Supplementary Material
SGAligner: 3D Scene Alignment with Scene Graphs

Sayan Deb Sarkar [1], Ondrej Miksik [2], Marc Pollefeys [1,2], Daniel Barath [1], Iro Armeni [1]

[1]Department of Computer Science, ETH Zurich, Switzerland
[2]Microsoft Mixed Reality & AI Lab, Zurich, Switzerland
sgaligner.github.io

Abstract

In the supplemental material, we provide additional details about the following:

1. *Visualisation on point cloud mosaicking given multiple individual observations (Section A),*
2. *Comparison to a retrieval-based approach (Section B),*
3. *Additional ablation on SGAligner to further understand the performance of node matching (Section C),*
4. *Information on the SGAligner benchmark, including details on the generated data, evaluation protocol, and metrics (Section D), and*
5. *Details on implementation (Section E).*

A. Application: Point Cloud Mosaicking

In Section 4.3 of the main paper, we demonstrate the potential of **SGAligner** on 3D point cloud mosaicking. Here, two success cases are shown in Figure 1 and a failure in Figure 2 (the graphs are shown simplified for visualisation purposes and do not represent the entire available graph).

B. Application: Finding Overlapping Scenes

In the main paper, we discussed that **SGAligner** provides less than $O(N^2)$ computation complexity when addressing the task of registering multiple 3D scenes for which we have no knowledge of whether they overlap or not. Another approach to avoid full registration on all pairs (standard registration methods), is to use a retrieval-based approach. We consider the following approach as baseline: (i) extract local 3D keypoints for all available 3D point clouds [15]; (ii) generate a 3D descriptor per extracted keypoint [9]; (iii) accumulate the 3D keypoint descriptors into a global descriptor for each point cloud [4], and (iii) perform kNN search to rank global descriptors based on the queried one. Similarly here, this experiment serves as a demonstration of the potential of **SGAligner** and does not aim to solve the task.

Specifically, given a point cloud, we extract keypoints from the entire scene based purely on geometry and without any notion of object-ness or semantics. We randomly select 500 keypoints and their descriptors per scene, which we use to train [4] so as to generate optimized global descriptors. During inference and given a query point cloud and its corresponding global descriptor, we perform a kNN search to identify the closest neighbors of it in the rest of the point clouds. We evaluate on Mean Reciprocal Rank (MRR) and compare with **SGAligner**.

Results are shown in Table 1. We evaluate on different point cloud densities, ranging from using the full point cloud density offered in [10] to random subsampling for 10, 20, 30, and 50%. Please note that in **SGAligner**, we do not use the entire scene, only objects in the scene graph. As described in the main paper, we downsample object point clouds using farthest point sampling to 512 points. We follow the same protocol here and perform this operation per subsampled scene level.

As expected, the retrieval-based method is performing well when there is a dense point cloud, since our method employs a limited amount of points per object instance. However, VLAD+KNN cannot retain a robust performance when density decreases, already reaching lower performance than **SGAligner** at 10% subsampling. In contrast, our approach is barely affected by a changing density since it already operates on lower-resolution point clouds. This showcases that the topological information encoded in 3D scene graphs can lead to more robust results when dealing with common failure cases in global descriptors (*i.e.*, changes in point cloud density).

