4DSegStreamer: Streaming 4D Panoptic Segmentation via Dual Threads

Ling Liu[1*]   Jun Tian[1*]   Li Yi[1,2,3†]

[1]IIIS, Tsinghua University

[2]Shanghai Qi Zhi Institute   [3]Shanghai AI Lab

https://llada60.github.io/4DSegStreamer

Abstract

4D panoptic segmentation in a streaming setting is critical for highly dynamic environments, such as evacuating dense crowds and autonomous driving in complex scenarios, where real-time, fine-grained perception within a constrained time budget is essential. In this paper, we introduce 4DSegStreamer, a novel framework that employs a Dual-Thread System to efficiently process streaming frames. The framework is general and can be seamlessly integrated into existing 3D and 4D segmentation methods to enable real-time capability. It also demonstrates superior robustness compared to existing streaming perception approaches, particularly under high FPS conditions. The system consists of a predictive thread and an inference thread. The predictive thread leverages historical motion and geometric information to extract features and forecast future dynamics. The inference thread ensures timely prediction for incoming frames by aligning with the latest memory and compensating for ego-motion and dynamic object movements. We evaluate 4DSegStreamer on the indoor HOI4D dataset and the outdoor SemanticKITTI and nuScenes datasets. Comprehensive experiments demonstrate the effectiveness of our approach, particularly in accurately predicting dynamic objects in complex scenes.
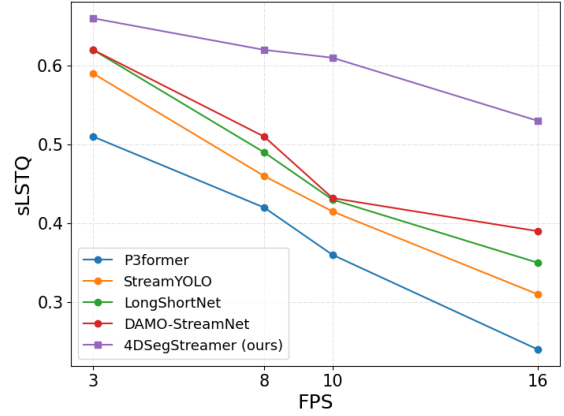
Figure 1. Comparison of streaming performance at different FPS settings on the SemanticKITTI dataset. Our 4DSegStreamer demonstrates significant performance gains and exhibits a slower performance decline as the FPS increases, indicating its robustness as a more advanced 4D streaming system for panoptic segmentation tasks, particularly in high-FPS scenarios.

1. Introduction

Map-free autonomous agents operating in highly dynamic environments require a comprehensive understanding of their surroundings and rapid response capabilities, essential for tasks such as outdoor autonomous driving and indoor robotic manipulation. While low latency may not be critical in static or map-available settings, it becomes a significant challenge in dynamic, map-free environments, where effective navigation and interaction rely on real-time perception. The primary goal of streaming perception is to generate accurate predictions for each incoming frame within a limited time budget, ensuring that perception results remain up-to-date and relevant to the current state of the environment.

Existing streaming perception research mainly focuses on tasks such as 2D object detection [13, 14, 17–20, 23, 26, 44, 46], 2D object tracking [22, 33], and 3D object detection [1, 6, 12, 16, 24, 37] in autonomous driving application, aiming to balance accuracy and latency. However, object bounding boxes are usually insufficient to provide finer-grained knowledge like the object shape or scene context, which is critical for downstream decision-making. For instance, in autonomous driving, relying solely on object detection does not allow the system to accurately identify areas like construction zones or sidewalks, which are essential to avoid for safe navigation.

To achieve a more comprehensive understanding of the

---

[*]Equal contribution
[†]Corresponding author

scene in a streaming setup, we focus on the challenging task of streaming 4D panoptic segmentation. Given a streaming sequence of point clouds, the goal is to predict panoptic segmentation on each frame within a strict time budget, enabling real-time scene perception. This task is particularly difficult due to the computational overhead and fine-grained perception requirements. Most existing 4D methods [2, 8–11, 21, 25, 29, 34, 39, 40, 43, 45, 47, 48] fail to achieve real-time perception and the fluctuations in computing resources introduce additional latency inconsistencies, further complicating streaming 4D panoptic segmentation task.

To address the challenges of real-time dense perception in streaming 4D panoptic segmentation, we introduce 4DSegStreamer, a general system designed to enable existing segmentation methods to operate in real time. 4DSegStreamer utilizes a novel dual-thread system with a predictive thread maintaining geometry and motion memories in the scene and an inference thread facilitating rapid inference at each time step. The key idea behind 4DSegStreamer involves dividing the streaming input into key frames and non-key frames based on the model's latency. In the predictive thread, we meticulously compute geometric and motion features at key frames and utilize these features to continuously update the memories, enabling long-term spatial-temporal perception. To support efficient memory queries, the memories are also utilized to predict future dynamics, guiding how a future frame can effectively adjust for potential movement when querying the geometry memory. In the inference thread, each incoming frame is first positionally aligned with the current geometry memory by compensating for the forecasted motion. It then swiftly queries the hash table-style memory to obtain per-point labels. The two threads together allow both fast and high-quality streaming 4D panoptic segmentation.

Our contributions to this work can be summarized as:

- We introduce a new task for streaming 4D panoptic segmentation, advancing real-time, fine-grained perception for autonomous systems in dynamic environments.
- We propose a novel dual-thread system that includes a predictive thread and an inference thread, which is general and applicable to existing segmentation methods to achieve real-time performance. The predictive thread continuously updates memories by leveraging historical motion and geometric features to forecast future dynamics. The inference thread retrieves relevant features from the memory through geometric alignment with the forecasted motion, using ego-pose transformation and inverse flow iteration.
- Through extensive evaluations in outdoor datasets SemanticKITTI and nuScenes, as well as the indoor HOI4D dataset, our system significantly outperforms existing SOTA streaming perception and 4D panoptic segmentation methods. Moreover, our approach demonstrates su-

perior robustness compared to other streaming perception methods shown in Fig. 1, particularly under high-FPS scenarios. These results highlight the effectiveness and value of our method for 4D streaming segmentation.

