Supplementary Material for SALVe: Semantic Alignment Verification for Floorplan Reconstruction from Sparse Panoramas

John Lambert[2]*, Yuguang Li[1], Ivaylo Boyadzhiev[1], Lambert Wixson[1], Manjunath Narayana[1], Will Hutchcroft[1], James Hays[2], Frank Dellaert[2], and Sing Bing Kang[1]

[1] Zillow Group
[2] Georgia Institute of Technology

Abstract. In this Supplementary Material, we provide additional analysis and implementation details. In Section 1, we provide qualitative comparisons of our floorplan reconstructions, vs. an upper-bound *oracle* baseline that uses ground-truth global pose estimation. In Section 2, we provide quantitative analysis of SALVe's relative pose classification accuracy with various input modalities. In Section 3, we provide pseudo-code for our layout stitching algorithm. In Sections 4 and 5, we report detailed quantitative analysis of W/D/O detection accuracy, and W/D/O and layout estimation failure cases. In Section 6, we describe the coordinate systems used in our work. In Sections 7, 8, and 9, we provide additional implementation details about rendering, vanishing-point based axis alignment, and pose graph optimization and spanning tree aggregation. In Section 10, we describe ablation experiments that compare the use of ground truth W/D/O and ground truth layout, vs. estimated W/D/O and estimated layout. In Section 11, we provide additional discussion about our evaluation procedures versus those of concurrent work [7]. In Sections 12-15, we provide additional analysis and further examples of positive and negative training examples. In Section 16, we discuss additional details about verifier training and data augmentation, and in Section 17, we discuss ethical concerns associated with the work

1 Qualitative Results: Predicted vs. Oracle Poses

In this section, we provide qualitative comparisons with a baseline that stitches predicted layouts placed at 'oracle' global pose locations (referred to by Section 6.1 of the main paper). For this baseline, the high fidelity of reconstructed shapes (middle column of Fig. 1 and Fig. 2) demonstrates the maturity of modern layout estimation networks. This baseline also illustrates the impact of global pose estimation on the entire system.

2 Additional Analysis of Relative Pose Classification Accuracy

Here we provide a more comprehensive quantitative analysis of the influence of input modalities and CNN backbone architecture on SALVe's relative pose classification accuracy (referred to in Section 5.5 of the main paper). We compare ceiling-only texture map input, vs. floor-only texture map input, vs. using both as input.

* Work completed during an internship at Zillow Group.

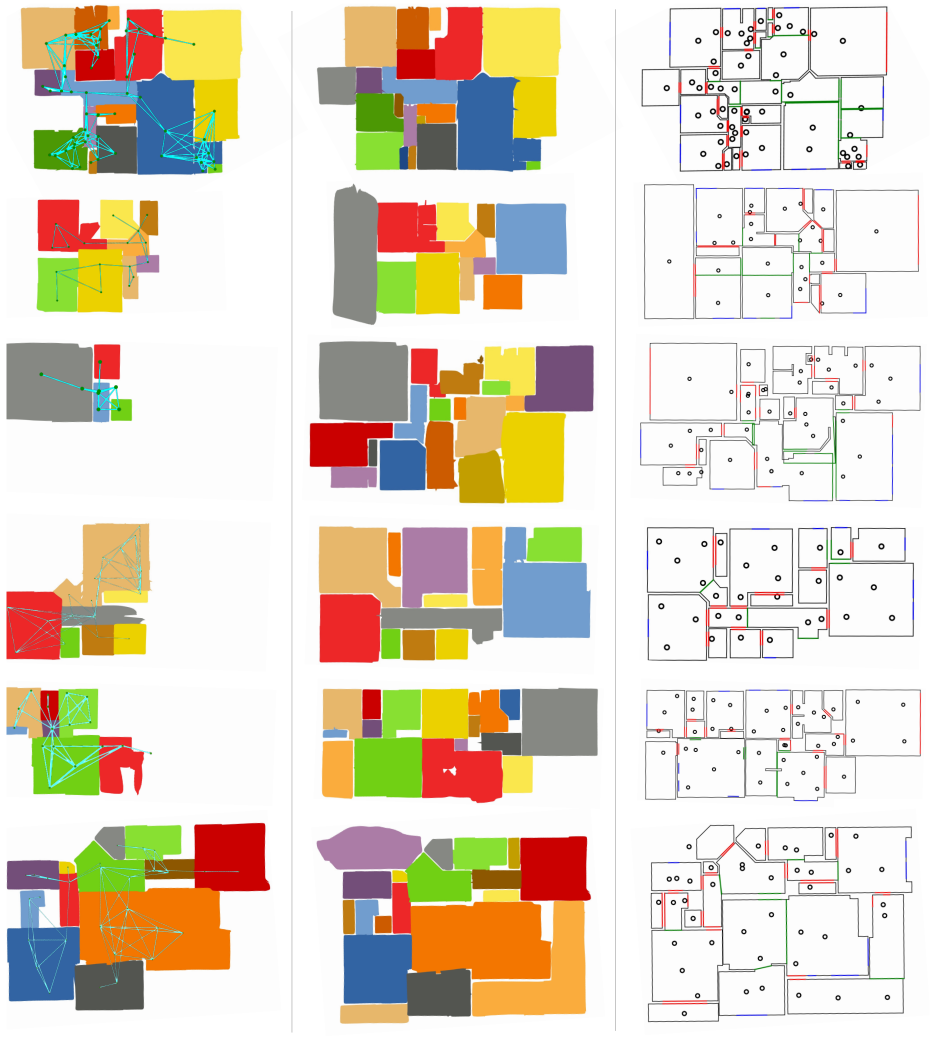


Fig. 1: Example floorplan results of varying completeness, comparing SALVe's performance vs. an upper bound (perfect global pose estimation). ***Left:*** predicted poses of the largest connected component and predicted room layout. ***Middle:*** oracle (ground truth) poses and predicted room layout. ***Right:*** ground truth floorplan with positions of captured panoramas.

