Supplementary Material
Zillow Indoor Dataset:
Annotated Floor Plans With 360^o Panoramas and 3D Room Layouts

1. Panoramic Image Capture

Discussed in Section 3.2 (Capture Process) of the main paper.

We provide photographers a custom iOS app to facilitate panoramic capture. They are asked to go through a checklist: (1) tidy up each room, (2) turn on all overhead and accent lights, (3) turn off fans and TVs, and disable any moving objects, (4) open all interior doors, and (5) avoid capturing people, pets, and objects that may display personal information. Also, they are instructed to plan a route through the home to capture multiple panoramas for all rooms, including connecting hallways and garages. For each room smaller than 8 feet across in any direction, they are asked to take one panorama outside and one panorama inside the door, or entryway. All interior doors are to be open during capture to facilitate multi-view reconstruction and spatial reasoning. If any room dimension exceeds 8 feet, photographers are asked to take additional panoramas; capture locations are chosen such that they are 8-10 feet apart and are within sight of each other.

1.1. Hardware Choice

There are many choices for 360° panoramic cameras with a wide range of price tags and features. Such cameras come with different lens configurations, per-pixel image quality and 3D/VR video capabilities. For example, there are (1) consumer level, widely available two fish-eye lenses systems[1], (2) professional level 360 3D/VR cameras[2], (3) customizable multi-camera systems supporting 6DoF head-motion parallax by computing depth from multi-view high-overlap stereo [6], and (4) specialized 360 RGB-D cameras based on stitching multiple limited FoV RGB-D scans from a fixed tilt and rotation point [1].

Our requirement for large scale and affordable pipeline necessitates a portable, easy to carry and charge, low-cost RGB camera. We use the Ricoh Theta (V and Z1), commodity 360° panoramic cameras with high resolution and accurate image stitching technology that have HDR support. We capture all panoramas with 3 auto bracketing exposure configurations using the default white balance settings with IMU-based vertical tilt correction enabled. Upon release of Zillow Indoor Dataset , we will provide the tilt-corrected LDR tone-mapped panoramas down-sampled to 2048×1024 image resolution.

[1] https://theta360.com/en/

[2] https://liveplanet.net/

1.2. Calibration for Room Scale

Users are instructed to keep the 360° panoramic camera on a tripod with a fixed height while capturing a home; this height can vary between different homes. In order to robustly compute the height of the tripod, and thus infer the geometric scale for the final 3D layouts and floor plans, photographers are asked to take a floor plan calibration image. They can use any of these two calibration targets: (1) a custom made floor marker, made from a flexible and durable mouse pad with a printed AprilTag [4] target, or (2) a US letter paper (8.5" $\times$ 11").

2. Annotation Tool

Discussed in Section 3.3.2 (Room Layout and Interior Features) of the main paper.

For holistic annotation tasks like ours, where we require *high accuracy* with *high throughput*, we found that it was critical to provide initial automation for all our tasks. Furthermore, it was also essential to regularly shadow annotators in order to make sure that the performance and error characteristics of the deployed machine learning (ML) models complement and enhance the human-in-the-loop experience.

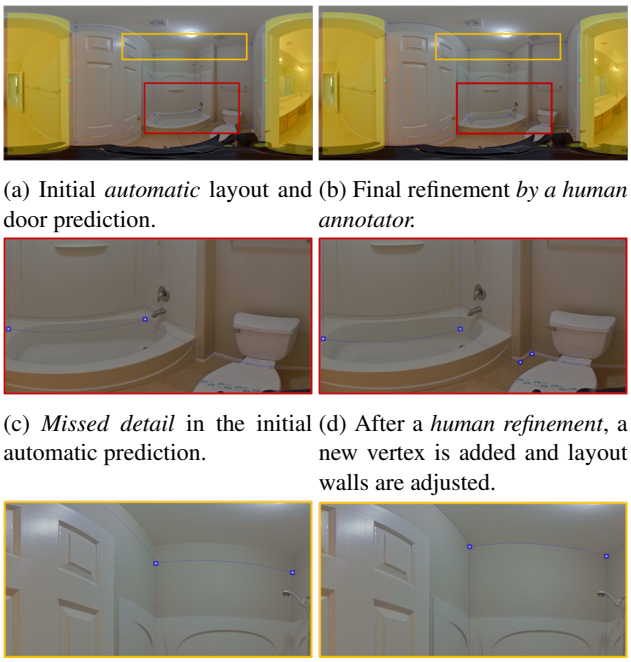
2.1. Room Layout and Interior Features

The goal of the room layout annotation task is to capture the main structural elements of an indoor space, such as the floor, walls, and ceilings boundaries; our goal is to recover the 3D CAD-like geometry of the space as if it was empty [5]. In addition, we collect boundaries for windows, doors, and openings. Also, we ask annotators to indicate if a ceiling is *flat* or *non-flat*. Those are essential semantic elements

for layout understanding and generation of the final floor plan schematic view.

Our internal layout annotation framework is similar in spirit to [11], but with important production-level features to enable high throughput. For example, we added features to exploit the mostly Manhattan nature of indoor environments by the simple push and pull metaphor where clicking and dragging over walls moves them in the direction of the plane normal. Also, we added the ability to create and snap to certain non-Manhattan corners. Other functionalities included dragging and merging corners to form a pre-defined set of angles, snapping the layout edges to strong image gradients, and adjusting labels for windows, doors, and openings. Furthermore, in order to ensure that the UX editing tools and the ML models are in sync, it was crucial to conduct deep and regular error analysis on when and how those models fail in practice. Thus, annotators can perform an easier verification and adjustment task.

Also, our annotation pipeline provides initial estimation for all our tasks, based on continuous training and deployment of state-of-the-art (SOTA) layout estimation models (using [9]), and windows/doors/openings bounding box detection (using [8]). Annotators can quickly zoom-in and manipulate either a wall, to maintain the Manhattan orientation similar to [11], or they can move every single vertex freely, or by applying constraints, such as to preserve orthogonality, or achieve a pre-defined non-Manhattan angle. We found the estimation models critical to a good balance of high throughput and high accuracy of our annotation pipeline (shown in Figure 1).



(a) Initial *automatic* layout and door prediction.

(b) Final refinement *by a human annotator.*

(c) *Missed detail* in the initial automatic prediction.

(d) After a *human refinement*, a new vertex is added and layout walls are adjusted.

(e) Early trained model *missing the ceiling height*.

(f) After a *human refinement*, the ceiling height is fixed and the ceiling *complexity* is indicated as *flat* vs *non-flat*.

Figure 1: **Room Layout and Interior Features**: Annotation pipeline with initial estimations of room layout and windows/doors/openings bounding boxes.

2.2. Merging of Different Rooms

The goal of different room merging is to generate accurate geometry transformation for room layouts at each floor level. Annotators use a UI to establish door-to-door and opening-to-opening correspondence between a room that is added to the floor (reference) and a new room (target). The reference and target room layouts are represented by S_r and S_t, from panorama I_r and I_t, respectively. These correspondences result in a 3D transformation matrix T_{tr} that aligns S_t with S_r. T_{tr} will assume doors or openings of S_r and S_t are aligned. $T_{tr}S_t$ is axis-aligned with S_r.