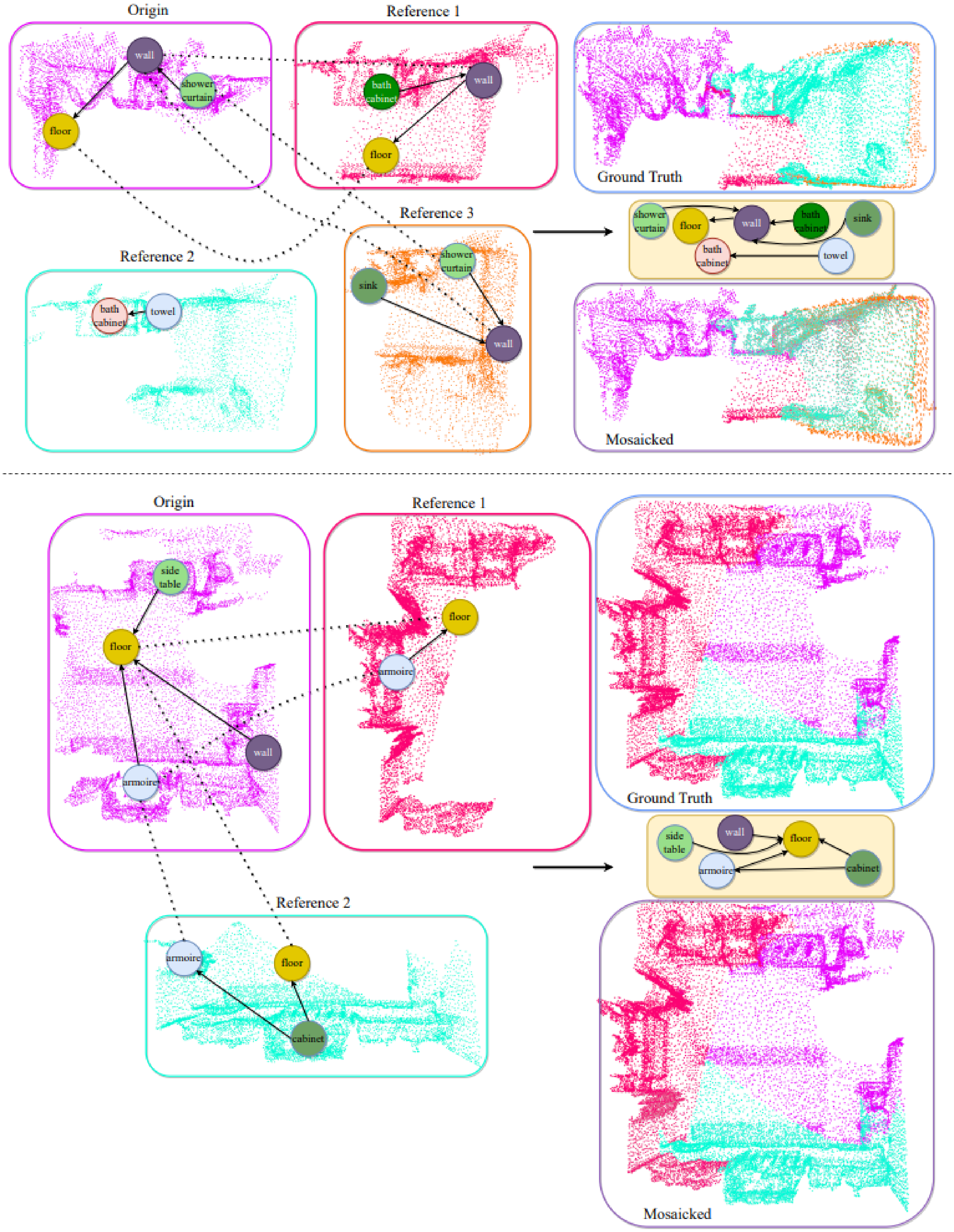


Figure 1. **Qualitative Results on Point Cloud Mosaicking.** Given partial point clouds of a scene and the corresponding 3D scene graphs, we showcase two example results on Point Cloud Mosaicking and the creation of a unified scene graph using the methodology discussed in Section A. The solid lines show the relationships between objects and dashed lines represent the ground truth entity pairs $\mathcal{F}$.

