

ReScene4D: Temporally Consistent Semantic Instance Segmentation of Evolving Indoor 3D Scenes

Supplementary Material

In this supplementary material we provide:

- A video discussing our method and results (Section A)
- Visualizations of instance mask and semantic predictions (Section B)
- Additional experimental results and discussion (Section C)
- Additional implementation details (Section D)
- A discussion of the limitations of our method (Section E)

A. Video

Supplementary video available at <https://www.easteine.com/rescene4d/> summarizes the method, quantitative and qualitative results, and our temporal sharing ablation experiment.

B. Qualitative Results

We provide additional qualitative results showcasing instance masks and semantic predictions on several scenes from the 3RScan validation set [39]. Note that the 3RScan test set is not available for download, nor is there an evaluation server, so all results are reported on the validation split. For visualization, we project the point cloud predictions onto the original mesh vertices.

We visualize the instance predictions from: Mask4Former [44]; Mask3D [32] with post-hoc temporal semantic matching; and our best-performing model, ReScene4D (Concerto with ST-serialization and contrastive loss) in Figures 4, 5, 6, 10, 11, 12, 13, 15, 14, 16, and 17. We omit Mask4D [27] due to its poor performance, and we exclude Mask3D with geometric matching, as semantic matching provides a better balance between overall t-mAP and per-stage mAP. While geometric matching can reliably track well-aligned static instances, it struggles in sequences with partial overlap or instance changes, often degrading mask quality in the second stage. Note, instance colors are assigned independently for each method and do not correspond between methods or to ground truth; colors indicate distinct instances in instance segmentation, consistent across the temporal sequence to show shared identities.

For semantic segmentation, we visualize results using semantic category colors defined by ScanNet [8]. These colors are consistent across ground truth and all methods, ensuring direct visual comparability of semantic predictions. Semantic predictions are presented in Figures 10 and 11. Across methods, semantic segmentation performance remains comparable, with similar difficulties arising for chal-

lenging classes such as pillows and other small objects. The challenge of the 3RScan dataset is evident in semantic predictions for all approaches, including ours, and in the degraded 3DSIS baseline mAP for single (non-temporal) 3D scans (see Figures 11 and 16). For example, Mask3D (without posthoc processing) achieves a validation mAP of 55.2 on ScanNet [8], but only 46.4 on 3RScan. This drop likely reflects the limited semantic diversity, increased noise, and smaller scale of 3RScan.

C. Additional Experimental Results

In this section, we further ablate the feature encoder and temporal sharing.

C.1. Temporal Sharing Experiments Details

Table 5 compares t-mAP for Concerto (Conc.), Sonata (Son.), and Minkowski (Mink.) under different temporal sharing strategies. As discussed below, the optimal strategy differs across backbones. Table 6, 7, and 8 present the full ablation of each backbone reporting the t-mAP and t-REC (mean and per change type). In addition, to complement our core evaluation with t-mAP and t-REC, we report and analyze the mean inter-stage instance feature cosine similarity, or t-sim, across our experiments. t-sim measures, for each instance that is present in multiple temporal scans, the average similarity of its internal feature representation between paired stages. Unlike mAP or recall, which are defined by mask matches and ground truth, t-sim quantifies the temporal consistency of the learned instance embeddings independent of downstream query refinement or matching success.

Concerto. Table 6 reports the full results of our Concerto backbone ablation study, discussed in Section 5 of the main paper, including additional change labels for static and added/removed instances. Figure 15 provides a visual comparison for the base spatio-temporal architecture and our best performing version.

- *Static instances:* Temporal recall (t-REC) for static objects is strongly correlated with overall t-mAP, as static objects are much more prevalent in the dataset and thus dominate evaluation metrics. As with overall mAP, each temporal information sharing strategy improves convergence for static objects—whether by bringing feature representations closer together (contrastive loss), enabling joint feature decoding, or aligning spatio-temporal masks. However, combining multiple strategies leads to diminishing returns.

