Supplementary Material for "Simoun: Synergizing Interactive Motion-appearance Understanding for Vision-based Reinforcement Learning"

Yangru Huang[1], Peixi Peng[2,4] *, Yifan Zhao[1], Yunpeng Zhai[1], Haoran Xu[3,4], Yonghong Tian[1,2,4] *
[1]School of Computer Science, Peking University
[2]School of Electronic and Computer Engineering, Peking University
[3]School of Intelligent Systems Engineering, Sun Yat-sen University [4]Peng Cheng Laboratory
yrhuang@stu.pku.edu.cn, {pxpeng, zhaoyf, ypzhai, yhtian}@pku.edu.cn,xuhr9@mail2.sysu.edu.cn

The contents of the supplementary material are organized as follows:

- Sec. 1 provides additional implementation details and experiment setup, including network architecture, RL environment description, and hyperparameter settings.
- Sec. 2 demonstrates the effectiveness of *Simoun* in more complex and challenging environments.
- Sec. 3 conducts various further ablations, including the effects of different losses (Sec. 3.1), the influence of input frame stack size for motion modeling (Sec. 3.2) and the parameter β (Sec. 3.3), performance with alternative fusion schemes such as bilinear fusion (Sec. 3.4), impacts of different augmentation methods for the appearance path in *Simoun* (Sec. 3.5), comparison between the proposed structural interactive module with vanilla cross-attention mechanism (Sec. 3.6), comparison with other intrinsic rewards (Sec. 3.7) and other "dual-path" networks in video recognition (Sec. 3.8), results on more environments (Sec. 3.9) including DrawerWorld benchmark and Atari-100k tasks, and the analyse of motion-appearance embedding space (Sec. 3.10).
- Sec. 4 analyses the limitation of *Simoun* in its current form and suggests possible directions for future studies.

1. Additional Implementation Details

Network Architecture Fig. A1 gives the detailed network architecture used in our approach. For the motion/appearance paths, the **observation encoder** is implemented with four convolutional (Conv) layers. Each Conv layer has 3×3 kernel size with 32 output channels, followed by a ReLU activation. For DMControl, all layers have a stride of 1 except for the first Conv layer, which has a stride of 2. However, for CARLA, the stride of all layers is set to 2. After the last Conv layer, a fully connected (FC) layer with layer normalization (LN) is used to reduce the dimension of features to 50. The weight matrices of the Conv and FC layers are initialized with orthogonal initialization [9] and the biases are set to zero. For DMControl, we further apply a Tanh layer after the LN layer following previous methods [16]. In addition, the input sizes of the two environments are also different. For DMControl, the input is rendered at 100×100 and cropped to 84×84. For Carla, the input is the horizontally concatenated images from three cameras on the roof of vehicles, which has a size of 84×252.

Both the **action-conditioned transition model** and the **reward prediction model** have a sequence of [FC, LN, ReLU, FC] layers. Differently, the former outputs the predicted latent vector of the next observation, and the latter outputs a single value of the predicted reward of the next observation. For the **critic network**, clipped double Q-learning [12, 3] is adopted for stability. Each critic network includes a 3-layer MLP with ReLU activations. The target critic network is updated by the critic network using momentum updating. For the **actor network**, a 3-layer MLP is employed similar to the critic network but outputs mean and covariance of the diagonal Gaussian representing the policy. We set the hidden dimension of the actor and critic MLPs to 1024.

Environment Details We use two common RL environments (DMControl and CARLA) in our experiments. Deepmind control (DMControl) suite is a physics-based simulation of Reinforcement Learning environments, based on MuJoCo physics [11]. Our setup on DMControl is similar to [16]. Six tasks are used for evaluation as shown in Fig. A2, including cartpole-swingup, reacher-easy, cheetah-run, walker-walk, finger-spin, and ball-in-cup-catch. CARLA is a simulator for autonomous driving

*Corresponding author.

