

3D Reconstruction of Building Models based on 2D Floorplans using Statistical Analysis

Pawel Gasiorowski
London Metropolitan University,
London, UK
p.gasiorowski1@londonmet.ac.uk

Vassil Vassilev
London Metropolitan University,
London, UK
v.vassilev@londonmet.ac.uk

Abstract— The study of buildings interiors has been in the center of attention of estates industry. Floor plans are the fundamental way of presenting the spatial arrangement of components found in a building such as rooms, corridors, doors, and windows. With the rapid advancement of image processing, computer vision and graphics technologies over the past few years we are at the point where it is feasible to reconstruct and visualize the interiors of a building in three-dimensional space for several case scenarios. In this paper we focus on approximated recreation of internal structure of a kindergarten building from the suburbs of Sofia, Bulgaria in 3D for future visualization of indoor pollution levels on a basis of data produced by sensors installed on the premises. We use well-established image processing algorithms for extracting features from a 2D floor plan, use statistical methods for their classification and convert them into GEOjson format for subsequent 3D reconstruction in Unity game engine.

Keywords—*Visualization Techniques, Visual Representation and Interaction, Building Modeling, 3D Reconstruction, Floorplans*

I. INTRODUCTION

Architects around the world have been using floorplans to translate their architectural vision into a formal document and subsequently manifest it in a form of a physical structure. Floorplans vary but their main principles remain the same: to accurately present the arrangements of spaces forming integral parts of a building. At the fundamental level they must contain line segments producing room boundaries and universally recognized arc symbols representing doors. Nowadays it is possible to manually recreate the 3D model of a building on a basis of a floorplan that can be utilized in a variety of arbitrary applications such as Augmented Reality (AR) [1], Virtual Reality (VR) [2] and assistive and interactive indoor navigation-based technologies [3] to name a few. However, the downside is that it may take a considerable amount of resources leading to expensive financial cost. The automatic or semi-automatic approach is required.

II. RELATED WORK

The concept of automatic reconstruction of 3D building from a 2D floor plan is not new but remains a challenging task due to number of various details that need to be extracted and correctly interpreted. The lengths and orientations of walls, spatial relationships between rooms or doors and windows locations may have an impact on the reconstruction process.

Ruwanthika, et al. [4] first clean the floorplan image from excessive details such as text labels and symbols with morphological operations so that only double line segments remain. Next, the Harris corner detector [5] is applied to extract

edge corners of line segments. Authors highlight the problem of detecting the inside and outside lines constituting a double thick wall that may result in a noise in the output data. They present their own solution where a single longest line segment is selected per wall to achieve better results.

Zhang et al. [6] perform the shape detection and 3D reconstruction from a grid map that was generated on a basis of a single line handheld radar data in five main steps. The algorithm uses region growing [7] for separating different spatial objects, Hough transform [8] for lines detection and their own clustering methods for the regularization of the floorplan formed out of the detected lines. The research present promising results, especially in noisy and offset documents. The authors assume the Manhattan-world regularities rules [9] that state all adjacent walls are perpendicular to each other, which may not hold in complex structures case.

Gerstweiler [10] converts a vectorized floor plan into a custom graph. It is subsequently traversed to perform template matching of specific architectural symbols and detect objects of interest such as doors, windows, and furniture. The recognition process was evaluated with several floorplans prepared by architects in advance.

Jin et al. [11] bring the idea further by developing a framework for constructing hierarchical semantic graphs composed out of two separate modules encoding both coarse and detailed information of a floorplan. The authors capture high- and low-level visual features and translate them into graph nodes. After that they depict the relationships between different components with a help of graph edges. Although the main goal of the research was not to pre-process the floorplan for 3D model reconstruction, the richness of extracted data can prove to be invaluable during such process.