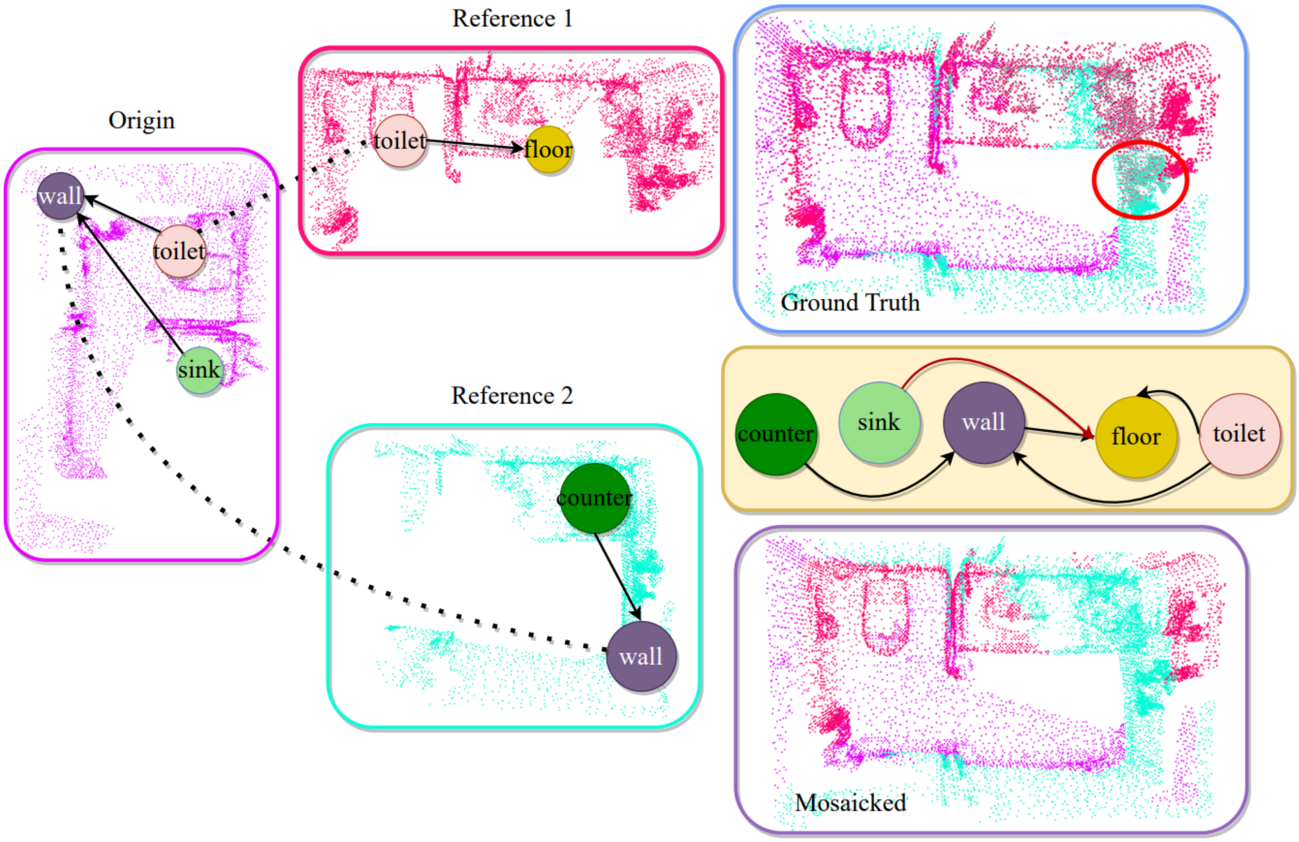


Figure 2. **Failure on Point Cloud Mosaicking.** Similar to the success cases in Fig. 1, we also showcase a failure case of our approach on point cloud mosaicking. The solid lines show the relationships between objects and dashed lines represent the ground truth entity pairs $\mathcal{F}$. The red circle on the ground truth point cloud depicts the area where the failure is the most visible and red arrow demonstrates the misalignment of **SGAligner** between a *sink* and a *floor*.

Subsampling %	**VLAD + KNN**	**SGAligner**
0	**<u>0.557</u>**	0.383
10	0.316	**0.356**
20	0.276	**0.343**
30	0.222	**0.339**
50	0.162	**0.312**

Table 1. **Mean Reciprocal Rank (↑) comparison with a retrieval-based approach.** *Best* results per subsampling level are in **bold**. *Overall best* in **<u>underlined bold</u>**.

C. Additional Ablation Studies

C.1. Analysis with Various Object Encoders

In our experiments, we choose PointNet [7] as our encoder because it is commonly employed in most scene graph methods [11], [13], [14]. Pointnet has been shown to perform in real-time scenarios [1], which makes it suitable for mobile robot applications. In Table 2, we present a comparison of object encoders, on the node matching task. Point Cloud Transformer (PCT) [2], being inherently permutation invariant to an unordered point cloud, shows an improvement in the metrics. These results also showcase that our method is robust and agnostic to the 3D visual encoder.

