SPARROW: Learning Spatial Precision and Temporal Referential Consistency in Pixel-Grounded Video MLLMs

Supplementary Material

A. TSF Datasets

To supervise TSF and dual-prompt grounding at scale, we curate a unified video benchmark tailored for referential understanding and fine-grained segmentation. We aggregate seven publicly available datasets, HC-STVG [48], VID-Sentence [7], A2D Sentences [15], LaSOT [14], MeViS [12], GOT-10k [17], and Ref-SAV [43, 59], and process them with a common offline pipeline, yielding a corpus of 30,646 video sequences and 45,231 question–answer pairs. In all cases, we standardize annotations into (video, referring text, trajectory, mask) tuples and precompute TSF tokens for SPARROW.

HC-STVG. [48] contains movie clips with tracking sequences and natural language descriptions of a person over specific temporal intervals. We use the official training split ($\sim$10K clips) for TSF pretraining and the validation split for evaluation. The original tracking annotations are often noisy, so we regenerate trajectories using GroundingDINO [31] plus CLDTracker [2]. Specifically, we: (i) detect the referent in annotated key frames with GroundingDINO, (ii) track the detected region across the whole clip with CLDTracker to obtain dense trajectories, and (iii) compare regenerated boxes with the HC-STVG ground truth, discarding frames whose IoU falls below a threshold. The resulting cleaned tracks are then fed to SAM2 [43] to obtain dense masks and to our TSF pipeline (Sec. 3.2) to generate compact tracked tokens.

A2D Sentences. [15] provides short tracking sequences (often only a few frames) and textual descriptions for multiple actors in each clip. To expose TSF to longer temporal context, we start from the annotated boxes and re-track the referent using CLDTracker, extending trajectories to up to 20 frames where possible. As in HC-STVG, we filter low-IoU frames, convert masks from SAM2 into bounding boxes for box-level supervision, and keep the dense masks for segmentation and TSF supervision.

LaSOT. [14] offers long-term tracking sequences with a single caption describing the target object across the entire video. Since captions are generic and sequences are long, we sub-sample each video into three non-overlapping 10-second segments (fixed-length frame windows) anchored around the target trajectory. Each segment inherits the original caption but is paired with the segment-specific trajectory and masks (obtained via GroundingDINO + CLDTracker + SAM2). This turns each long sequence into several shorter, more focused referential clips.

MeViS. [12] provides videos with dense referring segmentation masks and language expressions. We directly use the official splits, convert the per-frame masks into bounding boxes for proposal-level supervision, and keep the original masks for TSF and dual-prompt training. Since trajectories are already stable, we only run CLDTracker when annotations are sparse or missing in intermediate frames, ensuring temporally continuous supervision.

VID-Sentence. [7] contains videos with sentence-level descriptions and sparse annotations. We keep the original temporal spans and regenerate trajectories from a single annotated frame per instance using GroundingDINO and CLDTracker. SAM2 is then applied to produce per-frame masks consistent with the tracked boxes. No additional temporal cropping is used beyond trimming to the annotated span.

GOT-10k. [17] is a tracking benchmark with category, motion, and attribute labels for each tracked object. Following prior work, we convert these labels into natural language descriptions by concatenating them into short sentences (e.g., *"a bear is slowly walking"*). The provided trajectories are treated as initial boxes; we run SAM2 to obtain dense masks and apply our TSF pipeline to select a compact subset of representative regions via K-means clustering in the joint visual–spatial feature space.

Ref-SAV. [43, 59] offers long videos with referring expressions and high-quality SAM2-based masks. We treat Ref-SAV as a high-quality source of dense referential supervision: masks are converted into bounding boxes, trajectories are inferred from mask centroids, and TSF tokens are generated from cropped regions without additional tracking.

Offline TSF Pipeline. For all datasets, we run a unified offline pipeline that decouples heavy computation from training. Given a video and its referring expression, we: (i) detect the referent in one or more key frames with GroundingDINO, (ii) track detections through the clip with CLDTracker to obtain dense trajectories, (iii) crop tracked regions and encode them with the spatial encoder, (iv) apply K-means clustering to select K diverse appearances per trajectory, and (v) project the selected features into the LLM space as TSF tokens Z_{TSF} (Sec. 3.2). Optionally, we invoke SAM2 on the tracked boxes to generate per-frame masks,

which are stored alongside trajectories for segmentation supervision.

Question–Answer Construction. To convert the curated corpus into training samples suitable for SPARROW, we transform each (video, trajectory, mask, description) tuple into one or more question–answer pairs. We use a small pool of hand-crafted referential templates, such as ``What is the <region> doing during this video?''. For all datasets we use the caption or description directly as the answer, without modification. During training, we randomly sample a referential template and fill in the region placeholder, producing diverse yet standardized QA prompts. The final TSF dataset contains 30,646 video clips and 45,231 question–answer pairs.

Table 6. Curated datasets used to construct our TSF dataset.

Dataset	Video Clips	Q&A Pairs
HC-STVG [48]	10105	10105
MeViS [12]	1644	4489
A2D Sentences [15]	2508	5359
LaSOT [14]	2640	7920
VID-Sentence [7]	4045	7654
GOT-10k [17]	8195	8195
Ref-SAV [43, 59]	1509	1509
TSF Dataset	**30646**	**45231**

B. Evaluation Setup

VideoGLaMM. VideoGLaMM [38] is a video-centric MLLM designed for dense pixel-level grounding. It employs a dual spatio-temporal vision encoder consisting of (i) a CLIP ViT-L/14 image backbone for high-resolution spatial features and (ii) an InternVideo2-based temporal encoder for long-range motion modeling. Both streams are projected into the LLM token space via learnable V→L adapters and fused with text tokens in a Phi-3 Mini LLM (LoRA-tuned). Grounding is triggered using a special <SEG> token. For mask prediction, VideoGLaMM uses a SAM2-based spatio-temporal decoder that consumes L→V-transformed LLM embeddings along with multi-scale features from a frozen ViT encoder. Training is end-to-end, combining LLM cross-entropy on grounded captions with IoU-based mask losses.

GLUS. GLUS [28] is an MLLM-based RVOS framework that integrates global and local temporal reasoning. Built on LISA-7B with a SAM2 mask decoder, GLUS divides the video into context frames (uniformly sampled to capture global semantics) and query frames (short temporal windows used for mask prediction). The LLM autoregressively outputs frame-wise [SEG] tokens conditioned on the text query, context frames, and prior query frames. These tokens are decoded into masks using a SAM2 decoder enhanced with an end-to-end learnable memory bank, enabling long-term temporal modeling without an external VOS propagator. GLUS further improves language–object alignment via an object-level contrastive loss over [SEG] tokens and employs a self-refined key-frame selector trained from its own pseudo-IoU confidence.

