BADGR: Bundle Adjustment Diffusion Guided by Gradient for Extreme-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 [2]. 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 [8] 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 1 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 1. 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 2 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 2. 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 2 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

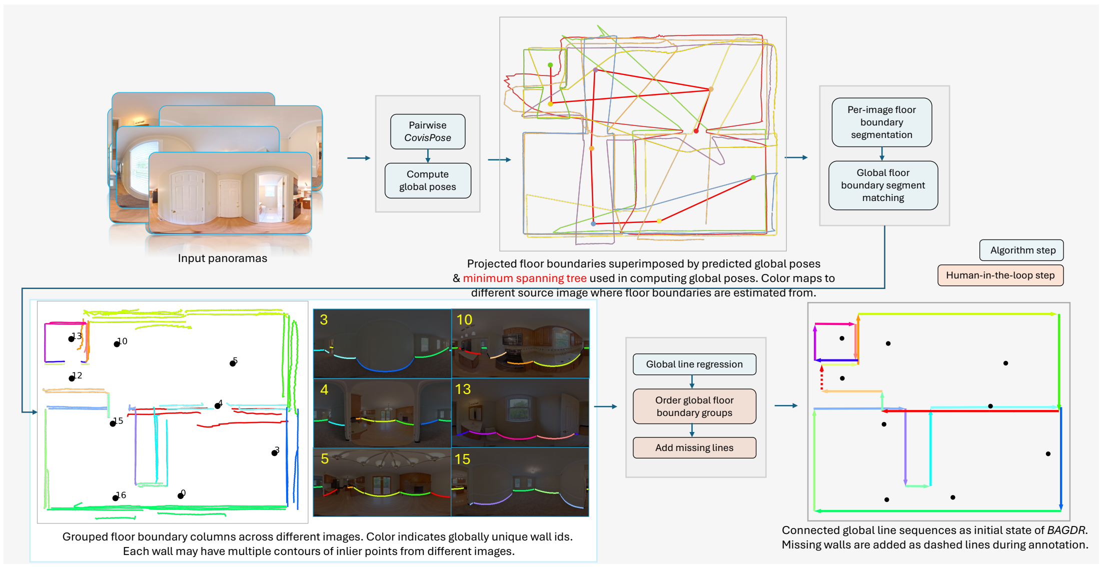

Input panoramas
Pairwise CovisPose
Compute global poses
Projected floor boundaries superimposed by predicted global poses & minimum spanning tree used in computing global poses. Color maps to different source image where floor boundaries are estimated from.
Per-image floor boundary segmentation
Global floor boundary segment matching
Algorithm step
Human-in-the-loop step
13
10
5
12
4
15
3
16
0
3
10
4
13
5
15
Grouped floor boundary columns across different images. Color indicates globally unique wall ids. Each wall may have multiple contours of inlier points from different images.
Global line regression
Order global floor boundary groups
Add missing lines
Connected global line sequences as initial state of BAGDR. Missing walls are added as dashed lines during annotation.

Figure 1. Overview of coarse scene initialization.

Initial Poses From input panoramas $\{P^i\}$, a modified *CovisPose* model [5] 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 [2] with the same image pairs as [5] and an additional corner loss function similar to that of [9]. 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 [6]. 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 [12].

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 1), 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 1). 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* [1] 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*) [4] and ***Extreme SfM*** (*E-SfM*) [7] 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 [4], 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% [4].

GraphCovis [6] and ***PanoPose*** [10] 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 [2] in table 3. It demonstrates *BADGR*'s robust performance among different sizes of homes and missing room scenarios.

Table 3. 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 [6].

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 2, 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.

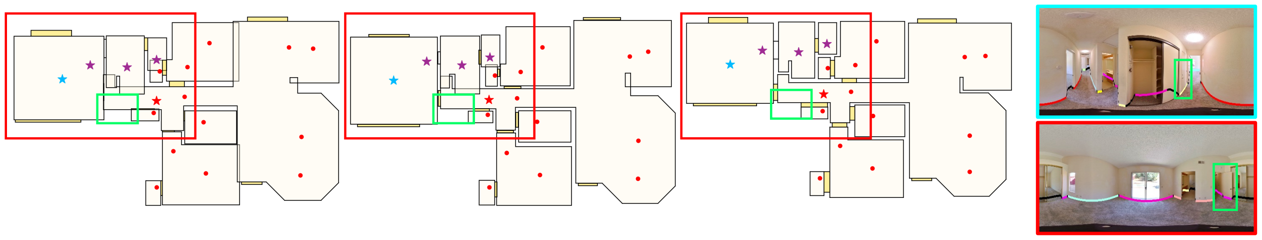


Figure 2. Failure case example caused by errors in coarse scene initialization. The colored lines in the images on the right represent estimated floor boundaries, with colors indicating their assigned unique global wall id. The heuristic failed to match two wall segments (highlighted in rectangles) using 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 3, 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 [9, 11], or encode pixel positions of pre-assigned columns to better constrain visible walls and infer invisible wall positions.