Encoder	**Mean**	**Hits @ ↑**				
	RR ↑	**K = 1**	**K = 2**	**K = 3**	**K = 4**	**K = 5**
PointNet [7]	0.950	0.923	0.957	0.974	0.982	0.987
PCT [2]	**0.965**	**0.947**	**0.968**	**0.983**	**0.988**	**0.991**

Table 2. **Comparison on node matching of SGAligner using various object encoders.** Best values are in **bold**.

C.2. Intra-Graph Alignment Recall

To further validate how our model performs on aligning nodes between two 3D scene graphs (source-reference) with no/partial overlap, we formulate *Intra-Graph Alignment Recall (IGAR)* metric. It measures what fraction of the nodes in the source graph are aligned (K=1), with nodes in the same source graph or, in other words, how many node matches out of total are self-aligned. We provide these results in Table 3. We do not explicitly model **not** self-matching nodes within the same graph, yet, *IGAR* values stand to show that our method rarely performs this.

$$IGAR = \frac{1}{M}\sum_{M}\frac{|pred\{n^i \equiv n^j\}|}{|\mathcal{F}|}, n \in \mathcal{N} \qquad (1)$$

where, $i \neq j$, $pred\{n^i \equiv n^j\}$ is the set of nodes in the graph which **SGAligner** aligned with nodes in the same graph, $\mathcal{F}$ is the set of ground truth anchor pairs and $\mathcal{N}$ denotes the set of objects in a single graph and M is the total number of graphs.

Method	IGAR ↓ (%)
$\mathcal{P}$	16.9
$\mathcal{P}+\mathcal{S}$	16.5
$\mathcal{P}+\mathcal{S}+\mathcal{R}$	13.1
SGAligner	**8.2**

Table 3. **Evaluation on node self-alignment. SGAligner** has not been explicitly modeled to not create self-matches but still is able to differentiate between nodes from the same and different graphs.

C.3. Confusion Matrix

We compute a confusion matrix to identify which object categories are most frequently misaligned during entity alignment and if our method fails on certain semantic classes (*e.g.*, *chair*, *table*, etc). In Figure 3, we show the confusion matrix on all 4 module combinations of **SGAligner**. As expected, the object encoder module $\mathcal{P}$, although performing well, confuses the most the *wall* and *floor* classes. This is due to the fact that purely on a semantic level, without encoding any positional/structural information, these classes are similar. We can further observe that **SGAligner** is robust to certain classes like *pillow*, *tv*, *lamp*, etc. Classes like *wall*, *floor*, and *fridge* are the ones easily susceptible to misalignment on the tested dataset, albeit less than in $\mathcal{P}$.

C.4. Robustness to Missing Geometric Information

In this section, we provide an ablation study of **SGAligner** on the 3D Scene Graph alignment task and evaluate how it performs on node alignment, when all the **geometric relationships** encoding positional information between the nodes such as `left` and `standing on` are missing. Results are in Table 4. As expected, the structure module $\mathcal{S}$ suffers from this compared to the full ground-truth experiment, since the number of edges encoded in the neighborhood of an entity gets reduced. However, overall, our method does not show a drastic drop in node alignment metrics due to the absence of geometric relationships. This can be attributed to the fact that we do not discriminate between different types of relationships in our encoders, however, this is a very important robustness characteristic, especially, while working with predicted scene graphs where the relationships could be missing or incorrectly labelled. This also shows that once trained with full ground truth, our method is able to handle missing data during inference which would be useful for a navigation agent.

Modalities	Mean RR ↑	Hits @ ↑ K = 1	K = 2	K = 3	K = 4	K = 5
$\mathcal{P}$	0.884	0.835	0.886	0.921	0.938	0.951
$\mathcal{P}+\mathcal{S}$	0.880	0.830	0.882	0.918	0.936	0.948
$\mathcal{P}+\mathcal{S}+\mathcal{R}$	0.893	0.844	0.898	0.933	0.949	0.959
SGAligner	**0.948**	**0.921**	**0.952**	**0.971**	**0.979**	**0.985**

Table 4. **Evaluation on node matching.** We compare the performance of **SGAligner** for different modality combinations, when all geometric edges are missing.

