In this appendix, we detail statistics regarding the Scan2CAD dataset in Sec. A. In Sec. B, we detail our evaluation metric for the alignment models. We show additional details for our keypoint correspondence prediction network in Sec. C and we show example correspondence predictions. We provide additional detail for our alignment algorithm in Sec E. In Sec. G, we describe the implementation details of the baseline approaches.

A. Dataset

A compilation of our dataset is presented in Fig. 9. As a full coverage was aimed during the annotation, we can see the variety and richness of the aligned objects.

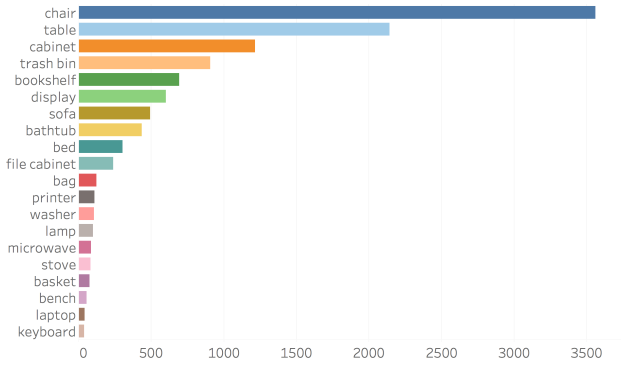

Figure 1: Distribution of top 20 categories of annotated objects in our Scan2CAD dataset.

Statistics We show the object category statistics of our dataset in Fig. 1. Since our dataset is constructed on scans of indoor environments, it contains many furniture categories (e.g., chairs, tables, and sofas). In addition, it also provides alignments for a wide range of other objects such as backpacks, keyboards, and monitors.

Timings The annotation timings per object and per scan are illustrated in Fig. 2 (top) and Fig. 2 (bottom). On an object level, the timings are relatively consistent with little variance in time. On a scan level, however, the variation in annotation time is larger which is due to variation in scene size. Larger scenes are likely to contain more objects and hence require longer annotation times.

Symmetries In order to take into account the natural symmetries of many object categories during our training and evaluation, we collected a set of symmetry type annotations for all instances of CAD models. Fig. 3 shows examples and total counts for all rotational symmetry annotations.

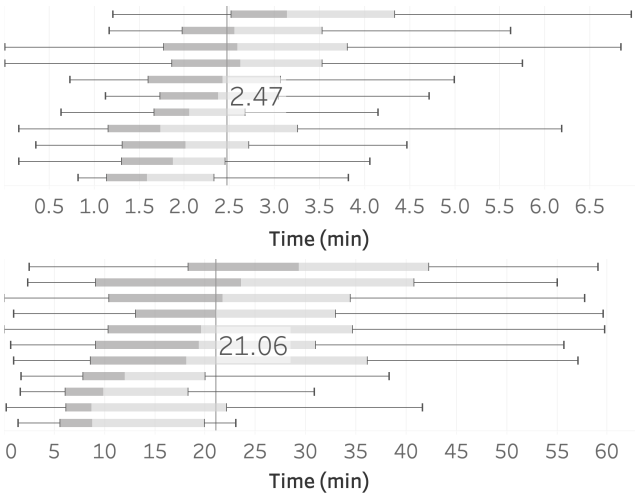

Figure 2: Annotation timing distributions for each annotated object (top) and for each annotated scene (bottom). Each row shows a box-whisker plot with the median time and interquartile range for an annotator. The vertical rule shows the overall median across annotators.

Figure 3: Examples of symmetry annotations.

B. Evaluation Metric

In this section, we describe the details of the algorithm for computing the alignment accuracy. To compute the accuracy, we do a greedy matching of aligned CAD models to the ground truth CAD models.