2. Related Work

2.1. Streaming Perception

In the streaming perception, the inherent challenge lies in predicting results in the future state, in order to minimize the temporal gap between input and output timestep. Most previous studies concentrate on developing forecasting modules specifically tailored for this streaming setting. Stream [26] firstly introduces the streaming setting and utilizes the Kalman Filters to predict future bounding boxes. StreamYOLO [44] designs a dual-flow perception module, which incorporates dynamic and static flows from previous and current features to predict the future state. DAMO-StreamNet [17] and LongShortNet [23] leverages spatial-temporal information by extracting long-term temporal motion from previous multi-frames and short-term spatial information from the current frame for future prediction. Different from previous researches which only forecast one frame ahead and thus the prediction output is limited within a single frame, DaDe [20] and MTD [19] considering previous prediction time, adaptively choose the corresponding future features. Transtreaming [46] designs an adaptive delay-aware transformer to select the prediction from multi-frames future that best matches future time.

Several studies have explored streaming perception in LiDAR-based 3D detection [1, 6, 12, 16, 24, 37]. Lidar Stream [16] segments full-scan LiDAR points into multiple slices, processing each slice at a higher frequency compared to using the full-scan input. Although ASAP [38] introduces a benchmark for online streaming 3D detection, it relies on camera-based methods using images as input.

2.2. 4D Point Cloud Sequence Perception

4D point cloud sequence perception methods integrate temporal consistency and spatial aggregation through advanced memory mechanisms. These methods are generally categorized into voxel-based [8, 25, 43, 45] and point-based [2, 9–11, 21, 21, 28, 29, 31, 34, 39, 40, 47, 48] approaches.

For the point-based methods, SpSequenceNet [34] aggregates 4D information on both a global and local scale through K-nearest neighbours. NSM4D [10] introduces a historical memory mechanism that maintains both geometric and motion features derived from motion flow information, thereby enhancing perception capabilities. Eq-4D-StOP [48] introduces a rotation-equivariant neural network that leverages the rotational symmetry of driving scenarios on the ground plane.

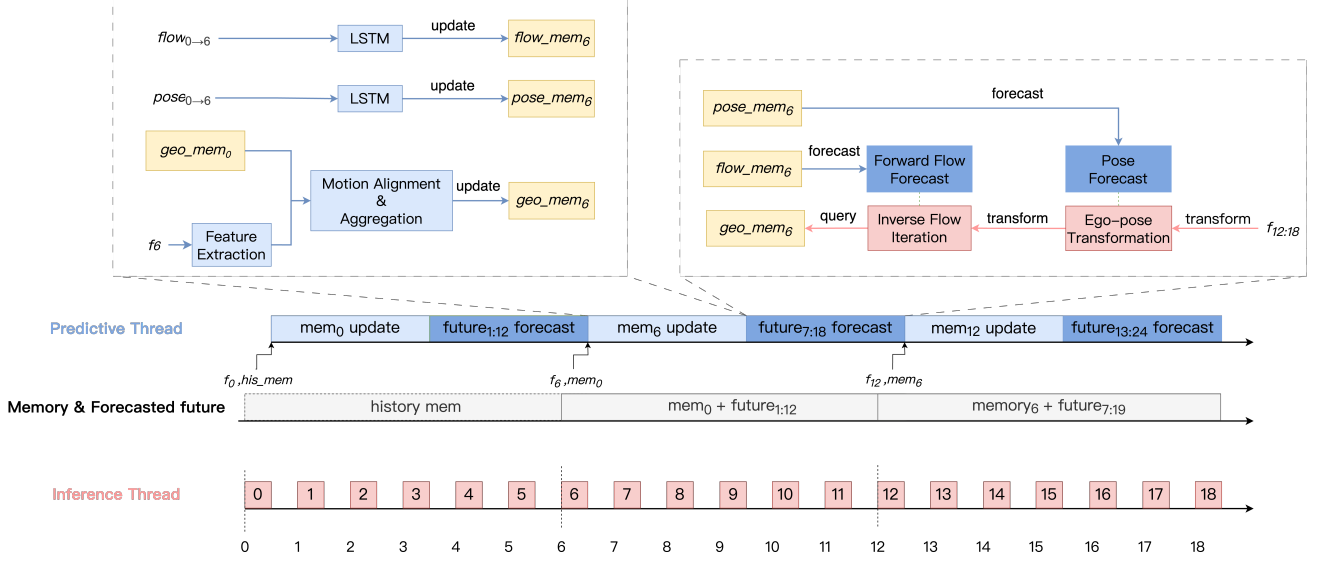For the voxel-based methods, SVQNet [8] develops a

Figure 2. **4DSegStreamer**: The dual-thread system consists of a predictive thread and an inference thread, enabling real-time query for unseen future frames. The predictive thread updates the geometric and motion memories with the latest extracted feature and leverages the historical information to forecast future dynamics. The inference thread retrieves per-point predictions by geometrically aligning them with the current memory using ego-pose and dynamic object alignment. Here, mem_i denotes the memory updated with the latest key frame f_i, while $f_{i:j}$ represents incoming frame$_{i,i+1,...,j}$.

voxel-adjacent framework that leverages historical knowledge with both local and global context understanding. This work is further optimized by the implementation of hash query mechanisms for computation acceleration, and is further accelerated by hash query mechanisms. MemorySeg [25] incorporates both point and voxel representations for contextual and fine-grained details learning. Mask4Former [45] introduces a transformer-based approach unifying semantic instance segmentation and 3D point cloud tracking.

2.3. Fast-slow Dual System Methods

The fast-slow system paradigm, merging efficient lightweight models with powerful large-scale models, has gained attention. For instance, DriveVLM-Dual [35] integrates 3D perception and trajectory planning with VLMs for real-time spatial reasoning, while FASIONAD [32] introduces an adaptive feedback framework for autonomous driving, combining fast and slow thinking to improve adaptability in dynamic environments.

While 4DSegStreamer is not explicitly designed as a fast-slow system, its dual-thread architecture shares some conceptual similarities. The predictive thread acts as a slow component, responsible for maintaining memory and forecasting future dynamics, while the inference thread acts as a fast component, enabling real-time inference through efficient feature retrieval. However, unlike traditional fast-slow systems that rely on separate models for fast and slow tasks, 4DSegStreamer integrates both components into a unified

pipeline, enabling seamless interaction between memory updates and real-time queries.

3. Streaming 4D Panoptic Segmentation

We propose a new task of streaming 4D panoptic segmentation. Similar to the traditional streaming perception paradigm, streaming 4D panoptic segmentation conducts the panoptic segmentation in an online manner. The key challenge is ensuring that each incoming frame is processed and predicted within an ignorant small time budget, even if the processing of the current frame is not complete. Our goal is to develop an approach that finds a trade-off between accuracy and efficiency to enable real-time inference for the Streaming 4D Panoptic Segmentation task.