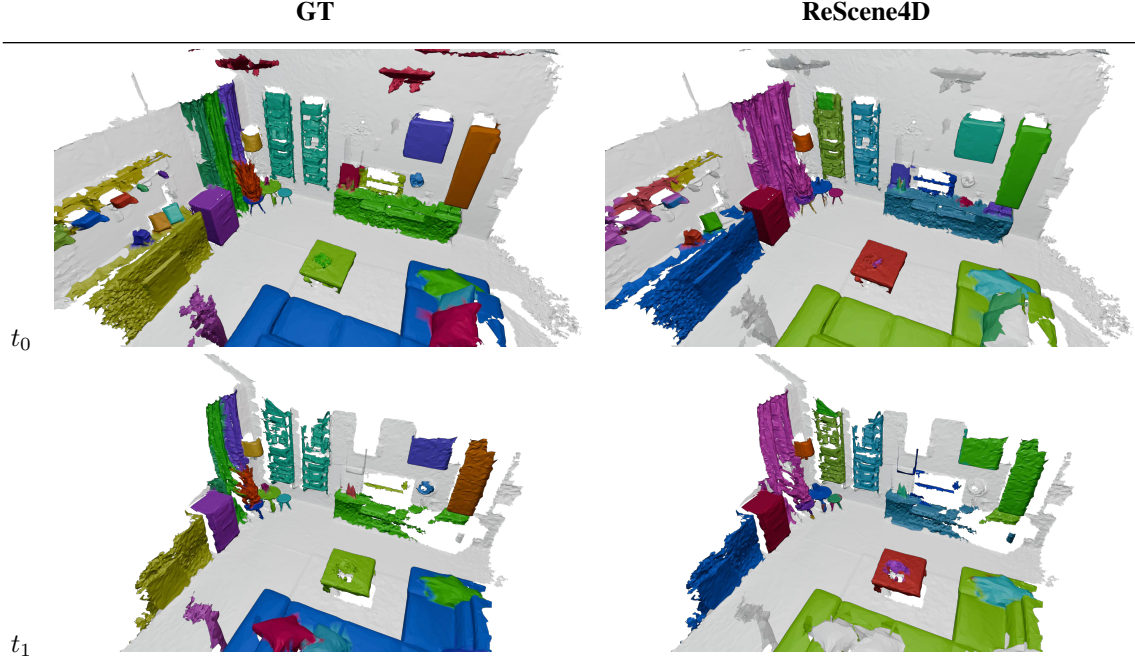


Figure 4. **Temporal Semantic Instance Segmentation Comparison.**

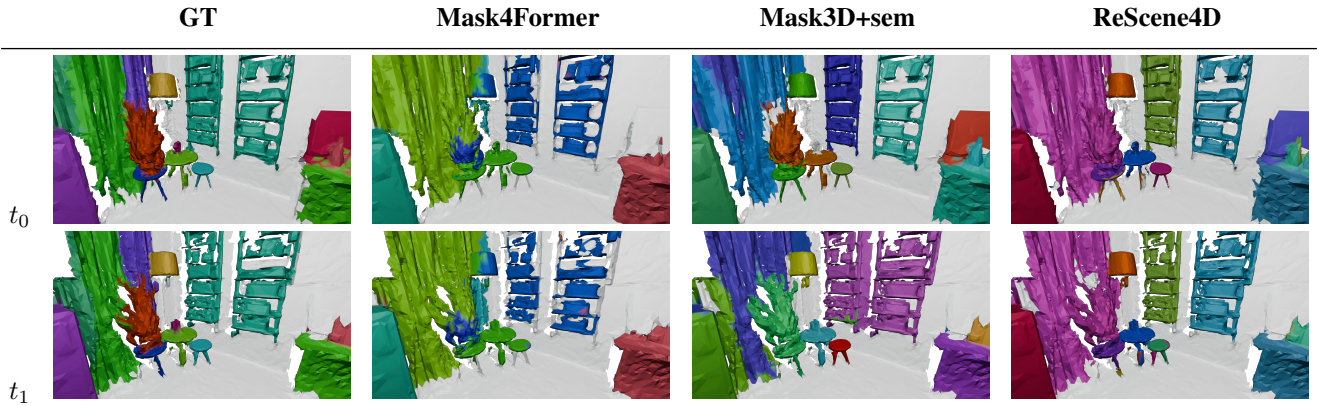


Figure 5. **Temporal Semantic Instance Segmentation Comparison—Close up of Figure 4 with baseline comparison.** While Mask4Former has a tendency to merge multiple objects into one instance and Mask3D fails to track identities across time, ReScene4D consistently identifies and tracks object instances across temporal scans, be that dynamic or static. It even considers the two bookcases next to the curtains as separate instances, in contrast to ground truth annotations. Non-systematic annotation of small objects throughout the dataset leads to our method missing some of the pillows on the couch, as well as the identification of a box on the top shelf of the bookcase as a separate instance.

- *Added and Removed instances:* We group added and removed objects together, as both represent instances present in one stage and absent in another. Due to very few such examples in the dataset (1.5% of instances in the validation set), this metric is sensitive to noise and initialization and may lack enough signal for models to learn to identify removals effectively. Performance for added/removed is lower with the Concerto backbone compared to Sonata and Minkowski; improvements in

static recall often come at the cost of added/removed recall (failure to produce temporal correspondences will lead to perfect removal prediction).

Sonata. Table 7 presents the full results of our Sonata backbone ablation study. Sonata achieves its highest t-mAP, t-mREC, and per-change-type performance when spatio-temporal masking is combined with spatio-temporal serialization. Serialization enables joint decoding of features

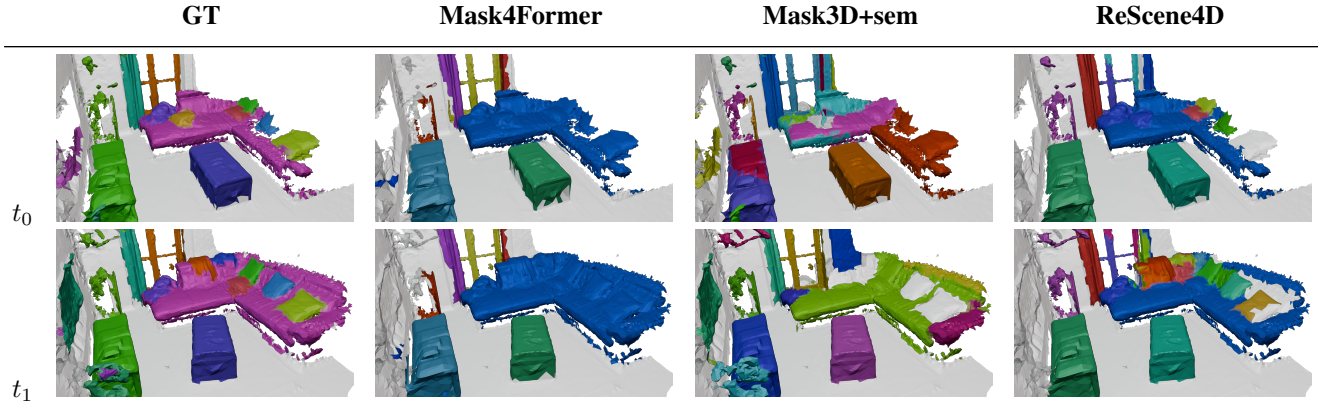


Figure 6. **Temporal Semantic Instance Segmentation Comparison—Common Baseline Errors.** Mask4Former maintains consistent identities across temporal stages but tends to merge objects—notably the couch and all pillows. Mask3D fails to track identities and has difficulty producing accurate instance masks in the first stage, where the couch is only partially observed. In contrast, ReScene4D correctly preserves identities despite bench movement and successfully segments most pillows, despite challenging non-rigid changes and partial observations.

L_{contr}	ST-serial	ST-mask	Conc.	Son.	Mink.
×	×	×	28.4	29.7	32.0
✓	×	×	34.1	25.7	31.5
×	✓	×	32.9	28.1	—
×	×	✓	32.4	28.3	30.6
✓	✓	×	34.8	29.6	—
✓	×	✓	33.2	27.8	32.0
×	✓	✓	32.5	33.2	—
✓	✓	✓	33.3	26.2	—

Table 5. **Temporal instance sharing ablations with different encoders for t-mAP.** Different encoders benefit from different strategies, depending on their strengths—e.g., contrastive loss harms Sonata since the features are mainly geometric, and Minkowski does not allow serialization as a ConvNet. Best result per encoder in **green**, baseline in **blue**.

across time, improving categories such as static and ambiguous. Masking alone enforces temporal consistency in instance masks during query refinement but does not explicitly promote feature alignment between temporal stages. When paired with spatio-temporal serialization, this feature alignment mechanism leads to more comprehensive improvements across change types.

In contrast to Concerto, contrastive loss significantly reduces performance in Sonata ablations. This emphasizes the dependence of temporal sharing strategies on backbone feature representations. Figure 7 illustrates this: a PCA visualization of point cloud features across the two different backbones (without temporal sharing modifications) shows Sonata features are primarily geometry-driven, and sensitive to global location. For large instances, substantial variation in point features within the same object may ob-

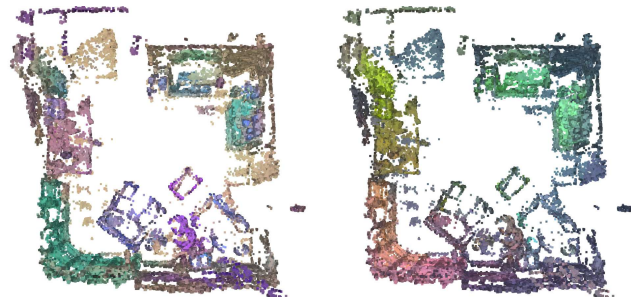


Figure 7. **Feature PCA for a sample scene per PTV3 encoder.** Left — *Concerto*, Right — *Sonata*.

scure the contrastive learning signal, potentially hindering effective temporal alignment. By contrast, Concerto generates more semantically oriented features. In this setting, contrastive loss aids discrimination between visually or semantically similar objects (e.g., ambiguous classes or varied instances such as different types of pillows), resulting in clear feature separation across time. Notably, with naturally strong geometric consistency, the Sonata canonical has better performance over the Concerto baseline, as its geometry-based features are stable across temporal stages and well-separated among instances within a scene. However, Concerto benefits more from temporal sharing and ultimately shows larger relative gains when these strategies are applied.

Minkowski. Table 8 presents the complete results of our Minkowski backbone ablations. Minkowski shows the least improvement over its canonical model—largely because its canonical version outperforms those of other backbones.

