Layered Controllable Video Generation

Jiahui Huang[1,2], Yuhe Jin[1], Kwang Moo Yi[1], and Leonid Sigal[1,2,3,4]

[1]University of British Columnbia [2]Vector Institute for AI
[3]CIFAR AI Chair [4]NSERC CRC Chair
{gabrie20,yuhejin,kmyi,lsigal}@cs.ubc.ca

Abstract. We introduce layered controllable video generation, where we, without any supervision, decompose the initial frame of a video into foreground and background layers, with which the user can control the video generation process by simply manipulating the foreground mask. The key challenges are the unsupervised foreground-background separation, which is ambiguous, and ability to anticipate user manipulations with access to only raw video sequences. We address these challenges by proposing a two-stage learning procedure. In the first stage, with the rich set of losses and dynamic foreground size prior, we learn how to separate the frame into foreground and background layers and, conditioned on these layers, how to generate the next frame using VQ-VAE generator. In the second stage, we fine-tune this network to anticipate edits to the mask, by fitting (parameterized) control to the mask from future frame. We demonstrate the effectiveness of this learning and the more granular control mechanism, while illustrating state-of-the-art performance on two benchmark datasets. Our project website/code can be found at gabriel-huang.github.io/layered_controllable_video_generation.

1 Introduction

Advances in deep generative models have led to impressive results in image and video synthesis. Typical forms of such models, including Variational Autoencoder (VAE) [32], Generative Adversarial (GAN) [19] and recurrent (RNN) [43] formulations, can produce complex and highly realistic content. However, synthesis of realistic images/videos, without the ability to control the depicted content in them, has limited practical utility. This has led to a variety of conditional generative tasks and formulations.

In the image domain, both coarse- (*e.g.*, sentence [70]) and fine-level (*e.g.*, layout [72] and instance attribute [18]) control signals have been explored. The progress on the video side, on the other hand, has generally been more modest, in part due to an added challenge of synthesizing temporally coherent content. *Future frame prediction* [12,16,17,34,38,56,55,57,64] conditions the future generated frames on one (or a couple) seed frame(s). But this provides very limited control as the object(s), or person(s), depicted in the conditioned frame can move in a multitude of ways, particularly as predictions are made longer into the future. To address this, a number of methods condition future frames on the

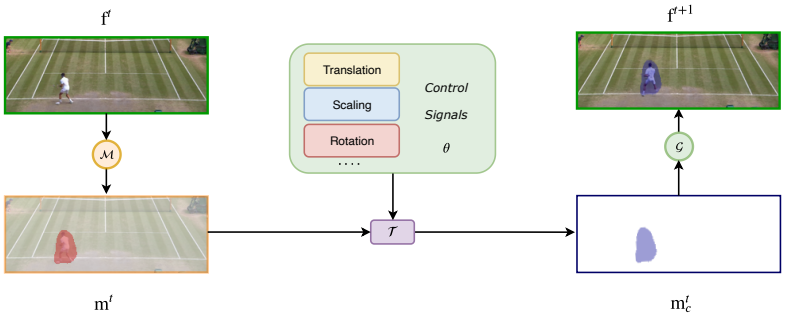

Fig. 1. Layered Controllable Video Generation. Illustration of the proposed task, where a frame at time t is first decomposed into a foreground/background layers, using the learned mask network $\mathcal{M}$, and then the user is allowed to modify this mask with control signals θ (*e.g.*, by shifting it by Δ_t) to control the next generated frame realized by generator $\mathcal{G}$. The foreground source and target mask are illustrated in red and blue.

action [22,52] and object [40] label. Still, they only provide very coarse global video-level control.

More recent approaches focus on the ability to control the video content on-the-fly at the frame-level. Typically, these methods are formulated as conditional auto-regressive (or recurrent) models that generate one frame at a time, conditioned, for example, on the discrete action label [30] or keypoint-based human pose specification [54,58,68] (*e.g.*, obtained from a target video source [69]). Such methods, however, require dense per-frame annotation of actions or poses at training time, which are costly to obtain and make it challenging to employ such approaches in realistic environments. The task of *playable video generation* [39], has been introduced to address these limitations. In playable video generation, the discrete action space is *discovered* in an unsupervised manner and can then be used as a conditioning signal in an auto-regressive probabilistic generative model. While obtaining impressive results, with no intermediate supervision and allowing frame-level control over the generative process, [39] is inherently limited to a single subject and a small set of discrete action controls.

Thus, in this work, we aim to allow richer and more granular control over the generated video content, while similarly requiring no supervision of any kind – *i.e.*, having only raw videos as training input, as in [39]. To do so, we make an observation (inspired by early works in vision [59,28,33]) that a video can be effectively decomposed into foreground / background layers. The background layer corresponds to static (or slowly changing) parts of the scene. The foreground layer, on the other hand, evolves as the dominant objects move. Importantly, the foreground mask itself contains position, size and shape information necessary to both characterize and render the moving objects appropriately. Hence, this foreground mask can in itself be leveraged as a useful level of abstraction that allows both intuitive and simple control over the generative process.

With the above intuition, we therefore propose an approach that automatically learns to segment video frames into foreground and background, and at the same time generates future frames conditioned on the segmentation and the past frame, iteratively. To allow on-the-fly control of video generation, we ex-

pose the foreground mask to the user for manipulation – *e.g.*, translations of the mask leads to corresponding in-plane motion of the object, resizing it would lead to the depth away/towards motion, and changes in the shape of the mask can control the object pose.

From the technical perspective, the challenge of formulating such a method is two fold. First, unsupervised foreground/background layer segmentation is highly ambiguous, particularly if the background is not assumed to be static and the foreground motions can be small. Second, user input needs to be anticipated to ensure model learns how to *react* to changes in the mask, without explicit access to such information. To this end, we propose a two-stage learning procedure. In the first stage, the network learns how to perform foreground/background separation and, conditioned on this layered representation, future frame prediction. Specifically, we introduce a set of sophisticated losses and a dynamic prior to learn how to predict a foreground mask and leverage VQ-VAE [44] to predict foreground and background latent content which is then fused and decoded to the next frame. In the second stage, we simulate user input and fine-tune the generative model such that this user input can be appropriately handled.

Contributions: Our contributions are multi-fold.

- From raw video data, our model learns to generate foreground/background separation masks in an unsupervised manner. We then leverage the foreground layer as a flexible (parametric) user control mechanism for the generative process. This provides both richer and more intuitive control compared to action vectors [39] or sparse trajectories [21].
- To effectively train our model we introduce two-stage training: the first stage tasked with learning how to separate layers and perform future frame prediction; the second, to adopting and anticipating user control.
- To prevent over-/under-segmentation, we regularize layer separation with sparsity loss and dynamic mask size prior.
- Finally, we validate our approach on multiple datasets and show that we are able to generate state-of-the-art results and, at the same time, allow higher level of control over the generated content without any supervision.