UniPixel. UniPixel [32] is a unified pixel-grounded MLLM that couples Qwen2.5-VL with a ViT-based visual encoder and a SAM2.1 segmentation head. It encodes visual prompts (points, boxes, masks) into single tokens via Fourier positional encodings combined with temporal embeddings, allowing the LLM to reason over structured visual cues. A central component is the object memory bank, a hashmap storing spatio-temporal object masks. Upon encountering <REF> tokens, the model performs memory pre-fill, segmenting all referenced objects and writing them to memory. During subsequent turns, <MEM> tokens inject object-level features back into the LLM, enabling robust multi-turn and multi-object reasoning. Segmentation is produced via a SAM2 decoder conditioned on downsampled <SEG> token embeddings. UniPixel is trained through a three-stage alignment pipeline over 851k regional captioning, 87k referring segmentation, and 1M mixed pixel-level reasoning samples, using a combination of LM loss, focal/dice loss, MAE, and objectness supervision.

C. Qualitative Analysis

C.1. Referring Video Object Segmentation (RVOS)

MeViS. Fig. 7 shows a case where the motion-centric query "rabbit moving from middle to left" requires distinguishing between two nearly identical rabbits in the same scene. VideoGLaMM, GLUS, and UniPixel often drift onto the stationary rabbit or partially fuse both instances, indicating difficulty maintaining identity under appearance similarity and occlusion. Our SPARROW isolates the correct rabbit throughout the sequence, preserving mask continuity even when the target moves through clutter. This demonstrates the benefit of our dual-prompt grounding and motion-aware temporal cues for queries that hinge on fine motion differences.

Fig. 8 presents a multi-object scenario with the expression "the three horses are moving in the water." The methods compared struggle to maintain three distinct trajectories: VideoGLaMM and GLUS frequently merge adjacent horses, while UniPixel intermittently loses instances during overlap or viewpoint changes. Our SPARROW keeps all three horses separated and stable across the entire clip, even during heavy interaction. This indicates that our grounding mechanism handles coordinated multi-object motion more reliably than prior pixel-grounding models.

Fig. 9 highlights a more dynamic interaction: "the two zebras playfully chasing each other." The fast motion, re-

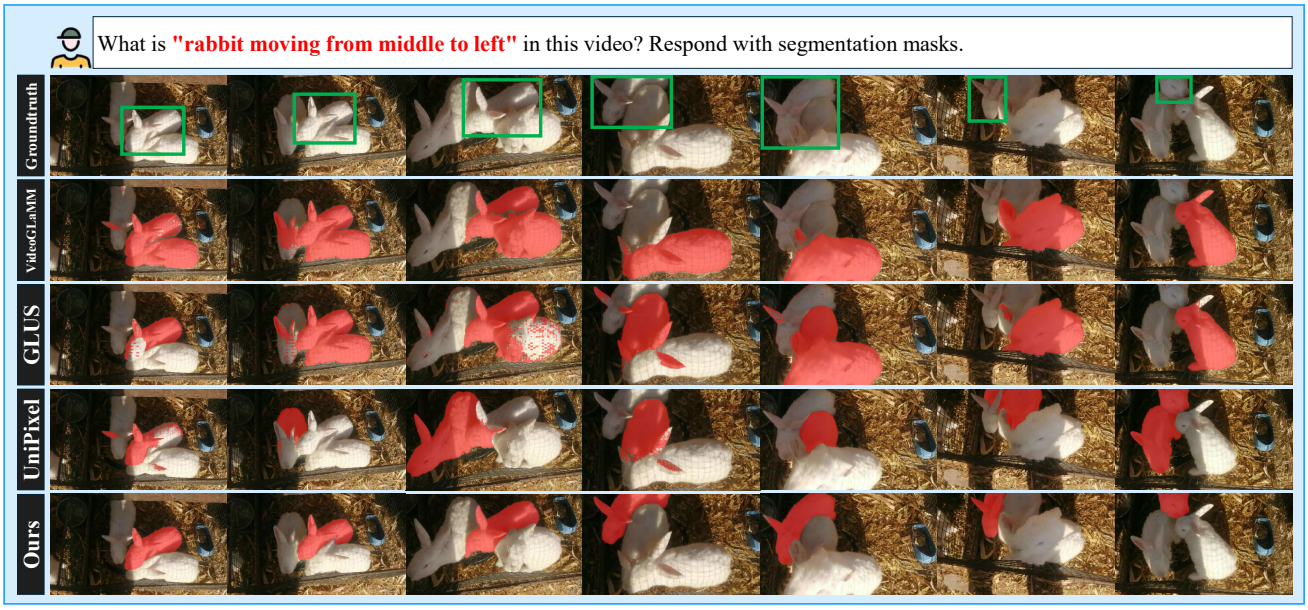

Figure 7. **Qualitative comparison on the MeViS RVOS task.** Given the motion-centric query "rabbit moving from middle to left," our method correctly grounds the intended rabbit from the first frame and maintains its identity throughout the entire sequence, despite the presence of multiple appearance-similar distractors. It produces temporally stable masks with clean boundaries and consistent spatial localization even under close interactions between the rabbits. In contrast, VideoGLaMM, UniPixel, and GLUS frequently drift between rabbits, mix identities, or generate unstable masks across frames. The strong referential grounding of our approach enables precise, distraction-robust tracking and segmentation of the queried rabbit across all frames.

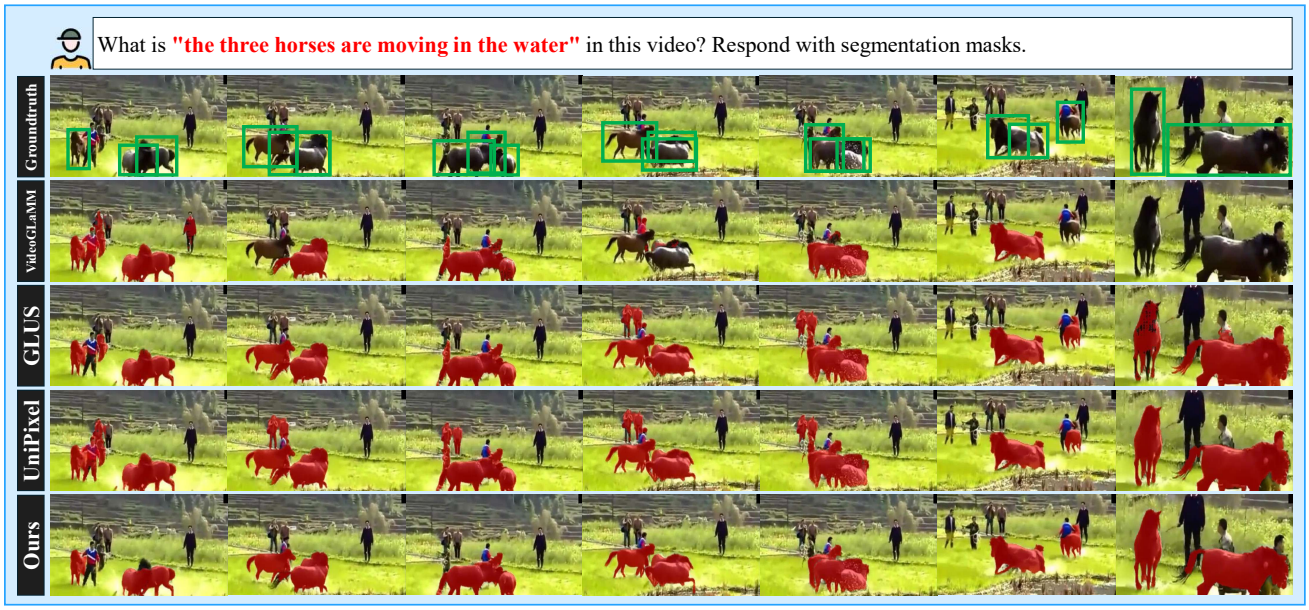

Figure 8. **Qualitative comparison on the MeViS RVOS task.** For the query "the three horses are moving in the water," our method accurately grounds all three horses from the first frame and preserves their individual identities throughout the entire sequence. It maintains clean instance separation, stable temporal masks, and sharp boundaries even as the horses move closely, interact, and occlude one another. In contrast, VideoGLaMM, UniPixel, and GLUS frequently merge instances, lose one or more horses across frames, or produce inconsistent and drifting masks. The strong multi-instance grounding of our approach enables reliable segmentation and tracking of all three horses as distinct, temporally consistent targets.

