Floor Plan Recognition and Vectorization Using Combination UNet, Faster-RCNN, Statistical Component Analysis and Ramer-Douglas-Peucker

Ilya Y. Surikov[1,2], Mikhail A. Nakhatovich[1,2], Sergey Y. Belyaev[1], and Daniil A. Savchuk[1,2](✉)

[1] Peter the Great St. Petersburg Polytechnic University (SPbPU), Polytechnicheskaya, 29, 195251 St. Petersburg, Russia
[2] ITSociety LTD, Diagonalnaya 4-1-170, 194100 St. Petersburg, Russia
dsavchuk@itsociety.su

Abstract. The floor plan recognition and vectorization problem from the image has a high market response due to the ability to be applied in such domains as design, automatic furniture fitting, property cost estimation, etc. Several approaches already exist on the market. Many of them are using just statistical or deep machine learning methods capable of recognizing a limited set of floor plan types or providing a semi-automatic tool for recognition. This paper introduces the approach based on the combination of statistical image processing methods in a row of machine learning techniques that allow training robust model for the different floor plan topologies. Faster R-CNN for the floor object detection with a mean average precision of 86% and UNet for the wall segmentation has shown the IoU metric results of about 99%. Both methods, combined with functional and component filtration, made it possible to implement the new approach for vectoring the floor plans.

Keywords: Floor plan analysis · Image processing · Deep machine learning · Transfer learning · Object detection · Augmentation

1 Introduction

The floor plan recognition and vectorization aim are to produce the standard vector format file (*.svg, *.dxf) describing the topology of the floor plan captured by the camera or by scan from the photo, print, or screenshot. Sometimes, especially for the old building – architecture documentation could be presented only in paper form, this is a problem for the digital property market. The method proposed in this paper allows recognizing and reconstructing the vector format of the floor plan that could be useful for 3D reconstruction, property price calculations, and design proposals, and so on. At the moment existed solution could be described as three following types:

N. Chaubey et al. (Eds.): COMS2 2020, CCIS 1235, pp. 16–28, 2020.
https://doi.org/10.1007/978-981-15-6648-6_2

1. Manual.
2. Could be any vector tool like CorelDraw, AutoCAD.
3. Semi-automatic.
4. Making some preprocessing before the user could fix things manually. Have a bunch of drawbacks cause some complicated cases could probably save not so much time rather than Manual methods.
5. Automatic.
6. Methods that consume only the image and returning the vector output file. Nothing must be done manually.

Mostly all automatic solutions use only deep learning or only statistical methods applicate to specific cases or datasets with known image conditions and structure. The primary purpose of this paper is to describe an approach that uses the combination of methods like computer vision, computational geometry, statistical analysis, and deep learning to enhance the general result metrics and make an approach independent on which type of plan used and what image conditions were to predict.

Our main contributions are summarized as follows:

– We proposed the approach that allows us to recognize and vectorize floor plans of different topology and different image conditions with a better IoU indicator than presented in other papers.
– We have shown an approach of enlarging small dataset using back perspective transform with physical photography. This approach has shown an increase in 1.5% in the IoU indicator and allowed us to build the solution robust to shadows.