2 Related works

Video generation. Early video generation techniques proposed to generate a video as a whole. Most of these convert a noise vector, sampled from simple distribution or a prior, to a video, using a GAN [55,2,61] or VAE [6,12] formulation. More recent architectures leveraged transformer-based formulations [67,41,46,64] that have generally resulted in higher quality video outputs. As an alternative to 3D (transposed) convolution techniques, that generate all frames at once, recurrent auto-regressive variants [29] have also been explored. While these early works largely focused on the video quality and resolution, more recently, the focus has shifted to conditioned or controlled video generation.

Video generation with global control. Future frame prediction considers the task of generating a video conditioned on a few starting seed frames. Early ap-

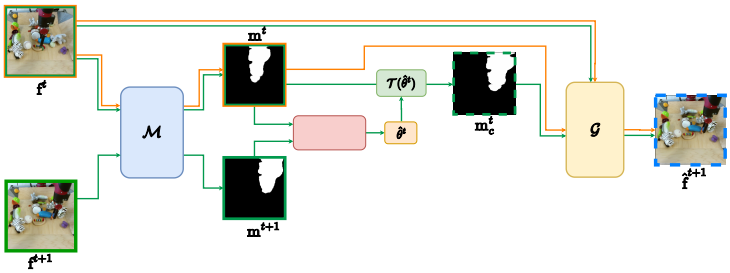

Fig. 2. Illustration of the Proposed Two-stage Training. The flow of Stage I is represented in orange and Stage II in green. $\mathcal{M}$ denotes the *mask network*, which estimates the foreground/background mask, and $\mathcal{G}$ denotes the *generator network* that takes the current frame and a mask to produce the next frame. $\mathcal{O}(\cdot)$ is the optimization procedure described in Eq.(12). $\mathcal{T}(\cdot)$ is a differentiable function that transforms a mask to a target shape using *user control signal* θ^t.

proaches to future prediction employed deterministic models [16,38,56,65] that failed to model uncertainty in the future induced by variability of unfolding events. To overcome this limitation, later methods, based on VAE [6,12], GAN [36,35], and probabilistic formulations [66], attempted to introduce real-world stochasticity into the generative process. Action label conditioning, in combination with seed frames or not, where a video sequence is generated conditioned on an input action label [31,62] is also popular; some such approaches leverage disentangled factored representations (*e.g.*, of subject identity and action [22]). Other types of global conditioning signals include action-object tuples [40]. However, these methods, collectively, require action annotations for training, and, more importantly, do not allow control at the frame-level.

Video generation with frame-level control. More granular control, at the frame-level, has also been explored. Pose-guided generative models first generate a sequence sparse [58,54,68] or dense [69] keypoint human poses, either predictively [54,58] or from a source video [69], and then use these for conditional generation of respective video frames. However, these methods are only applicable to videos of human subjects and require either pose annotations or a pre-trained pose detector. Alternatively, individual frames can be conditioned on action labels [30,11,42]. However, these methods require dense frame-level annotations, which are only available in limited environments such as video games. Closest to our work, Menapace *et al.* proposed Playable Video Generation [39], which, in an unsupervised manner, discovers semantically consistent actions meanwhile generating the next frame conditioned on the past frames and an input action, thus providing user control to the generation process. However, their method is limited to a small set of discrete action controls and explicitly assumes a single moving object. In contrast, our method allows richer and more granular control, and can be used to generate, and control, videos with multiple objects.

Video generation with pixel-level control. It is worth mentioning that some prior works attempted to use dense semantic segmentation [45,60] and sparse trajectories [21] to control video generation. While such approaches allow granular control at a pixel-level, these representations are incredibly difficult for

a user to produce or modify. We also use a form of (foreground) segmentation for control, however, it is unsupervised, class-agnostic, and can be easily controlled either parametrically or non-parametrically.

Unsupervised object segmentation. Layered representations have long history in computer vision [59,28,33], and are supported by neuroscience evidence [63]. Traditional techniques for this rely on feature clustering [1,5,24] and statistical background modeling and subtraction [9,14,50]. Such techniques work best for videos where the background is (mostly) static, lighting fixed and the foreground is fast moving; we refer readers to [20] for an extensive analysis and discussion. More recent techniques have focused on generative formulations for the task. In particular, Bielski *et al.* proposed the PerturbGAN [8], their model generates the foreground and background layers separately, and uses a perturbation strategy to enforce the generation of semantically meaningful masks. Related, MarioNette [49] learns to decompose scenes into a background and a learned dictionary of sprites. Other approaches focus on separation of videos into natural layers (*e.g.*, to factorize secondary effects such as shadows and reflections [4]) and to control which layer to attend to [3]. Similar to [55], we decompose frames and separately model foreground/background content that is then composed/fused together to produce video. However, unlike [55] and others, we allow the user to have frame-level control over the foreground mask using both intuitive parametric and nonparametric controls.

3 Method

Our fundamental goal is frame-level controllable video generation. We address this by proposing a model that first segments the input image into foreground / background layers using a mask and then allows user to control the generative process by applying (parameterized) modifications to this mask. While it is possible to train such a model directly, we find that it is difficult in such a case to learn to disentangle foreground/background mask prediction and the controlled generation process (see our ablation study in Section 4.2 for details).

We address this by separating the training into two stages (see Figure 2). Similar to other vision domains (*e.g.*, recognition [7,10], multi-modal learning [37,51]), where it was shown that pre-training was helpful for a number of downstream tasks, we introduce a related pre-training (Stage I) task and formulation, which ultimately helps in our final controllable video generation (Stage II).

Intuitively, **Stage I** pre-training, which we describe in Section 3.1, learns how to preform foreground/background separation, using $\mathcal{M}(\cdot)$, and, conditioned on this layered representation, is optimized for future frame prediction. This stage does not consider controlability within the generation process. Instead, it is designed to predict the most *likely* future frame(s) given a single input frame. Note that the task is effectively one of forecasting, but without attempting to model full distribution over potential futures, rather just a single most likely video trajectory exemplified by the observed video itself.

Given the pre-trained foreground/background mask predictor $\mathcal{M}(\cdot)$ and generator $\mathcal{G}(\cdot)$, learned in Stage I, in **Stage II** (see Section 3.2) we effectively *fine-tune* this model to take into account user control. Notably, Stage I and Stage II share foreground/background separation, but in Stage I the generator $\mathcal{G}(\cdot)$ learns the expected *dynamics* as part of its generative process. The main goal of Stage II then is to fine-tune $\mathcal{G}(\cdot)$ in such a way as to allow direct conditioning of said dynamics based on user controlled edits to the mask.