L_{contr}	ST-serial	ST-mask	t-mAP	t-Sim	t-mREC	Amb.	Rig.	Non-Rig.	Static	Add. & Rem.
×	×	×	28.4	0.8640	41.8	20.4	44.9	62.1	40.0	21.6
✓	×	×	34.1	<u>0.9160</u>	49.6	42.8	48.4	63.2	43.9	18.8
×	✓	×	32.9	0.8846	48.8	43.2	40.9	67.0	<u>44.0</u>	15.7
×	×	✓	32.4	0.8797	48.5	42.3	40.2	70.7	40.8	17.9
✓	✓	×	34.8	0.9198	52.1	<u>47.2</u>	48.6	66.5	46.1	<u>21.7</u>
✓	×	✓	33.2	0.9116	53.5	50.2	<u>54.4</u>	65.7	43.6	26.5
×	✓	✓	32.5	0.8980	50.4	43.2	50.0	65.1	43.1	17.5
✓	✓	✓	33.3	0.9198	<u>53.0</u>	43.9	56.4	<u>68.0</u>	43.7	16.2

Table 6. **Concerto Ablation of temporal information sharing strategies in 4DSIS.** Rows show combinations of contrastive loss (L_{contr}), spatio-temporal serialization (ST-serial), and spatio-temporal masking (ST-mask), with t-mAP and temporal recall (t-mREC) reported for ambiguous, rigid, non-rigid, static, and added/removed instances. Best overall t-mAP is in **green**, best per metric in **bold**, second best per metric underlined, baseline in **blue**, and **gray** denotes combinations that do not exceed individual method performance.

L_{contr}	ST-serial	ST-mask	t-mAP	t-Sim	t-mREC	Amb.	Rig.	Non-Rig.	Static	Add. & Rem.
×	×	×	29.7	0.8053	45.3	30.9	<u>49.2</u>	<u>58.9</u>	42.3	<u>33.4</u>
✓	×	×	25.7	0.8767	42.1	30.0	43.7	55.6	39.3	21.3
×	✓	×	28.1	0.8949	<u>46.1</u>	<u>39.3</u>	49.0	53.4	<u>42.5</u>	26.3
×	×	✓	28.3	0.8713	44.6	26.4	52.9	57.8	41.3	28.2
✓	✓	×	29.6	0.8811	44.8	32.4	48.6	56.8	41.5	31.5
✓	×	✓	27.8	0.8708	42.3	31.5	52.1	48.0	39.8	26.4
×	✓	✓	33.2	<u>0.8885</u>	52.3	44.6	56.1	65.2	43.2	29.0
✓	✓	✓	26.2	0.8791	39.5	20.7	51.0	48.8	37.6	35.7

Table 7. **Sonata Ablation of temporal information sharing strategies in 4DSIS.** Rows show combinations of contrastive loss (L_{contr}), spatio-temporal serialization (ST-serial), and spatio-temporal masking (ST-mask), with t-mAP and temporal recall (t-mREC) reported for ambiguous, rigid, non-rigid, static, and added/removed instances. Best overall t-mAP is in **green**, best per metric in **bold**, second best per metric underlined, baseline in **blue**, and **gray** denotes combinations that do not exceed individual method performance.

This is expected: it is the only encoder trained directly on this task. However, training the backbone from scratch prioritizes learning 3D feature representations and adapting to the dataset distribution, resulting in most improvements stemming from spatial learning rather than effective use of temporal information. The key takeaway is that joint spatio-temporal query refinement—central to all of our approaches—provides a substantial advantage over existing models such as Mask3D and Mask4Former.

Individually, both contrastive loss and spatio-temporal masking moderately enhance performance for certain change types, particularly in t-mREC. When combined, these strategies yield gains for ambiguous, rigid, and non-rigid categories compared to the canonical model. With full temporal information sharing, Minkowski maintains overall t-mAP while improving recall across change types, which better reflects performance relative to the frequency of changes in the dataset.

Analysis of the t-Sim Metric. As expected, temporal information sharing strategies consistently improve t-sim over

the canonical backbone for all encoders (see Tables 6, 7, 8), though the magnitude and baseline differ:

- **Concerto:** Displays a substantial increase in t-sim when temporal sharing mechanisms are used, with the best temporal consistency from the combination of contrastive loss and serialization. These settings also correspond to strong t-REC and recall for rigid instances.
- **Sonata:** Shows the largest improvement in t-sim between base and sharing variants. For Sonata, t-sim most clearly identifies serialization as the best mechanism for temporal sharing with respect to temporal feature consistency.
- **Minkowski:** For Minkowski, temporal sharing mechanisms yield smaller absolute gains in t-sim compared to the other backbones but remain beneficial, especially when both contrastive loss and temporal masking are employed. The base Minkowski model, trained directly on the 4DSIS 3RScan task and without explicit temporal information sharing, already achieves a high t-sim (0.92). In contrast, Concerto and Sonata, as pre-trained encoders not specifically designed for 4DSIS, show lower baseline inter-stage consistency.

L_{contr}	ST-mask	t-mAP	t-Sim	t-mREC	Amb.	Rig.	Non-Rig.	Static	Add. & Rem.
×	×	32.0	0.9228	44.9	44.6	49.2	41.6	<u>44.9</u>	<u>29.8</u>
✓	×	31.5	<u>0.9251</u>	<u>47.3</u>	43.5	56.0	<u>44.8</u>	45.6	32.3
×	✓	30.6	0.9238	48.6	49.4	<u>54.4</u>	48.0	44.2	28.5
✓	✓	32.0	0.9261	47.2	<u>48.9</u>	52.9	44.5	44.2	29.0

Table 8. **Minkowski Ablation of temporal information sharing strategies in 4DSIS.** Rows show combinations of contrastive loss (L_{contr}), spatio-temporal serialization (ST-serial), and spatio-temporal masking (ST-mask), with t-mAP and temporal recall (t-mREC) reported for ambiguous, rigid, non-rigid, static, and added/removed instances. Best overall t-mAP is in green, best per metric in **bold**, second best per metric underlined, baseline in blue, and gray denotes combinations that do not exceed individual method performance.

While t-sim is a useful indicator of temporal feature consistency, it does not perfectly track segmentation performance. The highest t-sim configurations nearly always achieve high t-mAP, but not always the overall best. However, t-sim aligns more closely with t-REC, where high temporal consistency is important for instance recovery across change types. Notably, Minkowski achieves the highest absolute t-sim but not the best t-mAP or t-REC. This suggests that while highly consistent features are necessary for robust tracking, segmentation quality also hinges on effective query refinement and mask generation. Overall, t-sim provides a valuable auxiliary diagnostic for assessing temporal consistency in feature space, complementing mask-based metrics and revealing the encoder-dependent effects of explicit temporal sharing.

C.2. Longer Sequence Lengths

We show preliminary results for sequences $T > 2$ with our model (inference only) outperforming Mask3D+geo in Figure 8. When $T > 2$, identity drift is more challenging. Our metric t-mAP reflects this degradation, while per stage 3DSIS AP remains robust. This experiment demonstrates limitations in our model and suggests better long-term tracking will require training with more data and longer sequences.

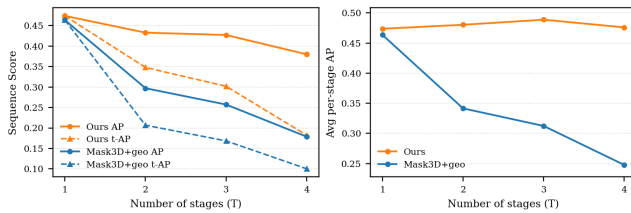


Figure 8. 4DSIS and Per-Stage 3DSIS evaluation metrics for ReScene4D inference on sequences of varying lengths.

C.3. Analysis of Baseline Performance

Mask3D and Post-hoc Matching. Semantic and geometric matching in Mask3D struggle with partial overlaps and significant scene changes (see Figure 11). Semantic matching is restricted to instances within the same predicted se-

mantic class, so matches fail if corresponding objects are classified differently across scans—especially problematic in cases with partial observations. Meanwhile, geometric matching relies on simple nearest-neighbor assignments, which quickly breaks down as scene dynamics increase. Combining the two would require manually and potentially per-sequence tuned heuristics to become reliable. In contrast, our joint instance query refinement and direct spatio-temporal mask prediction robustly handle evolving indoor scenes without manual tuning.

