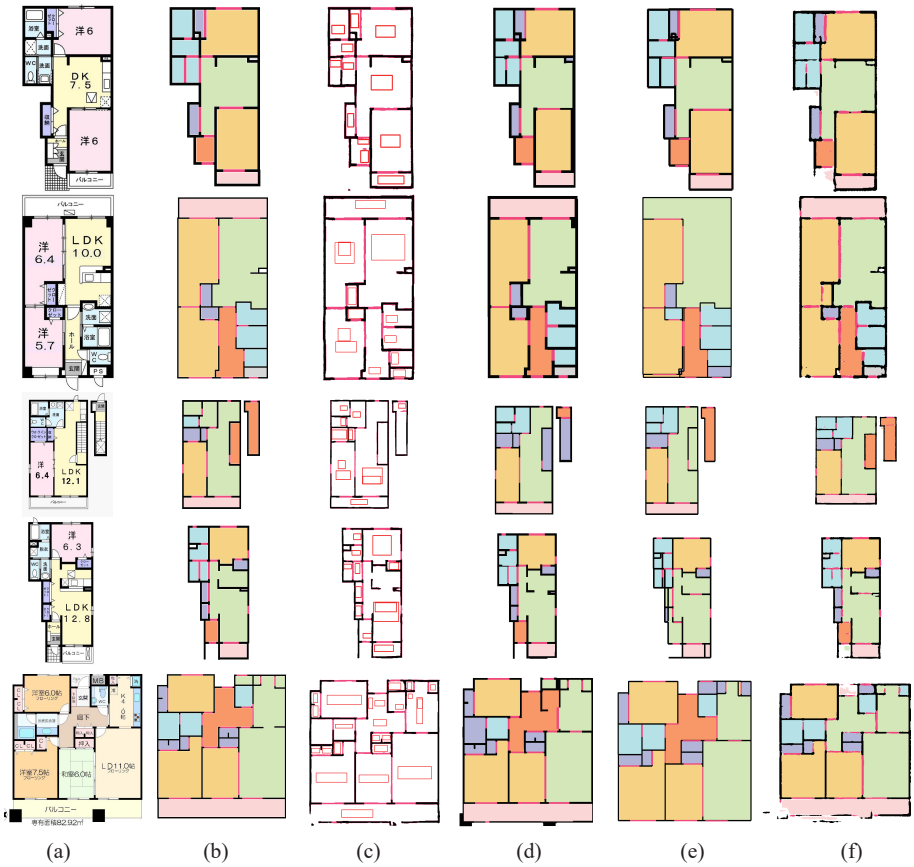


Figure 9: Visual comparison on R2V dataset. From left to right: input floor plan (a), ground-truth (b), our method without (c) and with room segmentation (d), Raster-to-Vector [3] (e), and DeepFloorPlan with postprocessing [5] (f).

CubiCasa dataset is annotated as the SVG vector graphics format, we follow its method introduced in [7] to convert the SVG vector graphics into pixel image annotations for training and comparison.

Fig. 10 shows the visual comparisons, and Table 3 reports the numerical statistics about the performance. Again, the Raster-to-Vector network can effectively predict the door and window elements and find the room areas, but its results lack these isolated wall lines as their sides belong to the same room, and they contain some non-existent door elements when these wall junctions

Figure 10: Visual comparison on CubiCasa dataset. From left to right: input floor plan (a), ground-truth (b), our method without (c) and with room segmentation (d), Raster-to-Vector [3] (e), and DeepFloorPlan with postprocessing [5] (f).

without doors form a closed loop, as shown in Fig. 10 (e). The prediction results of walls, doors, and windows in DeepFloorPlan always contain noise due to pixel-by-pixel prediction, and the room predictions may be wrong. Besides, the areas where multiple rooms exist are only be assigned to one room type due to the limitation of their post-processing (see Fig. 10 (f)). In comparison, we eliminate most of the noise and extract the room semantics by combining the text prediction results, which are the most similar to the ground-truth, as shown in Fig. 10 (d). Similarly, a large number of balcony rooms in the CubiCasa dataset have no identification text as shown in Fig. 10 (a), and there are no boundary elements to close them, so we cannot detect these areas and the prediction results are low.

