Supplement Material

Anonymous submission

1 Eventstream-like Data Visualization

Figure 1 illustrates eventstream-like representations extracted from video sequences, validating their effectiveness in capturing motion dynamics. Each example comprises, from top to bottom, the eventstream-like representation, the corresponding RGB frame, and the ground truth segmentation mask.

As shown, the eventstream-like data effectively highlights regions with significant pixel-level intensity changes, which are closely associated with object motion. Interestingly, these representations exhibit characteristics similar to those observed in real event camera data, where motion-induced positive and negative polarity changes are clearly discernible. Regions involving abnormal or object-induced motion are distinctly emphasized and show strong alignment with the ground truth.

These visualizations demonstrate that the eventstream-like representation provides valuable and interpretable motion cues. By encoding pixel-wise temporal intensity variations, it complements appearance information and substantially improves the model's ability to detect and localize moving objects, even under challenging scenarios such as background clutter or low-texture regions.

2 Eventstream-like Data Acquiring

Here we present the Pesudocode of the Eventstream-like data in Alg. 1.

Algorithm 1 details the extraction of an interpretable eventstream-like representation for motion modeling. Given two consecutive RGB frames I_{t-1} and I_t, Step 1 converts them into grayscale images G_{t-1} and G_t to suppress color noise and reduce computation. Step 2 compensates for global motion by estimating a homography matrix H via ORB feature matching, then aligns G_{t-1} to obtain G'_t.

In Step 3, a motion residual map is computed as:

$$D_t = G_t - G'_t.$$

In Step 4, we generate positive and negative motion maps by comparing D_t with a locally adaptive threshold τ_t computed from Gaussian statistics:

$$P_t(u,v) = 1\left[D_t(u,v) > \tau_t(u,v)\right],$$

$$N_t(u,v) = 1\left[D_t(u,v) < -\tau_t(u,v)\right].$$

Algorithm 1: Eventstream-like Data Extraction with Global Motion Compensation

Input: Consecutive RGB frames I_{t-1} and I_t
Output: Eventstream-like representation E_t

Step 1: Grayscale Conversion
Suppress color noise and reduce complexity:
$G_{t-1} \leftarrow \text{Grayscale}(I_{t-1})$
$G_t \leftarrow \text{Grayscale}(I_t)$

Step 2: Global Motion Compensation
Estimate homography via ORB matching with RANSAC and align frames:
$H \leftarrow \text{EstimateHomography}(G_{t-1}, G_t)$
$G'_t \leftarrow \text{Warp}(G_{t-1}, H)$

Step 3: Motion Residual Computation
Compute residual map:
$D_t \leftarrow G_t - G'_t$

Step 4: Motion Map Generation
Compute adaptive threshold map based on local Gaussian statistics:
$\tau_t \leftarrow \text{AdaptiveThreshold}(D_t)$
Generate binary positive and negative motion maps:
$P_t(u,v) = 1\left[D_t(u,v) > \tau_t(u,v)\right]$
$N_t(u,v) = 1\left[D_t(u,v) < -\tau_t(u,v)\right]$

Step 5: Eventstream Construction
Projecting motion maps with RGB channels:
$E_t^1(u,v) = P_t(u,v) \cdot R_t(u,v)$
$E_t^2(u,v) = N_t(u,v) \cdot B_t(u,v)$
$E_t = [E_t^1, E_t^2]$

Step 6: Morphological Refinement
Remove spurious blobs:
$E_t \leftarrow \text{MorphologicalOps}(E_t)$