For a given aligned scene **id-scan** with N aligned CAD models, we query the ground truth alignment for the given scene. The evaluation script then iterates through all aligned candidate models and checks whether there is a ground truth CAD model of the same class where the alignment error is

Data: 1 **id-scan**, N CADs (**id**, **cat**, **pose**)
Result: accuracy in %
Init:
Get N GT-CADs from database with *key*=**id-scan**
Set thresholds $t_t = 20cm, t_r = 20°, t_s = 20\%$
counter = 0;
for *c* **in** *CADs* **do**
  id, **cat**, **pose** = c
  for *c-gt* **in** *GT-CADs* **do**
    $\textbf{id}_{\text{GT}}$, $\textbf{cat}_{\text{GT}}$, $\textbf{pose}_{\text{GT}}$ = c-gt
    if ***cat*** == $\textbf{\textit{cat}}_{GT}$ **then**
      ϵ_t = `Distance` (**pose**.t, $\textbf{pose}_{\text{GT}}$.t)
      ϵ_r = `Distance` (**pose**.r, $\textbf{pose}_{\text{GT}}$.r, $\textbf{sym}_{\text{GT}}$)
      ϵ_s = `Distance` (**pose**.s, $\textbf{pose}_{\text{GT}}$.s)
      if $\epsilon_t \leq t_t$ *and* $\epsilon_r \leq t_r$ *and* $\epsilon_s \leq t_s$ **then**
        counter ++
        remove $\textbf{id}_{\text{GT}}$ from GT-CADs
        break
      end
    end
  end
end
Output: accuracy = counter/N

Algorithm 1: Pseudo code of our evaluation benchmark. **id, cat, pose** denotes the id, category label and 9DoF alignment transformation for a particular CAD model. Note that the rotation distance function takes symmetries into account.

below the given bounds; if one is found, then the counter (of positive alignments) is incremented and the respective ground truth CAD model is removed from the ground truth pool. See Alg. 1 for the pseudo-code.

C. Correspondence Prediction Network

Network details The details of the building blocks for our correspondence prediction network are depicted in Fig. 4. See Figure 4 of the main paper for the full architecture. We introduce the following blocks:

- **ConvBlocks** are the most atomic blocks and consist of a sequence of **Conv3-BatchNorm-ReLU** layers as commonly found in other literature.
- **ResBlocks** are essentially residual skip connecting layers.
- **BigBlocks** contain two **ResBlocks** in succession.

Training curves Fig. 5 shows how much data is required for training the alignment approach. The curves show predicted compatibility scores of our network. We train our

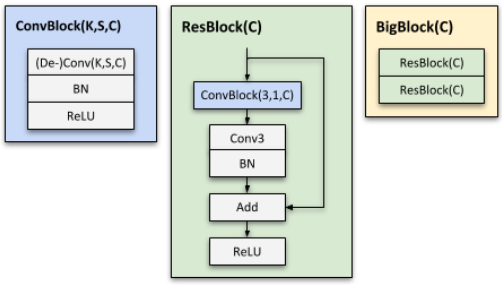

Figure 4: CNN building blocks for our Scan2CAD architecture. **K, S, C** stand for *kernel-size*, *stride* and *num-channels* respectively.

3D CNN approach with different numbers of training samples (full, half and quarter of the dataset), and show both training and validation curves for each of the three experiments. When using only a quarter or half of the dataset, we see severe overfitting. This implies that our entire dataset provides significantly better generalization.

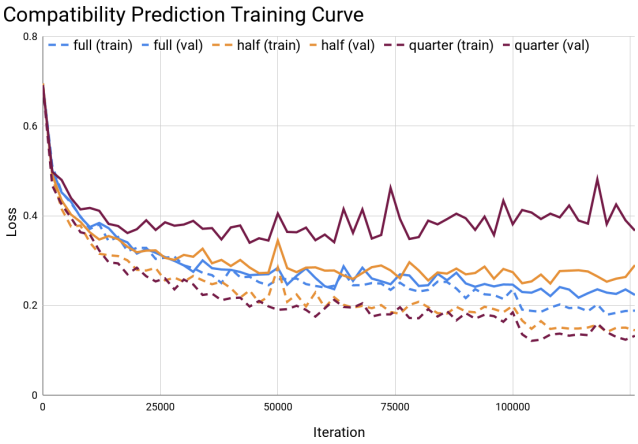

Figure 5: Training and validation curves for varying training data sizes showing the probability score predictions. Experiments are carried out with full, half, and a quarter of the data set size. We see severe overfitting for half and quarter dataset training experiments, while our full training corpus mitigates overfitting.

In Fig. 6, we show the Precision-recall curve of the compatibility prediction of a our ablations (see Sec. 7.1 in the main paper). The PR-curves underline the strength of our best preforming network variation.