Table 3: Comparison on CubiCasa dataset.

Methods	*overall_accu*	*class_accu*								*mean_accu*	*mean_IoU*
		Wall	Door-and-window	Closet	Bathroom & etc.	Living room& etc.	Bedroom	Hall	Balcony		
Raster-to-Vector	0.66	0.65	0.62	0.53	0.66	0.72	0.56	0.36	0.43	0.67	0.56
DeepFloorPlan	0.80	0.81	0.70	0.68	0.50	0.75	0.72	0.56	0.72	0.69	0.57
DeepFloorPlan*	0.82	0.81	0.70	0.74	0.55	0.78	0.76	0.36	**0.77**	0.70	0.64
Our	0.84	0.86	0.76(window) / 0.70(door)	×	×	×	×	×	×	0.77	0.58
Our*	**0.89**	**0.87**	**0.78**	**0.79**	**0.82**	**0.93**	**0.94**	**0.81**	0.37	**0.80**	**0.74**

5.3. Ablation studies

Next, we present ablation studies to verify the effectiveness of the designed framework. First, to evaluate our joint neural network, we train two independent networks for graphical element recognition using U-Net and text detection using SSD (we denote such a two-network method as "U-Net & SSD"). All networks are trained using the same parameters and epochs. Performance and time-cost comparison is summarized in Table 4. In terms of recognition performance, the results of our proposed joint network are similar to or slightly better than the results of two independent networks. Since the U-Net and SSD decoders in our joint neural network use the same VGG encoder, they can simultaneously influence the weights of the VGG encoder. Therefore, by balancing the two decoders, the results of our network are more stable compared with two independent networks. In terms of time-cost, the training and testing time of our joint network is similar to the U-Net time alone and is almost half the sum of the two-network method. Thus our proposed method is more efficient.

In addition, we conduct another ablation study on our RuralHomeData to prove the effectiveness of the image blocks. As shown in Table 5, the graphical elements prediction and text detection results of our network with image blocks are far better than the network without image blocks. Meanwhile, we also compare the different methods of combination modules used in the U-Net decoder (see Fig. 3), including direct concatenation, element-wise sum, and element-wise product. The result shows that the element-wise product is more effective. Furthermore, we convert 21 room types in our RuralHomeData into 5 types shown in [5] and use the U-net network without text detection branch to train the

R2 A2

Table 4: Ablation study of our joint neural network on three datasets. We report the experimental results of our method and U-Net& SSD on elements recognition and text detection.

Dataset	Methods	Pixel Prediction			Text detection		Training time (min)	Test time (s)
		overall_accu	*mean_accu*	*mean_IoU*	*mAP*07	*mAP*12		
RuralHomeData	Our	0.85	0.85	0.76	0.70	0.73	236	65
	U-Net & SSD	0.86	0.84	0.77	0.69	0.73	230 & 171	54 & 39
R2v	Our	0.88	0.86	0.64	0.68	0.73	233	22
	U-Net & SSD	0.87	0.86	0.65	0.67	0.72	205 & 172	19 & 16
CubiCasa	Our	0.84	0.77	0.58	0.59	0.63	233	79
	U-Net & SSD	0.81	0.80	0.57	0.60	0.63	202 & 171	65 & 56

Table 5: Ablation study of the procedure of splitting an image into multiple blocks on RuralHomeData and the methods of combination module used in U-Net decoder.

Method	Elements Recognition			Text Dection	
	overall_accu	*mean_accu*	*mean_IoU*	*mAP*07	*mAP*12
NoBlock	0.62	0.50	0.35	0.19	0.16
Block (concat)	0.82	0.84	0.73	0.70	**0.74**
Block (Eltw-sum)	0.83	0.82	0.72	0.70	0.73
Block (Eltw-prod)	**0.85**	**0.85**	**0.76**	0.70	0.73

Table 6: Ablation study of the effectiveness of text information for floorplan semantic segmentation on RuralHomeData, where the U-Net is the room segmentation network without text detection branch.

