MOSO: Decomposing MOtion, Scene and Object for Video Prediction

Mingzhen Sun\(^1\),\(^2\) Weining Wang\(^1\) Xinxin Zhu\(^1\) Jing Liu\(^1\),\(^2\),\(^\ast\)

\(^1\)The Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences (CASIA)
\(^2\)School of Artificial Intelligence, University of Chinese Academy of Sciences (UCAS)

sunmingzhen2020@ia.ac.cn \{weining.wang, xinxin.zhu, jliu\}@nlpr.ia.ac.cn

Abstract

Motion, scene and object are three primary visual components of a video. In particular, objects represent the foreground, scenes represent the background, and motion traces their dynamics. Based on this insight, we propose a two-stage MOtion, Scene and Object decomposition framework (MOSO)\(^1\) for video prediction, consisting of MOSO-VQVAE and MOSO-Transformer. In the first stage, MOSO-VQVAE decomposes a previous video clip into the motion, scene and object components, and represents them as distinct groups of discrete tokens. Then, in the second stage, MOSO-Transformer predicts the object and scene tokens of the subsequent video clip based on the previous tokens and adds dynamic motion at the token level to the generated object and scene tokens. Our framework can be easily extended to unconditional video generation and video frame interpolation tasks. Experimental results demonstrate that our method achieves new state-of-the-art performance on five challenging benchmarks for video prediction and unconditional video generation: BAIR, RoboNet, KTH, KITTI and UCF101. In addition, MOSO can produce realistic videos by combining objects and scenes from different videos.

1. Introduction

Video prediction aims to generate future video frames based on a past video without any additional annotations [6, 18], which is important for video perception systems, such as autonomous driving [25], robotic navigation [16] and decision making in daily life [5], etc. Considering that video is a spatio-temporal record of moving objects, an ideal solution of video prediction should depict visual content in the spatial domain accurately and predict motions in the temporal domain reasonably. However, easily distorted object identities and infinite possibilities of motion trajectories make video prediction a challenging task.

Recently, several works [15, 44] propose to decompose video signals into content and motion, with content encoding the static parts, i.e., scene and object identities, and motion encoding the dynamic parts, i.e., visual changes. This decomposition allows two specific encoders to be developed, one for storing static content signals and the other for simulating dynamic motion signals. However, these methods do not distinguish between foreground objects and background scenes, which usually have distinct motion patterns. Motions of scenes can be caused by camera movements or environment changes, e.g., a breeze, whereas motions of objects such as jogging are always more local and routine. When scenes and objects are treated as a unity, their motion patterns cannot be handled in a distinct manner, resulting in blurry and distorted visual appearances. As depicted in Fig. 1, it is obvious that the moving subject (i.e., the wrestling man) is more clear in the video obtained by separating objects from scenes than that by treating them as a single entity traditionally.

\(^\ast\) Corresponding Author
\(^1\) Codes have been released in https://github.com/iva-mzsun/MOSO
Based on the above insight, we propose a two-stage MOTion, Scene and Object decomposition framework (MOSO) for video prediction. We distinguish objects from scenes and utilize motion signals to guide their integration. In the first stage, MOSO-VQVAE is developed to learn motion, scene and object decomposition encoding and video decoding in a self-supervised manner. Each decomposed component is equipped with an independent encoder to learn its features and to produce a distinct group of discrete tokens. To deal with different motion patterns, we integrate the object and scene features under the guidance of the corresponding motion feature. Then the video details can be decoded and rebuilt from the merged features. In particular, the decoding process is devised to be time-independent, so that a decomposed component or a single video frame can be decoded for flexible visualization.

In the second stage, MOSO-Transformer is proposed to generate a subsequent video clip based on a previous video clip. Motivated by the production of animation, which first determines character identities and then portrays a series of actions, MOSO-Transformer firstly predicts the object and scene tokens of the subsequent video clip from those of the previous video clip. Then the motion tokens of the subsequent video clip are generated based on the predicted scene and object tokens and the motion tokens of the previous video clip. The predicted object, scene, and motion tokens can be decoded to the subsequent video clip using MOSO-VQVAE. By modeling video prediction at the token level, MOSO-Transformer is relieved from the burden of modeling millions of pixels and can instead focus on capturing global context relationships. In addition, our framework can be easily extended to other video generation tasks, including unconditional video generation and video frame interpolation tasks, by simply revising the training or generation pipelines of MOSO-Transformer.