III. METHODOLOGY

Our algorithm for floorplan optimization and processing focuses on fast extraction of relevant features identifying doors architectural symbols and wall segments that are later used for 3D indoor scene reconstruction. The input is a 2D vectorized image depicting a floorplan. We begin by retrieving all contours from the converted binary image using the topological analysis and border following algorithm [12]. Unlike Hough transform line detector and Harris corner detector, the method additionally outputs the hierarchy of closed polygons that we use for shapes analysis and features extraction such as tracing of the building outline, detecting internal wall line segments and identifying doors arc symbols. The set U contains all detected contours.

A. Doors recognition

We declare $S = \emptyset$ to hold initial candidate doors contours. The generalisation of contours is done to obtain a set simplified shapes using the Douglas-Peucker method [13]. It is executed in iterations with ε parameter, specifying the maximum distance between the original curve and its approximation, set with 10 evenly spaced sample values over an interval between $1e-3$ and $1e-1$. Figure 1 depicts the intermediary polygons obtained at different iteration steps.

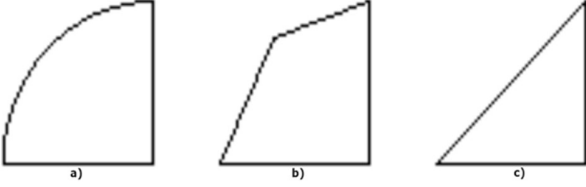


Fig. 1. Contour approximation using Douglas-Peucker algorithm with different ε parameters for door symbol estimation; a) $\varepsilon = 1e-3$ b) $\varepsilon = 4.5e-2$ c) $\varepsilon = 1e-1$.

The measurements of simplified shapes are used to propose candidate doors on a basis of vertices number forming a triangle. Specifically, if approximated contour contains three vertices at any iteration step forming a right-angled triangle in Euclidean space, then it is considered a possible door and pushed to S and removed from U .

After that we calculate summary statistics of candidate doors contours from S to receive threshold γ value representing the common door contour length in the input floorplan (1).

$$\gamma = \begin{cases} (\bar{S} - Md(S)) - (\bar{S} - Md(S) \times \alpha) & \text{if } \frac{3(\bar{S} - Md(S))}{\sigma} > 0 \\ (Md(S) - \bar{S}) - (Md(S) - \bar{S} \times \alpha) & \text{if } \frac{3(\bar{S} - Md(S))}{\sigma} < 0 \end{cases} \quad (1)$$

where $\bar{S}$ is the mean, $Md(S)$ is the median, $3(\bar{S} - Md(S))/\sigma$ is the Pearson's second coefficient of skewness [14]. The σ is the standard deviation of the distribution. The hyperparameter α is used to reduce the final value of γ that may be required in some cases. It must be set to a range of $[0,1]$.

During our experiments we concluded that subtracting around 10% of $\bar{S} - Md(S)$ from $\bar{S} - Md(S)$ in case of positively skewed distribution and 10% of $Md(S) - \bar{S}$ from $Md(S) - \bar{S}$ in case of negatively skewed distribution achieves better fit of the data. To that end setting α to $1e-1$ is recommended.

The number of vertices, frequency of appearances of vertices during iterations and common contour length values are used to filter out all contours from S that may have been mistakenly classified as doors. The true classification of contour o where $\{o\} \in S$ is considered successful if and only if the condition (2) is satisfied in which case the contour o is removed from S and pushed into S' .

$$S' = S' \cup \{o\} \text{ if } o_n = 3 \wedge o_m > 2 \wedge o_{cnt} \leq \gamma \quad (2)$$

where o_n is the number of vertices of approximated contour, o_m is the mode of contour's vertices numbers registered during iterations in the previous step and o_{cnt} is the length of the contour that is checked against γ threshold value calculated in (1).