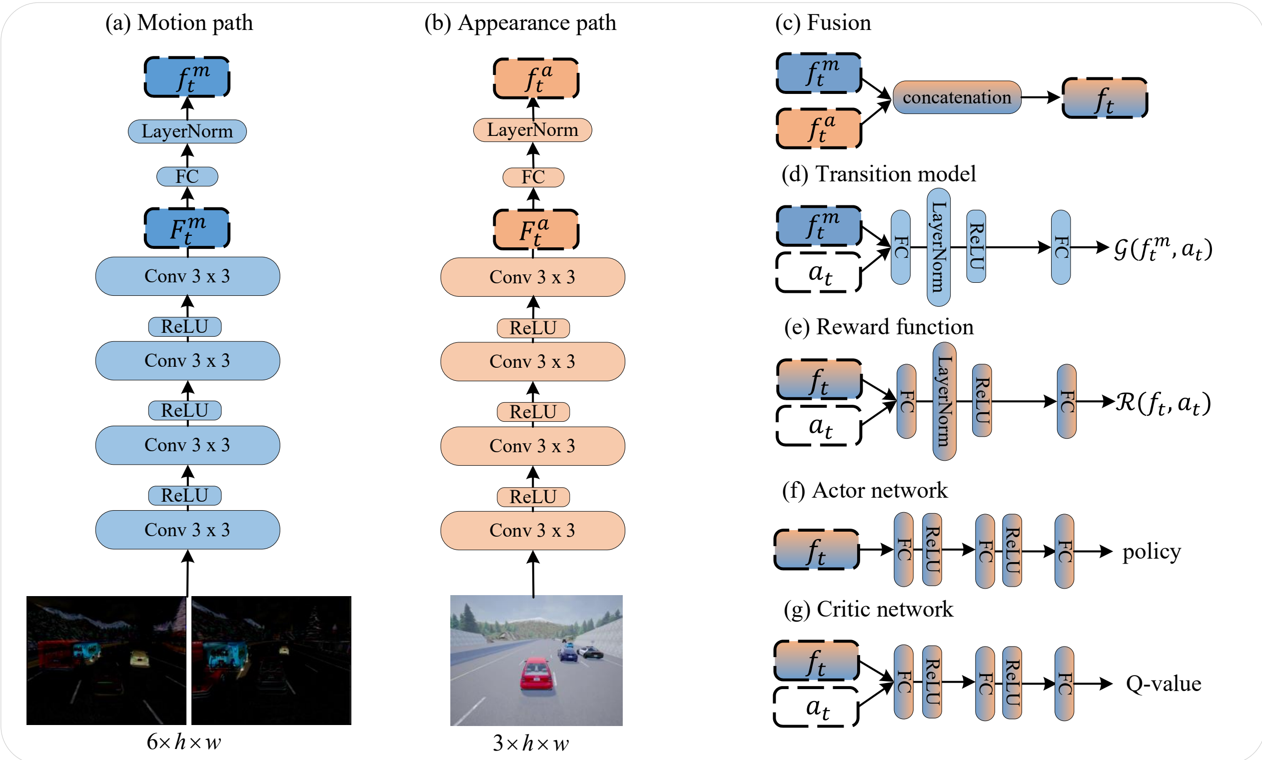


Figure A1. The detailed network architectures.

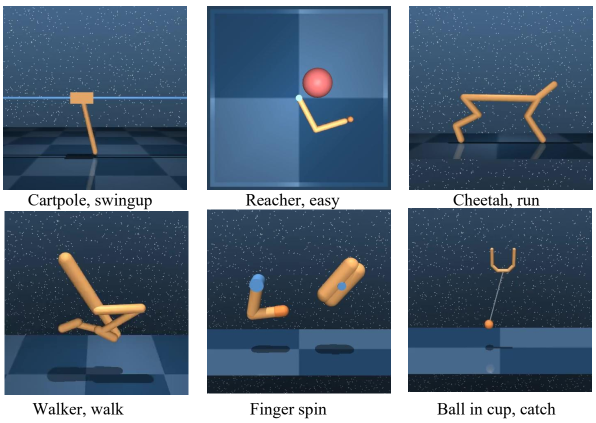


Figure A2. The adopted six tasks in DMControl.

(a) The third-person view (b) The first-person observation for agents

Figure A3. The environment views of CARLA: (a) the third-person view of the environment, and (b) the first-person view of the input observations for agents.

research. Following DBC [17], we adopt CARLA with version 0.9.6. The goal of the agent in CARLA is to drive on Highway 8 located in Town 4 and cover the maximum possible distance in 1000 time steps while avoiding collisions with 20 other moving vehicles. This will be accomplished under clear noon weather conditions. As shown in Fig. A3 (b), the RGB observation with 84×252 is obtained from three attached cameras. CARLA's output action is a 2D vector that includes both thrust (where braking is represented as negative thrust) and steering values. The reward function is designed following [17]:

$$r_t = v_{ego}^{\top} \hat{u}_{highway} \cdot \Delta t - \lambda_c \cdot collision - \lambda_s \cdot |steer| \quad (1)$$

where v_{ego} is the velocity vector of the ego vehicle, $\hat{u}_{highway}$ is the highway's unit vector, and Δt is the simulation time discretization. The first term aims to encourage progression along the highway as long as possible. Meanwhile, the last two terms penalize collision and excessive steering. $\lambda_c = 10^{-4}$ and $\lambda_s = 1$ are trade-off coefficients. The action repeat is set to 4 for all agents.