3.1 Stage I: Pre-training for mask-based generation

We first train our method with the focus of generating high-quality foreground-background segmentation masks, **without** introducing controlability into the generation process. Doing so requires two main objectives: (1) high-quality estimation of the current frame's segmentation mask $\mathbf{m}^t$; and (2) pretraining a frame generator for mask-based future frame prediction and generation. Writing the two objectives as loss terms $\mathcal{L}_{\text{mask}}$ and $\mathcal{L}_{\text{img}}$, respectively, the total loss for the first stage training of our method $\mathcal{L}_{\text{total}}$ can be written as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{mask}} + \mathcal{L}_{\text{img}}. \tag{1}$$

We detail each loss term in the following.

Regularizing the mask. To train our method to generate proper masks without any supervision, we regularize the mask with three losses: (1) $\mathcal{L}_{\text{bg}}$ – the contents of the background should not change; (2) $\mathcal{L}_{\text{fg}}$ – there should be as little amount of foreground as possible since classifying all pixels as foreground provides a trivial solution for $\mathcal{L}_{\text{bg}}$; and (3) $\mathcal{L}_{\text{bin}}$ – the masking should be binary for effective separation. We therefore write the mask regularization loss $\mathcal{L}_{\text{mask}}$ as

$$\mathcal{L}_{\text{mask}} = \lambda_{\text{bg}}\mathcal{L}_{\text{bg}} + \lambda_{\text{fg}}\mathcal{L}_{\text{fg}} + \lambda_{\text{bin}}\mathcal{L}_{\text{bin}}, \tag{2}$$

where λ_{bg}, λ_{fg}, and λ_{bin} are the hyperparameters controlling how much each loss term affects the mask regularization.

— $\mathcal{L}_{\text{bg}}$. We aim to ensure that the mask correctly identifies the background, *i.e.*, non-moving parts of the scene. Hence, we simply define it as the amount of change in the masked out (background) region between two consecutive frames. We write

$$\mathcal{L}_{\text{bg}} = \left\| (1 - \mathbf{m}^t) \odot \mathbf{f}^t - (1 - \mathbf{m}^t) \odot \mathbf{f}^{t+1} \right\|_1, \tag{3}$$

where $\odot$ denotes the elementwise multiplication. Note that we define this loss using the ℓ_1 norm, as changes in the scene are not strictly restricted to the mask – *e.g.* shadows of moving objects can occur in the background, or other scenic changes, such as a global illumination change can happen – and the ℓ_1 norm leaves room for the method to incorporate these changes if necessary.

— $\mathcal{L}_{\text{fg}}$. As mentioned earlier, $\mathcal{L}_{\text{bg}}$ alone, leaves room for a trivial solution— assigning $\mathbf{m}^t$=1 results in $\mathcal{L}_{\text{bg}}$=0 regardless of the values of $\mathbf{f}^t$ and $\mathbf{f}^{t+1}$. This

could be avoided by enforcing an additional loss term that penalizes having too many foreground pixels, but a naïve regularization is not sufficient, as the amount of the actual foreground pixels may drastically change from frame to frame – *e.g.*, robot arm moving close to the camera vs. further away.

We thus propose to regularize based on the amount of evident change between $\mathbf{f}^t$ and $\mathbf{f}^{t+1}$, approximated using simple background subtraction. Specifically, for a pixel index (i, j), if we denote whether the pixel changed between the two consecutive frames $\mathbf{f}^t$ and $\mathbf{f}^{t+1}$ as

$$\boldsymbol{\mu}_{ij}^t = \begin{cases} 1, & \text{if } \left\|\mathbf{f}_{ij}^{t+1} - \mathbf{f}_{ij}^t\right\|_1 > \tau \\ 0, & \text{otherwise} \end{cases}, \tag{4}$$

where τ is a threshold for controlling the sensitivity, we can use the average of $\boldsymbol{\mu}_{ij}^t$ as a rough estimate for how much of the pixels should be foreground, dynamically for each consecutive frame. We thus write

$$\mathcal{L}_{\text{fg}} = \max\left\{0, \left\|\mathbb{E}_{ij}\left[\mathbf{m}_{ij}^t\right] - \mathbb{E}_{ij}\left[\boldsymbol{\mu}_{ij}^t\right]\right\|_1\right\}, \tag{5}$$

where $\mathbf{m}_{ij}^t$ is the generated mask value at pixel index (i, j) at frame t, and $\boldsymbol{\mu}_{ij}^t$ is the variation between adjacent frames exceeds the threshold. We note that the balance between the hyperparameter settings related to $\mathcal{L}_{\text{fg}}$ and $\mathcal{L}_{\text{bg}}$ is important since they govern how the loss behaves—*e.g.* a wider mask that covers all potential changes in the scene, or a tight mask that only focuses on the actual changing locations. We empirically set the ratio between λ_{bg} and λ_{fg} to be 100:1, and we gradually decrease this ratio in training for faster convergence.

— $\mathcal{L}_{\text{bin}}$. Finally, there is one last loophole for the network to cheat its way through the two mask regularizors – by producing intermediate values in the range $[0, 1]$. In fact, we found in our early experiments that *soft masks* allows the deep network to encode information of the image in the mask alone, and any modification on the mask will result in degenerated frames (see *Supp. Mat.* Section D). Therefore, we encourage the mask to be binary [8]:

$$\mathcal{L}_{\text{bin}} = \min\{\mathbf{m}^t, (1 - \mathbf{m}^t)\}. \tag{6}$$

At test time we clip these pseudo-binary masks to 0/1 with a 0.5 threshold.

Learning to predict the next frame. To pretrain the generator for next frame prediction, we follow the VQGAN [15] framework. The choice of VQGAN is motivated by two factors: (1) its ability to preserve spatial information in the latent space, allowing effective masking that we need, and (2) its illustrated high-quality image generation performance. We note that VQGAN itself, or its use in this context, is *not* a contribution we claim. We therefore write

$$\mathcal{L}_{\text{img}} = \lambda_{\text{VQ}}\mathcal{L}_{\text{VQ}} + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}} + \lambda_{\text{percept}}\mathcal{L}_{\text{percept}}, \tag{7}$$

where λ_{VQ}, λ_{GAN}, λ_{percept} are hyperparameters controlling influence of terms.

— $\mathcal{L}_{\text{VQ}}$. Following the original VQ-VAE[44] formulation we write

$$\begin{aligned}\mathcal{L}_{\text{VQ}} = \left\|\mathbf{f}^{t+1} - \hat{\mathbf{f}}^{t+1}\right\|_1 + \left\|\text{sg}\left[\mathcal{E}(\mathbf{f}^t)\right] - \mathbf{z_q}\right\|_2^2 \\ + \left\|\text{sg}\left[\mathbf{z_q}\right] - \mathcal{E}(\mathbf{f}^t)\right\|_2^2\end{aligned}, \tag{8}$$

where $\mathbf{f}^{t+1}$ and $\hat{\mathbf{f}}^{t+1}$ are true and estimated next frame, sg$[\cdot]$ denotes the stop gradient operation, and $\mathbf{z_q}$ is the quantized latent variable of the VQ-VAE. Note that we use the ℓ_1 norm, instead of the ℓ_2, for the reconstruction part of the loss, as we emprically found it to be more stable in training.

