

SPARC: Scalable 3D Panoptic Segmentation with Reinforcement-driven Clustering

Haoling Huang¹, Ting Han², Jie Ouyang², Yiping Chen^{2,*}

¹ School of System Science and Engineering, Sun Yat-Sen University, Guangzhou, China

² School of Geospatial Engineering and Science, Sun Yat-Sen University, Zhuhai, China

Keywords: Panoptic Segmentation, Reinforcement Learning, Point Clouds, Superpoint Graph, Remote Sensing.

Abstract

Large-scale 3D panoptic segmentation is critical for digital twins and geospatial analysis, demanding models that process massive point clouds while distinguishing instances across highly diverse spatial scales. However, prevailing graph-based approaches rely on one-shot optimization, suffering from *short-sighted decisions* where irreversible local errors propagate globally, leading to severe under-segmentation at boundaries between objects of disparate scales. To overcome this short-sightedness, we present **SPARC**, a scalable framework that reframes graph clustering as a sequential, self-correcting decision process driven by hierarchical reinforcement learning. Specifically, SPARC employs a dual-level agent where a meta-controller adaptively determines instance completeness while a low-level policy iteratively refines edge affinities, enabling the model to revise early mistakes based on long-horizon rewards rather than greedy local cues. Complementing this, we introduce Semantic Voxel Partitioning (SVP) to generate semantically coherent superpoints, ensuring robust primitives that mitigate noise before clustering begins. Extensive experiments demonstrate that SPARC achieves state-of-the-art performance on the DALES dataset with a Panoptic Quality of 62.4%, surpassing existing methods by 9.8% and effectively resolving multi-scale segmentation ambiguities.

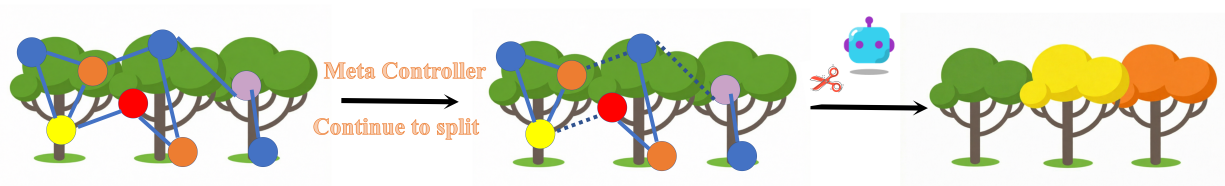


Figure 1. A simple example of how SPARC work. Starting from a superpoint graph, the Meta-Controller determines whether to terminate or continue splitting. When refinement is needed, the Low-level Controller adjusts edge affinities to separate ambiguous boundaries. This hierarchical process iteratively partitions the graph into semantically coherent instances, resulting in accurate panoptic segmentation.

1. Introduction

Large-scale 3D panoptic segmentation is fundamental for numerous real-world applications, including the creation of digital twins for industrial facilities and urban environments (Quattrocchi et al., 2022, Jiang et al., 2021), large-scale geospatial analysis for land and forest surveys (Smeeckaert et al., 2013, Hauglin et al., 2021), and comprehensive building inventory for city-scale mapping (Wang et al., 2020). This task requires models that can simultaneously process point clouds with millions of points, predict fine-grained semantic labels for each point, and distinguish individual object instances—a challenging combination referred to as 3D panoptic segmentation (Kirillov et al., 2019). Recent advances have explored various approaches for this task: some methods employ sparse LiDAR-based networks for autonomous driving scenarios (Aygun et al., 2021, Fong et al., 2022), while others utilize dense indoor scanning techniques with transformer architectures (Zhao et al., 2021, Lai et al., 2022). More recently, superpoint-based methods have emerged as a promising direction, leveraging graph-based representations to achieve efficient segmentation (Landrieu and Boussaha, 2019, Robert and Raguét, 2023). Notably, SuperCluster (Robert et al., 2024) reformulated panoptic segmentation as a graph clustering problem, achieving state-of-the-art results with significantly reduced computational

requirements.

Despite these advances, existing methods face critical limitations when applied to large-scale scenes with complex object distributions. Most graph-based approaches, including SuperCluster (Robert et al., 2024), formulate panoptic segmentation as a one-shot graph clustering problem, where all edge affinities are predicted and the entire graph is partitioned in a single optimization step. This formulation, while efficient, suffers from two fundamental issues when handling objects with highly varying spatial scales. First, the fixed clustering thresholds or globally shared parameters cannot adapt to local scene complexity: edges between large structures (e.g., building facades) require different decision boundaries than those between small objects (e.g., vehicles), yet one-shot methods apply uniform criteria across all scales. Second, attention-based or graph convolutional networks, which compute edge affinities through local message passing, tend to produce under-segmentation at boundaries where objects of disparate scales meet—the attention weights are biased toward dominant large-scale patterns, causing small adjacent instances to be erroneously merged. Fundamentally, these limitations stem from a fatal flaw in single-step optimization: **Short-sighted decisions**. By relying on a single optimization step, existing methods make definitive edge assignments without evaluating how local prediction er-

rors might propagate. In scenes with diverse object scales, an incorrect local merge can easily accumulate, leading to severe under-segmentation. Because traditional static graph partitioning lacks an iterative feedback mechanism to correct local decisions based on their impact on the overall segmentation, these approaches often struggle to handle the structural complexity of large-scale heterogeneous point clouds.

