Supplementary Materials: MSD: A Benchmark Dataset for Floor Plan Generation of Building Complexes

1 Dataset of MSD

1.1 Categorization and ID nesting

The Swiss Dwellings dataset (SD) [8] is stored as a large dataframe[1]. Each row in the dataframe corresponds to a geometrical detail *e.g.* living room, wall element, or balcony. The entity has a *type* and, further nested, *subtype* categorization (which form two of the columns in the dataframe):

- The **"feature"** type includes the following subtypes: "washing machine", "shower", "bathtub", "kitchen", "elevator", "built in furniture", "stairs", "toilet", "sink", "ramp".
- The **"separator"** type includes the following subtypes: "wall", "railing", "column".
- The **"opening"** type includes the following subtypes: "entrance door", "window", "door".
- The **"area"** type includes the following subtypes:: "radiation therapy", "office", "corridors and halls", "wintergarten", "salesroom", "studio", "open plan office", "outdoor void", "electrical supply", "workshop", "physio and rehabilitation", "living dining", "not defined", "shaft", "carpark", "corridor", "air", "dedicated medical room", "office space", "water supply", "garage", "medical room", "elevator", "balcony", "sanitary rooms", "staircase", "vehicle traffic area", "cold storage", "meeting room", "living room", "factory room", "showroom", "oil tank", "office tech room", "bedroom", "foyer", "room", "patio", "teaching room", "elevator facilities", "logistics", "garden", "canteen", "community room", "gas","operations facilities", "storeroom", "lobby", "shelter", "cloakroom", "technical area", "dining", "warehouse", "basement compartment", "loggia", "reception room", "bathroom", "basement", "common kitchen", "pram and bike storage room", "bike storage", "break room", "house technics facilities", "lightwell", "counter room", "transport shaft", "wash and dry room", "terrace", "arcade", "waiting room", "void", "heating", "kitchen", "sports rooms", "pram", "kitchen dining", "archive".

Here, the blue indicates which subtype category names are shared between the "feature" and "area" types.

[1] A dataframe is defined as a two-dimensional data structure, for which the naming is borrowed from the Pandas library in Python.

In addition to the type and subtype categories, each entity contains metadata about the relation to other entities. This relation is resembled by the nested positioning of the entities across different sites, buildings, floors, apartments, and units. The most high-level positional identifier (ID) is the *site* ID which tells you on which site the entity is located. A site could, for instance, be a set of buildings in the same neighborhood (in which the different buildings are likely to share similar characteristics). Second is the *building* ID which tells you in which building the entity is located. Third is the *plan* ID which corresponds to the floor plan layout prototype that the entity is part of. Fourth is the *floor* ID which corresponds to a particular floor at a specific elevation in a building. It is noteworthy to mention that different floors can originate from the same plan ID. Fifth is the *apartment* ID which tells you from which apartment the entity originates. It is important to note that the apartment ID is shared across different floors in the case of multi-story apartments *i.e.* apartments that stretch across multiple levels. Sixth and final is the *unit* ID which indicates from which apartment the entity originates. In contrast to the apartment ID, the unit ID is different for each floor.

Type, subtype, geometry, site ID, building ID, plan ID, floor ID, apartment ID, and unit ID define – besides other meta-level information such as elevation – the columns of the dataframe. The geometry is defined as a polygon, formatted as *well-known text* (WKT).

1.2 Filtering details

For filtering and cleaning, we follow the steps provided in Section **??**. Some details that were not mentioned specifically in the main text are provided below:

- **Feature removal.** All entities that are a "feature" (see Sec. 1.1) are removed entirely from the dataframe.
- **Residential-only filtering.** All floor plans that contain at least one entity for which the subtype category is not to be found in residential buildings (the subtypes indicated in red in Sec. 1.1) are removed from the dataframe.