4. Method

In this section, we introduce 4DSegStreamer (see Fig. 2) to address the challenges of streaming 4D panoptic segmentation. The key idea is to divide the streaming frames into key frames and non-key frames, where geometric and motion features are continuously extracted at key frames to update the memories, and subsequently used to accelerate inference for each future frame. 4DSegStreamer employs a novel dual-thread system comprising a predictive thread and an inference thread, which is general and can be applied to various segmentation methods to enable their real-time performance. The system contains three key stages, including
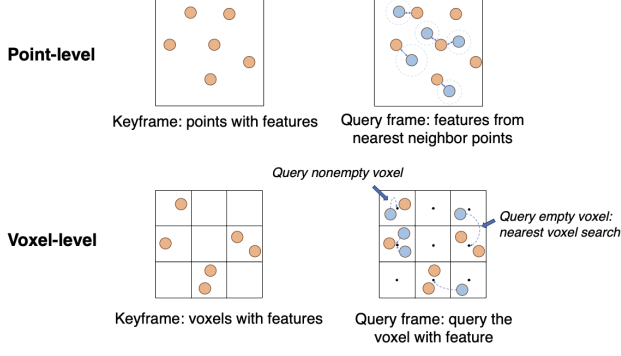
3

Figure 3. Point-level and voxel-level methods in inference thread: orange points indicate the extracted features corresponding keyframe points, while blue points indicate the aligned incoming frame points querying the features from memory.

memory update to maintain spatial-temporal information of geometric and motion features, ego-pose future alignment to cancel ego-motion, and dynamic object future alignment to eliminate dynamic object movement.

4.1. Dual-thread system

Unlike previous works in 2D streaming perception, which focus on object detection and tracking by predicting the transformation of bounding boxes, 4D panoptic segmentation must establish correspondences between past predictions and unseen future point clouds across multiple frames due to the latency. To address this challenge, we simplify the real-time inference problem using a dual-thread system. This system consists of a Predictive Thread for memory updating and future dynamics forecasting and an Inference Thread that allows incoming future points to quickly retrieve the corresponding features from memory, ensuring efficient inference within the limited time constraints.

Predictive thread. We continuously update the geometric and motion memories with the latest available frame as a key frame. Leveraging the spatial-temporal information in the motion memories, we forecast the future camera and dynamic object movement to align future frames with corresponding features in geometric memory, thereby accelerating the inference in the inference thread.

Inference thread. Each incoming frame is geometrically aligned with the latest memory using forecasted pose and flow. The corresponding features are then retrieved from the geometric memory using two query strategies, as illustrated in Fig. 3. In our approach, we use a hash table-style memory that allows direct access to corresponding voxel features via their indices and apply nearest neighbor search only for points querying empty voxels. These retrieved features are subsequently passed through a lightweight prediction head to produce the final output.

The dual-thread system operates in parallel and shares

the memory to process streaming point clouds in real time. The overall inference latency is primarily determined by the inference thread, which is lightweight and fast, while the predictive thread maintains long-term spatio-temporal memories by continuously updating them with the latest features. At each timestamp, the inference thread retrieves relevant features from memory through motion alignment, ensuring real-time inference.

4.2. Geometric Memory Update

Our system is general and can be integrated into both 3D and 4D segmentation backbones, where features are stored at the voxel level for fast query in the inference thread and aggregated to update using the latest keyframe via motion alignment. The memory system leverages a sparse variant of ConvGRU [3, 25] to perform geometric memory updates efficiently.

Upon the arrival of a keyframe, we first perform motion alignment by transforming the previous memory state h_{t-k} to the current frame, resulting in the aligned memory h'_{t-k}:

$$h'_{t-k} = f_{t-k \to t} \left(p_{t-k \to t} \cdot h_{t-k} \right) \tag{1}$$

where $p_{t-k \to t}$ denotes ego-pose transformation and $f_{t-k \to t}$ represents dynamic object flow transformation. Both transformation are applied to convert the memory coordinates into the current keyframe's coordinate space, aligning both static and dynamic objects.

Subsequently, the geometric memory is updated using the current frame's feature embeddings f_t:

$$
\begin{aligned}
z_t &= \sigma(\Psi_z(f_t, h'_{t-k})), \\
r_t &= \sigma(\Psi_r(f_t, h'_{t-k})), \\
\hat{h}_t &= tanh(\Psi_u(f_t, r_t, h'_{t-k})), \\
h_t &= \hat{h}_t \cdot z_t + \hat{h}_{t-k} \cdot (1 - z_t),
\end{aligned}
\tag{2}
$$

where Ψ_r, Ψ_z, Ψ_u are sparse 3D convolution blocks. z_t and r_t are activation gate and reset gate to update and reset the memory. The updated memory retains the latest spatial-temporal information to support future dynamics forecasting and efficient feature queries.

4.3. Ego-pose Future Alignment

As seen in Fig. 4, the static car in the incoming frame is positioned differently from the same car in memory. To ensure temporal consistency in dynamic environments, we utilize ego-pose forecasting to compensate for camera motion and align the current memory with future frames.

In many outdoor applications, such as autonomous driving, ego-pose information is typically available from onboard sensors. However, in indoor scenarios, such as an embodied robot operating in a room, obtaining pose information is often challenging and requires pose estimation.
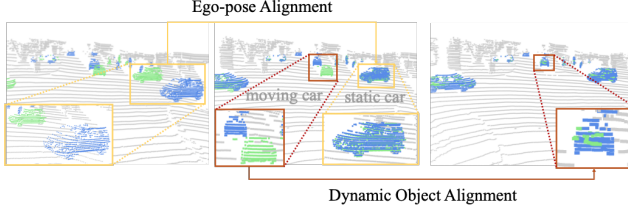
Figure 4. Ego-pose Alignment and Dynamic Object Alignment: The green points represent the previously processed frame that has been used to update the memories and the blue points are the current querying frame. The yellow box highlights static objects that can be aligned through ego-pose alignment. The red box indicates dynamic objects, which require dynamic object alignment to achieve proper alignment.

Depending on whether the camera pose is available, we define two settings:

- *Known pose setting*: we directly use the relative pose to align future frames with the feature memory coordinates.
- *Unknown pose setting*: we utilize the pose estimated by Suma++ [7] between key frames to update the ego-motion memory, and then use the ego-pose forecaster to propagate the future ego-pose motion, ensuring proper alignment and eliminating ego motion.