1.1 Previous Work

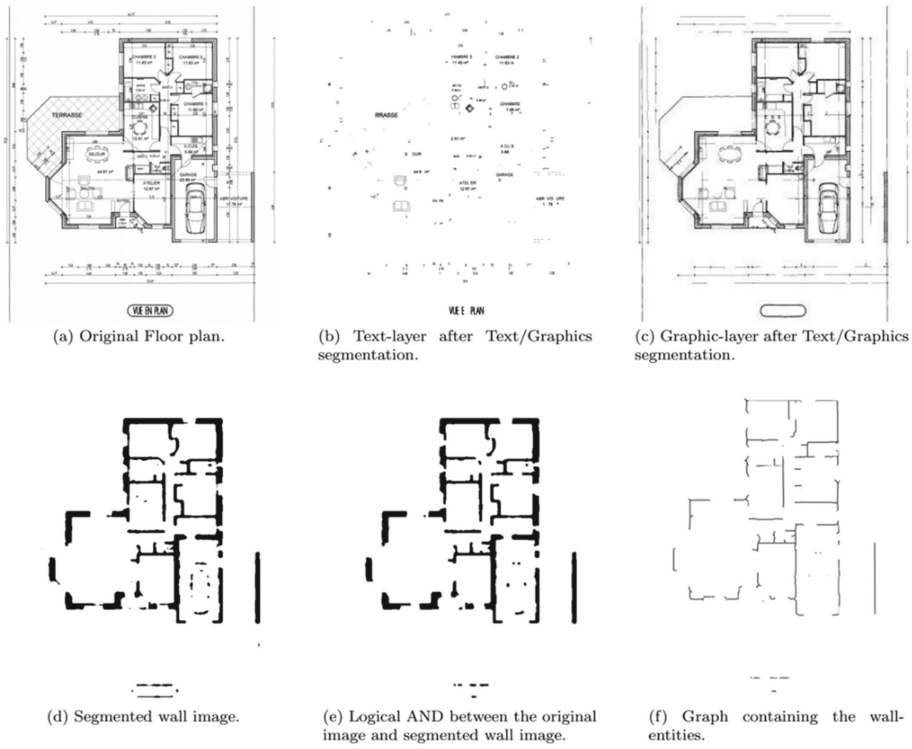
The introduction part of this paper has denoted that there are three general types of vectorization methods: manual, semi-automatic, and automatic.

The manual methods are usually general vector graphics software for developing the floor plan itself, for example, AutoCAD, CorelDraw, etc.

The investigated semi-automatic methods have many different approaches implementing the idea of the vectoring plan. Some methods used just for the preprocessing based on thresholding [1]. A bit more advanced methods starting from the thresholding and after this, switching to the edit mode, where the preserved object could be placed over the source image [2].

The essential type of methods in the context of this paper – is an automatic approach. One of the first published solutions for floor plan vectoring is based on statistical methods (blurred shape model, k-means, A*) [3]; the example result of vectoring depicted in Fig. 1.

Convolutional neural networks (CNNs) have been successfully applied in many fields, so as in the field of plan recognition. The wall segmentation approach [4] using the Fully-Convolutional Networks (FCN) shown the result of 89.9% by mean IoU metric. The example results presented in Fig. 2. As it could be observed, the best results have been achieved with FCN-2s. The approach in this paper is also based on CNN, and the UNet [5] has been chosen for this problem. It has a higher number of parameters than



(a) Original Floor plan.

(b) Text-layer after Text/Graphics segmentation.

(c) Graphic-layer after Text/Graphics segmentation.

(d) Segmented wall image.

(e) Logical AND between the original image and segmented wall image.

(f) Graph containing the wall-entities.

Fig. 1. Complete flow of wall recognition process in [3] (Image from [3]).

FCN, but, shows better accuracy. The comparison between architectures on different datasets is presented in Table 3. Since the real-time solution of this problem is not the main goal, the performance of the neural network was not considered.