D. SGAligner Benchmark

In this section, we offer qualitative explanations on our dataset generation procedure and discuss the evaluation metrics used to asses performance with respect to the various tasks we reported.

D.1. Dataset

In Sec. 4 of the main paper, we provide a description of the data generation procedure. In Figure 4, we report statistics on the spatial overlap of the generated pairs. We show examples of sub-scenes generated using this approach in Figure 5, alongside the camera trajectory used to capture the corresponding scan in [10]. We will make our code and benchmark public.

D.2. Evaluation Metrics

The evaluation metrics that we use to assess performance in Section 4.1 of the main paper, as well as in this supplementary material, are formally defined in this section.

D.2.1 Alignment Metrics

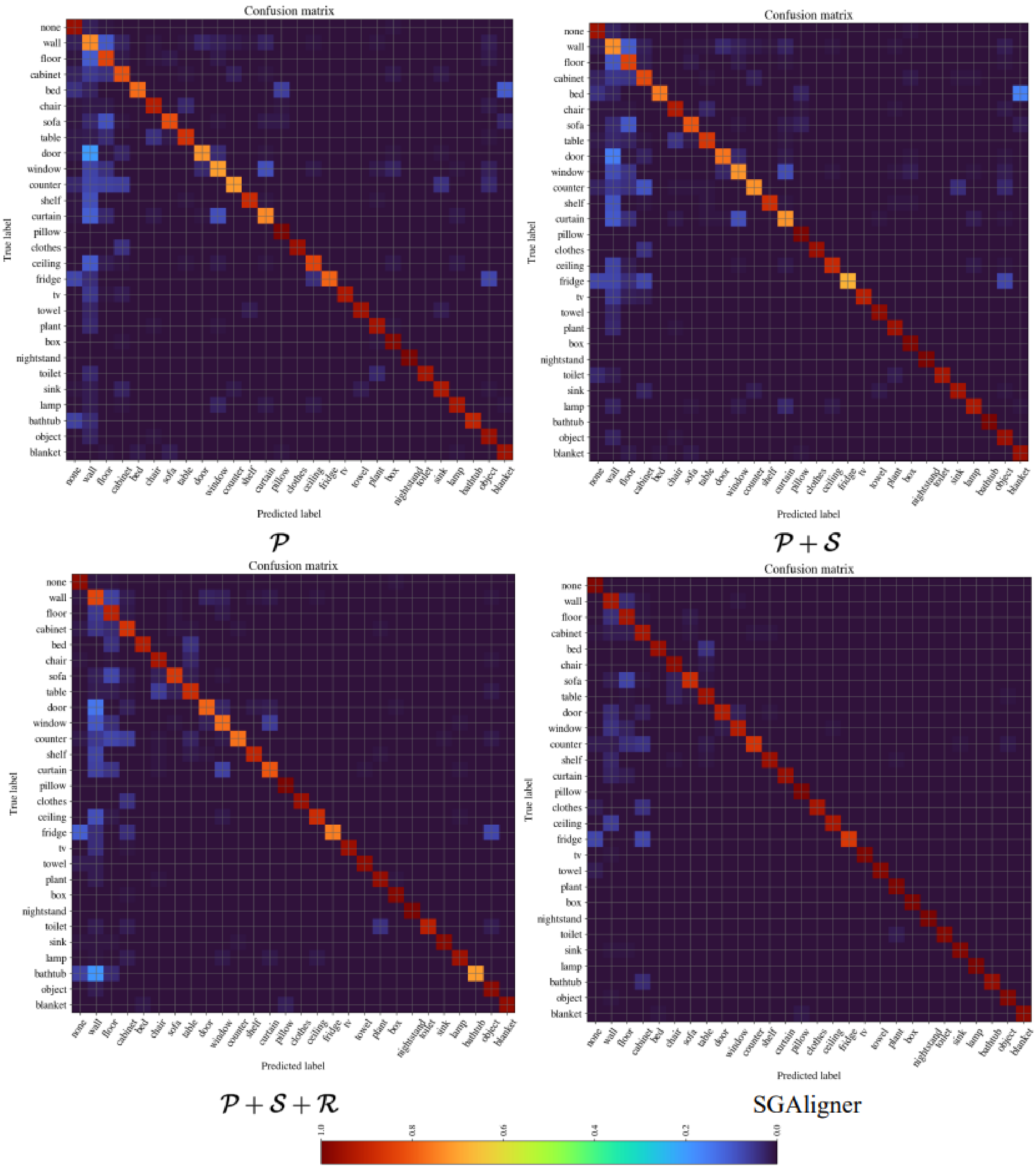
Inspired by works in multi-modal entity alignment [5], we define our alignment metrics as follows :