Here we introduce the unknown pose setting. When a keyframe x_t is coming, the estimator E will estimate the relative ego motion between last keyframe x_{t-k} and current keyframe x_t:

$$p_{t-k \to t} = E(x_{t-k}, x_t) \tag{3}$$

Then, utilize the key pose to update the ego-pose memory $memp_{t-k}$, we have:

$$memp_t = W(p_{t-k \to t}, memp_{t-k}) \tag{4}$$

where W indicates the memory update function which we use the LSTM [15]. In order to forecast the relative pose m frames ahead for the future frame x_{t+m} using pose forecaster F, we have:

$$p_{t \to t+m} = F(memp_t, m) \tag{5}$$

where the ego-pose forecaster is designed in a multi-head structure, with each head predicting the future pose for a fixed number of frames ahead.

4.4. Dynamic Object Future Alignment

Compared to static objects, dynamic objects exhibit both ego-motion and independent self-movement, with varying velocities and directions, as seen in the moving car in Fig. 4. To achieve fine-grained self-motion alignment for dynamic

objects and fast query, we introduce the Future Flow Forecasting in the predictive thread and the Inverse Forward Flow in the inference thread.

Future Flow Forecasting. During training, we use FastNSF [27] to obtain supervised ground truth flows. In inference time, the process is similar to ego-pose future alignment in Sec 4.3. We utilize zeroFlow [36], a lightweight model distilled from FastNSF, to estimate key flows between keyframes. These key flows are then input into the LSTM [15] to forecast future flows, supporting the fast alignment of dynamic objects across memory and incoming frames.

Inverse Forward Flow Iteration. To enable efficient feature querying during inference, we leverage forecasted forward flows to align the geometric memory with future frames. However, directly applying forward flows to the memory is time-consuming for the predictive thread, as it requires constructing a new nearest-neighbor tree at each future timestamp to enable fast access to the geometric memory. Although backward flow is more efficient that it maps incoming points to the pre-built nearest-neighbor tree of the latest memory, directly forecasting backward flow is challenging due to the unknown number and positions of future points, which leads to degraded performance (see Tab. 9).

To balance the efficiency and accuracy, we propose the Inverse Forward Flow Iteration. The goal of our method is to find the corresponding point x in history memory with the current query point y. The correspondence satisfies:

$$g(x) = x = y - flow(x) \tag{6}$$

where flow(x) indicates the forecasted forward flow at point x, and -flow(x) represents the inverse forward flow.

Then we want to find a fixed point x^* such that $x^* = g(x^*)$. Given an initial guess $x_0 = y$, define the iteration as:

$$x_{n+1} = g(x_n) = y - flow(x_n) \tag{7}$$

The sequence $\{x_n\}$ will converge to the fixed point x^* if g(x) is a contraction mapping, i.e., if there exists a constant $L \leq 1$, such that for all x_1 and x_2 satisfy:

$$|g(x_1) - g(x_2)| \leq L|x_1 - x_2| \tag{8}$$

The stopping iteration condition is

$$|x_{n+1} - x_n| \leq \epsilon \tag{9}$$

where ϵ is the predefined tolerance, indicating x_n has converged to the solution. To hold this condition, we need $g(x)$ to be Lipschitz continuous, and its Lipschitz constant $L \leq 1$. Thus, we assume $|flow'(x)| \leq 1$ for each differentiable point x. The detailed proof is provided in Supp. B.

The query point iteratively finds the local forecasted forward flow in memory, then backtracks through the inverse

of this forward flow. The process continues until the distance between current query position p' and the point p closely approximates the inverse of the forward flow. The pseudo-code for this process is as follows:

Algorithm 1 Iterative Inverse Forward Flow Method

Require: forecast forward flow query Q, stop threshold ϵ, maximum iterations N_{max}
1: **for** each point p in the non-key frame **do**
2:     Initialize current query position $p' \leftarrow p$
3:     Initialize iteration counter $n \leftarrow 0$
4:     Inverse(f) $\leftarrow -f$
5:     **while** $\|(p' - p) + Q(p')\| \geq \epsilon$ **and** $n < N_{max}$ **do**
6:         Query local forecast forward flow $f \leftarrow Q(p')$
7:         Update track position: $p' \leftarrow p + \text{Inverse}(f)$
8:         Increment iteration counter: $n \leftarrow n + 1$
9:     **end while**
10: **end for**

---

5. Experiments

We present the experimental setup and benchmark results on two widely used outdoor LiDAR-based panoptic segmentation datasets, SemanticKITTI[4] and nuScenes[5], as well as the indoor dataset HOI4D[30].

5.1. Settings

SemanticKITTI [4]. SemanticKITTI is a large-scale dataset for LiDAR-based panoptic segmentation, containing 23,201 outdoor scene frames at 10 fps. Unlike traditional 4D panoptic segmentation, streaming 4D panoptic segmentation also involves distinguishing between moving and static objects, since the ability to perceive moving objects is significant in streaming perception. This adds 6 additional classes for moving objects (e.g., "moving car") to the standard 19 semantic classes. In total, there are 25 classes, including 14 thing classes and 11 stuff classes.

nuScenes [5]. nuScenes is a publicly available autonomous driving dataset with 1,000 scenes captured at 2 fps. We extend the per-point semantic labels to distinguish between moving and non-moving objects using ground truth 3D bounding box attributes. This extension includes 8 moving object classes and 16 static object classes, totaling 18 thing classes and 6 stuff classes.

HOI4D [30]. HOI4D is a large-scale egocentric dataset focused on indoor human-object interactions. It contains 3,865 point cloud sequences, with 2,971 for training and 892 for testing. Each sequence has 300 frames captured at 15 fps.

Evaluation metrics. We use PQ and LSTQ in streaming setting (denoted as sPQ and sLSTQ) as our main metrics to evaluate panoptic segmentation performance. Furthermore,

we divide the sPQ into four components: sPQ_d for dynamic objects, sPQ_s for static objects, sPQ_{th} for thing classes, and sPQ_{st} for stuff classes. In the streaming setting, evaluation of each frame must occur at every input timestamp, according to the dataset's frame rate. If the computation for the current frame is not completed in time, we use the features from the last completed frame to query the results and perform the evaluation.

Implementation details. We choose P3Former [42] and Mask4Former [45] as our backbone model, which is originally a SOTA method for 3D and 4D panoptic segmentation. By incorporating the ego pose and flow alignment strategies we proposed, along with memory construction, they can also achieve good performance in 4D streaming panoptic segmentation. We first train the model on each dataset, then freeze it for feature extraction. The remaining components, including ego-pose forecasting, forward flow forecasting, and history memory aggregation, are trained subsequently. For the inverse flow iteration, the maximum iterations patience is set to 10. All models are trained on 4 NVIDIA GTX 3090 GPUs and evaluated on a single NVIDIA GTX 3090 GPU.