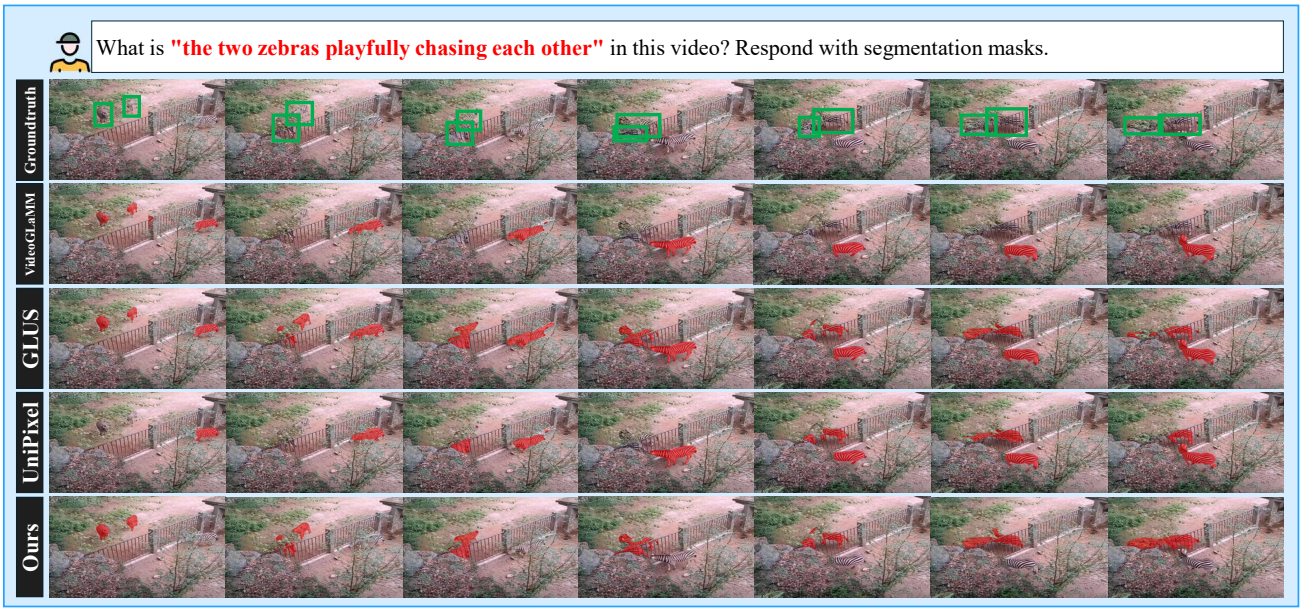

Figure 9. **Qualitative comparison on the MeViS RVOS task.** For the query "the two zebras playfully chasing each other," our method accurately grounds both zebras from the first frame and preserves their identities throughout the sequence, despite rapid motion, tight interaction, and repeated occlusions. It maintains clear instance separation, stable temporal masks, and precise spatial localization under fast, dynamic behavior. In contrast, VideoGLaMM, UniPixel, and GLUS frequently merge the two zebras, lose one during high-motion frames, or produce unstable and drifting masks. Our approach delivers robust multi-instance grounding and consistent segmentation of both zebras as distinct targets across the entire video.

Figure 10. **Qualitative comparison on the Ref-YTVOS RVOS task.** For the query "a black bird flying highest to the left," our method reliably grounds the correct bird from the first frame and maintains accurate tracking throughout the sequence, despite the densely packed flock and rapid aerial motion. It preserves clean spatial localization and stable masks even when multiple birds share nearly identical appearances and flight patterns. In contrast, VideoGLaMM, GLUS, and UniPixel frequently drift to neighboring birds, lose the target under fast movement, or produce inconsistent and flickering masks. Our approach demonstrates strong referential grounding and robust temporal consistency in challenging multi-instance, motion-heavy scenarios.

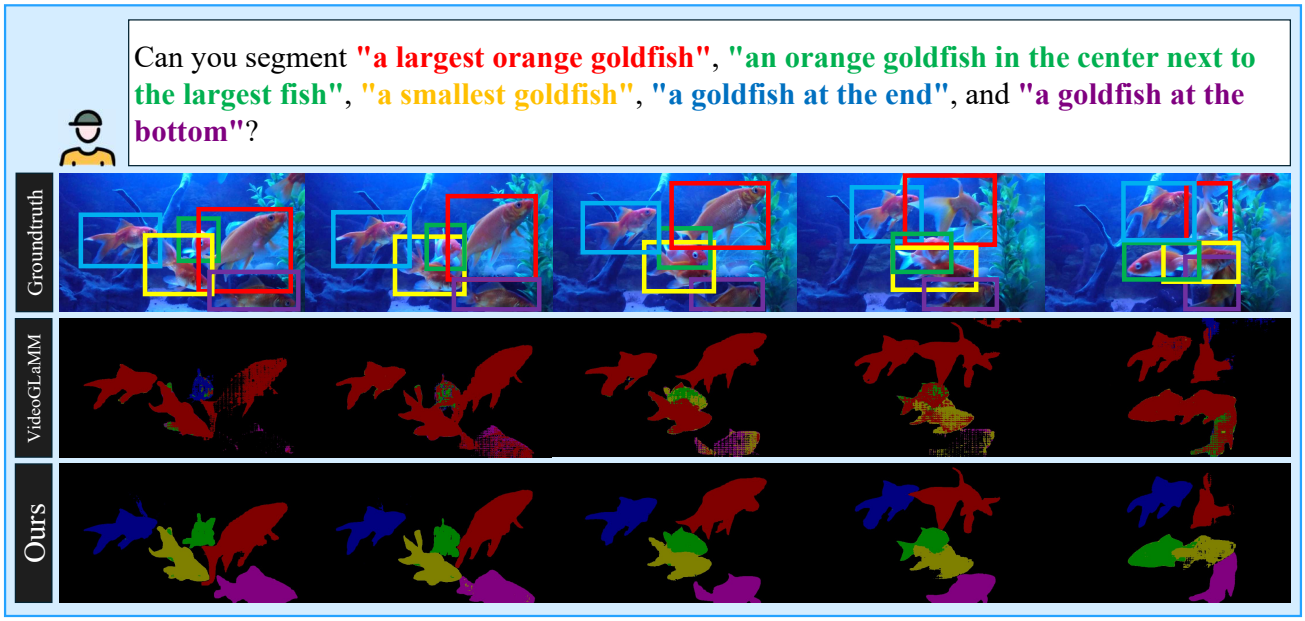

Figure 11. **Qualitative comparison on the Ref-DAVIS17 RVOS task.** Given multiple appearance-based prompts targeting distinct goldfish ("largest," "smallest," "center," "end," "bottom"), our method accurately grounds each described instance from the first frame and maintains clear separation throughout the sequence. It preserves stable masks, fine-grained localization, and consistent identities even when the fish exhibit similar colors, overlapping motion, and close interactions. In contrast, VideoGLaMM frequently merges instances, confuses similarly colored fish, or produces unstable and drifting segmentations. Our approach demonstrates strong referential grounding under simultaneous multi-query conditions, delivering reliable instance-specific masks across the entire sequence.