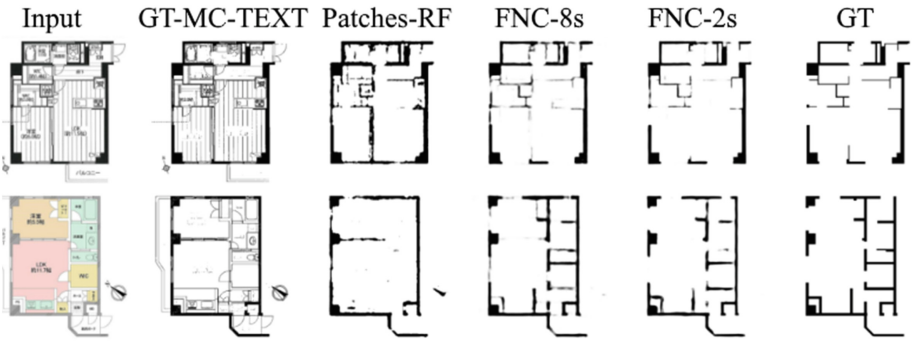


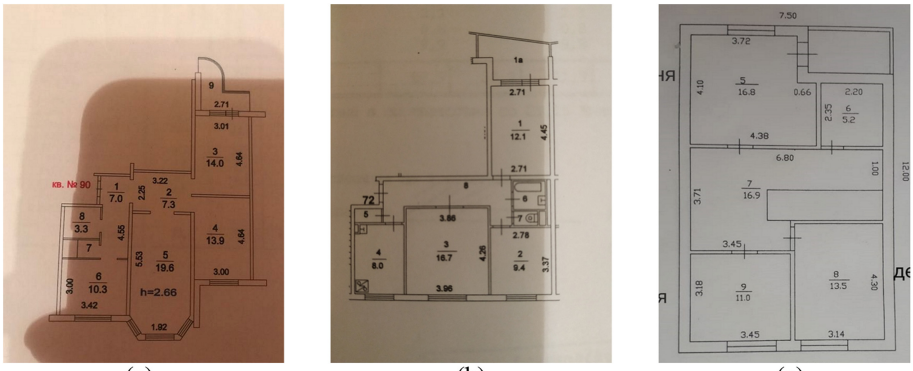
Fig. 2. Example plan segmentation [4], using FCN (Image from [4]).

The UNet architecture is used in [6] for door and wall recognition, but modified version U-Net+DCL, where the baseline UNet's deconvolution layers were replaced with a simplified version of pixel deconvolution layers for segmentation. The best result in the wall recognition task in this paper is 0.799 by mean IoU metric.

The object detection approach for filtering floor plan using Faster R-CNN [7] was applied in paper [8]. In this work, Faster R-CNN was chosen for object detection too since it shows high scores with fast convergence. They have achieved a mean average precision of 0.86, and a mean average recall of 0.92 on a dataset including 12 classes of objects.

2 The Floor Plan Datasets

The work has been performed for the specific type of plan named BTI (Bureau of Technical Inventarization), so 700 floor plan images have been collected from public real estate websites. The dataset consists mainly of scans, but private testing revealed that user input is usually a photo with different lighting conditions. An example of the user input is shown in Fig. 3.

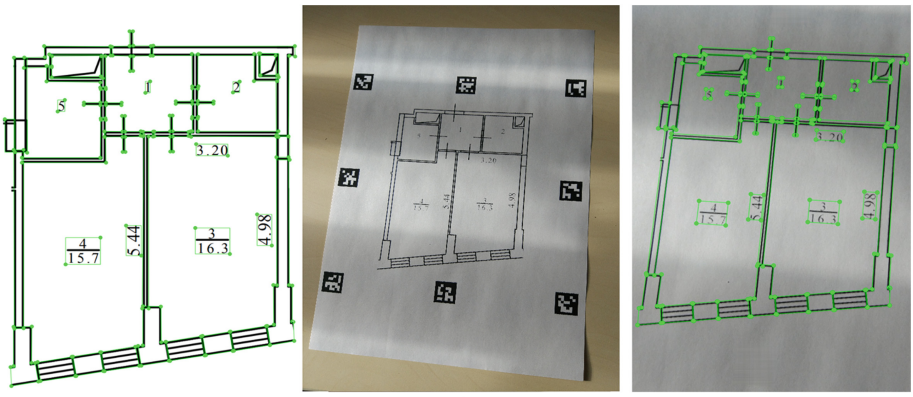


(a) (b) (c)

Fig. 3. Example of user input while private testing: (a) cascade shadow example; (b) sharp shadow border; (c) noised smooth shadow.

An approach based on inverse perspective transformation [9] was applied to build a model capable of recognizing floor plans in images with different lighting conditions, as well as to increase the number of images in the dataset. Some of the labeled plans were printed with ArUco markers from the OpenCV library [10] at a fixed distance from them. Each of the printed photos was captured 10 times in different viewpoints; an example of a photo is shown in Fig. 4(b). Markers are accurately detected by OpenCV function, so based on their coordinates, the perspective transformation matrix to the vertical plane is calculated. The layout of the original floor plan (Fig. 4(a)) is converted to the layout of the printed using an inverse matrix of the perspective transform. Then the image is cropped, and the markers are erased using the bilinear interpolation algorithm. An example of the result is presented in Fig. 4(c). Thereby, the dataset was increased to 2000 images.

Training on this expanded dataset made it possible to increase the accuracy of the IoU indicator by 3%, and to train a model resistant to shadows.



(a) (b) (c)

Fig. 4. (a) original image with labels; (b) photo; (c) the result of the transformation.