Mask4Former. We carefully examined Mask4Former [44] due to its architectural resemblance to our approach, notably its joint prediction of spatio-temporal instance queries and masks adapted from Mask3D [32]. As discussed in Section 5, Mask4Former was developed for outdoor LiDAR-based 4D panoptic segmentation, and is configured to operate only on input sequences ($T > 1$); adapting it for single-frame training would require substantial pipeline changes. To maintain a faithful and reproducible baseline, we train Mask4Former only on 3RScan sequence data and did not train it with a mixed 3RScan + ScanNet set. To account for the advantages of including ScanNet, we adopted a transfer learning setup for Mask4Former: it was initialized with a ScanNet-pretrained Minkowski backbone (from Mask3D) and finetuned on 3RScan. To completely isolate the impact of pretraining compared to dataset mixing, we also trained our method with the same backbone (ReScene4D M) under equivalent transfer learning training settings, where it outperforms Mask4Former by +14.9.

Mask4Former’s core design choice—superimposing point clouds from multiple scans—undermines per-stage performance in temporally sparse indoor environments. This approach yields strong instance identity consistency across scans, demonstrated by the minimal drop from single- to multi-stage metrics (see Tables 1 and 2 in the main paper, and Figures 5, 6, 10, and 12 for qualitative results). However, it incorporates temporal information (multiple scans) without an explicit temporal signal, severely degrading per-stage performance. Lacking differentiation

between temporal stages, Mask4Former cannot distinguish object movement from duplication—for example, if a chair moves slightly to the left, it may interpret this as two chairs next to each other, or, if a pair of bowls is set on a table, it may merge their identities assuming that duplication means motion instead of recognizing them as separate objects. This issue is especially pronounced for small objects. The confusing training signal from overlaid scans results in frequent instance merging and failed tracking of moved objects within and across scans. Additionally, if an object appears in a spatial location where another object was present in the previous stage, Mask4Former’s architecture prevents assigning different identities or masks to the overlapping region (see Figure 12).

Furthermore, its box loss enforces overly compact masks, which is ill-suited for objects with elongated shapes, such as doors or shelving units, leading to poor instance grouping in such cases. Since Mask4Former is also unable to train simultaneously with sequences and single-stage scans, we pretrain its Minkowski backbone using Mask3D on single-stage scans from ScanNet before training with the smaller 3RScan dataset. In contrast, our method can train directly on the mix of both datasets, hence allowing larger flexibility and an enhanced learning from both spatial and temporal signals.

Mask4D. Similar to Mask4Former, the original training codebase of Mask4D [27] is architecturally designed for sequential inputs and expects $T > 1$; however, its backbone architecture does not have existing ScanNet-pretrained single-scan models like Mask3D. Consequently, we trained the entire model, including its backbone, from scratch on the 3RScan dataset. Mask4D failed to converge reliably (see Table 1). We observed that Mask4D propagates a very limited number of instance queries, further restricting its capacity to reliably track objects over time. To ensure a fair comparison, we also trained our method (ReScene4D M) under equivalent training settings (3RScan only, from scratch), where it outperforms Mask4D by +19.0.

D. Additional Implementation Details

D.1. Mask3D Background

Additional Mask3D background is included for clarity and completeness of the description of our overall system [32]. These descriptions do not reflect our contributions.

Mask Module. The mask module assigns each instance query $X \in \mathbb{R}^{K \times D}$ a binary mask and semantic class. Instance queries are mapped through an MLP $f_{\text{mask}}(\cdot)$ to align with the backbone’s fine-grained features $F_0 \in \mathbb{R}^{M_0 \times D}$, accumulated as means over superpoints. The binary mask $B \in \{0, 1\}^{M_0 \times K}$ is obtained via dot prod-

uct, followed by sigmoid activation and thresholding $B = [\sigma(F_0 f_{\text{mask}}(X)^\top) > 0.5]$. For each query, semantic class probabilities are predicted via a linear projection onto $C + 1$ classes and softmax normalization.

Query Refinement. Instance queries $X \in \mathbb{R}^{K \times D}$ are initialized in a non-parametric fashion and progressively refined via a stack of transformer-based query decoder layers. Each layer alternates masked cross-attention to backbone features at multiple resolutions and self-attention between queries to mitigate query competition [32]. At each resolution r , backbone voxel features $F_r \in \mathbb{R}^{M_r \times D_r}$ are linearly projected to keys K and values V , while queries X yield queries Q . Each query attends only to foreground voxels from a mask module produced mask prediction from the full resolution super point features and the previous layer queries, pooling masks to match the feature hierarchy.

Loss. Mask3D supervises mask and semantic predictions using standard bipartite matching via the Hungarian algorithm, as in [32]. The assignment cost combines Dice and binary cross-entropy mask losses, and multi-class semantic cross-entropy:

$$C(k, k') = \lambda_{\text{dice}} \mathcal{L}_{\text{dice}} + \lambda_{\text{BCE}} \mathcal{L}_{\text{BCE}} + \lambda_{\text{cls}} \mathcal{L}_{\text{cls}}$$

where $C(k, \hat{k})$ is the cost of assigning predicted instance k to ground-truth instance \hat{k} ; $\mathcal{L}_{\text{dice}}$ is the Dice loss; \mathcal{L}_{BCE} is the binary cross-entropy loss over the mask; and \mathcal{L}_{cls} is the multi-class cross-entropy classification loss. The weights λ_{dice} , λ_{BCE} , and λ_{cls} balance the respective contributions of each loss component, and are set to $\lambda_{\text{dice}} = \lambda_{\text{cls}} = 2.0$, $\lambda_{\text{BCE}} = 5.0$.

D.2. ReScene4D Method Details

Defining 4D Spatio-Temporal Sequences. For each scene *sequence*, we annotate temporal changes by comparing instance labels and transformations between scans, labeling object ambiguities, non-rigid deformations, rigid movements, and added/removed instances. These per-vertex change labels are used for downstream analysis. Since these sequences have arbitrary lengths between measurements, we populate the time coordinate of the 4D point clouds with the index of each stage.

Contrastive Loss. InfoNCE loss traditionally employs cosine similarity scaled by a temperature parameter. To avoid dataset-dependent temperature tuning or learnable temperature, we adopt the temperature-free log-odds normalization [18]. Given superpoint features $f_i, f_j \in \mathbb{R}^D$, the log-odds normalized cosine similarity matrix $L \in \mathbb{R}^{S \times S}$ is:

$$L_{ij} = 2 \cdot \text{atanh} \left(\frac{f_i^\top f_j}{\|f_i\| \|f_j\|} \right) \quad (3)$$

where $f_i^\top f_j / (\|f_i\| \|f_j\|)$ is the cosine similarity between superpoints i and j .

Training Implementation. Unless otherwise specified, we adopt the hyperparameter settings from Mask3D [32], including 3D data augmentations such as random rotation and scaling, which are applied to entire registered sequences together. For all experiments, we use $N_q = 100$ queries and set the empty object loss weighting to $\lambda_{\text{empty}} = 0.2$, regardless of temporal sequence length. The mixed [3RScan:ScanNet] training ratio was selected based on preliminary Minkowski(M) backbone experiments for reasonable performance. Query initialization is performed via Farthest Point Sampling (FPS) on input point positions. Spatio-temporal point clouds are voxelized at a resolution of 2 cm. Models are trained for 450 epochs with a batch size of 32, using the AdamW optimizer and a one-cycle learning rate scheduler, with a maximum learning rate of 5×10^{-4} . During training, the PTv3 pre-trained encoders, Sonata and Concerto, are frozen while their decoder is trained from scratch. For PTv3 backbones, we train across 8 NVIDIA H100 GPUs for 26 hours. For Minkowski backbone experiments we train across 2 NVIDIA H100 GPUs for 42 hours. We utilize Stanford’s Marlowe computing cluster [17] for model training and evaluation.

