

AUTOMATIC 3D BUILDING MODEL GENERATION USING DEEP LEARNING METHODS BASED ON CITYJSON AND 2D FLOOR PLANS

R.G. Kippers¹, M. Koeva², M. van Keulen³, S.J. Oude Elberink²

¹ Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS), University of Twente - r.g.kippers@student.utwente.nl

² Faculty of Geo-Information Science and Earth Observation (ITC), University of Twente - (m.koeva,s.j.oudeelberink)@utwente.nl

³ Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS), University of Twente - m.vankeulen@utwente.nl

Commission IV, WG WG IV/10ICWG

KEY WORDS: deep learning, floor plans, 3D city models, indoor spatial data, building models

ABSTRACT:

In the past decade, a lot of effort is put into applying digital innovations to building life cycles. 3D Models have been proven to be efficient for decision making, scenario simulation and 3D data analysis during this life cycle. Creating such digital representation of a building can be a labour-intensive task, depending on the desired scale and level of detail (LOD). This research aims at creating a new automatic deep learning based method for building model reconstruction. It combines exterior and interior data sources: 1) 3D BAG, 2) archived floor plan images. To reconstruct 3D building models from the two data sources, an innovative combination of methods is proposed. In order to obtain the information needed from the floor plan images (walls, openings and labels), deep learning techniques have been used. In addition, post-processing techniques are introduced to transform the data in the required format. In order to fuse the extracted 2D data and the 3D exterior, a data fusion process is introduced. From the literature review, no prior research on automatic integration of CityGML/JSON and floor plan images has been found. Therefore, this method is a first approach to this data integration.

1. INTRODUCTION

Buildings have a significant role in our daily lives. Therefore, a lot of effort is put into improving them. One way of doing this is by applying digital innovations to a building's whole lifecycle, which consists of the following stages: planning, construction, operation, renovation and demolition (Ngwepe and Aigbavboa, 2015). In the past decade, a lot of effort is put into applying digital innovations to building life cycles (planning, construction, operation, renovation and demolition (Ngwepe and Aigbavboa, 2015)). 3D Models have been proven to be efficient for decision making, scenario simulation and 3D data analysis during this life cycle (Rajat Agarwal and Sridhar, 2016). Creating such digital representation of a building can be a labour-intensive task, depending on the desired scale and level of detail (LOD). This research aims at creating a new automatic deep learning based method for building model reconstruction. It combines exterior and interior data sources: 1) 3D BAG, the first fully automatically generated 3D building data set with level of detail 2.2¹, 2) archived floor plan images (e.g. scanned or exported from CAD software). From the literature review, no prior research on automatic integration of CityGML/JSON and floor plan images has been found. Therefore, a new method has been developed, combining these two internal and external data sources into full 3D models. Having precise linked 3D data provides opportunity for detailed analysis such as improved area calculation, facility management, urban planning, and energy simulation.

2. BACKGROUND

3D representations of buildings are getting more attention, and are used for a wide range of applications (Biljecki, 2016). 3D building objects do exist in both the BIM and GIS domain, but are used from a different perspective (Herle, 2020). BIM models are by default highly detailed and often represent a physical building with a small error margin. GIS is often used on a larger scale, with a lower granularity, and provides variety of opportunities for geolocation, analysis, simulations and documentation (Laat, 2010). 3D building objects in city models can be obtained automatically, in example by utilizing building footprints and boundary properties from data sources e.g. cadastre databases or aerial images (Amiranti, 2020, Balázs Dukai¹, 2020, Sander Oude Elberink and Commandeur, 2013). These automatically generated objects do not contain building internal structure. Research by Boeters (Boeters, 2015) introduced LOD2+, which is an extension to City GML 2, to support floor levels. In this paper, a method is introduced to estimate heights and include floor surfaces to the existing CityGML LOD 2 objects. Current ongoing research by iNOUS² aims at reconstructing IndoorGML based on building blueprints and LIDAR data. After that, this data will be combined with existing IFC (BIM) files. This research is not published yet.

CityJSON - CityJSON (Ledoux et al., 2019) is a JSON encoding for 3D city models. The introductory paper claims that, in comparison with CityGML (the current Open Geospatial Consortium standard) it is easier to use, and it is more compact with a compression factor of around six with real-world data (Ledoux et al., 2019).