Hits @ K represents the fraction of true anchor entities present in the top k predictions :

$$H_k(r_1, ..., r_n) = \frac{1}{n}\sum_{i=1}^{n} I[r_i \leq k] \qquad (2)$$

where, $I[x \leq y] = 1$ when $x \leq y$ else 0 and $k \in [1, 2, 3, 4, 5]$.

Mean Reciprocal Rank (MRR) corresponds to the arithmetic mean over the reciprocals of ranks of true triples.



$\mathcal{P}$

$\mathcal{P} + \mathcal{S}$

$\mathcal{P} + \mathcal{S} + \mathcal{R}$

SGAligner

Figure 3. **Object Confusion Matrices** of the 4 module combinations of **SGAligner**: object encoder ($\mathcal{P}$), object and structure encoders ($\mathcal{P} + \mathcal{S}$), object, structure and relationship encoders ($\mathcal{P} + \mathcal{S} + \mathcal{R}$), and the proposed method with all modules (**SGAligner**). High values indicate that an object (denoted on y-axis) is often recognized as the object denoted on x-axis – everything but the diagonal should be 0.

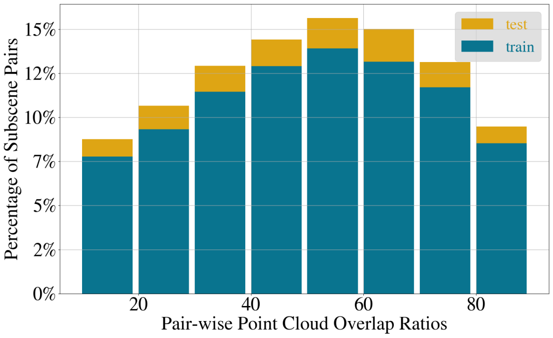


Figure 4. **Overlap statistics for the generated sub-scenes.**

$$MRR(r_1, ..., r_n) = \frac{1}{n} \sum_{i=1}^{n} r_i^{-1} \tag{3}$$

D.2.2 Registration Metrics

Feature Matching Recall (FMR) [3][8] measures the fraction of point cloud pairs for which, based on the number of inlier correspondences, it is likely that accurate transformation parameters can be recovered with a robust estimator such as RANSAC. It should be noted that FMR simply verifies whether the inlier ratio (IR) is higher than a threshold $\mathcal{T} = 0.05$. It does not examine if the transformation can actually be inferred from those correspondences, which is not always the case because of the possibility that their geometric arrangement is (almost) degenerate, such as when they are situated closely together or along a straight edge.

$$FMR = \frac{1}{M} \sum_{i=1}^{M} [\![IR_i > \mathcal{T}]\!] \tag{4}$$

where M is the number of all point cloud pairs.

Registration Recall (RR) is the fraction of registered point cloud pairs for which the transformation error is smaller than 0.2m. The transformation error is the root mean squared error of the ground truth correspondence $\mathcal{H}^*$ after applying the predicted transformation $\mathbf{T}_{P \to Q}$.

$$RMSE = \sqrt{\frac{1}{|\mathcal{H}^*|} \sum_{({p^*}_{x_i}, {q^*}_{y_i}) \in \mathcal{H}^*} ||T_{P \to Q}({p^*}_{x_i}) - {q^*}_{y_i}||_2^2} \tag{5}$$

$$RR = \frac{1}{M} \sum_{i=1} [\![RMSE_i < 0.2m]\!] \tag{6}$$

Relative Rotation Error (RRE) is the geodesic distance in degrees between estimated and ground-truth rotation matrices.

$$RRE = arccos(\frac{trace(R^T \cdot \bar{R} - 1)}{2}) \tag{7}$$

Relative Translation Error (RTE) is the the euclidean distance between estimated and ground-truth translation vectors.

$$RTE = ||t - \bar{t}|| \tag{8}$$

We compute mean RRE and RTE between all the registered point cloud pairs.