— $\mathcal{L}_{\text{GAN}}$. We train a discriminator $\mathcal{C}$ with the architecture from [25] and aim to improve the generation quality. We therefore write

$$\mathcal{L}_{\text{GAN}} = \log\left(\mathcal{C}\left(\mathbf{f}^{t+1}\right)\right) + \log\left(1 - \mathcal{C}\left(\hat{\mathbf{f}}^{t+1}\right)\right). \tag{9}$$

For the hyperparameter for this loss λ_{GAN}, we follow [15] and apply a dynamic weighing strategy, which stabilizes training.

— $\mathcal{L}_{\text{percept}}$. We use a pretrained VGG-16 network [48] to extract deep features and compute the perceptual loss. Denoting the deep feature extraction process as $\mathcal{V}$ we write

$$\mathcal{L}_{\text{percept}} = \left\|\mathcal{V}(\hat{\mathbf{f}}^{t+1}) - \mathcal{V}(\mathbf{f}^{t+1})\right\|_2. \tag{10}$$

3.2 Stage II: Fine-tuning for controllability

While the model trained in Section 3.1 is a generative model conditioned on the latent mask $\mathbf{m}^t$, it cannot be immediately used with any arbitrary mask – the generator $\mathcal{G}$ would expect a mask that aligns perfectly with $\mathbf{f}^t$, whereas our user edited mask $\mathbf{m}_c^t$ will not. In other words, we need a way to simulate user input, in terms of mask modifications, and incorporate it into the training. Therefore, we now discuss the second stage of our training setup, where we shift our focus to imbue our method with controllability.

We turn our attention to the assumption that the changes between the masks of two consecutive frames $\mathbf{m}^t$ and $\mathbf{m}^{t+1}$ can be approximated by a differentiable transformation function $\mathcal{T}(\cdot)$:

$$\mathbf{m}^{t+1} \approx \mathcal{T}\left(\mathbf{m}^t, \boldsymbol{\theta}^t\right). \tag{11}$$

In the crudest form, $\mathcal{T}(\cdot)$ can simply be shifting the mask $\mathbf{m}^t$ by Δx and Δy in horizontal and vertical directions, respectively, or for example, be an affine transformation. Both can be implemented differentiably as a parametric coordinate transformation on $\mathbf{m}^t$ [26,27]. We utilize this differentiabilty to find the "ground-truth" control signal $\hat{\boldsymbol{\theta}}^t$ (*a.k.a.*, the *pseudo user control*) using the following optimization procedure:

$$\hat{\boldsymbol{\theta}}^t \equiv \arg\min_{\boldsymbol{\theta}} \left\|\mathbf{m}^{t+1} - \mathcal{T}\left(\mathbf{m}^t, \boldsymbol{\theta}^t\right)\right\|. \tag{12}$$

Now, we can apply this control signal $\hat{\boldsymbol{\theta}}^t$ to the current mask $\mathbf{m}^t$ to obtain the *pseudo user-edited mask* using $\mathcal{T}(\cdot)$, and finetune the network which results in now *controllable* video generation:

$$\tilde{\mathbf{f}}^{t+1} = \mathcal{G}\left(\mathbf{f}^t, \left\lfloor \mathcal{T}\left(\mathbf{m}^t, \hat{\boldsymbol{\theta}}^t\right)\right\rfloor_{0.5}\right). \tag{13}$$

Where $\mathcal{G}$ denotes the frame generator, and $\lfloor\cdot\rfloor_{0.5}$ denoting the binarization operation. We then use $\tilde{\mathbf{f}}^{t+1}$ in our loss functions to fine-tune.

One noteworthy aspect of this second stage training is that, because we binarize the mask, no gradient flows through to $\mathcal{M}$. this leads to $\mathcal{L}_{\text{bg}}$, $\mathcal{L}_{\text{fg}}$, and $\mathcal{L}_{\text{bin}}$ not affecting training. While the latter two can be dropped since they are purely on how the mask network $\mathcal{M}$ behaves, completely dropping $\mathcal{L}_{\text{bg}}$ now has the danger of the generated image ignoring the mask. Hence, we replace $\mathbf{f}^{t+1}$ in Eq. (3) with $\tilde{\mathbf{f}}^{t+1}$, so that the generated image still obeys the mask conditioning. Hence, for the second stage training, instead of $\mathcal{L}_{\text{bg}}$, we utilize $\mathcal{L}'_{\text{bg}}$ where

$$\mathcal{L}'_{\text{bg}} = \left\|(1-\mathbf{m}^t)\odot\mathbf{f}^t - (1-\mathbf{m}^t)\odot\tilde{\mathbf{f}}^{t+1}\right\|_1, \tag{14}$$

which now enforces our fine-tuned generator $\mathcal{G}$ to still obey the provided mask.

3.3 Framework

As illustrated in Figure 2, our method is composed of mainly two components: the mask network and the generator network. The *mask network* $\mathcal{M}$ takes as input an image frame $\mathbf{f}^t \in \mathbb{R}^{3\times W\times H}$ at time t and outputs a mask:

$$\mathbf{m}^t = \mathcal{M}\left(\mathbf{f}^t\right), \tag{15}$$

where $\mathbf{m}^t \in \mathbb{R}^{W\times H}$, segmenting the foreground layer and the background layer. Then, our *generator network* $\mathcal{G}$ takes the mask and the frame as input to generate the next frame:

$$\hat{\mathbf{f}}^{t+1} = \mathcal{G}\left(\mathbf{f}^t, \mathbf{m}^t\right). \tag{16}$$

As shown in Figure 3, the generator $\mathcal{G}$ can be written as a composite function of an encoder $\mathcal{E}$, a decoder $\mathcal{D}$, and a learnable discrete code book $\mathcal{Z} = \{\mathbf{z}_k\}_{k=1}^K \subset \mathbb{R}^{n_z}$, where n_z is the dimension of each code, we encode the foreground layer and the background layer separately in our pipeline, and then merge them in the latent space before feeding them into the decoder:

$$\begin{aligned}
\mathbf{z}^t_{fg} &= \mathcal{E}\left(\mathbf{f}^t \odot \mathbf{m}^t\right) \\
\mathbf{z}^t_{bg} &= \mathcal{E}\left(\mathbf{f}^t \odot (1-\mathbf{m}^t)\right) \\
\hat{\mathbf{f}}^{t+1} &= \mathcal{D}\left(\mathsf{q}\left(\mathbf{z}^t_{fg}\right) + \mathsf{q}\left(\mathbf{z}^t_{bg}\right)\right)
\end{aligned}, \tag{17}$$

where $\odot$ is the element-wise product and q is the element-wise quantization function defined as:

$$\mathsf{q}(\mathbf{z})_{ij} := \underset{\mathbf{z}_k\in\mathcal{Z}}{\arg\min} \left\|\mathbf{z}_{ij} - \mathbf{z}_k\right\|, \tag{18}$$