In the UI, there are two viewports, one for I_r and another for I_t. Both S_r and $T_{tr}S_t$ are superimposed in I_r, and S_t and $T_{rt}S_r$ in I_t. Annotators finetune T_{tr} by moving room corners of either $T_{tr}S_t$ in I_r or $T_{rt}S_r$ in I_t. This operation is showed in Figure 2(g)(h).

2.2.1 Heuristics for Correspondence

We developed a set of heuristics to assist correspondence; this is done by providing candidates for matching pairs of doors or openings within the set of unmerged room layouts $\mathcal{U} = \{U_j, j = 1, ..., N_j\}$ to merged room layouts $\mathcal{M} = \{M_i, i = 1, ..., N_i\}$.

Each candidate matching pair (O_j^n from U_j and O_i^m from M_i) yields the transform $T_{ji,nm}$. This pairing also has a confidence score $K_{ji,nm}$, which is a weighted sum of the following:

1. *Room layout IoU confidence*. This term helps room layouts to have minimum overlaps. This score is based on region overlap between U_j and $\mathcal{M}$:

$$K_{ji,nm}^{\text{IoU}} = -\sum \frac{Area(\cup(\mathcal{M}, U_j))}{Area(\cap(\mathcal{M}, U_j))}. \quad (1)$$

2. *Wall/Doors/Openings (WDO) loop-closure confidence* $K_{ji,nm}^{\text{lc}}$. Similar to [3], walls corresponding to an external boundary or certain rooms must form a closed 1D loop. $K_{ji,nm}^{lc}$ measures WDO loop closure between target room S_t and $\mathcal{M}$.

$$\begin{aligned} K_{ji,nm}^{lc} &= \omega_{wall} \frac{\sum_{n=1}^{N_n} \mathcal{L}(E_{j,n}) \mathcal{P}(E_{j,n}, E_{\mathcal{M}})}{\sum_{n=1}^{N_n} \mathcal{L}(E_{j,n})} \\ &+ \omega_{dw} \frac{\sum_{n=0}^{N_o} \mathcal{P}(O_j^n, O_{\mathcal{M}}^{!(i,m)})}{N_0}, \end{aligned} \quad (2)$$

where $E_{j,n}$ is the 2D (top-down view) line segment of wall n from room layout U_j. Function $\mathcal{L}(E)$ is the length of edge E. $\mathcal{P}(E_{j,n}, E_{\mathcal{M}})$ returns $\{0, 1\}$ and it determines if edge $E_{j,n}$ is close and parallel to any edge from $\mathcal{M}$. ω_{wall} and ω_{dw} are the weights of sub-confidence generated from walls and doors & openings. $O_{\mathcal{M}}^{!(i,m)}$ represents all doors and openings except O_i^m.

3. *Doors/windows re-projection confidence* $K_{ji,nm}^r$. Let the 3D bounding box of a door or window be $\square_X$ from room layout X. And we also have 2D bounding box estimation from panorama X using Faster-RCNN [7] as $\square_X^{2D}$. From one-sided reprojection, we project $\square_{U_j,n}$ onto panorama of M_i as $\square_{U_j,n}^{2D\prime}$ and compute confidence score as $\text{IoU}_{ji,n} = \sum_m \text{IoU}(\square_{U_j,n}^{2D\prime}, \square_{M_i,m}^{2D})$. The doors/windows reprojection confidence is

$$K_{ji,nm}^{dw} = \frac{\sum_n \text{IoU}_{ji,n} + \sum_m \text{IoU}_{ij,m}}{2}. \quad (3)$$

4. *Panorama temporal confidence* $K_{ji,nm}^t$ is the difference between capture timestamps of M_i and U_j.

3. Floor Plan Cleanup

Discussed in Section 3.3.4 (Floor Plan Generation) of the main paper.

Our final annotation task is to obtain the complete, watertight 2D floor plan. This task is relatively straightforward for a human annotator that has access to the annotations from the previous stages. The challenge lies in "drawing" the final watertight 2D polygons so that they closely follow the primitives (walls and junctions) of the merged local room layouts, while resolving slight global inconsistencies, e.g. small drift or outer walls misalignment. After the room merge step, the floor map is composed of a group of room shapes, where each wall is constructed by one or multiple planes from individual room shapes which were created separately. The walls of the room merge output are not constrained globally at a floor level, hence lacking visual appeal and consistency. In the meantime, due to the fact that our input panoramic images do not usually cover all the spaces within the structure contour, e.g. closets, inaccessible space in wall, stairs, etc, the room merge floor plan does not have room shape coverage on these spaces. Thus, there will be missing islands from the room merge floor plan. We further ask annotators to (1) clean up and center room labels and dimensions (from a set of dictionary), (2) add missing spaces, such as small closets and stairs, and (3) indicate open to below or above polygons as well as unexplained spaces. Figure 3 illustrates two examples of such cleanups and the corresponding floor plans before and after cleanups.

	Door	**Window**	**Opening**
Avg. Precision	0.855	0.764	0.577

Table 1: Average precision of Faster-RCNN for detectiion of doors, windows and openings.

4. LayoutLoc

Discussed in Section 3.3.5 and Section 4.2 (Secondary Panorama Localization & Multi-View Registration) of the main paper.

Let the sets of primary and secondary panoramas be $\mathcal{P} = \{I_i, i = 1, ..., N_p\}$ and $\mathcal{S} = \{J_j, j = 1, ..., N_s\}$, respectively. During the floor map generation process, each room layout S_i is created from I_i. The location of I_i known within S_i (and hence within the final floor plan).

We use a separate procedure to localize the secondary panoramas; it starts with automated localization using LayoutLoc. Annotators then use a UI similar to that for room merge to refine the auto-generated camera poses by visual alignment. As a result, every secondary panorama J_j is associated with some room layout S_k.

LayoutLoc consists of three steps:

1. **Reference Room Layout Retrieval**. All panoramas have timestamps. Given timestamp $t(J_j)$ for secondary panorama J_j, we retrieve a set of primary panoramas with similar timestamps (and likely close spatial proximity): $\mathcal{T}_j = \{K_k, k = 1, ..., N_j\} \subset \mathcal{P}$. $\mathcal{T}_j$ with their layouts are references for J_j.

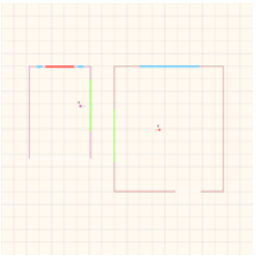
2. **Camera Pose Proposal**. In this step, we run room layout estimation model HorizonNet [9] and object detection model Faster-RCNN [7] on $J_j \in \mathcal{S}$ and $K_k \in \mathcal{T}_j$ as estimates of their annotated room layouts. To generate camera pose proposals for J_j relative to K_k, we hypothesize matches between room corners as well as doors and windows. For each hypothesized match, we also check for pairwise vanishing line alignment. Algorithmic details can be found in Algorithm 1. RCNN accuracies for detecting doors and windows on our dataset are shown in Table 1.