5.2. Streaming 4D Panoptic Segmentation in Outdoor datasets

SemanticKITTI [4]. Tab. 1 and 2 compare streaming 4D panoptic segmentation on the SemanticKITTI validation split in the unknown and known pose settings. We compare our method with StreamYOLO [44], LongShortNet [23], DAMO-StreamNet [17], Mask4Former [45], Eq-4D-StOP [48] and PTv3 [41]. Originally designed for 2D streaming object detection via temporal feature fusion, the first three models are adapted to 4D streaming by replacing their backbones with P3Former [42]. Mask4Former and Eq-4D-StOP are designed for 4D panoptic segmentation but are not optimized for streaming. PTv3 is a state-of-the-art method designed for 3D perception. We adapt it to 4D panoptic segmentation with flow propagation according to [2].

From both tables, we observe that 2D streaming methods perform poorly due to their reliance on real-time backbones, which are difficult to achieve in such a high-granularity task. Similarly, 4D panoptic segmentation methods also suffer significant performance degradation due to computational latency. PTv3 performs better than 4D methods due to its high efficiency, but it still suffers from performance drop. In contrast, our method outperforms all baseline models by a large margin in the streaming setting. Notably, in the unknown pose setting, our method achieves significant improvements of 7.7% and 15.2% in sLSTQ over PTv3 [41]when integrated with P3Former and Mask4Former respectively, demonstrating the effectiveness of our alignment strategies across both dynamic and static classes. When combined with Mask4Former, our method outperforms its

Table 1. SemanticKITTI validation set result in *unknown pose* streaming setting. The best is highlighted in **bold**. sX indicates the metric X in the streaming setting. PQ_d and PQ_s refer to the evaluation for dynamic and static points, respectively. PQ_{th} evaluates the thing class and PQ_{st} evaluates the stuff class.

Method	sLSTQ	S_{assoc}	S_{cls}	sPQ	sRQ	sSQ	sPQ_d	sPQ_s	sPQ_{th}	sPQ_{st}
StreamYOLO [44]	0.415	0.321	0.536	0.373	0.478	0.664	0.429	0.371	0.388	0.364
LongShortNet [23]	0.430	0.341	0.541	0.392	0.472	0.673	0.452	0.391	0.400	0.386
DAMO-StreamNet [17]	0.432	0.341	0.546	0.392	0.472	0.674	0.459	0.391	0.400	0.388
Mask4Former [45]	0.515	0.464	0.572	0.485	0.594	0.691	0.571	0.413	0.538	0.422
Eq-4D-StOP [48]	0.504	0.452	0.563	0.477	0.578	0.691	0.543	0.412	0.529	0.423
PTv3 [41]	0.536	0.492	0.586	0.567	0.612	0.704	0.638	0.464	0.575	0.459
4DSegStreamer (P3Former)	0.613	0.627	0.599	0.602	0.679	0.723	0.711	0.479	0.625	0.481
4DSegStreamer (Mask4Former)	**0.688**	**0.706**	**0.621**	**0.634**	**0.701**	**0.752**	**0.744**	**0.486**	**0.660**	**0.497**

Table 2. SemanticKITTI validation set result in *known pose* streaming setting. The best is highlighted in **bold**. sX indicates the metric X in the streaming setting. PQ_d and PQ_s refer to the evaluation for dynamic and static points, respectively. PQ_{th} evaluates the thing class and PQ_{st} evaluates the stuff class.

Method	sLSTQ	S_{assoc}	S_{cls}	sPQ	sRQ	sSQ	sPQ_d	sPQ_s	sPQ_{th}	sPQ_{st}
StreamYOLO [44]	0.439	0.356	0.541	0.384	0.468	0.715	0.432	0.383	0.392	0.382
LongShortNet [23]	0.446	0.360	0.553	0.412	0.489	0.719	0.459	0.410	0.413	0.399
DAMO-StreamNet [17]	0.446	0.362	0.551	0.425	0.489	0.724	0.460	0.412	0.414	0.401
Mask4Former [45]	0.564	0.539	0.592	0.520	0.613	0.734	0.623	0.460	0.592	0.467
Eq-4D-StOP [48]	0.557	0.530	0.585	0.520	0.619	0.732	0.625	0.459	0.594	0.465
4DSegStreamer (P3Former)	0.655	0.703	0.610	0.687	0.774	0.816	0.782	0.560	0.704	0.531
4DSegStreamer (Mask4Former)	**0.701**	**0.722**	**0.648**	**0.704**	**0.811**	**0.838**	**0.803**	**0.579**	**0.741**	**0.552**

Table 3. nuScenes validation set result in *unknown pose* streaming setting. The best is highlighted in **bold**.

Method	sLSTQ	sPQ	sPQ_d	sPQ_s
StreamYOLO [44]	0.596	0.581	0.569	0.591
LongShortNet [23]	0.610	0.603	0.579	0.607
DAMO-StreamNet [17]	0.623	0.607	0.601	0.612
Mask4Former [45]	0.648	0.636	0.634	0.641
Eq-4D-StOP [48]	0.650	0.642	0.633	0.658
PTv3 [41]	0.662	0.659	0.627	0.670
4DSegStreamer (P3)	0.693	0.683	0.675	0.690
4DSegStreamer (M4F)	**0.721**	**0.733**	**0.701**	**0.699**

Table 4. nuScenes validation set result in *known pose* streaming setting. The best is highlighted in **bold**.

Method	sLSTQ	sPQ	sPQ_d	sPQ_s
StreamYOLO [44]	0.613	0.593	0.583	0.613
LongShortNet [23]	0.628	0.6116	0.599	0.621
DAMO-StreamNet [17]	0.633	0.625	0.607	0.639
Mask4Former [45]	0.681	0.665	0.655	0.683
Eq-4D-StOP [48]	0.695	0.673	0.654	0.693
4DSegStreamer (P3)	0.747	0.723	0.711	0.733
4DSegStreamer (M4F)	**0.765**	**0.751**	**0.734**	**0.786**

combination with P3Former, as Mask4Former is specifically designed for 4D panoptic segmentation.

nuScenes [5]. We also compare the performance of 4D streaming panoptic segmentation on the nuScenes validation split [5]. Compared to SemanticKITTI[4], it has a slower frame rate, which allows many baseline methods to achieve real-time computation. However, in a streaming setting, even real-time methods experience at least a one-frame delay, leading to performance degradation. As shown in Tab. 3 and 4, our method outperforms all baseline

approaches in both known and unknown pose settings. Additionally, all models perform better in the known pose setting, as pose estimation in the unknown pose setting takes more time, further degrading performance.

