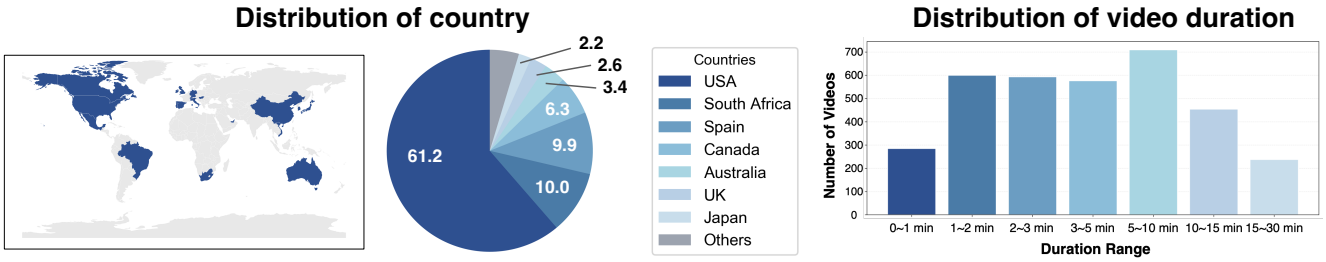

Figure 7. **Statistics of the `RoomTours` video collection.** Our `RoomTours` consists of 3,462 indoor walkthrough videos collected from 19 countries. (Left) Geographic distribution of the source countries and their proportions. (Right) Distribution of video durations, ranging from short clips to long walkthroughs, with a median length of 3.63 minutes.

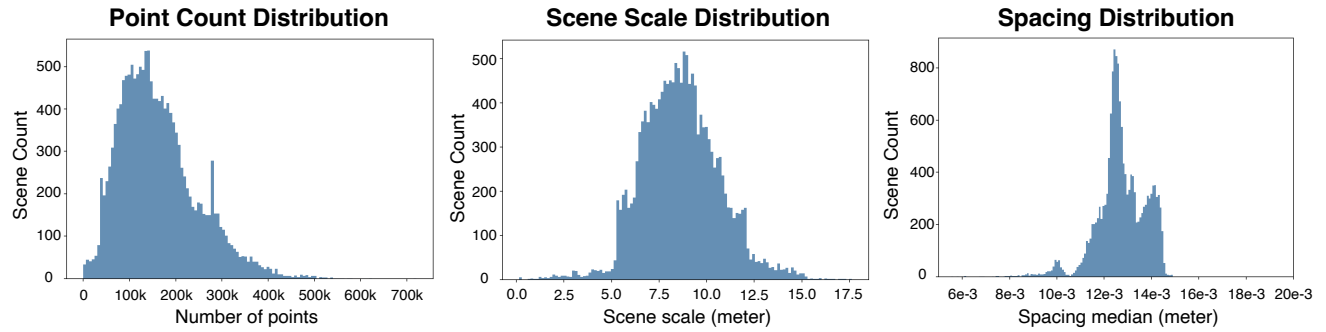

Figure 8. **Statistics of the video-generated point clouds.** We report the properties of the aligned scenes used for pre-training. (Left) Distribution of the number of points per scene before the align processing. (Center) Distribution of scene scales (the diagonal length of the axis-aligned bounding box). (Right) Distribution of the median point spacing, defined as the median kNN distance within each scene.

A. `RoomTours` Details

We construct `RoomTours` using a feedforward reconstruction model applied to unlabeled videos from the web. The dataset consists of VGPC obtained from both our collected videos and existing datasets (RealEstate10k [62], YouTube House Tours [10], and HouseTours [9]). In this section, we report statistics of our independently collected videos and the VGPC reconstructed from them.

For our own collection, we assume that indoor layouts and furniture vary across geographic regions. Therefore, we collected walkthrough videos from 19 countries, as shown in Fig. 7. In total, we gathered 3,462 videos, each with a median duration of 3.63 minutes.

For scene classification, we sample every frame at the native FPS of each video and resize all frames to the standard CLIP [34] ViT-B/32 resolution of 224×224. Each frame is first classified by the CLIP image encoder as either "indoor" or "outdoor." Frames predicted as indoor are then re-evaluated using room-type prompts (living room, bedroom, bathroom) to obtain scene categorization. This two-stage filtering produces a total of 15,921 indoor sequences, with each video contributing on average 4.59 sequences.