peated occlusions, and near-identical appearance make this a failure point for existing methods. VideoGLaMM tends to focus on a single zebra; UniPixel shows unstable boundaries and occasional identity swaps; GLUS often collapses both zebras into one mask. Our SPARROW keeps the two zebras separated throughout, even as they cross paths. This example underscores the advantage of our instance-aware temporal modeling in scenes where motion and appearance cues are both ambiguous.

Ref-YTVOS. Fig. 10 features a dense flock scenario with the query *"a black bird flying highest to the left."* The scene contains many visually similar birds with overlapping motion and strong blur, making appearance-based discrimination unreliable. VideoGLaMM and GLUS frequently jump to nearby birds with similar trajectories, and their masks destabilize as the flock spreads. UniPixel occasionally identifies the correct target but struggles to maintain identity during fast movement, often switching to birds at comparable heights. SPARROW consistently isolates the correct bird across the entire sequence. It maintains identity despite intersecting flight paths and brief occlusions, showing that the model can track a specific target even when spatial cues are subtle and temporal ambiguity is high. This example highlights SPARROW's advantage in large, multi-instance scenes where appearance similarity and rapid motion typically cause grounding failures.

Ref-DAVIS17. Fig. 11 shows a multi-query example involving several goldfish described by appearance and position (largest, smallest, center, end, bottom). VideoGLaMM produces overlapping masks and inconsistent instance separation, struggling to honor size- and location-based cues when the fish cluster tightly or share similar color. SPARROW cleanly distinguishes all five targets, maintaining boundaries even under occlusion and subtle pose changes. This demonstrates the model's ability to resolve fine-grained appearance references without collapsing instances.

Fig. 12 presents another multi-object case with three expressions: "a back shooting gun," "a black man," and "a rope." GLUS frequently produces masks that bleed across object boundaries, especially for thin structures such as the gun barrel and rope, and it occasionally merges the person with surrounding objects. SPARROW delivers sharp, stable masks for all three referents. The gun remains well-defined even during muzzle-flash frames, the person is consistently localized through large pose changes, and the rope retains its structure without fragmentation. This example highlights SPARROW's improved precision on small or elongated objects.

Fig. 13 shows an example with two linked appearance-based queries: "a man in a suit riding a scooter" and "a black scooter ridden by a man." UniPixel often produces fragmented masks, struggles to capture the full geometry

Figure 12. **Qualitative comparison on the Ref-DAVIS17 RVOS task.** For the queries "a back shooting gun," "a black man," and "a rope," our method cleanly grounds all three described targets from the first frame and maintains precise, instance-specific masks throughout the sequence. It preserves accurate boundaries and stable localization even on thin or small objects such as the gun barrel and rope. In contrast, GLUS produces unstable, overlapping masks, frequently blending instances or losing fine structural details under motion and occlusion. Our approach demonstrates stronger fine-grained appearance grounding and consistent multi-object separation across all frames.

Figure 13. **Qualitative comparison on Ref-DAVIS17 (RVOS).** For the expressions "a man in a suit riding a scooter" and "a black scooter ridden by a man," our method accurately grounds both the rider and scooter from the first frame and maintains stable, instance-consistent masks throughout the sequence, preserving sharp boundaries despite significant motion and viewpoint changes. In contrast, UniPixel produces fragmented masks, inconsistent localization, and frequent structural loss. Our approach demonstrates stronger appearance-based grounding and greater robustness to multi-object consistency across frames.

of the scooter, and inconsistently localizes the rider during viewpoint changes. SPARROW maintains coherent masks for both the rider and the scooter across the entire clip, preserving structural details such as the rear wheel and handlebar region. The model consistently assigns each query to the correct instance despite motion, partial occlusion, and background variation, illustrating stronger appearance consistency and temporal stability.

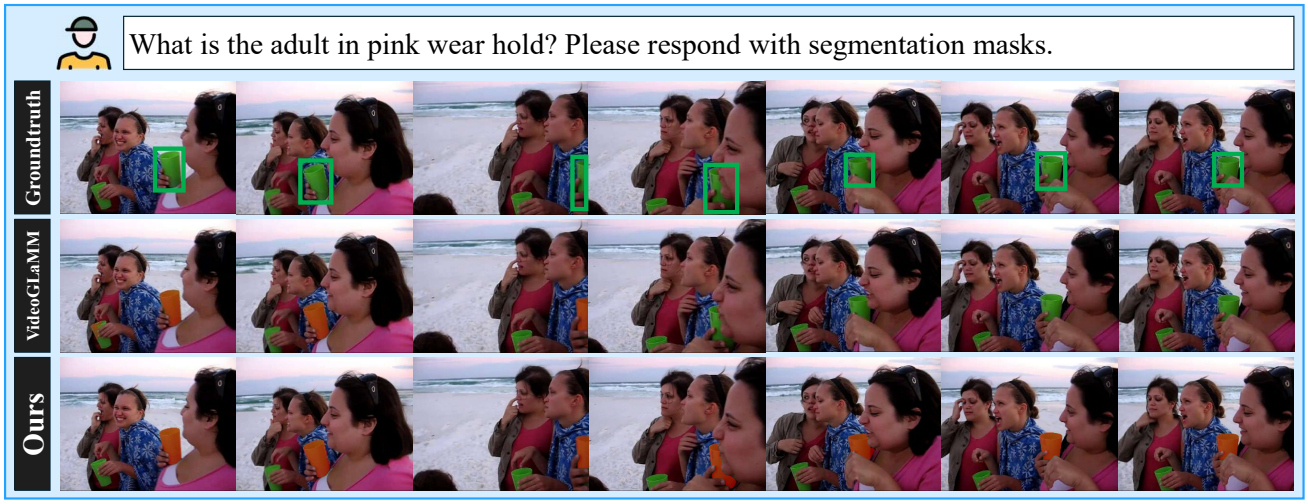

Figure 14. **Qualitative comparison on the VidSTG VG task.** For the query "What is the adult in pink wear hold?," our method consistently grounds the correct object—the orange cup—from the first frame and maintains accurate, stable masks despite hand motion, partial occlusions, and frequent interaction among multiple people. It preserves precise spatial extent and clean boundaries across the entire sequence. In contrast, VideoGLaMM produces drifting or incomplete masks and struggles to maintain consistent localization of the held object during motion. Our approach demonstrates strong spatio-temporal grounding and reliable object tracking on VidSTG.