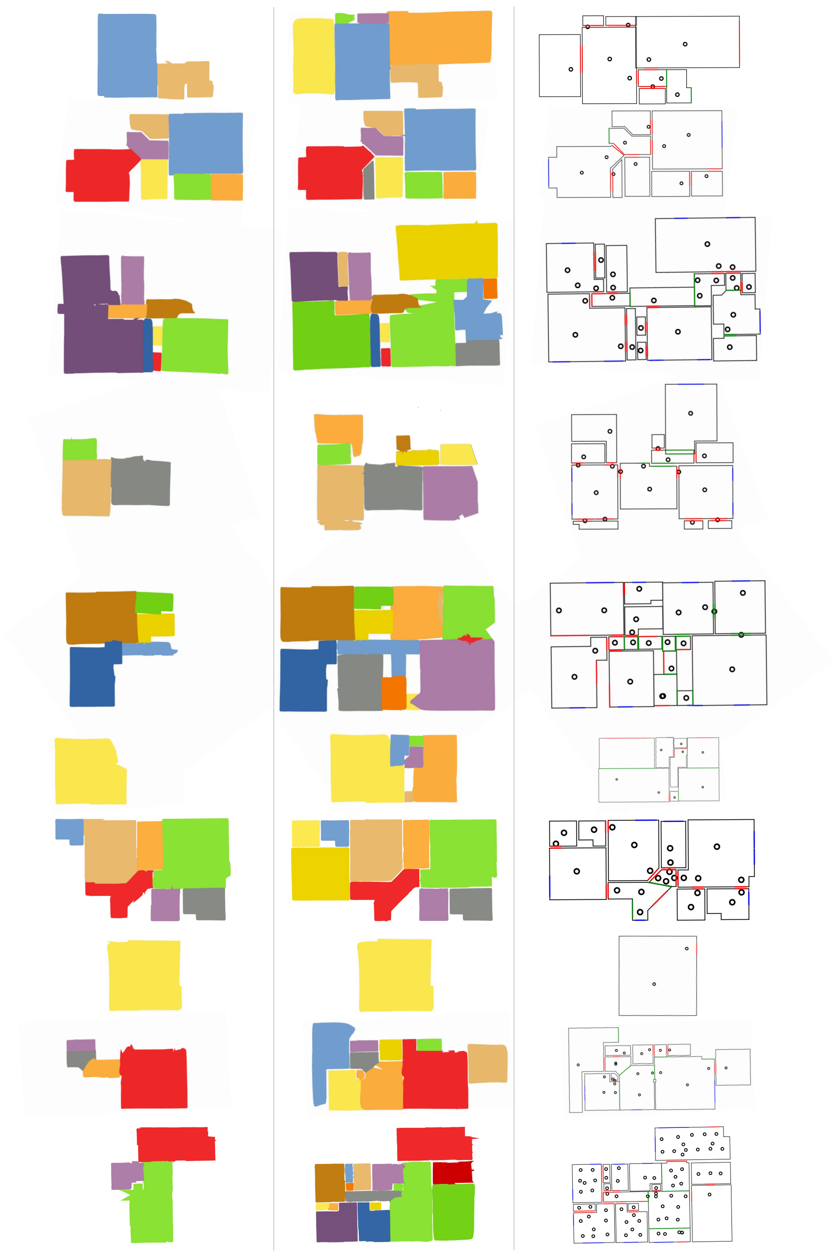


Fig. 2: Additional comparison between SALVe's performance and an upper bound (perfect global pose estimation). Each row corresponds to a single floor of a different home. ***Left:*** predicted poses of the largest connected component and predicted room layout. ***Middle:*** oracle poses and predicted room layout. ***Right:*** ground truth floorplan with positions of captured panoramas. Colored lines represent W/D/O objects – doors, openings and windows.

Table 1: Relative pose classification accuracy on the ZInD test split with different inputs and architectures. Precision, recall, and mean accuracy are reported. Extreme class imbalance means that with more expressive model architectures, gains in mean accuracy are minor, but gains in precision are significant.

Model Architecture	Ceiling Texture Map	Floor Texture Map	Prec.	Rec.	mAcc.
ResNet-50	✓	✓	0.77	0.91	0.96
ResNet-152	✓	✓	**0.85**	**0.91**	**0.95**
ResNet-152	✓		0.70	0.88	0.93
ResNet-152		✓	0.84	**0.91**	**0.95**

3 Details on Layout Stitching for Floorplan Reconstruction

This section provides additional details about the reconstruction algorithm mentioned in Section 4.5 and Figure 4 of the main paper.

Floorplan reconstruction involves three steps: (1) panorama room grouping, (2) highest confidence room contour extraction, and (3) floorplan stitching. Please see Algorithm 1 for implementation details. In Figure 3, we demonstrate the process of generating final room layout using estimated panorama poses grouped by step (1). Comparing plot (b) to plot (a), we can see that room contour confidence provide useful guidance in selecting the high confidence contour point among different views. In plot (c), each view-dependent room contour largely will agree with each other. In the end, we take the union of different view-dependent room contours to account for the occlusions from each panorama view.

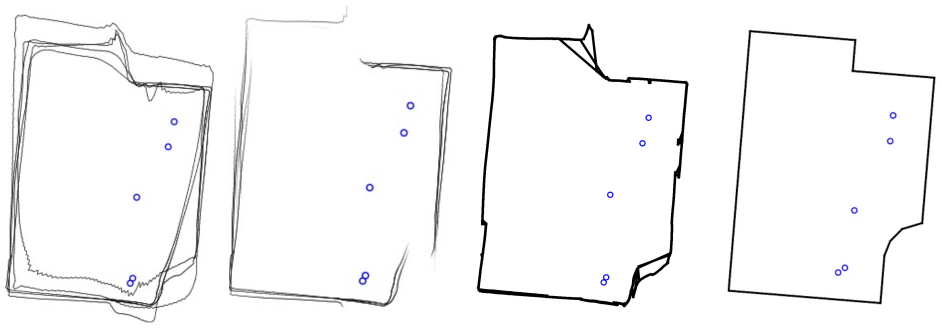


Fig. 3: Visualization of room shape reconstruction using localized panoramas grouped by room. **(a)** Predicted room layout and predicted panorama locations (blue dots). **(b)** Predicted room layout with contour confidence (transparency) and predicted panorama locations (blue dots). **(c)** Overlay of room contours generated by voting on the highest confidence contour point at each panorama column from each panorama view. The final room layout is the union of these view-dependent contours of highest confidence. **(d)** Ground truth room shape and ground truth panorama positions.

