

# Extreme Floorplan Reconstruction by Structure-Hallucinating Transformer Cascades

Sepidehsadat Hosseini  
Simon Fraser University  
sepidh@sfu.ca

Yasutaka Furukawa  
Simon Fraser University  
furukawa@sfu.ca

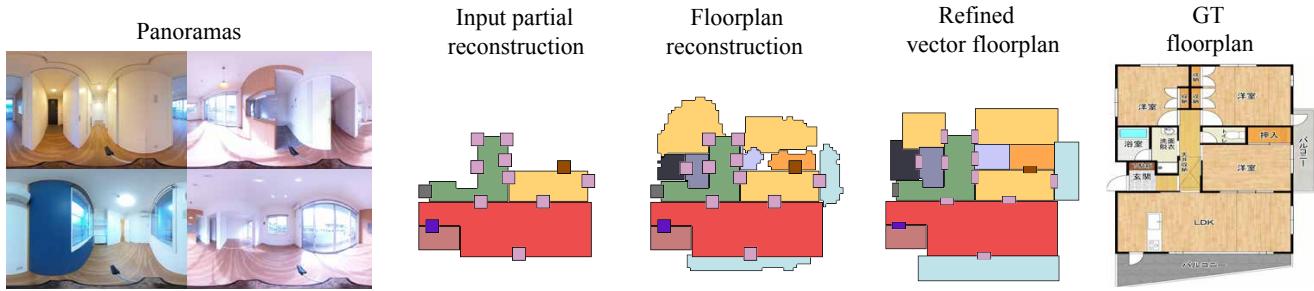


Figure 1. The paper presents a new extreme floorplan reconstruction task. An input is a partial floorplan inferred or curated from panorama images of a house/apartment. An output is a complete floorplan. A technical challenge is the hallucination of invisible rooms and doors.

## Abstract

*This paper presents an extreme floorplan reconstruction task, a new benchmark for the task, and a neural architecture as a solution. Given a partial floorplan reconstruction inferred from panorama images, the task is to reconstruct a complete floorplan including invisible architectural structures. The proposed neural network 1) encodes an input partial floorplan into a set of latent vectors by convolutional neural networks and a Transformer; and 2) reconstructs an entire floorplan while hallucinating invisible rooms and doors by cascading Transformer decoders. Qualitative and quantitative evaluations demonstrate the effectiveness of our approach over the benchmark of 701 houses, outperforming the state-of-the-art reconstruction techniques. A project website with supplementary document and code is [here](#).*

## 1. Introduction

Indoor panorama photography is exploding. Pioneered by Ricoh Theta, consumer-grade panorama cameras are prevalent on the market, whose applications range from real estate to entertainment and surveillance.

Ricoh Theta cameras alone have collected 100 million panoramas from residential houses. This indoor panorama collection allows house renters/buyers/realtors to browse immersive 360 views for tens of millions of houses.

However, a panorama collection is extremely sparse (*i.e.*, one panorama per room with little visual overlaps) and even leaves some rooms invisible, posing fundamental challenges for existing techniques to enable more advanced applications.

This paper presents a new extreme floorplan reconstruction task, a benchmark, and a solution, which could exploit 100 million panoramas to create floorplans for tens of millions of houses. The potential applications range from real estate and construction industries (*e.g.*, building code verification and property value assessment) to virtual and augmented reality. Concretely, the task is to take a partial floorplan inferred or manually curated from panorama images, and reconstruct a complete floorplan. The technical challenge lies in the reconstruction or “hallucination” of invisible rooms and doors. Inferring invisible image or geometry data has been studied in computer vision in the context of image inpainting [29], illumination inference [18], and surface reconstruction [25]. To our knowledge, this paper is the first to tackle the hallucination of architectural components, such as rooms and doors, at the scale of an entire house.

Our contribution is three fold: 1) A new extreme floorplan reconstruction task; 2) A benchmark consisting of 701 houses with ground-truth vector floorplan images; and 3) A neural architecture, whose cascading Transformer decoders reconstruct an entire floorplan, including invisible rooms/doors. Qualitative and quantitative evaluations demonstrate the effectiveness of our approach over the ex-

isting techniques. We will share our code, models, and data.

## 2. Related Work

We review related work in 1) floorplan reconstruction, 2) indoor panorama and floorplan datasets, 3) content hallucination, and 4) extreme pose estimation.

**Floorplan reconstruction:** Floorplan reconstruction has a long history in image processing research with strong ties with real-estate and construction industries. With the success of commodity depth sensors, the current methods take 3D point-clouds and reconstruct floorplan images, often with a combination of deep neural networks and optimization [10, 15, 19]. The emergence of panorama cameras meets growing demand in floorplan reconstruction from images alone [2]. Zillow indoor dataset provides panorama images and ground-truth floorplans, which have been reconstructed with manual interactions [6]. This paper tackles fully automated floorplan reconstruction from production data collected by real users with incomplete and sparse coverage.

**Indoor panorama and floorplan datasets:** SUN360 is a popular indoor panorama dataset [24]. For floorplans, RPLAN provides sixty thousand synthetic samples in a vector-graphics format [22]. LIFULL HOME’s dataset provides five million real samples in a raster format [11]. For both panorama images and floorplans, Structured3D provides 3,500 synthetic houses/apartments [30]. For real samples, Matterport3D provides 90 houses [4], while their coverage is extremely dense. ZIND dataset is the closest to ours with 1,500 real houses/apartments, where the panorama coverage is sparse, and human interventions were required for camera pose estimation and floorplan reconstruction [6]. Nonetheless, ensuring a panorama(s) is taken in every room (except stairs and corridors) makes the setup small-scale and highly controlled. Our data comes from uncontrolled crowd-sourcing, where many rooms are not photographed, posing fundamental challenges to existing techniques.