return E_t

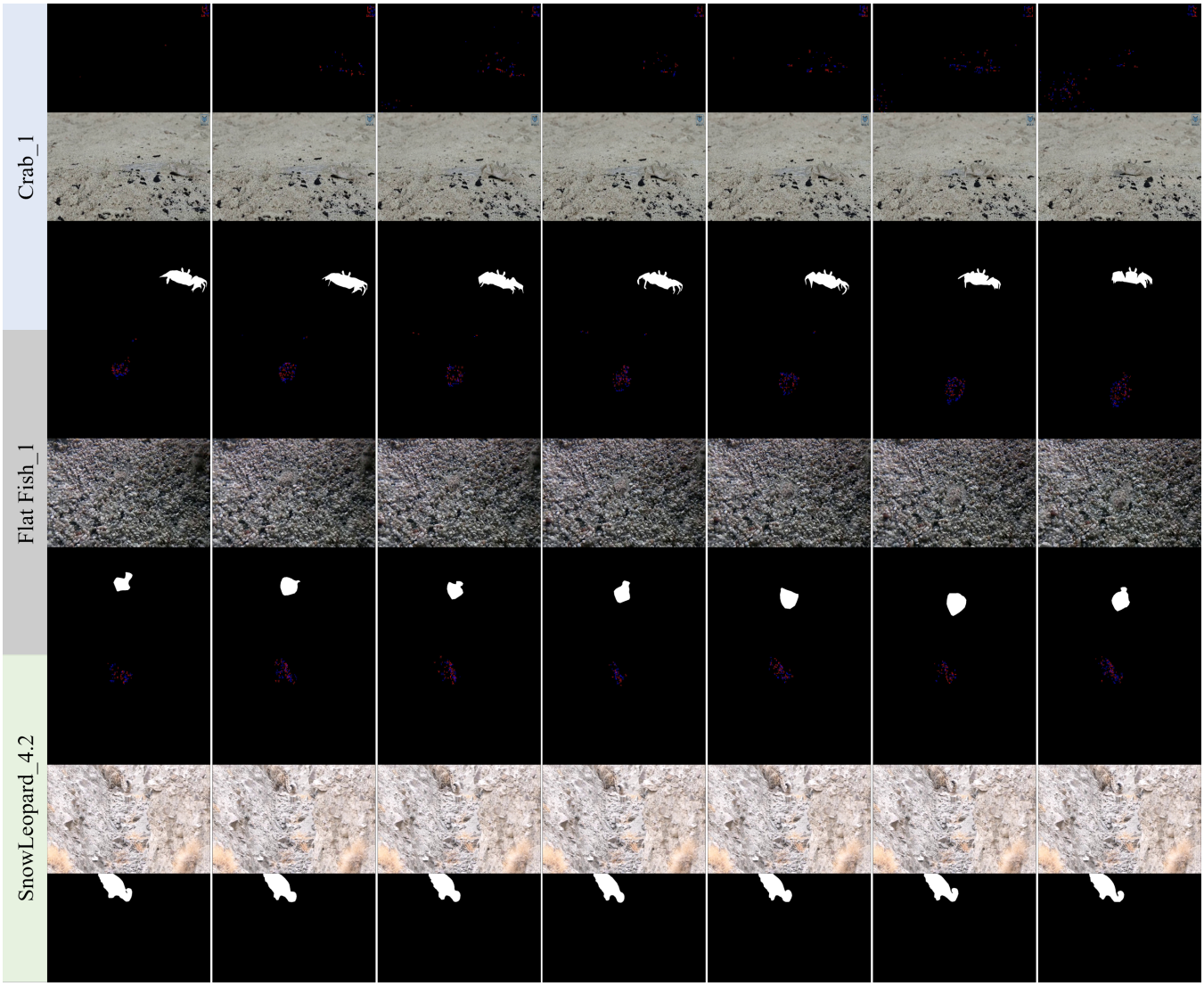

Figure 1: Visualization of eventstream-like data. Each example shows, from top to bottom, the generated eventstream-like representation, the corresponding RGB frame, and the ground truth mask. (Best viewed by zooming in.)

Step 5 project motion maps with the red and blue channels of I_t:

$$E_t^1(u,v) = P_t(u,v) \cdot R_t(u,v), \quad E_t^2(u,v) = N_t(u,v) \cdot B_t(u,v).$$

The resulting eventstream is:

$$E_t = [E_t^1, E_t^2].$$

Finally, Step 6 applies morphological operations to remove irrelevant blobs, resulting in the final eventstream-like representation E_t, which jointly encodes motion and appearance for downstream modeling.

3 Meta-Parameter Selection

To determine suitable weights for embedding and mask losses, we conduct an extensive grid search, as reported in Table 1. We vary the embedding loss weight in $\{0.5, 1.0, 1.5, 2.0\}$ and the mask loss weight in $\{0.2, 0.5, 1.0, 1.5\}$, resulting in 16 combinations. The best performance is achieved when setting the embedding loss to 1.0 and the mask loss to 0.5, yielding consistent improvements across all metrics.