In addition to the filtering steps above, we remove floor plans that have too many small disconnected parts. Specifically, we remove all floor plans that have 2 or more areas that are fully disconnected in the room graph (read: that are "floating" in the floor plan); removing an extra 388 (2.8%) floor plans.

1.3 Image extraction

The coordinates, x (east-to-west) and y (south-to-north), of the geometries in the dataframe are defined in meters and are usually centered around $(x, y) = (0, 0)$ for a given floor plan. To retain the information of the original coordinates within the image, two extra channels are added on top of the image canvas representing x and y. The mappings from x and y to the corresponding pixel locations x_i and

y_i (both defined on $[0, 512]$) for a given image size s (assumed to be square) are given by:

$$x_i = \left(x - x_{\min} + 0.5\,[\Delta_{yx}]_+\right) \cdot \frac{s}{\max\left(\Delta_x, \Delta_y\right)}, \tag{1}$$

$$y_i = \left(y - y_{\min} + 0.5\,[\Delta_{xy}]_+\right) \cdot \frac{s}{\max\left(\Delta_x, \Delta_y\right)}, \tag{2}$$

where $\Delta_x = x_{\max} - x_{\min}$ is the 'width' of the floor plan, $\Delta_y = y_{\max} - y_{\min}$ is the 'height' of the floor plan, $\Delta_{xy} = \Delta_x - \Delta_y$ and $\Delta_{yx} = -\Delta_{xy}$ are the relative differences between width and height, and $[\cdot]_+ = \max(0, \cdot)$. The red part maps all coordinate values above 0; the green part makes sure to put the floor plan in the middle of the square that starts at $(0, 0)$ and extends to $(\max(\Delta_x, \Delta_y), \max(\Delta_x, \Delta_y))$; the blue part makes sure to scale the square to the image domain.

1.4 Statistics

With a total of 5.3K+ floor plans, containing 18.9K+ units, and covering 165.3K+ areas, MSD is one of the few publicly available large-scale floor plan datasets[2]. Fig. 1 shows the room and unit distributions for MSD.

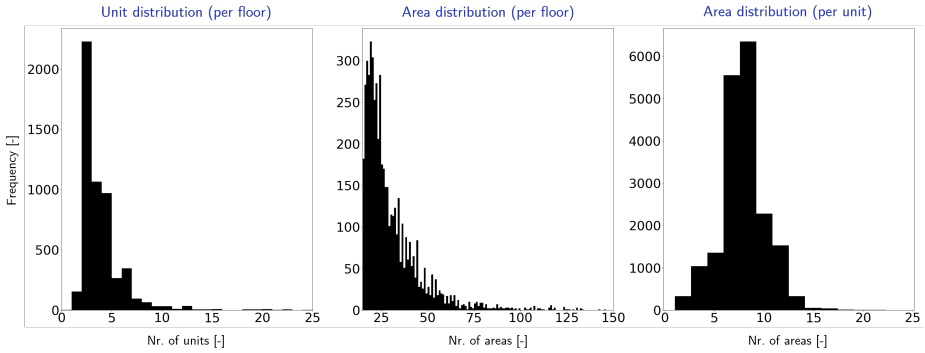

Fig. 1: Area and unit distributions MSD. The unit distribution per floor (right), area distribution per floor (middle), and area distribution per unit (right) are plotted as histograms. The x-axis specifies the number of units or areas, and the y-axis specifies the frequency. From the unit distribution plot, it is apparent that MSD comprises mostly floor plans that contain between 2 to 9 units. MSD comprises mostly floor plans that have between 15 and 50 areas, with a peak of around 25. The area distribution per unit is similar to RPLAN [11] and LIFULL [4], ranging between 3 and 15 areas per unit and a median around 7 areas per unit.

[2] For details on the sizes of the other floor plan datasets, please refer to [6].