Method	Pixel Prediction		
	overall_accu	*mean_accu*	*mean_IoU*
U-Net	0.63	0.64	0.52
Our	0.83	0.86	0.64
Our*	0.92	0.90	0.82

floorplan semantic segmentation. The experimental results show that we can get more satisfactory results with text information, see Table 6.

Finally, we adjust these key parameters λ, α, and γ separately while keeping other parameters with the default settings for training to prove their impact on our model. As shown in Table 7, we can get the best elements recognition and text detection results at the same time when we use the default parameters.

Table 7: Ablation study of three key parameters on RuralHomeData. Symbol - indicates the default set of these key parameters: $\lambda = 0.025$, $\alpha = 0.75$, and $\gamma = 2.0$.

λ	α	γ	Elements Recognition			Text Detection	
			overall_accu	*mean_accu*	*mean_IoU*	*mAP*07	*mAP*12
0.01	-	-	0.81	0.70	0.54	0.58	0.61
0.05	-	-	0.74	0.62	0.50	0.61	0.61
-	0.25	-	0.77	0.64	0.51	0.70	0.70
-	0.5	-	0.82	0.70	0.58	0.61	0.66
-	-	1.0	0.79	0.68	0.57	0.59	0.60
-	-	3.0	0.78	0.67	0.56	0.70	0.67
-	-	5.0	0.68	0.50	0.39	0.60	0.64
0.025	0.75	2.0	**0.85**	**0.85**	**0.76**	**0.70**	**0.73**

5.4. Applications

Room layout graph generation. Since we transformed the floor plan into different rooms with accurate semantic information, it is easy to construct a room layout graph from the room segmentation result, as shown in Fig. 11 (c). From the graph, we can obtain various room attributes, such as type, area, doors, and windows. Besides, we can also get the connection relationship between two rooms, including the proportion of the coplanar wall, and the areas of any doors, windows, walls, and other information.

3D Modeling. Furthermore, we use our floor plan recognition and room extraction results to reconstruct the 3D model based on the procedure of [35]. Two examples are shown in Fig. 11 (d). We can extract all room elements from the layouts including stairs, and the reconstructed 3D model is consistent with the 2D floor plan.

6. Conclusion and Future Work

We have presented a new floor plan image dataset, called RuralHomeData. To the best of our knowledge, it is the first dataset containing real-world floor plans of rural residence. To further understand such residential buildings, we also proposed a novel floor plan parsing framework that combines semantic

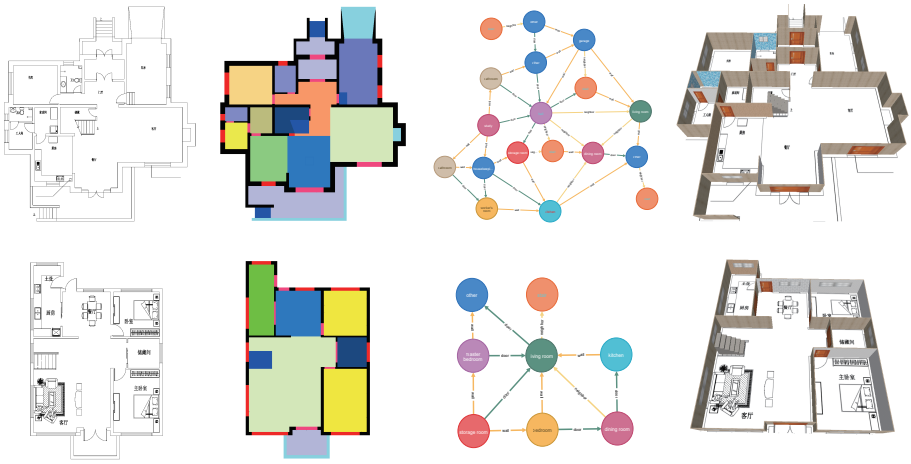


Figure 11: Two examples of room layout graph and 3D modeling results. From left to right: (a) input floor plan , (b) our room segmentation result, (c) room layout graph extracted from (b), and (d) the corresponding reconstructed 3D model.

neural networks with a post-processed room segmentation. We demonstrated the advantages of our approach by comparing with the state-of-the-art methods on our new dataset as well as previous benchmark datasets.

