BADGR: Bundle Adjustment Diffusion Conditioned by GRadients for Wide-Baseline Floor Plan Reconstruction

Yuguang Li[1,2]✉* Ivaylo Boyadzhiev† Zixuan Liu[1] Linda Shapiro[1‡] Alex Colburn[1‡]
[1]University of Washington [2]Zillow Group

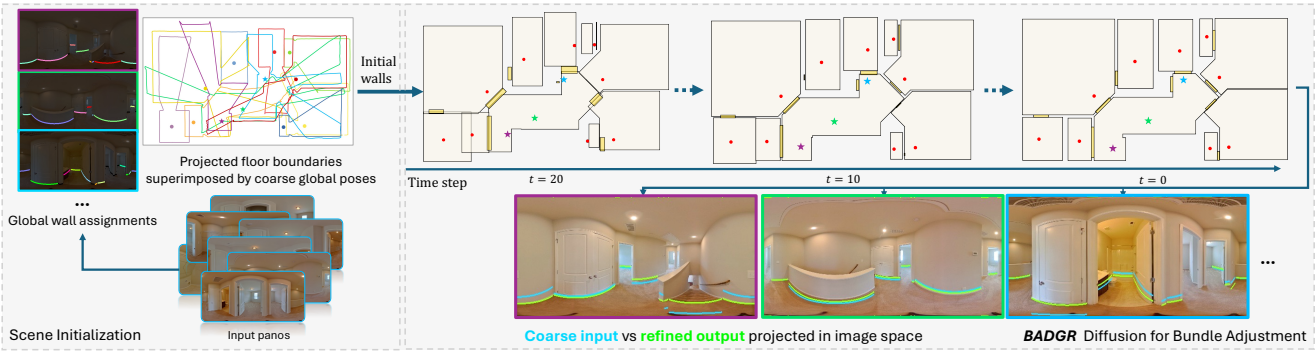

Figure 1. Overview of *BADGR*, a diffusion-based bundle adjustment (*BA*) model for generating precise, view-consistent camera poses and floor plan layouts. *BADGR* uses per-image floor boundaries and image column-to-wall assignments (upper left) as coarse input, refining poses and layouts through a gradient-conditioned denoising process (upper right). The bottom right shows view consistency by projecting the output layouts with the estimated poses.

Abstract

Reconstructing precise camera poses and floor plan layouts from wide-baseline RGB panoramas is a difficult and unsolved problem. We introduce *BADGR*, a novel diffusion model that jointly performs reconstruction and bundle adjustment (BA) to refine poses and layouts from a coarse state, using 1D floor boundary predictions from dozens of sparsely captured images. Unlike guided diffusion models, *BADGR* is conditioned on dense per-column outputs from a single-step *Levenberg Marquardt* (*LM*) optimizer and is trained to predict camera and wall positions, while minimizing reprojection errors for view consistency. The objective of layout generation from denoising diffusion process complements *BA* optimization by providing additional learned layout-structural constraints on top of the co-visible features across images. These constraints help *BADGR* make plausible guesses about spatial relationships, which constrain the pose graph, such as wall adjacency and collinearity, while also learning to mitigate errors from dense boundary observations using global context. *BADGR* trains exclusively on 2D floor plans, simplifying data acquisition, enabling robust augmentation, and supporting a variety of input densities. Our experiments validate our method, which significantly outperforms the state-of-the-art pose and floor plan layout reconstruction with different input densities. Visit project website at: https://badgr-diffusion.github.io.

*Work done at University of Washington.
†Work done as an independent researcher.
‡Equal contribution.

1. Introduction

Reconstructing floor plan layouts and camera poses has become an important task with many applications such as virtual touring, interior design, and autonomous navigation. High spatial accuracy in both objectives is essential for high-fidelity downstream tasks, such as cross-view scene editing and dense reconstruction. Existing solutions for image-based layout reconstruction are either coarse, limited to a single room, or, while more accurate, require either densely captured image inputs or sparser capture with RGB-D cameras, which can be costly in terms of equipment, data bandwidth and capture efforts.

This work aims to accurately reconstruct camera extrinsics and floor plan layouts from sparsely captured 360° panoramas without prior pose information [15]. Specifically, our goals are to: (1) reconstruct the floor plan as a unique set of closed-loop polygons defining rooms and doors [15, 55]; (2) estimate each camera pose for view-consistency [17]; (3) accommodate diverse capture densities, down to one image per room; and (4) ensure that generated floor plans remain plausible within the natural distribution of training data, even when certain walls are occluded.

Accurate reconstruction of camera poses and spatial layouts in wide-baseline indoor environments is challenged by limited co-visibility and sparse features. This process demands not only view-consistency but also layout-structural constraints, such as *Manhattan* or *Atlanta* frameworks [35], wall thickness, collinearity, and prior knowledge of room

layouts [15, 23, 58]. We propose *BADGR*, a conditional denoising diffusion probabilistic model (*DDPM*) trained to reconstruct and bundle adjust camera poses and layouts. *BADGR* employs a planar bundle adjustment (*BA*) module, to provide geometric guidance for conditioning the *DDPM* to maximize view-consistency from angle-constrained layouts and poses. The *DDPM* is also trained for the floor plan generation task. Combined with a reprojection loss, *BADGR* performs bundle adjustment through posterior sampling, with learned layout-structural constraints and the ability to handle noise from input features. The generative ability of the *DDPM* allows *BADGR* to predict plausible shapes of occluded layout sections based on training data. The *non-Markovian* inference process, i.e., predicting x_{t-1} from x_t by predicting x_0, allows *BADGR* to combine a score-based generative model with a nonlinear optimization process, without specifying step size during training. *BADGR* differs from guided-diffusion style models [11, 12, 16, 52], where the gradients from differentiable objective functions are used only during inference to guide a pre-trained diffusion model, leading to issues like slow convergence and deviation from data manifold. Our experiments show that *BADGR* is more accurate in performing *BA* tasks compared to guided-diffusion style models.

The contributions of this work are: 1) *BADGR* is the first learning-based approach to jointly refine deformable room layouts from polygon-based floor plans and camera poses from sparsely captured RGB panoramas, guided by visually-derived features; 2) *BADGR* contains a novel approach to train a diffusion model for both nonlinear optimization (i.e., *BA*) and generation tasks (i.e., floor plan generation), allowing it to reconstruct poses and layouts from visual inputs and make reasonable guesses from learned layout-structural constraints; 3) *BADGR* obtains state-of-the-art accuracy in wide-baseline camera pose estimation and layout estimation with multi-view 360° panoramas.

2. Related Work

Our work bridges floor plan reconstruction and wide-baseline pose estimation, jointly reasoning over layout constraints and cross-view consistency.

Image-based Floor Plan Reconstruction involves estimating camera poses and creating a unique set of layout polygons. Research has shown good accuracy in producing floor plans in such formats from registered RGB-D point cloud scans [4–7, 40, 55]. Generative models [27, 28, 37] also demonstrate deep learning's ability to model layout constraints, enabling floor plan generation from abstract inputs, like bubble diagrams [28]. Over the years, single-view room layout estimation has been extensively studied [22, 32, 42, 43, 50, 59]. However, noisy or unknown poses remain a key challenge for multi-view, wide-baseline layout reconstruction, as it requires 1) generating a single fused layout, 2) high accuracy in both camera poses and layouts for consistent views. Prior approaches attempted to regress poses and layouts from panorama pairs [23, 41, 51], yet often lack view-consistency without joint optimization [51].

Wide-Baseline Pose Estimation Analysis has shown [21, 23] that wide-baseline indoor pose estimation is challenging to traditional *Structure from Motion* (*SfM*), as images often contain featureless regions, repetitive textures, or narrow passageways, which limit co-visibility and cause drastic appearance changes across images [15]. Semantic constraints can help *SfM* establish anchors and loop-closures, but existing approaches often rely on heuristics and lack understanding of structural layouts [13, 14]. [23, 36] explored wide-baseline reconstruction as discrete problems, piecing rooms together like puzzles [20] and using vanishing points [56] to improve camera rotation estimates. However, these approaches can produce coarse poses and layouts with missing rooms from a larger floor plan. Recently, direct pose regression achieved good retrieval accuracy in solving coarse poses from a pair of [8, 21, 47] or up-to-5 [29] panorama inputs, along with predictions of dense correspondences among image columns and dense room layouts [21, 41]. A global pose graph can be later formed by merging local poses [23, 29], but this often lacks sufficient accuracy for good view-consistency. Furthermore, structural-layout constraints are frequently violated when superimposing projected single-view layouts on estimated poses [51], as shown in the starting floor plan of Figure 1.

Learned Optimization In traditional *SfM*, robust optimization often integrates sensor data and physical constraints iteratively to manage uncertainties in measurements and features. Recent data-driven approaches improved "front-end" feature extraction and matching [30, 34], adapting to noise and variability more effectively than classical methods. Differentiable optimizer, like differentiable *LM* [31, 46], have been used to train feature extraction with uncertainty prediction [26]. Uncertainty is also modeled from the parameter posterior distributions given initial measurements. *G3R* [9] iteratively optimizes 3D Gaussians using a gradient-conditioned 3D *U-Net*. *PoseDiffusion* [52], *PhysDiff* [54] apply posterior sampling of pre-trained diffusion models for improved reconstruction. However, posterior-guided sampling can run into conflicts between guidance and diffusion flow function, which affects convergence [10]. *Non-Markovian DDPMs*, such as denoising diffusion implicit models (*DDIM*) [38], enable conditional diffusion training without explicit transitions from t to t-1, facilitating integration with nonlinear optimization independent of step size.

3. Problem Statement

Objectives: Given a set of sparsely captured 360° indoor RGB panorama images $\{P^i\}$ in equirectangular projections without pose information, the overall pipeline aims to estimate 3 degrees-of-freedom (*DoF*) camera poses $\{E^i, E^i \in$

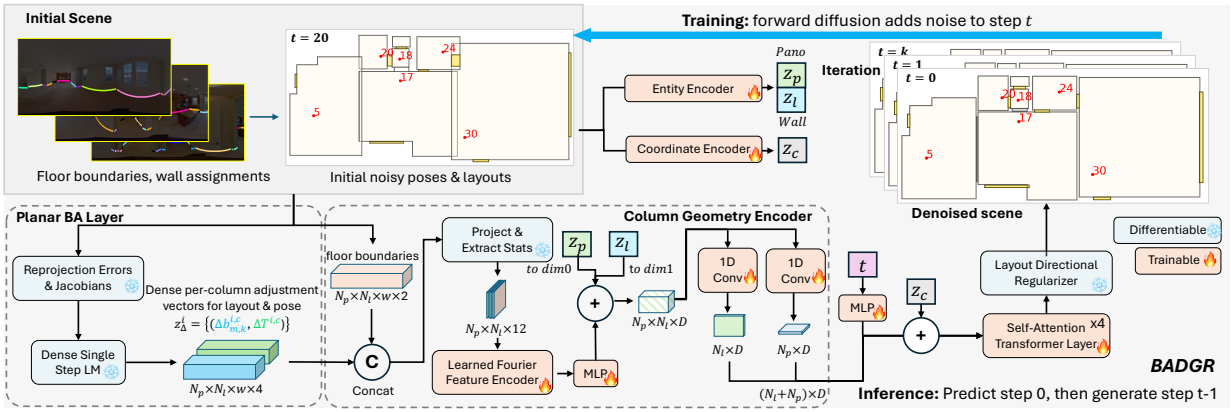

Figure 2. Architecture of *BADGR*. The forward process takes a ground truth scene, i.e. layouts and poses, adds noise to sample step t. The inference process uses a transformer, conditioned on dense per-column adjustments generated by the planar *BA* layer and compressed by the Column Geometry Encoder.