3. **Camera Pose Evaluation**. Once the proposals are generated, we score them and select the winning pose. Given the panoramas and their layouts (I_A, S_A, I_B, S_B), the confidence associated with the kth proposed camera pose $P_{BA,k}$ is computed based on the image reprojection errors of the room corners, doors, and windows. Note that since the confidence measure is symmetric, it is immaterial which are reference (primary) and secondary panoramas.

   Let X_A and X_B be the set of hypothesized matching corners for layouts S_A and S_B, respectively, and

(a) *Annotated* room layout and WDO boundaries of the *foyer panorama*.

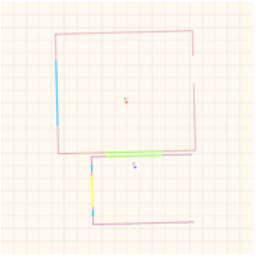
(b) *Annotated* room layout and WDO boundaries of the *dining room panorama*.



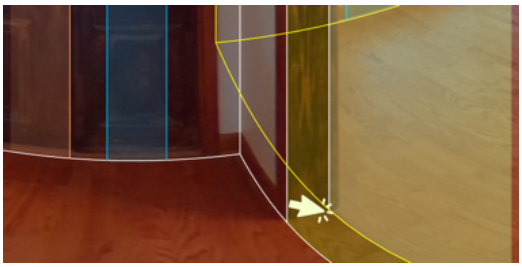
(c) *Initial* top-down view (*no alignment*).

(d) *Automatic proposal*: foyer point of view.

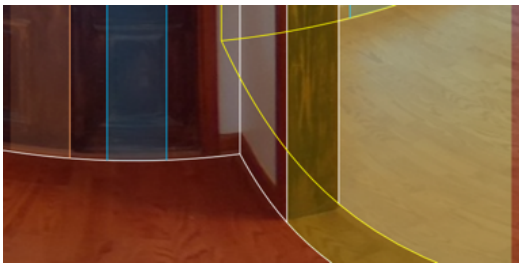
(e) *Automatic proposal*: dining-room point of view.



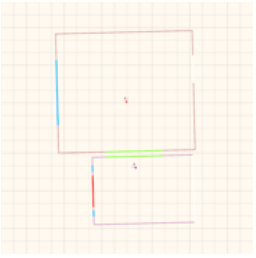
(f) *Automatic proposal*.



(g) *Automatic proposal* with small misalignment



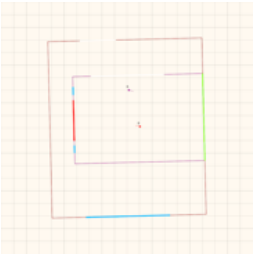
(h) *Human refinement* to snap corners



(i) *Final* result.

(j) *Low-ranked proposal*: foyer point of view.

(k) *Low-ranked proposal*: dining-room point of view.



(l) *Low-ranked proposal*.

Figure 2: **Assisted Room Merging**: optimized for *high accuracy* with *high throughput*. In the 1st row we show the annotated room layouts and WDO boundaries from the previous stage. In the last column we visualize the top-down layout projection before any proposed alignment. In the 2nd row, we demonstrate the highest-ranked automatic alignment, that would be surfaced to our human annotators, when they select to merge those two panoramas. This is based on our rank-and-propose pairing process, using a default door width, as described in Section 2.2. Given the current proposal, the layout of the other panorama is rendered with yellow outlines. Human annotators can further refine the room-to-room alignment, as shown in the 3rd row, by simply dragging the yellow outlines. In the last row, we depict an example of a low-ranked snapping proposal, where doors/openings marked in green to indicate the proposed snapping pair. Notice that multiple of our criteria, described in Section 2.2.1 are not well satisfied, e.g. (1) IoU is significant and (2) semantic WDO elements do now align well. Notice how in the highest-ranked proposal (2nd row), semantic elements match well, which is part of our multi-view consistency check that relies on running [8] on the full 360 image.

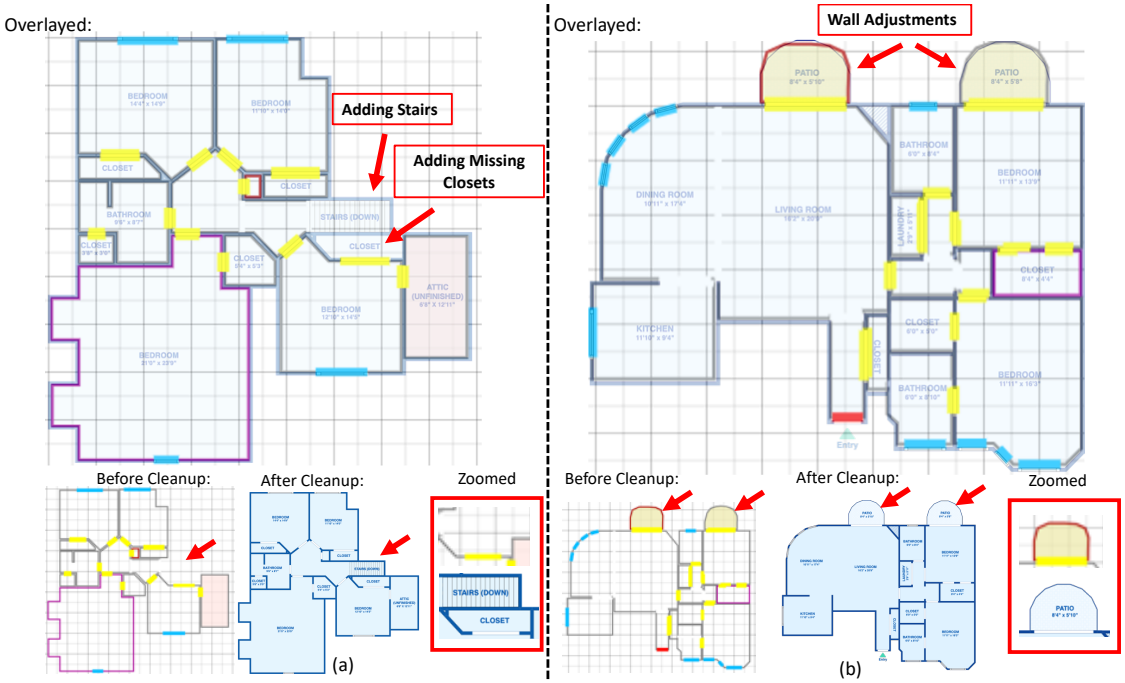


Figure 3: **Floor Plan Cleanup:** Two examples of changes applied to floor plans after cleanup. (a) Adding stairs and missing closets, (b) Wall alignments/refinements. Top is the overlay of before and after. Bottom shows before, after, and the zoomed versions of differences for each example. Red arrows point to the differences.