Hyperparameter Settings At the beginning of the training process, the agent acquires 1000 transitions for DMControl and 100 transitions for CARLA while following a random policy. Afterward, it gathers additional transitions based on the learned policy. These collected transitions are saved in a replay buffer, and the training batch size is configured to 128. Other detailed hyperparameters such as learning rate and update frequency are listed in Table A2. During the evaluation, the agents utilize the mean policy action obtained from the trained actor network. Each agent is evaluated once every 20,000 training environment steps, and the evaluation process is repeated for 20 episodes.

The components of prior state-of-the-art algorithms All methods in Table 1&2 in the main paper use SAC as the base. The components are summarized in below table. We clarify that ❶-❹ aren't merely auxiliary but strategically selected as a part of Simoun to make the best of Motion-Appearance modeling for better performance. We have also ablated ❶-❹ in Fig. A5.

Table A1. Used Components by Simoun: ❶ $\mathcal{L}_{tran}$, ❷ $\mathcal{L}_{con}$, ❸ $\mathcal{L}_{re}$, ❹ Data Aug for $\mathcal{L}_{con}$. Unused Components by Simoun: ①predict the future latent states multiple steps, ②reconstruction loss, ③backward dynamics model, ④Cycle consistency loss, ⑤prioritizing the experience replay, selecting the augmented inputs,

SAC None	Dreamer ❶❸②	CURL ❷❹	DrQ ❹	SVEA ❹	PlayVirtual ❶①③④
MLR ❶❷❹	CCLF ❶❷❹⑤	Flare None	DeepMDP ❶ ❸	SPR ❶❷❸❹①	Simoun ❶❷❸❹

2. Performance in Challenging Environments

Due to the complex nature of visual observations, it is crucial for vision-based RL agents to fit diverse complicated scenes. To verify such an ability of *Simoun*, we compare it with the latent flow model under two complex training environment settings on DMControl and CARLA. For DMControl, as shown in Fig. A4 (a), the original environment background is static and clean. To increase the task difficulty, we choose the video-hard mode demonstrated in Fig. A4 (b) to train the agents, in which the background is replaced with dynamic natural videos. Similarly, we use a miscellaneous weather setting on CARLA to train the agents, as shown in Fig. A4 (d). The CARLA environment has been modified to include various types of distractors such as sun, rain, shadows, clouds, and more, which are representative of different weather conditions. These weather conditions are dynamically assigned for each episode and may even change during the course of the same episode. Table A3 shows the performance under these challenging environments. It is evident that the performance of *Simoun* drops less compared with the latent flow model. Although latent flow explicitly models motion information by computing the difference of latent spatial vectors, exposure to complex environments with drastically changed appearance results in severely decreased performance. In contrast, *Simoun* is less affected. The results indicate that *Simoun* can handle complex environments due to the complementarity of the motion and appearance paths, which helps to ignore unwanted distracting elements.

Table A2. Hyperparameters used in DMControl and CARLA.

Hyperparameter	DMControl	CARLA
Init steps	1000	100
Input dimension	$3 \times 84 \times 84$	$3 \times 84 \times 252$
Action repeat	2(finger) 8(cartpole) 2(otherwise)	4
Discount factor λ	0.99	0.99
Number of training steps	500,000	100000
Replay buffer size	500,000	100000
Optimizer	Adam	Adam
Actor learning rate	1e-3	1e-3
Critic learning rate	1e-3	1e-3
log α Learning rate in SAC	1e-4	1e-3
Batch size	128	128
Actor update frequence	2	2
Critic target update frequence	2	2
Data augmentation	Random Conv	Random Conv
Intrinsic reward temperature weight C	0.2	0.2
Intrinsic reward decay weight λ	2e-5	2e-5

Table A3. Performance on more complex environments. Δ means the degree of performance degradation, and $\downarrow$ denotes that smaller values are better results. The results of cheetah-run task in DMControl benchmark are reported.