The reconstruction of a single scene takes approximately 5 minutes on average, although the actual time depends on the number of frames extracted from each video. We process our collected videos using eight NVIDIA H200 GPUs (144GB each), requiring roughly eight hours in total to generate the video-based point clouds. The raw outputs are pixel-aligned point clouds and therefore contain an extremely large number of points. Directly applying our post-processing pipeline to these raw point clouds leads to substantial computational overhead. To maintain efficiency, we randomly downsample each scene to 20k points before alignment and normalization. Fig. 8 reports the statistics of the post-processed VGPC, including the distribution of point counts and scene scale. After post-processing, the structural properties of the indoor scenes closely match those observed in standard ScanNet scenes.

B. Implementation Details

This section explains the details of our implementation. We begin by describing Point Transformer V3 (PTv3) [51] which is the backbone model used in our experiments. We then detail the pre-training setup and the configurations adopted for semantic segmentation and instance segmentation. Our implementation is based on PointCept [13].

Backbone network. PTv3 improves efficiency by removing the kNN query and the Relative Positional Encoding

Table 6. Configuration of PTv3 (Base) and PTv3 (Large)

Config	Value	
	PTv3 (Base)	PTv3 (Large)
order	Z + TZ + H + TH	Z + TZ + H + TH
stride	(2, 2, 2, 2)	(2, 2, 2, 2)
enc_depths	(3, 3, 3, 12, 3)	(3, 3, 3, 12, 3)
enc_channels	(48, 96, 192, 384, 512)	(64, 128, 256, 512, 768)
enc_num_head	(3, 6, 12, 24, 32)	(4, 8, 16, 32, 48)
enc_patch_size	1024 × 5	1024 × 5
mlp_ratio	4	4
qkv_bias	True	True
drop_path	0.3	0.3
head_in_channels	1088	1536
head_embed_channels	256	256
head_prototypes	4096	4096
params	121M	224M

Table 7. Pre-training configuration for LAM3C

Config	Value
laplacian kNN	24
huber delta	0.5
max radius distance	0.08
laplacian loss weight	2e-4 → 3e-3
noise consistency loss weight	0.05
teacher temperature	0.04 → 0.07
student temperature	0.1
views (global / local)	2 / 4
optimizer	AdamW
base learning rate	1e-3
layer-wise LR decay	0.9
weight decay	0.04 → 0.10
scheduler	OneCycleLR
batch size	16
iterations	145,600 (RoomTours-1k, PTv3-Base) 291,200 (RoomTours-49k, PTv3-Base) 436,800 (RoomTours-49k, PTv3-Large)

(RPE) modules in PTv2 [49]. The kNN query module was designed to capture local structure on unstructured point clouds, but it requires repeated distance computations at every iteration and occupies 28% of the forward time. The RPE module encodes relative positions, yet point cloud RPE must compute pairwise Euclidean distances for every point pair, accounting for 26% of the computation. PTv3 avoids both costs by abandoning the unordered-set formulation and serializing point clouds into a structured representation, which eliminates the need for kNN and RPE computations and leads to efficiency gains. PTv3 assigns a consistent order to points through four ordering strategies: Z-order (Z), Transposed Z-order (TZ), Hilbert (H), and Transposed Hilbert (TH).

We use PTv3 as the backbone model for all experiments, with two variants: PTv3 (Base) and PTv3 (Large). The Base variant follows the configuration used in Sonata, and the Large variant follows the configuration used in Concerto [60]. The detailed parameter settings are provided in Table 6.

Pre-training setting. We set the default parameters as shown in Table 7. The Laplacian smoothing loss hyperparameters were set based on experimental results and computational considerations during pre-training. In our formulation, the Laplacian smoothing loss penalizes pairwise feature differences between neighboring points on the kNN graph. When the feature difference on an edge is small, the Huber penalty behaves quadratically; when it exceeds δ, it becomes linear, reducing the influence of outlier edges. Other settings related to the clustering loss follow the configuration used in Sonata. Furthermore, our comparative experiments employed three settings: RoomTours-16k (PTv3-Base), RoomTours-49k (PTv3-Base), and RoomTours-49k (PTv3-Large). Based on empirical results from previous research, the number of iterations is scaled with both data size and model size.