Algorithm 1 LayoutLoc Pose Proposal Generation. Notes: $*_{*,p}$ refers to proposals, $*_{*,a}$ refers to annotated values, and θ is the layout horizontal orientation. The output is P_{BA}.

Require: Panoramas I_A and I_B
  $S_A \leftarrow$HorizonNet(I_A)
  $S_B \leftarrow$HorizonNet(I_B)
  $\phi_A \leftarrow$VanishingAngleEstimation(I_A)
  $\phi_B \leftarrow$VanishingAngleEstimation(I_B)
  for $X_{Ai} \in$ corners of S_A **do**
    for $X_{Bj} \in$ corners of S_B **do**
      for $\delta \in \{0°, 90°, 180°, 270°\}$ **do**
        $X_{BjAi} = X_{Ai} - X_{Bj}$
        $X_{Bj,p} = X_{BjAi}$.rotateAround$(X_{Ai}, \phi_B - \phi_A + \delta)$
        $\theta_{Bj,p} = \theta_{Aj,a} - \phi_B + \phi_A - \delta$
        $\{P_{BA}\}$.push$(X_{Bj,p}, \theta_{Bj,p})$
      end for
    end for
  end for

$\mathcal{G}_A(X)$ be the projection function that maps point X to panorama I_A. The confidence score associated with the corners is

$$Z_c = \frac{\Sigma_i \sqrt{\mathbf{d}_{ABi}^T \Omega \mathbf{d}_{ABi}} + \Sigma_i \sqrt{\mathbf{d}_{BAi}^T \Omega \mathbf{d}_{BAi}}}{2}, \quad (4)$$

where $\mathbf{d}_{ABi} = \mathcal{G}_A(X_{Ai}) - \mathcal{G}_A(X_{Bi})$ is the reprojection error for corner i in panorama A (with $\mathbf{d}_{BAi}$ similarly defined), Ω is a 2×2 diagonal matrix of weights derived from error distributions based on annotated data.

The other part of the confidence score is associated with door and window reprojection errors. Let the 3D bounding box of a door or window (not differentiated at this point) be $\square_X$ from panorama X (with X being A or B). The one-sided reprojection error is measured as IoU_{ABi} between $\mathcal{G}_A(\square_{Ai})$ and $\mathcal{G}_A(\square_{Bi})$ for the ith hypothesized matched door or window at panorama A. The door/window confidence score is

$$Z_{wd} = \frac{\Sigma_i(\text{IoU}_{ABi} + \text{IoU}_{BAi})}{2}. \quad (5)$$

The camera pose confidence is the sum of Z_c and Z_{wd}. The final camera pose associated with the largest confidence is selected.

Type	$\#_c$	$\#_s$	$\overline{\#}_p/c$	SfM			LayoutLoc			
				$\%_{>2}$	$\overline{x}[cm]$	$\overline{s}[cm]$	$\%_{>2}$	$\%_{=2}$	$\overline{x}[cm]$	$\overline{s}[cm]$
bedroom	2883	5803	3.12	0.34	**4.23**	1.05	**0.948**	0.954	6.78	7.99
garage	695	1288	3.04	0.58	**2.12**	0.25	**0.905**	0.929	17.41	16.05
entryway	369	707	3.01	0.81	**1.53**	0.32	**0.947**	0.981	6.33	8.44
kitchen	813	1580	3.13	0.69	**1.82**	0.79	**0.915**	0.899	13.43	16.19
living room	1365	3443	3.53	0.63	**3.65**	0.93	**0.965**	0.931	9.62	11.01
basement	143	284	2.99	0.41	**1.04**	0.08	**0.851**	0.803	11.37	13.41
bathroom	1757	2414	2.52	0.56	**3.25**	1.02	**0.908**	0.894	6.90	9.93
hallway	1382	2699	3.14	0.59	**3.04**	0.76	**0.915**	0.899	6.65	10.04
closet	1231	1428	2.39	0.32	**9.89**	3.70	**0.857**	0.816	6.81	10.43
other	2494	5885	2.77	0.66	**3.34**	0.62	**0.947**	0.917	8.32	12.69
Total	13158	25531	2.94	0.55	**3.29**	0.83	**0.933**	0.905	8.50	11.28

Table 2: Localization accuracies for SfM and *LayoutLoc*. $\#_c$: numbers of cliques, $\#_s$: numbers of secondary panoramas. $\overline{\#}_p/c$ average number of panoramas per clique. $\%_{>2}$: success rate of localized panoramas for clique sizes greater than 2, $\%_{=2}$: for clique sizes of 2.

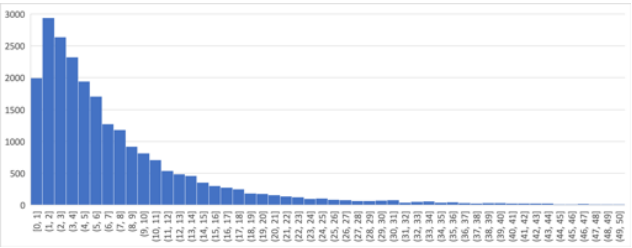


Figure 4: Overall distance error [cm] distribution of panorama localization using LayoutLoc.

Table 2 shows the per-room-type accuracy comparison between SfM and LayoutLoc. $\%_{>2}$ is the success rate when localizing panoramas from a clique of size 3 and above. Much more annotation work will be required to correct every SfM localization result for clique of size 2 (which is why we did not report results for cliques of size 2).

LayoutLoc significantly outperforms SfM at $\%_{>2}$ for all room types. LayoutLoc can also localize panoramas from clique of size 2 with correct floor map scale with similar success rate to $\%_{>2}$. Overall, 22.8% of secondary panoramas come from cliques of size 2. From $\overline{\#}_p/c$, we can find that many panoramas of room types such as bathroom and closets are in cliques of size 2. SfM tends have a lower success rate in (unfurnished) bedrooms, garages, and basements, as they tend to be mostly featureless. Layoutloc works uniformly well across all room types because it uses doors and windows as additional matching cues. Figure 4 along with metrics $\overline{x}[cm]$ and $\overline{s}[cm]$ in Table 2 show that LayoutLoc produces higher spatial error compared with SfM, especially in larger rooms such as garages, kitchens, and basements. Overall, the average error of layoutLoc is under 20cm for all room types.

Figure 5: **Window/Door/Opening:** First row: Examples of annotations of Window, Door and Openings in Zillow Indoor Dataset, Second row: Examples of Faster R-CNN [8] predictions.

5. Layout Annotations

5.1. Complete Geometry

Discussed in Section 3.3.3 (Room Merging) of the main paper.

Our human-based annotation process produces partial layouts separated by openings, which we expand and consolidate into complete geometry shared across panoramas. Co-localized secondary panoramas naturally inherit the complete layout within which they reside. In alternating rows, Figure 16 first shows partial layouts with openings in green, followed by complete geometry visualized from the three panoramic perspectives.

5.2. Door and Window Annotations

As mentioned, ZInD provides 2D bounding boxes of windows and doors. However, in the interest of throughput and efficiency, our floorplan generation pipeline uses the window and door left-right boundaries only. As such, as can be seen in the visualizations of Figure 9 for example, we commonly visualize these annotations as spanning from floor to ceiling.