Floor plan parsing - Previous work from Or et. al. (hang Or,

¹ 3D BAG, 3D BK TU Delft - <https://3dbag.nl>

² iNous, Pusan National University - <http://www.inous.ne>

2005) in 2005 introduced deterministic image processing and symbol recognition techniques to interpret floor plan scans. However, as with many computer vision problems, the focus has shifted from feature engineering and deterministic tasks to methods learning from training data (Ahti Kalervo et al., 2019). In order to obtain the multiple segmentation maps and labels, e.g. room types and point of interest (walls, icons, openings etc.), multi-task methods or networks should be used. The performance of multi-task networks do highly depend on the relative weighting between each task's loss (Kendall et al., 2018). The deep learning breakthrough for floor plan parsing was presented in research by Chen Liu (Liu, 2017), that used deep learning to vectorize rasterized images. It does so by using a discriminative network to obtain junctions, integer programming to obtain primitives and finally post-processing to obtain a vector format. In Table 1, other relevant work is shown.

Polygon overlap - To find the appropriate scale, rotation and orientation to project the exterior and floor plan outline, the maximum overlap of the two outlines needs to be found. De Berg (Berg, 2005) defined a method on finding optimal polygon overlap under translations. However, this method only works for convex polygons. Building blueprints do not have to be convex, so this function is not applicable. Milenkovic (Milenkovic, 1998) introduces a method for optimal overlap using rotations and movements, that also works for non-convex polygons. This method does not support other methods such as scaling. Har-Peled (Har-Peled, 2016) introduces a method that approximates the maximum overlap of a polygon under translations. For polygons close to convex, this problem can be solved in nearly linear time. Research by Ahn et. al. (Ahn et al., 2007) calculates the maximum overlap for two polygons using rigid motions (translation, rotation, scale, reflection and glide reflection). This method gives a rigid motion ϕ^{app} that is at least $1 - \alpha$ times the maximum over all rigid motions. Berg et. al. (De Berg et al., 1998) introduces a method for maximum overlap of two convex polygons under translations in $O(n + m)\log(n + m)$ time, where n and m are the number of vertices in the respective polygons. The method is an algorithm of steps with binary searches for new locations for the points based on the average centroid (geometric centre) of both polygons.

3. METHOD

To reconstruct 3D building models from the two data sources mentioned above, an innovative combination of methods is proposed (Figure 1). The exterior 3D data is in CityJSON format and the interior 2D data is provided by the municipality of Rijssen-Holten, who are interested in the current research. In addition, road data is downloaded from the national road database³. The next subsections elaborate on component I (interior) and II (data fusion).

3.1 Floor Plan Parsing

In order to obtain the information needed from the floor plan images, they are parsed into vectors with attached semantic attributes (Figure 1). The walls, openings and room labels are extracted separately, adopting the deep learning methods proposed Liu (Liu, 2017). In order to train above mentioned deep learning models, the CubiCasa (Ahti Kalervo et al., 2019)

dataset is used. This dataset consists of 5000 floor plan images (input) and annotated vector files (ground truth).

Walls: The first step is to obtain individual walls, using semantic segmentation, which is the process of labeling each pixel of an image with a corresponding object class (wall or other). For this, the U-Net network and Fast-SCNN are tested. After segmentation, individual walls are obtained by morphological transformations (closing, dilation and erosion) and post-processing. A single polygon is created by combining all contours (Figure 2). From this polygon, wall pieces are obtained by an algorithm that removes wall elements from the all-walls polygon, starting from the left-top. This algorithm also allows for diagonal walls, and is provided in Algorithm 1.

Openings: Openings (doors and windows) are identified using object detection. Four deep learning architectures are tested: 1) Faster-RCNN, 2) CenterNet, 3) SSD with MobileNet, and 4) SSD RetinaNet encoder. The object detection algorithms detect bounding boxes around the objects of interests. An example output can be seen in Figure 3.

Room labels: In order to obtain the room types, the labels in the floor plans are interpreted. For this, optical character recognition (OCR) is applied. There are a variety of off-the-shelf OCR techniques, of which a few are compared in the research done by Tomaschek (Tomaschek, 2018). It was concluded that Tesseract version 4 has the best performance. This consists of a new neural net (LSTM) based OCR engine, which focuses on line recognition. A computer vision algorithm that recognizes objects surrounded by white space is used as input for the Tesseract neural net.