Table 8. Indoor semantic segmentation settings

Linear Probing		Full Fine-Tuning	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	OneCycleLR	scheduler	OneCycleLR
criteria	CrossEntropy	criteria	CrossEntropy
learning rate	2e-3	learning rate	2e-3
block LR	N/A	block LR	2e-4
weight decay	2e-2	weight decay	2e-2
batch size	168	batch size	128
epochs	100	epochs	100

Table 9. Indoor instance segmentation settings

Linear Probing		Full Fine-Tuning	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	OneCycleLR	scheduler	OneCycleLR
criteria	CrossEntropy	criteria	CrossEntropy
learning rate	2e-3	learning rate	2e-3
block LR	N/A	block LR	2e-4
weight decay	5e-2	weight decay	5e-2
batch size	168	batch size	168 (ScanNet / S3DIS) 144 (ScanNet200) 96 (ScanNet++)
epochs	100	epochs	100

Downstream task setting. We evaluate the pre-training effect on two downstream tasks: semantic segmentation and instance segmentation in indoor scenes. Semantic segmentation and instance segmentation are evaluated on ScanNet [15], ScanNet200 [35], ScanNet++ [56], S3DIS [2]. For both tasks, we adopt the following fine-tuning protocols: Linear probing and Full fine-tuning. Linear probing freezes the entire backbone and trains only a linear classifier, whereas Full fine-tuning updates all model parameters. For the linear probing and the full fine-tuning protocol, we follow Sonata setting [53]. The configuration for semantic segmentation is provided in Table 8, and the configuration for instance segmentation is provided in Table 9.

k	LP	Full-FT
8	56.0	**75.9**
24	**57.1**	75.1
32	54.9	75.2

(a) **kNN neighborhood size**

radius	LP	Full-FT
0.04	55.3	**75.7**
0.08	**57.1**	75.1
0.12	55.3	75.4

(b) **Radius threshold for noisy point removal.**

δ	LP	Full-FT
0.05	55.7	**75.5**
0.5	**57.1**	75.1
1.0	55.4	75.3

(c) **Huber loss parameter** δ.

σ	LP	Full-FT
adaptive	**57.1**	75.1
0.03	56.0	**75.4**
0.08	56.2	75.0

(d) **Gaussian scale parameter** σ**.**

λ_{start}	λ_{base}	μ	LP	Full-FT
2e-4	3e-3	5e-2	**57.1**	75.1
2e-4	3e-4	2e-2	56.7	**75.7**
2e-4	3e-4	5e-2	56.5	75.2
2e-4	3e-3	2e-2	54.9	74.8

(e) **Loss Weights** λ **and** μ**.**

Table 10. **Hyperparameter Analysis Experiments** on the ScanNet semantic segmentation. We report Full Fine-tuning (Full-FT) and Linear Probing (LP) performance (%). Unless noted otherwise, the default configuration uses `RoomTours`-1k generated by π^3 and is pre-trained with PTv3 (Base). Both Full-FT and LP are trained for 100 epochs. Default settings are highlighted in gray.

C. Further Analysis Experiments

We analyze the key hyperparameters of LAM3C. Our ablations focus on the neighborhood size for Laplacian smoothing loss, the radius threshold for noisy point removal, the Huber loss parameter δ, the Gaussian scale parameter σ, and the loss weights λ and μ for Laplacian smoothing and noise consistency terms.

kNN neighborhood size (see Table 10a). We analyze the effect of the neighborhood size used to construct the kNN graph for the Laplacian smoothing loss. In our LAM3C, the Laplacian smoothing loss is computed on a kNN graph constructed for each VGPC. The value of k determines the size of the local neighborhood, thus controlling the strength and spatial extent of the smoothing imposed by the regularization.

We pre-train LAM3C with three values k=$\{8, 24, 32\}$, and evaluate the resulting models on ScanNet semantic segmentation. As shown in Table 10a, k=24 yields the best performance, particularly in the Linear Probing setting where representation quality is most directly reflected. We consider that smaller k results in unstable local neighborhoods, while larger k leads to over-smoothing.