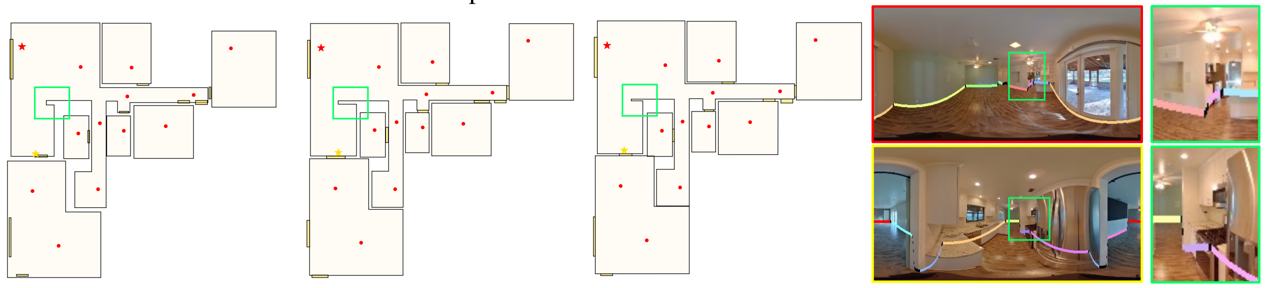


Figure 3. Failure case example where the highlighted wall is predicted with an incorrect length due to the limited image column coverage. The colored lines in image views represent similarly as in Figure 2.

BADGR assumes consistent floor heights throughout the area. When this assumption is violated, such as with a sunken floor (Figure 4), 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.

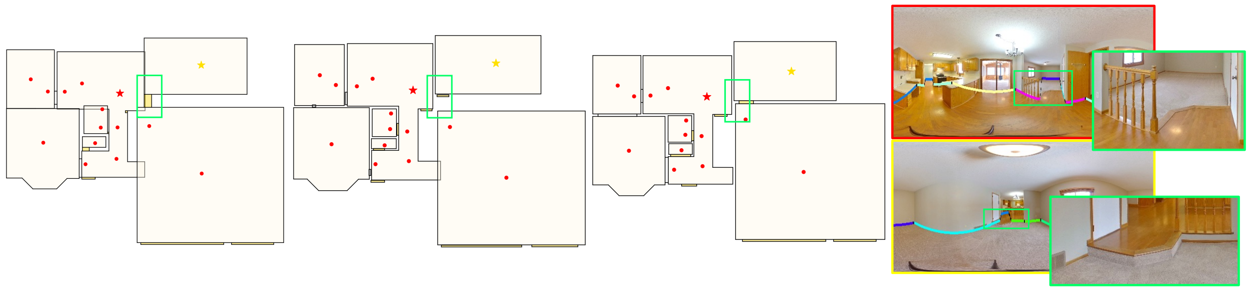


Figure 4. Failure case example due to inconsistent floor heights. The colored lines in image views represent similarly as in Figure 2.

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.