Our contributions are summarized as follows:

- We propose a novel two-stage framework MOSO for video prediction, which could decompose videos into motion, scene and object components and conduct video prediction at the token level.
- MOSO-VQVAE is proposed to learn motion, scene and object decomposition encoding and time-independently video decoding in a self-supervised manner, which allows video manipulation and flexible video decoding.
- MOSO-Transformer is proposed to first determine the scene and object identities of subsequent video clips and then predict subsequent motions at the token level.
- Qualitative and quantitative experiments on five challenging benchmarks of video prediction and unconditional video generation demonstrate that our proposed method achieves new state-of-the-art performance.

2. Related Work

Video Prediction The video prediction task has received increasing interest in the computer vision field. ConvLSTM [37] combines CNN and LSTM architectures and adopts an adversarial loss. MCnet [44] models pixel-level future video prediction with motion and content decomposition for the first time. GVSD [46] proposes a spatio-temporal CNN combined with adversarial training to untangle foreground objects from background scenes, while severe distortion of object appearances exists in their predicted video frames. MCVD [45] adopts a denoising diffusion model to conduct several video-related tasks conditioned on past and/or future frames. Although previous models can predict consistent subsequent videos, they still suffer from indistinct or distorted visual appearances since they lack a stable generator or fail to decouple different motion patterns. SLAMP [1] and vid2vid [47] decomposes video appearance and motion for video prediction with the help of optical flow. SADM [4] proposes a semantic-aware dynamic model that predicts and fuses the semantic maps (content) and optical flow maps (motion) of future video frames. In addition to optical flow and semantic maps, Wu et al. [54] further utilizes instance maps to help separate objects from backgrounds. Although these works also decompose video components, they are more complicated than MOSO since they require much more additional information. Furthermore, these previous works are primarily based on generative adversarial networks or recurrent neural networks, while MOSO follows a recently developed two-stage autoregressive generation framework, which demonstrates greater potential on open domain visual generation tasks.

Two-stage Visual Generation The two-stage framework is first proposed for image generation [11, 14, 33] and demonstrates excellent generation ability. Motivated by the success, several attempts have been made to extend the two-stage framework to video generation tasks [14, 52, 55, 55]. For video prediction, MaskViT [20] encodes videos by frame though VQ-GAN [14] and models video tokens with a bidirectional Transformer through window attention. For unconditional video generation, VideoGPT [55] encodes videos by employing 3D convolutions and axial attention, and then models video tokens in an auto-regressive manner. However, existing two-stage works for video tasks do not consider video component decomposition and are affected by flicker artifacts and expensive computation costs.

3. MOSO

In this section, we present our proposed framework MOSO in detail. MOSO is a novel two-stage framework for video prediction and consists of MOSO-VQVAE and MOSO-Transformer, where MOSO-VQVAE encodes decomposed video components to tokens and MOSO-
Transformer conducts video prediction at the token level. The overall framework of MOSO is drawn in Fig. 2.

We denote a $T$-frame video as $x^T$, which is an abbreviation for $\{x_t\}_{t=1}^T$, where $x_t \in \mathbb{R}^{H \times W \times C}$ is the $t$-th video frame, $H$ is the height, $W$ is the weight and $C$ is the number of channels. Video prediction requires predicting a subsequent video clip $x^{T+1}_{K+1}$ based on a previous one $x^K_T$.

3.1. Stage I: MOSO-VQVAE

MOSO-VQVAE consists of a codebook $E$, three video encoders (i.e., motion, scene and object encoders) and a time-independent video decoder. The codebook is denoted as $E = \{e_i\}_{i=1}^N$, where $e_i \in \mathbb{R}^D$ is the $i$-th codebook entry, $N$ is the codebook size and $D$ is the dimension. MOSO-VQVAE first roughly decomposes the motion, scene and object components of the input video $x^T$ with a preprocessing algorithm, obtaining the motion video $m^T$, the scene video $s^T$ and the object video $o^T$. Then the video encoder of each component takes its corresponding video as input and learns its feature. Each feature is then quantized using the codebook and a distinct group of tokens is available as a by-product. Finally, the video decoder integrates the quantized features to reconstruct the input video.

3.1.1 Decomposing

Considering that most of the scene motion is caused by camera movements or environment changes, scene motion is typically either tiny or extremely dramatic. Based on this observation, we propose an efficient preprocessing algorithm to separate one video into three distinct videos (i.e., motion, scene and object videos) without additional annotations. Specifically, frame difference is calculated and used as the motion video $m^T$. Then a lower threshold $c_{lb}$ and an upper threshold $c_{ub}$ are adopted to filter pixels with middle differences to obtain the object video $o^T$. The scene video $s^T$ is composed of the left pixels with differences less than $c_{lb}$ or higher than $c_{ub}$, corresponding to tiny or dramatic movements. More details are presented in the appendix.