To break this ceiling, we argue that panoptic segmentation must be reframed from a static graph task into a **sequential, self-correcting decision process**. The qualitative pipeline of SPARC is shown in Fig. 1. In this paper, we present **SPARC** (Scalable Panoptic segmentation with Adaptive Reinforcement-driven Clustering), a novel hierarchical reinforcement learning framework. This formulation provides the crucial missing piece: long-horizon credit assignment that naturally evaluates the global impact of local clustering decisions, enabling adaptive refinement of object boundaries at multiple scales. Our key contributions are:

- **Sequential clustering formulation:** We introduce the first reinforcement learning-based approach for 3D panoptic segmentation that treats graph clustering as a multi-step decision process, enabling adaptive refinement of object boundaries at multiple scales.
- **Hierarchical policy learning:** We design a dual-level agent architecture with a meta-controller that determines when to terminate clustering and a low-level controller that makes fine-grained edge decisions, allowing the model to handle objects with diverse spatial scales.
- **Semantic-aware superpoint initialization:** We propose a Semantic Voxel Partitioning (SVP) module that generates high-quality initial superpoints using MinkUNet features, providing robust primitives for the subsequent clustering process.

2. Method

Our method consists of two key components, as shown in Figure 2. First, we introduce Semantic Voxel Partitioning to produce semantically consistent superpoints that serve as high-quality primitives. Second, we formulate 3D panoptic segmentation as a sequential graph clustering problem, solved using a Hierarchical Reinforcement Learning framework.

2.1 Preliminaries

Superpoint. A superpoint is defined as a set of spatially adjacent points that share similar geometric and semantic properties. Formally, given a point cloud $P = \{p_1, \dots, p_N\}$, a superpoint $S_k \subset P$ is a subset of points that are homogeneous with respect to local geometric features and radiometric information. Superpoints serve as the basic computational units in our framework, significantly reducing the complexity from point-level to superpoint-level processing.

Superpoint Graph. We construct a superpoint graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, where $\mathcal{S} = \{S_1, \dots, S_M\}$ denotes the set of superpoints (nodes), and $\mathcal{E}$ represents the edges connecting adjacent superpoints. An edge $e_{ij} \in \mathcal{E}$ exists between superpoints S_i and S_j if they contain spatially adjacent points. Each edge is associated

with features encoding the geometric and semantic relationship between connected superpoints. *Ideally, edges connecting superpoints belonging to the same instance should be preserved during clustering, while those crossing instance boundaries should be cut.*

Subgraph. A subgraph $s \subseteq \mathcal{G}$ is a connected component of the superpoint graph formed by a subset of nodes and their inter-connecting edges. During the hierarchical clustering process, subgraphs represent intermediate cluster candidates that may correspond to complete object instances or require further refinement.

Task Objective. Given a large-scale 3D point cloud P , the goal of 3D panoptic segmentation is to assign each point $p_i \in P$ both a semantic label $cls(p_i) \in \{1, \dots, C\}$ and an instance index $obj(p_i) \in \mathbb{N}$. For "thing" classes (countable objects like cars, trees), each instance receives a unique index. For "stuff" classes (amorphous regions like ground, vegetation), all points of the same class share a common index. Our approach achieves this by progressively clustering superpoints into semantically consistent and spatially coherent instances through hierarchical reinforcement learning.

2.2 Semantic Voxel Partitioning

Directly applying reinforcement learning on raw point clouds is computationally prohibitive due to the massive state space. To address this, we introduce Semantic Voxel Partitioning (SVP), a preprocessing module designed to generate semantically consistent superpoints by grouping spatially adjacent points with identical semantic labels. Unlike traditional geometric over-segmentation methods (Landrieu and Boussaha, 2019) that rely solely on spatial proximity, SVP explicitly incorporates semantic information to ensure that superpoint boundaries align with object contours.

2.2.1 Voxelization and Semantic Labeling Given a raw point cloud $P = \{p_1, \dots, p_N\} \subset \mathbb{R}^3$, where N denotes the total number of points and each point p_i has coordinates $\text{pos}_i \in \mathbb{R}^3$, we first discretize the space into a regular voxel grid with resolution v_s . Each point p_i is assigned to a voxel indexed by:

$$\mathbf{v}(p_i) = \left\lfloor \frac{\text{pos}_i}{v_s} \right\rfloor \in \mathbb{Z}^3. \quad (1)$$

To enrich voxels with semantic context, we employ a pre-trained Minkowski UNet (Choy et al., 2019) to predict a semantic label $l_i \in \{1, \dots, C\}$ for each point p_i , where C is the number of semantic classes. For each voxel v , we compute its dominant semantic label as the mode of labels of contained points:

$$l(v) = \text{mode}\{l_i \mid \mathbf{v}(p_i) = v\}. \quad (2)$$

2.2.2 Semantic-Connected Component Labeling The core of SVP lies in grouping voxels into superpoints based on two criteria: (i) spatial adjacency and (ii) semantic consistency. Let $\mathcal{V} = \{v_1, \dots, v_K\}$ denote the set of K occupied voxels. We define two voxels v_i and v_j as *semantically connected* if they satisfy:

$$\text{adj}(v_i, v_j) = 1 \quad \text{and} \quad l(v_i) = l(v_j), \quad (3)$$

where $\text{adj}(v_i, v_j) = 1$ indicates that v_i and v_j are 6-connected neighbors (sharing a face).