2 Graph extraction from generated outputs

In our work, we have set the floor plan generation task to predict only walls and areas – not doors. Therefore, we cannot reliably extract the room graph, $\hat{g}_\mathrm{r}$[3], from the predicted floor plan because the edge formation critically depends on the door locations. Instead, we extract the adjacency graph, $\hat{g}_\mathrm{a}$, from the predicted floor plan in which the edges are formed when geometries are close enough. Note that the set of nodes of $\hat{g}_\mathrm{r}$ and $\hat{g}_\mathrm{a}$ are equivalent, and that the set of edges can be different.

For the graph extraction strategy, we use an algorithmic approach to extract $\hat{g}_\mathrm{a}$. Specifically, we assign an edge between a pair of nodes if and only if the minimum distance between the areas of that pair does not exceed a preset maximum distance, which is referred to as the buffer distance. For an image size of 512, we set the buffer distance to 5. Note that when two areas are overlapping, the minimum distance is 0, hence an edge is formed between overlapping areas.

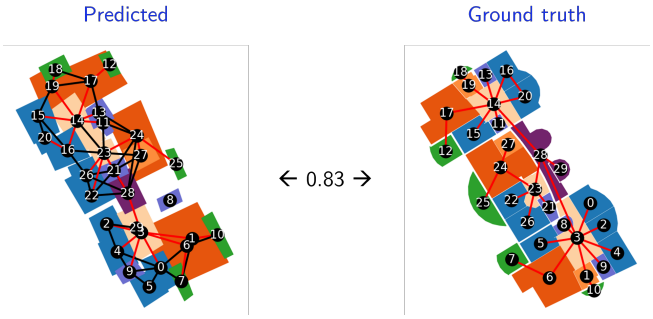

Fig. 2: Graph compatibility computation. Left: predicted floor plan including $\hat{g}_\mathrm{a}$. Right: ground truth floor plan including g_r. The set of edges in g_r that is retained in $\hat{g}_\mathrm{a}$ is colored in red. The node correspondence is made visual by enumerating the nodes graphically. The graph compatibility is computed by dividing the amount of red edges from $\hat{g}_\mathrm{a}$ by the total amount of edges in g_r, equaling $25/30 = 0.83$.

Access connectivity implies adjacency, but adjacency does not necessarily imply access connectivity, which means that the g_r is a subgraph of g_a. Essentially, the graph compatibility reflects to what extend g_r is contained in $\hat{g}_\mathrm{a}$, which is done by computing the ratio of the amount of the edges from g_r that are retained in $\hat{g}_\mathrm{a}$ with respect to the total amount of edges in $\hat{g}_\mathrm{a}$ (see Eq. (**??**) for details, and Fig. 2 for a visual elaboration).

For graph-based approaches *e.g.*, MHD Section **??**, the process of extracting graphs from predicted layouts can also introduce inaccuracies, particularly

[3] We use $\hat{g}$ to refer to the graph of the predicted floor plan and g to the graph of the ground truth floor plan. Additionally, the subscripts r and a refer to 'room' and 'adjacency' graph types.

in cases where the predicted layout encompasses numerous overlapping rooms. Nonetheless, our analysis indicates that these instances are infrequent: rooms overlap on average (for MHD) with 4.11 ± 2.25 other rooms. A typical prediction is given in Fig. 2. Therefore, we consider the algorithmic extraction of room edges to be a justifiable method for extracting the room graphs.

3 Modified HouseDiffusion (MHD)

3.1 Node classification with GAT

We train a GAT model [10] to predict the room graph given the zoning graph. The room and zoning graph are isomorphic, and thus this prediction task can be formulated as a node classification problem.

The GAT model consists of several stacked graph attention convolutional (GATConv) layers. The input to the model consists of the zoning type for each node, as well as the door type for each edge. The output of the last GATConv is concatenated with the initial node features, and subsequently fed into the final linear layer that maps the concatenated feature vectors to the correct output dimension for predicting the room types. Between each hidden layer, a ReLU is used as an activation function. We use dropout for regularization and use the Adam optimizer. The amount of GATConv layers is set to 5, the hidden sizes of each GATConv to 64, the learning rate to 0.001, and the batch size to 128. Including early stopping, for this setting the validation accuracy is 0.893.