DMControl	Original	Video hard	$\Delta\downarrow$
Simoun	693 ±53	422 ±46	271
Latent flow	610 ±38	232 ±35	378
CARLA	Clear noon	Variable weather	$\Delta\downarrow$
Simoun	281 ±30	213 ±24	68
Latent flow	182 ±11	89 ±13	93

3. Further Ablations

3.1. Effectiveness of Different Losses

As introduced in the main manuscript, several targeted loss terms are deliberately designed in *Simoun* as:

$$\mathcal{L} = \underbrace{\mathcal{L}_{tran}}_{Motion} + \underbrace{\mathcal{L}_{con}}_{Appearance} + \underbrace{\mathcal{L}_{re}}_{State} + \underbrace{\mathcal{L}_{Q} + \mathcal{L}_{\pi}}_{RL}, \tag{2}$$

where the first three terms are responsible for learning motion, appearance, and reward-related features, respectively. In order to verify the effectiveness of these three loss terms, we conduct experiments on CARLA by removing one loss term and keep the other two untouched. Results are shown in Fig. A5. Several observations can be made: 1) The model without $\mathcal{L}_{tran}$ gets obvious performance degradation. This indicates the importance of the underlying environment dynamics for motion modeling. Without the explicit transition loss, the motion-related information can not be learned

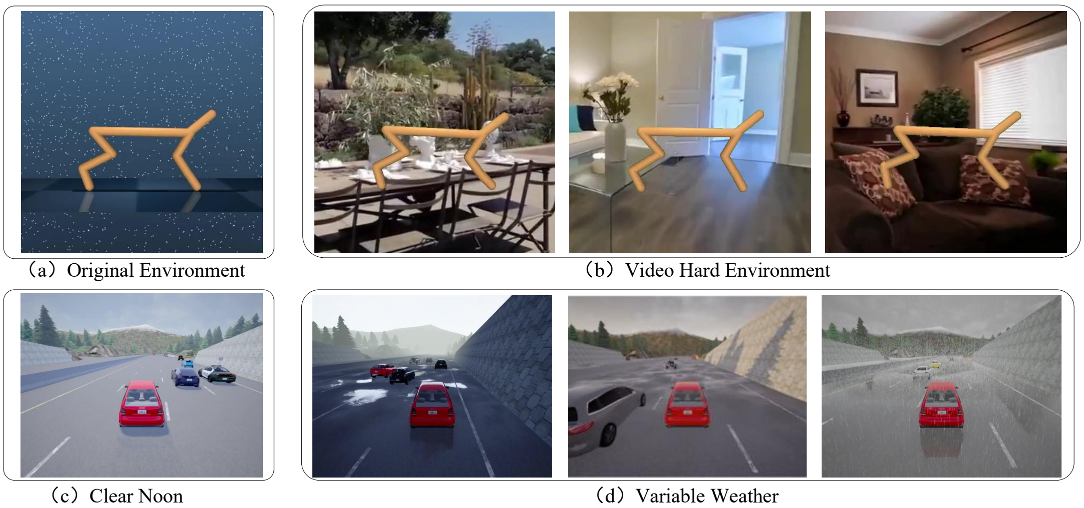


Figure A4. Demonstration of the complex environments: (a) original DMControl environment, (b) DMControl with video-hard mode, (c) original CARLA environment with clean noon weather, and (d) modified CARLA environment with a miscellaneous weather setting.

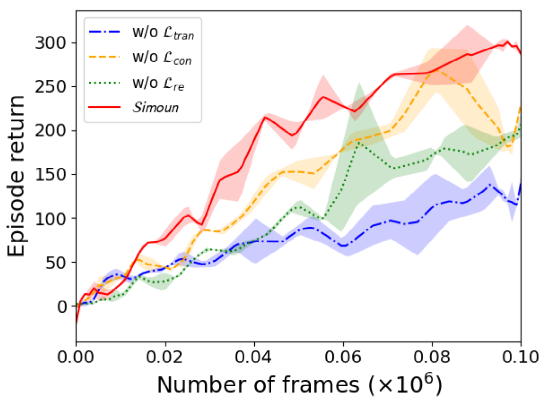


Figure A5. Performance on CARLA with different loss terms.

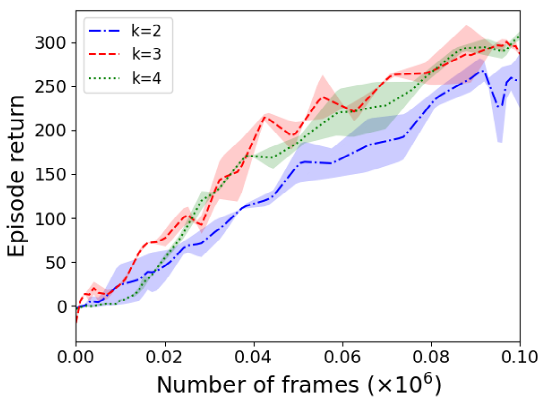


Figure A6. Performance with different input frame stack sizes.