It is worth noting that although both losses incorporate structural supervision (**?**), they serve different purposes and are activated differently. The embedding loss supervises the intermediate prompt prediction and is only active on condition frames (the first frame of each sequence), resulting in sparse gradients. In contrast, the mask loss is applied at every timestep, contributing to stable and temporally consistent learning throughout the sequence.

4 Two-Step Training vs. Mixed Training

To validate the effectiveness of our two-step training strategy, we compare it with a mixed training baseline. In the

Table 1: Performance comparison under different embedding and mask loss weight combinations on MoCA-Mask. Best results are marked in **bold**.

Emb Loss	Mask Loss	MoCA-Mask					
		$S_\alpha \uparrow$	$F_\beta^\omega \uparrow$	$E_m \uparrow$	$M \downarrow$	mDice ↑	mIoU ↑
0.5	0.2	0.627	0.314	0.756	0.018	0.317	0.280
	0.5	0.622	0.299	0.780	0.020	0.304	0.275
	1.0	0.726	0.520	0.877	0.011	0.521	0.452
	1.5	0.644	0.513	0.744	0.023	0.366	0.309
1.0	0.2	0.668	0.408	0.822	0.016	0.411	0.359
	0.5	**0.753**	**0.573**	**0.855**	**0.009**	**0.574**	**0.496**
	1.0	0.642	0.303	0.780	0.018	0.310	0.279
	1.5	0.676	0.414	0.802	0.012	0.415	0.307
1.5	0.2	0.684	0.383	0.820	0.019	0.396	0.296
	0.5	0.618	0.303	0.635	0.027	0.311	0.281
	1.0	0.744	0.547	0.857	0.011	0.562	0.483
	1.5	0.629	0.289	0.688	0.023	0.284	0.243
2.0	0.2	0.630	0.311	0.730	0.013	0.316	0.286
	0.5	0.662	0.401	0.724	0.021	0.305	0.351
	1.0	0.598	0.257	0.722	0.025	0.263	0.236
	1.5	0.734	0.525	0.850	0.017	0.537	0.461

Table 2: Comparison of two-step training and mixed training on the MoCA-Mask dataset.

Training Method	$S_\alpha \uparrow$	$F_\beta^\omega \uparrow$	$E_m \uparrow$	$M \downarrow$	mDice ↑	mIoU ↑
Mixed Training	0.596	0.240	0.624	0.012	0.244	0.201
Ours (Two-Step)	**0.753**	**0.573**	**0.855**	**0.009**	**0.574**	**0.496**

mixed training setup, the image dataset COD10K and the video dataset MoCA are merged, and the model is trained jointly in a single stage. As shown in Table 2, mixed training leads to a notable performance drop compared to our two-step training, even when applied to the vanilla SAM2.

We attribute this performance degradation to two main factors. First, there is a substantial domain gap between image-based and video-based camouflaged object detection datasets. Second, the embedding loss is selectively activated only on conditional frames (i.e., frames with index zero). Consequently, during mixed training, the embedding loss is constantly applied to image data but only sparsely applied to video data, resulting in imbalanced gradient updates and sub-optimal learning. This imbalance negatively impacts the model's ability to generalize effectively across video sequences.

5 Supplementary Qualitative Results

Figure 2 presents additional qualitative results on the CAD2016 dataset, covering three representative video sequences: *frog*, *scorpion*, and *snail*. Each sequence highlights different challenges commonly encountered in camouflaged object detection, and we analyze the performance of our method under these conditions.

In the **frog** sequence, the foreground object exhibits strong color similarity with the background, further complicated by severe motion blur, especially in the middle frames. Despite these challenging conditions, our method accurately localizes and segments the frog, preserving its overall structure and boundary details, even when the object appears heavily blurred (see columns 3 and 4). The predicted masks closely align with the ground truth, demonstrating robustness to both color camouflage and motion degradation.

In the **scorpion** sequence, the primary challenge stems from persistent occlusions. Large portions of the scorpion's body are frequently obscured, with only partial visibility (e.g., the tail in columns 1 and 2). Our method consistently tracks the scorpion's spatial extent and successfully recovers its complete body structure by leveraging temporal information, showing strong resilience to partial occlusions.

In the **snail** sequence, the main difficulty lies in capturing fine-grained structures, particularly the thin tentacles of the snail. As shown, our method faithfully recovers these subtle details and produces segmentations that closely match the ground truth. This demonstrates the model's capability to preserve intricate structures.