Algorithm 1 Floorplan Reconstruction for a Connected Component in Pose Graph

```
Inputs:
$\{I_i\}$: A list of the input panorama images in the connected component.
$\{\mathbf{T}_i\}$: Estimated panorama poses from pose graph, in top-down global 2D coordinates.
$\{(C_i, \sigma_i)\}$: Estimated room contour points and contour point confidence for panorama $I_i$, in
top-down global 2D coordinates. (One point per panorama column.)

Output:
$S^{opt}_{floorplan}$: Optimized floor plan polygon shape.

Solution:
% Step 1: Group panoramas that come from the same room
Initialize panorama connectivity graph $\mathit{\Gamma}$ with one node per pano $I_i$ and no edges
for $(\mathbf{T}_i, \mathbf{T}_j) \in \{\mathbf{T}_i\} \times \{\mathbf{T}_i\}$ do
    $IoU \leftarrow ComputeContourIoU(\mathbf{T}_i, C_i, \mathbf{T}_j, C_j)$
    if $IoU > Threshold$ then
        $\mathit{\Gamma}.AddEdge(i, j)$
    end if
end for
% Each connected component in $\mathit{\Gamma}$ is a room
$\mathcal{G} = \{G^r, r = 1, ..., N_{rooms}\} \leftarrow \mathit{\Gamma}.GetConnectedComponents()$

% Step 2: Extract highest confidence contour for each room
for $G^r \in \mathcal{G}$: do
    Let optimized room shape $S^{opt}_r = \emptyset$
    for $I_i \in G^r$ do
        $\mathcal{P}^i = \{(P^i_j, \sigma^i_j)\}\ \forall\ I_j \in G^r$, where $(P^i_j,\sigma^i_j)$ are the projections of $(C_j, \sigma_j)$ onto pano
$i$'s image
        In each image column of pano $i$, choose the most confident contour point from $\mathcal{P}^i$.
        $S^i \leftarrow$ the selected points, projected back into the 2D global coordinates using $\mathbf{T}_i$
    end for
    $S^{opt}_r = \bigcup_{I_i \in G^r} polygon(S^i)$
end for

% Step 3: Floorplan stitching
$S^{opt}_{floorplan} = \bigcup_{l=0}^{N_{rooms}} S^{opt}_r$
```

4 Details on W/D/O Detection Evaluation

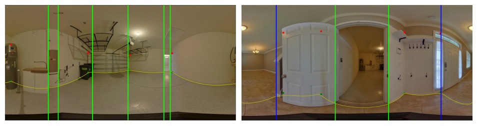
In Section 5.4 of the main paper, we report W/D/O detection results of our HorizonNet [8] model at a 70% IoU threshold. For completeness, we provide here an evaluation of 1d IoU at 50%, 70%, and 90% true positive thresholds (see Table 2).

Table 2: Additional W/D/O detection accuracy results.

	0.5 IoU			0.7 IoU			0.9 IoU		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
DOOR	0.88	**0.92**	0.90	0.87	**0.91**	0.89	0.86	0.81	0.84
WINDOW	**0.94**	0.91	**0.92**	**0.94**	0.89	**0.91**	**0.93**	**0.82**	**0.87**
OPENING	0.79	0.65	0.72	0.78	0.59	0.67	0.72	0.43	0.54

5 Layout and W/D/O Failure Cases

Section 5.4 of the main paper discusses the accuracy of W/D/O detection. Here we offer, in Figure 4, two examples of some the failure modes of the Layout and W/D/O model that provides the input to SALVe.


(a)

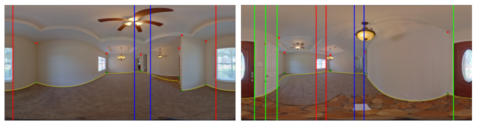
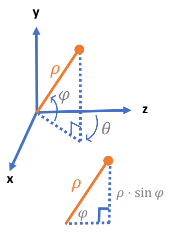

(b)

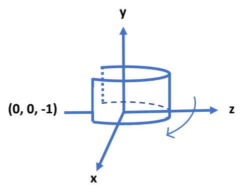
Fig. 4: Mistakes made by the joint HorizonNet + W/D/O model. Vertical lines indicate start and end columns for each W/D/O object – window, door, and opening. The yellow contour indicates the predicted floor-wall boundary, and dots indicate corner predictions (floor-wall corners in green, and ceiling-wall corners in red). Left and right images are panoramas across which we seek to match W/D/O objects. ***Top:*** A circuit breaker panel is mistakenly identified as a door **(top left)**, but redundancy still allows matching of the true garage door. This allows a relative pose hypothesis to be generated between the foyer and garage panoramas, that have very little visual overlap. ***Bottom:*** A false negative window prediction and inaccurate opening prediction **(bottom left)** makes matching with the **(bottom right)** panorama impossible using W/D/O.

6 Coordinate System Conventions

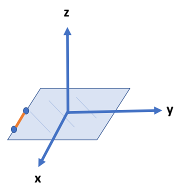
Figure 5 shows the coordinate systems used in our work: panoramic spherical coordinate system, room Cartesian coordinate system, world-normalized Cartesian coordinate system (with camera height set to 1.0), and the world-metric coordinate system. These are also the coordinate conventions used by ZInD[3].



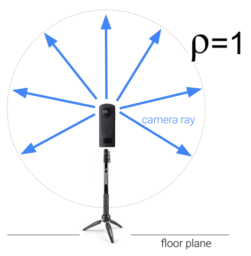
(a) Panoramic Spherical



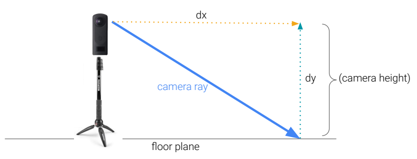
(b) Room Cartesian



(c) World-metric Cartesian



(d)



(e)

Fig. 5: Coordinate system conventions. For (b), note that an equirectangular projection of a panorama actually captures a sphere, not a cylinder.

Scaling to Metric Space. With known camera height, a predicted floor-wall boundary in pixel space with vertices $\{(u, v)_k\}_{k=1}^{K}$ can be mapped to 3D by first converting each vertex to spherical coordinates, and then to Cartesian coordinates, as follows:

$$\begin{aligned} \theta &= \left(u \cdot \frac{2\pi}{(w-1)}\right) - \pi, \qquad \theta \in [-\pi, \pi] \\ \varphi &= \pi\left(1 - \frac{v}{(h-1)}\right) - \frac{\pi}{2}, \qquad \varphi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \end{aligned} \tag{1}$$

where h and w is the height and width of input image in pixels.

[3] ZInD is publicly available under the following license: https://bridgedataoutput.com/zillowterms.