well. 2) The contrast loss $\mathcal{L}_{con}$ is constructed to force different perspectives of the same observation to approach each other in latent space so as to learn good appearance representation. The model without $\mathcal{L}_{con}$ also bears performance degradation. However, the performance drop is not as severe as removing $\mathcal{L}_{tran}$. Such a phenomenon indicates that motion modeling constraint is more essential than appearance modeling constraint, while both can improve performance if introduced. 3) $\mathcal{L}_{re}$ contributes positively to the performance. The reason is that $\mathcal{L}_{re}$ forces the model to pay more attention to reward-related information. Conversely, removing $\mathcal{L}_{re}$ while keeping the rest loss terms would make the model excessively describe the observation with detrimental features that are not relevant to the reward.

3.2. Influence of Input Frame Stack Sizes

To model motion-related information, a stack of $k = 3$ neighboring frames are used to get the residuals. To investigate the impact of different stack size k, we set k to $2, 3, 4$, and test the performance of *Simoun* on CARLA. The results are shown in Fig. A6. As we can see, increasing the number of stacked frames to 4 does not significantly impact the driving task, but reducing the number to 2 leads to substantial performance degradation. The results indicate that using only 2 frames is insufficient for adequately modeling the motion information.

3.3. Influence of β

We report results on CARLA using fixed β and give the learned β values along training in the Table A4. It is evident from the results that adaptive learning of β yields the maximum accumulated reward. As the number of training

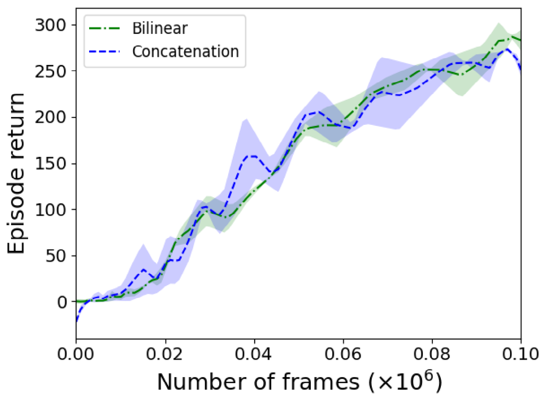


Figure A7. Performance of *Simoun* with different fusion schemes.

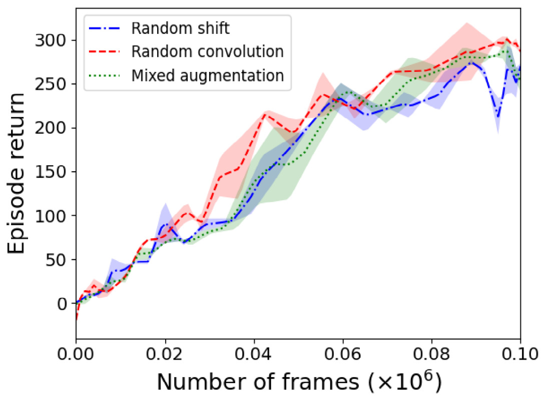


Figure A8. Performance of different augmentation methods used for calculating the contrastive loss.

steps progresses, β initially experiences a significant surge, peaks at the 30,000 step, and exhibits a minor decline at the end of training.

Table A4. The ablation study of β

β	0.25	0.5	0.75	learned β
Episode return	274	264	239	281
Training step	0	10000	30000	100000
Learned β value	0.00	0.15	0.23	0.21

3.4. Performance with Alternative Fusion Schemes

After acquiring the structure-enhanced motion and appearance features f_t^m and f_t^a, they are fused by a fusion function. To test the effect of different fusion schemes, we ablate two alternatives:

Concatenation fusion: This is the scheme used in the main manuscript for model simplicity. Concatenation fusion stacks f_t^m and f_t^a across the feature channels. The process of concatenation does not establish a clear correspondence between f_t^m and f_t^a, as it delegates the task of modeling their relationship to the subsequent layers.

Bilinear fusion: Bilinear fusion computes a matrix outer product of the two features. Specifically, given f_t^m and f_t^a, bilinear fusion is calculated as:

$$\mathbf{f}_t = \mathbf{f}_t^m \mathbf{W} \mathbf{f}_t^a + b, \tag{3}$$

where b is an additive bias and $\mathbf{W}$ is a learnable weight matrix. By employing this fusion scheme, the model becomes capable of capturing intricate relationships between the two features, leading to the creation of a more powerful feature.