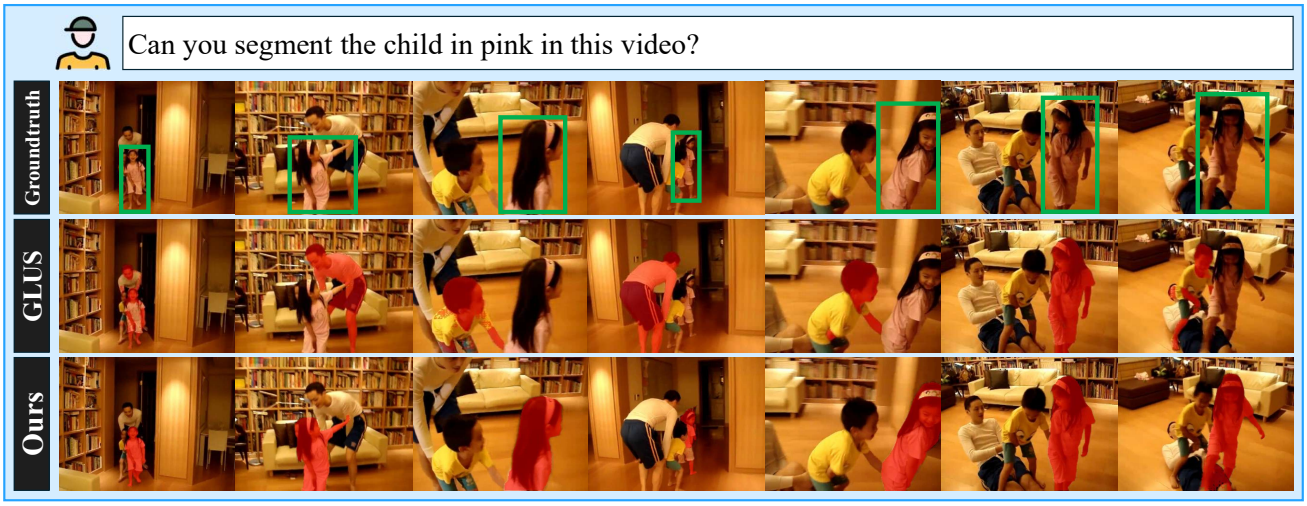

Figure 15. **Qualitative comparison on the VidSTG VG task.** For the query "the child in pink," our method accurately segments the correct child from the first frame and maintains stable masks throughout the sequence, preserving clean boundaries and consistent localization despite rapid motion, occlusions, and nearby interactions. In contrast, GLUS often drifts to other people or yields incomplete, unstable masks under motion, whereas our approach remains robust to distractors and maintains stronger temporal consistency.

C.2. Visual Grounding

Fig. 14 illustrates a grounding query from VidSTG: "What is the adult in pink wear hold?" The task requires identifying the adult in pink and localizing the object she is holding across the sequence. VideoGLaMM frequently produces incomplete or drifting masks, especially when the hands partially occlude the object or the viewpoint shifts. SPARROW consistently identifies the correct object, an orange cup, with clear boundaries and stable localization throughout the clip, demonstrating reliable spatio-temporal grounding under subtle hand–object interactions.

Fig. 15 presents a second grounding scenario involving the referent "the child in pink." The scene includes multiple interacting children and adults, frequent occlusions, and rapid motion. GLUS exhibits noticeable drift, intermittently switching to nearby individuals or failing to capture the full extent of the target during pose changes. SPARROW main-

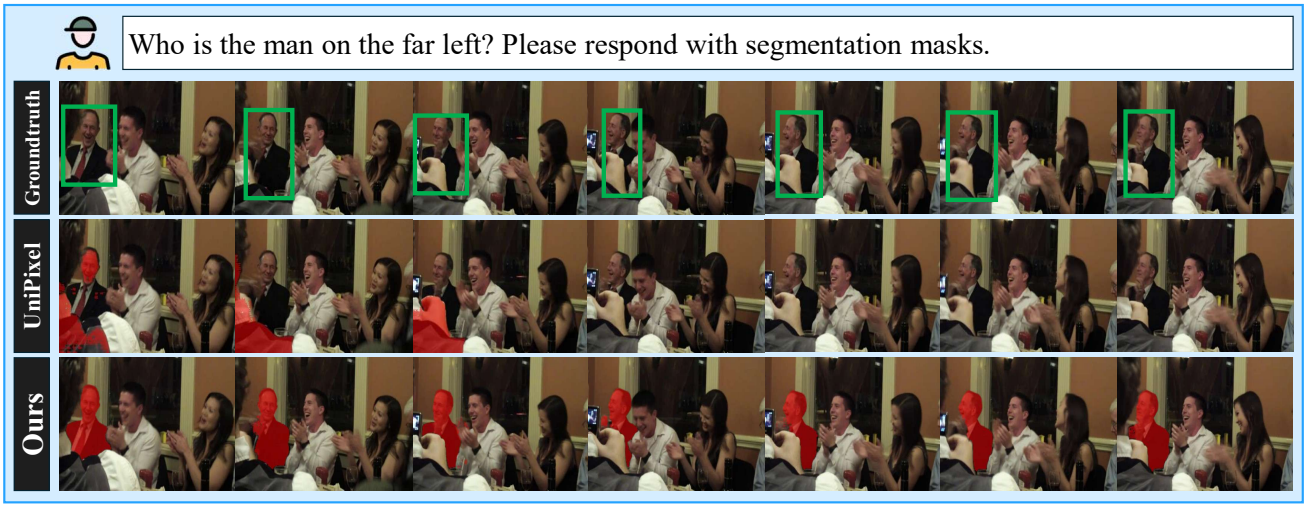

Figure 16. **Qualitative comparison on the VidSTG VG task.** For the query "Who is the man on the far left?," our method accurately grounds the correct individual from the first frame and preserves consistent identity assignments throughout the sequence, even during dense group interactions and frequent occlusions. It maintains stable masks and precise spatial localization across all frames. In contrast, UniPixel exhibits identity drift and inconsistent mask assignment, especially in cluttered moments with overlapping people. Our approach demonstrates stronger temporal grounding and robust identity preservation in challenging crowd scenarios.

tains a stable, identity-consistent mask across the entire sequence, tracking the correct child even during occlusion and close-proximity interactions. This example highlights SPARROW's robustness to distractors and improved temporal coherence in cluttered environments.

Finally, Fig. 16 illustrates the query *"Who is the man on the far left?"*, which requires maintaining identity under subtle pose changes in a crowded scene. UniPixel often shifts to adjacent individuals when poses or gestures are similar, causing identity switches and unstable boundaries. In contrast, SPARROW maintains the correct referent throughout, producing clean masks anchored to the intended individual despite crowd motion and visual ambiguity. This example highlights its robustness in identity-sensitive grounding with multiple similar candidates.

C.3. Video Grounded Conversation Generation (Video GCG)

Fig. 17 shows a clip where a woman in an orange shirt prepares ingredients by placing a potato in water, chopping more potatoes, and transferring them into a pot. VideoGLaMM generates a fluent narrative but frequently attributes actions that are not supported by the visual evidence, and its grounding does not reliably match the manipulated objects. GLUS fails to produce any grounded response, returning only `[SEG]`. UniPixel provides a coarse description but overlooks several action transitions and under-segments tool–hand interactions. In contrast, SPARROW accurately follows the sequence, generating grounded text aligned with each sub-action and maintaining precise masks on hands, utensils, and ingredients. This example highlights SPARROW's ability to couple fine-grained temporal actions with pixel-level grounding for grounded conversation generation.

