DMR: Decomposed Multi-Modality Representations for Frames and Events Fusion in Visual Reinforcement Learning

Supplementary Material

7. Released Code and Materials

The source code, benchmark, and demonstration materials of this paper are available at https://github.com/kyoran/DMR.

8. Environment Details

8.1. Scenarios

Our benchmark features two traffic scenarios: the HighBeam scenario, where an ego-vehicle experiences varying lighting conditions while encountering a cyclist, and the JayWalk scenario, where the ego-vehicle encounters both stationary and moving pedestrian obstacles intermittently. Moreover, the benchmark includes extreme weather conditions (Midnight and Hardrain) that can cause RGB camera failure or excessive noise in DVS cameras. Fig. 8 illustrates the scenarios under different weather conditions. To provide readers with a comprehensive view of the scenarios, the first and second columns show the Bird's Eye View (BEV) and Third Person View (TPV), respectively, under ideal lighting conditions. BEV and TPV are not used in the experiment. They are displayed solely for the purpose of facilitating a better understanding of this environment. The third and fourth columns display the original perception of the vehicles, which consist of RGB frames and DVS events.

Event-based cameras are typically sensitive to high-speed moving objects. When there are stationary or low-speed obstacles, event-based cameras fail to provide a sufficient number of events as essential perception signals. Low-speed or stationary obstacles are typically pedestrians, whose intention is hard to capture and predict, thus leading to vehicles being in a dilemma. Especially in low-light or block-the-view scenarios, pedestrians are difficult to be long-term tracked. We denote by 'JayWalk' (JW) the scenario that heightens the need for safety maneuvers in advance based on visual pedestrian signals captured in sparse time slices. Specifically, JW contains two jaywalking regions in 20-22m and 100-102m ahead of the ego-vehicle, which simulates the low-speed and different-size pedestrians. In this scenario, we set two pedestrians in each jaywalking region to directly cross the road from left to right and from right to left, respectively, regardless of the distance between the vehicle and themselves. The JW scenario under different weather conditions are illustrated in Fig. 8a.

Changes in illumination can cause significant fluctuations in the decision-making of visuomotor autonomous vehicles. Event-based cameras have a high dynamic range (120 dB), far exceeding that of frame-based cameras (60 dB). If approaching vehicles using their high beams, the intense oncoming headlights is challenging to the frame-based cameras, causing a flash of overexposure or underexposure. We denote by 'HighBeam' (HB) the scenario that simulates this phenomenon of sudden intensity changes. Specifically, HB contains three opposite vehicles which turn on high beams, and one slow-moving cyclist in front of the ego-vehicle. Frequent intensity changes cause significant differences in sequential perception frames, thus resulting in abnormal faults in the visuomotor decision. The HB scenario under different weather conditions are illustrated in Fig. 8b.

8.2. Reward Setting

The ego-vehicle's objective follows the common setup as in [21, 48], aiming to drive as far as possible without collisions within 500 steps. Therefore, the reward function is slightly modified to align with our challenging scenarios and designed as follows:

$$r_t = v_{ego}^{\top}\hat{u}_{lane}\cdot\Delta t - \lambda_c\cdot collision - \lambda_b\cdot brake - \lambda_s\cdot|steer| \tag{17}$$