**Content hallucination:** Inference of missing image or geometry has been a popular research topic. Image inpainting is a classical example [1, 7], where state-of-the-art techniques employ deep neural networks [28]. Neural illumination infers a spherical illumination image from a single perspective image [17]. Implicit neural surface representation infers an entire object surface from a single image [25]. Room layout estimation often infers invisible wall layouts behind objects [14, 20, 26]. This paper infers invisible architectural structures at a house scale (i.e., rooms and doors).

**Extreme pose estimation:** Given two RGB-D images with little to no visual overlaps, relative pose estimation is possible by inferring and aligning complete scene structures [27]. Room-layout estimations are registered to complete a floorplan [9]. Extreme structure from motion (SfM) algorithm

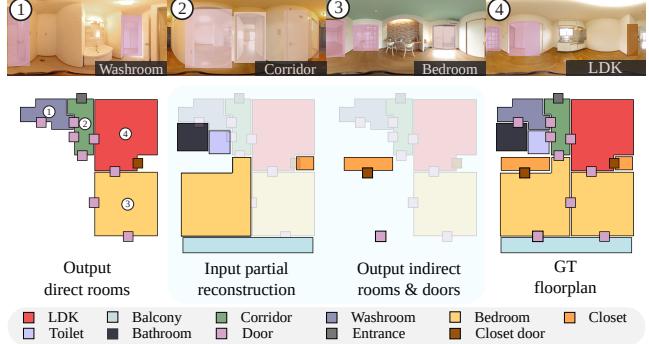


Figure 2. An extreme floorplan reconstruction dataset consists of 701 single-floor apartments/houses with ground-truth vector-graphics floorplan images. The technical challenge lies in the reconstruction of invisible rooms and doors.

Table 1. Dataset statistics. We divide the set into five groups based on the number of rooms (2-5, 6-9, 10-13, 14-17, 18+). In each group, the table reports (Top) the number of samples; (Middle) the ave/std of the number of three types of rooms; and (Bottom) the ave/std of the number of two types of doors.

# of Rooms	2-5	6-9	10-13	14-17	18+	
# of Samples	60	243	298	84	16	
Room	Visible	2.4/0.9	3.5/1.2	3.7/1.6	3.9/1.5	3.4/1.0
	Invis. direct	1.6/0.5	3.7/0.9	5.2/1.6	6.4/2.2	7.2/2.4
	Invis. indire.	0.7/0.9	1.4/1.4	3.7/2.2	5.0/2.8	8.2/3.2
Door	Visible	3.0/1.2	7.2/1.6	8.4/2.5	10.6/4.0	11.4/2.3
	Invisible	0.8/1.0	1.5/1.7	4.4/3.3	5.9/3.2	9.1/3.5

estimates the camera poses of panorama images with little to no visual overlaps by learning spatial arrangements of architectural components [16]. Instead of camera pose estimation, this paper takes aligned panorama images as a partial floorplan model and reconstructs a complete floorplan, including invisible rooms and doors.

## 3. Extreme Floorplan Reconstruction Problem

**Input:** The input is a partial floorplan reconstruction, consisting of rooms and doors, each of which is a 14-channel  $800 \times 800$  segmentation image in a top-down view. The raw input data is a set of panorama images in an equirectangular projection with a camera pose, a room layout, door/window detections, and a room type. Panoramas are acquired with a mono-pod and have roughly the same height from the floor, which allows us to produce an input partial floorplan as a 14-channel image from room layout estimations and camera poses. There are 10 room types (living room, kitchen, western style room, bathroom, balcony, corridor, Japanese style room, washroom, toilet, and closet) and 4 door types (standard-door, entrance-door, closet-door, and open-portal). A door is represented as a  $2 \times 2$  pixel region by the annotators. For the extreme-SfM reconstructions, we

identify the door center and replace it by a  $2 \times 2$  pixel region. Input data come from an extreme Structure from Motion algorithm [16] or human annotators. At testing, we fit an axis-aligned bounding box to the room/door masks in a 14-channel image, uniformly scale to fit at the center of a  $200 \times 200$  square, then add a padding of 300 pixels around, resulting in an  $800 \times 800$  image. At training, we use only ground-truth (GT) samples. We uniformly scale an image to fit at the center of a  $100 \times 100$  square and add a padding of 76 pixels all around to make a  $256 \times 256$  image. We apply the same augmentation process in DETR [3] (i.e., cropping and resizing), followed by 1) 50% chance of horizontal flipping and 2) 50% chance of rotation by 90 or -90 degrees.

**Output:** The output is a complete floorplan in a similar format as the input (i.e., a component-wise  $800 \times 800$  raster mask). Figure 2 shows a sample, where the house has 10 rooms and 12 doors. 10 doors are visible and given in the input, leaving 2 “invisible doors” to be reconstructed. 4 rooms are visible, leaving 6 “invisible rooms” to be reconstructed; 1) 5 of which are adjacent to the input reconstruction and are dubbed “direct invisible rooms”; and 2) the last of which is dubbed “indirect invisible room” (e.g., an invisible closet in an invisible room). Note that the output of our neural network is a raster floorplan image, which is converted to a vector-graphics floorplan by post-processing.

**Dataset:** The dataset consists of 701 houses (or apartments). The raw data are 2,355 panorama images captured by Ricoh Theta series in a production pipeline. The number of panoramas per house ranges from 1 to 7. Each house has a GT floorplan image, which we converted to a vector format by Raster2Vector [11] and manual fixes. Input partial floorplans are inferred by extreme Structure-from-Motion system [16] or created from the GT floorplans (i.e., dropping rooms that do not contain panorama centers).

Randomly sampled 651 houses are used for training while 50 houses are used for testing. To evaluate the robustness across different datasets, we created a synthetic one from the widely-used RPLAN dataset [22], while dropping some panoramas. We refer to the supplementary for more details on the datasets. Table 1 provides some statistics.