Correspondence predictions Visual results of the correspondence prediction are shown in Fig. 8. One can see that our correspondence prediction network predicts as well symmetry-equivalent correspondences. The scan input with a voxel resolution of 3cm and a grid dimension of 64

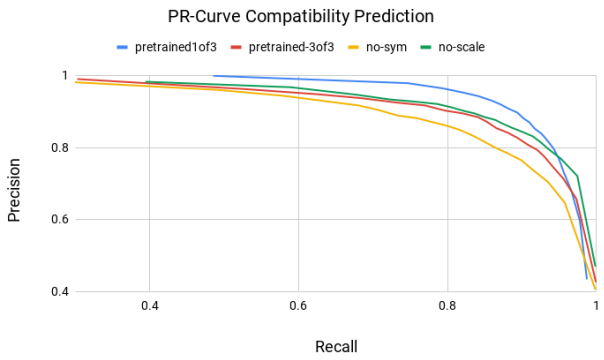

Figure 6: Precision-recall curve of our compatibility score predictions.

can cover 1.92m per dimension. A larger receptive field is needed for large objects in order infer correspondences from a more global semantic context (see left-hand side first and second row.).

D. Alignment Error Analysis

Our alignment results have different sensibility for each parameter block (translation, rotation, scale). In order to gauge the stringency of each parameter block we varied the threshold for one parameter block and held the other two constant at the default value (see Fig. 7). We observe that for the default thresholds $\epsilon_t = 0.2\text{m}, \epsilon_r = 20°, \epsilon_s = 20\%$ all thresholds

E. Alignment Algorithm Details

In order to remove misaligned objects, we prune objects after the alignment optimization based on the known free space of the given input scan. This is particularly important for the unconstrained ('in-the-wild') scenario where the set of ground truth CAD models to be aligned is not given as part of the input. For a given candidate transformation T_m (as described in Sec. 6 in the main paper), we compute:

$$c = \frac{\sum_x^{\Omega_{\text{CAD}}^{\text{occupied}}} \mathcal{O}_{\text{scan}}^{\text{seen}}(T_{\text{world}\to\text{vox,scan}} \cdot T_m^{-1} \cdot T_{\text{vox}\to\text{world,CAD}} \cdot x)^2}{|\Omega_{\text{CAD}}^{\text{occupied}}|}$$

$$\Omega_{\text{CAD}}^{\text{occupied}} = \{x \in \Omega_{\text{CAD}} \mid \mathcal{O}_{\text{CAD}}(x) < 1\}$$

$$\Omega_{\text{scan}}^{\text{seen}} = \{x \in \Omega_{\text{scan}} \mid \mathcal{O}_{\text{scan}}(x) > -\tau\}$$

$$\mathcal{O}_{\text{scan}}^{\text{seen}}(x) = \mathcal{O}_{\text{scan}}(x) \text{ if } x \in \Omega_{\text{scan}}^{\text{seen}} \text{ else } 0$$

where T_m^{-1} defines the transformation from CAD to scan, Ω defines a voxel grid space ($\subset \mathbb{N}^3$), τ is the truncation distance used in volumetric fusion (we use $\tau = 15\text{cm}$), and $\mathcal{O}$ are look-ups into the signed distance function or distance functions for the scan or CAD model. We also require that at least 30% of the CAD surface voxels $\Omega_{\text{CAD}}^{\text{occupied}}$ project into seen space of the scan voxel grid $\Omega_{\text{scan}}^{\text{seen}}$. Finally, we rank all alignments (of various models) per scene w.r.t. their confidence and prune all lower ranked models that are closer than 0.3m to a higher ranked model.

F. Alignment Optimization Analysis: Comparison to RANSAC

In Tab. 1, we additionally demonstrate the efficacy of our new alignment approach compared to alignment by RANSAC (using our predicted heatmap correspondences). Our alignment via heatmap optimization is more robust to outliers while also incorporating symmetries, resulting in significantly improved performance.

Method	avg. acc. in %
Our Heatmap CNN + RANSAC	18.27
Our Heatmap CNN + Heatmap optim.	31.68

Table 1: Our heatmap optimization for alignment in comparison to RANSAC. The input correspondences for RANSAC are provided by the maximum response of the predicted heatmap.

G. Baseline Method Details

In the following, we provide additional details for the used baseline approaches. FPFH and SHOT work on point clouds and compute geometric properties between points within a support region around a keypoint. We use the implementation provided in the Point Cloud Library [2].