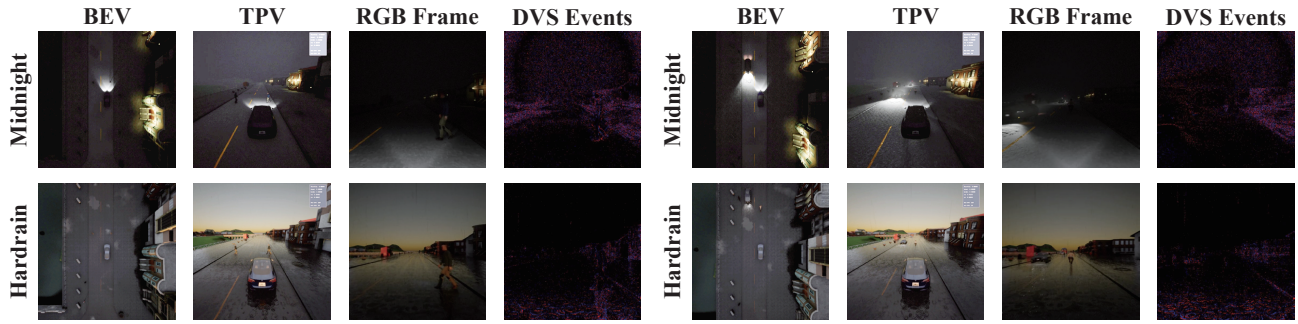
where v_{ego} is the ego-vehicle's velocity vector, $\hat{u}_{lane}$ is the unit vector of the lane corresponding to the location and heading of the ego-vehicle, and Δt is the simulation interval (i.e., $\frac{1}{20\text{Hertz}}$s). In this reward function, the first term is to encourage the ego-vehcile to drive along the lane as far as possible. The second, third, and fourth term penalize the collision, extra brake, and excessive steering, respectively. The trade-off coefficients $\lambda_c = 10^{-3}$, $\lambda_b = 0.1$, and $\lambda_s = 0.1$ are used to balance these penalties.

9. Additional Implementation Details

9.1. Network Architecture

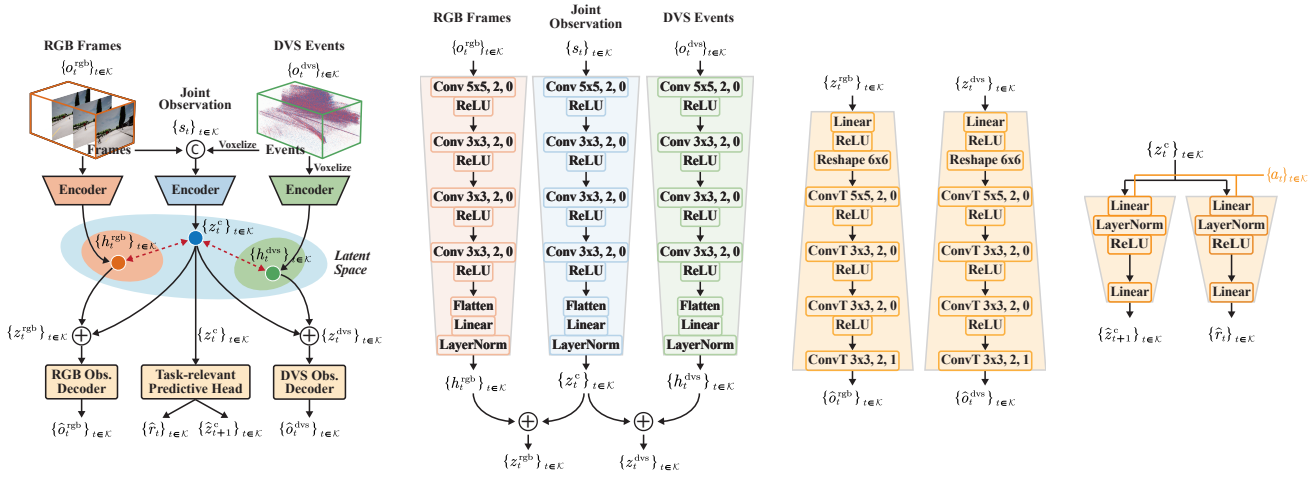
As shown in Fig. 9a, we annotate the symbols mentioned in the paper onto the framework diagram. These symbols can be classified into three distinct groups:

- Original perception observation input: RGB Frames $\{o_t^{\text{rgb}}\}_{t\in\mathcal{K}}$, DVS Events $\{o_t^{\text{dvs}}\}_{t\in\mathcal{K}}$, and the concatenation of RGB Frames and DVS Events$\{s_t\}_{t\in\mathcal{K}}$.
- Features in the latent space: RGB noise $\{h_t^{\text{rgb}}\}_{t\in\mathcal{K}}$, DVS noise $\{h_t^{\text{dvs}}\}_{t\in\mathcal{K}}$, co-feature $\{z_t^{\text{c}}\}_{t\in\mathcal{K}}$, RGB completeness feature $\{z_t^{\text{rgb}}\}_{t\in\mathcal{K}}$, and DVS completeness feature $\{z_t^{\text{dvs}}\}_{t\in\mathcal{K}}$.
- Predicted values from the decoders: Predicted state at



(a) JW scenario under Midnight and Hardrain weather conditions. (b) HB scenario under Midnight and Hardrain weather conditions.

Figure 8. Illustration of the proposed Carla autopilot benchmark. (The first and second columns show the Bird's Eye View (BEV) and Third Person View (TPV) under ideal lighting conditions, which are not utilized during the experiment. They are presented solely to enhance comprehension of this environment. The third and fourth columns are perceived observations.)



(a) Cooresponding symbols in DMR. (b) Three-branch encoders. (c) Reconstruction decoders. (d) Task-relevant head.

Figure 9. Detailed network architectures and corresponding symbols in Fig. 2.

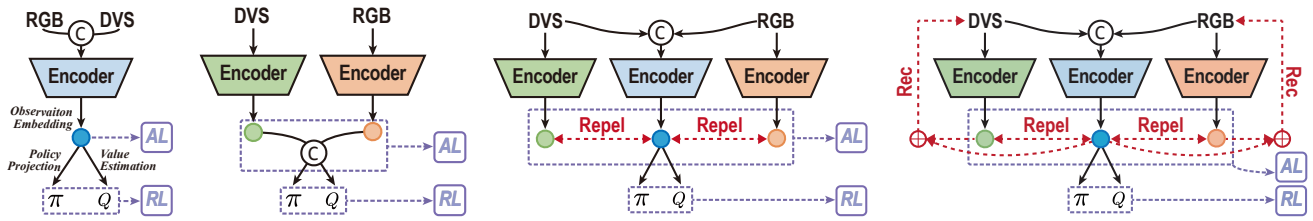
next time step $\{\hat{z}^{\mathrm{c}}_{t+1}\}_{t\in\mathcal{K}}$, and predicted reward at current time step $\{\hat{r}_t\}_{t\in\mathcal{K}}$.

Fig. 9b illustrates the encoder details in the three branches, where the numbers in "Conv" indicate the convolutional kernel size, stride, and padding, respectively. The encoders in these branches share the same structure, except for the input observation dimensions. The left and right branches output RGB noise $\{h^{\mathrm{rgb}}_i\}_{t\in\mathcal{K}}$ and DVS noise $\{h^{\mathrm{dvs}}_i\}_{t\in\mathcal{K}}$, respectively, while the intermediate branch outputs co-features $\{z^{\mathrm{c}}_i\}_{t\in\mathcal{K}}$. By combining these outputs (operations $\oplus$ in Fig. 9b, we can obtain the modality-specific completeness feature for RGB and DVS.

Fig. 9c showcases the decoder structure for the modality completeness constraint, where "ConvT" represents the transposed convolution module, and the numbers have the same meaning as in "Conv".

Fig. 9d displays the structure of the task-relevant predictive head. It is important to note that for these prediction heads, we utilized two tractable losses from DeepMDP [13] in this paper, which involve predicting the next-step co-features and the current-time reward. Therefore, the predictive head requires the action as input.