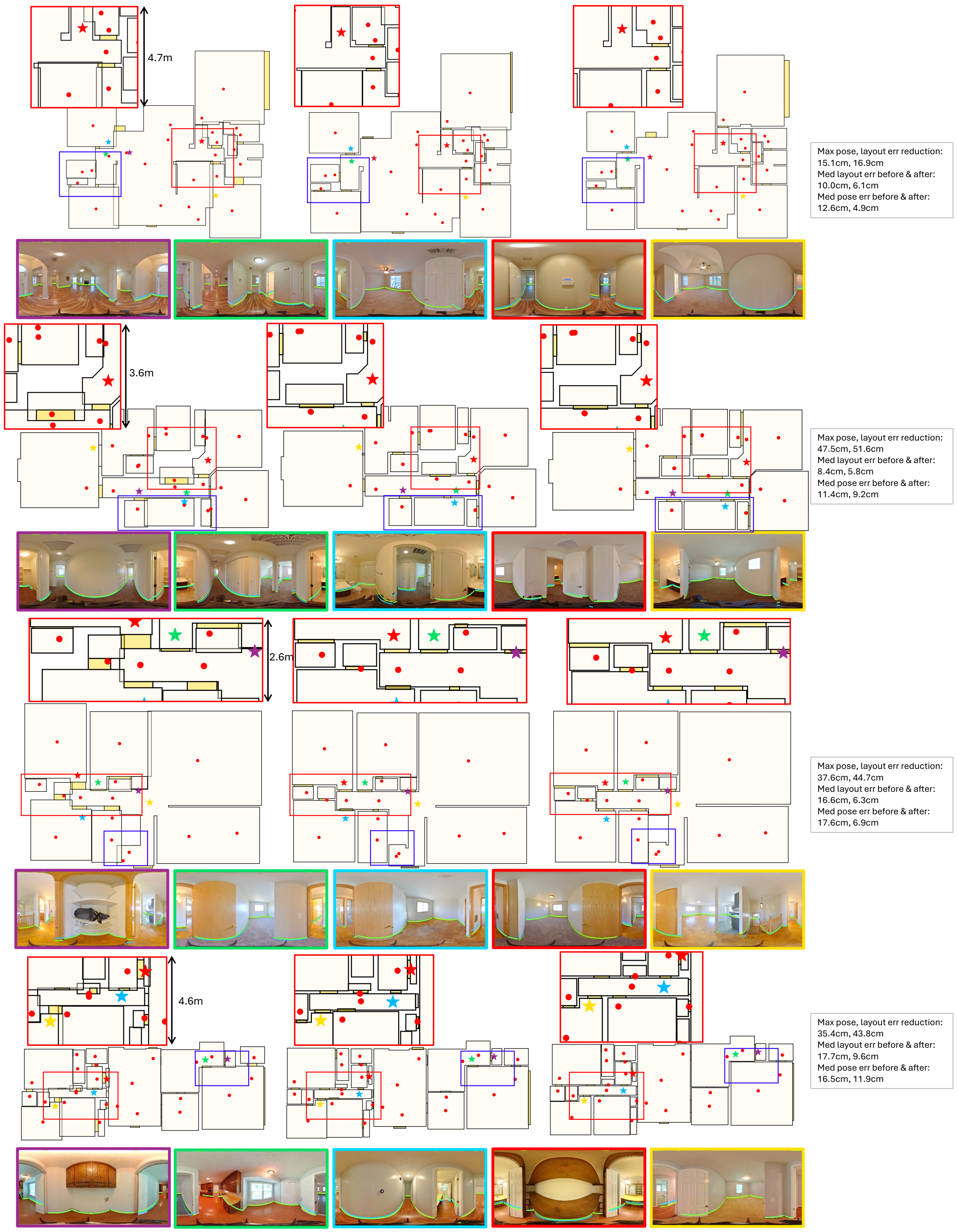

4.7m
Max pose, layout err reduction: 15.1cm, 16.9cm
Med layout err before & after: 10.0cm, 6.1cm
Med pose err before & after: 12.6cm, 4.9cm
3.6m
Max pose, layout err reduction: 47.5cm, 51.6cm
Med layout err before & after: 8.4cm, 5.8cm
Med pose err before & after: 11.4cm, 9.2cm
2.6m
Max pose, layout err reduction: 37.6cm, 44.7cm
Med layout err before & after: 16.6cm, 6.3cm
Med pose err before & after: 17.6cm, 6.9cm
4.6m
Max pose, layout err reduction: 35.4cm, 43.8cm
Med layout err before & after: 17.7cm, 9.6cm
Med pose err before & after: 16.5cm, 11.9cm

Figure 5. 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.