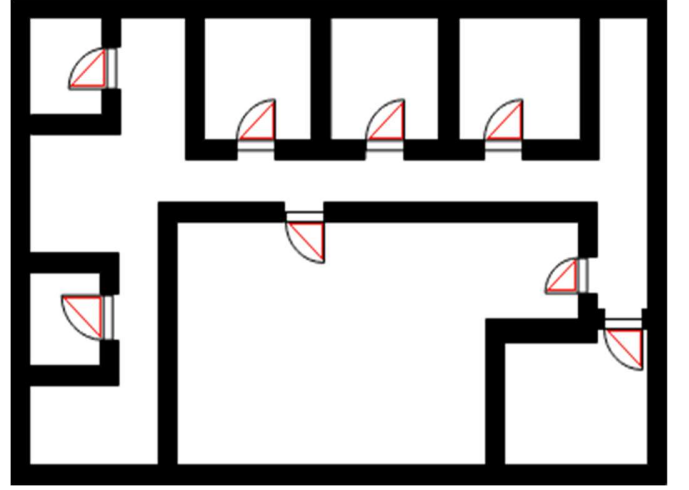


Fig. 2. Identified door shapes in a sample floorplan.

The result set S' containing all identified doors in a sample floorplan is presented in Figure 2. Our experiments suggest that this method can be used in most challenging case scenarios given that input image contains regular vectorized flat shapes.

B. Walls segments extraction

We find the minimum bounding box for each positively identified door shape and add it to the mask image. Next, the bitwise operation is applied to original floorplan image to create a mask suitable for eliminating all positively identified doors. After that we apply the morphological operations of both dilution and erosion [15] in two iterations to remove any surplus pixelated artifacts. The output of the pipeline process is a cleaned floorplan image ready for extracting walls segments as shown in Figure 3.

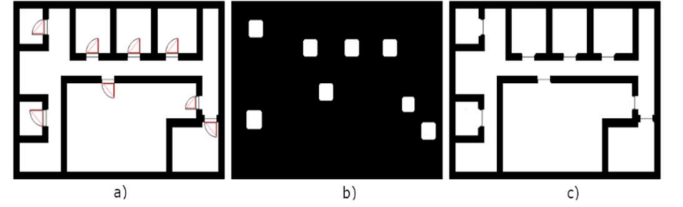


Fig. 3. Detecting and eliminating bounding boxes of doors from original floorplan on a basis of constructed binary mask image; a) identified doors b) mask image containing bounding boxes of detected doors c) cleaned floorplan image

We retrieve and reconstruct a full hierarchy of nested contours in the floorplan. Each contour o has an associated array of four values providing details on its position within the hierarchy. The data structure holds information on next o_{next} and previous o_{prev} contours on the same hierarchical level as well as first child o_{fc} and parent o_p contours residing on the lower and upper levels accordingly [16].

The hierarchy allows us to determine the degree of relationships between different polygons edges representing walls line segments. We have exploited these data to separate the exterior walls from the rest of detected contours.

- E is the set containing all contours extracted after removing doors such that $E = \{o \in U \mid \{o\} \cap S = \emptyset\}$

- W_{ext} is the set containing contours representing exterior walls segments from E such that $W_{ext} = \{o \in E \mid o_{fc} = o_p = -1\}$
- W_{int} is the set containing contours representing interior walls segments from E such that $W_{int} = \{o \in E \mid o \notin W_{ext} \wedge o_p > -1\}$
- For the resulting sets W_{ext} and W_{int} exist several properties such that $W_{ext} \subseteq E \wedge W_{int} \subseteq E \wedge W_{ext} \cap W_{int} = \emptyset$

After that we have applied another morphological transformation of erosion followed by dilation with a 3x3 kernel to remove door lines left in by previous operations.

C. GeoJson

We export the extracted visual features into a geospatial interchange format based on JavaScript Object Notation called GeoJSON. It is a universally recognized format that defines several types of JSON objects and the manner in which they are combined to represent data about geographic features, their properties, and their spatial extents [17].

There are three sections that describe the integrity of the analysed building. We comply with the rules enforced by GeoJSON when producing the file. To that end, we ensure to follow the key-value pairs conventions accordingly.

The first section contains coordinates of vertices forming the polygons of interior walls of rooms on all floors of the building. With the assumption that there are usually multiple rooms inside the building we have set the geometry type to MultiPolygon.