D. TSF Usage, Tradeoff, and Design

D.1. TSF Test-time Overhead

Relative to the baselines, enabling TSF adds (1) a single GroundingDINO pass on the first frame (batched multi-prompt inference), (2) a CLDTracker pass per frame per target, (3) CLIP ViT-L/14@336 crop-feature extraction per frame per target (batched on GPU), (4) lightweight K-means clustering per target (K=4 centroids), (5) a V→L projection generating four TSF tokens per target, and (6) four additional LLM tokens per target. On an A100, the empirical latency is well approximated by $\Delta t_{\text{TSF}}(n, T) \approx 0.093 + 0.020\, n\, T$ s, where n and T denote the number of tracked targets and frames, respectively. The detailed component-wise breakdown is summarized in Table 7. The cost scales linearly with both n and T, dominated by CLDTracker and CLIP crops, while all other components are negligible ($<$ 5 ms total). For n=3, T=300 (10 s at 30 FPS), the added time is $\approx$ 18.1 s ($\sim$60 ms/frame, or 20 ms/target/frame).

D.2. Accuracy–Latency Tradeoff (Train-only vs. Train+Inference)

The quantitative comparison of TSF usage modes is reported in the main paper. Training-only TSF improves

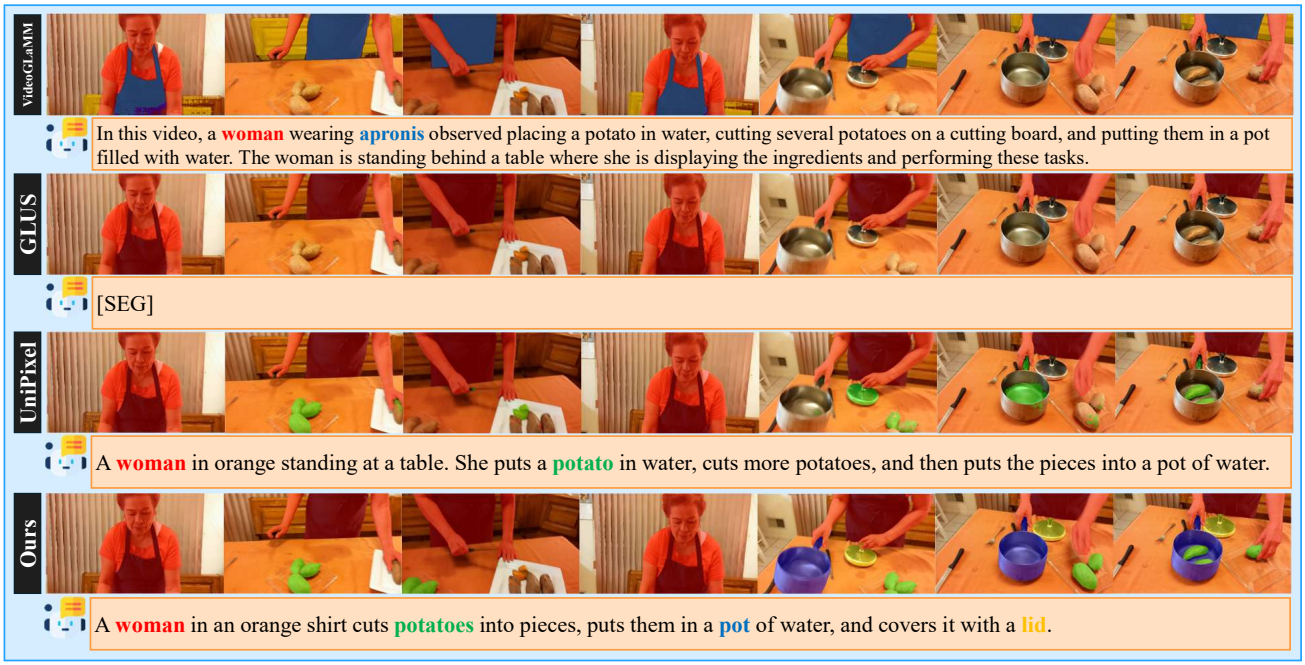

Figure 17. **Qualitative comparison on the Video GCG task.** In a cooking sequence where a woman places a potato in water, chops additional potatoes, and transfers the pieces into a pot, our method maintains accurate pixel-level grounding of all manipulated objects across the full series of actions. It produces a coherent, temporally aligned description that faithfully reflects each step of the manipulation process. In contrast, VideoGLaMM generates fluent but loosely grounded narratives, GLUS fails to produce a meaningful grounded response, and UniPixel captures only coarse actions while struggling with temporal transitions and hand–object interactions. Our approach demonstrates robust spatio-temporal grounding and precise correspondence between language and visual evidence throughout the sequence.

Table 7. **Component-wise analysis: TSF overhead at test time on A100.**

Component	Per-unit latency	Count	Added time (s)
Vision + Tracking			
GroundingDINO (first frame)	90 ms	1	0.090
CLDTracker	18 ms / frame / target	nT	$0.018\,nT$
CLIP ViT-L/14@336 (crops)	2 ms / crop (batched)	nT	$0.002\,nT$
Token construction + LLM			
K-means clustering (K=4)	$<$ 1 ms / target	n	≈ 0
V→L projection ($4n$ tok.)	0.1 ms / token	$4n$	$0.0004\,n$
LLM extra sequence (+$4n$ tok.)	0.5–1 ms / token	$4n$	$0.003\,n$
Total			$\Delta t_{\text{TSF}}(n,T) \approx 0.093 + 0.020\,nT$

$\mathcal{J}\&\mathcal{F}$ by +2.9 without any inference-time tracking, indicating that pseudo-tracked supervision teaches target persistence and reduces identity switches; when combined with `[BOX]`, the gain increases to +7.3 while remaining overhead-free. Train+inference TSF achieves the highest score (77.7, +8.2) but incurs the additional tracking and token-construction overhead summarized in Table 7. The extra gain reflects the benefit of supplying temporally grounded identity cues directly during decoding, especially for longer sequences where geometric drift accumulates. In practice, Train-only TSF is the default choice for latency-sensitive settings, while inference-time TSF can be enabled when maximum temporal stability outweighs the overhead.