We further illustrate the details of the four models used in the ablation experiments (Sec. 5.3.1) in Fig. 10. The types of learning constraints in DMR can be summarized as Auxiliary Learning (AL) and Reinforcement Learning (RL). This helps readers understand the individual roles of each module we designed. In Fig. 10a, the RGB and DVS modalities are concatenated at the input level and pass through the encoder to obtain hidden features. These features are then constrained by the auxiliary tasks and reinforcement learning of DeepMDP. In Fig. 10b, two branches are used, with RGB and DVS inputs respectively. The hidden features from each branch are then constrained by DeepMDP. In Fig. 10c, a structure similar to Fig. 10a is added as the middle branch, and the features from all three branches are



(a) M1: 1 Branch (b) M2: 2 Branch (c) M3: +Repel (d) M4: +Rec

Figure 10. Summary of network architectures of the four ablation models in Tab. 2. (AL: Auxiliary Learning, RL: Reinforcement Learning)

Type	Attribute	Value	Description
RGB cameras	image size	128, 128	Image height and width in pixels.
	rendering hertz	20	RGB frames are captured at a frequency of 20 times per second.
	fov	60	Horizontal field of view in degrees.
	exposure bright	11, 20	Minimum and maximum brightness for automatic exposure.
	exposure speed	1, 3	Speeds at which the adaptation occurs when transitioning from a bright to a dark environment, and vice versa.
	blur amount	1	Strength of motion blur.
	blur max distortion	0.8	Max distortion caused by motion blur. Percentage of screen width.
	blur min object screen size	0.4	Percentage of the screen width that objects must possess in order to have motion blur. A lower value indicates a reduction in draw calls.
	lens flare intensity	0.2	Intensity for the lens flare post-processing effect.
	shutter speed	100	RGB camera shutter speed in seconds.
DVS cameras	image size	128, 128	Image height and width in pixels.
	rendering hertz	500	DVS events are captured at a frequency of 500 times per second.
	fov	60	Horizontal field of view in degrees.
	Q	0.2	The positive and negative thresholds linked to changes in the brightness of individual pixels.
	Q_σ	0.1	White noise standard deviation for positive and negative events.
	logarithmic	True	Whether to work in the logarithmic intensity scale.
Depth cameras	image size	128, 128	Image height and width in pixels.
	rendering hertz	20	Depth frames are captured at a frequency of 20 times per second.
	fov	60	Horizontal field of view in degrees.
	logarithmic	True	Logarithmic depth is employed for improved illustration of closer objects.
LiDAR sensors	image size	128, 128	LiDAR point cloud data is projected onto a top-down perspective view.
	rotation freq.	20	Rotation frequency.
	fov	10, -30, 360	Angles of the highest, lowest laser, and horizontal field of view in degrees.
	channels	64	Number of lasers.
	range	100	Maximum distance of all lasers in meters.
	points per sec.	250,000	Points generated per second by all lasers.

Table 5. The settings of multi-modality cameras.

constrained by Repel in the latent space. The features from the middle branch are also used for DeepMDP constraints. In Fig. 10d, additional constraints are applied to ensure the completeness of each modality's features. This means that the decomposed features can be reconstructed back to their respective modality information through the $\oplus$ operation.

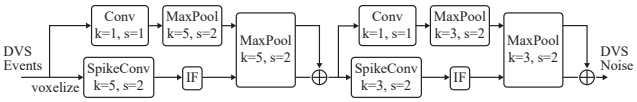


Figure 11. Alternative SNN-based DVS backbone structure.

Hyperparameter		Value
Input dimension	RGB frame	$2 \times 128 \times 128$
	DVS frame	$2 \times 128 \times 128$
	DVS voxel grid	$5 \times 128 \times 128$
	Depth frame	$1 \times 128 \times 128$
	LiDAR BEV	$1 \times 128 \times 128$
Action repeat		1
Frame skip		1
Frame stack		3
Initial sampling steps		1000
Maximum rollout steps		500
Number of training steps		110,000
Number of eval episodes		50
Replay buffer size		10,000
Initial α in SAC		0.01
Optimizer		Adam
Batch size		32
Actor learning rate		10^{-4}
Critic learning rate		10^{-4}
α learning rate		10^{-4}
Encoder learning rate		10^{-4}
Decoder learning rate		10^{-4}

Table 6. Full hyperparameter list used in DMR.