D.3. Metric (t-mAP) Details

We formalize additional details of our temporal mean Average Precision (**t-mAP**) and its treatment of ambiguous ground truth instances.

Matching. Similar to mAP, prior to thresholding, a prediction $pr_i(c)$ is considered *matched* to a ground truth instance $gt_j(c)$ if $\text{IoU}(pr_i, gt_j) > 0$, where both pr_i and gt_j belong to the same semantic class c . Each ground truth instance $gt_j(c)$ may be matched by multiple predictions $pr_i(c)$, forming the set of overlapping predictions for that ground truth:

$$\mathcal{P}(gt_j(c)) = \{pr_i \in pr(c) : \text{IoU}(pr_i, gt_j(c)) > 0\}.$$

Disambiguating. For t-AP, we also match predictions to ambiguously similar sets of ground truth instances. Let \mathcal{G} be an ambiguous group containing n_{amb} ground truth instances over T stages. At each stage t , there are up to n_{amb} ground truth instances resulting in $\mathcal{G} := \{gt_k(t) \mid k \in [1, n_{\text{amb}}], t \in [1, T]\}$. Let $\mathcal{P}(\mathcal{G})$ be the set of predictions with overlap to any member $gt_k(t)$ of \mathcal{G} . The objective is to pseudo-disambiguate each ambiguous group by finding a set of unique trajectories $\{\mathcal{G}'_i : i \in [n_{\text{amb}}]\}$ —each associating one ground truth instance per stage—guided by the predictions to prevent unfair penalization of symmetric identity swaps while penalizing merges. Figure 17 illustrates

Algorithm 1 Ambiguous Instance Assignment for t-mAP

Input: $\mathcal{G} = \{gt_k(t)\}, \mathcal{P}(\mathcal{G})$
 $W_{pr,k,t} \leftarrow \text{IoU}(pr(t), gt_k(t)) \cdot C_{pr(t)} \forall pr \in \mathcal{P}(\mathcal{G}), k \in [1, n_{\text{amb}}], t \in [1, T]$
Initialize $A \in \mathbb{Z}^{n_{\text{amb}} \times T} \leftarrow \text{unassigned}$
for $i = 1, \dots, n_{\text{amb}}$ **do**
 $p^* \leftarrow \arg \max_{pr} \sum_t \max_k W_{pr,k,t}$
for $t = 1, \dots, T$ **do**
 $k^* \leftarrow \arg \max_k W_{p^*,k,t}$
if $W_{p^*,k^*,t} > 0$ **then**
 $A_{i,t} \leftarrow k^*$
 $W_{k^*,t} \leftarrow 0$
 $W_{p^*,t} \leftarrow 0$
end if
end for
end for
Fill any $A_{k,t} = \text{unassigned}$ with available $gt_k(t)$
Partition \mathcal{G} into $\{\mathcal{G}'_i : i \in [1, n_{\text{amb}}]\}$ using A indices
return Pseudo gt and matches $\{(\mathcal{G}'_i, \mathcal{P}(\mathcal{G}'_i))\}$

instances of identity switches that should not be penalized by the evaluation metric.

For each $pr \in \mathcal{P}(\mathcal{G})$, $k \in [1, n_{\text{amb}}]$, and temporal stage $t \in [1, T]$, we define the assignment weight as:

$$W_{p,k,t} := \text{IoU}(pr(t), gt_k(t)) \cdot C_{pr(t)}$$

where $C_{pr(t)}$ is the confidence score of prediction pr . This weight quantifies how well each prediction supports a particular ground truth instance at each temporal stage, taking both overlap and prediction confidence into account.

The assignment process proceeds iteratively: at each step, the prediction with the highest total weight—i.e., exhibiting the best combination of overlap and confidence with a specific trajectory of ground truth instances—is selected and assigned to that trajectory. Subsequent predictions are assigned to the next-best groupings in a greedy fashion.

If a prediction does not overlap with a temporal stage, these stages in the trajectory are left unassigned. After all predictions have been exhausted or all k ground truth disambiguated trajectories are at least partially defined, any remaining unassigned ground truth instances are randomly assigned to available trajectories to ensure that each trajectory is completed. This process is defined in Algorithm 1 a toy visual example is included in Figure 9.

Assigning. For each class c and threshold τ , the sets of true positives, false positives, and false negatives are formulated from the overall set of predictions $pr(c)$ and ground truth instances $gt(c)$. Pseudo-disambiguated ground truth

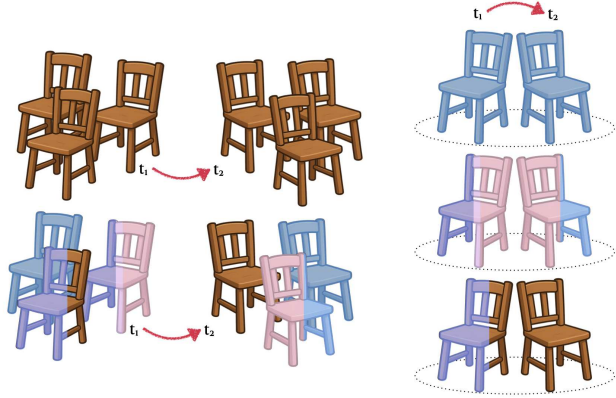


Figure 9. **Toy Example for Pseudo-Disambiguating Ambiguous Instance Groups.** Predictions are used to guide the process of creating disambiguated groups despite prediction conflicts or errors by ordering priority based on mask quality, overlap, and confidence

instances are treated as normal. Predictions are processed in order of descending confidence. A predicted instance $pr_i \in pr(c)$ is assigned as a true positive, TP_c , to a ground truth $gt_j \in gt(c)$ if their temporal intersection-over-union ($t\text{-IoU} > \tau$), and gt_j has not already been assigned. For each prediction, among all eligible ground truth instances, assignment is made to the gt_j with the highest overlap. Each ground truth instance receives at most one assigned prediction, and each prediction is assigned to at most one ground truth instance. False positives, FP_c , and False negatives, FN_c , are defined by the set of unpaired predictions and ground truth.

Once TP_c , FP_c , and FN_c are determined, precision-recall curves are computed using prediction confidences to calculate $t\text{-AP}$ for each class; $t\text{-mAP}$ is their mean. Additionally, we report per-change-type temporal recall by grouping true and false negatives by annotated instance change and aggregating recall across all classes.

We create a custom PyTorch Lightning Metric for our $t\text{-mAP}$ and $t\text{-mREC}$ which will be released publicly.

E. Limitations

Our experiments on 3RScan and ScanNet demonstrate improved performance and robustness over baseline methods, highlighting the importance of explicit temporal information sharing and unified architecture for joint predictions. However, the effectiveness of temporal modeling is closely linked to the dataset, with current benchmarks such as 3RScan limited by semantic class diversity and the distribution of changes (dominated by static instances). Despite these data limitations, methods with strict geometric assumptions (Mask3D+geo, Mask4Former) still do not outperform ours, even in scenes with minimal changes. This demonstrates that, even under conditions that ideally favor

geometric methods, our approach remains more robust to the unique challenges of temporally sparse 4D indoor segmentation, such as partial scans, noise, and small misalignment. Though good registration is generally achievable with modern methods, different training strategies for non-registered scans could be included in future work. Progress in this area urgently requires more diverse, large-scale annotated 4D indoor datasets of changing scenes to better support and evaluate advanced and realistic 4D tasks with more significant geometric variation.