3 Architecture

2 neural networks were used for solving the problem: UNet for semantic segmentation. The U-Net architecture is built upon the Fully Convolutional Network (FCN), and the two main differences comparing to FCN are that UNet is symmetric and the skip connections between the downsampling path and the upsampling path apply a concatenation operator instead of a sum. These skip connections intend to provide local information to the global information while upsampling. The UNet architecture is used in this paper since it has been successfully applied to many image segmentation tasks, and it does not require a dataset of a dozen thousands of images to achieve high results. Pre-trained on ImageNet dataset ResNet backbone is used. In addition to the UNet, the DeepLab3+ [11] model was tested as one of the state-of-the-art models for image segmentation, but results turned out to be worse than the UNet ones. The results were compared using the Intersection over Union (1).

As a model for object detection, pre-trained on ImageNet dataset Faster R-CNN was chosen as one of the most widely used state-of-the-art architecture, which shows high accuracy even when training on a small dataset. It uses Region Proposal Network (RPN) to reduce computational time and make a good accuracy as their predecessor method. Faster R-CNN spread out in many pieces of research in object detection [12–14].

4 General Approach

The pipeline for image processing consists of several steps was developed. The main requirement for the pipeline was to add new steps easily or modifying existing ones. A scheme of the pipeline is presented in Fig. 5.

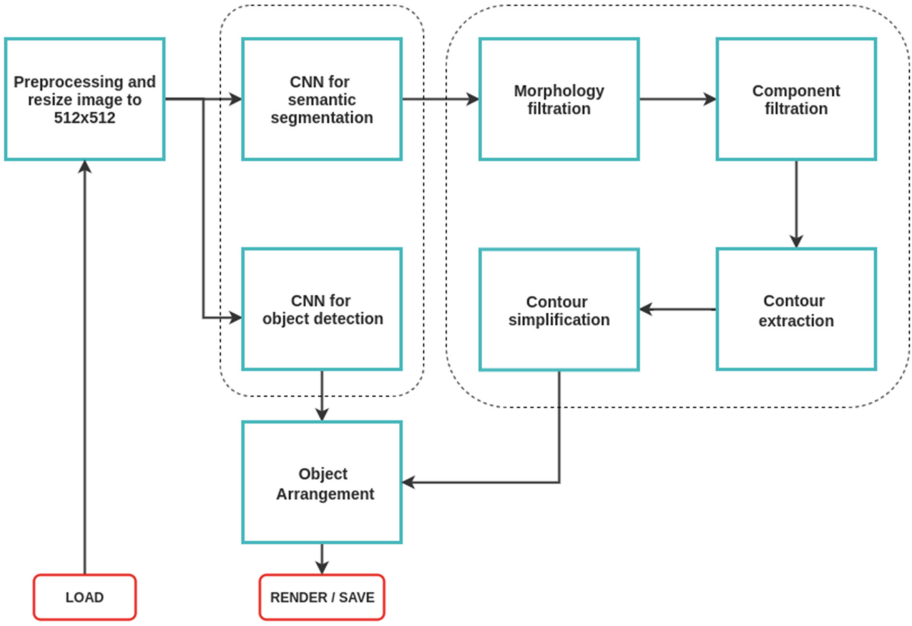


Fig. 5. Image processing pipeline.

The trained neural network consumes 512 × 512 RGB images as an input. In the first step, the image is resized from the original shape to the desired one. If the image is larger than required, compression is performed so that the bigger side has a length of 512. After that, the image is supplemented to shape 512 × 512. To avoid the appearance of image borders, next algorithm to blur the edges was applied to each of the smaller sides:

Algorithm 1. Removing image borders

Inputs: source I - image size m, n

Outputs: image 512x512 free of noize near the border

```
I←scaled(I,512 / max(m,n)) #scaling
w←(max(m,n)- min(m,n)) / 2 #width of strip
I[0:w] ← AVG(I[w]) #extrapolation of empty space up to 512x512
I[m - w: m] ← AVG(I[m-w])
blur_w ← 10
I[w-10:w+10] ← blur(I[w-10:w+10])#removing border
I[m-w-10:m-w+10] ←blur(I[m-w-10:m-w+10])
```

This method avoids the appearance of artifacts near the border in segmentation results, as shown in Fig. 6.

If the original image is smaller than required, the same algorithm is applied but using upsampling. The scaling result is processed by two neural networks.