5.3. Layout Complexity Classification

Discussed in Section 4.1 (Layout Estimation - Train/Val/Test Splits) of the main paper.

For single perspective layout estimation, we classify layouts as "simple" or "complex", based on the amount of occluded corners. Layouts produced by significant expansion and consolidation, such as those of open floor plans, are typically classified as complex due to significant occlusion. Figure 18 shows examples of both classes. The first three examples are of simple shapes, which we include in our dataset for layout estimation evaluation using [9]. We hope

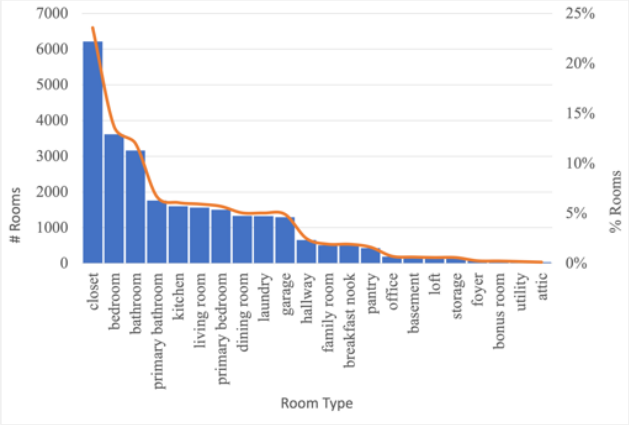


Figure 6: Histogram of room label types in Zillow Indoor Dataset

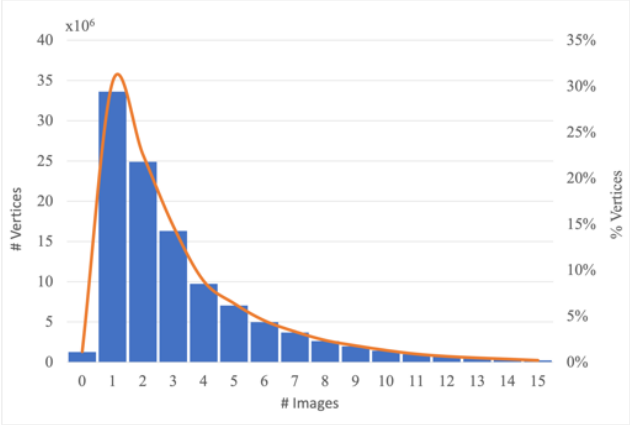


Figure 7: Histogram showing how many panoramas "see" a layout vertex. The mode is 2 and the average is 3.2.

that our complex shapes will support other research applications, such as multi-panorama layout estimation.

5.4. Statistics

Discussed in Section 5 (Discussion) of the main paper.

Room and panorama labels in ZInD went through a clean-up process to enforce uniformity. A distribution of room labels is displayed in Figure 6. Figure 10 shows representative examples of primary room categories in ZInD.

6. Derived Quantities

Discussed in Section 3.4 (Dataset Statistics) of the main paper.

6.1. Ray Casting and Visibility

In the floor plan, we apply ray casting to our complete geometry, to derive per-panorama visible layout, as well as covisibility between panorama pairs. 17 shows examples of this process for two sets of panorama pairs. The first row shows a top-down 2D map of the covisibility. The covisibility map is highlighted in green, as the intersection of the visibility map of panorama 1, in blue, and the visibility map of panorama 2, in red. The calculated score is contained in the text above. The next two rows show a visualization of the following quantities for each panorama: 1) The complete geometry layout 2) the derived visible geometry layout 3) The covisibility map from the panorama's perspective. Two pairs of panoramas are shown, with the first pair displaying low covisibility, and the second displaying intermediate covisibility.

6.2. Covisibility Score

The covisibility score is a measure of the amount of visual overlap between two cameras (say A and B). It depends on the camera poses, their fields of view, and scene geometry G (in our case, this is the floor plan). Let G_A be the geometry visible to camera A; G_B is similarly defined. The geometry visible to both cameras is $G_{AB} = G_A \cap G_B$. We define $\Theta_Y(X)$ as the visual occupancy of geometry X for camera Y, i.e., the fraction of Y's image occupied by X. Then the covisibility score is $\Omega_{ABG} = 0.5 * (\Theta_A(G_{AB}) + \Theta_B(G_{AB}))$. Note that Ω_{ABG} ranges from 0 to 1. In our implementation, we simplify the estimation of Ω_{ABG} by computing it in 2D domain (with G being the 2D floor plan) and field of view is specified over discrete 1D images.

6.3. Total Covisibility Histogram

To demonstrate the visual overlap produced by our dense localization, we interpolate along wall segments and for each point, compute the number of panoramas which observe this point. Figure 7 shows the co-visibility histogram, showing how many cameras observe how many vertices in layouts. We believe this is a good measure of visual density of our capture.

7. F-Score

Discussed in Section 4.1 (Layout Estimation) of the main paper.

In practice, we have found Intersection-over-Union (IoU) to be less effective at penalizing inaccuracy as shape complexity increases. As shown in the example in Figure 8, a high IoU can result for shapes that would require significant human touch-up in order to be suitable for floor plan construction. Structures such as bay windows and other detail geometry are important features which convey the uniqueness of a floor plan; rendering these structures correctly is of high importance.

In 3D reconstruction, Precision and Recall are defined as

Figure 8: Shape with IoU $> 95\%$ despite failure to capture the structure of a bay window, requiring significant human touch-up. The Precision, Recall, and F-Score for **corner detection** all equal 50%.

functions of the error between the reconstructed and ground truth point clouds, $\mathcal{R}$ and $\mathcal{G}$[10]. Precision is defined as

$$e_r = \min_{g \in \mathcal{G}} \|r - g\|_2 \tag{6}$$

$$P(t) = \frac{100}{|R|} \sum_{r \in \mathcal{R}} [e_r < t] \tag{7}$$

and Recall defined as,

$$e_g = \min_{r \in \mathcal{R}} \|r - g\|_2 \tag{8}$$

$$R(t) = \frac{100}{|G|} \sum_{g \in \mathcal{G}} [e_g < t] \tag{9}$$

This definition matches points with *any* neighbor, with no requirement of exclusivity. As a result, this definition does not emphasize the desired sparsity, or completeness, of layouts. Motivated by the downstream task of producing a sparse layout representation of a room, we adopt a method for computing the true postiives (TP), false positives (FP), and false negatives (FN), as is common in the object detection literature [2], where the onus is placed on the detector to both correctly filter out redundant detections, *and* to detect uniquely each individual ground truth instance. We have found that in practice it is important to penalize errors in detecting the true sparse set of corners, as these errors result in necessary fine-grained interactions in our human annotation workflow, which costs valuable time.

In practice, with the matching threshold set sufficiently low, this is avoided in the majority of cases; nonetheless, we impose this strictly to guard against any such cases. As the straightened camera and single flat-plane ceiling assumptions ensure that this computation is the same for both the ceiling and floor vertices, we compute these quantities on the floor vertices. The procedure for computing these elements of the confusion matrix is depicted in algorithm 1.