We next obtain ray directions (x, y, z) in Cartesian space by assuming that all points (θ, ϕ, ρ) lie on the unit sphere (see Figure 5d), i.e., $\rho = 1$, and

$$\begin{aligned} x &= \cos(\varphi)\sin(\theta) \\ y &= \sin(\varphi) \\ z &= \cos(\varphi)\cos(\theta). \end{aligned} \tag{2}$$

Finally, we rescale the length of each ray such that it intersects the ground plane at $y = 0$, i.e., the magnitude of its y coordinate is equal to the camera height h_c (see Figure 5e). These rescaled ray directions are now coordinates in meters.

7 Texture Mapping Procedure

In this section, we discuss the procedure we use when creating bird's eye view (BEV) texture maps, as mentioned in Section 5.2 of the main paper.

When texture mapping an orthographic view using the monocular estimated depth map from [9], we use all 3D points $\geq$ 1m below camera for rendering the floor, and all points $\geq$ 0.5m above the camera for rendering the ceiling. We render a 10×10m region, using a resolution of 0.02 m/pixel, creating a 500×500 image.

Afterwards, we apply dense interpolation to the initial texture map. When generating the orthographic imagery, the raw signal from pixel values at all backprojected depth map locations is sparse and insufficient. We rely upon interpolation to generate a dense canvas from the sparse canvas (see Figure 6, top). This interpolation also adds unwanted and undesirable interpolation artifacts (See Figure 6, middle subfigure). We design another step to identify the regions where the signal was too sparse to interpolate accurately. We convolve the canvas that is populated with sparse values with a box filter. We zero-out portions of an interpolated image where the signal is unreliable due to no measurements. If a $K \times K$ subgrid of an image has no sparse signals within it, and is initialized to a default value of zero, then convolution of the subgrid with a box filter of all 1's will be zero. In short, if the convolved output is zero in any ij cell, then we know that there was no true support for interpolation in this region, and we should mask out this interpolated value. We multiply the interpolated image with binary unreliability mask to zero out unreliable values. Convolution with a large kernel, e.g., 11×11 pixels in size on a 500p image, can be done on the GPU. We populate the canvas from bottom to top.

8 Vanishing Point Axis Alignment

Here we provide details about how we refine the hypothesized relative alignment described in Section 4.2 of the main paper.

To correct minor errors of W/D/O vertex localization, we compute vanishing points and convert them to a vanishing angle θ_{vp} with direction voting from line segments [11]. The vanishing angle θ_{vp} is defined as the horizontal angle between left edge of panorama and the first vanishing point from the left side of the panoramic image. We then refine the panorama horizontal rotation by aligning the pair of vanishing angles,

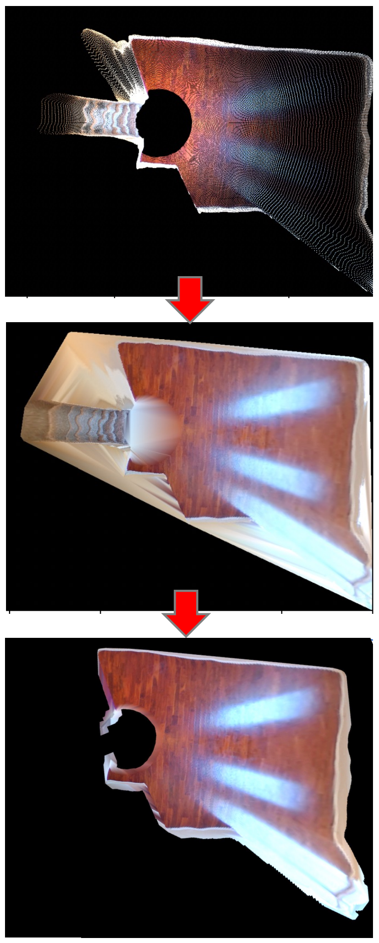


Fig. 6: Visualization of the sparse to dense interpolation scheme. ***Top*:** sparse texture map from mono-depth. ***Middle*:** linearly interpolated texture map. ***Bottom*:** result after removing interpolation artifacts.

while maintaining the distance between the matching W/D/O. The angular adjustment can be represented by: $\theta_{correction} = (\theta_{vp,1} - \theta_{vp,2}) - {}^{2}\theta_1$, where $\theta_{vp,1}$, $\theta_{vp,2}$ are the vanishing angles of panorama 1 and panorama 2, and ${}^{2}\theta_1$ corresponds to the relative rotation of panorama 1's pose in the room Cartesian coordinate system of panorama 2, i.e. of ${}^{2}\mathbf{T}_1$. We then rotate panorama 1's room vertices (in panorama 2's frame) about the W/D/O midpoint, and recompute $\hat{\mathbf{T}} = (\hat{x}, \hat{y}, \hat{\theta})$ by least-squares fitting between point sets to obtain $\hat{\mathbf{T}}_{corrected} = (\hat{x}', \hat{y}', \hat{\theta}')$ with fixed wall thickness.

9 Details on Global Pose Estimation

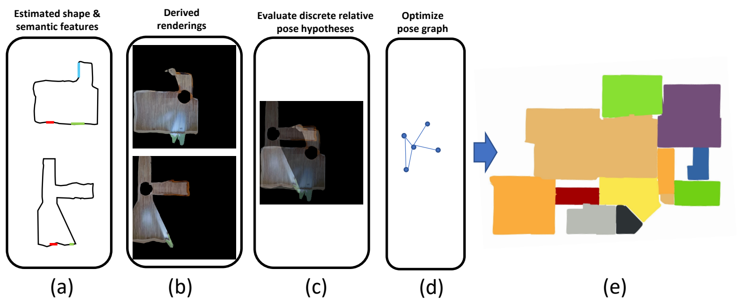

Estimated shape & semantic features
Derived renderings
Evaluate discrete relative pose hypotheses
Optimize pose graph
(a)
(b)
(c)
(d)
(e)

Fig. 7: Starting with sparse panoramas (1-3 per room), in (a) we infer layout and semantic elements (Windows, Doors, Openings, or W/D/O). From these, in (b) we generate birds eye view (BEV) renderings of floors and ceilings (ceilings not shown here). Next, plausible pairwise relative poses are hypothesized based on matching W/D/O. Each is accepted or rejected (c), by feeding the hypothesis-aligned renderings into our learned SALVe verifier. This example shows two aligned renderings computed by hypothesizing that a window can be used to align both shapes Brighter areas indicate overlap regions. SALVe is trained to evaluate these aggregated overlap regions and output an accept/reject decision about whether the hypothesized relative pose is plausible. From the plausible relative poses, a pose graph is created and optimized (d). This allows room layouts to be positioned in a world coordinate system and fused into a final reconstructed raster floorplan (e).