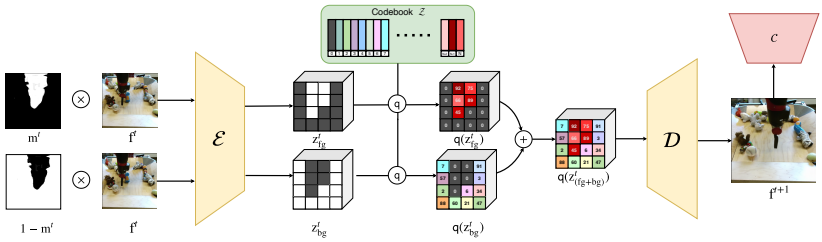

Fig. 3. Frame Generator $\mathcal{G}$. We employ VQGAN framework for the generator $\mathcal{G}$, comprising of an encoder $\mathcal{E}$, decoder $\mathcal{D}$, a learnable discrete codebook $\mathcal{Z}$ and a discriminator $\mathcal{C}$. We encode the foreground and the background layers separately and then merge them in the latent space before feeding them to the decoder $\mathcal{D}$ which generates the next frame in the sequence.

where i and j are the row and column indices. $\mathcal{G}$ and $\mathcal{M}$ are then used to generate the video via auto-regression.

With the above pipeline, during testing time, we enable on-the-fly user control by modifying the current frame mask $\mathbf{m}^t$ to create $\mathbf{m}^t_c$ and using it in place of $\mathbf{m}^t$ in Eq. (16). Mathematically, we write

$$\mathbf{m}^t_c = \mathcal{T}\left(\mathbf{m}^t, \boldsymbol{\theta}^t\right), \tag{19}$$

where $\mathcal{T}(\cdot)$ is the mask controlling operation described in Section 3.2. Other forms of control are also possible, including more granular non-parametric manipulation of the mask (see section "Action Mimicking" on our project website).

4 Experiments

While our method is *not* limited to a "single-agent" assumption, *i.e.*, single dominant moving agent in the scene, previous work, and notably [39] which is the closest and the most competitive baseline, are. Hence, for fair comparison, we adopt the single-agent setup for the majority of our experiments. We train / test on the following datasets:

BAIR Robot Pushing Dataset [13]. This dataset contains 44K video clips (256×256 resolution) of a single robot arm agent pushing toys on a flat surface.

Tennis Dataset [39]. This dataset contains 900 clips extracted from two full tennis matches on YouTube. These clips are cropped such that only the half of the court is visible. The resolution of each frame is (96×256).

4.1 Results

Evaluation Protocol. We compare our model against other conditional generative methods, focusing on quality of reconstructed sequences and controlability. We evaluate our model under three control protocols: two parametric (position, affine) and one non-parametric (direct non-differentiable control over mask):

– Ours /w `position` control: We first use our trained mask network $\mathcal{M}$ to extract masks from the ground truth test sequences, then we use Eq. (19) to

approximate those masks with control for generation, where θ is restricted to `positional` parameters, *i.e.*, x and y translation.
- Ours /w `affine` control: Similar to above, here our θ employs full `affine` transformation parameters, *i.e.*, translation, rotation, scaling and shearing.
- Ours /w `non-param` control: We use masks predicted from ground truth test sequences themselves to condition our generation. These masks can change at a pixel-level, hence constituting non-parametric control.

For testing, we generate video sequences conditioned on the first frame f_0 and the user input $\Theta = \{\theta_u^t\}_{t=1}^T$ in all cases.

Metrics. To quantitatively evaluate our results, we consider standard metrics:

- *Learned Perceptual Image Patch Similarity (LPIPS)* [71]: LPIPS measures the perceptual distance between generated and ground truth frames.
- *Fréchet Inception Distance (FID)* [23]: FID calculates the Fréchet distance between multivariate Gaussians fitted to the feature space of the Inception-v3 network of generated and ground truth frames.
- *Fréchet Video Distance (FVD)* [53]: FVD extends FID to the video domain. In addition to the quality of each frame, FVD also evaluates the temporal coherence between generated and ground truth sequences.
- *Average Detection Distance (ADD)* [39]: ADD first uses Faster-RCNN [47] to detect the target object in both generated and ground truth frames, then calculates the Euclidean distance between the bound box centers.
- *Missing Detection Rate (MDR)* [39]: MDR reports percentage of unsuccessful detections in generated vs. successful detections in ground truth sequences.
- *Rooted Mean Square Error of Displacement (RMSED)* [39]: RMSED, which we define, reports the RMSE of the displacement of ground truth locations vs. generated locations. See Figure 4 for more details.

LIPIPS, FID and FVD measure the quality of generated videos. ADD and MDR measure how the action label conditions the generated video, and RMSED measures the precision of control.

Baselines. CADDY [39] is the only unsupervised video generation method that allows frame-level user conditioning, thus we use it as our main baseline. We also include results of other frame-level conditioned methods: MoCoGAN [52], SAVP [36], and their high-resolution adaptations MoCoGAN+ and SAVP+ from [39].

Table 1. Results on the *BAIR* Dataset

Method	LPIPS ↓	FID ↓	FVD ↓	RMSED ↓
MoCoGAN [52]	0.466	198	1380	-
MoCoGAN+ (from [39])	0.201	66.1	849	0.211
SAVP [36]	0.433	220	1720	-
SAVP+ (from [39])	**0.154**	**27.2**	303	0.109
CADDY [39]	0.202	35.9	423	0.132
Ours /w `position` control	0.202	28.5	333	0.059
Ours /w `affine` control	0.201	30.1	**292**	0.035
Ours /w `non-param` control	0.176	29.3	293	**0.021**

Table 2. Results on the *Tennis* Dataset

Method	LPIPS ↓	FID ↓	FVD ↓	ADD ↓	MDR ↓
MoCoGAN [52]	0.266	132	3400	28.5	20.2
MoCoGAN+ (from [39])	0.166	56.8	1410	48.2	27.0
SAVP [36]	0.245	156	3270	10.7	19.7
SAVP+ (from [39])	0.104	25.2	223	13.4	19.2
CADDY [39]	0.102	13.7	239	8.85	1.01
Ours /w `position` control	0.122	10.1	215	4.30	**0.300**
Ours /w `affine` control	0.115	11.2	207	3.40	0.317
Ours /w `non-param` control	**0.100**	**8.68**	**204**	**1.76**	0.306

Quantitative Results. We report the results on the *BAIR* dataset in Table 1. We highlight that in terms of RMSED score, our method achieved the highest precision of control (more than $\times 5$ improvement compared to other baselines). In terms of generation quality, with similar level of abstraction of ground truth

information (ours: 6 continuous `affine` control parameters, CADDY: 7 discrete action labels), our model outperformed CADDY on all three evaluated metrics by a large margin, demonstrating that our model is of better generation quality. With `non-param` control, our generation quality is comparable to the SAVP+.