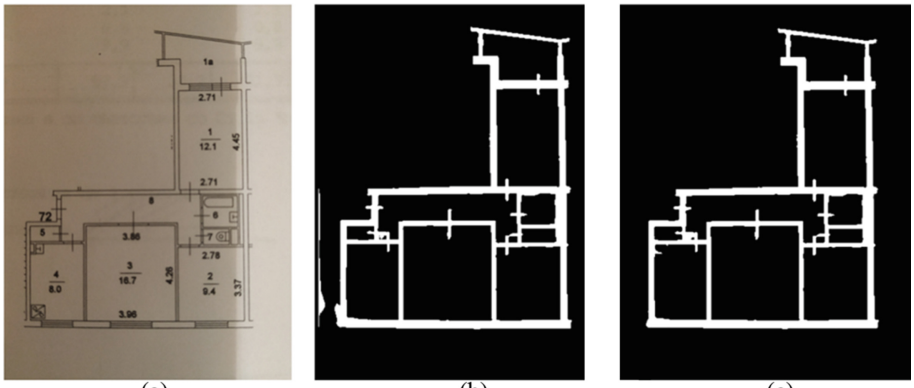


Fig. 6. (a) floor plan photo; (b) segmentation result for white background; (c) segmentation result for background with smoothed borders.

An example of the segmentation result can be seen in Fig. 7(b). It is noticeable that the result contains noise and little gaps in the walls. The morphological operations of erosion, dilatation, and closure with a 3×3 core and connectivity of 4th are used for eliminating this noise as well as inaccuracies. Figure 7(c) depicts the result of neural network pixel noise removal.

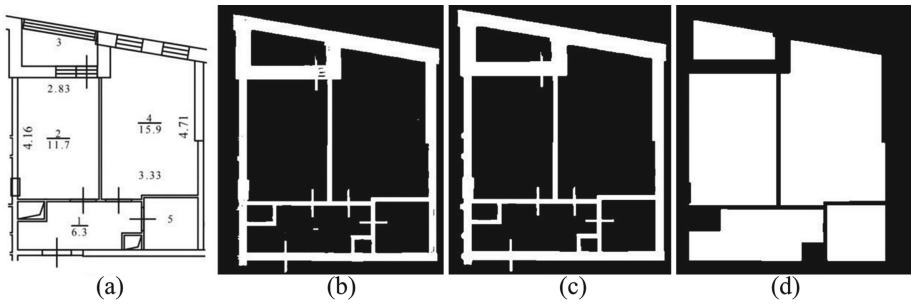
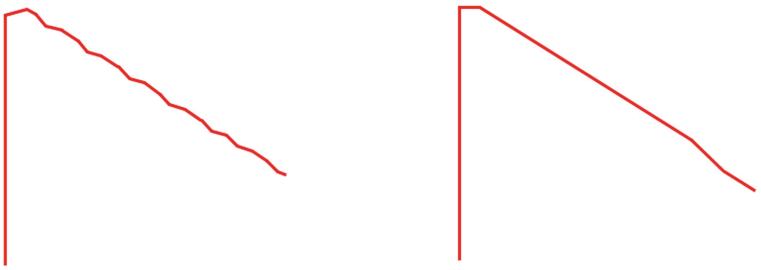


Fig. 7. (a) floor plan; (b) UNet segmentation result; (c) morphological filtration; (d) component filtration.

Based on the previous step, the connected components of the image are built and semantically representing the rooms. Morphological operations as above are applied to get rid of small defects at the border, such as part of a door segmented as a wall, but using scalable empirical constants depending on the size of the door, since it is standard and varies in small intervals of 600–1000 mm. Further, component filtration is used to remove connectivity components that are not rooms Fig. 7(d).

Based on the room components, an approximation of the components by the contours is used. Since the approximation is rough due to the pixelated source border, there is a need to simplify contours. So, Ramer – Douglas – Peucker algorithm is used [15] with the parameter $\varepsilon = 1.0$ (the parameter was selected empirically, with any increase contours

are distorted on most images, with any decrease there are no changes in contours). The algorithm reduces the number of points in the contour, thereby removes steps at non-parallel walls and also rectifying the corners of rooms, as shown in Fig. 8.



(a) (b)

Fig. 8. (a) component contour; (b) simplification result using Ramer – Douglas – Peucker algorithm.

Using the obtained contours of the rooms (internal walls borders), as well as the borders of the external walls obtained at the segmentation stage, the 1px-wide middle line is found using the Thinning Algorithm [16]. Hereupon, the wall thickness could be found.