**Metrics:** We borrow the precision/recall metrics of a floorplan reconstruction paper [5].<sup>1</sup> When the input is a damaged GT, there is no need to align reconstructions. We match reconstructed component-wise room masks with the GT, and calculate the precision/recall. We declare that a reconstructed room matches a GT, when the room types match, and the intersection over union (IoU) score is above 0.7. A GT room is matched at most once. In practice, we greedily find matches by: 1) Identifying the match with the highest IoU score; 2) Removing the matched pair; and 3)

<sup>1</sup>We borrow the room metrics but do not use the corner/edge metrics, which require a vector-graphics floorplan and are too harsh in our setting.

Repeat. The process is exactly the same for the door metrics except that the IoU threshold is 0.5, because door segments are small. When the input results from the extreme-SfM system, we need to align the reconstruction with the GT before calculating the metric. We exhaustively try all possible translations (at the granularity of a pixel in a top-down image space), then use the result with the best F1 score.

## 4. Neural Extreme Floorplan Reconstruction

Our end-to-end neural architecture consists of a CNN/Transformer based encoder and cascading Transformer decoders. The architecture takes a partial reconstruction and produces a complete floorplan as component-wise raster segmentation masks, which is refined to a vector-graphics floorplan (See Fig. 3). The section explains the key ideas and the design choices. We refer to the supplementary for the full architecture specification.

### 4.1. Visible room/door encoder

Convolutional neural networks (CNNs) with a Transformer encode an input partial reconstruction as a set of  $(W \times H \times 14)$  images into a  $(\frac{W}{32} \times \frac{H}{32} \times 256)$  feature map in two steps. At testing,  $W = H = 800$ . At training,  $W$  and  $H$  depend on the augmentation and are around 600. In the first step, a standard Res-Net [8] (either 50 or 101 layers) processes input images per architectural component category in three branches (i.e., room, door, or both). For example, in the room-branch, we feed each room-image to Res-Net, take the last layer of the last conv-block ( $\frac{W}{32} \times \frac{H}{32} \times 2048$ ), and apply a  $1 \times 1$  convolution to change the depth to 256. Element-wise maximum across all the rooms results in a  $(\frac{W}{32} \times \frac{H}{32} \times 256)$  feature map.

We treat the output as  $(\frac{W}{32} \times \frac{H}{32})$  tokens with 256-dim embedding for the next Transformer block. ResNet produces the same number of tokens from the other two branches. The both-branch is the same as the room-branch except that the input image of a room contains masks of its incident doors. In the door-branch, the input is a single  $(W \times H \times 14)$  image containing all the door masks (instance unaware). In the second step, a self-attention block from standard Transformer [21] (6 layers w/ 8 heads) takes the tokens from the three branches. Following the work by Zhou *et al.* [31], we add standard frequency position encoding to distinguish the branch-type and the X/Y position in the feature map:

$$\begin{aligned} \vec{f}_{xy} &\leftarrow \vec{f}_{xy} + [\vec{P}_{128}(x), \vec{P}_{128}(y)] + \vec{P}_{256}(\text{type}\|1) \\ \vec{P}_d(t) &= [\{\cos(10^{8i/d}t)\}, \{\sin(10^{8i/d}t)\}] \end{aligned} \quad (2)$$

$\vec{f}_{xy}$  denotes the 256-dim embedding of a token at position  $(x, y)$ , where  $x \in [1, \frac{W}{32}]$  and  $y \in [1, \frac{H}{32}]$ .  $\vec{P}_d(t)$  is a  $d$ -dimensional standard frequency position encoding, where

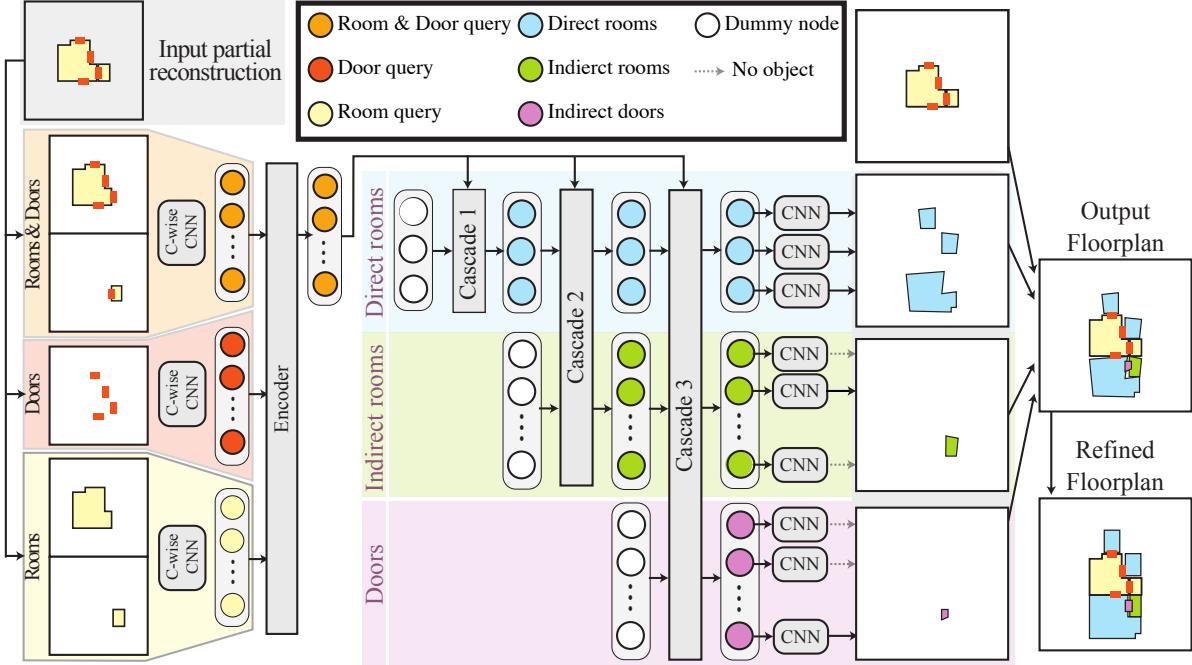


Figure 3. System overview. Category-wise CNNs with a Transformer block encodes an input partial reconstruction into a set of embedding vectors. The cascading Transformer decoders reconstruct invisible rooms and doors in three steps

$i$  is between 1 to  $d/2$ . ( $\text{type}$ ) is a scalar, indicating the branch-type (both=1001, room=1002, and door=1003). The tokens from the “both” branch are passed to the decoder.