When to Enable TSF at Inference By default we use *Train-only TSF*, i.e., TSF is used during training but omitted at inference. TSF-at-inference is most beneficial under challenging temporal conditions: (i) occlusion and reappearance, (ii) fast motion or motion blur, (iii) small or fragile targets, and (iv) crowded scenes with similar distractors. Fig. 18 shows a representative example where inference-time TSF reduces target dropout and drift. When

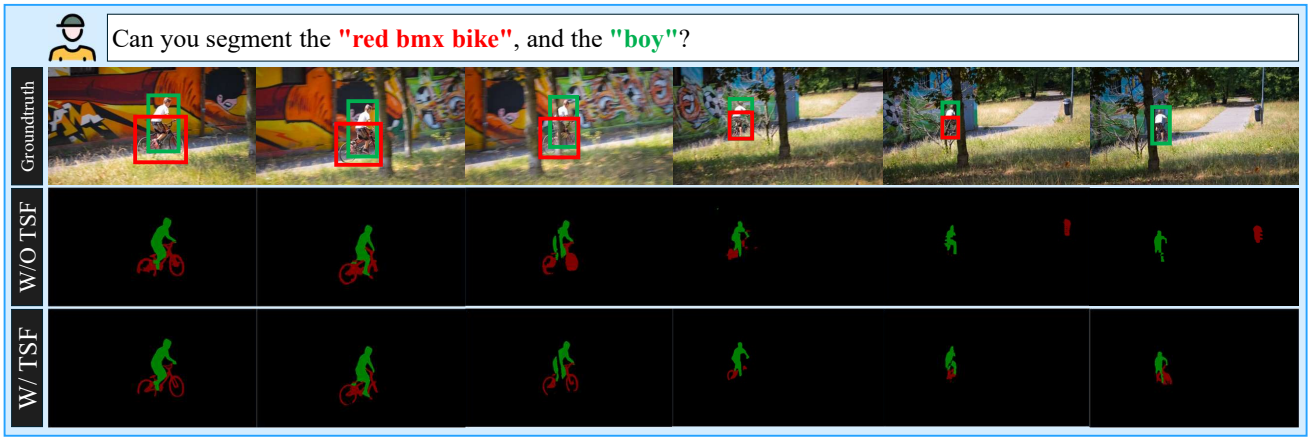

Figure 18. **When to enable TSF at inference (qualitative).** Query: segment the "red bmx bike" and the "boy". *TSF Off (default)*: the small/fragile referent (bike) is prone to *target dropout* and *drift/false positives* as the target becomes small or partially occluded. **TSF On (optional max-stability):** TSF provides a tracked, target-specific identity cue at inference, preserving consistent referent segmentation.

Table 8. **When to enable TSF at inference.**

Scenario	Recommendation
Short/simple clips, limited motion	Keep TSF off
Fast motion or motion blur	Enable TSF if accuracy is critical
Occlusion and re-appearance	Enable TSF
Small or fragile targets	Enable TSF if target dropout is likely
Crowded scenes with similar distractors	Enable TSF

enabled, the added cost is approximately ∼20 ms per target per frame on A100 (Table 7). Practical guidance is summarized in Table 8. This provides a simple accuracy–latency knob: keep TSF off at inference by default and enable it only when additional stability is critical.

D.3. TSF Injection Path Ablation

In SPARROW, TSF tokens are extracted from tracked region crops, encoded by the spatial visual encoder, and projected into the LLM space through the spatial V–L projector. This design is intentional: TSF is intended to serve as an object-centric appearance and identity cue, and its feature distribution is therefore naturally aligned with the spatial stream. By contrast, the temporal V–L projector is optimized for global spatio-temporal video features rather than crop-level appearance features. To validate this design choice, we compare several TSF injection variants while keeping the rest of the pipeline unchanged: (i) **No TSF**, where no TSF tokens are injected; (ii) **spatial-only**, where TSF is projected only through the spatial V–L projector; (iii) **temporal-only**, where the same crop-based TSF features are projected through the temporal V–L projector; and (iv) **dual-path** variants, which combine both projectors either by concatenation or summation.

Table 9 shows that **spatial-only** performs best. Projecting crop-based TSF features through the temporal projector alone does not improve over the no-TSF baseline, suggesting that the temporal projection space is not well matched to object-centric supervision. Dual-path variants provide moderate gains, but remain below the spatial-only design, indicating that mixing projection spaces can introduce redundant or mismatched cues that weaken the identity signal.

Table 9. **TSF injection path ablation on Ref-DAVIS17 (val).** We vary only the TSF projection path. "tok." denotes the number of additional TSF tokens per target.

TSF projection design	Extra tok.	J&F ↑
No TSF	0	72.5
Temporal-only: $W_h(F_g(B_j))$	+4	72.5
Dual-path (concat): $[W_g(\cdot); W_h(\cdot)]$	+8	74.9
Dual-path (sum): $W_g(\cdot) + W_h(\cdot)$	+4	74.7
Spatial-only (ours): $W_g(F_g(B_j))$	+4	**76.8**

D.4. Robustness to Noisy TSF Supervision

Since TSF supervision is constructed from pseudo-trajectories produced by GroundingDINO and CLDTracker, its quality may be affected by localization noise or occasional identity switches. We therefore stress-test robustness by corrupting only the training-time trajectories used for TSF extraction while keeping the model architecture, training schedules, and inference settings fixed.

Protocol. We evaluate on Ref-DAVIS17 (val) under the default Train-only TSF setting with `[BOX]` enabled. Only the teacher trajectories used to construct TSF tokens during training are corrupted; inference remains unchanged and does not use TSF tokens.

Corruption types. We consider three forms of corruption: (1) **Box jitter:** per frame, the box center is randomly shifted and its width/height are randomly rescaled by a fac-

Table 10. **Robustness of SPARROW to noisy TSF supervision on Ref-DAVIS17 (val).** We corrupt only the training-time teacher trajectories before TSF token extraction; the model architecture, training schedule, and inference setting remain unchanged. All results use the default Train-only TSF setting with `[BOX]` enabled.

Corruption applied to training trajectories	Level	J&F ↑
None (clean TSF)	–	76.8
Box jitter (random shift/scale)	±5%	76.5
	±10%	76.0
	±20%	75.2
ID-switch injection (contiguous segments)	5% frames	76.1
	10% frames	75.4
	20% frames	74.3
Trajectory dropout (remove TSF tokens)	10%	76.6
	30%	76.0
	50%	75.0
No TSF baseline (`[BOX]` on)	–	72.5

tor controlled by $p \in \{5\%, 10\%, 20\%\}$. (2) **ID-switch injection:** per trajectory, we replace contiguous trajectory segments whose total length is $p\%$ of frames with another trajectory from the same video, simulating identity mismatches. (3) **Trajectory dropout:** TSF tokens are removed for a fraction $p\%$ of frames prior to TSF token construction.

Results. Table 10 shows that performance degrades gracefully as training-time TSF supervision becomes noisier. Under moderate corruption, SPARROW remains consistently above the *No-TSF* baseline, indicating that TSF does not require perfectly accurate teacher trajectories to be beneficial. For example, with 10% ID-switch injection, SPARROW achieves 75.4 J&F, which remains +2.9 above the No-TSF baseline (72.5). These results suggest that the TSF mechanism is robust to realistic supervision noise.

E. Comparison of Proposal Head Variants

To justify our Transformer-based proposal module, we compare it with simpler alternatives under the same frozen-feature setting, where all methods operate on frozen SAM2 Hiera-L features. We evaluate: (i) **Direct LLM coordinates**, a proposer-free variant that predicts boxes directly from language outputs (Shikra-style); (ii) a **single-box MLP regressor** on pooled frozen visual features; and (iii) an **anchor-free dense convolutional head** (FCOS-style) with objectness, box offsets, and top-K selection. For completeness, we also include **DETR** and **Deformable-DETR** decoder heads.