$SE(2)\}$, i.e. xy position, horizontal rotation angle, and 2D floor plans represented as a set of closed-loop polygons $\{V_m, V_m = (v_{m,1}, v_{m,2}, ..., v_{m,k})\}$ for rooms and doors in a single global coordinate system, where m, k is room id and corner id, and $v_{m,k}$ is a 2D xy vertex coordinate. Our main focus, the proposed *BA*-based refinement component *BADGR*, aims to denoise camera xy positions $\mathcal{T}^i$ and angular constrained layouts $\{V_m\}$ by moving walls along their normal directions. *BADGR* optimizes for view-consistency on re-projected floor boundaries, assuming each image column contributes to the positional adjustment of a single wall and a camera, where the column-to-wall relations have been pre-assigned. *BADGR* also learns to maintain the layout-structural constraints of floor plans from training data.

Assumptions: We assume that all images are straightened [56] and can be connected to a single pose graph with co-visibility [15] to nearest neighbor greater than 10%. Relative camera heights are known across the floor. During *BADGR* optimization, wall angles are assumed to be fixed. The number of walls and wall connectivity are given; they can either come from merging the single-view layout estimation, post-processed under the *Atlanta World* assumption [22, 32], or via a quick human-in-the-loop process, see Section 5. Our floor plans follow the *Atlanta World* assumption [35] and have no curved walls.

4. Pipeline Overview

We designed a coarse-to-fine pipeline, as shown in Figure 1, that initializes a scene with estimated camera poses and room layouts, as closed-loop polygons, combining per-view floor boundary predictions and image column-to-wall assignments, similar to semantic column matches across multiple images [2, 50]. This coarse initialization is subsequently refined with *BADGR*, which focuses on accurate reconstruction. Starting from the coarse scene, rotations from estimated poses and walls are corrected via vanishing point snapping [56]. During *BADGR* refinement, the scene is angularly constrained: *BADGR* optimizes only the 2-*DoF* camera xy position $\mathcal{T}^i$ for each pose E^i and the 1-*DoF* line translation $b_{m,k}$ for each wall $l_{m,k}$. While various algorithms could initialize the coarse floor plan for *BADGR*, we outline a specific pipeline in Section 5 using practical algorithms for coarse scene setup. The scene representation is discussed in Section 6, with details of *BADGR*, the proposed learned *BA* diffusion model, in Section 7.

5. Coarse Scene Initialization

During inference, the scene is initialized with a slightly modified *CovisPose* model [21]. *CovisPose* is run on each pair of panoramas on the same floor to predict 1) relative camera pose $\tilde{E}^{(i,j)} \in SE(2)$, 2) floor boundaries $\{\tilde{\mathcal{B}}^i\}$, and 3) cross-view co-visibility, angular correspondences $\{\tilde{\alpha}^{i,j}\}$, $\{\tilde{\varphi}^{i,j}\}$. Additionally, we perform per-column binary classification to identify room corners $\{\tilde{\mathcal{V}}^i\}$. The model is trained on the *ZInD* dataset [15] with the same image pairs as [21] with an additional corner loss function similar to that of [42]. Pose pairs of co-visibility score greater than 0.1 are selected to create a minimal spanning tree of the pose graph, similar to [29]. $\tilde{E}^{i,j}$ are corrected through axis alignment with a 45° interval using predicted vanishing angles [56] prior to computing global poses $\tilde{E}^i$.

The per-pano floor boundaries $\{\hat{\mathcal{B}}^i\}$ are further refined and aggregated into uniquely identifiable set of global walls $l_{m,k}$, shared across P^i via an automatic process using room corners $\{\tilde{\mathcal{V}}^i\}$. Finally, an annotator uses an interactive application to provide global wall connectivity, and add missing room corners with their rough initial positions. The number of room corners and wall orientations are static input to *BADGR*. More details are provided in Supplementary.

6. Scene Representation for *BA* Optimization

Our proposed representation aims to uniquely define walls and cameras while assigning image columns to global walls, allowing *BA* to perform cross-view reprojections at any scene state. Each column links either to a specific wall or

remains unassigned, simplifying reprojections by avoiding wall occlusion handling during floor boundary rendering. The global scene comprises room layouts $\{V_m\}$, doors as polygons with at max N_l walls, camera extrinsics $\{E^i\}$ of N_p panoramas, per-panorama floor boundaries $\{\hat{\mathcal{B}}^{i,c}\}$ for columns $\{\mathcal{C}^{i,c}\}$ with image width w, and a column-to-wall semantic assignment represented as a one-hot 3D array of shape $N_p \times w \times N_l$ for mapping $\{\mathcal{M} : \mathcal{C}^{i,c} \rightarrow l_{m,k}\}$. The layout walls $\{V_m\}$ are represented as line segments $\{l_{m,k} \mid l_{m,k} = (v_{m,k}, v_{m,k+1})\}$ with each line represented in the Hesse normal form [1] allowing us to easily work with rotations and offsets $(\overrightarrow{v_{m,k}}, b_{m,k})$ from the origin. *BADGR* optimizes layout vertices V_m and camera positions $\mathcal{T}^i$, with these parameters normalized within $[-1, 1]$ as a continuous 2D coordinate array. Constant scene parameters include camera and wall rotation vectors $\mathcal{R}_i$, $\overrightarrow{v_{m,k}}$, camera height z^i and column-to-wall assignments $\{\mathcal{M}\}$. Details on scene initialization are covered in Section 5.

7. Bundle Adjustment Diffusion

Wide-baseline indoor reconstruction often suffers from a lack of robust matching features with sub-pixel accuracy. However, each wall can be observed by several image columns, with their floor boundary modeled as a line, as shown in Figure 3. *BADGR* tackles the multi-view floor plan and pose reconstruction problem using a planar bundle adjustment (*BA*), minimizing reprojection errors between the input predicted floor boundaries from image-based models and the projected wall positions. These errors are computed at the column level and are used to adjust the wall translation along its normal direction and the corresponding camera pose, optimized via a *Levenberg-Marquardt* (*LM*) step.

Noise in the floor boundaries can introduce errors in the final *BA* results. To mitigate the noisy signal and incorporate learned layout-structural constraints, *BADGR* integrates the planar *BA* mechanism into a conditional denoising diffusion process. Instead of averaging column-wise wall and camera movements, the model encodes these dense signals and uses them to condition a transformer-based diffusion model. In this framework, posterior sampling is employed to generate the final adjustments, using raw *BA* adjustments as inputs. The transformer model predicts the *xy* positions of camera poses and room vertices. To facilitate iterative interaction with planar *BA*, an angle-constrained scene representation is used, along with a *Layout Directional Regularizer* (*LDR*) to form a bidirectional map between 2D room vertex positions and wall positions while maintaining fixed angles. The details of this process are explained in the following section.

Architecture: Figure 2 shows the architecture of our proposed optimization pipeline. Taking inspiration from *HouseDiffusion* [37], *BADGR* has a diffusion model [19] based architecture with a truncated denoising inference process [25, 57] during training and inference. At each time t, our model takes a scene of 2D positions, i.e. $(\{\mathcal{T}^{i,(t)}\}, \{V_m^{(t)}\})$, conditioned on: 1) the wall and panorama metadata encoded by an Entity Encoder discussed below, 2) the floor boundary and adjustments embedding encoded by the Column Geometry Encoder, and generates an updated scene, i.e. $(\{\mathcal{T}^{i,(t-1)}\}, \{V_m^{(t-1)}\})$, for time $t-1$.

The architecture consists of: (a) a planar *BA* layer, (b) a Column Geometry Encoder module, where column-wise dense adjustments are encoded into a 1D embedding for each wall and panorama, (c) a Coordinate Encoder to embed 2D vectors for directional vector and *xy* coordinates, d) an Entity Encoder to generate an entity embeddings for metadata of each wall and panorama, functioning similarly as a positional encoding, and (e) a self-attention Transformer denoiser to integrate the embeddings along with time step and to estimate new 2D positions.

Entity Encoder: Similar to the "input conditions" from *HouseDiffusion*, a size-D identity embedding is generated for each wall and camera using metadata, e.g. wall direction vector, one-hot room type, room id and vertex id for wall, and camera direction vector, one-hot camera ids for panorama. Wall and camera embeddings are generated with separate *MLP* units.

Algorithm 1: BA Optimization on Column $\mathcal{C}^{i,c}$

Input: Wall parameters $(b_{m,k}, \overrightarrow{v_{m,k}})$, camera 2D pose $\mathcal{T}^i$, $\mathcal{R}^i$, camera height z^i, and floor boundary $\tilde{\mathcal{B}}^{i,c}$

Output: $\Delta b^{i,c}_{m,k}, \Delta \mathcal{T}^{i,c}_{m,k}$

$\overrightarrow{ray}^{i,c} = (\mathcal{T}^i, [sin\mathcal{R}^i, cos\mathcal{R}^i])$

$\overrightarrow{wall}_{m,k} = (\overrightarrow{v_{m,k}}, b_{m,k})$

Global CS [1]: $pt^{i,c}_{m,k} \leftarrow$ **Intersect**$(\overrightarrow{ray}^{i,c}, \overrightarrow{wall}_{m,k})$

Camera CS: $pt^{i,c}_{m,k} \leftarrow$ **GlobalToCam2D**$(pt^{i,c}_{m,k}, \mathcal{T}^i, \mathcal{R}^i)$

Projected boundary: $\hat{\mathcal{B}}^{i,c}_{m,k} =$ **Cam2DToPixel**$(pt^{i,c}_{m,k}, z^i)$**.row**

Reprojection error function: $\epsilon^{i,c}_{m,k}(b_{m,k}, \mathcal{T}^i) \leftarrow |\hat{\mathcal{B}}^{i,c}_{m,k} - \tilde{\mathcal{B}}^{i,c}|$

Jacobian matrix function: $\mathcal{J}^{i,c}_{m,k}(b_{m,k}, \mathcal{T}^i) \leftarrow$ **Jacrev**$(\epsilon^{i,c}_{m,k})$

Single-step *LM*:
$\Delta b^{i,c}_{m,k}, \Delta \mathcal{T}^{i,c}_{m,k} \leftarrow$ **LM**$(\epsilon^{i,c}_{m,k}, \mathcal{J}^{i,c}_{m,k}, b_{m,k}, \mathcal{T}^i)$

Final wall adjustment: $\Delta b^{i,c}_{m,k} \leftarrow$ **AdaptiveHuber**[3]$(\Delta b^{i,c}_{m,k})$

Final pose adjustment: $\Delta \mathcal{T}^{i,c}_{m,k} \leftarrow$ **AdaptiveHuber**$(\Delta \mathcal{T}^{i,c}_{m,k})$