## 4.2. Invisible room/door cascading decoders

Three cascaded transformer decoders reconstruct direct invisible rooms, indirect invisible rooms, and invisible doors (See Fig. 2).

**Direct invisible room decoder:** Direct invisible rooms are behind the doors detected in the panoramas. Let  $N_{vis,d}$  be the number of detected doors, which is the expected number of direct invisible rooms. Following DETR [3], we pass  $N_{vis,d}$  query tokens with learnable embeddings to a self-attention block, while feeding the tokens from the encoder via cross-attention. Each query token will contain an embedding of a direct invisible room to be reconstructed. The number of detected doors (i.e., query embeddings) varies. We prepare 20 embedding vectors, which is large enough during training. An output embedding at a query token is used to 1) classify the room-type as a 15-dim vector by a fully connected layer with soft-max; 2) regress the bounding box parameters (i.e., the center, the width, and the height) in the normalized image coordinate ( $x, y \in [0, 1]$ ) by a 3-layer MLP with ReLU (hidden dimension 256); and 3) estimate a binary segmentation mask by the panoptic segmentation head in DETR [3]. Note that the type classification labels consist of 10 room types, 4 door types, and “no

room/door”. The last label indicates that nothing is reconstructed. At testing, we use the category with the greatest value and keep pixels above the positive value in the segmentation mask. We do not use the bounding box for floorplan reconstruction, but for a loss function during training.

**Indirect invisible room decoder:** The second cascaded decoder reconstructs indirect invisible rooms via query tokens while passing the encoder tokens via cross-attention in the same way as the first cascade. The only difference is that the self-attention block also incorporates the direct room tokens from the first cascade. Both the direct and the indirect room tokens are used to predict the room-type, the bounding box parameters, and the segmentation mask by exactly the same network modules with the same loss functions. We assume 15 indirect invisible rooms at maximum and pass 15 query tokens with learnable embedding.

**Invisible door decoder:** The third decoder reconstructs invisible doors via query tokens in the same way as the other cascades. The difference is that self-attention incorporates all the tokens (direct rooms, indirect rooms, and doors). A complete floorplan is reconstructed after the cascade.

## 4.3. Loss Functions

Our neural network reconstructs a floorplan in three cascades. In each cascade, we take the reconstructed components, match them with the corresponding ground-truth components, and inject loss functions. The matching is

done in exactly the same way as DETR [3], based on the type classification and the bounding box estimation. The first cascade is for invisible direct rooms, and we match against only the GT invisible direct rooms. For the second cascade, we match against the GT invisible direct/indirect rooms. For the third cascade, we match against the GT invisible direct/indirect rooms and invisible doors.

Without loss of generality, we use the second cascade as an example to define loss functions, where reconstructed invisible room instances are matched against the corresponding ground-truth. The loss functions consist of the three terms for the room-type classification, the bounding box parameter regression, and the segmentation mask estimation:

$$\begin{aligned} L &= L_{type} + L_{bbox} + L_{seg} \\ L_{type} &= \frac{1}{|\mathcal{I}_{all}|} \sum_{i \in \mathcal{I}_{all}} -w(i) \log(p_i) \\ L_{bbox} &= \frac{1}{|\mathcal{I}_{match}|} \sum_{i \in \mathcal{I}_{match}} 5 \|b_i - \hat{b}_i\|_1 - 2 \text{IOU}(b_i, \hat{b}_i) \\ L_{seg} &= \frac{5}{|\mathcal{I}_{match}|} \sum_{i \in \mathcal{I}_{match}} L_{seg\_match}(i) \\ L_{seg\_match}(i) &= L_{dice}(m_i, \hat{m}_i) + \sum_{p \in \mathcal{P}} \frac{L_{focal}(m_i^p, \hat{m}_i^p)}{|\mathcal{P}|} \end{aligned}$$

$L_{type}$  is a standard softmax loss with per-category weighing.  $\mathcal{I}_{all}$  is a set of indexes of all the reconstructed room instances. With abuse of notation,  $p_i$  denotes the classification score of the GT room type (i.e., the room type of the matched GT).  $w(i)$  denotes a weight associated with the GT room type to compensate for the imbalance in the training data. Concretely, the weight is set inversely proportional to the number of samples in the training set.

$L_{bbox}$  sums the discrepancies of the bounding box estimation over the matched instances. The first term is the L1 norm of the 4-dimensional bounding box parameter vector (the center, the width, and the height in the normalized image coordinate).  $b_i$  denotes the estimated parameter vector and  $\hat{b}_i$  denotes the corresponding GT vector. The second term is the intersection over the union score between the reconstructed and the ground-truth bounding boxes.

$L_{seg}$  sums the discrepancy of the segmentation mask and the matched instance. The first term is the standard dice loss between the estimated mask ( $m_i$ ) and the GT ( $\hat{m}_i$ ). The second term is the average focal loss of the per-pixel mask value.  $\mathcal{P}$  denotes the set of pixels in the domain.  $m_i^p$  (resp.  $\hat{m}_i^p$ ) denotes the estimated (resp. GT) per-pixel mask value.

o

#### 4.4. Floorplan refinement and vectorization

Cascading decoders reconstruct a floorplan as raster images, where room boundaries are curvy, and door shapes

are irregular. We use a floorplan generative model (HouseGAN++ [13] by Nauata *et al.*) to refine our output and convert to a vector format. House-GAN++ is designed to produce floorplans from noise vectors, but can also be used to refine a design without changing the overall arrangement by specifying an entire floorplan as an input constraint. HouseGAN++ learns a bias to prefer straight shape boundaries and we use the same post-processing in the original work to produce a vector graphics floorplan. Note that HouseGAN++ alone, without our constraint fails to infer an accurate floorplan, as shown in the experimental results next. We refer to the supplementary for more details.