Here we provide details on the pose estimation and optimization referred to in Section 4.4 of the main paper.

No optimization (unfiltered spanning tree). For N images, the global motion can be parameterized by $N-1$ motions. In the pose graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, when the graph G has a single connected component, the spanning tree is a set of edges such that every vertex in $\mathcal{V}$ is reachable from every other vertex in $\mathcal{V}$. For a graph with N vertices, the minimum spanning tree always has $N-1$ edges. However, global pose estimation with this method is inherently susceptible to error due to contamination by outliers. To describe the method to compute a spanning tree, we assume the images are randomly

ordered. Starting from the first image as the root, we incrementally include images in sequence, adding each next image into the current tree at its shortest path from the root.
Pose Graph Optimization. Spanning tree solutions are susceptible to outlier edges in the tree. In order to exploit redundancy and utilize all the available information in the graph, *we use pose graph optimization*.

MAP inference for SLAM problems with Gaussian noise models is equivalent to solving a nonlinear least squares problem [3]. MAP inference comes down to maximizing the product of all factor graph potentials:

$$\mathbf{T}^{\text{MAP}} = \arg\max_{T} \prod_{(i,j)\in\mathcal{E}} \phi_{ij}(\mathbf{T}_i, \mathbf{T}_j), \tag{3}$$

where $\phi_{ij}(\mathbf{T}_i, \mathbf{T}_j)$ is a factor graph potential:

$$\phi_{ij}(\mathbf{T}_i, \mathbf{T}_j) \propto \exp\left\{ -\frac{1}{2}\|h_{ij}(\mathbf{T}_i, \mathbf{T}_j) - z_{ij}\|^2_{\Sigma_{ij}} \right\}, \tag{4}$$

with $h_{ij}(T_i, T_j) = T_i^{-1} \cdot T_j$ and z_{ij} is the estimated relative pose between images i and j from the alignment step described earlier.

The following objective function is then optimized using GTSAM:

$$\arg\min_{\mathbf{T}} \sum_{(i,j)\in\mathcal{E}} \rho\Big(\|h_{ij}(\mathbf{T}_i, \mathbf{T}_j) - z_{ij}\|^2_{\Sigma_{ij}}\Big), \tag{5}$$

making updates $\mathbf{T}_i \oplus \xi := \mathbf{T}_i \circ \exp(\hat{\xi})$, where $\xi \in \mathfrak{se}(2)$. Here, $\rho(\cdot)$ is a Huber noise model.

MAP inference over the pose graph with Gaussian noise models [3] is done by maximizing the product of all factor graph potentials. We initialize the solution from a greedy spanning tree and then optimize using GTSAM [2].

We follow the official GTSAM's PGO implementation example [4].

Once the pose graph is optimized, we use the estimated poses and room layout to create the final floorplan.

10 Ablation Experiments Using Oracle W/D/O Detection

In this section, we perform ablation experiments comparing global pose estimation and floorplan reconstruction results with estimated W/D/O locations and estimated layout, vs. a baseline that has access to ground truth W/D/O detections and ground truth layout.
How much worse is performance with predicted W/D/O and predicted layout vs. annotated D/W/O and annotated layout? In Table 3, we compute an upper bound for the completeness of our method, by using human-annotated W/D/O and human-annotated layout as input to the system. This measures the ability of the CNN to reason

[4] https://github.com/borglab/gtsam/blob/develop/python/gtsam/examples/Pose2SLAMExample.py

about photometric signal in a less noisy setting (there is still noise from HoHoNet). Note that annotations are not perfect.

The results indicate the already-strong localization precision of our system, with roughly similar camera pose estimation errors (in rotation and translation). We provide no vanishing-point axis-alignment post-processing to these generated relative poses, which leaves the ground-truth (GT) based system susceptible to higher rotation errors. However, the translation error of the model with access to GT W/D/O is still lower on average (22 cm vs. 25 cm).

With GT layout and GT W/D/O, the floorplan IoU is 91% higher – 0.86 median IoU vs. 0.45 IoU with our predicted poses and layout. The percentage of cameras localized is also much higher (93.44% vs. 57.14%) than the system without access to GT. These results are extremely promising, underscoring the significant potential for further improving the floorplan reconstruction completeness of our system by improving the layout estimation and W/D/O detection network.

Table 3: Ablation experiments on global pose estimation, comparing performance with estimated W/D/O locations and estimated layout, vs. performance with ground truth W/D/O locations and ground truth layout (*oracle*).

Input	Localization %		Tour Avg. Rotation Error (deg.)		Tour Avg. Translation Error (meters)		Floorplan IoU	
	mean	median	mean	median	mean	median	mean	median
Predicted WDO + Predicted Layout	60.70	57.14	**3.73**	**0.17**	**0.80**	0.25	0.49	0.45
GT WDO + GT Layout	**88.58**	**93.44**	5.02	0.21	0.98	**0.22**	**0.78**	**0.86**

11 Comparison with Extremal SfM [7]

In this section, we provide additional comparisons with concurrent work by Shabani *et al.* [7] (See Tables 4 and 5).

Differences in Assumptions.

- Shabani *et al.*'s one-panorama-per-room assumption limits the number of door hypotheses, as they assume door surface normals must point in opposite directions, whereas we consider twice as many hypotheses, i.e. when surface normals may also point in the same direction. The restricted door hypotheses would be analogous to querying a ZInD oracle for ground truth adjacency, which we do not do.

Differences in Method.

- They do not use openings. Accordingly, the rooms cannot be too large or complex enough to have significant amounts of self-occlusion.
- Exponential time: They rely upon a tree-type graph topology. They verify on all possible global graph configurations (room snapping combinations) with graph neural network. Conv MPN [10] machinery (follows up on HouseGAN [5] and HouseGAN++ [6] machinery). This requires exponential time.

Table 4: Summary of comparison of our method vs. that of *Extremal SfM* by Shabani *et al.* [7].