The second section contains coordinates of vertices of the building exterior walls. Because a single building can have only one boundary silhouette, the geometry is set to type Polygon.

The third and last section contains coordinates of the extreme points of the door bases lines. Since there may be multiple doors inside a building, we have set the geometry type to MultiLineString.

```
{
  "type": "Feature",
  "properties": {
    "@id": "way/221178941",
    "building": "kindergarten",
    "building:levels": "3"
  },
  "geometry": {
    "type": "MultiPolygon",
    "coordinates": [
      [
        [
          [ 23,23
            ...
          ],
          [
            [ 183,117
              ...
            ]
          ]
        ]
      ]
    ]
  },
  "type": "Feature",
```

```
"properties": {
  "@id": "way/221178941",
  "building:exterior": "silhouette"
},
"geometry": {
  "type": "Polygon",
  "coordinates": [
    [
      [ 10,11
        ...
      ]
    ]
  ]
},
{
  "type": "Feature",
  "properties": {
    "@id": "way/221178941",
    "building:interior": "doors"
  },
  "geometry": {
    "type": "MultiLineString",
    "coordinates": [
      [
        [ 338,181
          ...
        ],
        ...
      ]
    ]
  }
}
```

All coordinates are rounded since we are interested in the approximated version of the 3D building for the purpose of indoor pollution visualisation.

D. 3D building reconstruction

The output from the previous stage is the input to Unity application for 3D reconstruction and modelling. We deserialize the file using JSON framework for .NET from the popular Newtonsoft library [18] as it allows keys to contain special characters that are inseparable elements of GeoJSON describing geographic features, e.g.: @id declaring a unique ID for a feature. The plane figures such as polygons and lines are stored in data structures as list of vectors, where each vector has three components -x, -y and -z describing its location in 3D space.

Next, the reconstruction process is initiated. We find the centre point p_i between shared endpoints v^i and v^{i+1} of consecutive segments in the polygonal chain (3). Every point is projected onto XZ axes. We then add a primitive cube at the location p_i and scale it according to building measurements (4).

$$p_i = \begin{cases} [v_x^i + v_x^{i+1}/2, 0, v_z^i + v_z^{i+1}/2] & \text{if } i < n - 1 \\ [v_x^0 + v_x^{n-1}/2, 0, v_z^0 + v_z^{n-1}/2] & \text{otherwise} \end{cases} \quad (3)$$

$$p_s' = [p_x', p_y', p_z'] \quad (4)$$

$$p_z^{s'} = \sqrt{(v_x^{i+1} - v_x^i)^2 + (v_y^{i+1} - v_y^i)^2 + (v_z^{i+1} - v_z^i)^2}$$

where $p_x^{s'}$ is the wall thickness, $p_y^{s'}$ is the wall height and $p_z^{s'}$ is the length of the wall segment.

After setting up the location and scaling the wall segment we compute the quaternion [19] q_r representing the orientation of the cube p_{rot} towards a normalized directed vector calculated as $w = (v^{i+1} - v^i)/p_z^{s'}$ in 3D space (5).

$$p_{rot} = \begin{cases} \alpha \hat{c}_r q_r & \text{if } \|c_r\| > 1e-4 \\ q_r & \text{otherwise} \end{cases} \quad (5)$$

where

$$\begin{aligned} \alpha &= \theta_{v_r u} \\ v_r &= q_r u \\ c_r &= v_r \times u \end{aligned}$$

and $\alpha \hat{c}_r$ is the quaternion representing the rotation around an axis determined by normalized perpendicular vector $\hat{c}_r$ by angle α .

The quaternion q_r representing the rotation aligning the input vector towards the desired direction is calculated with (6).

$$q_r = (\sqrt{1 + (w \cdot f)}, w \cdot r, w \cdot u, w \cdot f) \quad (6)$$

where

$$\begin{aligned} f &= (0, 0, 1) \\ r &= (1, 0, 0) \\ u &= (0, 1, 0) \end{aligned}$$

The process of reconstructing doors is similar as they are of type `MultiLineString` inside GeoJSON. The main difference is that unlike walls segments, doors can only consist of two extreme points, i.e.: only one straight line segment.