One limitation of our approach is that we cannot predict the semantic information of the rooms missing text labels as we have explained in the comparison results. In addition, the image blocks will lose global information, making it impossible to directly predict the semantic information of pixel room type.

In future work, we would like to predict those room types according to the existed room information through the neural network using the room knowledge graph. On the other hand, we will add the global feature to optimize the room type pixel prediction result. Besides, we are also interested in exploring new data-driven techniques for automated floor plan generation for rural residential buildings.

Acknowledgments

We thank anonymous reviewer for their valuable comments. This work is partially funded by the National Key R&D Program of China (2018YFD1100901), the National Natural Science Foundation of China (61802406, U2003109, 61761003), the Key Research Program of Frontier Sciences CAS (QYZDY-SSW-SYS004), the Strategic Priority Research Program of CAS (XDA23090304), the Youth Innovation Promotion Association of CAS (Y201935), and the Fundamental Research Funds for the Central Universities.

References

[1] S. Ahmed, M. Liwicki, M. Weber, A. Dengel, Improved automatic analysis of architectural floor plans, in: 2011 International Conference on Document Analysis and Recognition, IEEE, 2011, pp. 864–869.

[2] L.-P. de las Heras, S. Ahmed, M. Liwicki, E. Valveny, G. Sánchez, Statistical segmentation and structural recognition for floor plan interpretation, International Journal on Document Analysis and Recognition (IJDAR) 17 (3) (2014) 221–237.

[3] C. Liu, J. Wu, P. Kohli, Y. Furukawa, Raster-to-vector: Revisiting floorplan transformation, in: IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2195–2203.

[4] T. Yamasaki, J. Zhang, Y. Takada, Apartment structure estimation using fully convolutional networks and graph model, in: Proceedings of the 2018 ACM Workshop on Multimedia for Real Estate Tech, 2018, pp. 1–6.

[5] Z. Zeng, X. Li, Y. K. Yu, C.-W. Fu, Deep floor plan recognition using a multi-task network with room-boundary-guided attention, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9096–9104.

[6] L.-P. de las Heras, O. R. Terrades, S. Robles, G. Sánchez, Cvc-fp and sgt: a new database for structural floor plan analysis and its groundtruthing tool, International Journal on Document Analysis and Recognition (IJDAR) 18 (1) (2015) 15–30.

[7] A. Kalervo, J. Ylioinas, M. Häikiö, A. Karhu, J. Kannala, Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis, in: Scandinavian Conference on Image Analysis, Springer, 2019, pp. 28–40.

[8] W. Wu, X.-M. Fu, R. Tang, Y. Wang, Y.-H. Qi, L. Liu, Data-driven interior plan generation for residential buildings, ACM Trans. on Graphics (Proc. SIGGRAPH Asia) 38 (6) (2019) 1–12.

[9] P. Dosch, K. Tombre, C. Ah-Soon, G. Masini, A complete system for the analysis of architectural drawings, International Journal on Document Analysis and Recognition 3 (2) (2000) 102–116.

[10] S.-h. Or, K.-H. Wong, Y.-k. Yu, M. M.-y. Chang, H. Kong, Highly automatic approach to architectural floorplan image understanding & model generation, Pattern Recognition (2005) 25–32.

[11] S. Macé, H. Locteau, E. Valveny, S. Tabbone, A system to detect rooms in architectural floor plan images, in: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, 2010, pp. 167–174.

[12] L.-P. de las Heras, D. Fernández, E. Valveny, J. Lladós, G. Sánchez, Unsupervised wall detector in architectural floor plans, in: 12th International Conference on Document Analysis and Recognition, IEEE, 2013, pp. 1245–1249.

[13] L.-P. de las Heras, J. Mas, E. Valveny, et al., Wall patch-based segmentation in architectural floorplans, in: 2011 International Conference on Document Analysis and Recognition, IEEE, 2011, pp. 1270–1274.

[14] S. Ahmed, M. Liwicki, M. Weber, A. Dengel, Automatic room detection and room labeling from architectural floor plans, in: 2012 10th IAPR In-

ternational Workshop on Document Analysis Systems, IEEE, 2012, pp. 339–343.