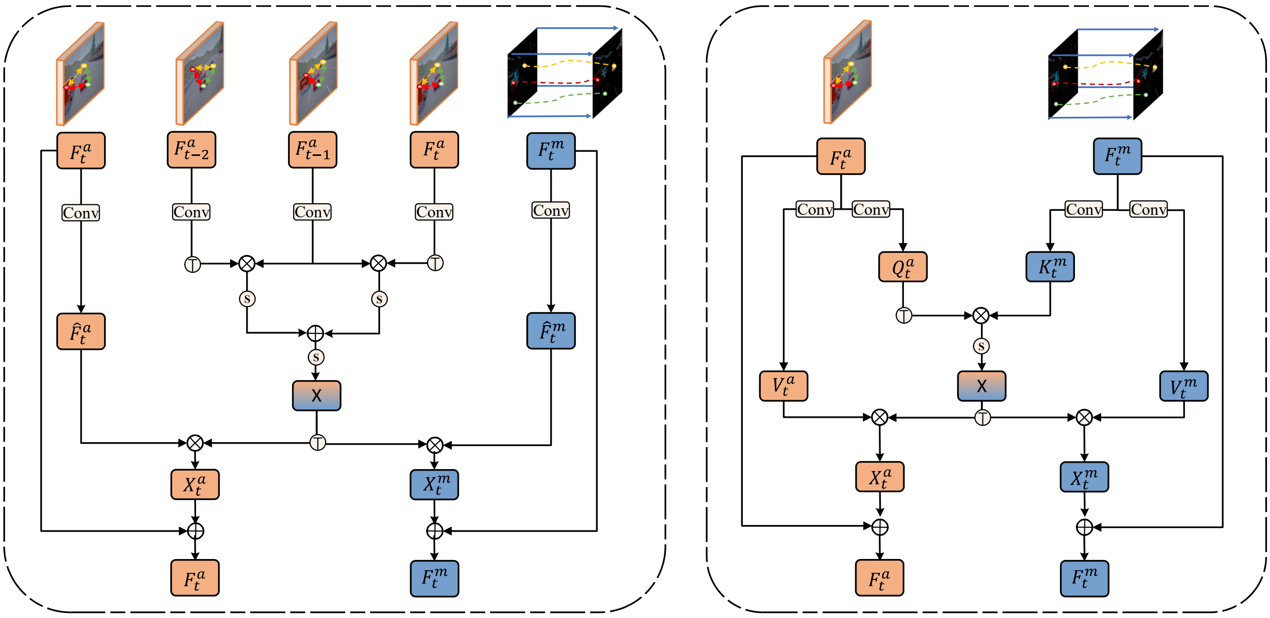
Experiment results of the two fusion schemes on CARLA are shown in Fig. A7. As we can see, both schemes are able to achieve comparable performance. However, bilinear fusion behaves more stable with less performance variation. Meanwhile, the episode returns at the last steps are slightly better than concatenation. The results indicate that *Simoun* can further benefit from complex fusion schemes of motion and appearance features. This fact provides *Simoun* with extra design space for deciding proper fusion methods. However, choosing the best fusion method is beyond the scope of this manuscript, and therefore we only explore the selected two schemes described above.

3.5. Impact of Different Augmentation Methods

In *Simoun* we adopt an unsupervised contrastive loss for appearance modeling. The contrastive loss works by constructing positive sample pairs using data augmentations. Here we investigate the effectiveness of various data augmentation methods, including a weak augmentation random shift [16], a strong augmentation random convolution [5], and a mixed augmentation. Random shift pads on each side of the input image and randomly crop the image to its original size. Random convolution augments the image color by passing the input observation through a random convolutional layer. Mixed augmentation adopts both random shift and random convolution by selecting only one of them at each time. In previous studies SECANT [1] and SVEA [4], the researchers show that strong augmentation is inherently non-deterministic, resulting in lower performance and training divergence. In contrast, as shown in Fig. A8, *Simoun* with strong augmentation can still achieve improved performance. The results indicate that the dual-path architecture of *Simoun* is able to alleviate the non-deterministic problem, where the motion path is not affected by augmentation and therefore can stabilize the state representation.

3.6. Comparison with Other Interactive Modules

The structural interactive module of *Simoun* leverages the complementarity between the two network paths by extracting latent motion-appearance structures. The key elements of the structural interactive module are the inter-frame attention and spatial gating operation for motion and appearance structure extraction, respectively. To verify the



(a) Structural Interactive Module (b) Vanilla Cross Attention

Figure A9. Architectures of two interactive modules: (a) the proposed structural interactive module and (b) vanilla cross-attention.