We then perform connected component labeling on the voxel grid to identify maximal sets of semantically connected voxels. Formally, a superpoint S_k is defined as a connected component:

$$S_k = \{p \in P \mid \mathbf{v}(p) \in \mathcal{C}_k\}, \quad (4)$$

where $\mathcal{C}_k \subset \mathcal{V}$ is the k -th connected component satisfying the semantic connectivity constraint. This process generates a superpoint set $\mathcal{S} = \{S_1, \dots, S_M\}$, where M denotes the total number of superpoints and typically $M \ll N$.

2.2.3 Graph Construction The resulting superpoints serve as the nodes $\mathcal{S}$ of our initial superpoint graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$. An edge $e_{ij} \in \mathcal{E}$ is established between superpoints S_i and S_j if they contain spatially adjacent voxels, representing potential instance boundaries. Each node S_i is associated with: (i) a semantic label $l(S_i)$ inherited from its constituent voxels, and (ii) geometric features such as centroid position and bounding box dimensions. This graph structure serves as the input state for the Hierarchical Reinforcement Learning agent. By operating on $\mathcal{G}$ instead of the raw point cloud P , we reduce the computational complexity from $\mathcal{O}(N)$ to $\mathcal{O}(M)$, enabling the agent to efficiently explore clustering policies on large-scale scenes during the subsequent training phase.

In essence, Semantic Voxel Partitioning acts as an intelligent state-space compressor. By transforming millions of raw points into a compact set of M semantically pure superpoint nodes ($M \ll N$), SVP eliminates redundant geometric noise. This semantic-aware initialization provides the subsequent HRL agent with a robust, highly discriminative graph structure, which is the absolute prerequisite for stable and scalable reinforcement learning optimization.

2.3 Hierarchical Reinforcement Learning

Given the initial superpoint graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$ produced by SVP, SPARC performs multi-step clustering through a dual-level Hierarchical Reinforcement Learning (HRL) agent. We model the clustering process as a Markov Decision Process (MDP) and optimize the policy using Proximal Policy Optimization (PPO) (?). Our key insight is that effective panoptic segmentation requires a coarse-to-fine strategy: large objects should be separated at a global level before refining boundaries between smaller components. Our HRL architecture explicitly encodes this hierarchical prior through a meta-controller that decides when to terminate clustering and a low-level controller that makes fine-grained edge decisions.

2.3.1 MDP Formulation To mathematically enable sequential reasoning, we cast the graph clustering process as an episodic Markov Decision Process (MDP), where the state represents the current instance partition and actions dictate local boundary refinements. We formalize this task as an MDP tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where:

State Space ($\mathcal{S}$). At each timestep t , the state s_t is a subgraph of $\mathcal{G}$ representing a candidate instance cluster. Formally, $s_t = (\mathcal{V}_t, \mathcal{E}_t)$ where $\mathcal{V}_t \subseteq \mathcal{S}$ denotes the set of superpoints in the current cluster and $\mathcal{E}_t$ contains edges between them. Each node $v \in \mathcal{V}_t$ is associated with a feature vector $\mathbf{x}_v \in \mathbb{R}^{d_n}$ encoding geometric properties (e.g., centroid position, bounding box dimensions) and semantic predictions from the Supertransformer(SPT) backbone. Each edge $e_{ij} \in \mathcal{E}_t$ has features $\mathbf{e}_{ij} \in \mathbb{R}^{d_e}$ capturing relative position, feature difference, and semantic compatibility.

Action Space ($\mathcal{A}$). The action space is hierarchical. The high-level action $a_H \in \{\text{terminate}, \text{continue}\}$ is discrete, determining whether the current subgraph forms a complete instance. If $a_H = \text{continue}$, the low-level policy outputs continuous actions for each edge: $a_{L,ij}$, representing the affinity modulation signal. A positive value increases the edge affinity (promoting merging), while a negative value decreases it (promoting cutting). This continuous formulation allows for fine-grained control over the clustering process compared to hard binary decisions.

State Transition ($\mathcal{P}$). Executing a low-level action modifies the graph structure: cutting edge e_{ij} partitions the subgraph into connected components, each becoming a new state for further processing. The meta-controller's terminate action removes the subgraph from the active queue and outputs it as a final instance.

Discount Factor (γ). We set $\gamma = 0.99$ to encourage long-term planning, ensuring that early decisions account for their impact on final panoptic segmentation quality.

2.3.2 Hierarchical Policy Architecture Our hierarchical reinforcement learning agent employs a dual-level policy architecture to efficiently handle the multi-scale nature of 3D panoptic segmentation. The architecture consists of three key components: (i) a graph encoder that extracts features from superpoint graphs, (ii) a high-level meta-controller that makes coarse-grained decisions about instance completeness, and (iii) a low-level controller that performs fine-grained edge-level actions. Both policies are optimized jointly using Proximal Policy Optimization (PPO) with a hybrid reward mechanism that balances local action correctness and global segmentation quality.