Planar *BA* Layer: This layer generates camera and wall position adjustments by comparing the projected floor boundary from a given scene state, i.e. layouts and camera poses, with the input predicted floor boundary. The adjustments are computed densely on each image column with a *LM* optimization algorithm, which is set to run a single step. The process is demonstrated in Figure 3. Compared to *gradient descent* (*GD*) based optimization, *LM* is more efficient for convergence as it combines *GD* with the *Gauss-Newton* algorithm [18]. *LM* optimization requires

[1] Coordinate System (CS)

Figure 3. Column-wise planar *BA* module. Positional adjustment for walls and cameras are computed for each image column. At each column, the associated wall $l_{m,k}$ is projected with the current camera pose E^i. The adjustments are computed by comparing the projected point to the floor boundary $\tilde{\mathcal{B}}^{i,c}$ value. The dense per-column adjustments are estimated in parallel with our *BA* layer implementation.

the computation of the *Jacobian* matrix from each adjusted parameter at each column position. To enable efficient training of the proposed learned *BA* diffusion model, the *LM*-adjusted parameters for each feature column are limited to three primary values: $\Delta b^{i,c}_{m,k}$ representing wall bias and $\Delta\mathcal{T}^{i,c}_{m,k}$ denoting the *xy* displacement of the camera. This approach reduces computational complexity and memory overhead, enabling an efficient planar *BA* implementation as a PyTorch layer. However, this constraint necessitates pre-assigning each image column to a designated global wall to streamline the reprojection process by directly referencing the corresponding wall and camera positions without occlusion checks for multiple walls. This initial column-to-wall assignment is established during scene initialization, using feature-matching models such as *CovisPose* to ensure robust alignment prior to *BADGR* optimization. The *BA* layer is implemented with the differentiable non-linear optimization library *Theseus* [31]; this library builds on PyTorch and applies a sparse solver with both CUDA and CPU implementations. See Algorithm 1 for implementation details.

Column Geometry Encoder: This encoder is designed to compress the dense adjustments $(\Delta b^{i,c}_{m,k}, \Delta\mathcal{T}^{i,c}_{m,k})$ and the floor boundary $\{\tilde{\mathcal{B}}^{i,c}\}$ into a 1D embedding for each wall and camera. The input array of shape $N_p \times N_l \times w \times 6$ is stored as three 2D vectors densely collected from feature columns of N_p panoramas. Each column is assigned to at max a single wall. The resulting 1D embeddings, $N_p \times D$ for walls and $N_l \times D$ for cameras, are computed by compressing the three remaining axes. In our experiments, feature dimension $D = 1024$. Specifically, from a sparse array of shape $N_p \times N_l \times w \times 6$, the Column Geometry Encoder reduces the w dimension by computing the projected mean and standard deviation vectors along their corresponding wall normal and wall vector axes, among the valid values along the w axis. The Coordinate Encoder, which is based on *learned Fourier Features* [45], then encodes each stat vector into a size-D embedding z_c by learning two sets of weights for mean and standard deviation vectors. The weights for mean vectors are also used to encode $(\{\mathcal{T}^i\}, \{V_m\})$. These features are concatenated and fed into a single *Fully-Connected* (*FC*) layer to generate a geometric guidance embedding of shape $N_p \times N_l \times 1024$. Since adjustments $(\Delta b^{i,c}_{m,k}, \Delta\mathcal{T}^{i,c}_{m,k})$ are relatively small numbers compared to $(b_{m,k}, \mathcal{T}^i)$, We scale them by a factor of 100 prior to input to Column Geometry Encoder.

To further compress features at per-wall level, the entity embedding of cameras produced by the Entity Encoder is added into each of the N_l array columns. 2-layer 1D convolutions [48] followed by *LeakyReLu* [33] and a single *FC* layer are applied to produce a per-wall geometric guidance embedding of shape $N_p \times D$. Similarly, the entity embedding of walls is added into the N_p array rows. A separate similar 1D convolutional network is used to generate per-camera geometric guidance embeddings of shape $N_l \times D$.

Transformer Denoiser: Four layers of self-attention mechanisms [49] are used to denoise the input scene and reason about the relationships between different scene entities and geometric guidance. The layers use input / output features of shape $(N_p + N_l) \times D$. The output features are fed into a *FC* layer to produce *xy* coordinates as final outputs for $(\{\mathcal{T}^i\}, \{V_m\})$. In each attention layer, 3 types of attention heads with different masking schemes, i.e. Component-wise Self Attention mask (CSA), Global Self Attention mask (GSA) and Relational Cross Attention mask (RSA), were applied, similar to *HouseDiffusion* [37]. For the additional camera inputs, all cells related to valid cameras were left unmasked. Each masked type above contains 4 heads. The outputs of attention layers are summed and fed into add and norm layers. At the end, a single-layer *MLP* is used to predict diffusion noise from current time stamps t_n to t_0.

Angle-Constrained Layouts: These layouts are critical to connect the planar *BA* module with the Transformer Denoiser in an iterative pipeline. The Planar *BA* module processes layouts as inputs and outputs with line representation $(\overrightarrow{v_{m,k}}, b_{m,k})$ of 1-*DoF* walls, while the Transformer works with 2D coordinates $\{V_{m,k}\}$. Essentially, the Angle-Constrained Layouts enable two-way mappings between $(\overrightarrow{v_{m,k}}, b_{m,k})$ and $\{V_{m,k}\}$, by defining half of the $\{V_{m,k}\}$ *xy* coordinate values predicted from the Transformer Denoiser as irrelevant to the final layout prediction. With wall directions $\overrightarrow{v_{m,k}}$ as fixed vectors, the *DoF* of $\{V_m\}$ are reduced by half using the xy validity mask $\tau_{m,k}$ of the same shape as $\{V_m\}$. $\tau_{m,k}$ is generated in the scene initialization stage, along with the wall directional vector $\{\overrightarrow{v_{m,k}}\}$. The invalid values defined in $\tau_{m,k}$ will be overwritten by outputs of the proposed *Layout Directional Regularizer* (*LDR*).

The *LDR* starts from a point $v_{m,k}$, with two valid xy coordinates, and updates the invalid *xy* positions of next vertex $v_{m,k+1}$, using wall direction $\overrightarrow{v_{m,k}}$, $v_{m,k+1}$ and validility $\tau_{m,k+1}$. The *LDR* is applied around the loop of each layout polygon to update all vertices. Angle constrained walls allow *BADGR* to be trained to predict wall movement along

the normal direction, while still using 2D coordinates to represent the layout for design simplicity. $\tau_{m,k}$ is also used as an input condition and a mask in L2 loss computation.

Diffusion Model: *BADGR* adopts the *DDPM* process to learn to invert a diffusion process which adds noise to data with function $q(x)$ [19]. In the forward process, Gaussian noise is added to directly produce x_t from x_0, same as [19]. We train the *non-Markovian* diffusion processes, to predict noise from x_t to x_0 and uses them to interpolate x_{t-1} [38]. This allows *BADGR* to combine nonlinear optimization and diffusion without explicitly defining each step size. Loss for measuring view-consistency can be directly applied to the predicted x_0, which reuses the weighting scheme of a regular *DDPM* model for different time stamp during training. For inference, *probability flow ODE* [39] is used to iteratively denoise samples.

Loss Function: *BADGR* is trained to perform *BA* optimization, with 1) input conditions from dense column-wise *BA* adjustments, and 2) a reprojection loss, similar to the traditional *BA*, to regularize view-consistency. The loss function can be written as:

$$\mathcal{L}^{(t)} = \mathcal{L}^{(t)}_{L2} + \mathcal{W}_{proj} * \mathcal{L}^{(t)}_{proj} \tag{1}$$

where $\mathcal{L}^{(t)}_{L2}$ is the masked L2 reconstruction loss, and is only computed for valid *xy* coordinates defined by $\tau_{m,k}$, from our Angle-constrained layouts. It's computed as:

$$\mathcal{L}^{(t)}_{L2} = \|((v^{pred}_{m,k}, \mathcal{T}^{i,pred}) - (v^{gt}_{m,k}, \mathcal{T}^{i,gt}) * \tau_{m,k})\|_2 \tag{2}$$

$\mathcal{L}^{(t)}_{proj}$ is the layout-to-image re-projection loss, which is computed among all the columns with pre-assigned global wall for *BA* adjustment using estimated scene at time 0.

$$\mathcal{L}^{(t)}_{proj} = \|\epsilon^{i,c,\widetilde{t=0}}_{m,k}\|_1 \tag{3}$$

The process is described in steps 1-6 in Algorithm 1 measured in pixel units. $\mathcal{W}^{proj}$ is the time independent weight for the projection loss, set to 100.

8. Experiments

We design experiments using two types of floor plan data. First, we employ an end-to-end pipeline involving coarse scene initialization followed by *BADGR*, trained on a newly introduced FloorPlan-60K dataset and evaluated on ZInD [15], which has similar data collection and distribution characteristics. Second, we assess *BADGR* independently by training and testing it on the RPLAN dataset [53], with controlled noise added to both layout and virtual camera poses. In the Supplementary, we report RPLAN-trained accuracy evaluated on ZInD to highlight *BADGR*'s key ability to train on 2D schematic views and generalize across datasets. While it doesn't perform as well as the FloorPlan60K-trained model, RPLAN-trained *BADGR* still significantly outperforms our baseline approach.

8.1. Experiments with FloorPlan-60K Data

We use FloorPlan-60K, an extended version of ZInD [15], generated through a similar production pipeline to provide the scale needed for our diffusion-based *BA* training. FloorPlan-60K includes 68,147 floor plans, with an average of 8.7 rooms per plan and around 6.9 walls per room. Most walls align with Manhattan-world assumptions: 96.2% are at 90 degrees, 3.0% at 45 or 135 degrees, and 0.8% form other angles. We have permission from Zillow to use this dataset, with a public release pending from the owner. At a minimum, model weights will be available for reproducibility. Our training stage uses only layouts and simulated poses, minimizing privacy concerns. All evaluations are conducted on the public ZInD test set, ensuring reproducibility. For end-to-end testing, we use panoramas from ZInD, starting with initial coarse room layouts and positions from the *CoVisPose+* method (see section 8.1.1), which serve as inputs to our main contribution: the learned-based global refinement, or *BADGR*. Details on training and inference are provided in the Supplementary.

8.1.1. Baseline Models

***CovisPose*:** The modified *CovisPose* model (see Section 5) was applied to each pair of straightened panoramas [56] from a given floor. For the first baseline, we used the original *CovisPose* method followed by the *Greedy Spanning Tree* (*GST*) algorithm [29] to generate global poses. For the second baseline, which is also the initial state of *BADGR*, we introduced *CovisPose+*, an improved version of *CovisPose* that incorporates vanishing point snapping in the pairwise pose estimation step before applying the *GST*.

***BA-Only*:** This method applies only the planar *BA* layer to refine camera and wall positions iteratively, starting from the coarse initialization in Section 5. First, per-column adjustments are made as described in steps 8-10 of Algorithm 1. These adjustments are then grouped by wall and camera, and averaged to produce the final position updates. To resolve conflicts during averaging, majority voting selects the dominant direction for each adjustment. The *LM* optimization is run for 100 iterations with a step size 2.5 times larger than the original.