3.1.2 Encoding

The scene encoder consists of several downsample layers, a temporal compressing layer and a residual stack. Given a scene video, the downsample layers, i.e., 2D convolutions with stride 2, downsample each frame by a factor of $f$. Then the frame features are concatenated in the channel dimension and compose feature $z'_m \in \mathbb{R}^{H/f \times W/f \times D}$, where $TD$ is the number of channels. The temporal compressing layer, i.e., linear projection $R^{TD \rightarrow D}$, reduces the number of channels to $D$. The residual stack, composed of several residual layers $[21]$, then learns the scene feature $z_s$. The object encoder has the same structure as the scene encoder but takes an object video as input and outputs an object feature $z_o$.

The motion encoder replaces the temporal compressing layer in the scene encoder with a temporal self-attention layer and outputs motion feature $z_m$. In particular, the downsampled frame features are concatenated in the temporal dimension and split into $N_t$ parts, composing feature $z'_m \in \mathbb{R}^{H/f \times W/f \times N_t \times T/N_t \times D}$. Then the temporal self-attention is conducted on each $Z \in \{z'_m[n\times t]\}$:

$$Q = ZW_Q, K = ZW_K, V = ZW_V$$

$$Y = \text{softmax}(QK^T/\sqrt{D})V$$

where $h \in \{1, \ldots, \frac{H}{f}\}, w \in \{1, \ldots, \frac{W}{f}\}$, $n \in \{1, \ldots, N_t\}$ and $W_Q, W_K$ and $W_V$ are learnable $D \times D$ projection matrices.

The codebook is used to quantize each feature $z \in \{z_s, z_o, z_m\}$ and obtain discrete tokens as:

$$VQ(x^T)[h,w,t] = e_r, \quad VQ(x^T)[h,w,t] = r$$

$$r = \text{arg min}_{1 \leq i \leq N} ||e_i - z[h,w,t]||^2_2$$

where $h \in \{1, \ldots, \frac{H}{f}\}, w \in \{1, \ldots, \frac{W}{f}\}$, $t \in \{1, \ldots, T\}$, $z_s, z_o \in \mathbb{R}^{H/f \times W/f \times T}$, $z_m \in \mathbb{R}^{H/f \times W/f \times T \times N_t}$. $VQ$ obtains quantized features, $Q$ obtains maps of tokens, $e_i$ is the $i$-th codebook entry and $||*||^2$ denotes the calculation of L2 norm. The quantized features have the same shape as the raw features. Note that the temporal dimension of the scene and object features is 1, which is omitted for conciseness.

3.1.3 Decoding

The video decoder consists of a merge module, a residual stack and several upsample layers, i.e., transposed 2D convolutions with stride 2. To allow gradients to back-propagate to the encoders, the video decoder stops gradients of each quantized feature, obtaining $\tilde{z} = s_g(VQ(x^T) - z) + z$, where $s_g$ denotes the operator of stop-gradient.

The merge module dynamically integrates the object feature $\tilde{z}_o$ and the scene feature $\tilde{z}_s$ according to the $t$-th motion feature $\tilde{z}'_m$ to obtain the video feature $z^t$ for the $t$-th video frame. It first obtains three multi-scale weight features:

$$w^{t}_1 = F_1([\tilde{z}_o, \tilde{z}_s, \tilde{z}'_m])$$

$$w^{t}_2 = F_2([\downarrow_2(\tilde{z}_o), \downarrow_2(\tilde{z}_s), \downarrow_2(\tilde{z}'_m)])$$

$$w^{t}_3 = F_3([\downarrow_4(\tilde{z}_o), \downarrow_4(\tilde{z}_s), \downarrow_4(\tilde{z}'_m)])$$

where $[\star]$ concatenates input features along the channel dimension and $F_1, F_2$ and $F_3$ are linear functions $R^{3D \rightarrow D}$. $\downarrow_f$ denotes a downsample function implemented by 2D convolutions with stride 2, where $f$ denotes the sample factor. The weight features are then merged by:

$$w^{t}_1 = w^{t}_2 + \uparrow_2(w^{t}_3)$$

$$w^{t} = \sigma(F(w^{t}_1 + \uparrow_2(w^{t}_2)))$$

18729
\[ \sigma(*) \] denotes the sigmoid activation function and \( \mathcal{F} : R^D \rightarrow R^D \) is a linear function. \( \uparrow_f \) denotes an upsampling function implemented by bilinear interpolation, where \( f \) denotes the sample factor. The \( t \)-th video frame feature is then obtained by:

\[ z^t = \bar{z}_s \odot w^t + \bar{z}_o \odot (1 - w^t) \tag{8} \]

where \( \odot \) denotes Hadamard product.