Graph Encoder. Both policy levels share a Graph Neural Network encoder F_{enc} based on GraphSAGE (Hamilton et al., 2017). The encoder processes a subgraph $s = (\mathcal{V}_s, \mathcal{E}_s)$ extracted from the initial superpoint graph, where $\mathcal{V}_s$ denotes the set of superpoint nodes and $\mathcal{E}_s$ represents edges between adjacent superpoints. Each node $v \in \mathcal{V}_s$ is associated with a feature vector $\mathbf{x}_v \in \mathbb{R}^{d_n}$ encoding geometric properties (centroid position, bounding box dimensions) and semantic predictions from the SPT backbone. Each edge $e_{uv} \in \mathcal{E}_s$ has features $\mathbf{e}_{uv} \in \mathbb{R}^{d_e}$ capturing relative position, feature difference, and semantic compatibility between connected superpoints.

The encoder performs $L = 3$ layers of message passing to propagate information across the subgraph structure:

$$\mathbf{h}_v^{(l+1)} = \sigma \left(\mathbf{W}_1^{(l)} \mathbf{h}_v^{(l)} + \mathbf{W}_2^{(l)} \cdot \text{mean}_{u \in \mathcal{N}(v)} \left[\mathbf{h}_u^{(l)} \odot \phi(\mathbf{e}_{uv}) \right] \right), \quad (5)$$

where $\mathbf{h}_v^{(l)}$ is the hidden state of node v at layer l , $\mathcal{N}(v)$ denotes the set of neighboring nodes, ϕ is an MLP projecting edge features to match node dimensions, and $\odot$ represents element-wise multiplication. After L layers of aggregation, the final node embeddings capture both local geometric cues and global contextual information within the subgraph.

To obtain a single representation for the entire subgraph, we apply global mean pooling over all node embeddings:

$$\mathbf{g}_s = \frac{1}{|\mathcal{V}_s|} \sum_{v \in \mathcal{V}_s} \mathbf{h}_v^{(L)}. \quad (6)$$

The resulting graph-level embedding $\mathbf{g}_s \in \mathbb{R}^{d_g}$ serves as a shared representation for both the policy and value networks

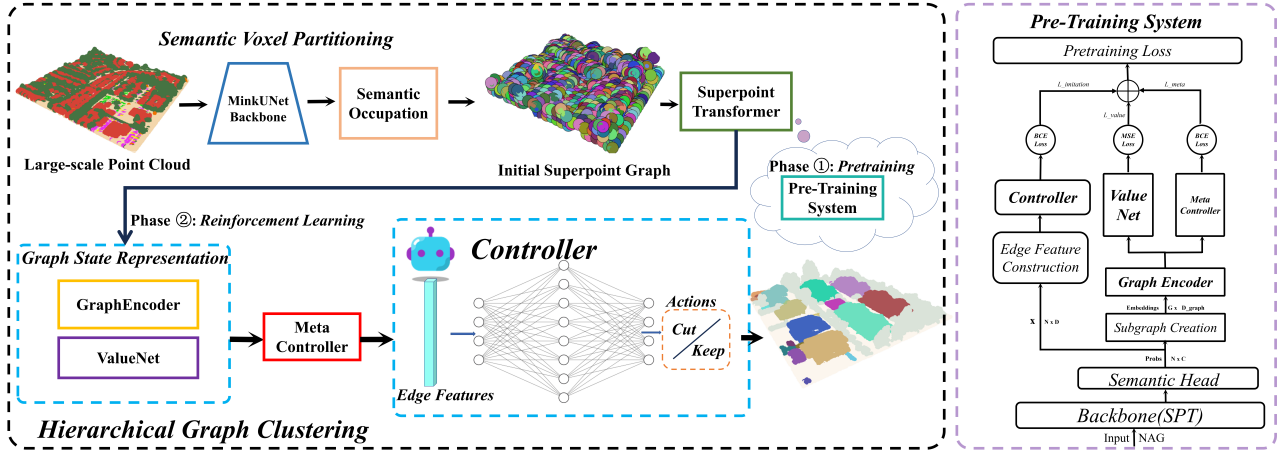


Figure 2. Overview of the SPARC framework.

within the meta-controller. The *policy network* determines the termination action based on semantic purity, while the *value network* estimates the expected cumulative reward to facilitate stable optimization during training.

Meta-Controller (High-Level Policy). The meta-controller π_H decides whether to terminate clustering for subgraph s . It processes $\mathbf{g}_s$ through a two-layer MLP F_{meta} with ReLU activations:

$$P(a_H = \text{terminate} | s) = \sigma(F_{meta}(\mathbf{g}_s)), \quad (7)$$

where σ is the sigmoid function. To encourage semantic consistency, we augment $\mathbf{g}_s$ with the entropy of predicted semantic distributions within s ; high entropy indicates mixed semantics and discourages premature termination. Concurrently, a value head F_{value} estimates the state value $V(s) = F_{value}(\mathbf{g}_s)$ used for advantage estimation in PPO.