3.2 Minimum rotated rectangle approximation (MRR)

To be able to represent each area with a fixed number of corners, we propose to take the minimum rotated rectangle of each area. The minimum rotated rectangle of an area is the rotated rectangle that fully encloses the area polygon with minimal area. Approximating the areas of a floor plan by their minimum rotated rectangle (MRR) works best when drawing the area rectangles in order from largest to smallest, such that small areas occlude larger areas (see Fig. 3 for a visual clarification).

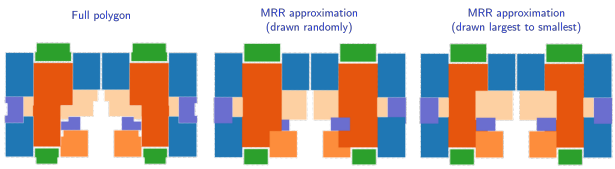

Fig. 3: MRR. Left: the original floor plan (containing polygons with arbitrarily many corners). Middle: the result of applying MRR, but drawn in random order. Right: the result of applying MRR and drawing the largest to the smallest area.

3.3 Diffusion model

We borrow the model architecture of *HouseDiffusion* (HD) [7], and extend it to suite MSD. The attention layer in the transformer model is modified by adding cross attention between room corners and wall segments. Additionally, the relational cross attention (RCA) as defined in HD is modified to incorporate the edge attributes as well. RCA is modified because, in contrast to RPLAN in which areas are solely connected by a door, edges in MSD have a "connectivity" attribute representing the type of connectivity being "door", "front door", or "passage".

We set the batch size to 32 and trained for 300k steps. Other hyperparameters are left the same as in HD implementation.

3.4 Building structure pre-processing

To be able to effectively condition MHD on walls, we first convert the binary image of the building structure to a set of straight lines. The line elements are extracted from the binary image of the building structure by following the steps below:

- **Morphological thinning.** We start by morphological thinning of the binary image of the building structure. Morphological thinning (see page 671 in [1], and an overview of thinning techniques in [3]) essentially creates a new binary image in which line thicknesses are reduced to a minimum (ideally one pixel). The resulting binary image is a skeletonized version of the original version.
- **Skeleton network extraction.** From the skeletonized image we extract the skeleton network graph in which nodes represent joints and corners of the skeleton, and edges the geometry of the curves between the nodes.
- **Simplify skeleton network into set of lines.** The edges of the skeleton network graph, which contain the geometry of the curves between two nodes, are converted to a set of straight lines.

Fig. 4 visualizes the processing steps of the line extraction algorithm.

3.5 Wall-cross attention (WCA)

Each wall element w_i extracted by the line extraction algorithm is a vector that represents the start and end points of the line. Similar to HD, we augment w_i by uniformly sampling 7 points between the start and end points. Equivalent to the corner embeddings in HD, a single-layer MLP embeds the 4-D w_i into a 512-D embedding vector: $\hat{w}_i = \mathrm{MLP}_{\mathrm{w}}(\mathrm{AUW}(w_i))$, in which AUW is the sampling function (similarly named as in HD). Note that the wall elements do not get updated during the denoising process.

The wall embeddings are used as additional input to MHD. In the original model, the attention layer consists of three types of masked attention with room

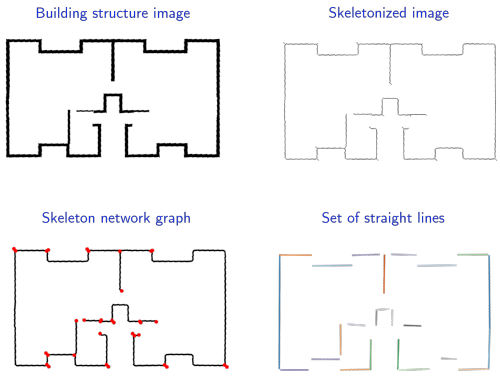

Fig. 4: Line element extraction. Top-left: building structure as a binary image in which black indicates the necessary structural elements of the building. Top-right: a skeletonized building structure in which all the walls are reduced to one-pixel width. Bottom-left: a skeleton network graph in which the nodes (joints and corners) are red dots. Bottom-right: a simplified set of straight lines approximating the complete building structure as a set of straight-line elements.