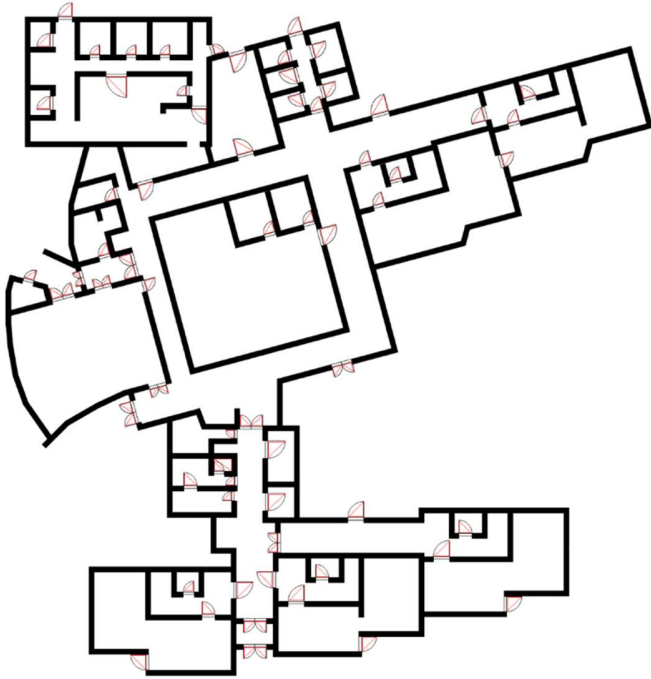


Fig. 4. The 2D processed digital floorplan of kindergarten from the suburbs of Sofia, Bulgaria.

RESULTS

The reconstruction pipeline was used in two stages to create a 3D model of kindergarten from the suburbs of Sofia, Bulgaria. First, we have created a simplified 2D digital floorplan of the building as shown in Figure 4. After that our processor extracted

the relevant visual features such as doors and exported them into GeoJSON file. The file was supplied as an asset to a Unity game project. We parsed the contents of the file and reconstructed every element in a loop using the built-in primitive cube geometry as shown in Figure 5.

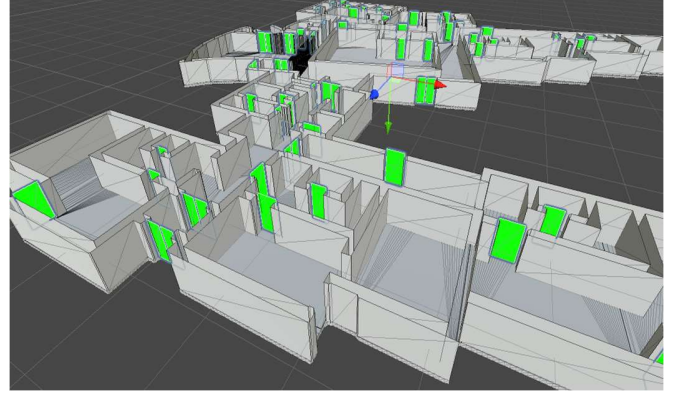


Fig. 5. The reconstructed 3D model of a kindergarten building from the suburbs of Sofia, Bulgaria on the basis of processed digital floorplan and generated GeoJSON file.

CONCLUSION

In this paper we have proposed and demonstrated a fast 3D building approximation reconstruction tool for the purpose of visualizing indoor pollution levels. We have divided the process into two separate stages. At feature extraction stage we capture relevant 2D features such as wall segments and doors and convert them into GeoJSON format for subsequent parsing. The 3D model reconstruction stage is concerned with parsing the previously generated file and using primitive geometries to automatically create a building. The digital floorplans are processed inside a pipeline utilizing state-of-the-art computer vision algorithms widely adopted within the industry, academia, and community.