Computational complexity remains a challenge; the use of temporally paired scans increases memory and processing demands, restricting our experiments to short temporal windows and moderate scene sizes.

Additionally, our focus was to demonstrate the impact of temporal information sharing, and we did not perform extensive hyperparameter tuning or incorporate the latest 3DSIS query and feature refinement approaches presented in recent literature [20, 23, 24, 40]. Future work could integrate these advances to further boost prediction accuracy and efficiency. As annotated 4D datasets expand and more sophisticated modeling strategies emerge, we expect temporal indoor segmentation methods to become increasingly robust and scalable.

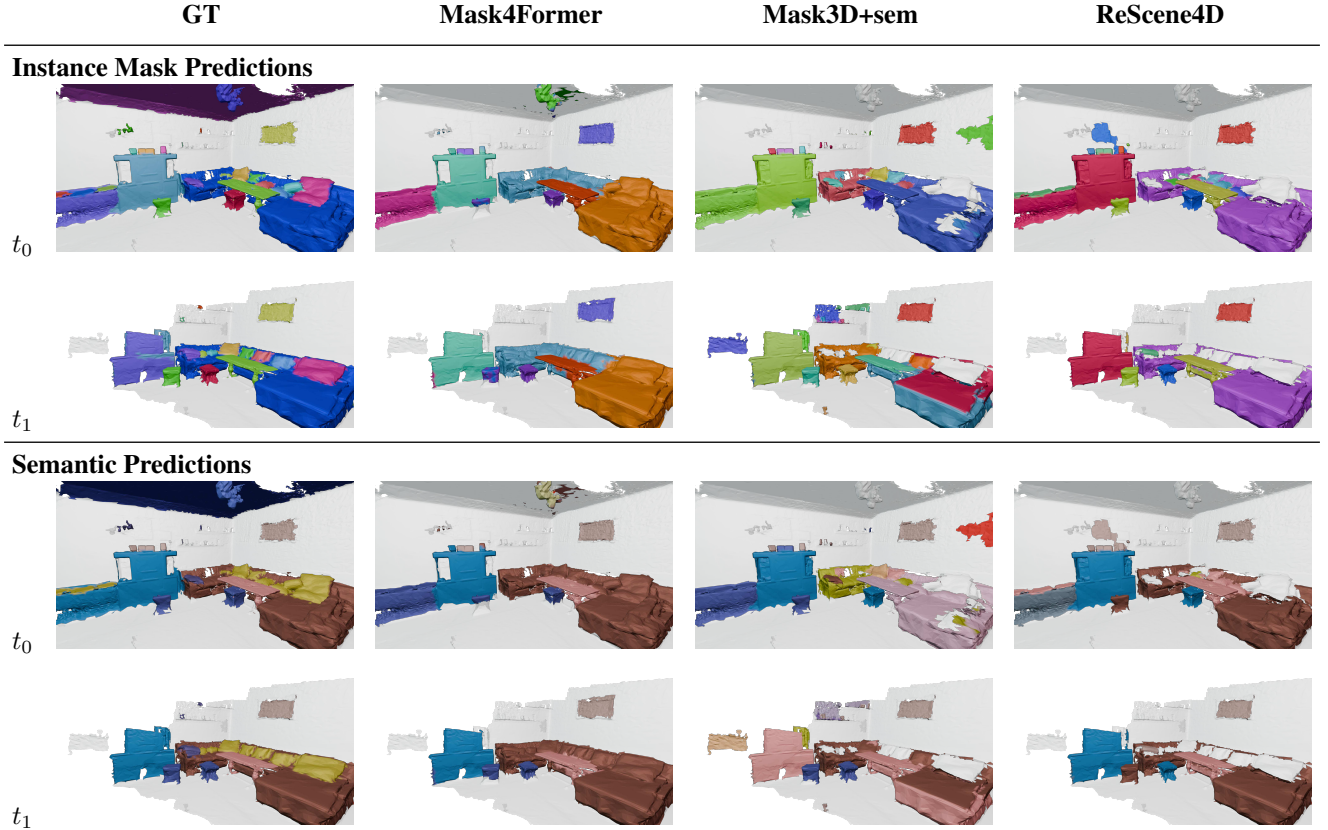


Figure 10. **Temporal Segmentation Comparison—Baseline Comparison.** Mask4former merges the two stools as one instance in a patched manner and considers the sofa as two different instances, despite predicting the same class label. Mask3D produces several semantic segmentation errors between the two scans (and the ground truth), hence leading to incorrect temporal association of instances. ReScene4D robustly associates semantics and instances across time even if classifying a stool as a sofa.