Radius threshold for noisy point removal (see Table 10b). We analyze the effect of the radius threshold used to remove distant neighbors when constructing the kNN graph for the Laplacian smoothing loss. In VGPC, a noisy reconstructed point may cause a query point to retrieve faraway points among its top-k neighbors, which violates the locality assumption required for Laplacian regularization. To mitigate this issue, we apply a distance threshold r_{max} and discard neighbors whose Euclidean distance exceeds this value.

We evaluate three thresholds r=$\{0.04, 0.08, 0.12\}$, by pre-training LAM3C with each setting and conducting semantic segmentation on ScanNet. As shown in Table 10b, r=0.08 achieves the best performance, particularly in the Linear Probing setting, indicating that this threshold strikes an effective balance between removing noisy outliers and preserving sufficient local structure.

Huber loss parameter δ (see Table 10c). We analyze the effect of the Huber loss parameter δ, which determines the transition point between the quadratic and linear regimes of the Huber loss. In our formulation, the Laplacian smoothing loss penalizes pairwise feature differences between neighboring points on the kNN graph. When the feature difference on an edge is small, the Huber penalty behaves quadratically; when it exceeds δ, it becomes linear, reducing the influence of outlier edges.

We evaluate three settings δ=$\{0.05, 0.5, 1.0\}$ and assess the pre-trained models on ScanNet semantic segmentation. As shown in Table 10c, δ=0.5 achieves the best Linear Probing performance, suggesting that it provides an effective balance between sensitivity to local variations and robustness to noisy neighborhoods.

Gaussian scale parameter σ (see Table 10d). We analyze the effect of the Gaussian scale parameter σ, which controls the decay rate of the distance-based weighting used in the Laplacian smoothing loss. We assign each edge a distance-based weight $w_{ij} = \exp(-\|p_i - p_j\|^2/\sigma^2)$, where a larger σ results in a slower decay of the weights, allowing distant points to retain non-negligible influence. Conversely, a smaller σ produces a sharper decay, restricting the effective neighborhood to only very close points. To account for the varying density of VGPC, our method adaptively sets σ as the median of the kNN distances for each point cloud.

We compare three settings, σ=$\{$adaptive$, 0.03, 0.08\}$, and evaluate the pre-trained models on ScanNet semantic segmentation. As shown in Table 10d, the adaptive setting achieves the best performance, particularly in the Linear Probing evaluation, indicating that adaptive scaling pro-

Table 11. **Estimating the effect of test/val set contamination.** All self-supervised methods are evaluated by full fine-tuning (Full-FT) or linear probing (LP) on PTv3 (Base) for 100 epochs. We use mIoU as evaluation metric.

Semantic Seg.	Test/Val set		ScanNet [15]		ScanNet200 [35]		ScanNet++ Val [56]		S3DIS Area 5 [2]	
Methods	With	Without	LP	Full-FT	LP	Full-FT	LP	Full-FT	LP	Full-FT
Sonata (ScanNet)	✓	–	**67.3**	**77.4**	26.6	**32.7**	**35.1**	**42.4**	**63.4**	**72.5**
Sonata (ScanNet)	–	✓	67.1	75.4	**27.2**	32.2	34.3	41.7	61.2	72.2

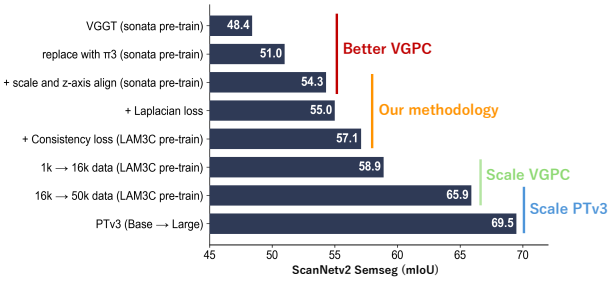

Figure 9. Component synergy on ScanNet semantic segmentation (linear probing). Replacing VGGT with pi^3, adding indoor alignment, and introducing LAM3C each improve performance, with further gains from scaling RoomTours and using PTv3-Large.

vides robustness across point clouds with diverse geometric densities.