Given a vertex distance matrix D, with dimensions corresponding to the number of predicted vertices, n_{pred}, and the number of ground truth vertices, n_{gt}, and matching threshold t, we iterate over the maximum number of possible true positives. We take the minimum of the distance matrix and compare this to the matching threshold, and aggregate those pairs that satisfy this criterion as the number of true positives. We set the corresponding row and column of the matching pair to a large number to prevent further matching, thus imposing exclusivity.

Algorithm 2 Calculate TP, FP, FN

Require: Distance matrix $\underset{n_{pred} \times n_{gt}}{\mathrm{D}}$, matching threshold t

```
TP← 0
for i = 1 to min({n_pred, n_gt}) do
  if min(D) < t then
    TP += 1
    j, k ← arg min(D)
    D[j, :] ← inf
    D[:, k] ← inf
  else
    break
  end if
end for
FP← n_pred − TP
FN← n_gt − TP
```

In addition to application on corner image pixel coordinates, this metric may also be applied in the floor plane's 2D coordinates. Here, we share the results in image pixel space, which simplifies the selection of the matching threshold, t. For this work we select t as 1% of the image width.

This method can be further motivated as a related exten-

sion of the corner error commonly reported for cuboid-only layouts, with a greedy matching performed to associate vertex pairs.

We show examples of F-score performance in Figures 12 - 15. These configurations come from Table 5 found in the main paper. The predicted layout is orange with corners denoted by triangles. The GT layout is in blue with corners denoted by crosses. Matched corners are highlighted in green.

References

[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision (3DV)*, 2017. 1

[2] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, Sept. 2009. 8

[3] Chen Liu, Jiajun Wu, Pushmeet Kohli, and Yasutaka Furukawa. Raster-to-vector: Revisiting floorplan transformation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[4] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407, May 2011. 1

[5] Giovanni Pintore, Marco Agus, and Enrico Gobbetti. AtlantaNet: Inferring the 3D indoor layout from a single 360 image beyond the Manhattan world assumption. In *European Conference on Computer Vision (ECCV)*, 2020. 1

[6] Albert Parra Pozo, Michael Toksvig, Terry Filiba Schrager, Joyce Hsu, Uday Mathur, Alexander Sorkine-Hornung, Rick Szeliski, and Brian Cabral. An integrated 6dof video camera and system design. *ACM Trans. Graph.*, 38(6), Nov. 2019. 1

[7] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(6):1137–1149, June 2017. 3

[8] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015. 2, 4, 6

[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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1047–1056, 2019. 2, 3, 6

[10] Maxim Tatarchenko, Stephan R. Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? *CoRR*, abs/1905.03678, 2019. 8

[11] Shang-Ta Yang, Chi-Han Peng, Peter Wonka, and Hung-Kuo Chu. Panoannotator: A semi-automatic tool for indoor panorama layout annotation. In *SIGGRAPH Asia Posters*, pages 1–2, December 2018. 2

[12] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3D: A large photo-realistic dataset for structured 3D modeling. In *European Conference on Computer Vision (ECCV)*, 2020. 12