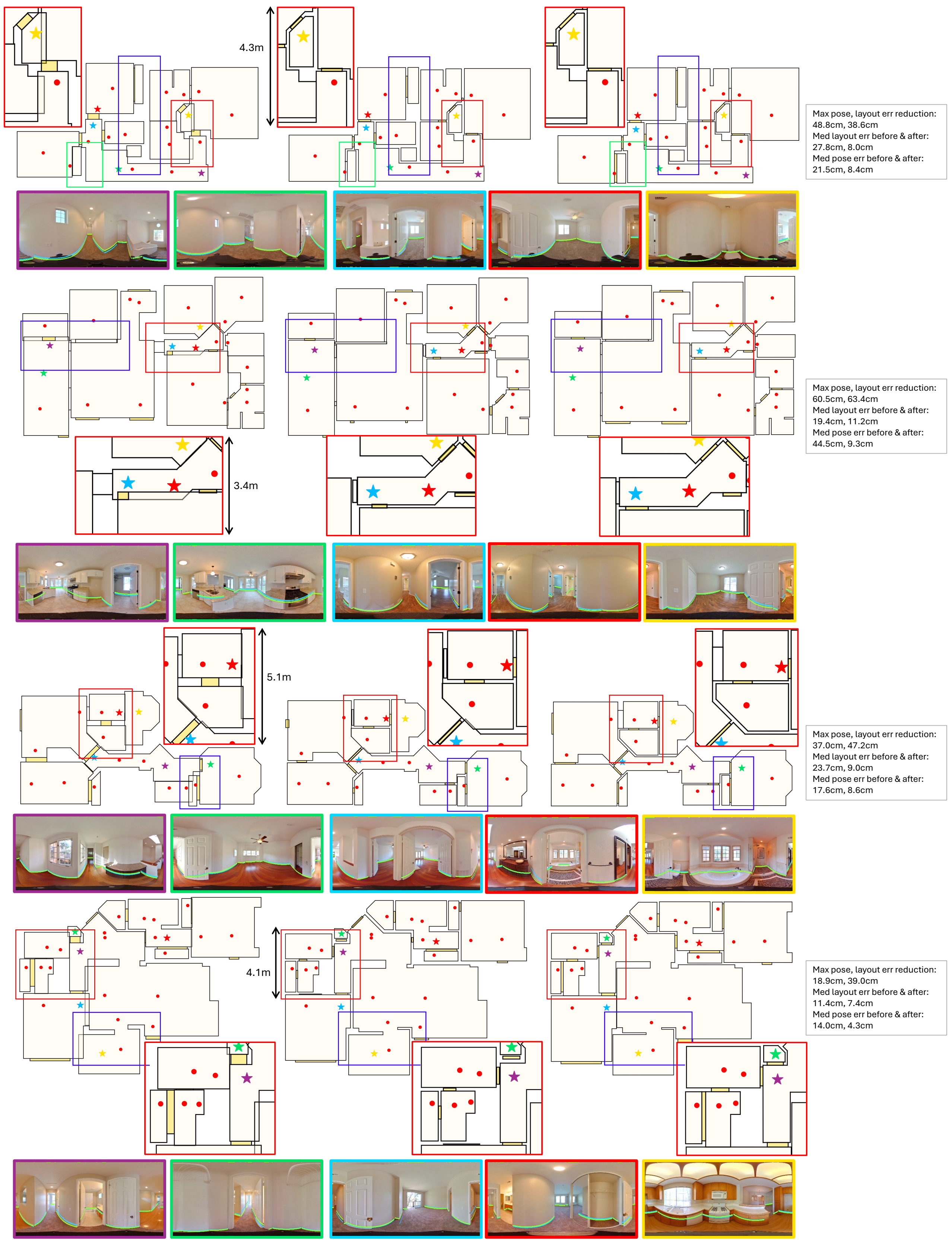
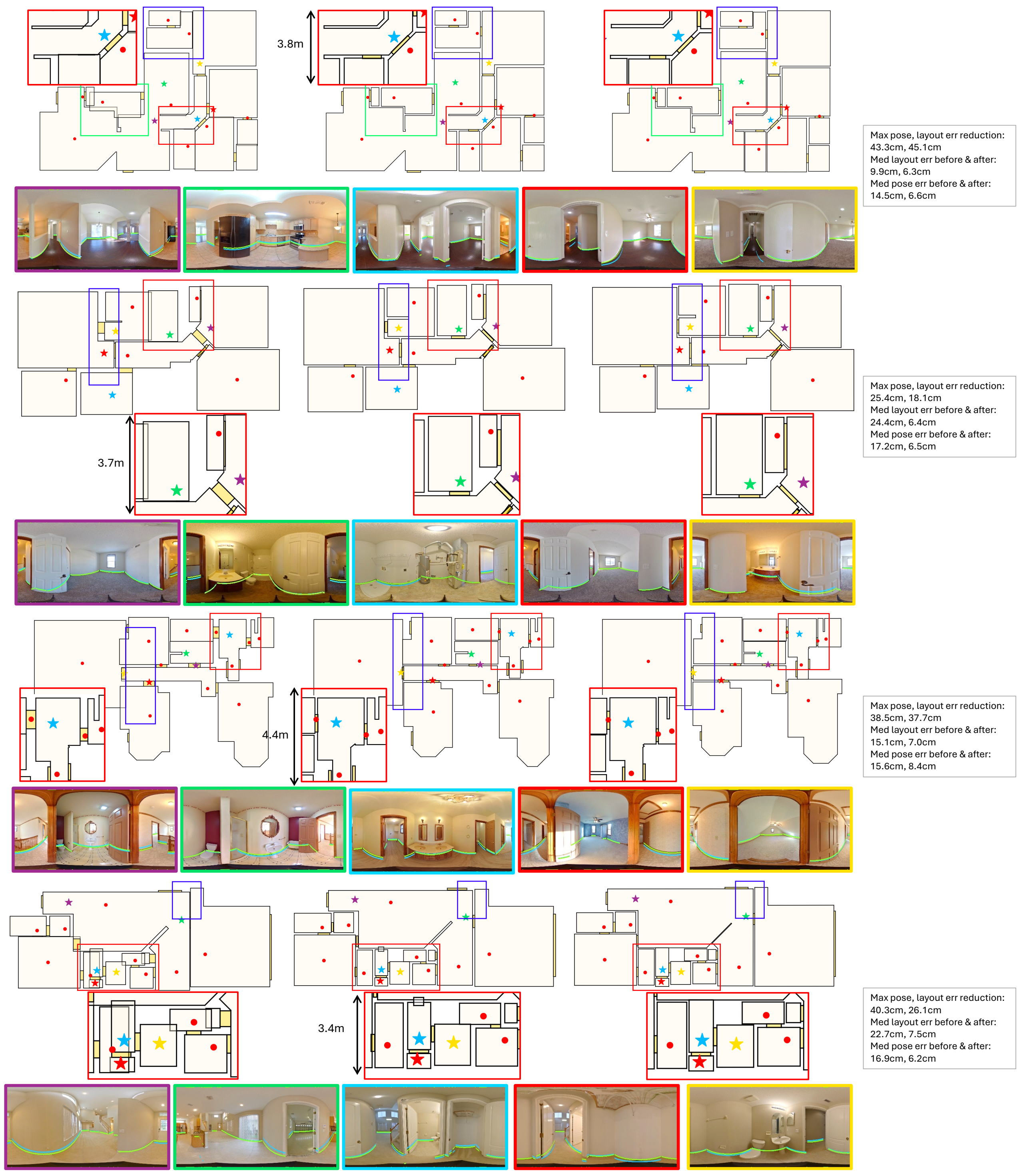