corner embeddings. Specifically, the attention module in MHD is modified by adding an extra cross-attention operation between all room corner embeddings and wall embeddings, referred to as wall cross attention (WCA). The room corners are used as a query, and the wall embeddings as keys and values. All attention operations in the attention layer are summed together. Fig. 5 provides a zoomed-in version of Fig. **??**.

3.6 RCA with door type embedding

We modify the RCA module to discern between different connectivity types. Standard doors, passages, and front doors are each assigned a unique learned embedding. The RCA attention is applied separately for each door type, with the attention mask modified to only act on room corners connected by the specified type. On each application, the room corner embeddings are modified when used as keys and values by summing with the embedding of the door type.

4 Graph-informed U-Net

4.1 Visual explanation of model.

Fig. 6 shows the architecture of the U-Net model coupled with the GCN. While the U-Net learns a representation for the building structure, the GCN learns a representation for the zoning graph. The two representations are concatenated and simultaneously upsampled by the decoder of the U-Net, outputting the floor plan as a segmented image with the same resolution as the building structure.

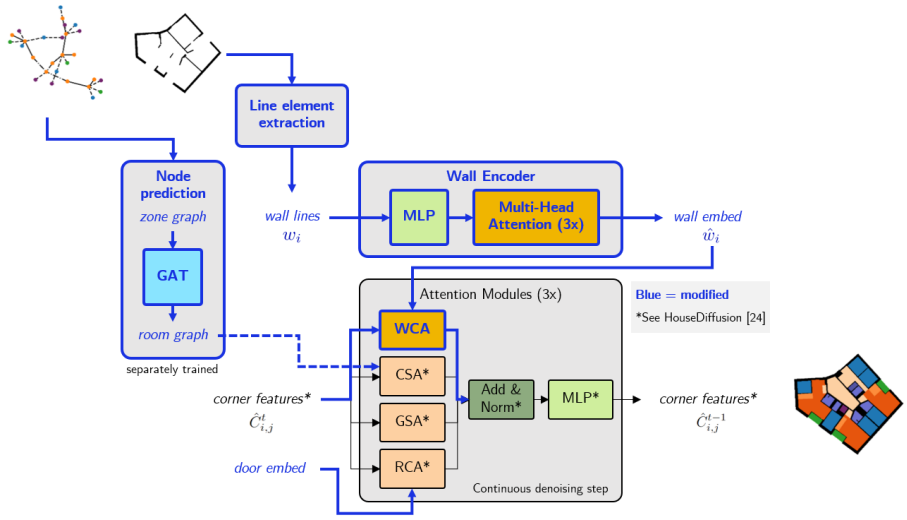

Fig. 5: Modified HouseDiffusion (MHD). A wall encoder is used to map the pre-processed building structure into corresponding wall embeddings. MHD expands HD [7] by introducing an extra attention module (WCA) between the wall embeddings from and corner features of the rooms. A GAT is separately trained to predict the room types from the zoning types, which are used to "color" the full layout.

4.2 Boundary pre-processing

Initially, the building structure's binary image consists purely of the structural necessary components: black ("0") for structure and white ("1") for non-structure. To better guide the model, we use Segment Anything [2] to predict the interior and exterior of the floor plans and explicitly input that information as well. Before we use Segment Anything, the binary images is substantially padded with extra pixels (white pixels). The padding ensures that the segmentation algorithm can reliably infer the largest area as the exterior, even in cases where the building structure is not completely closed. Once the masks are created, the largest mask is selected as background. The pre-processed image contains the following channels:

1. **"In-wall-out".** This channel marks the interior of the building as '1', the boundaries as '0.5', and the exterior as '0'.
2. **"In-out".** This channel marks the interior of the building as '1' and the exterior as '0', focusing on distinguishing between the interior and exterior spaces without structural details.
3. **"Raw-boundary".** This channel contains the original building structure.