5.3. Streaming 4D Panoptic Segmentation in Indoor dataset

HOI4D [30]. We also evaluate our model in indoor scenarios. We compare our approach with StreamYOLO [44], LongShortNet [23], DAMO-StreamNet [17], NSM4D [10] and PTv3 [41]. As shown in Tab. 5, our method outper-

Table 5. HOI4D test set result in *unknown pose* streaming setting. The best is highlighted in **bold**.

Method	sLSTQ	sPQ	sPQ$_d$	sPQ$_s$
StreamYOLO [44]	0.373	0.336	0.362	0.324
LongShortNet [23]	0.377	0.335	0.354	0.323
DAMO-StreamNet [17]	0.375	0.335	0.351	0.324
NSM4D [10]	0.314	0.305	0.315	0.303
PTv3 [41]	0.445	0.417	0.397	0.445
4DSegStreamer (P3)	0.483	0.455	0.431	0.490
4DSegStreamer (M4F)	**0.511**	**0.482**	**0.457**	**0.533**

Table 6. General evaluation of different backbones. $w/o\ streamer$ is vanilla backbone. $w\ streamer$ is 3D or 4D backbone with our 4DSegStreamer.

Method	sLSTQ$_{w/o\ streamer}$	sLSTQ$_{w\ streamer}$
Mask4Former [45]	0.515	**0.688**
Eq-4D-StOP [48]	0.504	**0.674**
P3former [42]	0.304	**0.613**

Table 7. Ablation study in unknown pose streaming setting. $P3$ indicates the P3former backbone. Mem represents the memory module. $Pose$ and $Flow$ denote multi-frames future pose and flow forecasting, respectively. $M\ Flow$ indicates the moving mask to assign non-zero flow only to moving objects.

Method	sLSTQ	sLSTQ$_d$	sLSTQ$_s$
P3 [42]	0.304	0.265	0.357
P3+Mem	0.349	0.292	0.408
P3+Mem+Pose	0.497	0.488	0.501
P3+Mem+Pose+Flow	0.591	0.667	0.514
P3+Mem+Pose+M Flow	**0.613**	**0.682**	**0.516**

Table 8. Ablation study in known pose streaming setting. Pose is given and Flow is multi-head forecasting. Mem represents the memory module. $Flow$ denotes multi-frame future flow forecasting.

Method	sLSTQ	sLSTQ$_d$	sLSTQ$_s$
P3+Mem+GTpose	0.563	0.534	0.592
P3+Mem+GTpose+Flow	**0.655**	**0.698**	**0.601**

Table 9. Ablation study of different flow forecasting methods.

Method	sLSTQ	sLSTQ$_d$	sLSTQ$_s$
Backward flow	0.565	0.637	0.483
Forward flow	0.589	0.667	0.497
Inverse forward flow	0.586	0.662	0.502
Inverse brute search	0.591	0.669	0.501
Inverse flow iteration	**0.613**	**0.682**	**0.516**

forms all other approaches, surpassing the runner-up by 6.6% in terms of sLSTQ. This demonstrates that our method exhibits strong generalization ability, performing well not only in outdoor scenarios but also in indoor scenes.

5.4. Ablations for System

In this section, we conduct several groups of ablation studies on SemanticKITTI [4] validation set to demonstrate the effectiveness of 4DSegStreamer.

General to 3D and 4D backbone. Tab 6 demonstrates that integrating our plug-and-play 4DSegStreamer consistently boosts the perfomance across various SOTA 3D and 4D backbones, with significnt improvements observed. This highlights the generality and effectiveness of our framework in enabling real-time capability.

Effects of Components. Pose alignment mitigates the ego-pose motion, resulting in improvements to both sLSTQ$_d$ and sLSTQ$_s$. Building on this, incorporating flow alignment further refines the handling of moving objects, significantly boosting the model's performance on sLSTQ$_d$. We evaluate our method under both unknown-pose (Tab. 7) and known-pose settings (Tab. 8), where the latter provides ground-truth ego poses. Results demonstrate that our memory module, pose alignment, and dynamic object alignment continuously enhance streaming performance. Moreover, applying a non-moving object mask brings additional gains.

Flow Forecasting Strategies. We compare different flow forecasting strategies in Tab. 9. The "Inverse Forward Flow" represents a single iteration of the Inverse Flow Iteration algorithm, while the "Inverse Brute Search" algorithm directly searches for the forward flow within a restricted region that points to the target position. As shown in the table, forward flow forecasting does not achieve the best performance due to the high time consumption associated with repeated kd-tree construction. Additionally, backward flow forecasting performs poorly, as it is challenging to predict the backward flow without knowledge of the future position. In contrast, our proposed Inverse Flow Iteration algorithm shows superior performance in terms of sLSTQ.

6. Conclusion

In this work, we propose 4DSegStreamer, an efficient 4D streaming panoptic segmentation method that optimizes accuracy-latency trade-offs. We develop a dual-thread system to synchronize current and future point clouds within temporal constraints, complemented by an ego-pose forecaster and inverse forward flow iteration for motion alignment. Evaluated across diverse indoor and outdoor panoptic segmentation datasets, our method demonstrates robust performance in streaming scenarios.

References

[1] Mazen Abdelfattah, Kaiwen Yuan, Z Jane Wang, and Rabab Ward. Multi-modal streaming 3d object detection. *IEEE Robotics and Automation Letters*, 2023. 1, 2

[2] Mehmet Aygun, Aljosa Osep, Mark Weber, Maxim Maximov, Cyrill Stachniss, Jens Behley, and Laura Leal-Taixé. 4d panoptic lidar segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5527–5537, 2021. 2, 6

[3] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015. 4

[4] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019. 6, 7, 8

[5] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multi-modal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 6, 7

[6] Qi Chen, Sourabh Vora, and Oscar Beijbom. Polarstream: Streaming object detection and segmentation with polar pillars. *Advances in Neural Information Processing Systems*, 34:26871–26883, 2021. 1, 2

[7] Xieyuanli Chen, Andres Milioto, Emanuele Palazzolo, Philippe Giguere, Jens Behley, and Cyrill Stachniss. Suma++: Efficient lidar-based semantic slam. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4530–4537. IEEE, 2019. 5

[8] Xuechao Chen, Shuangjie Xu, Xiaoyi Zou, Tongyi Cao, Dit-Yan Yeung, and Lu Fang. Svqnet: Sparse voxel-adjacent query network for 4d spatio-temporal lidar semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8569–8578, 2023. 2