Overall, these supplementary results further validate the generalization ability of our approach. Our method consistently achieves accurate and high-quality segmentations under diverse challenging scenarios, including motion blur, occlusion, and fine-structure recovery.

6 Discussion on MoCA & CAD2016 Datasets

Under identical experimental settings, the performance on CAD2016 consistently surpasses that on MoCA, even when training is performed solely on MoCA-mask data. To investigate this discrepancy, we analyze the foreground distribution of both datasets.

As shown in Figures 3 and 4, the average foreground pro-

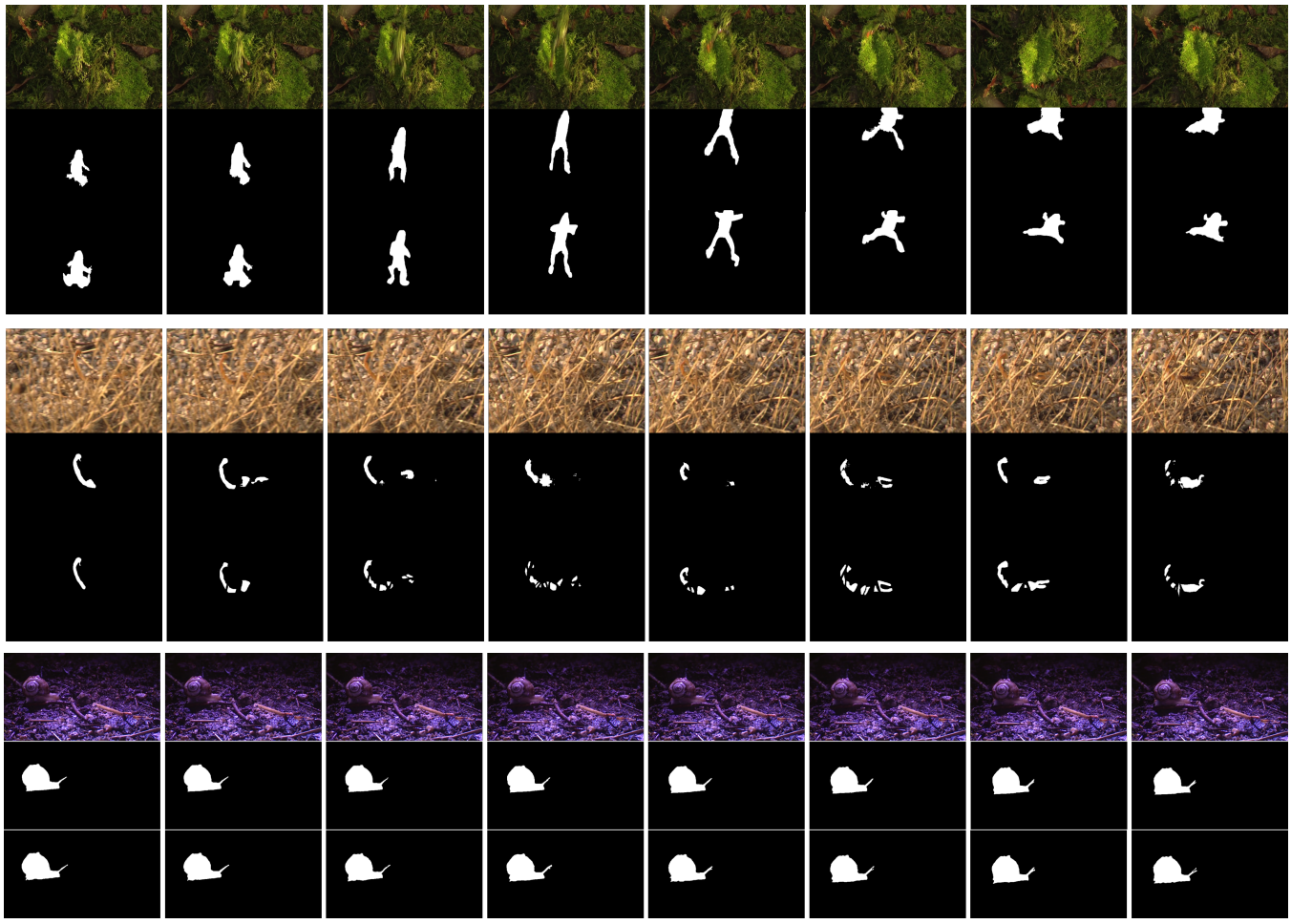

Figure 2: Additional qualitative results on the CAD2016 dataset. The figure shows three representative sequences: *frog*, *scorpion*, and *snail* (from top to bottom). Each sample includes the current frame, the predicted mask by our method, and the corresponding ground truth. (Best viewed by zooming in.)