Algorithm 1 Create individual walls

```
while all-walls.area > 0.01 do
  top-left ← top-left point from all-walls polygon
  highest-angle ← point with connecting edge to top-left
  with highest  $\theta$ 
  highest-dx ← point with connecting edge from top-left
  with highest  $\Delta x$ 
  wall ← Polygon from top-left to highest-angle
  direction ← -1 if highest angle point  $x_j$  top-left point  $x_i$ 
  else 1
  while intersection(all-walls,wall).area == wall.area do
    Extend wall in direction on x-axis
  end while
  if wall.area == 0 then
    Make bounding box around top-left, highest-angle and
    highest-delta points,
    find the intersection with all-walls and select the largest
    element as wall
  end if
  all-walls ← Symmetric difference between all-walls and
  wall
end while
```

3.2 Merging the 3D BAG and floor plan data

The parsed floor plan is merged with the 3D BAG data. Since the floor plans do not indicate an orientation (e.g. to the North), the roads are used to rotate the 3D BAG building with respect to the floor plan. For each facade with an adjacent road, the maximum polygon overlap (Jaccard index, IoU), is optimized using trust region methods (Conn et al., 2000) (Figure 4). Real-life object sizes are estimated based on the values in Table 2. Inconsistent areas are added to the final CityJSON object if the inconsistency area $5m^2$. The final building, consisting of floor planes, interior and exterior walls, doors, windows, and roof is stored as CityJSON type Building object.

³ NWB - <https://nationaalwegenbestand.nl/>

Paper	Targets	Method	Architecture	Loss
(Surikov, 2020)	Walls	Semantic segmentation	U-Net	IoU
	Window/Door/Icons	Object detection	Faster-RCNN	Mean average precision (MaP)
(Liu, 2017)	Wall or room type	Semantic segmentation	Modified ResNet-152	Pixelwise soft-max cross entropy loss
	Icons	Semantic segmentation	Modified ResNet-152	Pixelwise soft-max cross entropy loss
(Yang, 2018)	Doors	Semantic segmentation	U-Net+DCL	mIoU, mean accuracy
	Walls	Semantic segmentation	U-Net+DCL	mIoU, mean accuracy
(Ahti Kalervo et al., 2019)	Wall or room type	Semantic segmentation	Modified ResNet-152	Cross-entropy loss
	Icons	Semantic segmentation	Modified ResNet-152	Cross-entropy loss
(Zeng, 2019)	Room boundary (wall, door, window)	Classification and detection	VGG16	Cross-and-within-task weighted loss
	Room type	Classification and detection	VGG16	Cross-and-within-task weighted loss
(H, 2020)	Wall or room type	Object detection	Multiple	Dice loss, WCE loss
	Doors	Object detection	Multiple	Dice loss, WCE loss
(Wu, 2020)	Walls, doors, icons	Instance segmentation	Mask R-CNN	Multi-task loss function of Mask-RCNN

Table 1. Deep learning methods comparison

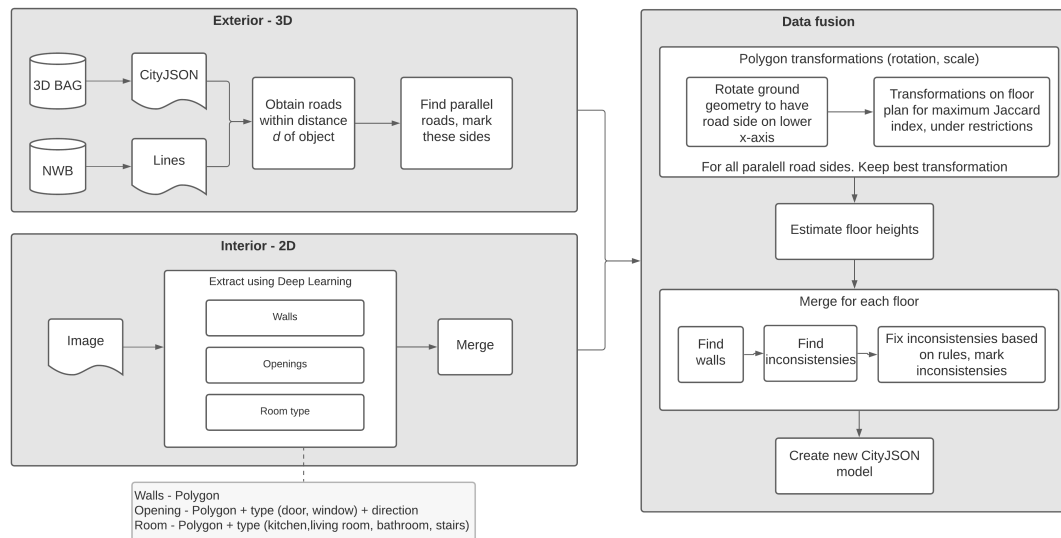


Figure 1. Global design, including component I and II (section 3.1 and 3.2)