9.2. Multi-Modality Observation Details

Carla stands out as one of the few autonomous driving simulators that offers support for a rich set of scenarios with varying lighting and weather conditions. Carla also supports the generation of asynchronous events and RGB frames simultaneously. In Carla, DVS events are generated by utilizing the ESIM event camera plugin at a frequency of 500Hz. These events are subsequently transformed into asynchronous event streams. In order to create more authentic autonomous driving scenarios, we made minor modifications to the default parameters of the RGB and DVS cameras, including adjustments to exposure and motion blur. For precise details regarding the parameter settings, please consult Tab. 5. Notably, DVS events and RGB frames are synchronized and aligned at a rendering rate of 20 hertz. This allows us to precisely preprocess fixed-interval DVS events into a DVS voxel grid. Additionally, to assess the efficacy of our approach, we offer various modality combinations, such as RGB+Depth and RGB+LiDAR. The specifics of Depth and LiDAR are also provided in Tab. 5.

Methods		Runtime (ms)	Params (M)
TransFuser		11.716	3.971
EFNet		8.507	4.129
FPNet		8.411	4.192
RENet		9.127	5.850
DMR	Test	**5.232**	**1.464**
	Train	10.059	4.342

Table 7. Model statistics of multi-modality methods.

9.3. Network structure of alternative DVS backbone

As discussed in Sec. 3, we modify the backbone in the DVS branch by employing an advanced SNN structure using SpikingJelly [9], instead of the conventional 4-layer CNN structure. The detailed SNN structure is provided in Fig. 11. Due to the spike-based information flow (0 or 1) between layers, information loss typically accumulates as the network deepens. Therefore, we utilized cross-layer connections to compensate for this loss in our 2-layer SNN structure. Integrate-and-Fire (IF) neurons are used to generate spikes after spiking convolutional operations (SpikeConv). k and s represent kernel size and step of Spike-Conv. The output dimension remains consistent with the previous 4-layer CNN structure.

9.4. Hyperparameter Settings

A full list of hyperparameters in DMR is provided in Tab. 6.

10. Additional Experiments

10.1. Model statistics

In our framework, the RGB and DVS branches function as auxiliary components, augmenting the learning capacity of the intermediate branch. Consequently, during testing, these branches can be omitted, as illustrated by the dashed boxes in Fig. 2, rendering our architecture more lightweight. We present the runtime and model parameter statistics of multi-modality methods in Tab. 7. We run these methods on an Nvidia GeForce RTX 4080 for 100 iterations, reporting the average runtime in milliseconds. The results indicate that TransFuser uses the most runtime, despite having fewer parameters. This is mainly because TransFuser fully exploits the advantages of TransFormer architecture by adaptively learning self-attention scores [6], which greatly reduces parameters but significantly increases computational complexity. Other SOTA methods employ interactive modality fusion modules that introduce redundant parameters. In contrast, DMR does not explicitly employ the self-attention technique but still achieves reasonable CAM mappings with SOTA performance. In DMR, the RGB and DVS branches are auxiliary branches, which are

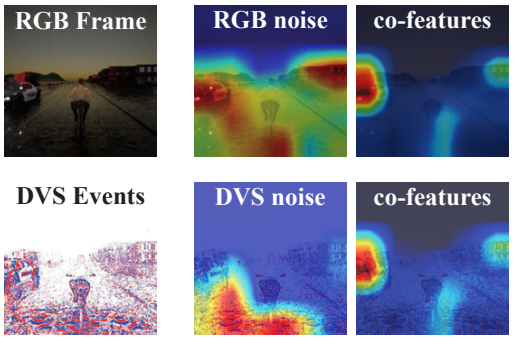


Figure 12. Additional analysis of CAM based on DMR.

Metrics	RGB	DVS	RGB+DVS
	SAC only		SAC+DMR
DT	36±10	90±36	**213±18**
ER	16±5	49±27	**180±28**

Table 8. Performance comparisons with different RL baselines under JW-Midnight.