Other Methods: *CovisPose* [21] outperforms point-matching *SfM*-based methods [24, 44] and recent learning-based approaches like *DirectionNet* [8] and *SALVe* [23] in wide-baseline indoor pose estimation. While *GraphCovis* estimates poses from three to five panoramas, it cannot be applied directly to our test cases, which typically involve more than 10 panoramas (see Table 1). Additionally, the lower pose errors shown in [29] are limited to configurations with only three panoramas, unsuitable as a comprehensive baseline. Similarly, *PanoPose* [47] estimates relative poses between pairs of panoramas but lacks publicly available code for reproducibility.

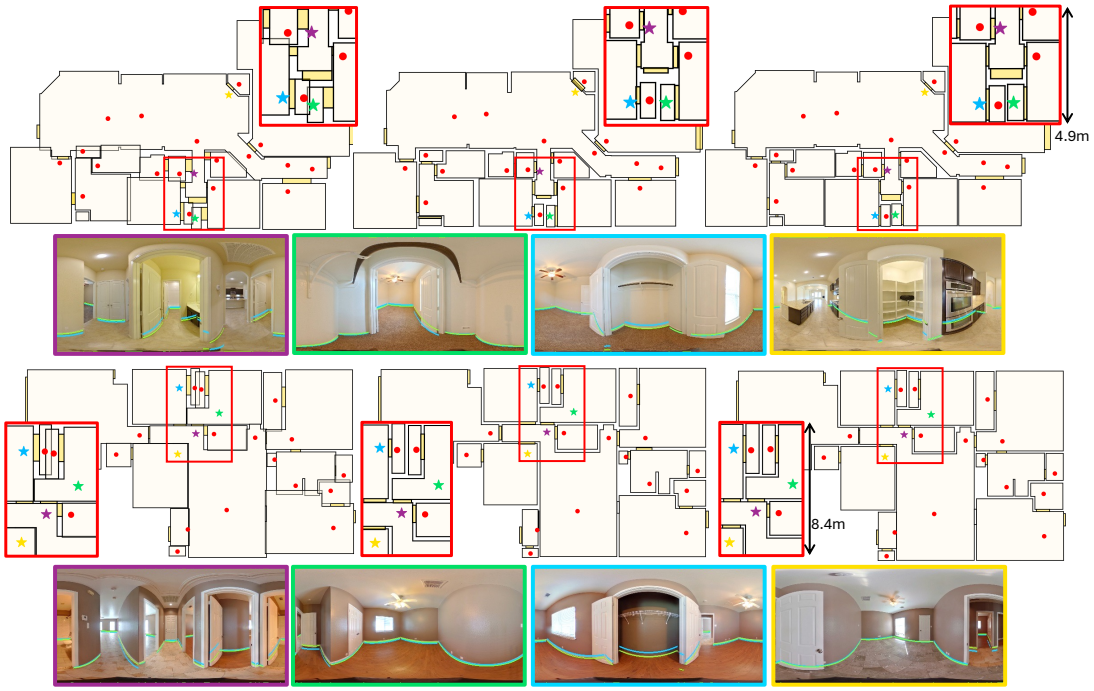

Figure 4. Qualitative results: top-down layouts and poses before (left), after *BADGR* optimization (middle), and GT (right). The reprojected geometry, before and after optimization, is shown in several images, highlighting the improved view-consistency, border colors indicate the capture positions. Example areas with significant improvements are highlighted and zoomed in. More examples in Supplementary.

Table 1. Pose and layout error from predictions on ZInD dataset. Note that *BA*-Only and *BADGR* don't optimize camera rotations, hence share the same rotation errors as *CovisPose+*.

Imgs/ Rm	Methods	Pose Rot.(°)			Pose Transl.(cm)			Visible walls (cm)				#walls
		Mn	Med	Std	Mn	Med	Std	Mn	Med	Std	p90	#panos
0.6	*CovisPose*	1.83	1.25	1.48	22.9	17.4	13.4	14.0	8.0	12.8	32.4	
	CovisPose+	**0.24**	**0.20**	**0.30**	20.7	15.7	12.0	11.5	6.9	10.5	26.0	59.4
	BA-Only	**0.24**	**0.20**	**0.30**	19.1	12.2	10.9	12.8	6.8	11.9	29.3	8.0
	BADGR	**0.24**	**0.20**	**0.30**	**12.2**	**9.5**	**7.2**	**7.1**	**4.5**	**6.7**	**15.3**	
1	*CovisPose*	1.88	1.36	1.40	23.0	18.0	13.2	15.0	9.0	13.0	39.1	
	CovisPose+	**0.26**	**0.21**	**0.30**	19.7	15.6	11.8	12.3	7.0	10.8	27.4	65.7
	BA-Only	**0.26**	**0.21**	**0.30**	17.6	12.6	12.0	12.6	6.7	11.4	27.5	10.4
	BADGR	**0.26**	**0.21**	**0.30**	**11.2**	**8.8**	**6.5**	**7.0**	**4.6**	**6.6**	**15.8**	
2	*CovisPose*	1.90	1.49	1.31	22.3	17.7	12.6	14.7	8.8	12.5	33.6	
	CovisPose+	**0.26**	**0.19**	**0.30**	17.9	14.5	10.2	12.0	7.0	9.0	26.0	65.9
	BA-Only	**0.26**	**0.19**	**0.30**	14.3	10.8	9.2	10.5	6.2	9.9	23.3	16.0
	BADGR	**0.26**	**0.19**	**0.30**	**10.7**	**8.9**	**6.0**	**6.4**	**4.4**	**5.9**	**13.9**	
3	*CovisPose*	1.70	1.19	1.23	21.5	17.1	11.9	14.3	8.5	12.1	31.8	
	CovisPose+	**0.23**	**0.19**	**0.27**	18.1	14.8	10.5	12.3	7.2	10.2	26.9	66.5
	BA-Only	**0.23**	**0.19**	**0.27**	13.4	10.7	8.2	10.6	6.2	9.0	22.1	18.3
	BADGR	**0.23**	**0.19**	**0.27**	**10.6**	**8.9**	**6.0**	**6.6**	**4.3**	**6.0**	**14.6**	

8.1.2. Results

We evaluate models on panorama subsets with varying input densities, as shown in Table 1. Pose errors (mean, median, standard deviation, and p90) are computed after aligning predicted and ground truth pose graphs using RANSAC [29], with similar metrics for layout errors. The number of images per room used in experiments is defined with partial rooms, where a complete ZInD room contains one or more partial rooms [15].

As shown in Table 1, the *BA-Only* baseline reduces pose and layout errors from *CovisPose+* when input density is sufficient, e.g. more than one image per room, with accuracy improving as input images increase. *BADGR* follows a similar trend, consistently achieving lower layout and pose errors across different input densities, including in extreme-baseline scenario with 0.6 images per room. This highlights the effectiveness of *BADGR*'s learned layout-structural constraints and its understanding of global context when compared to the view-consistency-only approach from the *BA-Only* baseline.

8.2. Experiments with RPLAN

The RPLAN dataset [53] is used to evaluate *BADGR* in controlled noise experiments during training and testing. RPLAN is a large-scale dataset of real residential floor plans, each with 4 to 8 Manhattan rooms, spanning 65 to 120 square meters, but without real-world scale. After preprocessing [37] and aligning door edges with walls, we use 57,303 plans for training and 19,000 for testing. The *BADGR* model is trained for 20 diffusion steps with an embedding size $D = 512$, handling up to 100 walls and 15 cameras, using the training procedure from FloorPlan-60K. Since RPLAN lacks real indoor images, we use simulated camera poses and rendered floor boundaries for evaluation. Noisy layouts and poses are created by adding Gaussian noise (mean = 0, standard deviation = 3.3%) to the ground truth wall and camera positions. These inputs are normalized to a range of [-1, 1], and error distances (shown in Table 2) are reported in this normalized space.

Simulated Noise in Floor Boundary Inputs We added

Table 2. Pose and layout errors from predictions on RPLAN dataset. Since RPLAN floors are under 120 square meters, the reported distance error in percentage roughly translates to 1% to 0.1 meter in real scale.

Imgs / Rms	State / Method	Pose Err Dist (%) Mn	Med	Std	p90	Visible Layout Err Dist (%) Mn	Med	Std	p90	# walls # panos
1	Start	3.15	2.94	1.56	4.98	1.69	1.46	1.25	3.42	46.7 8.0
	BA-Only	0.58	0.41	0.31	0.93	0.62	0.30	0.73	1.25	
	BADGR	**0.35**	**0.29**	**0.18**	**0.55**	**0.33**	**0.15**	**0.38**	**0.58**	
2	Start	3.21	2.93	1.59	5.20	1.70	1.45	1.26	3.44	46.7 15.0
	BA-Only	0.62	0.31	0.38	1.09	0.58	0.23	0.73	1.11	
	BADGR	**0.34**	**0.23**	**0.17**	**0.54**	**0.27**	**0.13**	**0.33**	**0.47**	

Table 3. Errors from predictions on RPLAN dataset with simulated noise in floor boundaries. First column contains input density, the chance and max scale of noise added to each visible wall.

Imgs/Rms %noise	State/ Method	Pose Err Dist (%) Mn	Med	Std	p90	Visible Layout Err Dist (%) Mn	Med	Std	p90
1	Start	3.17	2.87	1.54	4.98	1.69	1.46	1.24	3.40
5% chance	*BA-Only*	1.47	0.60	1.19	2.70	2.22	0.52	2.81	2.53
2%scale	***BADGR***	**0.59**	**0.42**	**0.35**	**0.98**	**0.49**	**0.27**	**0.50**	**0.95**
1	Start	3.17	2.92	1.57	4.99	1.69	1.45	1.24	3.41
10% chance	*BA-Only*	1.63	0.79	1.25	2.93	1.95	0.62	3.07	2.67
2%scale	***BADGR***	**0.64**	**0.46**	**0.36**	**1.04**	**0.51**	**0.29**	**0.52**	**1.00**
2	Start	3.20	2.86	1.58	5.18	1.69	1.45	1.24	3.40
5% chance	*BA-Only*	1.48	0.58	1.19	2.49	1.24	0.45	2.69	2.23
2% scale	***BADGR***	**0.65**	**0.41**	**0.33**	**1.05**	**0.43**	**0.22**	**0.45**	**0.80**
2	Start	3.16	2.84	1.57	5.11	1.69	1.45	1.26	3.41
10% chance	*BA-Only*	1.57	0.74	1.22	2.74	2.16	0.55	0.60	2.51
2% scale	***BADGR***	**0.72**	**0.46**	**0.38**	**1.17**	**0.56**	**0.25**	**0.57**	**0.87**

noise to each rendered floor boundary to simulate bias from boundary prediction models, caused by factors like occlusion and limited training data. Before rendering the floor boundary for each simulated camera position, we randomly translate each visible wall along its normal direction, adjusting the opposite side to avoid self-intersection. The walls altered are selected by chance. Noise follows a uniform distribution with a max scale, and is applied independently for each rendering. The resulting floor boundaries guide the *BADGR* denoising process during testing.

With a maximum noise level of 2% (about 20 cm), both *BADGR* and *BA-Only* show increased pose and layout errors compared to no noise. However, *BADGR* is significantly less impacted by input perturbations than *BA-Only*. In terms of absolute distance error for poses and layouts, *BADGR* achieves lower errors compared to the ZInD test case when approximate scale is applied, likely due to the simpler structure of RPLAN floor plans.