The residual stack has the same structure as that in the encoders. It operates on video frame features and is blind to their temporal index. Finally, several upsample layers spatially upsample video features to reconstruct video frames. The reconstructed \( t \)-th video frame is denoted as \( \bar{x}_t \).

Notably, our video decoder is time-independent since the merge module, the residual stack and the upsample layers are agnostic to the temporal position of each video frame and have no access to other motion features \( \bar{x}_m^t, k \neq t \) when reconstructing the \( t \)-th video frame. In other words, we only need to obtain the scene, object and \( t \)-th motion features when decoding the \( t \)-th video frame. It not only allows flexible visualization but also facilitates the modeling process in the second stage. In particular, the decomposed scenes and objects can be decoded and visualized by the video decoder based on their tokens, as discussed in the appendix.

### 3.1.4 Training

The overall loss function \( L \) for training MOSO-VQVAE consists of the reconstruction loss \( L_{\text{rec}} \) and the commitment loss \( L_{\text{com}} \). The reconstruction loss \( L_{\text{rec}} \) optimizes the video encoders and the video decoder by minimizing the L2 distance and the perceptual loss \([57]\) between each pair.
of input and reconstructed video frames. The commitment loss $L_{com}$ optimizes the encoders to ensure the training of the encoders keeps in step with that of the codebook, thus preventing divergence:

$$L_{com} = \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{t=1}^{T} \left\| z_{h,w,t} - sg(VQ(x_{h,w,t})) \right\|^2$$

(9)

where $sg$ denotes the operator of stop-gradient and $\|*\|^2$ denotes the calculation of L2 norm. The exponential moving average (EMA) optimization method is used to optimize the codebook entries as in [35].

### 3.2. Stage II: MOSO-Transformer

MOSO-Transformer is comprised of three bidirectional Transformers: TransformerSO, TransformerG, and TransformerM. Given a previous video clip $x^K_t$, MOSO-Transformer predicts the subsequent video clip $x_{K+1}$ at the token level though pretrained MOSO-VQVAE. Since MOSO-VQVAE takes fixed length $T$-frame videos as input, we first pad the previous video clip $x^K_t$ to $T$ frames by duplicating the last frame, obtaining a pseudo video $\hat{x}_1^T$:

$$\hat{x}_1^T = \{x_1, \ldots, x_{K-1}, x_K, x_K, \ldots, x_K\}, \quad K \leq T$$

(10)

Then the motion, scene and object tokens of $\hat{x}_1^T$ can be produced by MOSO-VQVAE. Based on the scene and object tokens of $\hat{x}_1^T$, TransformerSO is trained to generate the scene and object tokens of $x_1^T$:

$$P(Q_s(x_1^T), Q_o(x_1^T); Q_s(\hat{x}_1^T), Q_o(\hat{x}_1^T))$$

(11)

To provide guidance for the generation of motion tokens, TransformerSO is used to learn the guidance embeddings $\{h_1, \ldots, h_T\}$ based on the scene and object tokens of $x_1^T$. Considering that both TransformerSO and TransformerG require learning the scene and object content at the token level, we train them with shared parameters.

Rethinking the temporal split operation demonstrated in Eq. (1), when $T$ can be exactly divided by $K$ and the temporal dimension is $N_t = \frac{T}{K}$ parts, the motion tokens of the target video $x_1^T$ and the pseudo video $\hat{x}_1^T$ satisfy:

$$Q(x_1^T) = Q(\hat{x}_1^T), \quad 1 \leq t \leq K$$

(12)

Namely, the motion tokens for the first $K$ video frames in the target and pseudo videos are totally the same. A detailed explanation of this property and a more general situation are presented in the appendix.