Year y	Interior wall width	Exterior wall width	Level height	Floorplane height	Door ground offset	Door height	Window ground offset	Window height	Roof height
$y < 1950$	10	25	220	20	2	210	60	150	20
$1950 \leq y \leq 1992$	10	30	240	26	2	220	60	150	30
$y > 1992$	10	35	260	28	2	232	60	150	40

Table 2. Reference sizes in cm

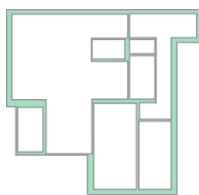


Figure 2. Floor plan outline

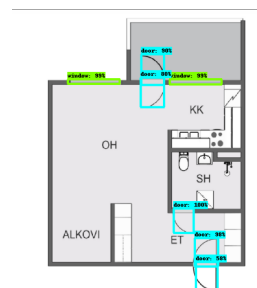


Figure 3. Object Detection Output

4. EVALUATION

4.1 Floor Plan Parsing

For the final method, the deep learning architectures U-Net (Ronneberger et al., 2015) (IoU of 0.8859) SSD with Mobilenet v1 FPN, RetinaNet50 (Liu et al., 2016) (average precision 0.744 and average recall 0.595) are selected.

To evaluate the described methods output, 10 random samples

from the CubiCasa5K dataset and 6 Dutch samples are inspected. Since the study area of this method is the Netherlands and CubiCasa5K does not contain Dutch floor plan samples, 6 Dutch samples are chosen. For each sample, the true positives (TP, correct items), false positives (FP, output that should not be in the result), and false negatives (FN, missing items) are given. The precision and recall are added to each table, and the harmonic mean of the precision and recall (F1-score) are calculated. To find the accuracy of the

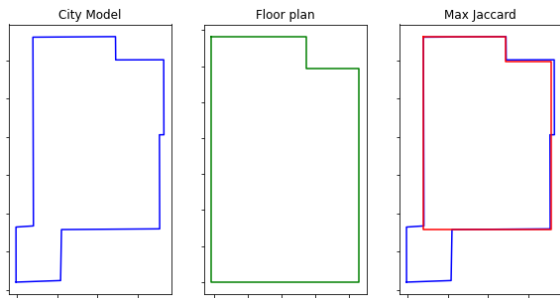


Figure 4. Maximum Jaccard / IoU

door direction, the ratio between correct directions and true positive doors is given. An object is considered true positive if more than 75% of the ground truth wall area is covered. In Table 3, the F1 scores for the CubiCasa5K and Dutch samples are presented.

	Wall	Door	Window
CubiCasa5k	0.9	0.93	0.85
Dutch floor plan samples	0.83	0.66	0.69

Table 3. F1 scores per type

4.2 Merging the 3D BAG and floor plan data

In order to evaluate the proposed method, the method output is compared with a ground-truth BIM-model. The BIM model and its corresponding CityJSON output is shown for each side and storey. The quantitative accuracy is shown using Intersection over Union (IoU, Equation 1). In order to calculate the IoU, polygons are drawn on all objects (such as walls, openings and roofs) on the facade from 4 sides (front, back, left, and right) for both the ground truth and the proposed model output. For each polygon, the IoU is calculated. In this paper, results for two buildings (identified by BAG ID 0163100000538672 and 0164100000294714) are shown in Figure 5, Table 4 and Figure 6, Table 5. The average IoU is 0.7673 and 0.6399

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|}, 0 \leq IoU(A, B) \leq 1 \quad (1)$$

5. DISCUSSION

The overall performance for floor plan parsing seems better when floor plans are less complicated and less detailed. This is probably due to homogeneous training data and can be improved by including more complicated floor plan images (e.g. office spaces and larger buildings) to the training data set. During training, a Nvidia GeForce GTX 1080 GPU was used. This GPU was launched in 2016 and currently there are more powerful alternatives available that allow for more complex deep learning networks, which could improve performance.

For data integration, the average Intersection over Union is 0.7673. No similar benchmark is available in other scientific literature, but for such challenging task this result is sufficient.