[15] J. Ravagli, Z. Ziran, S. Marinai, Text recognition and classification in floor plan images, in: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), Vol. 1, IEEE, 2019, pp. 1–6.

[16] A. F. Biten, R. Tito, A. Mafla, L. Gomez, M. Rusinol, E. Valveny, C. Jawahar, D. Karatzas, Scene text visual question answering, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 4291–4301.

[17] J. Mun, M. Cho, B. Han, Text-guided attention model for image captioning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 31, 2017.

[18] X. Kong, M. Mao, W. Wang, J. Liu, B. Xu, Voprec: Vector representation learning of papers with text information and structural identity for recommendation, IEEE Transactions on Emerging Topics in Computing (2018).

[19] Y. Wang, H. Yang, X. Bai, X. Qian, L. Ma, J. Lu, B. Li, X. Fan, Pfan++: Bi-directional image-text retrieval with position focused attention network, IEEE Transactions on Multimedia (2020).

[20] S. Dodge, J. Xu, B. Stenger, Parsing floor plan images, in: 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), IEEE, 2017, pp. 358–361.

[21] W. Huang, H. Zheng, Architectural drawings recognition and generation through machine learning, in: Proceedings of the 38th Annual Conference of the Association for Computer Aided Design in Architecture, 2018, pp. 18–20.

[22] H. Jang, J. H. Yang, Y. Kiyun, Automatic Wall Detection and Building Topology and Property of 2D Floor Plan (Short Paper), in: 10th Inter-

national Conference on Geographic Information Science (GIScience 2018), Vol. 114 of Leibniz International Proceedings in Informatics (LIPIcs), 2018, pp. 33:1–33:5.

[23] D. Cheng, R. Liao, S. Fidler, R. Urtasun, Darnet: Deep active ray network for building segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 7431–7439.

[24] P. Jing, Y. Su, Z. Li, L. Nie, Learning robust affinity graph representation for multi-view clustering, Information Sciences 544 (2020) 155–167.

[25] P. Jing, S. Ye, L. Nie, J. Liu, Y. Su, Low-rank regularized multi-representation learning for fashion compatibility prediction, IEEE Transactions on Multimedia 22 (6) (2019) 1555–1566.

[26] W. Wu, L. Fan, L. Liu, P. Wonka, Miqp-based layout design for building interiors, in: Computer Graphics Forum, Vol. 37, Wiley Online Library, 2018, pp. 511–521.

[27] C. Zou, A. Colburn, Q. Shan, D. Hoiem, Layoutnet: Reconstructing the 3d room layout from a single rgb image, in: IEEE Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2051–2059.

[28] C. Sun, C.-W. Hsiao, M. Sun, H.-T. Chen, Horizonnet: Learning room layout with 1d representation and pano stretch data augmentation, in: IEEE Computer Vision and Pattern Recognition (CVPR), 2019, pp. 1047–1056.

[29] J. Chen, C. Liu, J. Wu, Y. Furukawa, Floor-sp: Inverse cad for floorplans by sequential room-wise shortest path, in: IEEE International Conference on Computer Vision (ICCV), 2019, pp. 2661–2670.

[30] S.-T. Yang, F.-E. Wang, C.-H. Peng, P. Wonka, M. Sun, H.-K. Chu, Dula-net: A dual-projection network for estimating room layouts from a single rgb panorama, in: IEEE Computer Vision and Pattern Recognition (CVPR), 2019, pp. 3363–3372.

[31] C. Liu, J. Wu, Y. Furukawa, Floornet: A unified framework for floorplan reconstruction from 3d scans, in: European Conference on Computer Vision (ECCV), 2018, pp. 201–217.

[32] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representations (ICLR), 2015.

[33] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical image computing and computer-assisted intervention, Springer, 2015, pp. 234–241.

[34] W. Liu, D. nguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European Conference on Computer Vision (ECCV), 2016, pp. 21–37.

[35] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, X. Bai, Richer convolutional features for edge detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 3000–3009.

[36] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, in: IEEE Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2881–2890.

[37] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in: European Conference on Computer Vision (ECCV), 2018, pp. 801–818.