Chamfer Distance measures the quality of registration. Following [12], [3], we use the *modified* Chamfer distance metric :

$$\begin{aligned} \bar{CD}(P, Q) = \frac{1}{|P|} \sum_{p \in P} min_{q \in Q_{raw}} ||{T_P}^Q(p) - q||_2^2 + \\ \frac{1}{|Q|} \sum_{q \in Q} min_{p \in P_{raw}} ||q - {T_P}^Q(p)||_2^2 \end{aligned} \tag{9}$$

where, P_{raw} and Q_{raw} are $raw/clean$ source and target point clouds respectively.

D.2.3 Reconstruction Metrics

The definition of full 3D reconstruction metrics is provided in Table 5.

Metric	Definition
Acc	$mean_{p \in P}(min_{p^* \in P^*} \lVert p - p^* \rVert)$
Comp	$mean_{p^* \in P^*}(min_{p \in P} \lVert p - p^* \rVert)$
Precision	$mean_{p \in P}(min_{p^* \in P^*} \lVert p - p^* \rVert < 0.05)$
Recall	$mean_{p^* \in P^*}(min_{p \in P} \lVert p - p^* \rVert < 0.05)$
F1-Score	$\frac{2 * precision * recall}{precision + recall}$

Table 5. **3D Reconstruction Metric Definitions.** p and p^* are ground truth and predicted point clouds respectively.

E. Implementation Details

Inspired by MCLEA [5], we use a multi-layered GAT with 2 layers and each hidden unit being 128-dimensional. All the modules output a 100-dimensional embedding and the joint embedding, being a weighted concatenation, is 400-dimensional. We use $\mathcal{T}_1$ for ICL loss as 0.1 and $\mathcal{T}_2$ for IAL loss as 1.0. We train our model for 50 epochs on a NVIDIA GeForce RTX 3060 Ti 8GB GPU with a batch size of 4 using AdamW [6] optimizer and a learning rate of 0.001.

References

[1] Lin Bai, Yecheng Lyu, and Xinming Huang. Pointnet on fpga for real-time lidar point cloud processing. 10 2020. 3

[2] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R. Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199, Apr 2021. 3