## 5. Experimental Results

We have implemented the proposed system using PyTorch 1.10.0 and Python 3.9.7, and used a workstation with a 3.70GHz Intel i9-10900X CPU (20 cores) and an NVIDIA RTX A6000 GPU. Due to memory limitation, we use a batch size of 1 during training, while accumulating gradients over 16 samples for a parameter update. We use AdamW optimizer, while setting the learning rate to  $10^{-5}$  for the CNN module in the encoder and  $10^{-4}$  for the rest of the network (i.e., a Transformer block in the encoder and the three cascading Transformer decoders). We divide the learning rate by 10 after every 100 epochs. We train for 240 epochs, which takes roughly 22 hours with the workstation.

**Competing methods:** We compare against the four competing methods.

- The first one is UNet with a public implementation [12].
- The second one is Mask-RCNN with two different backbones (ResNet-50 and ResNet-101) with an official implementation from Meta AI Research [23].
- The third one is Housegan++ [13], which needs a full bubble diagram where rooms are nodes in that bubble diagram to produce the house layout. However in our problem there are missing rooms where Housegan++ is not able to predict them. we use our model which works the best in missing room prediction to predict missing rooms and create input bubble diagrams for Housegan++. In all iterations we pass visible rooms segmentation masks to Housegan++ and from second iteration we also pass 50% of invisible room predicted segmentation mask from previous iteration to the network too. We continue iterations until 10<sup>th</sup> iteration.
- The fourth one is DETR again with the same two backbones (ResNet-50 and ResNet-101) with an official implementation by the authors [3].

These methods take a single image as an input, while our input is a set of 14-channel images (instance-aware). We perform pixel-wise max-pooling over the images and produce a single 14-channel image as their input. Our output is a set of architectural components, each of which is a binary segmentation mask (i.e., a probability distribution over the

Table 2. The main quantitative evaluations. We compare against UNet, Mask-RCNN (Mrcnn), and DETR with ResNet-50 or ResNet-101 backbone. Input partial floorplans are either ground-truth (left) or outputs from the extreme SfM system [16] (right).

Method	Manual (GT)									Extreme-SfM output								
	Rooms			Doors			Rooms			Doors								
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
UNet	6.2	8.4	7.1	0.8	0.3	0.4	4.5	6.8	5.4	1.2	0.3	0.5						
Mrcnn(50)	38.1	24.2	29.6	12.0	13.4	12.7	37.5	23.2	28.6	12.3	11.4	11.8						
Mrcnn(101)	39.2	18.8	25.4	14.6	13.9	14.3	39.1	22.6	28.6	13.9	12.6	13.2						
Housegan++	33.7	33.1	33.4	10.3	9.9	10.1	25.0	24.4	24.7	12.6	12.2	12.4						
DETR(50)	30.6	47.3	37.1	13.0	13.4	13.2	27.1	41.5	32.8	11.0	13.9	12.3						
DETR(101)	30.7	50.4	38.1	14.2	16.1	15.1	26.8	46.3	33.9	13.3	15.3	14.2						
Ours(50)	49.2	50.8	50.0	20.2	19.3	19.7	37.1	41.0	38.9	14.0	18.4	15.9						
Ours(101)	46.2	41.4	43.7	15.7	14.8	15.2	31.9	39.9	35.4	10.3	13.1	11.5						
Ours(50, Best) + Refinement	56.2	53.1	54.6	21.0	20.4	21.1	40.6	44.9	42.6	15.6	19.1	17.17						