Based on the above property, TransformerM only needs to generate motion tokens of the last $T-K$ frames of $x_1^T$ based on the guidance embeddings $\{h_1, \ldots, h_T\}$ and the motion tokens of the first $K$ given video frames of $\hat{x}_1^T$. Following the training pipeline in [7], we first randomly mask $\gamma(r)$ proportion of motion tokens with mask tokens $[M]$, where $r$ is a scalar and $\gamma(r)$ is a monotonically declining function with $\gamma(0) = 1$ and $\gamma(1) = 0$. Then TransformerM is
trained through a cross-entropy loss to model the distribution of masked motion tokens based on unmasked motion tokens, the guidance embeddings and the motion tokens of the first $K$ video frames of $\tilde{x}_1$: $$P(\{Q_m(x_1^T)_{n\neq[M]}\}; \{Q_m(x_1^T)_{n\neq[M]}\}; \{Q_m(\tilde{x}_1^T)^r\}; \{h_t\})$$ $\text{K} + 1 \leq n \leq T,\ 1 \leq r \leq K,\ 1 \leq t \leq T$ \hspace{1cm} (13)

3.2.1 Predicting

Inspired by the production of animation, which first creates figure characters and then depicts a series of actions, we predict the subsequent video clip by first determining its scene and object identities and then producing dynamic motions at the token level. In particular, Transformer$_{SO}$ firstly outputs the distribution of the scene and object tokens of $x_1^T$ as in Eq. (11). Then the scene and object tokens are randomly sampled and the guidance embeddings can be obtained through the Transformer$_G$. Given the guidance embeddings and a template filled with mask tokens $[M]$, Transformer$_M$ performs the following two steps for $S$ iterations to predict the motion tokens: (1) outputting the distribution of motion tokens as specified in Eq. (13) and sampling motion tokens in the masked positions; (2) randomly masking $\gamma(s/S)$ proportion of all motion tokens with previous unmasked tokens fixed, where $s$ denotes the $s$-th iteration. Finally, the predicted video frames $x_{T+1}^T$ are decoded from the generated tokens by MOSO-VQVAE. In addition, our MOSO-Transformer can be easily extended to other tasks involving unconditional video generation and video frame interpolation as shown in Fig. 2.

4. Experiments

In this section, we compare our method with prior works for video prediction and unconditional video generation tasks on five challenging benchmarks, including BAIR [13], RoboNet [9], KTH [36], KITTI [19] and UCF101 [39]. Fig. 3 shows the qualitative comparison of video prediction. We start with introducing evaluation metrics and experimental implementations.

Metrics We adopt five evaluation metrics: Fréchet Video Distance (FVD), Fréchet Instance Distance (FID), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) and Learned Perceptual Image Patch Similarity (LPIPS). FVD [42] measures the distance between distributions of predicted videos and real-world videos. We calculate FVD with codes released by StyleGAN-V$^2$ [38], which has been proven to be a precise implementation of the official one. To obtain the FVD score, we pass those given previous video frames and calculate between the predicted frames and corresponding ground truths following MaskViT [20]. For unconditional video generation,

we follow StyleGAN-V [38] and calculate FVD on 2048 randomly sampled videos. FID [22] evaluates the generated videos by frame. PSNR [24], SSIM [48] and LPIPS [57] measure frame-wise similarity between predicted and ground truth videos. We conduct one trial per video on the BAIR datasets. On the KTH, KITTI and RoboNet datasets, we follow [3, 20] and report the best SSIM, PSNR and LPIPS scores over 100 trials per video to account for the stochastic nature of video prediction. FVD is calculated over all 100 trials with batch size being 256 following [20].

Implementation MOSO-VQVAE encodes videos by clip. When training MOSO-VQVAE, each video clip has a fixed length $T$, which is set as 16 for BAIR, 12 for RoboNet and 20 for KTH and KITTI. $N_t$ is set as 1 for BAIR, 6 for RoboNet, 2 for KTH and 4 for KITTI. For UCF101, $T$ and $N_t$ are set as 12 and 3 for video prediction or 16 and 1 for unconditional video generation. The batch size is 32 for $64^2$ resolution and 4 for others. The codebook size $N$ is 16384. The preprocessing algorithm is used for the first 50k iterations. After that, the scene and object encoders take raw video frames as input and learn scene and object identities in a self-supervised manner, and the motion encoder still takes the frame difference as input. For MOSO-Transformer, the batch size is 16 for $64^2$ resolution and 4 for others. $\gamma(s/S)$ is set to decay as cosine and $S$ is set as 16. For long video prediction, tokens of the last video clip are used as previous tokens for iterative prediction. DeepSpeed [34] and mixed-precision FP16/FP32 [26] are utilized for fast training. Experiments are conducted on 4 A100s.