Metrics	RGB	DVS	DRFuser	DMR
RMSE	1.31705	2.10519	1.62361	**1.00223**
MAE	0.46748	0.44107	0.99732	**0.37352**

Table 9. Performance comparisons on the DDD20 dataset.

used to enhance learning a better representation of the intermediate branch. As a result, these branches can be omitted in the test phase. In summary, DMR shows a significant reduction in both runtime and parameters when testing.

10.1.1 Additional CAM examples

From Fig. 12, it is evident that DVS noise captures the presence of the puddle and raindrops behind the bicycle, whereas RGB noise encompasses a broader range of areas, including the left-side vehicle, right-side buildings, the puddle beneath the bicycle, and parts of the road. Conversely, co-features have the ability to concentrate more on decision-relevant regions, such as the left-side vehicle, the cyclist directly ahead, and the buildings on the right. It is important to note that RGB noise contains certain objects that are relevant to the task at hand, such as the left-side vehicle. Due to its extensive coverage, RGB noise exhibits higher discriminability compared to co-features. Consequently, it does not significantly impact the extraction of co-features.

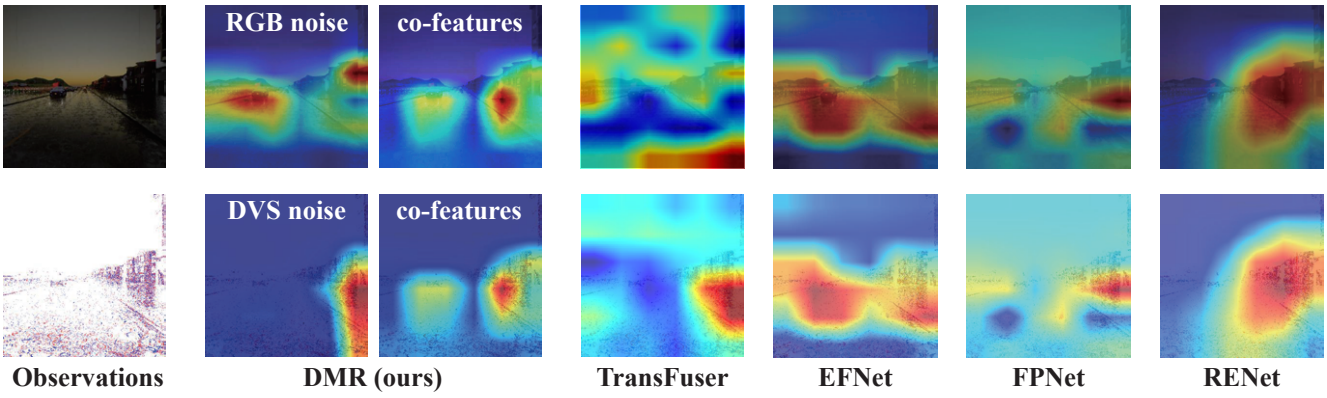
We visualize class activation maps (CAM) of several SOTA methods, including our own method DMR, in Fig. 13. Fig. 13 expands on Fig. 6 in the main text. It is clear that the current SOTA fusion methods encounter difficulties in distinguishing noise from task-relevant information via TD constraint. These SOTA methods strive to implicitly separate noise from valuable information and align features across various modalities. In contrast, our method DMR can explicitly separate different types of information and extract combined task-relevant features.

Fig. 14 illustrates a vehicle with high beam headlights approaching from a distance to near in the opposite lane at three different time instances, Time #1, #2, and #3. It is clear that the RGB noise emphasizes the vehicle's high-beam headlights and the buildings on the right, whereas the DVS noise focuses on the dense event region on the right. Both types of noise contain a substantial amount of task-irrelevant information, covering unnecessary broad areas. In contrast, the co-features generate a more focused area that is relevant for RL by excluding irrelevant regions. These areas precisely cover the vehicle on the opposite lane and the right roadside, which are crucial cues for driving policies. The variations in CAM closely mirror the alterations in the real scene throughout the entire process (i.e., Fig. 14a→Fig. 14b→Fig. 14c). When the vehicle approaches, the RGB noise broadens due to illumination changes, and the co-features focus more on the vehicle. In co-features, there is also a gradual increase in emphasis on the left roadside, and the CAM more uniformly covers the right roadside.