Controller (Low-Level Policy). When refinement is needed, the low-level policy π_L evaluates each edge e_{ij} to generate a modulation signal for its affinity. Instead of making a hard “cut” or “keep” decision, the controller adjusts the edge weight to encourage or discourage merging. The edge state $\mathbf{s}_{ij}$ concatenates: (i) the absolute difference and average of node features, (ii) the semantic probability difference:

$$\mathbf{s}_{ij} = \text{concat} \left(|\mathbf{x}_i - \mathbf{x}_j|, \frac{\mathbf{x}_i + \mathbf{x}_j}{2}, |\mathbf{p}_i - \mathbf{p}_j| \right), \quad (8)$$

where $\mathbf{p}_i$ is the semantic probability vector. The policy outputs a continuous modulation signal Δw_{ij} via a three-layer MLP F_{low} with a tanh activation to bound the output:

$$\Delta w_{ij} = \alpha \cdot \tanh(F_{low}(\mathbf{s}_{ij})), \quad (9)$$

where α is a scaling factor. A positive Δw_{ij} increases the edge affinity (promoting merging), while a negative value decreases the affinity (promoting cutting). The updated affinity $w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}$ is then used in the graph clustering optimization. This continuous modulation allows for finer-grained control over boundary refinement compared to binary actions.

Hierarchical Interaction. The two policies operate in a nested loop: the meta-controller evaluates the current subgraph; if continue is selected, the controller processes all edges in parallel; the resulting cuts partition the graph, and each component is recursively evaluated. This design enables adaptive computation:

Algorithm 1 Sequential Clustering with HRL

Require: Superpoint graph $\mathcal{G}$, policies π_H, π_L

Ensure: Set of instance clusters $\mathcal{I}$

```

1:  $\mathcal{Q} \leftarrow \{\mathcal{G}\}; \mathcal{I} \leftarrow \emptyset$ 
2: while  $\mathcal{Q} \neq \emptyset$  do
3:    $s \leftarrow \text{pop}(\mathcal{Q})$ 
4:   if  $\pi_H(s) = \text{terminate}$  then
5:      $\mathcal{I} \leftarrow \mathcal{I} \cup \{s\}$ 
6:   else
7:      $\mathcal{E}_{cut} \leftarrow \{e_{ij} \mid \pi_L(e_{ij}) = \text{cut}\}$ 
8:      $\mathcal{C} \leftarrow \text{ConnectedComponents}(s \setminus \mathcal{E}_{cut})$ 
9:      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \mathcal{C}$ 
10:  end if
11: end while
12: return  $\mathcal{I}$ 

```

simple regions terminate early, while ambiguous boundaries receive iterative refinement.

2.3.3 Sequential Clustering Execution Algorithm 1 outlines the inference procedure. Starting from the full superpoint graph, we maintain a queue of active subgraphs. At each iteration, we sample a subgraph and query the meta-controller. If terminate is selected, the subgraph is added to the output instances. Otherwise, the controller predicts edge actions, the graph is partitioned along cut edges, and new components are enqueued. The process terminates when the queue is empty.

For “stuff” classes (e.g., ground, vegetation), we skip instance splitting: subgraphs predicted as stuff are directly output as single regions. This respects the panoptic segmentation definition where stuff classes share a common instance index.

2.3.4 Graph Clustering via Cut-Pursuit The final instance segmentation is obtained by applying the Cut-Pursuit algorithm (Landrieu and Obozinski, 2017) to the modified superpoint graph. Cut-Pursuit is a working-set algorithm that minimizes functions regularized by graph-structured total variation, producing piecewise constant functions on general weighted graphs.

Given the adjusted edge affinities $\mathbf{w}^{(t)}$ from our hierarchical RL agent, Cut-Pursuit solves the following optimization problem:

$$\min_{\mathbf{u} \in \mathbb{R}^{M \times d}} \sum_{i=1}^M \|\mathbf{x}_i - \mathbf{u}_i\|^2 + \lambda \sum_{(i,j) \in \mathcal{E}} w_{ij}^{(t)} \|\mathbf{u}_i - \mathbf{u}_j\|, \quad (10)$$

where $\mathbf{x}_i$ represents the feature vector of superpoint S_i , $\mathbf{u}_i$ is the

optimized piecewise constant approximation, and λ controls the regularization strength.

The algorithm operates iteratively through two key steps: (i) *reduction* - solving a simplified optimization problem on the current partition, and (ii) *expansion* - identifying potential splits by computing minimum cuts on the graph. This process continues until convergence, yielding a partition where each connected component corresponds to a predicted instance.

Non-differentiability and RL Integration. Compared to simple threshold-based cutting, Cut-Pursuit ensures global energy minimization, guaranteeing that the final partitions are spatially coherent. Our RL agent thus focuses on guiding this global optimizer via affinity modulation, rather than making hard binary cuts locally. This precludes end-to-end gradient-based optimization. Instead, our hierarchical RL agent learns to modulate edge affinities through trial-and-error interactions, receiving feedback via the hybrid reward mechanism described in Section 2.3.5. The RL policy implicitly learns to adjust affinities such that the subsequent Cut-Pursuit optimization produces high-quality instance partitions, effectively bridging the gap between continuous affinity modulation and discrete segmentation outcomes.

2.3.5 Hybrid Reward Mechanism Training HRL policies requires careful reward design to balance local correctness and global segmentation quality. We employ a hybrid reward combining intrinsic and extrinsic signals.

Intrinsic Reward. For each low-level action on edge e_{ij} , we compute an immediate reward based on ground-truth affinity $\hat{a}_{ij}$:

$$r_{int}(a_{L,ij}) = \mathbb{I}[a_{L,ij} = \hat{a}_{ij}] \cdot w_{ij}, \quad (11)$$

where $\mathbb{I}$ is the indicator function and w_{ij} weights edges by their geometric significance (longer edges receive higher weight). This dense reward provides frequent feedback during early training.