4.1. Video Prediction

BAIR. The BAIR robot pushing dataset [13] records random movements of robotic arms. Following prior work [53], we predict subsequent 15 video frames given only 1 previous video frame, and all videos are resized to $64 \times 64$ resolution. Qualitative comparison of MOSO with MaskViT is given in Fig. 3. Our MOSO achieves new state-of-the-art performance on this dataset as reported in Table 1, outperforming prior best work by 3.3 FVD.

RoboNet. The RoboNet dataset [9] contains more than 15 million videos of 7 robotic arms pushing things in different bins. Following prior works [3, 51], we randomly select 256 videos for testing and predict 10 subsequent video frames given 2 previous video frames. As shown in Table 2, MOSO achieves significant improvements on all metrics than prior works at both $64^2$ and $256^2$ resolution.

Both MOSO and MaskViT follow the pipeline of VQVAE and Transformer, while MOSO outperforms MaskViT by a large margin on the RoboNet dataset. We attribute this phenomenon to two reasons. Firstly, MaskViT models videos by frame, which can lead to flicker artifacts in videos with static backgrounds like RoboNet [20], while MOSO models videos by clip and thus obtains better con-

---

$^2$https://github.com/universome/stylegan-v
Table 1. Quantitative comparison with other methods on BAIR for video prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>Params</th>
<th>FVD ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVT [32]</td>
<td>-</td>
<td>152.8</td>
</tr>
<tr>
<td>SAVP [27]</td>
<td>-</td>
<td>116.4</td>
</tr>
<tr>
<td>DVD-GAN-FP [8]</td>
<td>-</td>
<td>109.8</td>
</tr>
<tr>
<td>VT (S) [49]</td>
<td>46M</td>
<td>106.0</td>
</tr>
<tr>
<td>TrtV-GAN-FP [28]</td>
<td></td>
<td>103.3</td>
</tr>
<tr>
<td>VideoGPT [55]</td>
<td>-</td>
<td>103.3</td>
</tr>
<tr>
<td>CCVS [30]</td>
<td>-</td>
<td>99.0</td>
</tr>
<tr>
<td>VT (L) [19]</td>
<td>373M</td>
<td>94.0</td>
</tr>
<tr>
<td>MaskViT [20]</td>
<td>180M</td>
<td>93.7</td>
</tr>
<tr>
<td>FinVid [3]</td>
<td>302M</td>
<td>93.6</td>
</tr>
<tr>
<td>MCVD [45]</td>
<td>251M</td>
<td>89.5</td>
</tr>
<tr>
<td>RaMVD [23]</td>
<td>-</td>
<td>89.2</td>
</tr>
<tr>
<td>NUWA [53]</td>
<td>870M</td>
<td>86.9</td>
</tr>
<tr>
<td>MOSO</td>
<td>265M</td>
<td>83.6</td>
</tr>
</tbody>
</table>

Table 2. Comparison with other methods on RoboNet for video prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>Params</th>
<th>H × W</th>
<th>FVD ↓</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaskViT [20]</td>
<td>257M</td>
<td>64²</td>
<td>133.5</td>
<td>23.2</td>
<td>80.5</td>
<td>0.042</td>
</tr>
<tr>
<td>SVG [43]</td>
<td>298M</td>
<td>64²</td>
<td>123.2</td>
<td>23.9</td>
<td>87.8</td>
<td>0.060</td>
</tr>
<tr>
<td>GHVAE [50]</td>
<td>599M</td>
<td>64²</td>
<td>95.2</td>
<td>24.7</td>
<td>89.1</td>
<td>0.036</td>
</tr>
<tr>
<td>FinVid [3]</td>
<td>302M</td>
<td>64²</td>
<td>62.5</td>
<td>28.2</td>
<td>89.3</td>
<td>0.024</td>
</tr>
<tr>
<td>MOSO</td>
<td>265M</td>
<td>64²</td>
<td>53.3</td>
<td>31.2</td>
<td>92.1</td>
<td>0.017</td>
</tr>
<tr>
<td>MaskViT [20]</td>
<td>228M</td>
<td>256²</td>
<td>211.7</td>
<td>20.4</td>
<td>67.1</td>
<td>0.170</td>
</tr>
<tr>
<td>MOSO</td>
<td>265M</td>
<td>256²</td>
<td>91.5</td>
<td>26.3</td>
<td>79.9</td>
<td>0.096</td>
</tr>
</tbody>
</table>