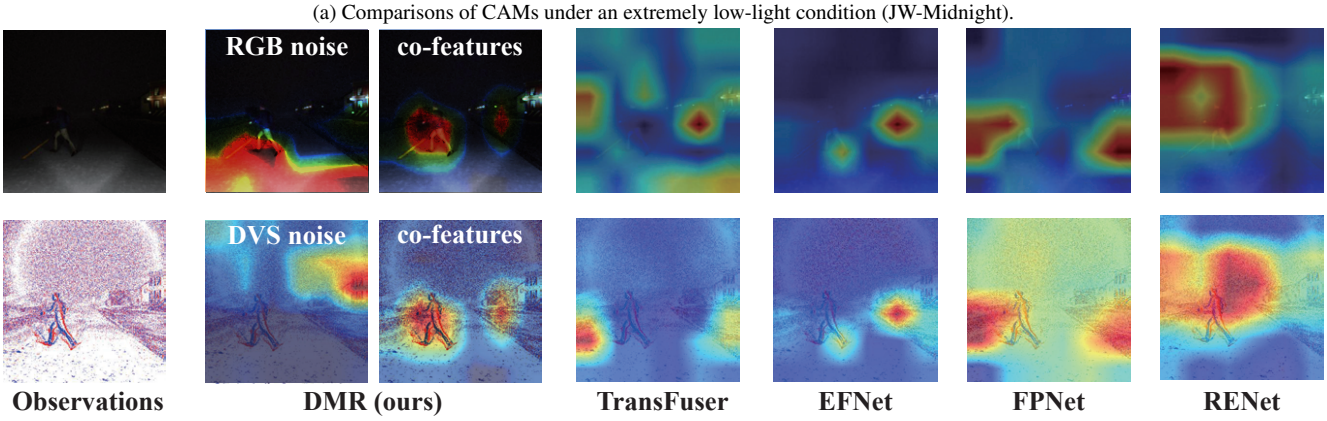
10.1.2 Additional ablations

In Tab. 8, we observe that our method can be treated as a plug, capable of enhancing the performance of single-modality baseline RL (SAC [14, 15]).

Besides, as mentioned in Sec. 5.1, we conduct experiments on a real-world offline dataset DDD [55], specifically aimed at validating the performance of offline learning. This part serves as supplementary validation and can be disregarded if deemed unnecessary. DDD dataset exhibits substantial variations in weather and illumination conditions. The selection and partitioning of the training and testing sets, as well as the preprocessing of DVS's binsize, can significantly influence the results. For instance, despite both methods in DDD [55] and DRFuser [56] employing a similar ResNet backbone network, a notable disparity in their outcomes is observed. Owing to the absence of specific details regarding the partitioning of training and testing sets or hyperparameters and preprocessing method by DRFuser, we replicated their findings using their source code and our own dataset partitioning as outlined in Tab. 9. We specifically select recordings (rec1500329526, rec1500394622, rec1501292394, rec1501292394, rec1498392691, and rec1498658145) and preprocess them using a binsize of 0.05 seconds. The training and testing ratio is 8:2. The training set comprises 9,824