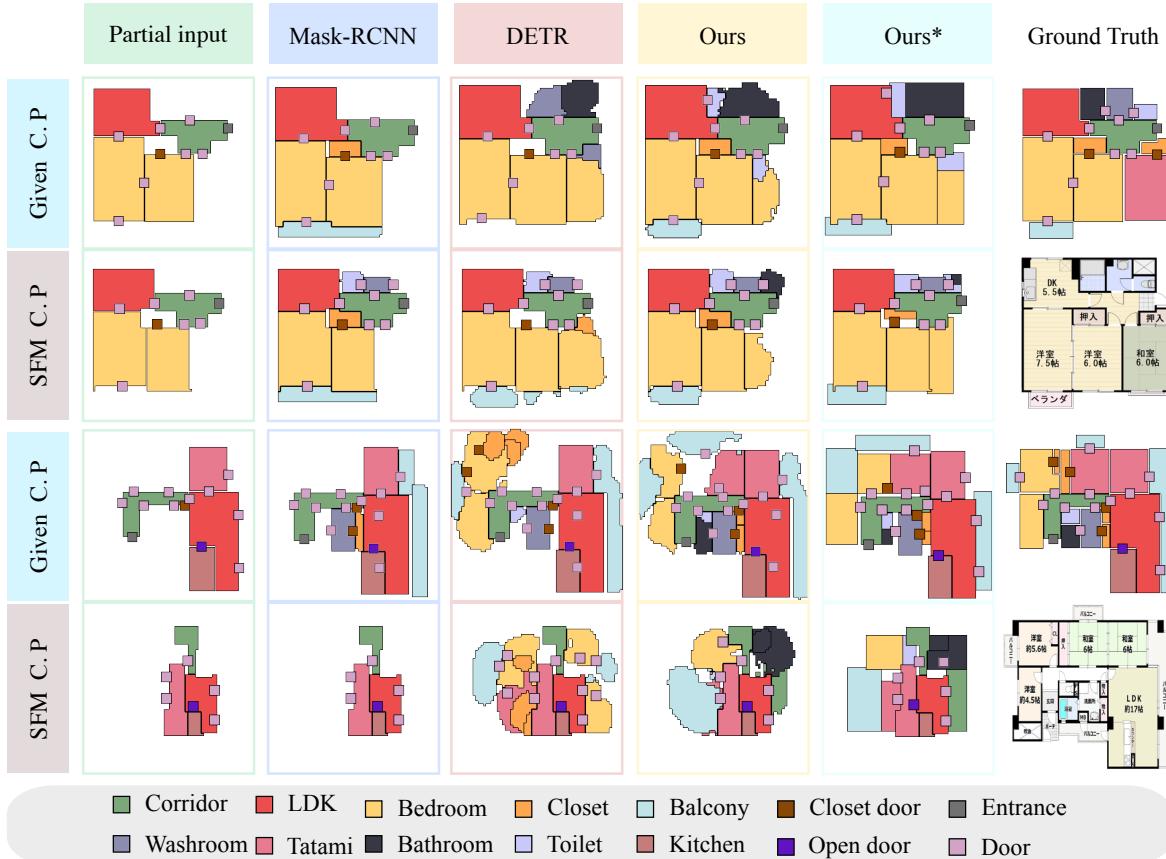


Figure 4. Main qualitative evaluations. The figure shows reconstruction results before and after the final refinement for four houses, (Ours, Ours\*). For each method, we show both when the input partial reconstruction is from the ground-truth or the extreme SfM system [16]. At the right, we show both our ground-truth vector-graphics floorplans as well as the original raster floorplan images by an architect.

room/door types) and the bounding box parameters, which Mask-RCNN and DETR directly produce.

For UNet, the output is a single instance-unaware 14-channel image. We keep pixels above the threshold 0.5, find connected components, and discard small components

whose areas are less than 4 pixels to produce a floorplan. We choose the value of 0.5 because the average F1 score did not change much when we vary the threshold from 0.1 to 0.9. For Mask-RCNN, we set a threshold of 0.6 on the confidence prediction. We vary the parameter and pick the

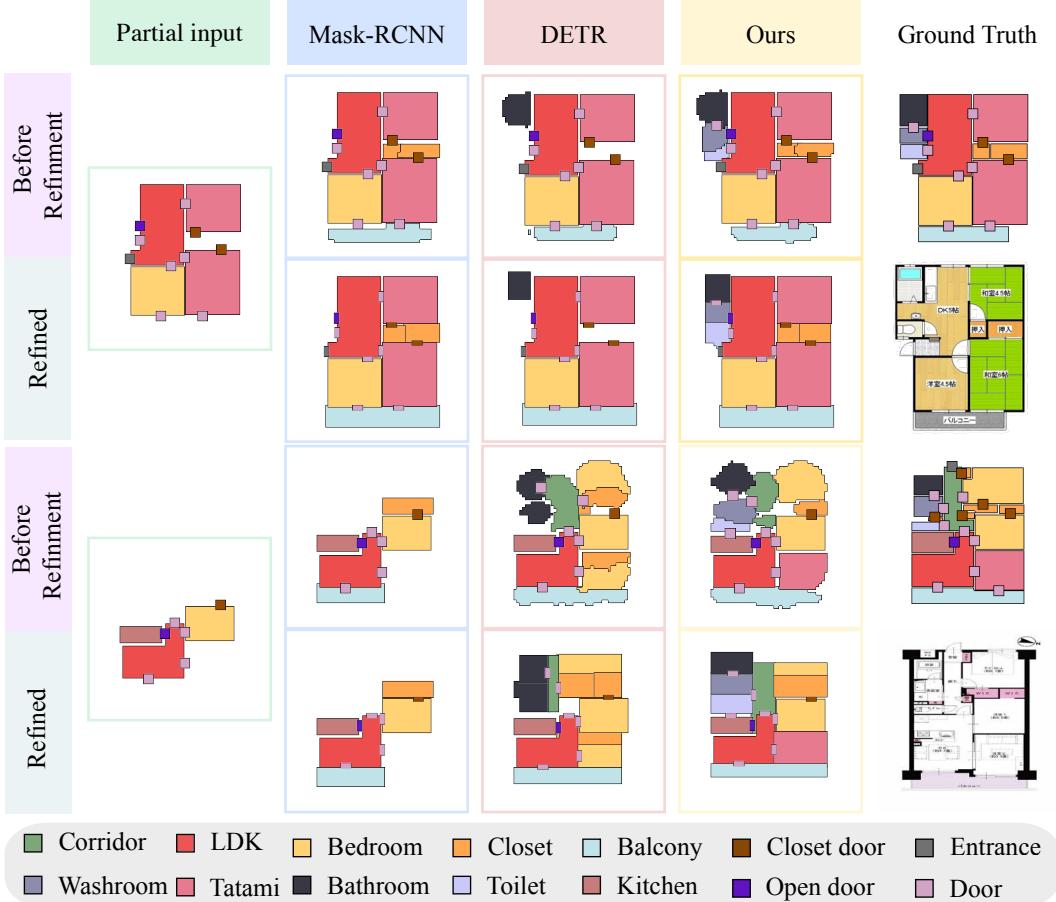


Figure 5. Reconstructed floorplans before and after the refinement by House-GAN++ and several heuristics. The same refinement process is used for all the methods. The input partial reconstructions are ground-truth in this figure.

one with the highest average F1 score. DETR does not require a threshold: A room/door is not generated when the probability of type ‘‘no room/door’’ is the greatest. See the supplementary for results with other thresholds.