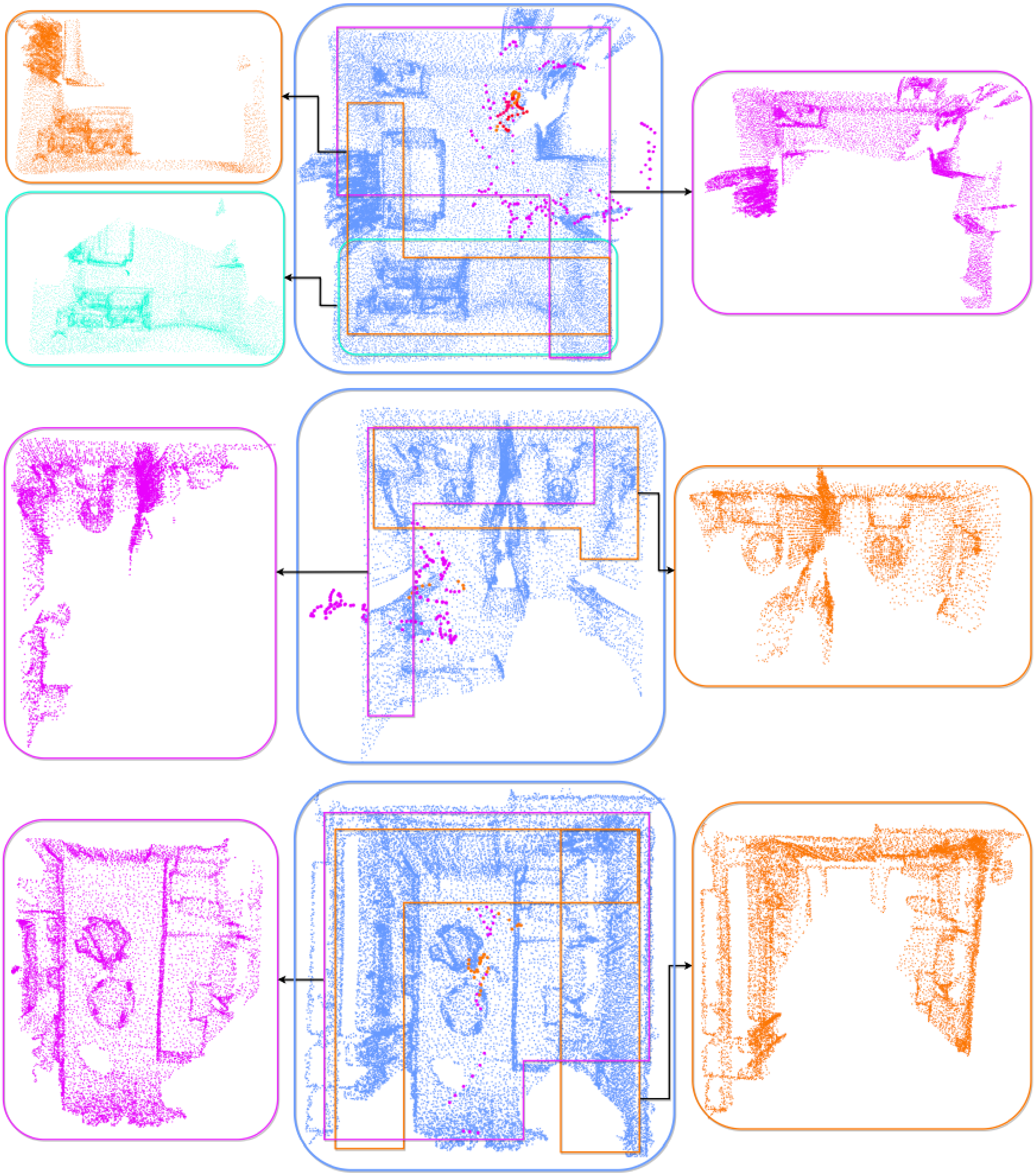


Figure 5. **Visualization of data generation process.** We visualise the creation of subscenes using our approach from Section 4 of the main paper. Given a point cloud from [10] (middle column), we create multiple sub-scenes (left and right columns) and showcase the used camera trajectory. The colors depict the camera trajectory and 3D spatial area of each sub-scene in the parent scene.

[3] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3d point clouds with low overlap. In *Proceedings of the IEEE/CVF conference on computer vision and pattern*

recognition, 2021. 6

[4] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE, 2010. 1

[5] Zhenxi Lin, Ziheng Zhang, Meng Wang, Yinghui Shi, Xian Wu, and Yefeng Zheng. Multi-modal contrastive representation learning for entity alignment. *arXiv preprint arXiv:2209.00891*, 2022. 4, 6

[6] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 6

[7] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 3

[8] Zheng Qin, Hao Yu, Changjian Wang, Yulan Guo, Yuxing Peng, and Kai Xu. Geometric transformer for fast and robust point cloud registration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022. 6

[9] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009. 1

[10] Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7658–7667, 2019. 1, 4, 7

[11] Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7515–7525, 2021. 3

[12] Zi Jian Yew and Gim Hee Lee. Rpm-net: Robust point matching using learned features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11824–11833, 2020. 6

[13] Chaoyi Zhang, Jianhui Yu, Yang Song, and Weidong Cai. Exploiting edge-oriented reasoning for 3d point-based scene graph analysis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9705–9715, June 2021. 3

[14] Shoulong Zhang, Shuai Li, Aimin Hao, and Hong Qin. Knowledge-inspired 3d scene graph prediction in point cloud. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 18620–18632, 2021. 3

[15] Yu Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*, pages 689–696. IEEE, 2009. 1