It has been shown that the data enrichment by generating 3D feature descriptors from 2D floorplans is sufficient for a successful approximation of the building in 3D environments such as Unity game engine. As part of future work, we want to incorporate image processing algorithms for recognizing additional symbols common to the architectural blueprints such as windows, furniture and so on.

ACKNOWLEDGEMENT

The work reported here has been conducted with the financial support from the Bulgarian government (OPNOIR Grant Agreement No. BG05M2OP001-1.003-0002-C01) and HEIF Fund – UK.

REFERENCES

- [1] Azuma, R.T., 1997. A survey of augmented reality. *Presence: teleoperators & virtual environments*, 6(4), pp.355-385.
- [2] Zheng, J.M., Chan, K.W. and Gibson, I., 1998. Virtual reality. *Ieee Potentials*, 17(2), pp.20-23.
- [3] Tian, X.; Shen, R.; Liu, D.; Wen, Y.; Wang, X. Performance analysis of RSS fingerprinting based indoor localization. *IEEE Trans. Mob. Comput.* 2016, 16, 2847–2861.
- [4] Ruwanthika, R.G.N., Amarasekera, P.A.D.B.M., Chandrasiri, R.U.I.B., Rangana, D.M.A.I., Nugaliyadde, A. and Mallawarachchi, Y., January. Dynamic 3D model construction using architectural house plans. In 2017

- 6th National Conference on Technology and Management (NCTM) (pp. 181- 184). IEEE. 2017
- [5] Derpanis, K.G., 2004. The harris corner detector. *York University*, vol. 2, pp.1-2.
- [6] Zhang, X., Fang, Z., Lu, Z., Xiao, J., Cheng, X. and Zhang, X., 2021, May. 3D reconstruction of weak feature indoor scenes based on hector SLAM and floorplan generation. In 2021 IEEE 7th International Conference on Virtual Reality (ICVR) (pp. 117-126). IEEE.
- [7] Hojjatoleslami, S.A. and Kittler, J., 1998. Region growing: a new approach. *IEEE Transactions on Image processing*, 7(7), pp.1079-1084.
- [8] Hough, P.V., 1962. Method and means for recognizing complex patterns. U.S. Patent 3,069,654.
- [9] Coughlan, J. and Yuille, A.L., 2000. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. *Advances in Neural Information Processing Systems*, 13.
- [10] G. Gerstweiler, "Guiding People in Complex Indoor Environments Using Augmented Reality," *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, Tuebingen/Reutlingen, Germany, 2018, pp. 801-802, doi: 10.1109/VR.2018.8446138.
- [11] J. Jin, Z. Xue and B. Leng, "SHRAG: Semantic Hierarchical Graph for Floorplan Representation," *2022 International Conference on 3D Vision (3DV)*, Prague, Czech Republic, 2022, pp. 271-279, doi: 10.1109/3DV57658.2022.00039.
- [12] Suzuki, S., 1985. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, 30(1), pp.32-46.
- [13] Visvalingam, M. and Whyatt, J.D., 1990, September. The Douglas-Peucker algorithm for line simplification: re-evaluation through visualization. In *Computer graphics forum* (Vol. 9, No. 3, pp. 213-225). Oxford, UK: Blackwell Publishing Ltd.
- [14] Doane, D.P. and Seward, L.E., 2011. Measuring skewness: a forgotten statistic?. *Journal of statistics education*, 19(2).
- [15] Goyal, M., 2011. Morphological image processing. *IJCST*, 2(4), p.59.
- [16] Bradski, G. and Kachler, A., 2008. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc."
- [17] Butler, H., Daly, M., Doyle, A., Gillies, S., Hagen, S. and Schaub, T., 2016. *The geojson format* (No. rfc7946).
- [18] Newtonsoft, 2023. Popular high-performance JSON framework for .NET. Available at: <https://www.newtonsoft.com/json> (Accessed: 16 August 2023).
- [19] Rose, D., 2015. Rotation Quaternions and How to Use Them. Available at: <https://danceswithcode.net/engineeringnotes/index.html> (Accessed: 17 August 2023)