For fair evaluation, we have used 1) the same category based weighing in Sect. 4.3 for the type classification loss for Mask-RCNN and DETR; 2) the same parameter update schedule (once every 16 samples); 3) the same data augmentation steps; and 4) the same learning rate schedules.<sup>2</sup>

**Quantitative evaluations:** Table 2 provides our main quantitative evaluations, comparing against three competing methods. The best, the second best, and the third best results are shown in cyan, orange, and magenta, respectively. Note that the last row is our result with the post-processing refinement step (Sect. 4.4), which is not an end-to-end system and given as a reference. Our system achieves the highest F1 score in all the settings. Note that Mask-RCNN

<sup>2</sup>We have two learning rate schedules, one for Transformer and the other for CNN. Exactly the same schedule is used for DETR. Our schedule for the CNN module is used for Unet and Mask-RCNN.

(101) outperforms ours in precision, but its recall is low. Similarly, DETR (101) outperforms in recall, but its precision is low. Interestingly, a smaller backbone (ResNet-50) achieves better results in our system, which is probably because the main learning happens in our Transformer blocks and large CNNs merely introduce overfitting. Table 5 presents results on larger synthetic dataset [22], where the proposed method outperforms all the other baselines.

Table 3 provides an ablation study on the category-wise CNN encoder. The bottom row is our final system. In the fifth entry, the door branch receives multiple images (instance aware), which shows comparable room metrics but worse performance for doors. This suggests that doors are already cleanly separated and an instance unaware single image representation is more efficient and effective. The last two rows suggest that the room information should be passed as the instance-aware representation.

Table 4 provide an ablation study on the cascading decoders. The middle three rows show that dropping any of the cascades downgrades performance. While the third cas-

Table 3. Ablation study on the category-wise CNN encoder. The left three columns shows how the input partial reconstructions are passed to each of the three CNN branches. (m) indicates that a branch receives multiple images (instance-aware) as an input. (s) indicates that a branch receives a single image (instance-unaware) that contains masks of all the instances. No-mark indicates that a branch is not used.

both	room	door	Direct Rooms (First Cas.)			Rooms (Second Cas.)			Rooms (Full Pipeline)			Doors (Full Pipeline)		
			Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
s			56.3	62.6	59.3	40.8	45.1	42.8	43.8	46.5	42.8	14.8	14.5	14.6
m			67.8	63.5	65.5	34.6	43.9	38.7	35.8	48.2	41.8	6.2	7.5	6.7
m	m		63.7	62.5	63.1	35.3	44.6	39.4	36.0	48.6	41.3	7.1	6.9	7.0
m		s	60.7	61.1	60.8	47.6	45.5	46.3	40.6	47.6	43.9	14.1	15.6	14.8
m	m	m	66.2	64.7	65.8	46.7	44.9	45.8	48.4	49.2	48.8	10.6	11.2	10.9
s	s	s	57.2	54.2	55.6	37.6	39.7	38.6	38.4	40.1	39.2	17.8	18.9	18.3
m	m	s	68.2	64.2	66.1	49.6	46.6	48.0	49.2	50.8	50.0	20.2	19.3	19.7

Table 4. Ablation study on the cascading decoders. We turn on and off each of the three cascades (indicated by the left columns) and retrain the entire model. The table also reports the metrics of reconstructions by the first and the second cascades. Note that the first (resp. second) cascade reconstructs only direct rooms (resp. direct and indirect rooms), respectively, whose metrics are reported.

Cascades	Direct Rooms (First Cas.)			Rooms (Second Cas.)			Rooms (Full Pipeline)			Doors (Full Pipeline)				
	1st	2nd	3rd	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1		
✓	x	x	x	x	x	x	36.3	41.1	38.5	14.5	14.6	14.5		
✓	✓	x	x	39.4	46.3	42.6	43.1	47.0	44.9	14.9	15.2	15.0		
✓	✓	✓	67.5	60.3	63.6	x	x	x	36.2	41.4	38.6	16.3	16.1	16.2
✓	✓		44.8	46.3	45.5	40.9	42.0	41.5	40.9	42.0	41.5	x	x	x
✓	✓	✓	68.2	64.2	66.1	49.6	46.6	48.2	49.2	50.8	50.0	20.2	19.3	19.7

Table 5. Quantitative evaluations on larger synthetic dataset, RPLAN [22]. We compare against UNet, Mask-RCNN (Mrcnn), House-GAN++, and DETR with ResNet-50 1 backbone. Input partial floorplans are ground-truth. The precision, the recall, and the F1 score are reported for the rooms and the doors.

Method	Rooms			Doors		
	Pre.	Rec.	F1	Pre.	Rec.	F1
UNeT	9.1	10.3	9.6	2.0	1.9	1.9
Mrcnn(50)	46.4	37.6	41.5	11.5	10.3	10.8
Mrcnn(101)	48.9	36.8	41.7	11.1	9.5	10.2
House-GAN++	47.6	45.3	46.4	15.8	12.7	14.0
DETR(50)	50.5	52.1	51.3	15.4	16.3	15.8
DETR(101)	54.0	50.6	52.2	16.3	16.6	16.4
Ours(50)	64.7	51.3	57.2	20.8	18.7	19.1
Ours(101)	69.8	51.4	59.2	21.1	19.5	20.3

cade alone (first row) produces reasonable results, the proposed system with the three cascades achieves the best result in every metric. The last two rows show an interesting phenomena where the third cascade improves the performance of the first two cascades via gradient propagation.

**Qualitative evaluation:** Figure 4 provides the main qualitative evaluations against Mask-RCNN and DETR. “Ours\*” shows the final floorplan models after the refinement by

House-GAN++. Figure 5 shows reconstructed floorplans by different methods before and after the refinement. The refinement process successfully turns raw floorplan reconstructions with many artifacts (*e.g.*, curvy room boundaries and gaps between rooms) into clean floorplan models. The input and the corresponding ground-truth reveal that this is a challenging reconstruction task, unlike any existing problems. Mask-RCNN is often capable of inferring closets or balconies that are direct neighbors of the input partial reconstruction. However, it fails to infer indirect invisible rooms or doors that are far away from the input reconstruction. With the power of the Transformer, DETR reconstructs indirect rooms and doors more than Mask-RCNN. Nonetheless, our approach infers many invisible structures successfully, in particular when manual (ground-truth) partial reconstructions are given. Please see the supplementary for more reconstruction examples.