Performance and efficiency. Table 11 shows that direct regression and shallow heads underperform under frozen visual features, suggesting that this regime requires stronger proposal reasoning than a lightweight box predictor can provide. The anchor-free dense head improves over direct regression but remains below Transformer decoders in $\mathcal{J}\&\mathcal{F}$. DETR-style proposers perform best overall, consistent with stronger class-agnostic objectness modeling and improved proposal coverage. Among all variants, **Deformable-DETR** achieves the best accuracy–efficiency tradeoff, improving over DETR while slightly reducing parameters and latency. Notably, the added parameters remain negligible compared to the MLLM's billion-param scale.

Table 11. **Proposal generation variants (frozen SAM2 Hiera-L features).** Performance on Ref-DAVIS17 (val) measured by $\mathcal{J}\&\mathcal{F}$. *Latency is the incremental overhead of the proposal head only* (feature extraction shared).

Method	+Params (M)↓	+Latency (ms)↓	$\mathcal{J}\&\mathcal{F}$ ↑
Direct LLM coords (single box)	0.00	0.0	67.8
MLP regressor (single box)	0.35	0.4	68.5
Anchor-free conv head	1.90	1.2	70.4
DETR decoder head	12.33	6.2	71.2
Deformable-DETR decoder head (ours)	11.38	5.6	72.5

Proposal-set recall analysis. Because the language-conditioned filtering stage can only select from the available proposals, any ground-truth object missed at this stage is unrecoverable. We therefore evaluate oracle recall, defined as whether any of the top-K predicted boxes overlap a ground-truth box above an IoU threshold.

Table 12. **Proposal-set oracle recall** under frozen SAM2 Hiera-L features. Recall@IoU measures whether any of the top-K boxes overlaps a GT box above the threshold.

Method	K	Recall@0.5↑	Recall@0.75↑
Direct LLM coords (single box)	1	50.0	20.0
MLP regressor (single box)	1	55.0	28.0
Anchor-free conv head	300	86.0	60.0
DETR decoder head	300	90.0	65.0
Deformable-DETR decoder head (ours)	300	93.0	70.0

As shown in Table 12, Transformer-based proposers achieve higher oracle recall than regression or dense prediction heads, yielding a higher-coverage candidate set for language-conditioned selection and refinement. This matches our design goal: the proposer maximizes candidate coverage, while the language module focuses on selecting and refining the correct region rather than compensating for missed detections.

F. Compute, Training, and Overhead

F.1. Stage-wise Cost Analysis

SPARROW introduces two additional components relative to baseline video-MLLM pipelines: (i) a one-time offline TSF supervision generation stage and (ii) a short two-stage fine-tuning procedure. Using the latency model in Appendix D.1, processing a typical clip (n=1, T=300) for TSF generation requires approximately $\sim$ 6.1 s (Table 7). This preprocessing step is executed once, is not part of the

training loop, and does not affect default inference.

Training proceeds in two stages: *Stage 1 (TSF injection)*, where only multimodal adapters and LoRA parameters are optimized while the visual encoders and SAM2 decoder remain frozen; and *Stage 2 (Dual-prompt box grounding)*, where the proposal generator is frozen and only the [BOX] adapter and filtration head are updated.

Table 13. **Training cost breakdown for SPARROW.** Offline TSF generation is one-time preprocessing and is not part of the training loop or default inference pipeline.

Component	Wall-clock	GPU-hours	A100 GPU-days
Offline TSF precomputation (30,646 clips)	51.9 h (1×A100)	51.9	2.16
Baseline fine-tuning (per backbone)	10 h (8×A100)	80	3.33
Stage 1: TSF injection	12 h (8×A100)	96	4.00
Stage 2: Dual-prompt box learning	2 h (8×A100)	16	0.67
SPARROW training (S1 + S2)	**14 h**	**112**	**4.67**
Increment over baseline	–	**+32**	**+1.34**

As shown in Table 13, the combined SPARROW training (Stages 1+2) requires 112 GPU-hours (4.67 A100 GPU-days), corresponding to a modest +32 GPU-hours over baseline fine-tuning. To facilitate reproducibility, we release precomputed TSF trajectories and tokens, eliminating the need to rerun detection and tracking.

Table 14. **Execution summary across preprocessing, training, and inference.**

Component	Stage	When executed	Runtime cost
GroundingDINO + CLDTracker	TSF generation	Offline (one-time)	∼6.1 s / clip
Crop encoding + K-means	TSF generation	Offline (one-time)	included in total TSF cost
TSF Injection (Stage 1)	Training	Per backbone	∼4.0 GPU-days
Dual-Prompt Learning (Stage 2)	Training	Per backbone	∼0.67 GPU-days
SPARROW inference (default)	Inference	Per video	no external detector/tracker
TSF-enabled inference (optional)	Inference	Per frame	∼20 ms / target / frame

Table 14 clarifies when each component is executed. Heavy tracking modules (GroundingDINO and CLDTracker) are confined to the one-time TSF generation stage and are never invoked during default inference, preserving the deployment characteristics of the underlying backbone. TSF-at-inference remains an optional mode that adds ∼20 ms per target per frame (Table 7).

F.2. Inference Overhead

We evaluate the inference overhead of SPARROW under identical evaluation settings for each backbone. Unless otherwise stated, all measurements are obtained on a single A100 GPU with batch size 1, using the same input resolution, frame sampling strategy, clip length, decoding settings, and evaluation pipeline as in the main experiments.

Inference setting. By default, SPARROW does not use TSF during inference. Accordingly, GroundingDINO, CLDTracker, CLIP-based crop processing, and K-means selection are not executed at test time. The resulting runtime overhead comes only from the added dual-prompt modules and the class-agnostic proposal head.

Reported metrics. We report three quantities: (i) end-to-end throughput in frames per second (FPS), (ii) vision GFLOPs per frame for the modules executed at inference, and (iii) total parameter count. GFLOPs/frame includes the visual backbone/encoder, mask decoder, and SPARROW heads/proposer, but excludes autoregressive LLM token generation, as its cost depends on the output length. End-to-end FPS captures the full runtime impact, including decoding, and is measured by averaging over the Ref-DAVIS17 (val) evaluation pipeline.

Table 15. **Inference overhead of SPARROW under identical evaluation settings.** We report total parameters (in Billion), end-to-end FPS, and vision GFLOPs per frame. +SP denotes the backbone augmented with SPARROW modules.

Backbone	Params	FPS	GFLOPs/f	+SP Params	+SP FPS	+SP GFLOPs/f
VideoGLaMM	5.435	2.40	38927	5.452	2.40	38962
UniPixel	3.881	15.38	1530	3.898	15.04	1565
GLUS	7.288	6.374	2680	7.305	6.29	2715

Results. Table 15 shows that SPARROW introduces only modest inference overhead across all three backbones. The parameter increase is limited to +0.017B, while the throughput reduction is minor: VideoGLaMM is unchanged within measurement noise (2.40→2.40 FPS), UniPixel drops from 15.38 to 15.04 FPS, and GLUS drops from 6.374 to 6.29 FPS. The increase in GFLOPs/frame is similarly small, indicating that the performance gains are not driven by substantial additional test-time computation.