The method presented by Li *et al.* [1] takes the free space around a keypoint into account to compute a descriptor distance between a keypoint in scan and another keypoint in a CAD object. Here, we use the original implementation from the authors and modified it such that it works within a consistent evaluation framework together with the other methods. However, since we are not restricted to real-time constraints, we neglect the computation of the geometric primitives around the keypoints, which helps to find good initial rotation estimations. Instead, we computed all 36 rotation variants to find the smallest distance. We also replace the original 1-point RANSAC with another RANSAC as described below.

3DMatch [3] takes as input a 3D volumetric patch from a TDF around a keypoint and computes via a series of 3D convolutions and max-poolings a 512 dimensional feature vector. In order to train 3DMatch, we assemble a correspondence dataset as described in Sec. 5.3 in the main paper. We train the network for 25 epochs using the original contrastive loss with a margin of 1. During test time,

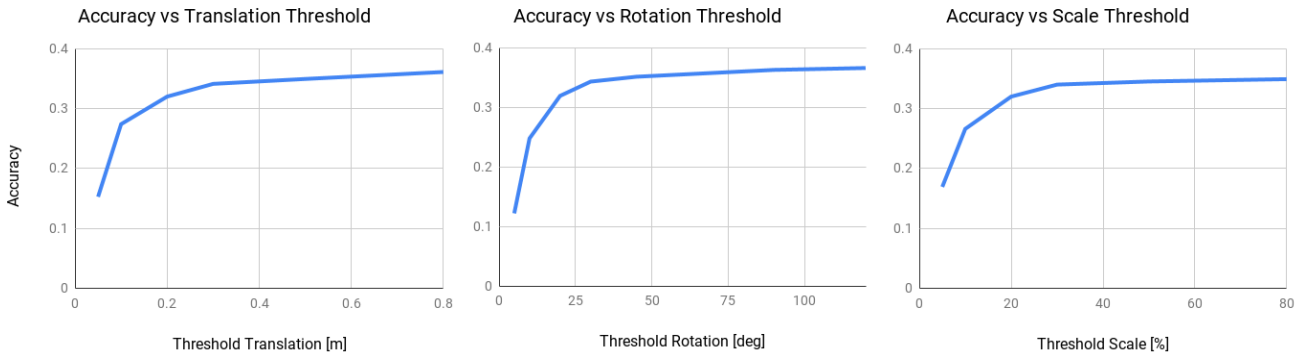

Figure 7: Accuracy vs. varying thresholds for translation (left), rotation (middle) and scale (right). Only one threshold is varied whereas the remaining ones were held constant at their default value either $\epsilon_t = 0.2\text{m}, \epsilon_r = 20^\circ, \epsilon_s = 20\%$.

we extract the 3D patch around a detected Harris keypoint of both CAD object and scan and separately compute their feature vector. In addition to the evaluation in the main paper, for 3DMatch, we additionally show the performance of 3DMatch when trained only on real only (scan-scan correspondences from ScanNet), as shown in Tab. 2. This suffers dramatically in matching the different characteristics of scan-CAD at test time. Our approach to predict scan-CAD heatmap correspondences results in significantly higher alignment accuracy compared to both 3DMatch trained on scan-CAD as well as scan-scan.

For each method, we compute the feature descriptors for all keypoints in the scan and the CAD objects, respectively. We then find correspondences between pairs of keypoints if their height difference is less than $0.8m$ and if the L2 distance between the descriptors is below a certain threshold. Due to potential re-occurring structures in scan and CAD we select the top-8 correspondences with the smallest descriptor distances for each keypoint in the scan.

After establishing potential correspondences between the scan and a CAD object, we use a RANSAC outlier rejection method to filter out wrong correspondences and find a suitable transformation to align the CAD object within the scene. During each RANSAC iteration, we estimate the translation parameters and the up-right rotation by selecting 3 random correspondences. If the transformation estimate gives a higher number of inliers than previous estimates, we keep this transformation. The threshold of the Euclidean distance for which a correspondence is considered as an inlier is set to $0.20m$. We use a fixed scale determined by the class average scale from our Scan2CAD train set. For a given registration for a specific CAD model, we mark off all keypoints in the scan which were considered as inliers as well as all scan keypoints which are located inside the bounding box of the aligned CAD model. These marked keypoints will be ignored for the registration of later CAD models.

To find optimal parameter for FPFH, SHOT, and Li *et al.*, we construct an additional correspondence benchmark and ran a hyperparameter search based on the validation set.