[9] Ayush Dewan and Wolfram Burgard. Deeptemporalseg: Temporally consistent semantic segmentation of 3d lidar scans. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2624–2630. IEEE, 2020. 2

[10] Yuhao Dong, Zhuoyang Zhang, Yunze Liu, and Li Yi. Nsm4d: Neural scene model based online 4d point cloud sequence understanding. *arXiv preprint arXiv:2310.08326*, 2023. 2, 7, 8

[11] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point 4d transformer networks for spatio-temporal modeling in point cloud videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14204–14213, 2021. 2

[12] Davi Frossard, Shun Da Suo, Sergio Casas, James Tu, and Raquel Urtasun. Strobe: Streaming object detection from lidar packets. In *Conference on Robot Learning*, pages 1174–1183. PMLR, 2021. 1, 2

[13] Weizhen Ge, Xin Wang, Zhaoyong Mao, Jing Ren, and Junge Shen. Streamtrack: real-time meta-detector for streaming perception in full-speed domain driving scenarios. *Applied Intelligence*, pages 1–17, 2024. 1

[14] Anurag Ghosh, Vaibhav Balloli, Akshay Nambi, Aditya Singh, and Tanuja Ganu. Chanakya: Learning runtime decisions for adaptive real-time perception. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[15] Alex Graves and Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012. 5

[16] Wei Han, Zhengdong Zhang, Benjamin Caine, Brandon Yang, Christoph Sprunk, Ouais Alsharif, Jiquan Ngiam, Vijay Vasudevan, Jonathon Shlens, and Zhifeng Chen. Streaming object detection for 3-d point clouds. In *European Conference on Computer Vision*, pages 423–441. Springer, 2020. 1, 2

[17] Jun-Yan He, Zhi-Qi Cheng, Chenyang Li, Wangmeng Xiang, Binghui Chen, Bin Luo, Yifeng Geng, and Xuansong Xie. Damo-streamnet: Optimizing streaming perception in autonomous driving. *arXiv preprint arXiv:2303.17144*, 2023. 1, 2, 6, 7, 8

[18] Xiang Huang, Zhi-Qi Cheng, Jun-Yan He, Chenyang Li, Wangmeng Xiang, Baigui Sun, and Xiao Wu. Dyronet: Dynamic routing and low-rank adapters for autonomous driving streaming perception. *CoRR*, 2024.

[19] Yihui Huang and Ningjiang Chen. Mtd: Multi-timestep detector for delayed streaming perception. In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pages 337–349. Springer, 2023. 2, 1

[20] Wonwoo Jo, Kyungshin Lee, Jaewon Baik, Sangsun Lee, Dongho Choi, and Hyunkyoo Park. Dade: delay-adaptive detector for streaming perception. *arXiv preprint arXiv:2212.11558*, 2022. 1, 2

[21] Lars Kreuzberg, Idil Esen Zulfikar, Sabarinath Mahadevan, Francis Engelmann, and Bastian Leibe. 4d-stop: Panoptic segmentation of 4d lidar using spatio-temporal object proposal generation and aggregation. In *European Conference on Computer Vision*, pages 537–553. Springer, 2022. 2

[22] Bowen Li, Ziyuan Huang, Junjie Ye, Yiming Li, Sebastian Scherer, Hang Zhao, and Changhong Fu. Pvt++: a simple end-to-end latency-aware visual tracking framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10006–10016, 2023. 1

[23] Chenyang Li, Zhi-Qi Cheng, Jun-Yan He, Pengyu Li, Bin Luo, Hanyuan Chen, Yifeng Geng, Jin-Peng Lan, and Xuansong Xie. Longshortnet: Exploring temporal and semantic features fusion in streaming perception. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. 1, 2, 6, 7, 8

[24] Dianze Li, Jianing Li, and Yonghong Tian. Sodformer: Streaming object detection with transformer using events and frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 1, 2

[25] Enxu Li, Sergio Casas, and Raquel Urtasun. Memoryseg: Online lidar semantic segmentation with a latent memory. In

Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 745–754, 2023. 2, 3, 4

[26] Mengtian Li, Yu-Xiong Wang, and Deva Ramanan. Towards streaming perception. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 473–488. Springer, 2020. 1, 2

[27] Xueqian Li, Jianqiao Zheng, Francesco Ferroni, Jhony Kaesemodel Pontes, and Simon Lucey. Fast neural scene flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9878–9890, 2023. 5

[28] Zhiheng Li, Yubo Cui, Jiexi Zhong, and Zheng Fang. Streammos: Streaming moving object segmentation with multi-view perception and dual-span memory. *arXiv preprint arXiv:2407.17905*, 2024. 2

[29] Jiahui Liu, Chirui Chang, Jianhui Liu, Xiaoyang Wu, Lan Ma, and Xiaojuan Qi. Mars3d: A plug-and-play motion-aware model for semantic segmentation on multi-scan 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9372–9381, 2023. 2

[30] Yunze Liu, Yun Liu, Che Jiang, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, Zhoujie Fu, He Wang, and Li Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21013–21022, 2022. 6, 7

[31] Rodrigo Marcuzzi, Lucas Nunes, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Mask-based panoptic lidar segmentation for autonomous driving. *IEEE Robotics and Automation Letters*, 8(2):1141–1148, 2023. 2

[32] Kangan Qian, Zhikun Ma, Yangfan He, Ziang Luo, Tianyu Shi, Tianze Zhu, Jiayin Li, Jianhui Wang, Ziyu Chen, Xiao He, et al. Fasionad: Fast and slow fusion thinking systems for human-like autonomous driving with adaptive feedback. *arXiv preprint arXiv:2411.18013*, 2024. 3

[33] Gur-Eyal Sela, Ionel Gog, Justin Wong, Kumar Krishna Agrawal, Xiangxi Mo, Sukrit Kalra, Peter Schafhalter, Eric Leong, Xin Wang, Bharathan Balaji, et al. Context-aware streaming perception in dynamic environments. In *European Conference on Computer Vision*, pages 621–638. Springer, 2022. 1

[34] Hanyu Shi, Guosheng Lin, Hao Wang, Tzu-Yi Hung, and Zhenhua Wang. Spsequencenet: Semantic segmentation network on 4d point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4574–4583, 2020. 2

[35] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024. 3

[36] Kyle Vedder, Neehar Peri, Nathaniel Chodosh, Ishan Khatri, Eric Eaton, Dinesh Jayaraman, Yang Liu, Deva Ramanan, and James Hays. Zeroflow: Scalable scene flow via distillation. *arXiv preprint arXiv:2305.10424*, 2023. 5