	Extremal SfM [7]	Ours
Computational Complexity	Exponential Time $O(n!)$	Polynomial Time $O(n^2k^2)$
# Panoramas / Room	1	≥ 1
# Panoramas / Floor	3.4	23.2
# Floors in Test Set	46	291
Door Configuration	Opposite facing surface normals	Any configuration
Supported Room Type	Small size, Little self-occlusion	Any
Home Type	Apartment	Residential Home
Wall Assumption	Manhattan	None
Evaluated Error Types	Translation	Rotation and translation
Input Signal	BEV mask	BEV (image)
Verifier Type	GNN on tree-structured graph	CNN on pairwise renderings

Table 5: A more detailed comparison of our input, method, and evaluation vs. those of Shabani *et al.* [7].

Comparison Type	Extremal SfM (Shabani et al. [7]	Ours
Input	– Assumes no room shape overlap in dataset. – Requires exactly one panorama per room. – Requires each input panorama to see most of the room, including W/D. This would be problematic for complex rooms where one panorama sees only a fraction of the room. – Demonstrated on apartments.	– Handles any amount of overlap, but not zero overlap. – Requires one or more panoramas per room. – Has no requirement on panorama capture locations. – Evaluated on ZInD with complex room layouts, including open floorplans.
Method	– Uses W/D alignment, but not openings. – Uses HorizonNet [8] for layout and separately predicted W/D objects at test time. – Uses room topologic information and BEV semantic masks (with no photometric information) directly as input to GNN. – Relies on a tree-type graph topology. They verify on all possible global graph configurations (room snapping combinations) with graph neural network Conv. MPN [10]. – Runs in exponential time with number of rooms.	– Uses W/D/O alignment to generate pairwise initial pose hypotheses. – Uses predicted room layout from HorizonNet [8] with joint W/D/O predictions and wall-floor boundary uncertainty. – Uses BEV photometric signal in panorama pairwise pose verification. – Uses global filtering and optimization similar to traditional Global SfM. Our method is also able to refine coarse panorama poses. – Runs in polynomial time with number of rooms.
Evaluation	– Evaluated on a small dataset of 46 apartments. The dataset contains 3.4 panoramas per apartment with all Manhattan room layouts. – Generates top-5 possible floorplans. – Localization is considered a success if any of K possible solutions has estimated global poses with mean positional error below δ meters. Rotation error is not considered.	– Evaluated on the test split of ZInD with 291 floorplans of residential homes. ZInD contains 23.2 panoramas per floorplan with Manhattan and non-Manhattan room layouts. – Generates 1 final floorplan. – Evaluated by per-panorama average pose rotation and translation error, and localization completeness.

Differences in Evaluation.

- Instead of computing a mean error per pano, they just count the successes. This does not take into account catastrophic failures (see their Figure 9). We have been evaluating as "you get one shot, and for every pano you try to localize, it will go into your mean error". By comparison, they find the minimum subset out of all panos they localized that are good.

Unfortunately, at the time of our submission their code and dataset used were not publicly available, so a comparison of accuracies on a common dataset is not possible at this time.

12 Analysis of Computational Complexity

We compute over the ZInD train/val/test sets putative estimates of these constants n, k. On average for each ZInD tour, we find $n \approx 23.2$. When evaluating $\mathcal{O}(n^2k^2)$, we find that each floor has on average 10795 (mean) and 8188.0 (median) putative alignments.

After pruning away impossible hypotheses via width ratios, n is unchanged but k is reduced, leaving $\mathcal{O}(n^2k^2) \approx 5804.5$ (mean) and 3441.0 (median). Although there are just as many panoramas to match with, we effectively have fewer instances of each W/D/O type we can feasibly match with (thus reducing k).

This leads to a highly imbalanced classification problem, with negative-to-positive ratios of 18:1 on average, when using predicted layout and predicted W/D/O locations.
Oracle Layout Generator Baseline. For alignments generated from GT W/D/O and GT layout, we halve the computational complexity by discarding those with penetrated freespace (an average negative-to-positive ratio of 7:1). For predicted layout, we cannot prune alignments by freespace penetration heuristics due to arbitrary predicted locations of openings in large rooms.

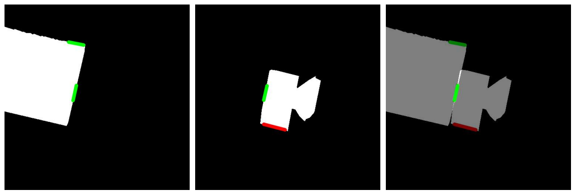
13 Details on Layout-Only Rasterization Baseline

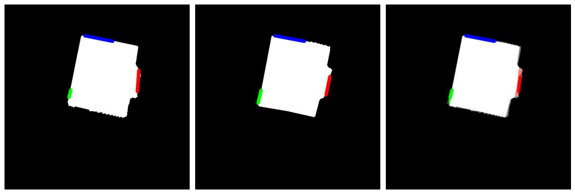
In Table 2 (Section 6) of the main paper, we reported results of a model that has no access to photometric information as input, but only to rasterized BEV layout. In Figure 8, we show examples of such input.

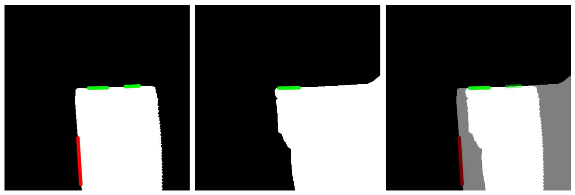
14 Additional Discussion Points

14.1 Accuracy vs. Amount of Visual Overlap

Poor accuracy for examples with small support in the dataset (see Figure 9) shows the potential for hard negative mining in future work. Negative examples with high IoU (see Figure 9d, right) are few in the dataset, and present high error (see Figure 9a, right).


(a) Positive example pair 1.


(b) Positive example pair 2.


(c) Negative example pair 1.

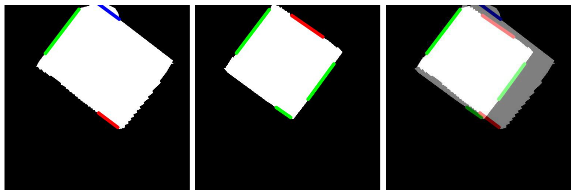
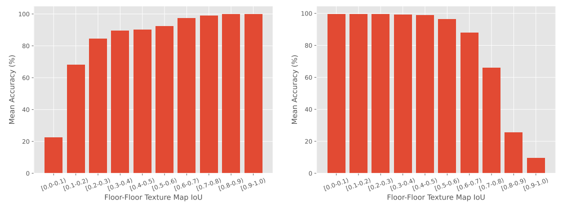
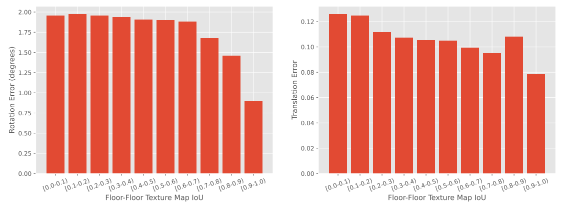

(d) Negative example pair 2.