The result of simplified contours in a row with the result of object detection is used to arrange doors, windows, and other objects. Doors and windows are placed by the method of intersecting segments, so they become part of the wall. A K-D tree [17] is constructed for reducing the enumeration of segments when searching for intersections.

5 Experiments

5.1 Performance

The developed approach has shown the ability to process user input that does not correlate with the training data. The group of methods from start to result works for 2–3 s using the following configuration: Intel Core i5, GeForce GTX 1080Ti, and 16 GB RAM, which allows recognizing incoming plans in production environments without long delays.

5.2 Accuracy

The test set contains 300 different plans, both scans and photos. There were deep learning tasks: semantic segmentation and object detection. Table 1 shows the mean average precision (mAP) score [18] for object detection using Faster R-CNN.

Intersection over Union (IoU) with threshold = 0.7 was chosen as the main metric for the validation of the segmentation model. The IoU of a predicted set of wall pixels and a set of true wall pixels is calculated as:

$$IoU = \frac{TP}{TP + FP + FN} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|} \quad (1)$$

Table 1. Faster R-CNN mAP score at different IoU thresholds.

IoU threshold	mAP, collected dataset	mAP, collected dataset expanded with photography
0.5	0.852	0.860
0.75	0.735	0.744

where TP are the true positives, FP false positives, and FN false negatives.

As aforementioned, the UNet model is used for semantic segmentation. Different backbones were tried: ResNet-34, ResNet-50, ResNet-101. They all showed approximately the same accuracy, but ResNet-50 turned out to be the best. Table 2 shows a comparison of the results for these backbones.

Table 2. Comparison of different backbones for UNet.

	ResNet-34	ResNet-50	ResNet-101
IoU, test images	0.986	0.989	0.985

Model weights are updated using binary cross-entropy soft-dice loss (2) with dice weight (dw) = 0.7, where P - predicted, T – target:

$$BCESoftDiceLoss(P,\ T) = (1 - dw) * BCE(P,\ T)\ -\ dw * SoftDice(P,\ T) \quad (2)$$

Also were tried Lovasz loss [19] and a sum of losses (3), but unfortunately, it did not improve the results.

$$SummLoss\ =\ 0.8 * WBCE + SoftDice\ +\ Lovasz \quad (3)$$

The IoU metric obtained on the test set of source data presented in our dataset is presented in Table 3. The results of training on the public floor plan dataset [20] are also presented. The public dataset contains 5000 images with labels for segmentation; 1000 of them were used for training, 200 for validation, and 3800 for testing.

Table 3. IoU score for different neural network architectures.

	IoU, collected dataset	IoU, collected dataset expanded with photography	IoU, public dataset
FCN-2s	0.951	0.947	0.945
DeepLab3+	0.944	0.933	0.922
UNet	0.974	0.989	0.975

Examples of the recognition results of test images are presented in Fig. 9, while results from private testing are shown in Fig. 10.