Extrinsic Reward. To encourage globally coherent segmentation, we compute a sparse reward after each episode based on Panoptic Recognition Quality (RQ). Let P_{final} be the predicted partition and P_{base} a baseline obtained by single-threshold clustering. The extrinsic reward is:

$$r_{ext} = \text{RQ}(P_{final}) - \text{RQ}(P_{base}), \quad (12)$$

where RQ measures the fraction of true instances correctly detected and classified. This reward aligns policy optimization with the ultimate evaluation metric.

Reward Aggregation. The total reward for a trajectory τ is:

$$R(\tau) = \sum_t \gamma^t \left(\lambda_{int} \sum_e r_{int}^{(t)}(e) + \lambda_{ext} \cdot r_{ext} \right), \quad (13)$$

with $\lambda_{int} = 0.3$ and $\lambda_{ext} = 0.7$ balancing immediate and long-term objectives. We normalize rewards per batch to stabilize training.

2.3.6 Optimization with Proximal Policy Optimization We optimize both policies jointly using PPO (Schulman et al., 2017), which provides stable updates for continuous action

spaces. The objective function is:

$$\mathcal{L}^{PPO}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t \right) \right], \quad (14)$$

where $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{old}}(a_t|s_t)$ is the probability ratio, $\hat{A}_t$ is the Generalized Advantage Estimate (GAE) (Schulman et al., 2015), and $\epsilon = 0.2$ controls the clipping range.

Advantage Estimation. We compute advantages using GAE with $\lambda = 0.95$:

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma \lambda)^l \delta_{t+l}, \quad \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t), \quad (15)$$

where V is a learned value function sharing the graph encoder with the policy.

Training Procedure. We train on mini-batches of 32 trajectories sampled from diverse scenes. Each trajectory is generated by running the current policy on a randomly sampled subgraph. We use AdamW optimizer with learning rate 3×10^{-4} , gradient clipping at norm 0.5, and entropy regularization ($\beta = 0.01$) to encourage exploration. The meta-controller and controller share the graph encoder but have separate policy heads; we update them with separate PPO objectives but shared experience replay.

Efficiency Considerations. To scale to large scenes, we employ three strategies: (i) parallel evaluation of independent subgraphs across GPU cores; (ii) early termination for high-confidence predictions (meta-controller probability > 0.9); (iii) edge action batching via vectorized MLP inference.

2.3.7 Summary of HRL Design. In summary, by formulating the clustering task as an MDP, designing a hierarchical policy for multi-scale decision making, and employing a hybrid reward mechanism guided by PPO, the HRL module achieves adaptive optimization of the graph clustering process. This ensures that local edge adjustments contribute positively to the global panoptic segmentation quality, effectively addressing the **Short-sighted decisions** discussed in the introduction.

2.4 Pre-Training System

To stabilize the subsequent reinforcement learning optimization and provide meaningful initialization for the hierarchical policies, we introduce a supervised pre-training phase. This phase serves three critical purposes: (i) teaching the controller to recognize semantically pure subgraphs, (ii) enabling the value network to estimate state quality before RL exploration begins, and (iii) training the meta-controller to make informed termination decisions based on ground-truth instance boundaries.

Architecture and Loss Functions. As illustrated in Figure 2, the pre-training system shares the same backbone architecture as the main HRL framework but optimizes components using supervised signals. The Graph Encoder processes subgraphs created from ground-truth instances, producing embeddings that feed into three parallel heads: (i) the Controller predicts edge affinities using Mean Squared Error (MSE) loss against ground-truth edge labels, (ii) the Value Net estimates state values using MSE loss against discounted cumulative rewards computed from ground-truth partitions, and (iii) the Meta-Controller predicts termination actions using BCE loss

Table 1. Performance comparison on the DALES dataset.

Model	SS	PS			
	mIoU	mIoU (IS)	PQ	RQ	SQ
ConvPoint(Boulch, 2020)	67.4	—	—	—	—
PointNet++(Qi et al., 2017)	68.3	—	—	—	—
KPConv(Thomas et al., 2019)	81.1	—	—	—	—
SuperCluster(Robert et al., 2024)	73.6	75.1	52.6	59.1	85.9
SPARC (ours)	79.4	78.6	62.4	69.0	88.9

Table 2. Performance comparison on the Kitti360 dataset.

Model	SS	PS			
	mIoU	mIoU (IS)	PQ	RQ	SQ
Minkowski(Choy et al., 2019)	50.6	—	—	—	—
DeepViewAgg(Cui et al., 2021)	54.8	—	—	—	—
SPT(Robert and Raguét, 2023)	55.20	—	—	—	—
SuperCluster(Robert et al., 2024)	54.13	53.75	37.66	45.47	68.84
SPARC (ours)	54.58	53.60	37.15	47.50	63.97

against instance completeness labels. The total pre-training loss is formulated as:

$$\mathcal{L}_{pre} = \lambda_{imitation}\mathcal{L}_{imitation} + \lambda_{value}\mathcal{L}_{value} + \lambda_{meta}\mathcal{L}_{meta}, \quad (16)$$

where $\lambda_{imitation} = 0.4$, $\lambda_{value} = 0.3$, and $\lambda_{meta} = 0.3$ balance the three objectives.