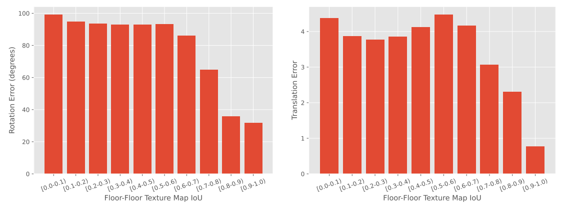
Fig. 8: Examples of layout-only rasterized input. Each row represents an alignment pair. ***Left:*** rendering for panorama 1. ***Middle:*** rendering for panorama 2. ***Right:*** blended images (for visualization only).



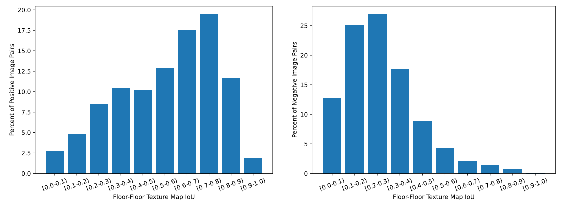
(a) Classification Accuracy vs. Visual Overlap (for GT Positives vs. Negatives)



(b) Rotation and Translation Error vs. Visual Overlap (for GT Positives)



(c) Rotation and Translation Error vs. Visual Overlap (for GT Negatives)



(d) Visual Overlap Distribution in ZinD (for GT Positives vs. Negatives)

Fig. 9: **(Row 1)** Classification accuracy vs. overlap for the GT **positive** class only **(left)** and **negative** class only **(right)** for ResNet-152 model. **(Row 2)** Relative pose rotation error **(left)** and translation **(right)** vs. amount of visual overlap for GT *positive* examples. **(Row 3)** Relative pose rotation error **(left)** and translation **(right)** vs. amount of visual overlap for GT *negative* examples. **(Row 4)** Distribution of visual overlap (IoU) over rendered buildings for positive pairs **(left)** and negative pairs **(right)** from SE(2) alignments generated from predicted W/D/O's.

14.2 Additional Details on Evaluation Metrics

To compute both the ground truth mask and estimated binary mask for IoU computation, we rasterize the scene to a grid resolution of 10×10 centimeters per cell (we found an even finer resolution did not affect results significantly).

Error Rate in Supervision Generation. Noisy generation of ground truth is susceptible to false negatives. In other words, the generator falsely assumes that there are no 'positive' alignments for certain panorama pairs, due to errors just above the maximum tolerated rotation or translation thresholds. At inference time, these lead to false positive predictions, as the model identifies these pairs as positives, conflicting with the ground truth. We find that for spatially adjacent rooms, no putative 'positive' hypothesis is generated for less than 9.67% of panorama pairs, due to layout estimation or W/D/O prediction errors.

14.3 Model Learning

Previous work has proven that domain-knowledge of indoor space, such as room intersections, loop closure, and multi-view alignment, can be helpful in solving the 'room merge' problem [1]. On the other hand, visual overlap of floor and ceiling areas from different texture images provides helpful clues, such as light source reflections, paneling direction of wood flooring, and shared ceiling features, to verify panorama registration [4]. We extract undistorted floor and ceiling orthographic views from each panorama using inferred depth and register each view using an estimated W/D/O alignment hypothesis. While inferred depth signal alone suffers from inaccuracies at very close or far range and near reflective objects such as mirrors, the orthographic views still contain small distortion and therefore provides a strong signal for alignment verification. We train a model to implicitly verify the aligned texture signals (such as light source reflections, paneling direction of wood flooring, and shared ceiling features), as well as model other priors on room adjacency, such as the fact that bathrooms and bedrooms are often adjacent.

While ceiling features on a mosaiced ceiling image have been used for robot localization for at least two decades [4], success of registration using traditional explicit image-based matching is highly dependent on significant appearance similarity. This is typically not the case for our work, due to the large baselines and potentially very different times of capture.

Aligned orthographic views can help identify shared floor texture around room openings, identifying common objects (i.e. refrigerators), or known priors on room adjacency, such as the fact that bathrooms and bedrooms are often adjacent. axis-alignment of walls between two layouts. Because floorplan adjacency is governed by strong priors, such as primary bedrooms are attached to primary bathrooms, such signals can be learned.

Whereas previous works have employed domain-knowledge to manually define features for room-merge costs to employ in ranking [1], we set out to learn such features from data. Previous work has defined costs that aim to minimize room intersections, maximize loop closure, maximize multi-view alignment of semantic elements, and produce the most axis-aligned floorplans [1]. However, some of these costs are only applicable if an annotator can identify which panoramas were captured in different rooms, as layouts within the same room should instead *maximize* room intersection, while those captured in separate rooms should *minimize* room intersection. Each window from panorama $\mathbf{I}_a$ should reproject onto a window in panorama $\mathbf{I}_b$ only if the panoramas were captured in the same room, which is an unknown latent variable. IoU should be high between the two overlaid room-layouts; however, this is not true if they are in separate rooms (cross-room). If, on the other hand, we knew a "same-room" label, then layout-IoU would be useful during localization.

Available Signals for Learning Priors. Many possible complementary signals can be employed in the reconstruction problem. The $360°$ image suffers from significant distortion, while inferred depth suffers from inaccuracies at very close or far range and near reflective objects such as mirrors. Taken together, however, undistorted texture can be extracted in an orthographic manner. As the floor alone can have highly homogeneous texture or varied lighting, the ceiling can also provide helpful clues. Registration of inferred layout alone, with consideration of the image content, can lead to implausible arrangements. No single signal is sufficient.

Human annotators use a variety of different cues to solve the merge task, most of them grounded in visual features within the image. For example, they often rely upon identifying shared floor texture around room openings, identifying common objects (i.e. refrigerators), or known priors on room adjacency, such as the fact that bathrooms and bedrooms are often adjacent. axis-alignment of walls between two layouts. Because floorplan adjacency is governed by strong priors, such as primary bedrooms are attached to primary bathrooms, such signals can be learned. Additional such relationships include hallway-to-bedroom adjacency.