Loss weights λ and μ (see Table 10e). We analyze the impact of the weighting coefficients λ and μ used for the Laplacian smoothing loss and Noise consistency loss, respectively. As defined in Eq. (5), the overall objective consists of the distillation loss from Sonata and two additional regularization terms. Here, we focus on the balance between these two regularized losses, which are introduced to stabilize representation learning from VGPC. In our formulation, the coefficient λ for the Laplacian smoothing loss follows a scheduled progression during training, whereas the coefficient μ for the Noise consistency loss is fixed.

We evaluate four combinations of (λ, μ) listed in Table 10e, pre-train LAM3C under each setting, and assess the resulting models on ScanNet semantic segmentation. As shown in Table 10e, the configuration $\lambda_{\text{start}}=2\times10^{-4}$, $\lambda_{\text{base}}=3\times10^{-3}$, and $\mu=5\times10^{-2}$ achieves the best Linear Probing performance, indicating that this balance of regularization strengths leads to more stable and discriminative representations.

Component synergy (see Figure 9). We add a combination analysis on ScanNet linear probing. Keeping the same Sonata pre-training, switching the reconstruction model (VGGT→π^3), adding indoor alignment, and further adding LAM3C consistently improve performance, with continued gains when increasing data scale and model capacity, indicating complementary and synergistic effects between components.

D. Fairness considerations for baseline

In the paper, we compare against Sonata as it is the current state of the art. However, the official Sonata models are pre-trained on all splits of ScanNet, ScanNet++, and S3DIS including the {train / val / test} splits used for downstream evaluation[2]. This setup is not comparable to our LAM3C, which, even during finetuning evaluations, does not see any {test / val} data. Such {test / val} inclusion has been common practice in previous indoor 3D-SSL methods [50, 52], but blurs the true generalisation performance measurement.

To ensure a fair comparison, we construct a fair Sonata baseline by pre-training Sonata using only the train split of ScanNet, completely removing the corresponding {val / test} scenes from pre-training.

We first examine how excluding the {val / test} scenes affects Sonata's performance when pre-trained on ScanNet. Table 11 reports the results for semantic segmentation. Removing {val / test} leads to a consistent drop in performance, confirming that the standard Sonata configuration benefits from the exposure to the evaluation scenes. This leakage provides prior knowledge of scene and object layouts, leading to performance that cannot be attributed purely to generalization. Although this paper focuses on ScanNet, we expect the performance gap to widen when combining multiple datasets such as ScanNet++, S3DIS, and others, where leakage accumulates across datasets.

These findings highlight a fundamental issue in how 3D-SSL methods have been evaluated. While a deeper investigation is beyond the scope of this paper, we ensure fairness in all experiments by using Sonata models pre-trained strictly on the train split only.

E. Discussion

Our findings show that high-fidelity 3D scans are not a prerequisite for effective 3D-SSL. Despite being noisy, incomplete, and inconsistent, VGPC reconstructed from unlabeled videos prove sufficient for 3D-SSL. Perhaps most notably, LAM3C performs strongly in linear probing, a setting that has not been widely explored in previous 3D-SSL methods.

[2]The official configuration can be verified in the Sonata GitHub repository:
https://github.com/Pointcept/Pointcept/blob/main/configs/sonata/pretrain-sonata-v1m1-0-base.py

This indicates that these imperfect reconstructions still encode rich geometric cues relevant to real-world tasks. Overall, our results highlight a promising direction toward scalable 3D data generation. Instead of costly real 3D scans, unlabeled videos offer a far more scalable source of 3D supervision, where the key is learning to handle noisy and missing regions effectively.

F. Limitations and outlook

Although the reconstruction models used in this paper use real point clouds during training, our core contribution lies in showing that previously untapped resources such as web videos can be used for 3D-SSL and become more useful at scale. By piggybacking on reconstruction models, our approach benefits from the fast progress in this domain and might enable unifying 3D reconstruction and recognition models. Our method is designed for predominantly static indoor scenes and may be limited in highly dynamic environments, where reconstruction quality degrades due to many moving objects. While our noise regularization mitigates some outliers in `RoomTours`, explicit modeling of dense dynamics and motion remains an important direction for future work.