8.3. Ablation Studies

We demonstrate the impact of *BADGR* in recovering poses and layouts by comparing to four models and training three *BADGR* variants in Table 4. Our conditional diffusion model *BADGR* outperforms guided diffusion model *BA+DM* in both tasks. *BADGR* trained with *BA* inputs only has a larger improvement compared with *BADGR* trained with reprojection loss only. We also found that although having higher errors from the first four models, the resulting layouts are mostly plausible looking, with improved layout and pose accuracy from the starting point.

Table 4. Ablation analysis on different variants of *BADGR*. All models are trained on FloorPlan-60k datasets and tested on ZInD test set. Diffusion model (*DM*) is *BADGR* trained without planar *BA* layer and reprojection loss. *BA+DM* is a guided diffusion model, where *BA* adjustment is added to the diffusion adjustment from *DM* above without *BA* conditioning.

BA Inputs	Reproj Loss	Imgs/Rms Method	1 Vis. walls (cm) Mn	Std	Pose (cm) Mn	Std	2 Vis. walls (cm) Mn	Std	Pose (cm) Mn	Std
✗	✗	*DM*	8.6	8.3	26.7	17.4	8.6	8.1	25.7	17.7
✗	✓	*BADGR*	8.4	8.4	23.6	14.1	8.3	8.1	24.4	14.3
✗	✗	*BA + DM*	8.6	9.0	15.0	10.0	8.0	8.3	13.4	8.7
✓	✗	*BADGR*	7.2	6.6	12.0	6.8	6.9	6.4	11.3	6.5
✓	✓	*BADGR*	**7.0**	**6.6**	**11.2**	**6.5**	**6.4**	**5.9**	**10.7**	**6.0**

8.4. Qualitative Results

As shown in Figure 4, BADGR improves layouts and view-consistency, even in extreme cases with minimal visual overlap, from a coarsely initialized scene. *BADGR* is able to learn the physical constraints from training data and correct issues like overlapping room layouts and varying wall thicknesses. While pose errors are not easily visualized in a top-down view, the image view reveals these inaccuracies and highlights the substantial improvements in both poses and layouts achieved by *BADGR*. Further evaluation of reprojection errors is provided in the Supplementary.

9. Conclusion

We present *BADGR*, a novel diffusion model that unites layout reconstruction with *BA*-style optimization, refining coarse poses and layouts, such as those derived from multiple 360° panoramas using methods like [21, 23, 47], into accurate, view-consistent floor plans. This is the first learning-based approach designed to handle full-scale indoor environments with up to 30 capture points, achieving enhanced spatial coherence through an integrated approach combining planar *BA* with diffusion-based structural constraints. Trained exclusively on schematic floor plans, *BADGR* adeptly addresses complex layouts and supports robust data augmentation techniques, including simulated camera poses. By leveraging a conditional diffusion model to guide nonlinear optimization, *BADGR* learns structural constraints and models spatial relationships and observation error, all through an efficient and effective training process.

Acknowledgments. We are grateful to Sing Bing Kang for discussions and paper edits on this work. In addition, we appreciate Zillow Group on providing the FloorPlan-60K dataset throughout this research.

References

[1] Hesse normal form. https://en.wikipedia.org/wiki/Hesse_normal_form. Accessed: 2025-03-18. 4

[2] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54 (10):105–112, 2011. 3

[3] Jonathan T Barron. A general and adaptive robust loss function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4331–4339, 2019. 4

[4] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-SP: Inverse CAD for floorplans by sequential room-wise shortest path. In *ICCV*, 2019. 2

[5] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-sp: Inverse cad for floorplans by sequential room-wise shortest path. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2661–2670, 2019.

[6] Jiacheng Chen, Yiming Qian, and Yasutaka Furukawa. Heat: Holistic edge attention transformer for structured reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3866–3875, 2022.

[7] Jiacheng Chen, Ruizhi Deng, and Yasutaka Furukawa. Polydiffuse: Polygonal shape reconstruction via guided set diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[8] Kefan Chen, Noah Snavely, and Ameesh Makadia. Wide-baseline relative camera pose estimation with directional learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3258–3268, 2021. 2, 6

[9] Yun Chen, Jingkang Wang, Ze Yang, Sivabalan Manivasagam, and Raquel Urtasun. G3r: Gradient guided generalizable reconstruction. In *European Conference on Computer Vision*, pages 305–323. Springer, 2025. 2

[10] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696, 2022. 2

[11] Hyungjin Chung, Jeongsol Kim, Sehui Kim, and Jong Chul Ye. Parallel diffusion models of operator and image for blind inverse problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6059–6069, 2023. 2

[12] Hyungjin Chung, Dohoon Ryu, Michael T McCann, Marc L Klasky, and Jong Chul Ye. Solving 3d inverse problems using pre-trained 2d diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22542–22551, 2023. 2

[13] Andrea Cohen, Torsten Sattler, and Marc Pollefeys. Merging the unmatchable: Stitching visually disconnected SfM models. In *ICCV*, 2015. 2

[14] Andrea Cohen, Johannes L. Schönberger, Pablo Speciale, Torsten Sattler, Jan-Michael Frahm, and Marc Pollefeys. Indoor-outdoor 3D reconstruction alignment. In *ECCV*, 2016. 2

[15] 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*, pages 2133–2143, 2021. 1, 2, 3, 6, 7

[16] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2

[17] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015. 1

[18] Henri P Gavin. The levenberg-marquardt algorithm for nonlinear least squares curve-fitting problems. *Department of Civil and Environmental Engineering Duke University August*, 3, 2019. 4

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 4, 6, 8

[20] Sepidehsadat Sepid Hossieni, Mohammad Amin Shabani, Saghar Irandoust, and Yasutaka Furukawa. Puzzlefusion: unleashing the power of diffusion models for spatial puzzle solving. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[21] Will Hutchcroft, Yuguang Li, Ivaylo Boyadzhiev, Zhiqiang Wan, Haiyan Wang, and Sing Bing Kang. Covispose: Co-visibility pose transformer for wide-baseline relative pose estimation in 360-degree indoor panoramas. In *European Conference on Computer Vision*, pages 615–633. Springer, 2022. 2, 3, 6, 8

[22] Zhigang Jiang, Zhongzheng Xiang, Jinhua Xu, and Ming Zhao. Lgt-net: Indoor panoramic room layout estimation with geometry-aware transformer network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1654–1663, 2022. 2, 3

[23] John Lambert, Yuguang Li, Ivaylo Boyadzhiev, Lambert Wixson, Manjunath Narayana, Will Hutchcroft, James Hays, Frank Dellaert, and Sing Bing Kang. Salve: Semantic alignment verification for floorplan reconstruction from sparse panoramas. In *European Conference on Computer Vision*, pages 647–664. Springer, 2022. 2, 6, 8

[24] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004. 6

[25] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 4

[26] Dominik Muhle, Lukas Koestler, Krishna Murthy Jatavallabhula, and Daniel Cremers. Learning correspondence uncertainty via differentiable nonlinear least squares. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13102–13112, 2023. 2

[27] Nelson Nauata, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa. House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In *ECCV*, 2020. 2

[28] Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa. House-gan++: Generative adversarial layout refinement network towards intelligent computational agent for professional architects. In *CVPR*, 2021. 2

[29] Negar Nejatishahidin, Will Hutchcroft, Manjunath Narayana, Ivaylo Boyadzhiev, Yuguang Li, Naji Khosravan, Jana Košecká, and Sing Bing Kang. Graph-covis: Gnn-based multi-view panorama global pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6459–6468, 2023. 2, 3, 6, 7

[30] Rémi Pautrat, Iago Suárez, Yifan Yu, Marc Pollefeys, and Viktor Larsson. Gluestick: Robust image matching by sticking points and lines together. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9706–9716, 2023. 2

[31] Luis Pineda, Taosha Fan, Maurizio Monge, Shobha Venkataraman, Paloma Sodhi, Ricky TQ Chen, Joseph Ortiz, Daniel DeTone, Austin Wang, Stuart Anderson, et al. Theseus: A library for differentiable nonlinear optimization. *Advances in Neural Information Processing Systems*, 35:3801–3818, 2022. 2, 5

[32] Giovanni Pintore, Marco Agus, and Enrico Gobbetti. Atlantanet: inferring the 3d indoor layout from a single 360-degree image beyond the manhattan world assumption. In *European Conference on Computer Vision*, pages 432–448. Springer, 2020. 2, 3

[33] Alec Radford. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 5

[34] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 2

[35] Grant Schindler and Frank Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, pages I–I. IEEE, 2004. 1, 3

[36] 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

[37] Mohammad Amin Shabani, Sepidehsadat Hosseini, and Yasutaka Furukawa. Housediffusion: Vector floorplan generation via a diffusion model with discrete and continuous denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5466–5475, 2023. 2, 4, 5, 7

[38] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2, 6

[39] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 6, 1

[40] 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*, pages 16034–16043, 2021. 2

[41] Jheng-Wei Su, Chi-Han Peng, Peter Wonka, and Hung-Kuo Chu. Gpr-net: Multi-view layout estimation via a geometry-aware panorama registration network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6469–6478, 2023. 2

[42] Cheng Sun, Chi-Wei Hsiao, Min Sun, and Hwann-Tzong Chen. Horizonnet: Learning room layout with 1d representation and pano stretch data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1047–1056, 2019. 2, 3, 4

[43] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Hohonet: 360 indoor holistic understanding with latent horizontal features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2573–2582, 2021. 2

[44] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 6

[45] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020. 5

[46] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*, 2018. 2

[47] Diantao Tu, Hainan Cui, Xianwei Zheng, and Shuhan Shen. Panopose: Self-supervised relative pose estimation for panoramic images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20009–20018, 2024. 2, 6, 8, 3

[48] Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu, et al. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 12, 2016. 5

[49] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 5

[50] Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai. Led2-net: Monocular 360deg layout estimation via differentiable depth rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12956–12965, 2021. 2, 3, 4

[51] Haiyan Wang, Will Hutchcroft, Yuguang Li, Zhiqiang Wan, Ivaylo Boyadzhiev, Yingli Tian, and Sing Bing Kang. Psmnet: Position-aware stereo merging network for room layout estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8616–8625, 2022. 2

[52] Jianyuan Wang, Christian Rupprecht, and David Novotny. Posediffusion: Solving pose estimation via diffusion-aided

bundle adjustment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9773–9783, 2023. 2

[53] 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 (TOG)*, 38(6):1–12, 2019. 6, 7

[54] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16010–16021, 2023. 2

[55] Yuanwen Yue, Theodora Kontogianni, Konrad Schindler, and Francis Engelmann. Connecting the dots: Floorplan reconstruction using two-level queries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 845–854, 2023. 1, 2

[56] Yinda Zhang, Shuran Song, Ping Tan, and Jianxiong Xiao. Panocontext: A whole-room 3d context model for panoramic scene understanding. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 668–686. Springer, 2014. 2, 3, 6

[57] Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Truncated diffusion probabilistic models and diffusion-based adversarial auto-encoders. *arXiv preprint arXiv:2202.09671*, 2022. 4

[58] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 519–535. Springer, 2020. 2

[59] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2051–2059, 2018. 2

BADGR: Bundle Adjustment Diffusion Conditioned by GRadients for Wide-Baseline Floor Plan Reconstruction

Supplementary Material

1. Implementation Details with FloorPlan-60K Data

BADGR is trained using a 2D 'cleanup' layer of floor plans, where larger spaces are represented by unions of multiple partially annotated room shapes, following the annotation approach of ZInD [15]. Panorama poses are randomly sampled within each room. For each input image, *BADGR* simulates data with a CUDA-based 1D renderer, given floor plan layouts and a sampled camera pose. The renderer operates on connected rooms through doors, omitting door polygons and matching wall segments along the front and back planes. Random masking is applied on $\{\hat{\mathcal{B}}^i\}$ and $\mathcal{M}$ to occasionally bypass the BA layer for selected image columns. During diffusion training, scenes are rotated by $[0°, 90°, 180°, 270°]$. For evaluation, we use the ZInD test set (275 floor plans), with initial scenes, floor boundary depths, and column-to-wall assignments estimated from real panorama images, as detailed in Section 8.1 of the main paper.