6. CONCLUSION

The aim of this research was to create a methodology for the reconstruction of 3D building models using the 3D BAG CityJSON and floor plan images. In this paper, a method is proposed to reconstruct 3D building models using deep learning methods based on CityJSON and 2D floor plans. The method can be seen in Figure 1. The performance of this method is measured in Intersection over Union (IoU) and seems sufficient for such challenging task. A way of improving the method is by using more representative training data.

An interesting future direction to obtain walls and openings is to use generative design or parametric design to estimate a buildings internal structure. In example, (Abrishami et al., 2014) proposed a system for generative design for BIM models. By observing floor plans, building 3D exterior, and meta data such as building year, type (terraced, apartment, semi-detached, detached), the most likely internal structure may be estimated by a machine learning model.

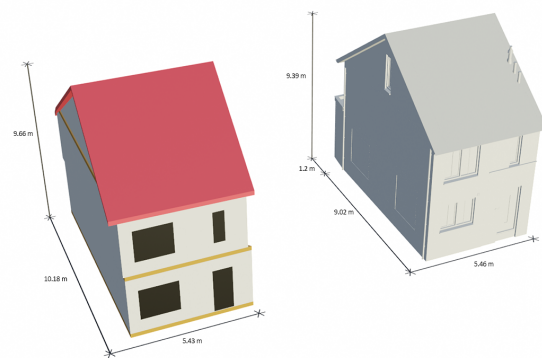


Figure 5. Visual results (in Blender) for BAG ID 0164100000294714

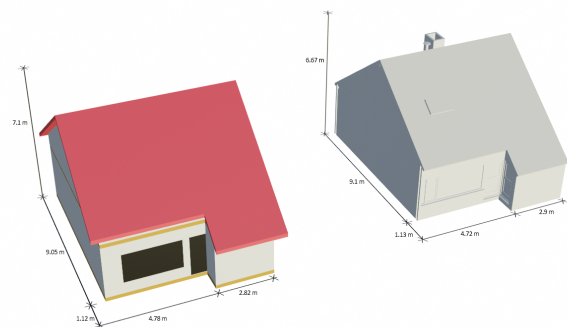


Figure 6. Visual results (in Blender) for BAG ID 0163100000538672

Object	IoU
Front view – Roof	0.7585
Front view – Facade	0.9402
Front view – Ground floor window	0.8311
Front view – Ground floor door	0.6379
Front view – First floor window 1	0.85
Front view – First floor window 2	0.4668
Back view – Facade	0.9056
Back view – Roof	0.76
Back view – Ground floor window	0.5046
Back view – First floor window 1	0.6033
Back view – First floor window 2	0.8173
Left view – Facade ground floor	0.9347
Left view – Facade first floor	0.7819
Right view – Facade ground floor	0.9347
Right view – Facade first floor	0.7819
Average	0.7673

Table 4. IoU results for BAG ID 016410000294714

Object	IoU
Front view – Roof	0.7424
Front view – Facade	0.8898
Front view – Ground floor window 1	0.7198
Front view – Ground floor door 1	0.7703
Front view – Ground floor window 2	0.8472
Front view – First floor window 1	0
Back view – Roof	0.7005
Back view – Facade	0.8423
Back view – First floor window	0.6763
Back view – First floor door	0.6751
Back view – First floor window	0
Left view – Wall first floor	0.7274
Right view – Wall first floor	0.7274
Average	0.6399