Table 2 shows the results on the *Tennis* dataset. In terms of generation quality (LPIPS, FID, FVD), all our adaptations outperformed all other comparing methods. Specifically, in terms of FID score, our model is up to 37% better than the closest baseline ([39]). Further, in terms of control precision, our method achieves the lowest error on ADD and MDR (improvement of 80% & 70% respectively), indicating our method is able to generate consistent players with accurate control. One can see that simple positional control works much better here compared to the *BAIR* dataset. This can be attributed to largely in plane motion of the subject.

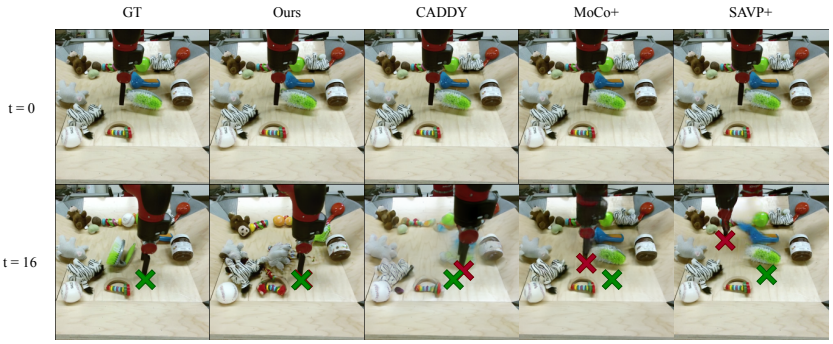

Fig. 4. Qualitative Results On the BAIR Dataset. We labelled the robot arm positions for half of the testing sequences (128 videos out of 256) from both generated videos and ground truth videos. As illustrated on the figure: GT location is marked green and generated locations are marked red. We calculate RMSE of the displacement between GT locations and generated locations to arrive at the RMSED score.

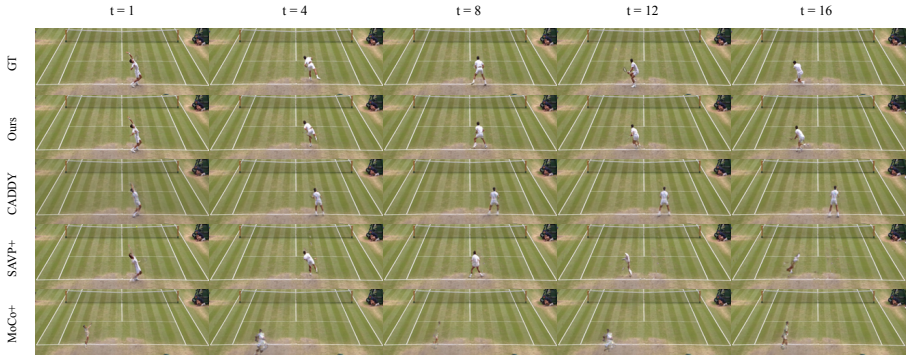

Fig. 5. Qualitative Results On the Tennis dataset.

Qualitative Results. In Figure 4 and Figure 5, we show generated sequences on the *BAIR* and *Tennis* dataset (we used our model with `affine` control for both

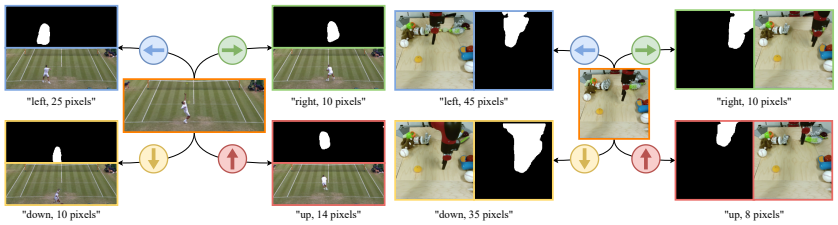

Fig. 6. Effectiveness of Control. Illustrated is how our model precisely reacts to different controlling signals starting from the same initial frame. We illustrate position parameters; however, other affine control parameters are also possible (*e.g.*, scale, rotation and shear).

cases shown). In terms of image quality, our method is superior to competitors. In terms of control accuracy, unlike other competing methods, our method is able to precisely place the robot arm and the tennis player in the correct position.

In Figure 6, we show the results of our model reacting to different user control signals. On the *Tennis* dataset, our method not only moves the player in the correct direction, it's also able to generate plausible motions of the player itself. On the *BAIR* dataset, our model is able to "hullcinate" what's missing in the original frame and generate frames with respect to the control signal, *i.e.*, in the "down, 35 pixels" example, our model successfully generates the upper part of the robot arm, not available on the input frame.

In Figure 7, we show that our method is capable of generating and controlling videos with multiple moving objects by simply overlaying two individually controlled mask sequences together, *i.e.*, producing 2 and 3 players in this example. As far as we know, our method is the only video generation method that allows frame-level control of multiple objects acting in the same scene. We provide more visual results in the *Supp. Mat.*

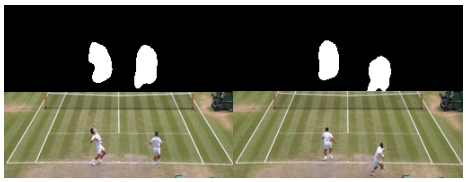

Fig. 7. Control of Multiple Agents. Our method is able to generate videos with multiple moving objects that can be controlled individually by their respective masks.

4.2 Ablation Study

Mask losses. Here we explore impact of our key design choices have on the quality of generated results and the foreground mask. We show quantitative and qualitative results in Figure 8. The background loss $\mathcal{L}_{\text{bg}}$ enforces the network to generate meaningful masks, without it, the mask network fails to generate a reasonable mask (all zeros). The foreground constraint $\mathcal{L}_{\text{fg}}$ shrinks the mask as much as possible. Without this term the network learns a travail solution, where $\mathcal{L}_{\text{bg}}$ in Eq.(3) becomes 0 – labeling everything as the foreground (all ones). When computing the foreground loss $\mathcal{L}_{\text{fg}}$, we introduce a dynamic mask size prior. We ablate this choice by instead using a fixed global prior of 0.15 as in [8]. Visuals show that if we do not use dynamic prior, the network tends to generate masks

with a fixed size, which leads to hollow masks for samples with larger foreground. To prevent information leaking from soft masks, we binarize the masks with thresholding the mask value, without the binary loss $\mathcal{L}_{\text{bin}}$, some pixels on the mask fails to pass the threshold and leave some defects on the binarized mask. Overall, the ablations show that all our design choices are important.