Table 3. Comparison with other methods on KITTI for video prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>Params</th>
<th>H × W</th>
<th>FVD ↓</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVG [43]</td>
<td>298M</td>
<td>64²</td>
<td>1217.3</td>
<td>15.0</td>
<td>41.9</td>
<td>0.327</td>
</tr>
<tr>
<td>FinVid [3]</td>
<td>302M</td>
<td>64²</td>
<td>884.5</td>
<td>17.1</td>
<td>49.1</td>
<td>0.217</td>
</tr>
<tr>
<td>GHVAE [50]</td>
<td>599M</td>
<td>64²</td>
<td>552.9</td>
<td>15.8</td>
<td>51.2</td>
<td>0.286</td>
</tr>
<tr>
<td>MaskViT [20]</td>
<td>181M</td>
<td>64²</td>
<td>401.9</td>
<td>27.2</td>
<td>58.1</td>
<td>0.089</td>
</tr>
<tr>
<td>MOSO</td>
<td>177M</td>
<td>64²</td>
<td>395.3</td>
<td>25.6</td>
<td>74.8</td>
<td>0.086</td>
</tr>
<tr>
<td>MaskViT [20]</td>
<td>228M</td>
<td>256²</td>
<td>446.1</td>
<td>26.2</td>
<td>40.7</td>
<td>0.270</td>
</tr>
<tr>
<td>MOSO</td>
<td>219M</td>
<td>256²</td>
<td>516.6</td>
<td>21.1</td>
<td>59.2</td>
<td>0.265</td>
</tr>
</tbody>
</table>

Table 4. Comparison with prior works on KTH for video prediction.

<table>
<thead>
<tr>
<th>Method</th>
<th>H × W</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>LPIPS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVG-LP [10]</td>
<td>64²</td>
<td>23.9</td>
<td>80.0</td>
<td>0.129</td>
</tr>
<tr>
<td>Struct-VRNN [29]</td>
<td>64²</td>
<td>24.3</td>
<td>76.6</td>
<td>0.124</td>
</tr>
<tr>
<td>SV2P [2]</td>
<td>64²</td>
<td>25.9</td>
<td>78.2</td>
<td>0.232</td>
</tr>
<tr>
<td>SAVP [27]</td>
<td>64²</td>
<td>26.0</td>
<td>80.6</td>
<td>0.116</td>
</tr>
<tr>
<td>MCVD [45]</td>
<td>64²</td>
<td>26.4</td>
<td>81.2</td>
<td>-</td>
</tr>
<tr>
<td>GK [17]</td>
<td>64²</td>
<td>27.1</td>
<td>83.7</td>
<td>0.092</td>
</tr>
<tr>
<td>MOSO</td>
<td>64²</td>
<td>29.8</td>
<td>82.2</td>
<td>0.083</td>
</tr>
</tbody>
</table>

KITT. The KITTI dataset [19] is a challenging dataset with only 57 training videos and dramatic moving scenes. MaskViT [20] outperforms prior works by a large margin on this dataset, and we achieve comparable performance with MaskViT at both 64² and 128² resolutions as reported in Table 3. Different from the VQ-GAN adopted in MaskViT, which sets the weight of the adversarial loss as 1, MOSO-VQVAE adopts a smaller loss weight 0.1 as specified in the ablation study. Despite such loss weight helps MOSO-VQVAE achieve outstanding performance on other datasets, it seems to be too small for KITTI to compete with the perceptual loss with weight 1, leading to some checkerboard artifacts in the predicted video frames as shown in Fig. 3. These checkerboard artifacts are produced by the ResNet-50 network used in the perceptual loss as indicated in [12, 31].

4.2. Other Video Generation Tasks

Unconditional Video Generation. MOSO-Transformer can be trained for unconditional video generation by replacing the scene and object tokens of the given video clip with mask tokens and removing given motion tokens. We quantitatively compare MOSO with other models on the UCF101 dataset for unconditional video generation as reported in Table 5. The results demonstrate that our MOSO outperforms the previous method [38] by 219.1 on FVD.

Video Frame Interpolation. MOSO can directly perform video frame interpolation after being trained for video prediction. Based on raw motion, scene and object tokens, MOSO-Transformer generates interpolated video frames by initializing their motion tokens with mask tokens and then removing mask tokens through steps, where is 16. Samples on the RoboNet, KTH and KITTI datasets are given in Fig. 4, which shows that MOSO could interpolate consistent video frames.