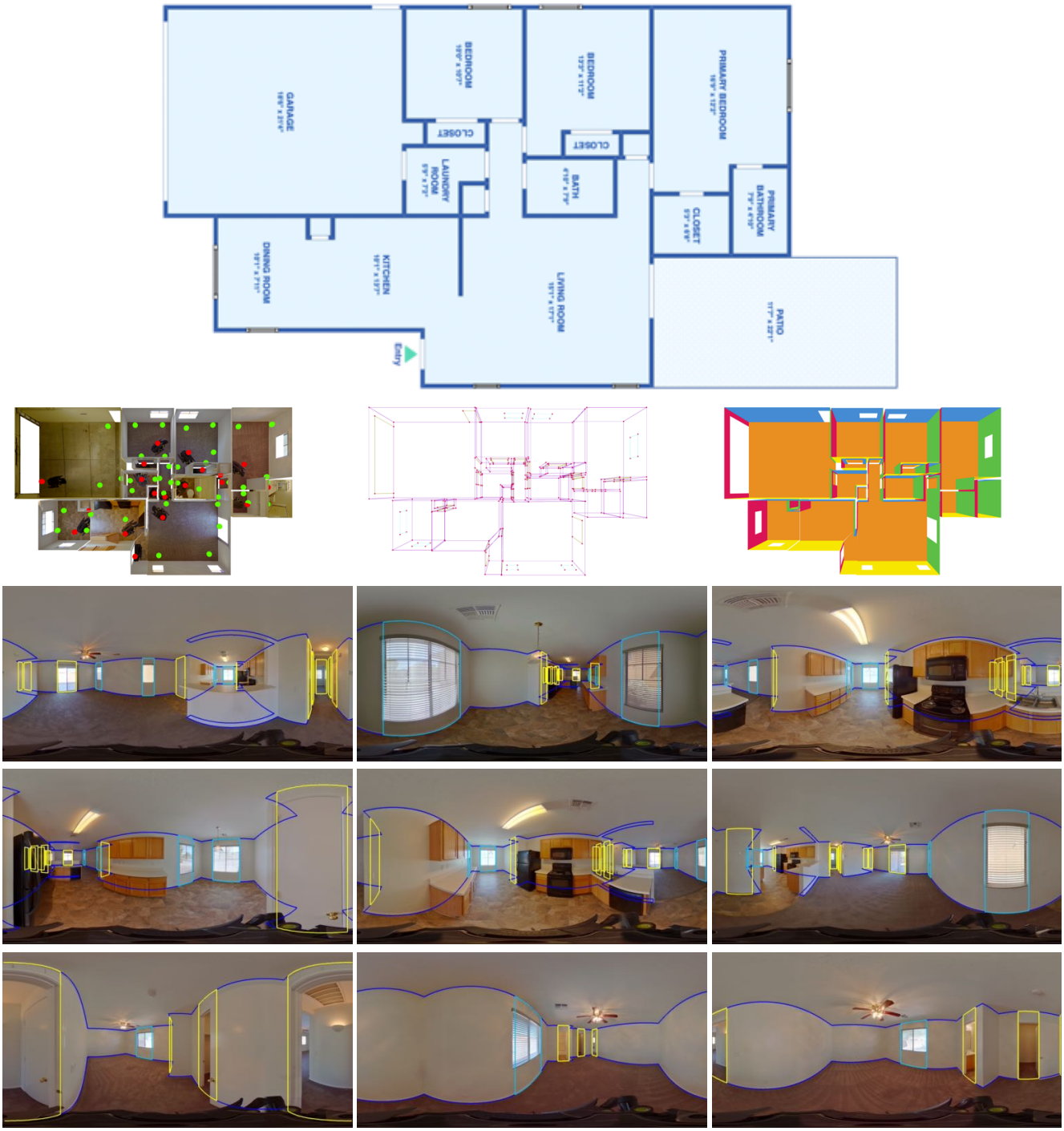
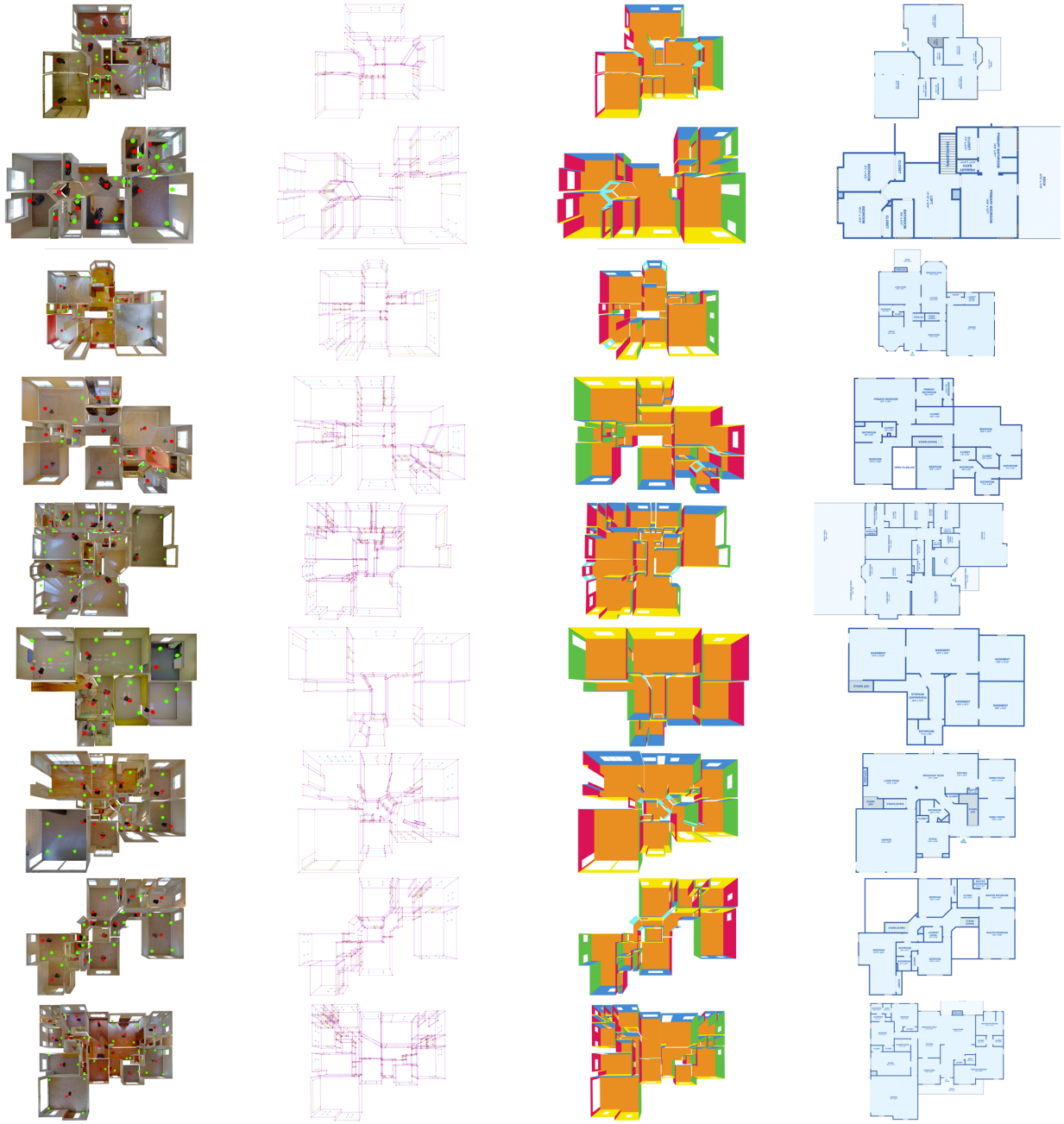


Figure 9: **Zillow Indoor Dataset Sample**: An example of the level of density found within ZInD for one home. Row 1 is the 2D Floor Plan. Row 2 are the 3D ground truth structures. Row 3-4 are panoramas from the dining room, kitchen, and living room. Row 5 are panoramas from the primary bedroom.

Figure 10: **Room Types with Complete Geometry**: Representative examples of primary room types contained in ZInD, as summarized in figure 6. In row order, we show bathrooms, bedrooms, dining rooms, kitchens, living rooms, and garages. As shown in the panorama in row 5, column 1, for our complete geometry we adopt the median ceiling height of the input partial An example of the level of density foundshapes. Certain content above the ceiling line, such as skylight windows, are not annotated.



(a) 3D Floor Plan (Textured) (b) Wireframe (c) Planes (d) 2D Floor Plan (Cleanup)

Figure 11: **2D/3D Primitives**: Examples of 3D ground truth structures in the Zillow Indoor Dataset dataset, similar to [12]. In the 3D textured floor plan, the red dots indicate the *primary* panoramas and the green dots indicate the *secondary* panoramas. In the wireframe the yellow lines denote a door and the blue lines a window. The planes are colored by the normal. The 2D floor plan represents the result of the final cleanup stage, where (1) small wall misalignments are fixed (2) missing spaces, like small closets and staircases, are added (3) outdoor spaces, like decks, patios and balconies, are delineated, which panoramas we have explicitly flagged and removed (4) room labels and dimensions are verified. Note that the windows and doors heights are fixed for visualization purposes only, which is a limitation of our current rendering routine. The underlying Zillow Indoor Dataset dataset has a human-annotated 2D bounding-box for every door and window.

Figure 12: **Flat Ceiling:** Panoramas with flat ceilings in decreasing row order of F-score. The predicted layout is orange with corners denoted by triangles. The GT layout is in blue with corners denoted by crosses. Matched corners are highlighted in green.

Figure 13: **Non-Flat Ceiling:** Panoramas with non-flat ceilings in decreasing row order of F-score. The predicted layout is orange with corners denoted by triangles. The GT layout is in blue with corners denoted by crosses. Matched corners are highlighted in green.

Figure 14: **4 Corners:** Panoramas with 4 corners in decreasing row order of F-score. The predicted layout is orange with corners denoted by triangles. The GT layout is in blue with corners denoted by crosses. Matched corners are highlighted in green.

Figure 15: **Non-Manhattan:** Panoramas with non-Manhattan room types in decreasing row order of F-score. The predicted layout is orange with corners denoted by triangles. The GT layout is in blue with corners denoted by crosses. Matched corners are highlighted in green.

Figure 16: **Complete Geometry**: In alternating rows, visualizations of our original annotations, which consist of partial polygons separated by openings, followed by our complete geometry annotations, which consist of joint geometry shared between multiple panoramas.