BADGR has a capacity of 300 walls and 30 panoramas, which is selected to accommodate 99% of the floor plans from FloorPlan-60K data. It is trained with a batch size of 48, and with a learning rate of 10^{-4} for 140 epochs, 10^{-5} for 50 epochs, and 10^{-6} for 50 epochs by stepwise decay. *BADGR* is trained for the last 20 steps of a 1000-step diffusion process, using a second-moment schedule sampler for time t. *Ordinary Differential Equations* (*ODE*) sampling [39] is used during the *BADGR* inference process. Training peaks at 55GB GPU memory usage on a single GPU. During inference, *BADGR* processes a batch size of 1 in approximately 25 seconds on a CPU-only Apple M1 MacBook Pro with 32GB of memory, and around 4.0 secs on an A100 GPU.

2. Cross dataset training and validation

We additionally trained a *BADGR* model of a max capacity of 300 walls and 30 cameras with the RPLAN training set, and with a similar settings of sampling camera positions for generating simulated floor boundaries and column-to-wall assignments. This model is evaluated on the ZInD test set. The results are listed in Table 5 alongside with existing results for comparison. Although *BADGR* trained with RPLAN dataset doesn't produce similar or higher accuracy than *BADGR* trained with FloorPlan-60K dataset, it still outperform the *CovisPose+* and *BA-Only* baselines. This trend is expected as RPLAN contains Manhattan floors only and overall have less rooms and panoramas during training.

Table 5. Pose and layout errors tested on the ZInD dataset, trained with various datasets. Row 3 of each block presents additional results compared to the main paper. Mn, Med, and Std denote mean, median, and standard deviation, respectively. We also report the 90th percentile (p90) of the absolute translation errors for the estimated camera poses.

Imgs/ Rm	Methods	Training Set	Camera Translation(cm)				Visible walls (cm)			
			Mn	Med	Std	p90	Mn	Med	Std	p90
0.6	*CovisPose+*	ZInD	20.7	15.7	12.0	33.6	11.5	6.9	10.5	26.0
	BA Only	N/A	19.1	12.2	10.9	32.7	12.8	6.8	11.9	29.3
	BADGR	RPLAN	14.7	11.4	8.0	24.0	9.4	5.8	8.6	20.6
	BADGR	FloorPlan-60K	**12.2**	**9.5**	**7.2**	**18.5**	**7.1**	**4.5**	**6.7**	**15.3**
1	*CovisPose+*	ZInD	19.7	15.6	11.8	33.7	12.3	7.0	10.8	27.4
	BA Only	N/A	17.6	12.6	12.0	34.2	12.6	6.7	11.4	27.5
	BADGR	RPLAN	15.1	11.1	8.5	26.5	9.2	5.9	8.8	20.6
	BADGR	FloorPlan-60K	**11.2**	**8.8**	**6.5**	**18.6**	**7.0**	**4.6**	**6.6**	**15.8**
2	*CovisPose+*	ZInD	17.9	14.5	10.2	30.8	12.0	7.0	9.0	26.0
	BA Only	N/A	14.3	10.8	9.2	30.1	10.5	6.2	9.9	23.3
	BADGR	RPLAN	14.1	10.5	8.6	27.8	9.1	5.6	8.7	20.1
	BADGR	FloorPlan-60K	**10.7**	**8.9**	**6.0**	**17.1**	**6.4**	**4.4**	**5.9**	**13.9**
3	*CovisPose+*	ZInD	18.1	14.8	10.5	31.6	12.3	7.2	10.2	26.9
	BA Only	N/A	13.4	10.7	8.2	30.5	10.6	6.2	9.0	22.1
	BADGR	RPLAN	13.3	10.2	7.7	22.9	9.4	5.9	8.7	21.1
	BADGR	FloorPlan-60K	**10.6**	**8.9**	**6.0**	**17.5**	**6.6**	**4.3**	**6.0**	**14.6**

3. Reprojection Errors

Reprojection errors are reported in Table 6 to measure the view-consistency between the floor boundary projected from the predicted layout and poses and the per-image estimations, similar to the blue and green lines in the bottom-right images of Figure 1 of the main paper. The stats are computed from the per-column reprojection errors across all wall-assigned image columns, which is defined in Algorithm 1 of the main paper.

Table 6. Reprojection errors (*L1* distance by pixel relative to an image size of 256 × 512) for wall-assigned columns, which measures view-consistency compared to the predicted floor boundary. Alongside Table 1 of the main paper, we observe that while *BADGR* sometimes produces higher re-projection errors than *BA-Only*, it consistently achieves lower layout and pose errors. This suggests that reprojection error influences accuracy but is not the sole factor in achieving high reconstruction accuracy. The stats are collected similarly to those from Table 1 of the main paper.

Img/Rm	0.6				1				2				3			
Method	Mn	Med	Std	p90	Mn	Med	Std	p90	Mn	Med	Std	p90	Mn	Med	Std	p90
CovisPose+	1.38	0.92	1.81	2.77	1.52	0.95	2.08	3.05	1.69	1.03	2.40	3.67	1.79	1.05	2.57	3.93
BA-Only	0.70	**0.29**	1.21	1.64	0.78	**0.32**	1.46	1.88	0.81	0.39	1.48	2.07	0.89	0.39	1.45	2.06
BADGR	**0.65**	0.31	1.17	**1.36**	**0.75**	**0.32**	1.34	1.77	**0.78**	**0.35**	1.37	1.81	0.81	**0.36**	1.33	2.04
GT Scene + Predicted Boundary	0.91	0.82	**0.56**	2.77	0.90	0.80	**0.56**	**1.55**	0.89	0.78	**0.56**	**1.56**	**0.89**	0.77	**0.57**	**1.56**

As Table 6 shows, overall reprojection error increases with the number of input images. This is caused by the accumulating pose errors and inconsistencies in floor boundary estimates across overlapping regions. Both *BA-Only* and *BADGR* consistently show lower reprojection errors compared to *CovisPose+*. In most cases, *BADGR* reports slightly lower reprojection errors than *BA-Only*, likely because BA-Only can get stuck in local minima of the loss function and uses a PyTorch implementation that also considers memory and speed. In this implementation, adjustments are computed at per-column level and averaged to update poses and walls, rather than optimizing the total reprojection error across all columns.

4. Coarse Scene Initialization for Inference

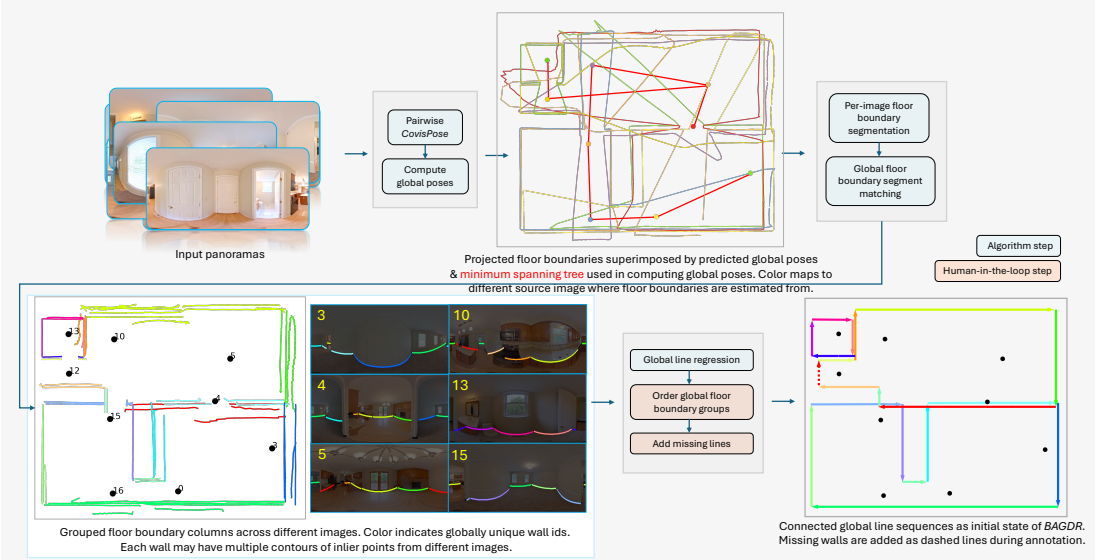

Figure 5. Overview of coarse scene initialization.

Initial Poses From input panoramas $\{P^i\}$, a modified *CovisPose* model [21] is executed exhaustively on each pair of panoramas from the same floor. This model has the same exact architecture as the original *CovisPose* model predicting: 1) relative camera pose $\tilde{E}^{(i,j)} \in SE(2)$, 2) floor boundaries $\{\tilde{\mathcal{B}}^i\}$, 3) cross-view co-visibility, angular correspondences $\{\tilde{\alpha}^{i,j}\}$, $\{\tilde{\varphi}^{i,j}\}$. It additionally predicts binary classification of room corners $\{\tilde{\mathcal{V}}^i\}$ for each column. The model is trained on the *ZInD* dataset [15] with the same image pairs as [21] and an additional corner loss function similar to that of [42]. Pose pairs of co-visibility score greater than 0.1 are selected to create a minimal spanning tree of the pose graph using a greedy algorithm, similar to [29]. Prior to computing global poses $\tilde{E}^i$, $\tilde{E}^{i,j}$ are corrected through axis alignment with a 45° interval using predicted vanishing angles [56].

Initial Walls The per-panorama floor boundary $\{\hat{\mathcal{B}}^i\}$ is segmented with room corners $\{\hat{\mathcal{V}}^i\}$. Inlier boundary points are then extracted with *RANSAC*, and initial wall parameters $(\overrightarrow{v^i_{m,k}}, b^i_{m,k})$ are computed for each local wall detected from panoramas P^i. Voting-based heuristics are used to match inlier boundary points, which maps to per-panorama local line segments, between panorama pairs using $\{\tilde{\alpha}^{i,j}\}$, $\{\tilde{\varphi}^{i,j}\}$ and $(\overrightarrow{v^i_{m,k}}, b^i_{m,k})$. Pairwise local line matches are aggregated into a unique global wall identity for wall $l_{m,k}$ shared across P^i. The estimated wall parameters, i.e. $(\overrightarrow{v_{m,k}}, \hat{b}_{m,k})$, are computed with linear regression, where $\overrightarrow{v_{m,k}}$ is selectively axis-aligned with a $45°$ interval. Only wall angles closer to $10°$ to the vanishing directions, e.g. 0, 45, 90, 135, are corrected. Finally, an annotator uses a graphics interface to: 1) provide global wall connectivity (shown as arrows in the bottom right image of Figure 5), and 2) add missing room corners with their rough initial positions with guidance from the images and topdown projected floor boundaries (dotted lines in the bottom right image of Figure 5). The number of room corners and wall orientations are static input to *BADGR*.