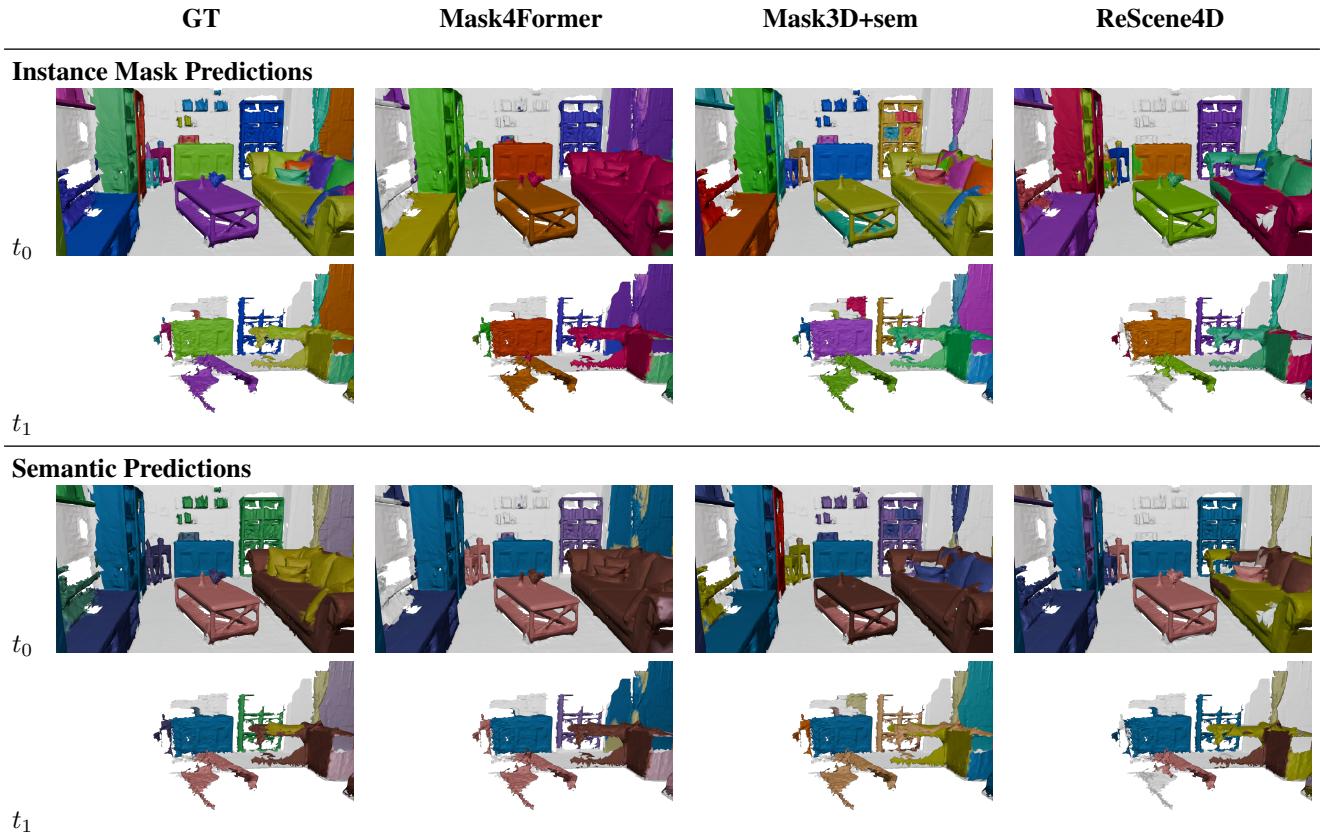


Figure 11. **Temporal Segmentation Comparison—Baseline Comparison.** Errors in the Mask3D baseline arising from partial observations can be corrected by incorporating temporal information; ReScene4D reliably identifies the partially visible coffee table, as shown in both instance and semantic segmentation predictions. Mask4Former also benefits from temporal overlay when there are no object changes, relying heavily on spatial alignment. All methods struggle with inconsistently annotated couches and pillows in 3RScan, where labels are often interchangeable between these objects across different scenes in the dataset.

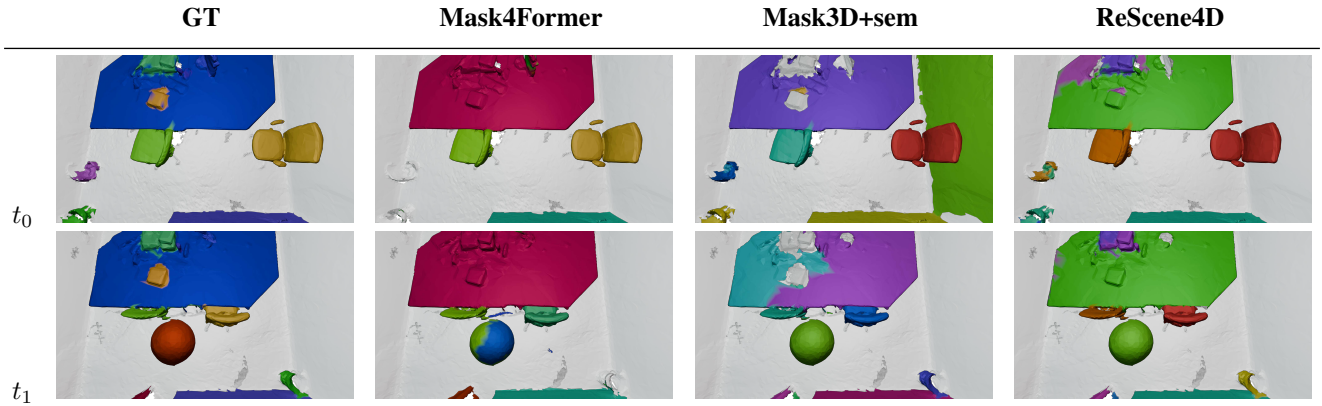


Figure 12. **4D Semantic Instance Segmentation—Baseline Comparison.** Mask4Former fails to identify moved chair identities when there is no spatial overlap and confuses the ball mask due to overlapping spatial regions between stages. Mask3D fails to correctly associate one of the chairs. In contrast, ReScene4D accurately tracks and masks the moving chairs.

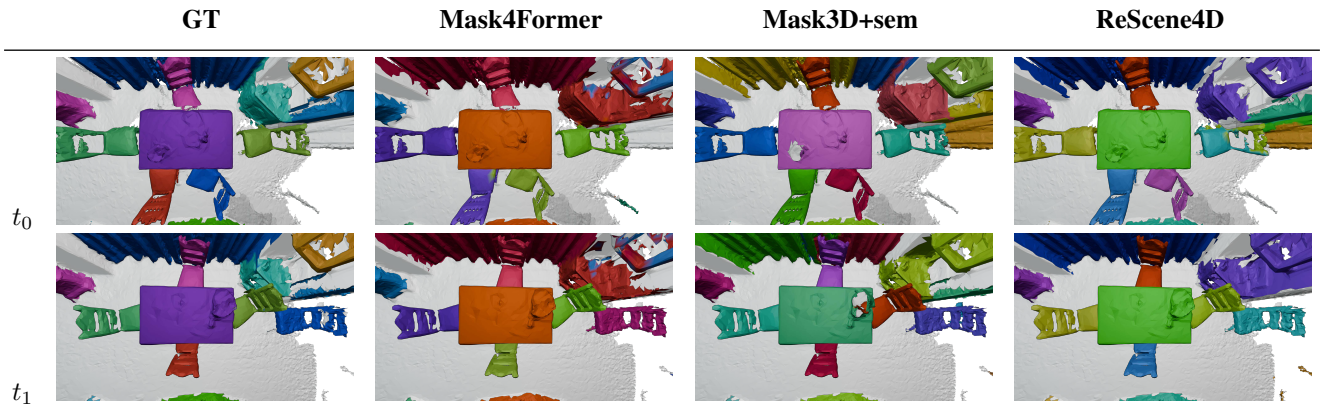


Figure 13. **4D Semantic Instance Segmentation—Baseline Comparison.** Ambiguous scenarios, with a set of objects with similar features (here *chairs*), present a challenge and remain a limitation for all 4DSIS methods. In this example, Mask4Former merges the identities of two chairs into one instance in the first scan while still tracking a single chair in the next scan. Mask3D loses track of two chairs in the temporal matching and fails to match static instances such as the table and the curtains. ReScene4D primarily maintains correct identities, with only a single missed chair tracking.

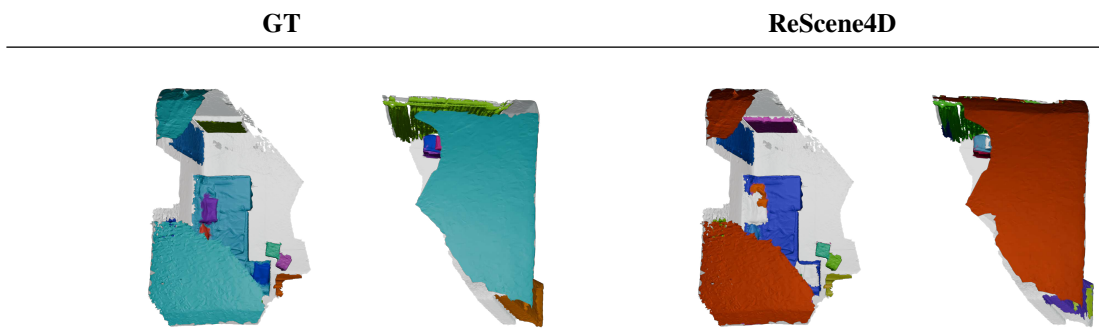


Figure 14. **4D Semantic Instance Segmentation—Without Spatial Overlap.** In scans of the same scene that have zero overlap, a common ceiling instance is correctly identified by ReScene4D without forcing matches between non-spatially overlapping objects.