Training Procedure. We sample subgraphs from ground-truth instances and construct edge labels based on whether connected superpoints belong to the same instance. The Value Net targets are computed as the Panoptic Quality (RQ) improvement achievable from each state. This imitation learning phase runs for 100 epochs before switching to PPO-based RL optimization, ensuring that the policies start from a reasonable operating point rather than random initialization.

In essence, the Pre-Training System acts as a knowledge injection mechanism. By learning from ground-truth partitions before exploration, the hierarchical policies acquire fundamental understanding of semantic purity, boundary quality, and instance completeness. This warm-start significantly reduces the sample complexity of subsequent RL training and prevents early-stage instability that could arise from purely random action selection.

2.5 Pipeline Summary

Our proposed SPARC framework operates as a dynamic, coarse-to-fine engine: SVP first provides a compressed semantic superpoint graph; the Meta-Controller then parses the scene globally to isolate simple, complete instances; simultaneously, the Low-Level Controller iteratively focuses on ambiguous boundaries, gently adjusting affinities where scale variance causes confusion. Guided by the hybrid PPO reward, this progressive refinement directly optimizes the final panoptic quality, elegantly bridging the gap between local point features and global instance coherence.

3. Experiments

3.1 Configuration

DALES (Varney et al., 2020) contains millions of points from aerial LiDAR scans, with diverse object scales and complex outdoor scenes. KITTI-360 (Liao et al., 2022) comprises large-scale point clouds from ground-based mobile scanning, with diverse urban instances and complex street-level environments. To

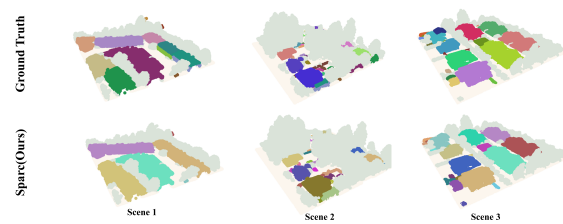


Figure 3. Qualitative visualization on 3 representative scenes

ensure a fair comparison, we re-evaluated all competing methods under a unified setup using a single NVIDIA A800-SXM4-80GB GPU.

3.2 Main Results

To demonstrate its practical performance, the quantitative results are shown in Table 1. SPARC achieves state-of-the-art performance, reaching a PQ of 62.4%, which is a 9.8% improvement over the strong SuperCluster baseline. The significant margin clearly validates our central hypothesis: recasting panoptic segmentation as a sequential decision problem and solving it with a hierarchical reinforcement learning agent provides a highly effective strategy for complex scenes. Figure 3 shows SPARC resolves ambiguous boundaries, resulting in qualitatively accurate panoptic segmentation.

3.3 Ablation Study

To validate the effectiveness of our core designs, we conduct extensive ablation studies on the DALES dataset. Specifically, we investigate the impact of the SVP module and the HRL framework.

Impact of Semantic Voxel Partitioning (SVP). As shown in Table 3, compared to the SuperCluster baseline which relies solely on geometric clustering, our Baseline (w/ SVP) significantly improves the panoptic quality (PQ) from 54.60% to 61.29%. This substantial gain of 6.69% confirms our hypothesis: incorporating semantic priors directly into the primitive generation phase effectively suppresses ambiguous boundaries and prevents catastrophic errors early in the pipeline.

Impact of Hierarchical Reinforcement Learning (HRL). Replacing the flat single-step optimization with our sequential HRL framework further pushes the performance boundaries. The full SPARC model achieves the highest PQ of 62.40%, outperforming the Baseline (w/ HRL but without SVP) and the Baseline (w/ SVP but without HRL). The progressive refinement driven by the Meta-Controller and Low-Level policy specifically tackles multi-scale object interference, demonstrating that long-horizon reward assignment leads to far superior instance integrity than greedy, short-sighted edge predictions.

3.4 Cross-Modality Analysis

While SPARC achieves substantial gains on the aerial LiDAR dataset (DALES), as shown in Table 2, its improvement on the mobile laser scanning dataset (KITTI-360) is more modest, with a slight drop in PQ (37.15% vs. 37.66% of SuperCluster) despite better RQ (47.50% vs. 45.47%). We attribute this modality gap to three fundamental factors:

Perspective-Induced Occlusion. Aerial LiDAR (ALS) provides a top-down view with minimal occlusion, allowing SVP to generate semantically pure superpoints by grouping

spatially adjacent voxels with consistent labels. In contrast, mobile scanning (MLS) suffers from severe side-view occlusions (e.g., buildings blocking vehicles, pedestrians behind poles), causing SVP to fragment object instances into disconnected components. This fragmentation propagates to the HRL stage, where the meta-controller may prematurely terminate clustering on incomplete subgraphs.

Density Heterogeneity and Edge Signal Stability. ALS point clouds exhibit relatively uniform density across large areas, enabling stable edge affinity modulation by the low-level controller. MLS data, however, shows extreme density variation: dense near the sensor and sparse at distance. In sparse regions, the geometric features used for edge state representation (s_{ij}) become noisy, leading to unstable Δw_{ij} predictions and suboptimal Cut-Pursuit optimization.

Scale Distribution Mismatch. The hierarchical design of SPARC assumes a coarse-to-fine refinement strategy that prioritizes separating large objects first. This prior aligns well with ALS scenes dominated by buildings and terrain. MLS scenes, however, contain dense clusters of small-scale objects (e.g., traffic signs, cyclists) that require fine-grained decisions from the outset. Our current meta-controller may not adapt greatly to this reversed scale priority.