4.3 Model, training, and evaluation details

The encoder of the U-Net comprises four convolutional layers, each with layers that double the channel dimensions from 64 to 512 (64 → 128 → 256 → 512). The convolutional layers all consist of (in order): 3x3 convolution, batch norm, ReLU, and 2x2 maxpool. The GCN consists of a stack of several graph convolutional (GConv) layers, each with a hidden feature size of 256. Global mean pooling is used to compute a graph-level feature vector of size 256. We use the Adam optimizer, and we use the cross-entropy loss.

We found the following optimal settings during training: the amount of GConv layers is 2, a learning rate equal to 0.001, the batch size is 16, and the hidden sizes of each GConv layer are 256.

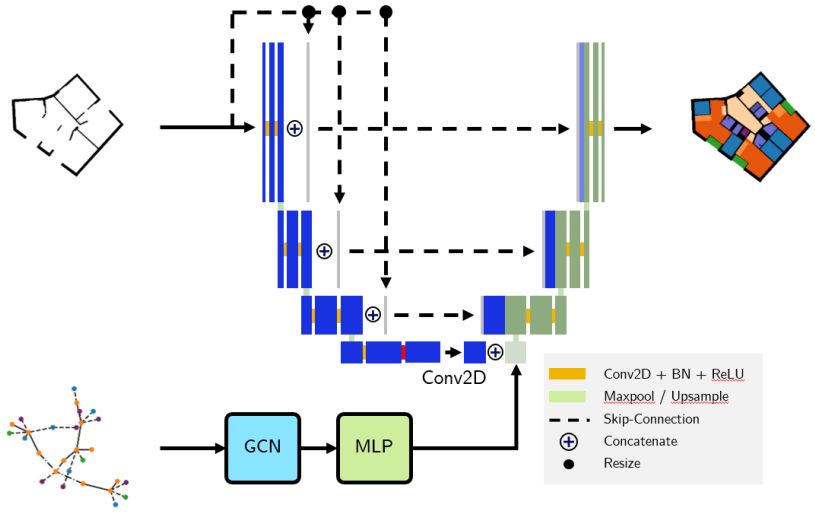

Fig. 6: Graph-informed U-Net (UN). UN takes the building structure (image) as input to the U-Net. The U-Net is composed of an encoder and decoder using the conventional up- and down-sampling 2D convolutions, resp., and includes skip connections between the encoder and decoder feature maps at equivalent feature map scales. A GCN is used to map the zoning graph to a feature vector which is concatenated to the latent space of the U-Net.

5 Additional experiments

5.1 Extra baselines: HouseGAN++ and FLNet

We also ran and evaluated *FLNet* [9] and *HouseGAN++* [5] (See Table. 1). Both required re-purposing to make them applicable to the task we set. All hyperparameters, besides those stated in Table. 1, are equivalent to those in the original

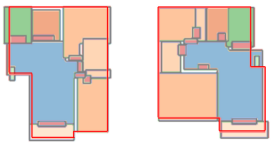

Fig. 7: MHD on RPLAN. Two example predictions of MHD on RPLAN [11]. The generated layouts follow the boundary reasonably well. The input graphs in both cases are equivalent, showing that the model can cope with a wide variety of differently-shaped boundaries.

publications. These additional methods do not perform well, demonstrating the need for our proposed dataset with more realistic building complexes.