Table 5. IoU results for BAG ID 016310000538672

REFERENCES

- Abrishami, S., Goulding, J., Rahimian, F. P., Ganah, A., 2014. Integration of BIM and generative design to exploit AEC conceptual design innovation. *Information Technology in Construction*, 19, 350–359.
- Ahn, H.-K., Cheong, O., Park, C.-D., Shin, C.-S., Vigneron, A., 2007. Maximizing the overlap of two planar convex sets under rigid motions. *Computational Geometry*, 37(1), 3–15.
- Ahti Kalervo et al., 2019. CubiCasa5K: A Dataset and an Improved Multi-Task Model for Floorplan Image Analysis. <https://arxiv.org/pdf/1904.01920v1.pdf>.
- Amiranti, A. Y., 2020. Investigating 3D input data for solar photovoltaic potentials in The Netherlands. <http://essay.utwente.nl/85178/>.
- Balázs Dukai¹, Ravi Peters¹, T. W. T. C. H. L. T. B. M. P. V. v. A. W. v. H. J. S., 2020. Generating, storing, updating and disseminating a countrywide 3D model. <https://repository.tudelft.nl/islandora/object/uuid>
- Berg, D., 2005. Computing the Maximum Overlap of Two Convex Polygons Under Translations. <https://hal.inria.fr/inria-00413175/document>.
- Biljecki, H. Ledoux, J. S., 2016. An improved LOD specification for 3D building models. <https://repository.tudelft.nl/islandora/object/uuid>
- Boeters, R; Arroyo Ohori, G. B. F. Z. S., 2015. Automatically enhancing CityGML LOD2 models with a corresponding indoor geometry. <http://pure.tudelft.nl/ws/portalfiles/portal/4615279>.
- Conn, A. R., Gould, N. I., Toint, P. L., 2000. *Trust region methods*. SIAM.
- De Berg, M., Cheong, O., Devillers, O., Van Kreveld, M., Teillaud, M., 1998. Computing the maximum overlap of two convex polygons under translations. *Theory of computing systems*, 31(5), 613–628.
- H, J., 2020. Indoor Reconstruction from Floorplan Images with a Deep Learning Approach. <https://www.mdpi.com/2220-9964/9/2/65>.
- hang Or, S., 2005. Highly Automatic Approach to Architectural Floorplan Image Understanding Model Generation. [https://citeseerx-ist-psu.edu/viewdoc/download?doi=10.1.1.138.4263](https://citeseerx.ist-psu.edu/viewdoc/download?doi=10.1.1.138.4263).
- Har-Peled, 2016. Approximating the Maximum Overlap of Polygons under Translation. <https://link.springer.com/article/10.1007/s00453-016-0152-9>.
- Herle, 2020. How to Obtain Interoperability Between Geospatial and Building Information Modelling? <https://link.springer.com/content/pdf/10.1007/s41064-020-00090-4.pdf>.
- Kendall, A., Gal, Y., Cipolla, R., 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.
- Laat, R., 2010. Integration of BIM and GIS: The Development of the CityGML GeoBIM Extension. https://link.springer.com/chapter/10.1007/978-3-642-12670-3_3.
- Ledoux, H., Ohori, K. A., Kumar, K., Dukai, B., Labetski, A., Vitalis, S., 2019. CityJSON: A compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4(1), 1–12.
- Liu, C., 2017. Raster-to-Vector: Revisiting Floorplan Transformation. <http://art-programmer.github.io/floorplan-transformation.html>.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A. C., 2016. Ssd: Single shot multibox detector. *European conference on computer vision*, Springer, 21–37.
- Milenkovic, 1998. Rotational polygon overlap minimization and compaction. <https://pdf.sciencedirectassets.com/271512/1-s2.0-S0925772100X00296/1-s2.0-S0925772198000121/main.pdf>.
- Ngwepe, L., Aigbavboa, C., 2015. A theoretical review of building life cycle stages and their related environmental impacts. *J. Civil Eng. Environ. Technol*, 2, 7–15.
- Rajat Agarwal, S. C., Sridhar, M., 2016.

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

Sander Oude Elberink, Jantien Stoter, H. L., Commandeur, T., 2013. Generation and Dissemination of a National Virtual 3D City and Landscape Model for the Netherlands. https://3d.bk.tudelft.nl/hledoux/pdfs/13_pers.pdf.

Surikov, I. Y., 2020. Floor Plan Recognition and Vectorization Using Combination UNet, Faster-RCNN, Statistical Component Analysis and Ramer-Douglas-Peucker. https://link.springer.com/chapter/10.1007/978-981-15-6648-6_2.

Tomaschek, M., 2018. Evaluation of off-the-shelf OCR technologies. PhD thesis, PhD Thesis, Masaryk University.

Wu, 2020. Indoor mapping and modeling by parsing floor plan images. <https://www.tandfonline.com/doi/full/10.1080/13658816.2020.1781130>.

Yang, J., 2018. Semantic Segmentation in Architectural Floor Plans for Detecting Walls and Doors. <https://ieeexplore-ieee.org/stamp/stamp.jsp?arnumber=8633243>.

Zeng, Z., 2019. Deep Floor Plan Recognition Using a Multi-Task Network with Room-Boundary-Guided Attention. <https://arxiv.org/pdf/1908.11025.pdf>.