One-stage training VS. Two-stage training. Breaking our training procedure into two stages is a crucial design for the performance of our method. As described in Eq. (12), a well-trained mask generator is a prerequisite for finding the *pseudo user control* $\hat{\boldsymbol{\theta}}$, which we use to introduce controllability to our model. Nevertheless, we still experimented with training the model with one single shot (training Stage II directly by replacing $\mathbf{f}^t$ with $\mathbf{f}^{t+1}$). This leads to vastly poorer performance during test time (Figure 8, "single-stage training").

Method	LPIPS ↓	FID ↓	FVD ↓
w/o $\mathcal{L}_{fg}$	0.333	60.1	816
w/o $\mathcal{L}_{bg}$	0.306	97.1	796
w/o $\mathcal{L}_{bin}$	0.222	59.2	398
w/o mask prior	0.208	55.0	**279**
single-stage training	0.608	302.3	6614
full	**0.176**	**29.3**	293

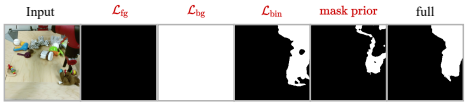

Fig. 8. Ablation of Design Choices. We ablate various loss terms, the use of dynamic mask prior and the two-stage training design. Thumbnails below illustrate the effect these components have on the estimated mask itself; full model producing the most coherent mask.

5 Conclusions

We have introduced layered controllable video generation, an unsupervised method that decomposes frames into foreground and background, with which the user can control the generative process at a frame-level by altering the foreground mask. Our core contributions are the framework itself, and the two-stage training strategy that allows our model to learn to both separate and control on its own. We show that various degrees of control can be implemented with our method, from parametric (position, affine) to complete non-parametric control with the mask. Our results on *BAIR* and *Tennis* datasets show that our method outperforms the state-of-the-art in both quality and control.

6 Acknowledgements

This work was funded, in part, by the Vector Institute for AI, Canada CIFAR AI Chair, NSERC CRC and an NSERC Discovery and Discovery Accelerator Grants. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute www.vectorinstitute.ai/#partners. Additional hardware support was provided by John R. Evans Leaders Fund CFI grant and Compute Canada under the Resource Allocation Competition awards of the two PIs. We would also like to thank Willi Menapace for his help in answering our questions and helping with fair comparisons to [39].

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P.V., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods **34**, 2274–2282 (2012) 5
2. Acharya, D., Huang, Z., Paudel, D.P., Gool, L.V.: Towards high resolution video generation with progressive growing of sliced wasserstein gans. ArXiv preprint (2018) 3
3. Alayrac, J.B., Carreira, J., Arandjelovic, R., Zisserman, A.: Controllable attention for structured layered video decomposition. In: Int. Conf. Comput. Vis. (2019) 5
4. Alayrac, J.B., Carreira, J., Zisserman, A.: The visual centrifuge: Model-free layered video representations. In: IEEE Conf. Comput. Vis. Pattern Recog. (2019) 5
5. Alexe, B., Deselaers, T., Ferrari, V.: Classcut for unsupervised class segmentation. In: Eur. Conf. Comput. Vis. (2010) 5
6. Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R.H., Levine, S.: Stochastic variational video prediction (2018) 3, 4
7. Bao, H., Dong, L., Wei, F.: Beit: Bert pre-training of image transformers. arXiv preprint arXiv:2106.08254 (2021) 5
8. Bielski, A., Favaro, P.: Emergence of object segmentation in perturbed generative models. In: Adv. Neural Inform. Process. Syst. (2019) 5, 7, 13
9. Bouwmans, T., Porikli, F., Höferlin, B., Vacavant, A.: Background modeling and foreground detection for video surveillance. CRC press (2014) 5
10. Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., Sutskever, I.: Generative pretraining from pixels. In: International Conference on Machine Learning (ICML). pp. 1691–1703 (2020) 5
11. Chiappa, S., Racanière, S., Wierstra, D., Mohamed, S.: Recurrent environment simulators. In: Int. Conf. Learn. Represent. (2017) 4
12. Denton, E.L., Fergus, R.: Stochastic video generation with a learned prior. In: Int. Conf. Mach. Learn. (2018) 1, 3, 4
13. Ebert, F., Finn, C., Lee, A.X., Levine, S.: Self-supervised visual planning with temporal skip connections. In: Conf. Robot Learn. (2017) 10
14. Elgammal, A., Harwood, D., Davis, L.: Non-parametric model for background subtraction. In: Eur. Conf. Comput. Vis. (2000) 5
15. Esser, P., Rombach, R., Ommer, B.: Taming Transformers for High-Resolution Image Synthesis. In: IEEE Conf. Comput. Vis. Pattern Recog. (2021) 7, 8, 2
16. Finn, C., Goodfellow, I., Levine, S.: Unsupervised learning for physical interaction through video prediction. In: Adv. Neural Inform. Process. Syst. (2016) 1, 4
17. Franceschi, J.Y., Delasalles, E., Chen, M., Lamprier, S., Gallinari, P.: Stochastic latent residual video prediction. In: Int. Conf. Mach. Learn. (2020) 1
18. Frolov, S., Sharma, A., Hees, J., amd Federico Raue, T.K., Dengel, A.: Attrlostgan: Attribute controlled image synthesis from reconfigurable layout and style. In: Ger. Conf. Pattern Recog. (2021) 1
19. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Adv. Neural Inform. Process. Syst. (2014) 1
20. Goyette, N., Jodoin, P.M., Porikli, F., Konrad, J., Ishwar, P.: changedetection.net: A New Change Detection Benchmark Dataset. In: IEEE Conf. Comput. Vis. Pattern Recog. (2012) 5
21. Hao, Z., Huang, X., Belongie, S.: Controllable video generation with sparse trajectories. In: IEEE Conf. Comput. Vis. Pattern Recog. (2018) 3, 4