During testing, a subset of panoramas are selected as described in Section 8.1 of the main paper and Section 1 of the Supplement. To generate the coarse initial layouts, we use the connectivity of the annotated global scene as discussed above, re-compute parameter $(\overrightarrow{v^i_{m,k}}, b^i_{m,k})$ of visible walls using the inlier boundary point from the selected panoramas, and inherit the parameters of invisible walls from the initial coarse scene generated with all available panoramas from the ZInD dataset. Only rooms with visible walls are included in the coarse initial layouts. *PolyDiffuse* [7] also uses simple annotation during initialization. Our paper focuses on the difficult step of global refinement. Automating this annotation is future work to automate an end-to-end pipeline.

5. Discussion

PuzzleFusion (*PF*) [20] and ***Extreme SfM*** (*E-SfM*) [36] also produce floor plan layouts and camera poses. Here we provide a discussion on their differences to *BADGR*. Both *PF* and *E-SfM* estimate the rotation and translation of given unposed non-deformable room layouts by solving jigsaw puzzles. Camera poses are then inferred from the puzzle solution. This has different objectives than ours: 1) within the same room their relative positions among individual walls and multiple camera poses stay unchanged; 2) neither method uses information from a set of horizontal-facing images without precise poses as input constraints to guide optimization for view-consistency. Both can be used for initialization of *BADGR* like *CovisPose*. Code and weights of *PF* trained on RPLAN aren't publicly available. We contacted the authors, and the code no longer runs. *E-SfM* takes hours or even days to process a single house [20], so neither can be used as baselines. *BADGR* solves a different task as we are deforming room shapes. Instead, we simulate *BADGR* refining *PF*-initialized layouts by adding Gaussian noise (10.55 *Mean Positional Error in pixels* (MPE) matches *PF*) to room translations, with relative poses among cameras and walls within each room given for initialization. *BADGR* reduces the MPE↓ of room placement from 10.55 or 4.1% (normalized by 256×256 pixel resolution) (full RPLAN) to 0.93% (77.3% lower), calculated by average shift of each vertex. We also report 0.98%, 1.45% mean translation errors in layout and poses. MPE of *E-SfM* is only reported for small RPLAN as 29.44 or 11.5% [20].

GraphCovis [29] and ***PanoPose*** [47] didn't publicly release their code. *GraphCovis* estimates global poses among up-to-5 input panorama images. We compared the pose errors of *BADGR* with *GraphCovis* under the similar input settings originally evaluated on ZInD [15] in table 7. It demonstrates *BADGR*'s robust performance among different sizes of homes and missing room scenarios.

Table 7. Statistics of *absolute translation error* and *absolute rotation error* on group of three, four, and five panoramas for *GraphCovis* (*GC*) and *BADGR*. The accuracy of *GraphCovis* is imported from Table 1 of [29].

Group-Size	*GC* Rot ↓ (°)			*BADGR* Rot ↓ (°)			*GC* Transl. ↓ (cm)			*BADGR* Transl. ↓ (cm)		
# imgs	Mn	Med	Std	Mn	Med	Std	Mn	Med	Std	Dist (cm)	Med	Std
3	2.00	0.85	9.15	**0.25**	**0.20**	**0.28**	**8.1**	**3.8**	29.2	9.2	6.1	**5.9**
4	3.19	0.94	13.36	**0.26**	**0.22**	**0.30**	15.3	**6.1**	43.0	**10.6**	**6.1**	**6.0**
5	3.29	1.07	12.04	**0.25**	**0.19**	**0.30**	17.2	8.2	38.4	**10.4**	**6.3**	**6.7**

6. Failure Cases

We present three failure cases to highlight the challenges and opportunities for *BADGR*. Overall, *BADGR* achieves high accuracy when input images are minimally-connected by covisible walls through column-to-wall assignments from the initial coarse scene. However, since *BADGR* is trained on simulated panorama poses and column-to-wall assignments, the model can struggle when faced with scenarios outside the training distribution. An example is shown in Figure 6, where the initial scene contains large errors over wide areas, and the column-to-wall assignments fail to establish critical covisible walls between panoramas. This underscores the need for future development of an end-to-end initialization method to establish global column-to-wall assignments.

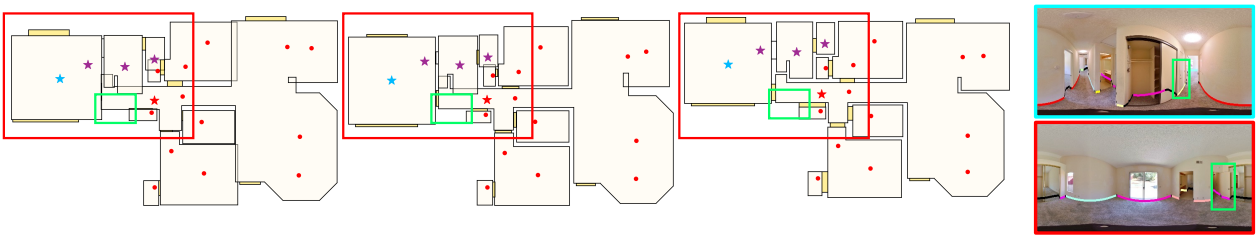

Figure 6. Failure case example caused by errors in coarse scene initialization. The colored lines in the images on the right represent estimated floor boundaries, with colors indicating their assigned unique global wall id. The heuristic failed to match two wall segments (highlighted in rectangles) using dense column correspondences and floor boundaries from the CovisPose model. Additionally, the large initial error in the highlighted section (highlighted in rectangles) may fall near the boundary of the 20-step truncated diffusion data distribution, contributing to the issue.

BADGR relies on floor boundaries for positional information along the normal direction of the target surface. This explains the failure case in Figure 7, where the wall length is estimated incorrectly due to a lack of guidance to the model. Future work could incorporate cues from wall junctions, similar to [42, 50], or encode pixel positions of pre-assigned columns to better constrain visible walls and infer invisible wall positions.

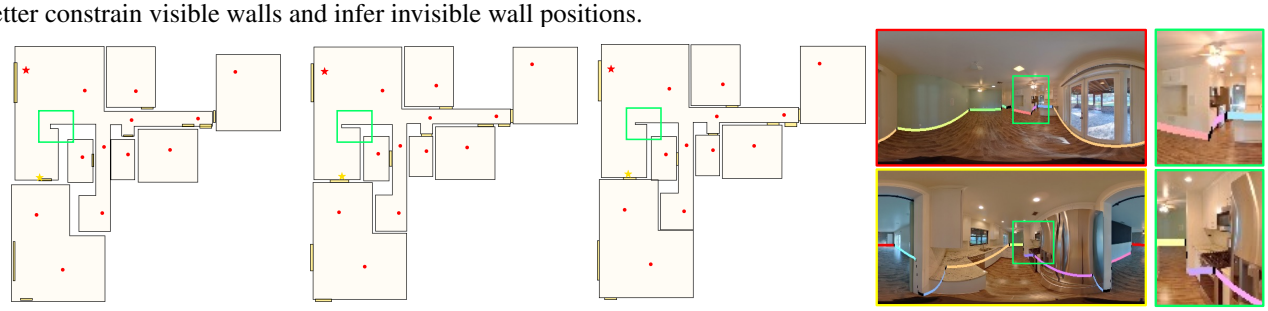

Figure 7. Failure case example where the highlighted wall is predicted with an incorrect length due to the limited image column coverage. The colored lines in image views represent similarly as in Figure 6.

BADGR assumes consistent floor heights throughout the area. When this assumption is violated, such as with a sunken floor (Figure 8), planar *BA* may place walls farther than their actual positions. Future work may include extending *BADGR* to represent varying camera height and wall heights, and expanding the training data to cover this issue.

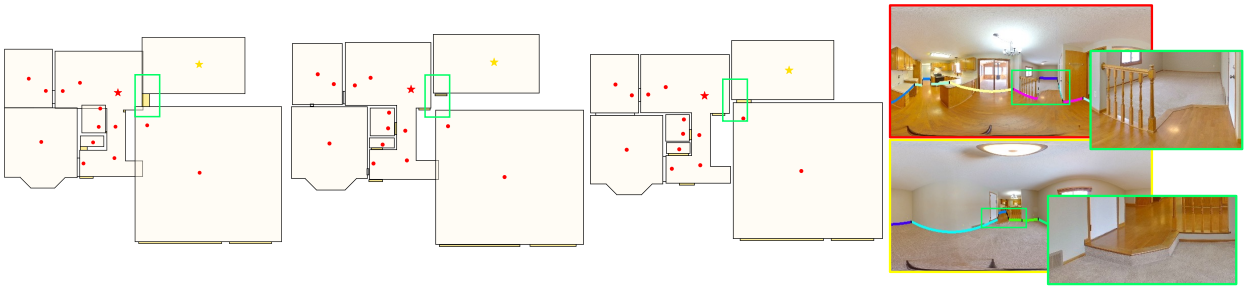

Figure 8. Failure case example due to inconsistent floor heights. The colored lines in image views represent similarly as in Figure 6.

7. More Qualitative Results

The reprojected wall lines in the image are drawn with a thickness equal to 1.1% of the image height. This thickness can cover significant floor distances, especially when walls are near the image center. For example, at a pitch angle of 30° (floor distance of 0.58 camera heights), the line covers 4.5% of the camera height; at 45° (1 camera height), it covers 7.4%; at 60° (1.73 camera height), 14.7%; and at 75° (3.73 camera height), 55.2%. While the blue and green lines represented in small images sometimes appear to overlap, particularly for walls farther from the camera, *BADGR* processes continuous inputs and outputs for coordinates, enabling higher precision. See quantitative results for more precise details.

Figure 9. More qualitative results trained on FloorPlan-60K dataset and tested on ZInD dataset (page 1), with input densities at a maximum of 2 input images from each input partial room. The topdown views from left to right are before, after *BADGR* optimization and the ground truth.