Method	avg. acc. in %
3D Match + ScanNet (only real data)	0.26
3D Match + our dataset	10.29
Our method + our dataset	31.68

Table 2: Comparison to 3DMatch trained with only real data, trained on our data, and our result; evaluation on our test set.

References

[1] Y. Li, A. Dai, L. Guibas, and M. Nießner. Database-assisted object retrieval for real-time 3D reconstruction. In *Computer Graphics Forum*, volume 34, pages 435–446. Wiley Online Library, 2015.

[2] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *Robotics and automation (ICRA), 2011 IEEE International Conference on*, pages 1–4. IEEE, 2011.

[3] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 199–208. IEEE, 2017.

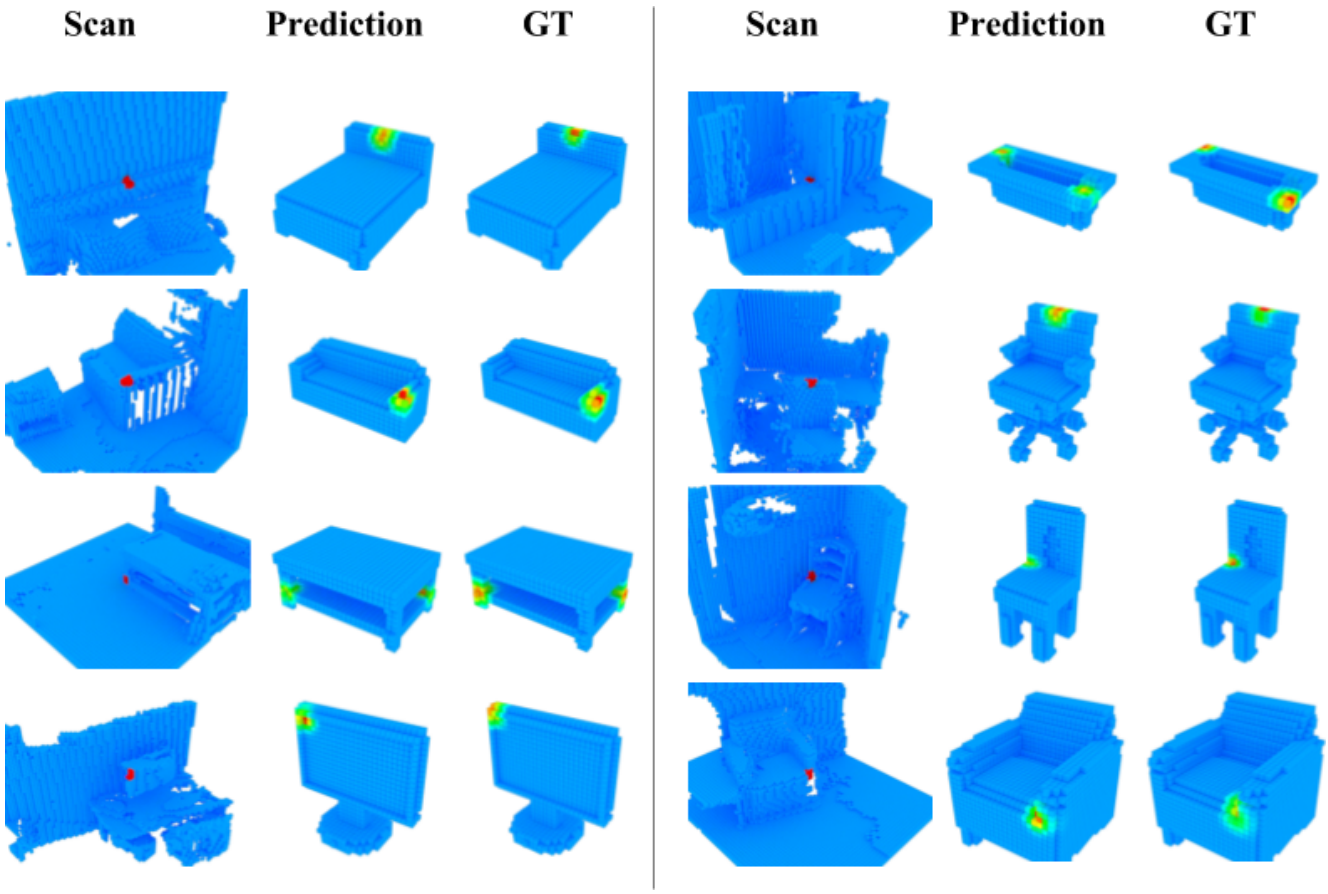

Figure 8: Sample correspondence predictions over a range of various CAD models. Heatmaps contain symmetry-equivalent correspondences.