**Concluding marks:** This paper presents a new extreme floorplan reconstruction task, a new benchmark, and a neural architecture as a solution. The task is challenging and our results are not always satisfactory. Major failure modes are 1) missing rooms; 2) inaccurate room shapes, in particular corridors; and 3) inaccessible rooms without any doors. We hope that this paper starts an avenue of new research toward an ultimate extreme floorplan reconstruction system

capable of reconstructing accurate and realistic floorplans for tens of millions of houses with sparse panorama coverage out there. We will share all our code, model, and data, except the panorama images, for privacy concerns.

**Acknowledgement:** This research is partially supported by NSERC Discovery Grants with Accelerator Supplements and DND/NSERC Discovery Grant Supplement.

## References

- [1] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424, 2000. [2](#)
- [2] Ricardo Cabral and Yasutaka Furukawa. Piecewise planar and compact floorplan reconstruction from images. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 628–635, 2014. [2](#)
- [3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing. [3, 4, 5](#)
- [4] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. [2](#)
- [5] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-sp: Inverse cad for floorplans by sequential room-wise shortest path. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. [3](#)
- [6] Steve Cruz, Will Hutchcroft, Yuguang Li, Naji Khosravan, Ivaylo Boyadzhiev, and Sing Bing Kang. Zillow indoor dataset: Annotated floor plans with 360deg panoramas and 3d room layouts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2133–2143, June 2021. [2](#)
- [7] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (ToG)*, 26(3):4–es, 2007. [2](#)
- [8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. [3](#)
- [9] Cheng Lin, Changjian Li, and Wenping Wang. Floorplanningjigsaw: Jointly estimating scene layout and aligning partial scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5674–5683, 2019. [2](#)
- [10] Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floornet: A unified framework for floorplan reconstruction from 3d scans. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. [2](#)
- [11] Chen Liu, Jiajun Wu, Pushmeet Kohli, and Yasutaka Furukawa. Raster-to-vector: Revisiting floorplan transformation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2214–2222, 2017. [2, 3](#)
- [12] ALEXANDRE . MILESI. U-net: Semantic segmentation with pytorch. <https://github.com/milesial/Pytorch-UNet>, 2018. [5](#)
- [13] Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa. Housegan++: Generative adversarial layout refinement network towards intelligent computational agent for professional architects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13632–13641, 2021. [5](#)
- [14] Giovanni Pintore, Marco Agus, and Enrico Gobbetti. AtlantaNet: Inferring the 3D indoor layout from a single 360 image beyond the Manhattan world assumption. In *Proc. ECCV*, August 2020. [2](#)
- [15] Giovanni Pintore, Claudio Mura, Fabio Ganovelli, Lizeth Fuentes-Perez, Renato Pajarola, and Enrico Gobbetti. State-of-the-art in automatic 3d reconstruction of structured indoor environments. In *Computer Graphics Forum*, volume 39, pages 667–699. Wiley Online Library, 2020. [2](#)
- [16] Mohammad Amin Shabani, Weilian Song, Makoto Odamaki, Hirochika Fujiki, and Yasutaka Furukawa. Extreme structure from motion for indoor panoramas without visual overlaps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5703–5711, 2021. [2, 3, 6](#)
- [17] Shuran Song and Thomas Funkhouser. Neural illumination: Lighting prediction for indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6918–6926, 2019. [2](#)
- [18] Shuran Song and Thomas A. Funkhouser. Neural illumination: Lighting prediction for indoor environments. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6911–6919, 2019. [1](#)
- [19] Sinisa Stekovic, Mahdi Rad, Friedrich Fraundorfer, and Vincent Lepetit. Montefloor: Extending mcts for reconstructing accurate large-scale floor plans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16034–16043, October 2021.
- [20] Cheng Sun, Chi-Wei Hsiao, Min Sun, and Hwann-Tzong Chen. Horizonnet: Learning room layout with 1d representation and pano stretch data augmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. [2](#)
- [21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. [3](#)
- [22] Wenming Wu, Xiao-Ming Fu, Rui Tang, Yuhan Wang, Yu-Hao Qi, and Ligang Liu. Data-driven interior plan generation for residential buildings. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 38(6), 2019. [2, 3, 7, 8](#)
- [23] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019. [5](#)
- [24] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *2012 IEEE Conference on Computer*

- Vision and Pattern Recognition*, pages 2695–2702. IEEE, 2012. 2
- [25] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 2
- [26] Shang-Ta Yang, Fu-En Wang, Chi-Han Peng, Peter Wonka, Min Sun, and Hung-Kuo Chu. Dula-net: A dual-projection network for estimating room layouts from a single RGB panorama. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 3363–3372, 2019. 2
- [27] Zhenpei Yang, Jeffrey Z Pan, Linjie Luo, Xiaowei Zhou, Kristen Grauman, and Qixing Huang. Extreme relative pose estimation for rgb-d scans via scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4531–4540, 2019. 2
- [28] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4471–4480, 2019. 2
- [29] Yingchen Yu, Fangneng Zhan, Rongliang Wu, Jianxiong Pan, Kaiwen Cui, Shijian Lu, Feiying Ma, Xuansong Xie, and Chunyan Miao. Diverse image inpainting with bidirectional and autoregressive transformers. *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 1
- [30] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *Proceedings of The European Conference on Computer Vision (ECCV)*, 2020. 2
- [31] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pages 11106–11115. AAAI Press, 2021. 3