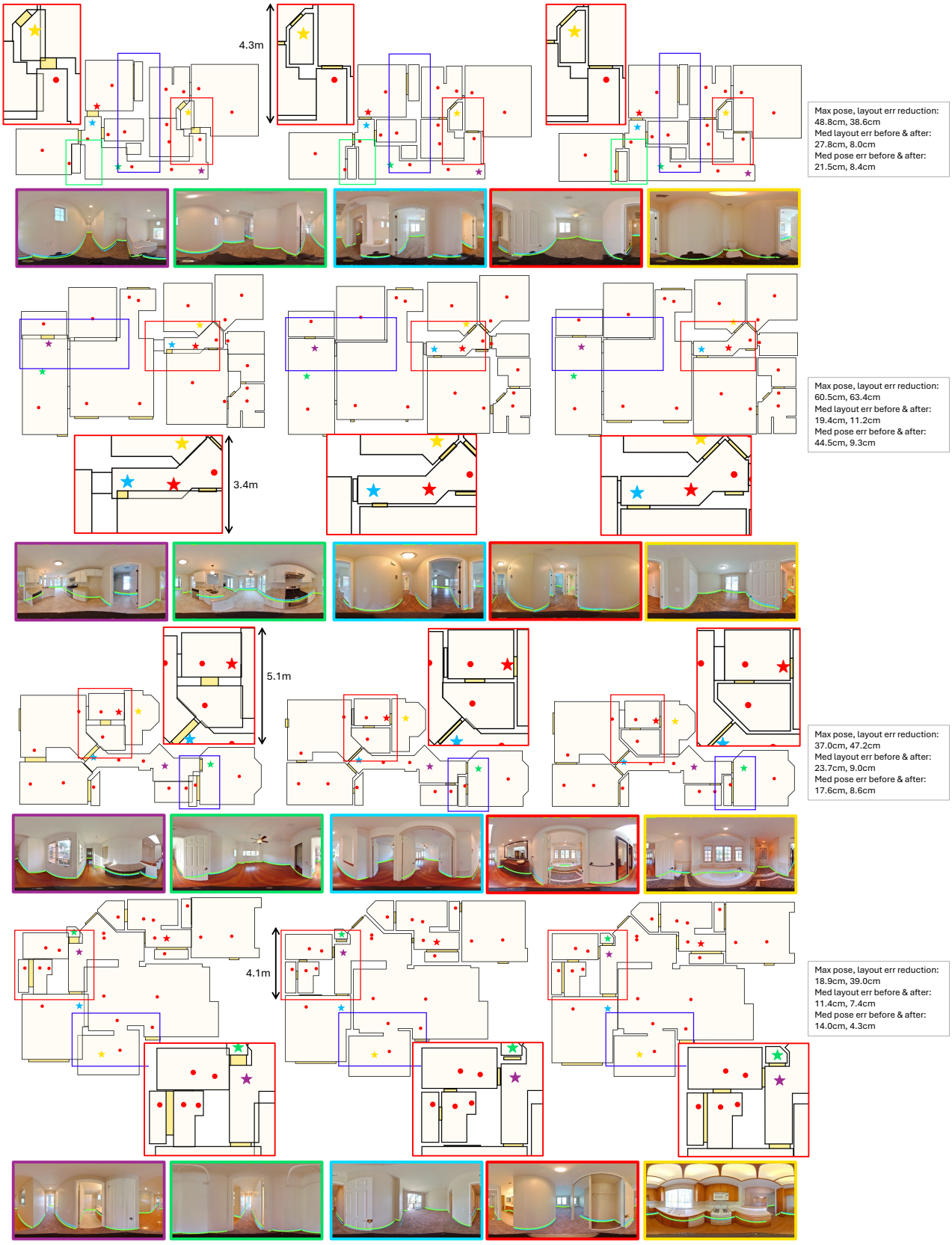

Figure 10. More qualitative results trained on FloorPlan-60K dataset and tested on ZInD dataset (page 2), with input densities at a maximum of 2 input images from each partial room. The topdown views from left to right are before, after *BADGR* optimization and the ground truth.

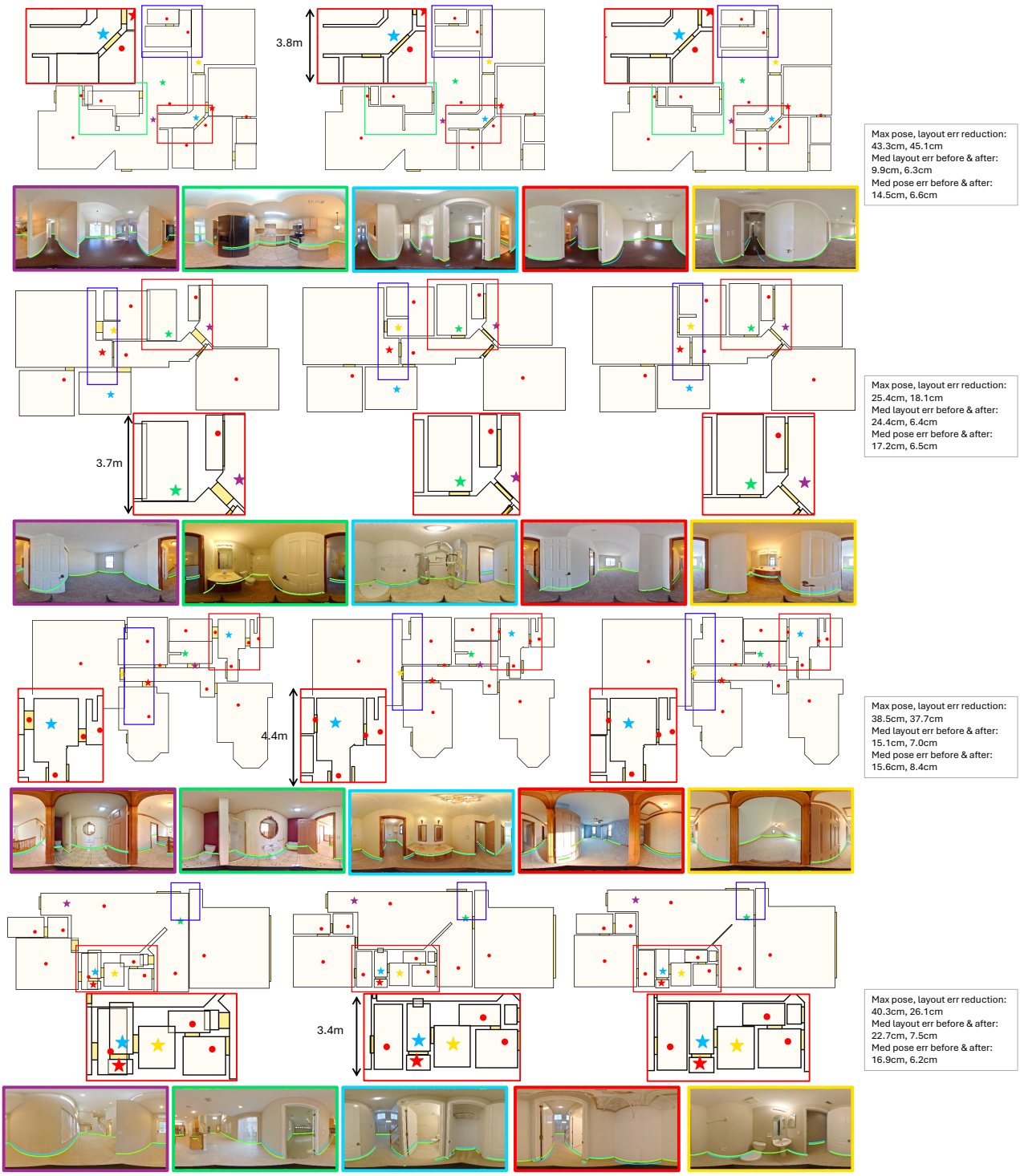

Figure 11. More qualitative results trained on FloorPlan-60K dataset and tested on ZInD dataset (page 3), with input densities at a maximum of 1 input images from each partial room. The topdown views from left to right are before, after *BADGR* optimization and the ground truth.

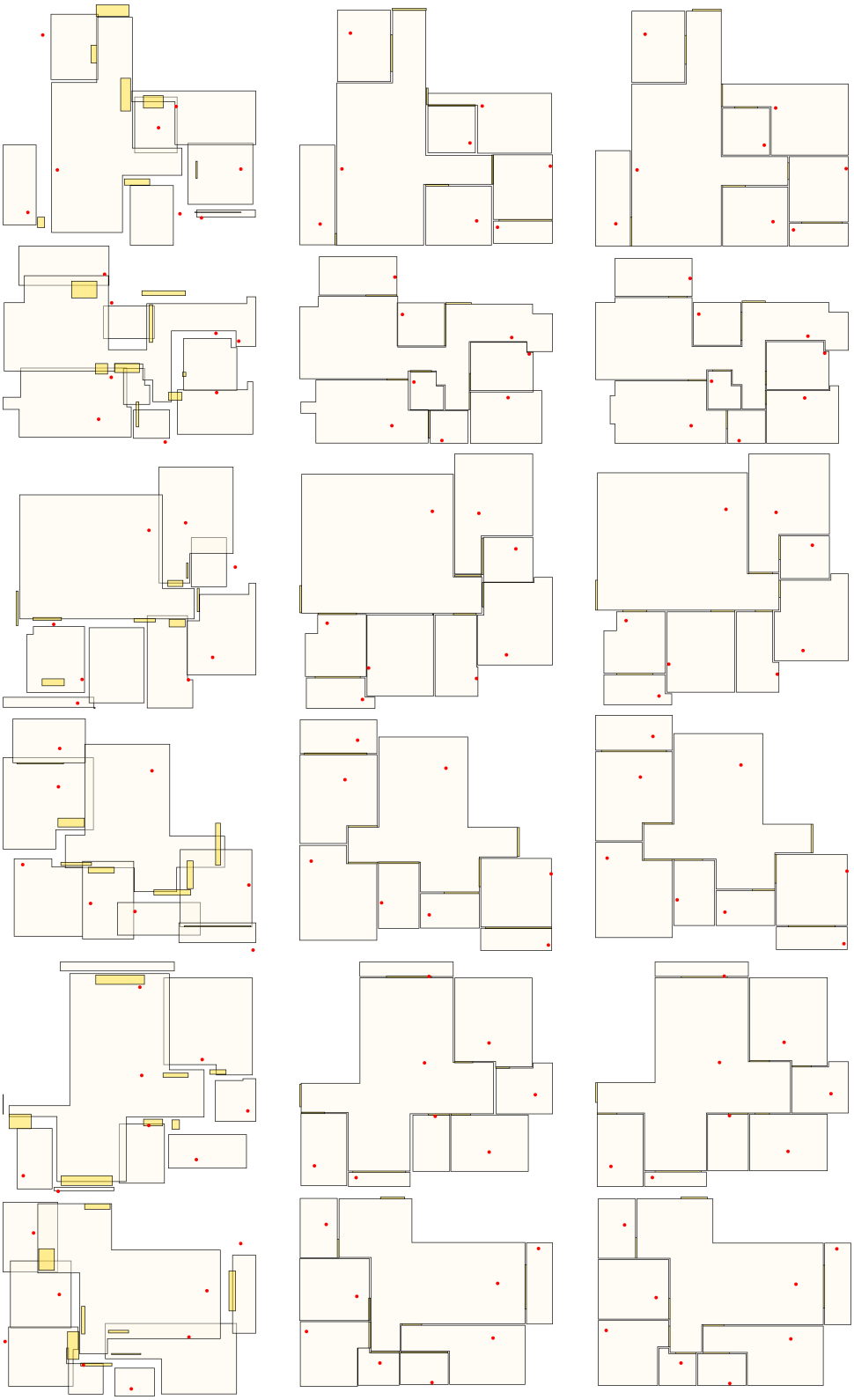

Figure 12. Qualitative results trained and tested on RPLAN dataset (page 4), with input densities at one input image from each partial room. The topdown views from left to right are before, after *BADGR* optimization and the ground truth. The initial state is created by adding Gaussian noise from 20-step diffusion q-sampling [19] into the ground truth poses and layouts. Details see Section 8.2 of the main paper. This figure demonstrates *BADGR*'s capability to refine initial scenes with much higher noise than those from the ZInD test cases.