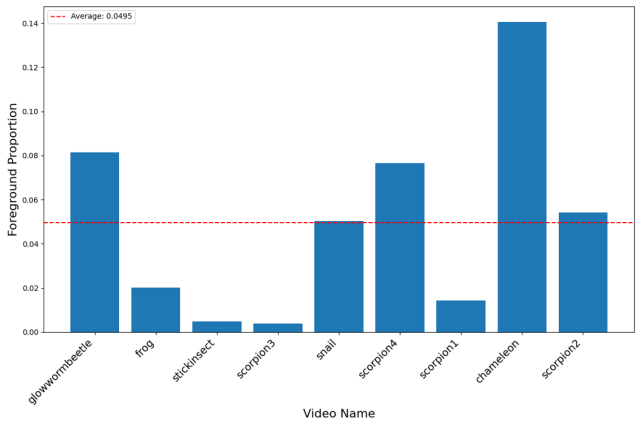

Figure 3: Foreground proportion statistics for CAD2016. The red dashed line denotes the average foreground proportion across videos.

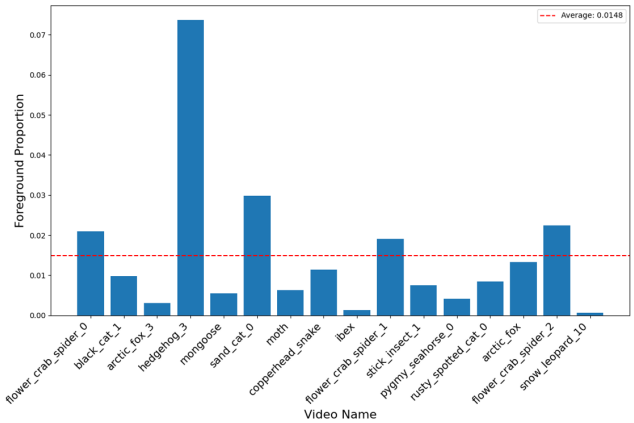

Figure 4: Foreground proportion statistics for MoCA. The red dashed line denotes the average foreground proportion across videos.

portion in CAD2016 is approximately 3.34 times larger than that in MoCA. This implies that objects in MoCA are generally smaller and more challenging to detect. The smaller object size and higher level of camouflage in MoCA naturally lead to a lower detection performance compared to CAD2016.

This observation provides a plausible explanation for the performance gap and highlights the importance of foreground scale in camouflaged object detection tasks.

7 Parameter and Performance Analysis

Our model consists of 98.1 million parameters, including 23.5 million trainable and 74.6 million non-trainable parameters during the second-stage training. In comparison to existing methods, our model is substantially more lightweight than FSPNet (274.24M) (Huang et al. 2023), TSP-SAM (727.1M total, with 89.75M tunable) (Hui et al. 2024), and SAM-PM (313.33M) (Meeran, Mantha et al. 2024), while remaining comparable in size to SLTNet (82.41M) (Cheng et al. 2022).

Despite having fewer or comparable parameters, our model consistently outperforms existing methods across all benchmarks. Notably, it surpasses the second-best method by over 15% in key evaluation metrics, demonstrating highly efficient parameter utilization. These results highlight the effectiveness of our architectural design in striking a favorable balance between model complexity and segmentation performance.

References

Cheng, X.; Xiong, H.; Fan, D.-P.; Zhong, Y.; Harandi, M.; Drummond, T.; and Ge, Z. 2022. Implicit motion handling for video camouflaged object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13864–13873.

Huang, Z.; Dai, H.; Xiang, T.-Z.; Wang, S.; Chen, H.-X.; Qin, J.; and Xiong, H. 2023. Feature shrinkage pyramid for camouflaged object detection with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 5557–5566.

Hui, W.; Zhu, Z.; Zheng, S.; and Zhao, Y. 2024. Endow SAM with Keen Eyes: Temporal-spatial Prompt Learning for Video Camouflaged Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19058–19067.

Meeran, M. N.; Mantha, B. P.; et al. 2024. SAM-PM: enhancing video camouflaged object detection using spatio-temporal attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1857–1866.