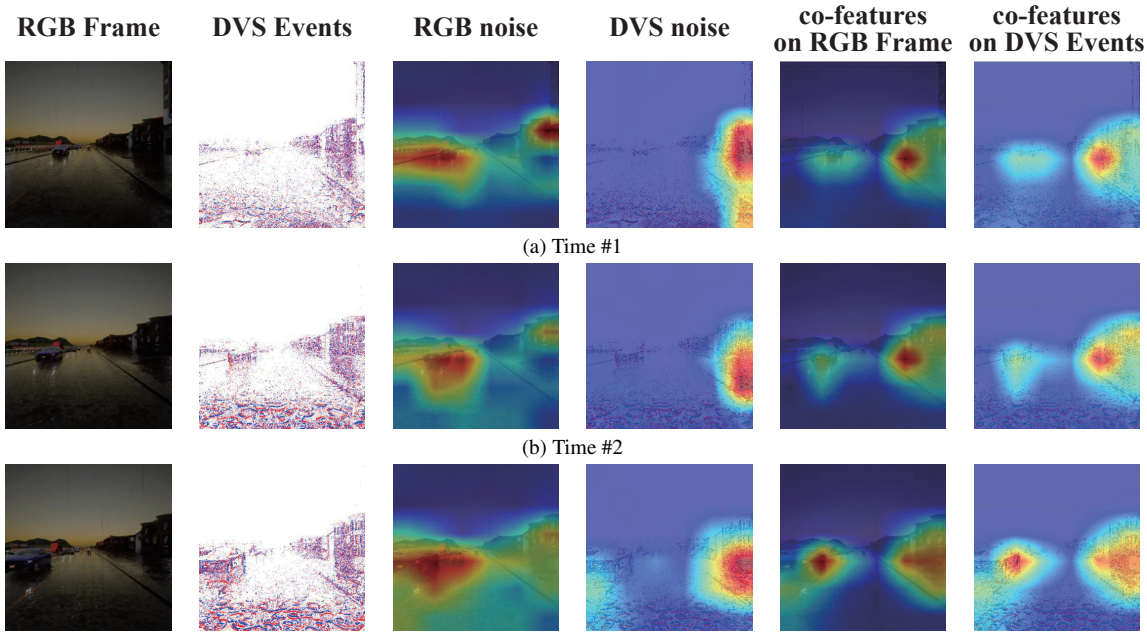


(a) Comparisons of CAMs under an extremely low-light condition (JW-Midnight).



(b) Comparisons of CAMs under a rapidly-changing illumination condition (HB-Hardrain).

Figure 13. CAMs under different illumination conditions.



(a) Time #1

(b) Time #2

(c) Time #3

Figure 14. A long-sequence demonstration of CAM under HB-Hardrain scenario.

image pairs, while the test set consists of 2,456 image pairs. We utilize root mean square error (RMSE) and mean absolute error (MAE) to assess performance. Notably, our results reveal an improvement compared to single-modality models and surpass the performance achieved by DRFuser.

11. Future Direction

DMR is a promising multi-modality visual fusion framework combing RGB frames and DVS events that has the potential to improve the efficiency of RL. However, in the context of existing autonomous driving scenarios, commonly used sensors such as LiDAR, Radar, and depth cameras are already well integrated into current autonomous driving systems. DMR falls short in effectively handling more than two modalities. Consequently, when dealing with more than two modalities, two urgent problems arise: 1) the extraction of complementary features between modalities to form combined task-relevant features, and 2) the elimination of noise within each modality. Furthermore, we have observed that visual reinforcement learning lacks pixel-level supervision, leading to low stability in decision-making performance. This issue becomes even more pronounced in multi-modality scenarios. Therefore, we propose that studying stability in multi-modality visual RL from the perspective of removing modality noise and extracting task-relevant information through collaboration is a feasible approach. Our future research will focus on exploring the generalization of modalities, the application capabilities, and the decision stability in order to fully harness the potential value of multi-modality visual information in RL.

References

[55] Yuhuang Hu, Jonathan Binas, Daniel Neil, Shih-Chii Liu, and Tobi Delbruck. Ddd20 end-to-end event camera driving dataset: Fusing frames and events with deep learning for improved steering prediction. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020. 5

[56] Farzeen Munir, Shoaib Azam, Kin-Choong Yow, Byung Geun Lee, and Moongu Jeon. Multimodal fusion for sensorimotor control in steering angle prediction. *Engineering Applications of Artificial Intelligence*, 126:107087, 2023.

5