4.3m
Max pose, layout err reduction: 48.8cm, 38.6cm
Med layout err before & after: 27.8cm, 8.0cm
Med pose err before & after: 21.5cm, 8.4cm
3.4m
Max pose, layout err reduction: 60.5cm, 63.4cm
Med layout err before & after: 19.4cm, 11.2cm
Med pose err before & after: 44.5cm, 9.3cm
5.1m
Max pose, layout err reduction: 37.0cm, 47.2cm
Med layout err before & after: 23.7cm, 9.0cm
Med pose err before & after: 17.6cm, 8.6cm
4.1m
Max pose, layout err reduction: 18.9cm, 39.0cm
Med layout err before & after: 11.4cm, 7.4cm
Med pose err before & after: 14.0cm, 4.3cm

Figure 6. 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.



Max pose, layout err reduction:
43.3cm, 45.1cm
Med layout err before & after:
9.9cm, 6.3cm
Med pose err before & after:
14.5cm, 6.6cm

Max pose, layout err reduction:
25.4cm, 18.1cm
Med layout err before & after:
24.4cm, 6.4cm
Med pose err before & after:
17.2cm, 6.5cm

Max pose, layout err reduction:
38.5cm, 37.7cm
Med layout err before & after:
15.1cm, 7.0cm
Med pose err before & after:
15.6cm, 8.4cm

Max pose, layout err reduction:
40.3cm, 26.1cm
Med layout err before & after:
22.7cm, 7.5cm
Med pose err before & after:
16.9cm, 6.2cm

Figure 7. 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.