[37] Sourabh Vora and Qi Chen. Streaming object detection and segmentation with polar pillars, 2023. US Patent 11,798,289. 1, 2

[38] Xiaofeng Wang, Zheng Zhu, Yunpeng Zhang, Guan Huang, Yun Ye, Wenbo Xu, Ziwei Chen, and Xingang Wang. Are we ready for vision-centric driving streaming perception? the asap benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9600–9610, 2023. 2, 1

[39] Hao Wen, Yunze Liu, Jingwei Huang, Bo Duan, and Li Yi. Point primitive transformer for long-term 4d point cloud video understanding. In *European Conference on Computer Vision*, pages 19–35. Springer, 2022. 2

[40] Xiaopei Wu, Yuenan Hou, Xiaoshui Huang, Binbin Lin, Tong He, Xinge Zhu, Yuexin Ma, Boxi Wu, Haifeng Liu, Deng Cai, et al. Taseg: Temporal aggregation network for lidar semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15311–15320, 2024. 2

[41] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024. 6, 7, 8

[42] Zeqi Xiao, Wenwei Zhang, Tai Wang, Chen Change Loy, Dahua Lin, and Jiangmiao Pang. Position-guided point cloud panoptic segmentation transformer. *International Journal of Computer Vision*, pages 1–16, 2024. 6, 8

[43] Xiuwei Xu, Chong Xia, Ziwei Wang, Linqing Zhao, Yueqi Duan, Jie Zhou, and Jiwen Lu. Memory-based adapters for online 3d scene perception. *arXiv preprint arXiv:2403.06974*, 2024. 2

[44] Jinrong Yang, Songtao Liu, Zeming Li, Xiaoping Li, and Jian Sun. Real-time object detection for streaming perception. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5385–5395, 2022. 1, 2, 6, 7, 8

[45] Kadir Yilmaz, Jonas Schult, Alexey Nekrasov, and Bastian Leibe. Mask4former: Mask transformer for 4d panoptic segmentation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9418–9425. IEEE, 2024. 2, 3, 6, 7, 8

[46] Xiang Zhang, Yufei Cui, Chenchen Fu, Weiwei Wu, Zihao Wang, Yuyang Sun, and Xue Liu. Transtreaming: Adaptive delay-aware transformer for real-time streaming perception. *arXiv preprint arXiv:2409.06584*, 2024. 1, 2

[47] Yunsong Zhou, Hongzi Zhu, Chunqin Li, Tiankai Cui, Shan Chang, and Minyi Guo. Tempnet: Online semantic segmentation on large-scale point cloud series. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7118–7127, 2021. 2

[48] Minghan Zhu, Shizhong Han, Hong Cai, Shubhankar Borse, Maani Ghaffari, and Fatih Porikli. 4d panoptic segmentation as invariant and equivariant field prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22488–22498, 2023. 2, 6, 7, 8

4DSegStreamer: Streaming 4D Panoptic Segmentation via Dual Threads
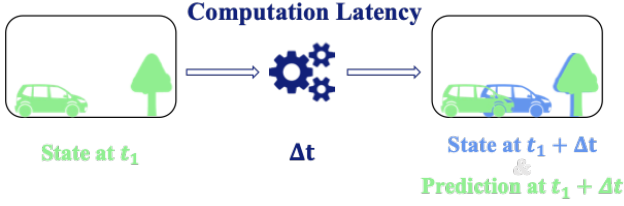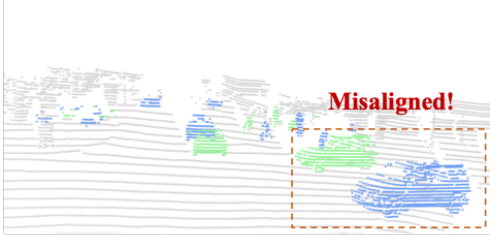
Supplementary Material

Figure S1. Streaming Perception Setting: Green points denote dynamic objects from the processed frame, whereas blue points represent the current frame at the time of prediction generated by the algorithm.

A. Streaming Perception Setting

Based on previous works [17, 19, 20, 23, 26, 38, 44, 46] in streaming perception, our 4D streaming panoptic segmentation addresses a similar challenge by explicitly considering the impact of algorithmic processing latency on the final prediction and the scene at output time. As illustrated in Fig. S1, predictions from existing methods are misaligned with the actual scene due to this latency. This misalignment can lead to perception inaccuracies, posing potential risks when robotic systems operate in highly dynamic environments.

B. Forward Flow Iteration Proof

To find the flow between the current query point and history position in geometric memory, we use the forward flow iteration. The iteration converges if Eq. 8 holds, then the following equation holds

$$1 \geq L \geq \frac{|g(x_0 + \Delta x) - g(x_0 - \Delta x)|}{|(x_0 + \Delta x) - (x_0 - \Delta x)|}$$
$$= \left| \frac{f(x_0 + \Delta x) - f(x_0 - \Delta x)}{(x_0 + \Delta x) - (x_0 - \Delta x)} \right| = |f'(x_0)|$$

For point x on a rigid object and the flow $f(x, t)$ representing velocity, the derivative $|f'(x)|$ can be expressed as:

Table S1. Performance of different GPUs with different latency.

	4DSegStreamer(M4F)	PTv3	Mask4Former
sLSTQ$_{A40}$	0.681	0.526	0.501
sLSTQ$_{3090}$	0.688	0.536	0.504
sLSTQ$_{A100}$	0.702	0.561	0.538

$$|f'(x)| = \left| \frac{\partial f(x,t)}{\partial x} \right| = \left| \frac{\partial (v + \omega \times (x - x_c))}{\partial x} \right|$$
$$= \left| \frac{\partial (\omega \times x)}{\partial x} \right| = \left| [\omega]_\times \right| = |\omega|$$

where x_c is the rotation center of the rigid body, v is the translational velocity, ω is angular velocity, $[\omega]_\times$ is the cross-product matrix. The iteration converges when $|\omega| \leq 1$. In real-world scenarios, most rigid objects exhibit low angular velocity, allowing the iteration converges reliably.

While perfect convergence cannot be guaranteed in practice, our experiments show robust convergence in 97.4% of scenes in the SemanticKITTI dataset.

C. Performance of different GPUs

Table S1 presents the performance of our method across different GPUs under streaming settings. Since the model's runtime speed and GPU processing capability significantly impact the metric performance, the choice of hardware plays a crucial role. Notably, the A40 and 3090 graphics cards exhibit comparable performance due to their similar computational efficiency. In contrast, the A100 demonstrates a substantial speed advantage over the 3090, leading to a 1.4% improvement in our model's performance on the A100.