Video Manipulation. By separating object identities
Table 5. Quantitative comparison with other methods on UCF101 for unconditional video generation.

<table>
<thead>
<tr>
<th>Method</th>
<th>FVD ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoCoGAN [41]</td>
<td>2886.9</td>
</tr>
<tr>
<td>+StyleGAN2 backbone</td>
<td>1821.4</td>
</tr>
<tr>
<td>MoCoGAN-HD [40]</td>
<td>1729.6</td>
</tr>
<tr>
<td>VideoGPT [55]</td>
<td>2880.6</td>
</tr>
<tr>
<td>DIGAN [56]</td>
<td>1630.2</td>
</tr>
<tr>
<td>StyleGAN-V [38]</td>
<td>1431.0</td>
</tr>
<tr>
<td>MOSO</td>
<td>1202.6</td>
</tr>
</tbody>
</table>

Table 6. Ablation studies on KTH 64² for video prediction. non decom.: non-decomposition. mo. co.: motion and content decomposition. mo. sc. ob.: motion, scene and object decomposition. pre. alg. denotes the preprocessing algorithm. Trans. denotes Transformer.

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
<th>LPIPS↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>non decom. (f_{non}: 8)</td>
<td>605M+308M</td>
<td>21.5</td>
<td>62.8</td>
<td>0.202</td>
</tr>
<tr>
<td>mo. co. (f_{mo}:8,f_{co}:4)</td>
<td>611M+265M</td>
<td>25.4</td>
<td>68.8</td>
<td>0.147</td>
</tr>
<tr>
<td>mo. co. (f_{mo}:8,f_{co}:2)</td>
<td>535M+265M</td>
<td>27.1</td>
<td>75.5</td>
<td>0.129</td>
</tr>
<tr>
<td>mo. sc. ob. (MOSO)</td>
<td>593M+265M</td>
<td>29.8</td>
<td>82.2</td>
<td>0.083</td>
</tr>
<tr>
<td>MOSO - pre. alg.</td>
<td>593M+265M</td>
<td>29.2</td>
<td>82.0</td>
<td>0.086</td>
</tr>
<tr>
<td>MOSO - merge Module</td>
<td>557M+265M</td>
<td>28.3</td>
<td>79.3</td>
<td>0.125</td>
</tr>
<tr>
<td>MOSO + single Trans.</td>
<td>593M+286M</td>
<td>28.9</td>
<td>80.1</td>
<td>0.116</td>
</tr>
</tbody>
</table>

Table 7. Number of tokens changes with downsample factors.

<table>
<thead>
<tr>
<th>Method</th>
<th>f_{non}</th>
<th>f_{mo}</th>
<th>f_{co}</th>
<th>f_{s}</th>
<th>f_{o}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Down. factor</td>
<td>8</td>
<td>4</td>
<td>(8,4)</td>
<td>(8,2)</td>
<td>(8,4,4)</td>
</tr>
<tr>
<td>Token length</td>
<td>1280</td>
<td>5120</td>
<td>1536</td>
<td>2304</td>
<td>1792</td>
</tr>
<tr>
<td>FLOPs(×10^9)</td>
<td>389</td>
<td>2382</td>
<td>481</td>
<td>796</td>
<td>581</td>
</tr>
</tbody>
</table>

fair comparison, we ensure the parameters of both VQVAE and the Transformer of all involved models are comparable. As reported in Table 6, the performance of MOSO outperforms the non-decomposition method and the motion and content decomposition method by a large margin.

5. Conclusion and Discussions

In this paper, we propose a novel two-stage motion, scene and object decomposition framework for video prediction. Extensive experiments show that our method achieves new state-of-the-art performance on several challenging benchmarks, demonstrating the importance of decomposing the motion, scene and object video components.

Our research has two main limitations. First, although the preprocessing algorithm is efficient and effective, it is not delicate enough. More expressive and powerful tools, e.g. optical flow, may help decompose the motion, scene and object components better. Second, we do not enlarge the model and dataset for video prediction, while several works [33, 53] have revealed the potential of a huge Transformer for the open-domain visual generation. Future works are highly-encouraged for further exploration.

6. Acknowledgments

We thank Yonghua Pan, Jiawei Liu and Zihan Qin for reviewing early drafts and helpful suggestions. This work was supported by the National Key Research and Development Program of China (No. 2020AAA0106400) and National Natural Science Foundation of China (U21B2043, 62102419, 62102416), and was sponsored by CAAI-Huawei MindSpore Open Fund.
References


[38] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 6, 7