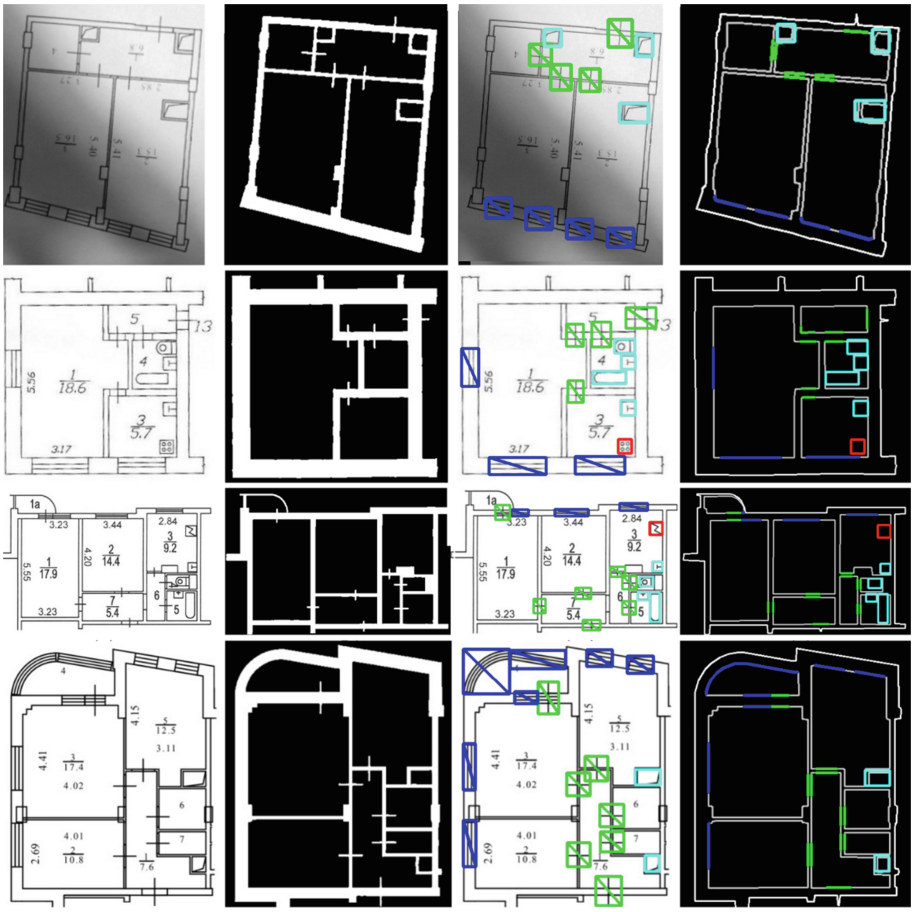


Fig. 9. Recognition system results on test images: (a) source image; (b) UNet segmentation; (c) Faster-RCNN object detection; (d) resulting vectorization.

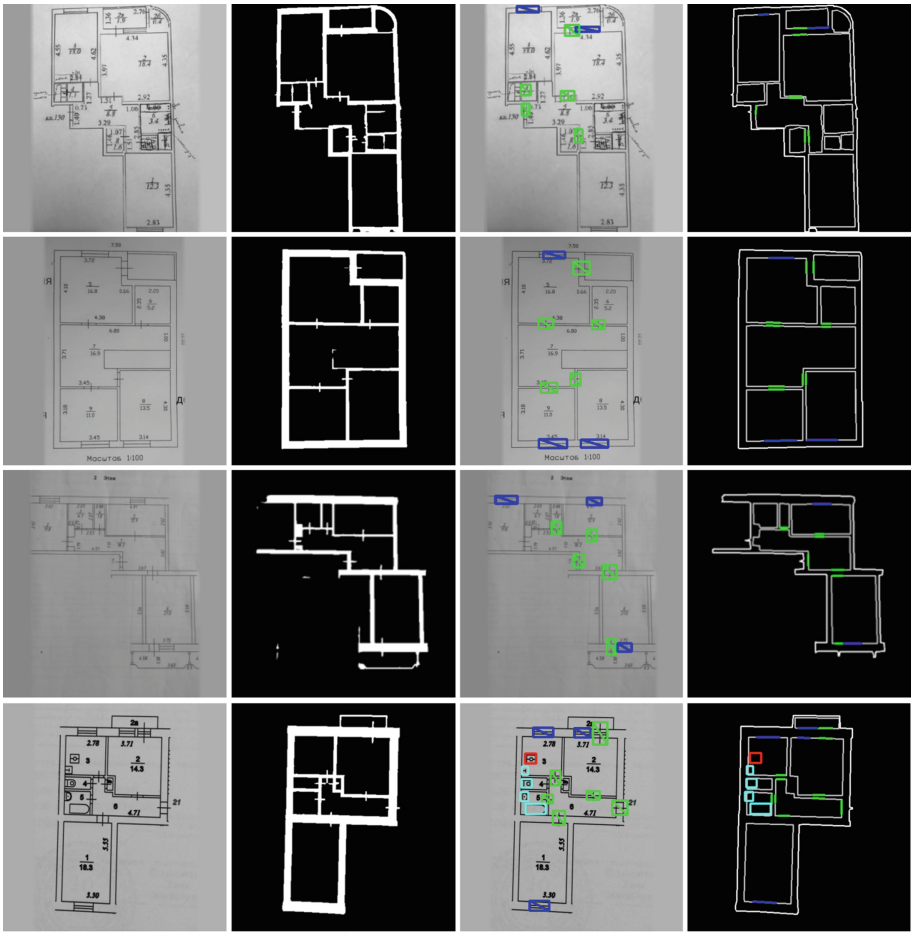


Fig. 10. Recognition system results on users' images: (a) source image; (b) UNet segmentation; (c) Faster-RCNN object detection; (d) resulting vectorization.