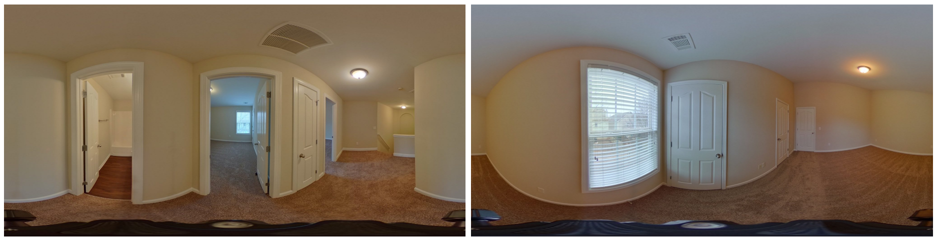
15 Additional Examples of Illumination Changes

In Figure 10, we provide an example of extreme illumination changes in ZInD, which prevent the use of classical image alignment algorithms for BEV image registration.

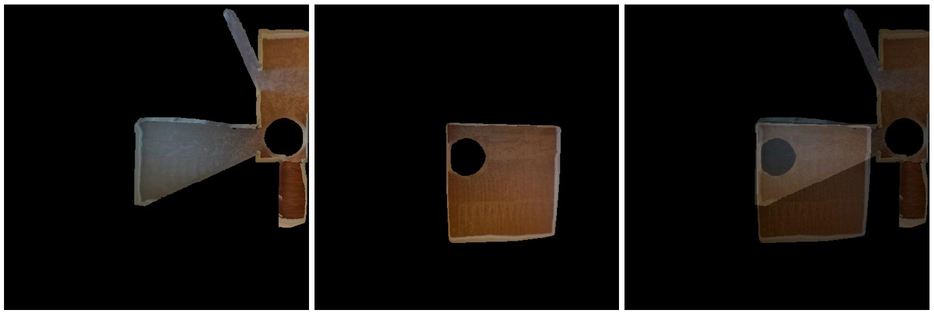
16 Verifier Data Augmentation and Training Details

Verifier data augmentation. We resize BEV texture maps to 234×234 resolution, sample random 224×224 crops, randomly flip them, and then normalize crops using the ImageNet mean and standard deviation.

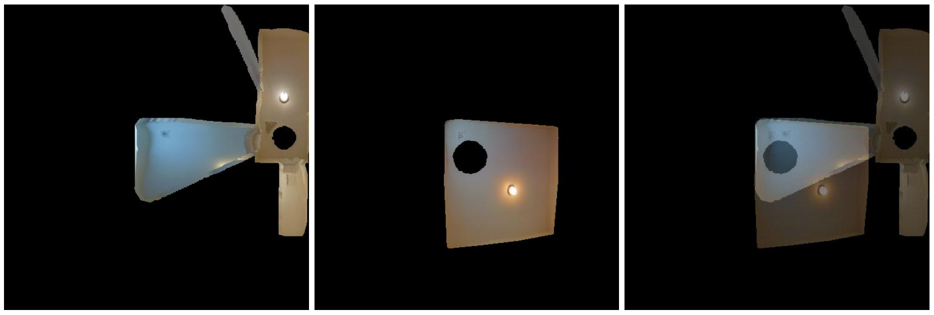
Verifier training. We use a ResNet-152 architecture with ImageNet-pretrained weights, training for 50 epochs, with an initial learning rate of 1×10^{-3}, polynomial learning rate decay with a decay factor of 0.9 per iteration, and a weight decay of 1×10^{-4}. We use a batch size of 256 examples on 3 NVIDIA Quadro RTX 6000 GPUs.



(a) Input panorama pair.



(b) Orthographic floor texture maps, with an extreme illumination change.



(c) Orthographic ceiling texture maps.

Fig. 10: Example of an extreme illumination change, as the carpet color appears to shift from brown to grey **(middle)**, and ceiling from warm yellow to light blue **(bottom)**.

17 Ethical/Privacy/Transparency/Fairness/Social Impact Concerns

Floor plan reconstruction, in general, could potentially lead to privacy issues. However, schematic floorplans are able to abstract away details of the real interior space, thereby revealing the layout and functionality of a home while hiding personal information (PI) and personally identifiable information (PII) information from the images used to reconstruct it. In other words, we can build the floorplan from 360 panos and then immediately use the floorplan as a medium (to convey the space), while suffering from fewer privacy issues compared to releasing all the 360 images used to create it.

References

1. Cruz, S., Hutchcroft, W., Li, Y., Khosravan, N., Boyadzhiev, I., Kang, S.B.: Zillow indoor dataset: Annotated floor plans with 360deg panoramas and 3D room layouts. In: CVPR (2021)
2. Dellaert, F.: Factor graphs and GTSAM: A hands-on introduction. Tech. rep., Georgia Institute of Technology (2012)
3. Dellaert, F.: Factor graphs: Exploiting structure in robotics. Annual Review of Control, Robotics, and Autonomous Systems **4**, 141–166 (2021)
4. Dellaert, F., Burgard, W., Fox, D., Thrun, S.: Using the condensation algorithm for robust, vision-based mobile robot localization. In: CVPR (1999)
5. Nauata, N., Chang, K.H., Cheng, C.Y., Mori, G., Furukawa, Y.: House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In: ECCV (2020)
6. 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: CVPR (2021)
7. Shabani, M.A., Song, W., Odamaki, M., Fujiki, H., Furukawa, Y.: Extreme structure from motion for indoor panoramas without visual overlaps. In: ICCV (2021)
8. Sun, C., Hsiao, C.W., Sun, M., Chen, H.T.: Horizonnet: Learning room layout with 1D representation and pano stretch data augmentation. In: CVPR (2019)
9. Sun, C., Sun, M., Chen, H.T.: Hohonet: 360 indoor holistic understanding with latent horizontal features. In: CVPR (2021)
10. Zhang, F., Nauata, N., Furukawa, Y.: Conv-MPN: Convolutional message passing neural network for structured outdoor architecture reconstruction. In: CVPR (2020)
11. Zhang, Y., Song, S., Tan, P., Xiao, J.: Panocontext: A whole-room 3D context model for panoramic scene understanding. In: ECCV (2014)