Table 1: HouseGAN++: 128 x 128 masks, 388k steps, learn. rates: 1e-5 generator, 4e-5 discriminator, structural masks as input. **FLNet**: 128 x 128 masks, 50 epochs. The scores are averaged over all floor plans in the test set. User studies are done for MHD and U-Net: 7 architects, each 50 random IDs. **Topology**: whether the organization of the spaces makes sense. **Proportions**: whether the room proportions make sense. Scoring: {"yes": 1, "unsure": 0.5, "no": 0}.

	MIoU (↑)	Compatibility (↑)	Topology (↑)	Proportions (↑)
FLNet	19.3	n.a.	n.a.	n.a.
HouseGAN++	11.6	64.2	n.a.	n.a.
MHD	21.8	76.2	0.461 ± 0.138	0.514 ± 0.143
U-Net	42.4	n.a.	0.439 ± 0.148	0.371 ± 0.171

5.2 MHD on RPLAN

Not surprisingly, we successfully trained MHD on RPLAN [11], with seemingly similar performance to HD. To train MHD on RPLAN, we extract the boundary of the layouts first. The boundary (as a set of walls) and room graph serve as inputs to MHD. Similar to HD, doors are also predicted (dark red and light green for interior and front doors, resp., in Fig. 7). Further training details are equivalent to training on MSD. Two typical examples are shown in Fig. 7.

5.3 Evaluating complexity

To better evaluate the complexity, qualitative evaluation (besides the important instrumental measures) will play an essential role. We are actively researching the evaluation methods for topologically more complex floorplans, and some preliminary results of our study are shown in Table. 1 (right). Nonetheless, we

believe that *both* quantitative as well as qualitative measures play an important role.

References

1. Gonzalez, R., Woods, R., Masters, B.: Digital Image Processing (3rd Ed.). J. Biomed. Opt. (2009)
2. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment Anything. http://arxiv.org/abs/2304.02643 (2023)
3. Lam, L., Lee, S.W., Suen, C.: Thinning Methodologies: A Comprehensive Survey. IEEE Trans. Pattern Anal. Mach. Intell. **14** (1992)
4. Ltd., L.C.: LIFULL Home's - A Large-scale Database of Residential Floor Plans. https://www.nii.ac.jp/dsc/idr/en/lifull/ (2015)
5. Nauata, N., Hosseini, S., Chang, K.H., Chu, H., Cheng, C.Y., Furukawa, Y.: House-GAN++: Generative Adversarial Layout Refinement Network towards Intelligent Computational Agent for Professional Architects. In: IEEE Conf. Comput. Vis. Pattern Recog. (2021)
6. Pizarro, P.N., Hitschfeld, N., Sipiran, I., Saavedra, J.M.: Automatic Floor Plan Analysis and Recognition. Automation in Construction **140** (2022)
7. Shabani, M.A., Hosseini, S., Furukawa, Y.: HouseDiffusion: Vector Floorplan Generation via a Diffusion Model with Discrete and Continuous Denoising. In: IEEE Conf. Comput. Vis. Pattern Recog. Vancouver, BC, Canada (2023)
8. Standfest, M., Franzen, M., Schröder, Y., Medina, L.G., Hernandez, Y.V., Buck, J.H., Tan, Y.L., Niedzwiecka, M., Colmegna, R.: Swiss Dwellings: A Large Dataset of Apartment Models Including Aggregated Geolocation-based Simulation Results Covering Viewshed, Natural Light, Traffic Noise, Centrality and Geometric Analysis. https://zenodo.org/records/7788422 (2022)
9. Upadhyay, A., Dubey, A., Arora, V., Kuriakose, S.M., Agarawal, S.: FLNet: Graph Constrained Floor Layout Generation. In: Int. C. Multimedia Expo Worksh. (2022)
10. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks. In: Int. Conf. Learn. Represent. (2018)
11. Wu, W., Fu, X.M., Tang, R., Wang, Y., Qi, Y.H., Liu, L.: Data-driven Interior Plan Generation for Residential Buildings. ACM Trans. Graph. **38** (2019)