6 Conclusions

The developed approach allows recognizing and building vector representation for the floor plan using the combined methods of deep learning (semantic segmentation with UNet, object detection with Faster R-CNN) in a row with statistical methods (morphology, component filtration, and Ramer - Douglas - Peucker algorithm). The segmentation has shown an accuracy of about 99% by the IoU metric, while object detection has shown 86% by the mAP metric. Also, the dataset enlargement approach using a back-perspective transform was tested. This way of augmenting the dataset introduces natural spatial noise to the image that reduces risks of overfitting and allows make the processing algorithm more robust to shadows. The method developed performs the whole data

processing for one input for about 2 s. It allows using this approach for cloud-based recognition systems or any other productive deployment.

References

1. PlanTracer. http://www.plantracer.ru. Accessed 26 Oct 2019
2. PlanCAD. System of automated design of floor plans https://sapr.ru/article/18006. Accessed 26 Oct 2019
3. de las Heras, L.-P., Ahmed, S., Liwicki, M., Valveny, E., Sánchez, G.: Statistical segmentation and structural recognition for floor plan interpretation. Int. J. Doc. Anal. Recogn. (IJDAR) **17**(3), 221–237 (2014). https://doi.org/10.1007/s10032-013-0215-2
4. Dodge, S., Xu, J., Stenger, B.: Parsing floor plan images. In: Fifteenth IAPR Conference on Machine Vision Applications (MVA) (2017)
5. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. Comput. Res. Repository (2015)
6. Yang, J., Jang, H., Kim J.: Semantic Segmentation in architectural floor plans for detecting walls and doors. In: 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Beijing, China, 2018, pp. 1–9 (2018). https://doi.org/10.1109/cisp-bmei.2018.8633243
7. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
8. Ziran, Z., Marinai, S.: Object Detection in Floor Plan Images. In: Pancioni, L., Schwenker, F., Trentin, E. (eds.) ANNPR 2018. LNCS (LNAI), vol. 11081, pp. 383–394. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99978-4_30
9. Borgefors, G.: Distance transformations in digital images. Comput. Vis. Graph. Image Process. **34**, 344–371 (1986)
10. OpenCV. https://opencv.org. Accessed 26 Oct 2019
11. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: The European Conference on Computer Vision (ECCV), pp. 801–818 (2018)
12. Zhu, B., Wu, X., Yang, L., Shen, Y., Wu, L: Automatic detection of books based on faster R-CNN. In: Third International Conference on Digital Information Processing, Data Mning, and Wireless Communication (DIPDMWC), pp. 8–12 (2016)
13. Zhang, H., Du, Y., Ning, S., Zhang, Y., Yang, S., Du, C.: Pedestrian detection method based on faster R-CNN. In: 13th International Conference on Computational Intelligence and Security (CIS), pp. 427–430
14. Xu, Z., Wu, Z., Feng, J.: CFUN: combining faster R-CNN and U-net network for efficient whole heart segmentation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
15. Ramer, U.: An iterative procedure for the polygonal approximation of plane curves. Comput. Graphics Image Process. **1**(3), 244–256 (1972)
16. Gonzalez, R., Woods, R.: Digital Image Processing, pp. 541–545. Addison-Wesley Publishing Company (1992)
17. Bentley, J.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9), 509–517 (1975)
18. Beitzel, S.M., Jensen, E.C., Frieder, O.: MAP. In: Liu, L., Özsu, M.T. (eds.) Encyclopedia of Database Systems. Springer, Boston (2009). https://doi.org/10.1007/978-3-540-29678-2_3313

19. Berman, M., Rannen Triki, A., Blaschko, M.: The Lovász-Softmax loss: a tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
20. Kalervo, A., Ylioinas, J., Häikiö, M., Karhu, A., Kannala, J.: CubiCasa5K: a dataset and an improved multi-task model for floorplan image analysis. In: Felsberg, M., Forssén, P.-E., Sintorn, I.-M., Unger, J. (eds.) SCIA 2019. LNCS, vol. 11482, pp. 28–40. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20205-7_3