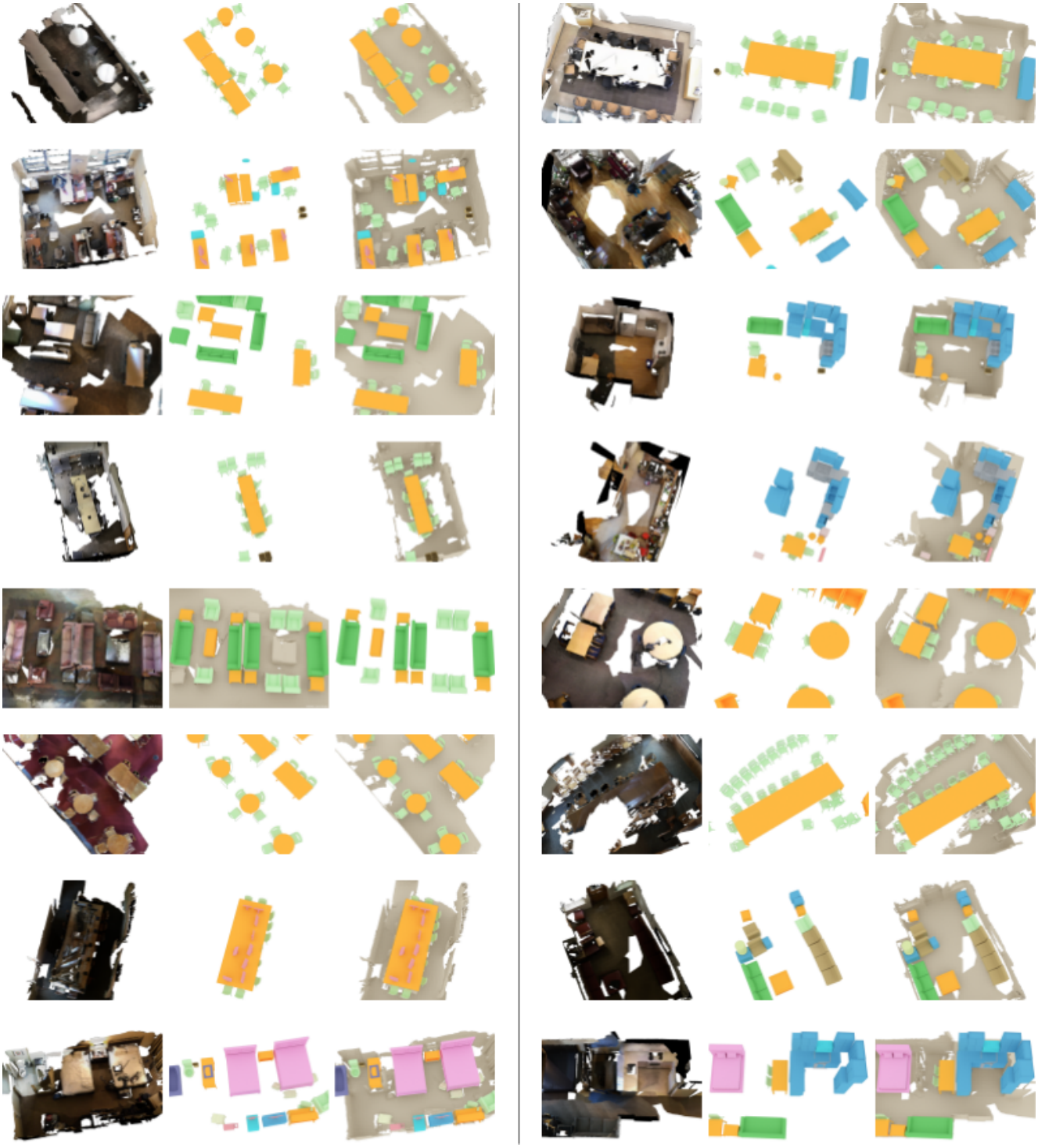

Figure 9: Samples of annotated scenes. Left: 3D scan. Center: annotated CAD model arrangement; right: overlay CAD models onto scan.