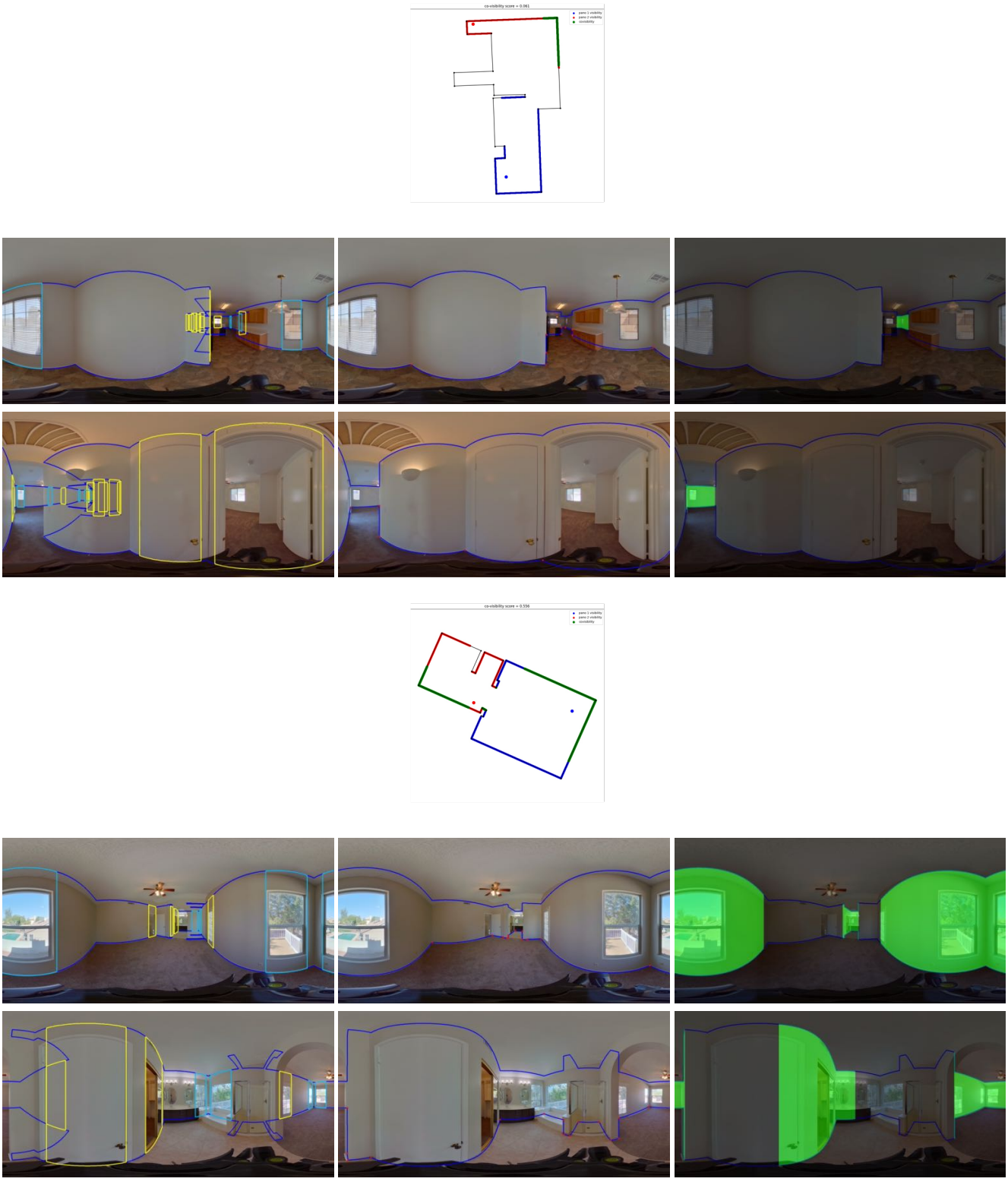


Figure 17: **Visible layout, covisibility map and covisibility score:** Examples of the derived visible layout, as well as the covisibility map and score calculation between pano pairs. We use ray casting to derive the visible layout polygon for each pano, as well as the covisibility map between pano pairs. In these two examples, the first pano pair has low covisibility, while the second pair has mid-range covisibility. The covisibility score is calculated as in section 6.2.

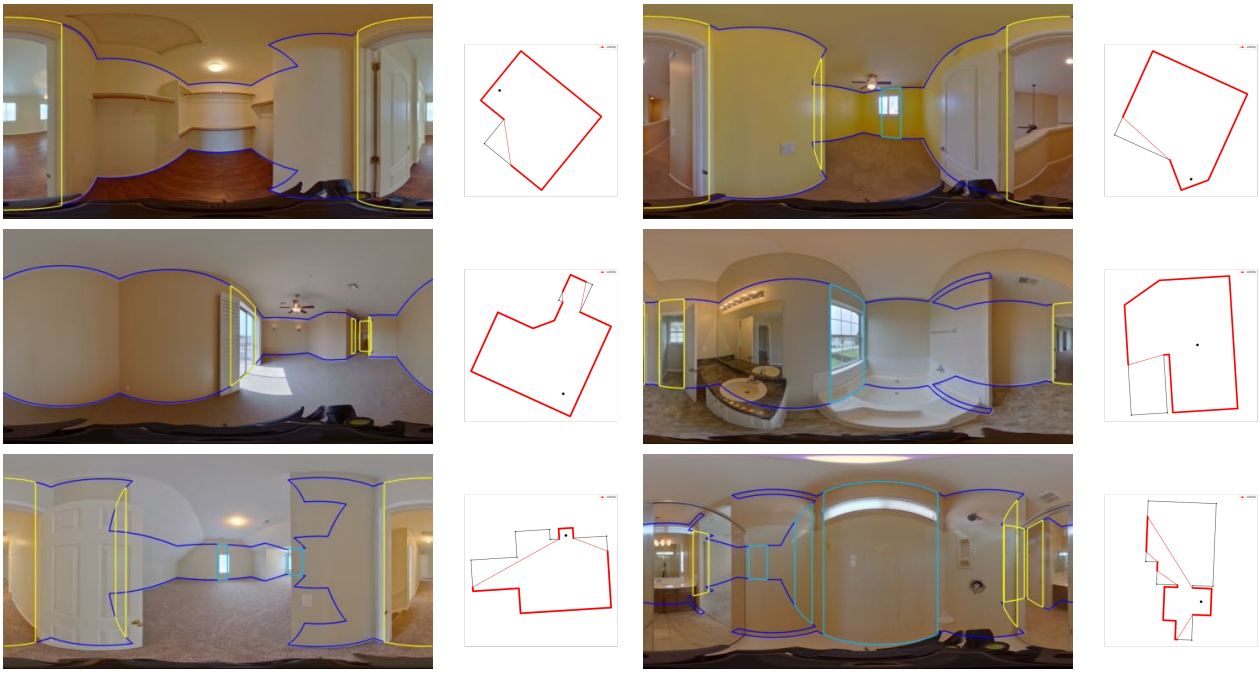


Figure 18: **Classifying room shapes as simple or complex for single perspective layout estimation**: The first three are examples of simple layouts, while the following three are complex. We withhold complex layouts from training and evaluation due to the extensive structural occlusion.