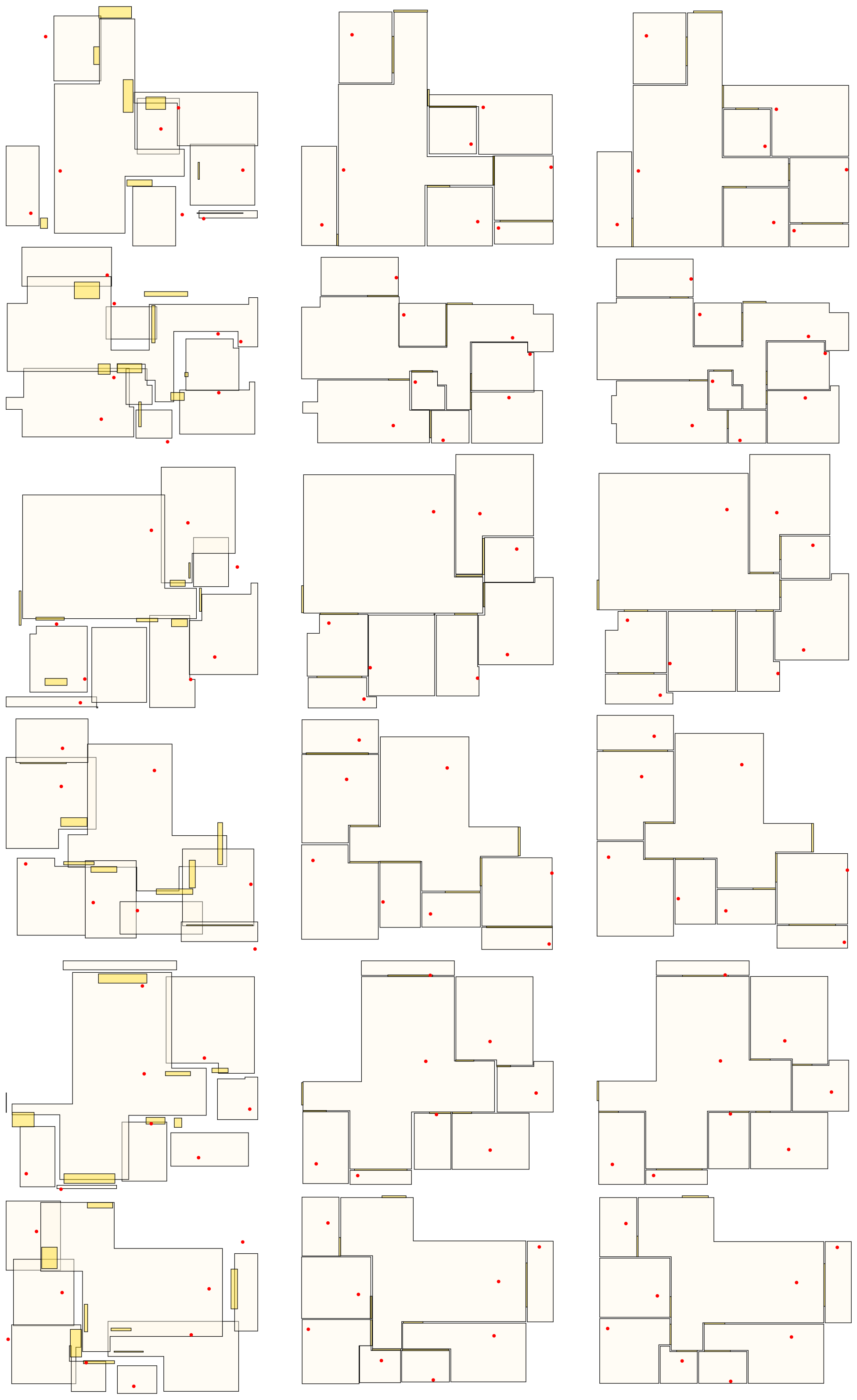


Figure 8. Qualitative results trained and tested on RPLAN dataset (page 4), with input densities at one input image from each partial room. The topdown views from left to right are before, after *BADGR* optimization and the ground truth. The initial state is created by adding Gaussian noise from 20-step diffusion q-sampling [3] 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.

References

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

[2] 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

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

[4] 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. 3

[5] 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

[6] 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

[7] 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. 3

[8] 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. 1

[9] 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, 4

[10] 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. 3

[11] 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. 4

[12] 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