Implications and Future Directions. These observations suggest two promising extensions: (i) incorporating view-aware augmentation in SVP to improve robustness to occlusion patterns, and (ii) designing a scale-adaptive meta-controller that dynamically adjusts its termination threshold based on local point density and object size estimates. We leave these investigations to future work.

Table 3. Performance comparison on the DALES dataset.

Model	DALES		Panoptic Quality (%)		
	mIoU (SS)	mIoU (IS)	PQ	RQ	SQ
SuperCluster (Robert et al., 2024)	74.70	75.10	54.60	62.40	84.50
Baseline (w/ SVP)	79.09	76.20	61.29	68.32	88.29
Baseline (w/ HRL)	78.92	77.35	61.85	68.68	88.52
SPARC (ours)	79.38	78.57	62.40	69.04	88.85

4. Conclusion

In this work, we propose SPARC, the first framework to apply Hierarchical Reinforcement Learning to large-scale 3D panoptic segmentation. By reformulating instance merging as a sequential decision process, SPARC effectively overcomes the intrinsic limitations of conventional greedy clustering, especially in scenes with extreme object-scale variation. The integration of the Semantic Voxel Partitioning for high-quality superpoint initialization and a dual-level Hierarchical Reinforcement Learning policy for iterative refinement forms a highly effective and scalable pipeline. Our state-of-the-art performance on the DALES dataset demonstrates the strength of this design and highlights the broader potential of reinforcement learning as a compelling new paradigm for complex 3D perception tasks.

References

Aygun, M., Osep, A., Weber, M., Maximov, M., Stachniss, C., Behley, J., Leal-Taixé, L., 2021. 4d panoptic lidar segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5527–5537.

Boulch, A., 2020. ConvPoint: Continuous convolutions for point cloud processing. *Computers & Graphics*, 88, 24 - 34.

Choy, C., Gwak, J., Savarese, S., 2019. 4d spatio-temporal convnets: Minkowski convolutional neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3075–3084.

Cui, J., Zhong, Z., Liu, S., Yu, B., Jia, J., 2021. Parametric contrastive learning. *Proceedings of the IEEE/CVF international conference on computer vision*, 715–724.

Fong, W. K., Mohan, R., Hurtado, J. V., Zhou, L., Caesar, H., Beijbom, O., Valada, A., 2022. Panoptic nusenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *IEEE Robotics and Automation Letters*, 7(2), 3795–3802.

Hamilton, W., Ying, Z., Leskovec, J., 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hauglin, M., Rahlf, J., Schumacher, J., Astrup, R., Breidenbach, J., 2021. Large scale mapping of forest attributes using heterogeneous sets of airborne laser scanning and National Forest Inventory data. *Forest Ecosystems*, 8, 65.

Jiang, Y., Yin, S., Li, K., Luo, H., Kaynak, O., 2021. Industrial applications of digital twins. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2207), 20200360. <https://doi.org/10.1098/rsta.2020.0360>.

Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P., 2019. Panoptic segmentation. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9404–9413.

Lai, X., Liu, J., Jiang, L., Wang, L., Zhao, H., Liu, S., Qi, X., Jia, J., 2022. Stratified transformer for 3d point cloud segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8500–8509.

Landrieu, L., Boussaha, M., 2019. Point cloud oversegmentation with graph-structured deep metric learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7440–7449.

Landrieu, L., Obozinski, G., 2017. Cut Pursuit: fast algorithms to learn piecewise constant functions on general weighted graphs. *SIAM Journal on Imaging Sciences*, Vol. 10(No. 4), pp. 1724–1766. <https://hal.science/hal-01306779>.

Liao, Y., Xie, J., Geiger, A., 2022. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3), 3292–3310.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv preprint arXiv:1706.02413*.

Quattrocchi, C., Di Mauro, D., Furnari, A., Farinella, G. M., 2022. Panoptic segmentation in industrial environments using synthetic and real data. *International Conference on Image Analysis and Processing*, Springer, 275–286.

Robert, D., Raguét, H., 2023. Efficient 3D Semantic Segmentation with Superpoint Transformer. *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Robert, D., Raguet, H., Landrieu, L., 2024. Scalable 3D Panoptic Segmentation As Superpoint Graph Clustering. *Proceedings of the IEEE International Conference on 3D Vision*.

Schulman, J., Moritz, P., Levine, S., Jordan, M., Abbeel, P., 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Smeeckaert, J., Mallet, C., David, N., Chehata, N., Ferraz, A., 2013. Large-scale classification of water areas using airborne topographic lidar data. *Remote sensing of environment*, 138, 134–148.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L. J., 2019. KPConv: Flexible and Deformable Convolution for Point Clouds. *Proceedings of the IEEE International Conference on Computer Vision*.

Varney, N., Asari, V. K., Graehling, Q., 2020. Dales: A large-scale aerial lidar data set for semantic segmentation. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 186–187.

Wang, C., Wen, C., Dai, Y., Yu, S., Liu, M., 2020. Urban 3D modeling using mobile laser scanning: A review. *Virtual Reality & Intelligent Hardware*, 2(3), 175–212.

Zhao, H., Jiang, L., Jia, J., Torr, P. H., Koltun, V., 2021. Point transformer. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16259–16268.