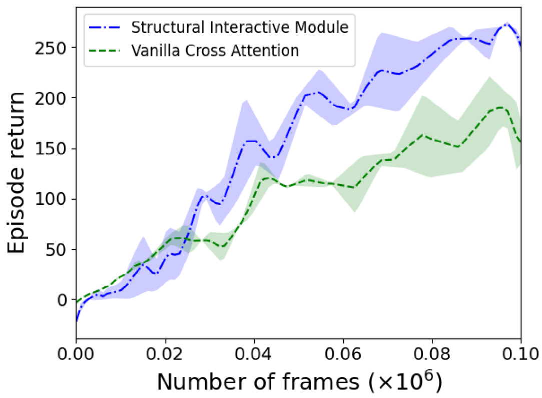


Figure A10. Comparison of different interactive modules.

effectiveness of these two elements, we compare the structural interactive module with a vanilla cross-attention module commonly used for cross-path interaction.

Specifically, as shown in Fig. A9 (b), the inputs of the vanilla cross-attention module are motion feature map $\mathbf{F}_t^m$ and appearance feature map $\mathbf{F}_t^a$. A cross-path attention map $\mathbf{X}$ can be obtained as:

$$\mathbf{X} = \sigma(\mathbf{Q}_t^{a\top} \mathbf{K}_t^m), \tag{4}$$

where $\mathbf{Q}_t^a$ denotes a new feature map generated by feeding $\mathbf{F}_t^a$ to a convolution layer and $\mathbf{K}_t^m$ is obtained from $\mathbf{F}_t^m$ similarly. σ denotes the Softmax function. Then $\mathbf{X}$ is used to modulate and update features from both paths:

$$\mathbf{F}_t^m = \mathbf{F}_t^m + \mathbf{V}_t^m \cdot \mathbf{X}, \mathbf{F}_t^a = \mathbf{F}_t^a + \mathbf{V}_t^a \cdot \mathbf{X}, \tag{5}$$

where $\mathbf{V}_t^m$ and $\mathbf{V}_t^a$ are features obtained by passing $\mathbf{F}_t^m$ and $\mathbf{F}_t^a$ to another two convolutional layers, respectively.

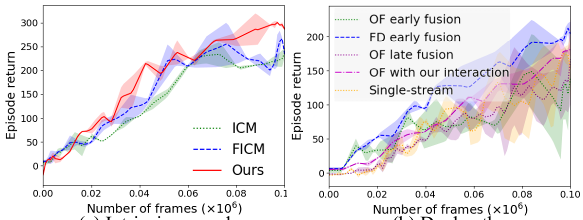
Experiment results of the two modules on CARLA are shown in Fig. A10. As can be seen, the performance of cross-attention is rather poor, which indicates that the motion features and appearance features learned by the two paths should not be aligned directly. In contrast, the two kinds of features in the structural interactive module are excavated in a complementary manner by focusing on their own characteristics, which greatly improves feature quality.

3.7. Comparison with Other Intrinsic Rewards

Intrinsic reward mechanisms play a pivotal role in enhancing the learning capabilities of agents, especially in challenging environments. When compared with previous intrinsic rewards methods ICM [8] and FICM [15] in a direct and fair manner, our approach has obvious advantages as shown in Fig. A11(a). This indicates that our curiosity module is able to foster a more effective learning process by providing incentives for the agent to investigate states characterized by discrepancies between motion and appearance.

3.8. Compare to "Dual-path" Networks

Here we also compare our method with previous "Dual-path" Networks in video recognition with two-stream fu-



(a) Intrinsic rewards (b) Dual paths

Figure A11. Comparison with other intrinsic rewards.

sion strategies (late fusion or early fusion with optical stream [10, 2]), and single-stream architecture variants [14]. These video-based methods [10, 2, 14] focus on long video clips while overlooking what RL emphasizes: frame-level dynamics caused by actions at each step (*i.e.*, "fine-grained"). *e.g.*, frame-shift in [14] obviously disrupts RL dynamics. We use frame difference (FD) instead of optical flows (OF) due to OF's computational cost and potential inaccuracies. As shown in Fig. A11(b), all other methods with OF in Fig. A11(b) still trail our method with FD. For fusion strategies, we model dynamics from both appearance and motion paths. Since it is essential for an agent to know "what is moving" and "how they moved", the motion-appearance interaction yields a more comprehensive grasp of environment dynamics by digging dynamic information from the appearance path (feature-level cross-frame variation) and motion path (pixel-level cross-frame variation).