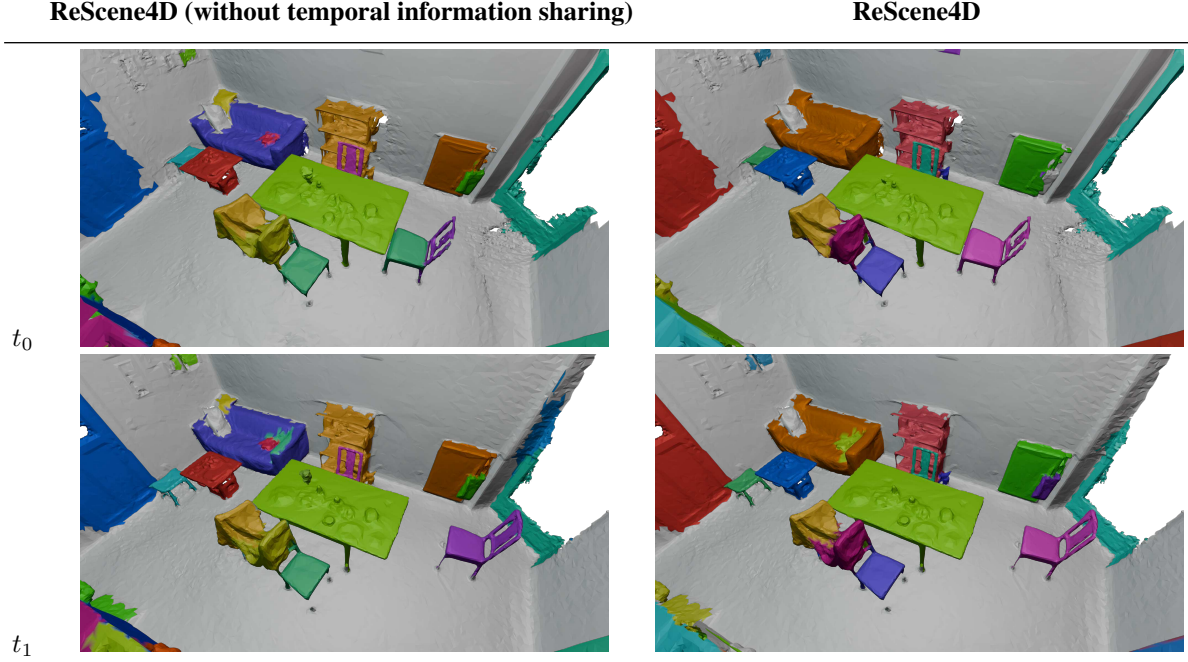


Figure 15. **4D Semantic Instance Segmentation Architecture Ablation.** Comparison between our best-performing ReScene4D, which shares information temporally via cross-time contrastive loss and ST-masking (left), and a standard spatio-temporal architecture that decodes joint queries without explicit temporal information sharing. While the standard method benefits from unified 4D modeling, it struggles with ambiguous instances—e.g., incorrectly segmenting two chair seats as one due to reliance on semantic consistency. Explicit temporal information sharing helps disambiguate instances by introducing geometric and contrastive cues without rigid constraints.

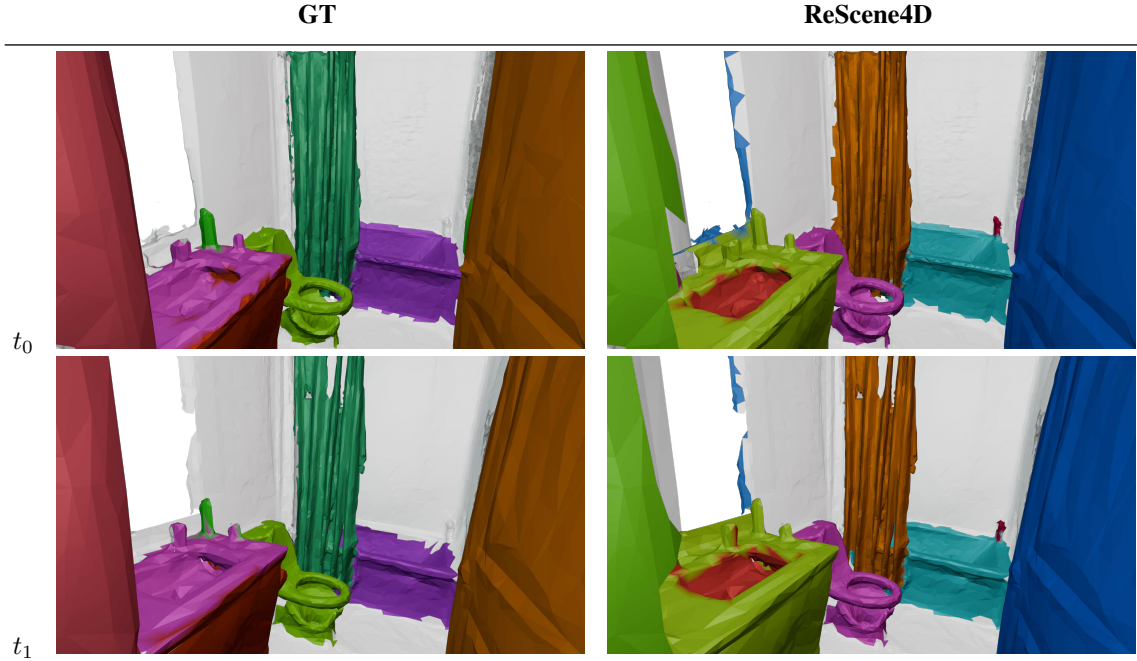


Figure 16. **4D Semantic Instance Segmentation—Dataset Annotation.** Here we illustrate ground truth annotation inconsistencies, particularly for small instances, where ReScene4D predicts objects not labeled and missing an annotated object. For example, the soap bottle on the sink is annotated in the ground truth but missed by us, vs. the shampoo bottle on the right corner of the bathtub is correctly and consistently segmented by us but is non-existent in the ground truth. We are thus penalized in both cases.

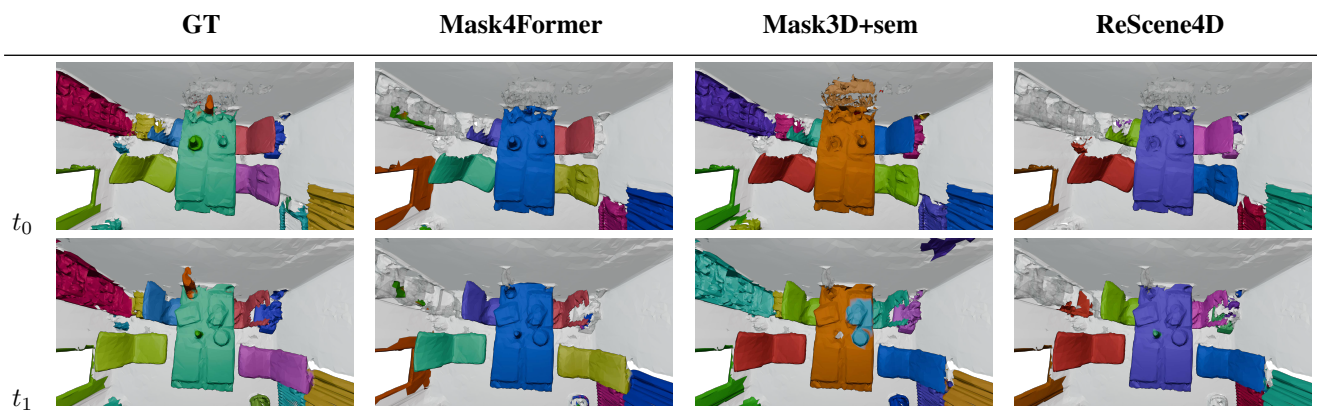


Figure 17. **4D Semantic Instance Segmentation—Metric Highlight.** All methods correctly associate ambiguous chairs, but Mask3D visually shuffles identities within the group due to the lack of geometric information, not matching chairs to their nearest neighbor even though they might have remained static. Our metric, t-mAP, does not penalize such cases, as this represents a valid solution.