3.9. Results on More Environments

In our study, outcomes from both the DrawerWorld benchmark [13] and Atari-100k tasks [7] are presented in Table A5 and Table A6, respectively. The DrawerWorld benchmark, adapted from Meta World, focuses on manipulation tasks in grid settings and assesses performance in textured environments. Here we specifically utilize fabric textures for testing. Our chosen metric for evaluation is the success rate, quantified as the ratio of successful endeavors to open or close a drawer over a total of 100 attempts. As evident from Table A5, our methodology, Simoun, consistently outperforms the SPR baseline in the DrawerWorld benchmark. Regarding the Atari framework, it is characterized by its pixelated 2D visuals and a set of discrete player actions. The results in Table A6 underscore the efficacy of our approach in the Breakout and Hero tasks.

Table A5. The results on DrawerWorld robotic manipulation tasks.

	DrawerOpen		DrawerClose	
Success%	SVEA	Ours	SVEA	Ours
Grid (train)	92	96	93	95
Fabric (test)	61	74	24	43

Table A6. The results on Atari-100k tasks.

Atari100k	SPR	Simoun
Breakout	17.1	20.7
Hero	7019.2	6973.4

3.10. The Analyse of Embedding Space

To assess the alterations in the embedding space of motion and appearance paths, we employ two prevalent metrics: mutual information and pearson correlation coefficient. Firstly, we find that motion-appearance paths are interactive and they do have a **closer** (without explicitly constraining) embedding space with increased correlation after training, see Table A7. Secondly, the notion that the difference in image space closely aligns with the difference in feature space is an interesting constraint that leads to a robust flat embedding space which approximately ensures Lipschitz continuity [6]. Simoun happens to implicitly run towards this, which provides another view of why it works.

Table A7. The discrepancy of $|\mathbf{f}_t^m|$ and $|\mathbf{f}_{t-1}^a - \mathbf{f}_{t-2}^a| + |\mathbf{f}_t^a - \mathbf{f}_{t-1}^a|$ embedding space.

Measurements	Initial	Trained
Mutual Infomation	0.0059	0.1700
Pearson Correlation Coefficient	0.0081	0.0558

4. Limitation and Future Direction

Simoun is a promising framework for vision-based reinforcement learning. However, one limitation of *Simoun* is that the motion and appearance features are both learned from the environment itself. In some extreme cases, learning solely from the environment might be difficult due to significantly varying motion and appearance patterns. In this case, leveraging the rich knowledge in other data sources becomes critical. Therefore, a possible future direction is to incorporate *Simoun* with external high-capacity vision models pre-trained on large datasets. By leveraging the abundant motion and appearance features extracted from high-performance models (*e.g.*, CNNs and vision transformers for large-scale video understanding), *Simoun* has the potential to achieve even better performance.

References

[1] Linxi Fan, Guanzhi Wang, De-An Huang, Zhiding Yu, Li Fei-Fei, Yuke Zhu, and Animashree Anandkumar. Secant: Self-expert cloning for zero-shot generalization of visual policies. In *International Conference on Machine Learning*, 2021. 6

[2] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Computer Vision and Pattern Recognition*, 2016. 8

[3] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, 2018. 2

[4] Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. In *Advances in Neural Information Processing Systems*, 2021. 6

[5] Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. In *International Conference on Machine Learning*, 2020. 6

[6] Haozhe Liu, Wentian Zhang, Bing Li, Haoqian Wu, Nanjun He, Yawen Huang, Yuexiang Li, Bernard Ghanem, and Yefeng Zheng. Improving gan training via feature space shrinkage. *arXiv preprint arXiv:2303.01559*, 2023. 8

[7] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 8

[8] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, 2017. 7

[9] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013. 2

[10] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, 2014. 8

[11] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, 2012. 2

[12] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2016. 2

[13] Xudong Wang, Long Lian, and Stella X Yu. Unsupervised visual attention and invariance for reinforcement learning. In *Computer Vision and Pattern Recognition*, 2021. 8

[14] Wangmeng Xiang, Chao Li, Biao Wang, Xihan Wei, Xian-Sheng Hua, and Lei Zhang. Spatiotemporal self-attention modeling with temporal patch shift for action recognition. In *European Conference on Computer Vision*, 2022. 8

[15] Hsuan-Kung Yang, Po-Han Chiang, Min-Fong Hong, and Chun-Yi Lee. Flow-based intrinsic curiosity module. *arXiv preprint arXiv:1905.10071*, 2019. 7

[16] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2020. 2, 6

[17] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020. 3