22. He, J., Lehrmann, A., Marino, J., Marino, J., Sigal, L.: Probabilistic video generation using holistic attribute control. In: Eur. Conf. Comput. Vis. (2018) 2, 4
23. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Adv. Neural Inform. Process. Syst. (2017) 11
24. Hochbaum, D.S., Singh, V.: An efficient algorithm for co-segmentation. In: ICCV. pp. 269–276 (2009) 5
25. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks (2017) 8, 2
26. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: Adv. Neural Inform. Process. Syst. (2015) 8
27. Jiang, W., Sun, W., Tagliasacchi, A., Trulls, E., Yi, K.M.: Linearized Multi-Sampling for Differentiable Image Transformation. In: IEEE Conf. Comput. Vis. Pattern Recog. (2019) 8
28. Jojic, N., Frey, B.J.: Learning flexible sprites in video layers. In: IEEE Conf. Comput. Vis. Pattern Recog. (2001) 2, 5
29. Kalchbrenner, N., van den Oord, A., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., Kavukcuoglu, K.: Video pixel networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 1771–1779. PMLR (06–11 Aug 2017) 3
30. Kim, S.W., Zhou, Y., Philion, J., Torralba, A., Fidler, S.: Learning to simulate dynamic environments with gamegan. In: IEEE Conf. Comput. Vis. Pattern Recog. (2020) 2, 4
31. Kim, Y., Nam, S., Cho, I., Kim, S.J.: Unsupervised keypoint learning for guiding class-conditional video prediction. In: Adv. Neural Inform. Process. Syst. (2019) 4
32. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Int. Conf. Learn. Represent. (2013) 1
33. Kumar, M.P., Torr, P.H., Zisserman, A.: Learning layered motion segmentations of video (2008) 2, 5
34. Kumar, M., nd Dumitru Erhan, M.B., Finn, C., Levine, S., Dinh, L., Kingma, D.: Videoflow: A conditional flow-based model for stochastic video generation. In: Int. Conf. Learn. Represent. (2020) 1
35. Kwon, Y.H., Park, M.G.: Predicting future frames using retrospective cycle gan (2019) 4
36. Lee, A.X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., Levine, S.: Stochastic adversarial video prediction. ArXiv preprint (2018) 4, 11
37. Lu, J., Batra, D., Parikh, D., Lee, S.: VilBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In: Conference on Neural Information Processing Systems (NeurIPS) (2019) 5
38. Mathieu, M., Couprie, C., LeCun, Y.: Deep multi-scale video prediction beyond mean square error. In: Int. Conf. Learn. Represent. (2016) 1, 4
39. Menapace, W., Lathuilière, S., Tulyakov, S., Siarohin, A., Ricci, E.: Playable Video Generation. In: IEEE Conf. Comput. Vis. Pattern Recog. (2021) 2, 3, 4, 10, 11, 12, 14
40. Nawhal, M., Zhai, M., Lehrmann, A., Sigal, L., Mori, G.: Generating videos of zero-shot compositions of actions and objects. In: Eur. Conf. Comput. Vis. (2020) 2, 4
41. Neimark, D., Bar, O., Zohar, M., Asselmann, D.: Video transformer network. ArXiv preprint (2021) 3

42. Oh, J., Guo, X., Lee, H., Lewis, R.L., Singh, S.: Action-conditional video prediction using deep networks in atari games. In: Adv. Neural Inform. Process. Syst. (2015) 4
43. van den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. In: Int. Conf. Mach. Learn. (2016) 1
44. van den Oord, A., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. In: Adv. Neural Inform. Process. Syst. (2017) 3, 8
45. Pan, J., Wang, C., Jia, X., Shao, J., Sheng, L., Yan, J., Wang, X.: Video Generation From Single Semantic Label Map. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3728–3737 (06 2019) 4
46. Rakhimov, R., Volkhonskiy, D., Artemov, A., Zorin, D., Burnaev, E.: Latent video transformer. In: Int. Joint Conf. Comput. Vis. Imag. Comput. Graph. Theory Appl. (2021) 3
47. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**, 1137–1149 (2015) 11
48. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2015) 8
49. Smirnov, D., Gharbi, M., Fisher, M., Guizilini, V., Efros, A.A., Solomon, J.: Marionette: Self-supervised sprite learning. In: Adv. Neural Inform. Process. Syst. (2021) 5
50. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: IEEE Conf. Comput. Vis. Pattern Recog. (1999) 5
51. Su, W., Zhu, X., Cao, Y., Li, B., Lu, L., Wei, F., Dai, J.: VL-BERT: Pre-training of generic visual-linguistic representations. In: International Conference on Learning Representations (ICLR) (2020) 5
52. Tulyakov, S., Liu, M.Y., Yang, X., Kautz, J.: Mocogan: Decomposing motion and content for video generation. In: IEEE Conf. Comput. Vis. Pattern Recog. (2018) 2, 11
53. Unterthiner, T., van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., Gelly, S.: Towards accurate generative models of video: A new metric & challenges. ArXiv preprint (2018) 11
54. Villegas, R., Yang, J., Zou, Y., Sohn, S., Lin, X., Lee, H.: Learning to generate long-term future via hierarchical prediction. In: Int. Conf. Mach. Learn. (2017) 2, 4
55. Vondrick, C., Pirsiavash, H., Torralba, A.: Generating videos with scene dynamics. In: Adv. Neural Inform. Process. Syst. (2016) 1, 3, 5
56. Vondrick, C., Pirsiavash, H., Torralba, A.: Anticipating the future by watching unlabeled video. In: Int. Conf. Learn. Represent. (2017) 1, 4
57. Vondrick, C., Torralba, A.: Generating the future with adversarial transformers. In: IEEE Conf. Comput. Vis. Pattern Recog. (2017) 1
58. Walker, J., Marino, K., Gupta, A., Hebert, M.: The pose knows: Video forecasting by generating pose futures. In: Int. Conf. Comput. Vis. (2017) 2, 4
59. Wang, J.Y., Adelson, E.H.: Representing Moving Images with Layers (1994) 2, 5
60. Wang, T.C., Liu, M.Y., Zhu, J.Y., Liu, G., Tao, A., Kautz, J., Catanzaro, B.: Video-to-video synthesis. In: Adv. Neural Inform. Process. Syst. (2019) 4
61. Wang, Y., Bilinski, P.T., Brémond, F., Dantcheva, A.: G3an: Disentangling appearance and motion for video generation (2020) 3
62. Wang, Y., Bilinski, P.T., Brémond, F., Dantcheva, A.: Imaginator: Conditional spatio-temporal gan for video generation. In: IEEE Winter Conf. Appl. Comput. Vis. (2020) 4

63. Webster, M.A.: Color vision: Appearance is a many-layered thing. In: Current Biology (2009) 5
64. Weissenborn, D., Tackstrom, O., Uszkoreit, J.: Scaling autoregressive video models. In: Int. Conf. Learn. Represent. (2020) 1, 3
65. Wichers, N., Villegas, R., Erhan, D., Lee, H.: Hierarchical long-term video prediction without supervision. In: ICML (2018) 4
66. Xue, T., Wu, J., Bouman, K.L., Freeman, W.T.: Visual dynamics: Stochastic future generation via layered cross convolutional networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**, 2236–2250 (2019) 4
67. Yan, W., Zhang, Y., Abbeel, P., Srinivas, A.: Videogpt: Video generation using vq-vae and transformers. ArXiv preprint (2021) 3
68. Yang, C., Wang, Z., Zhu, X., Huang, C., Shi, J., Lin, D.: Pose guided human video generation. In: Eur. Conf. Comput. Vis. (2018) 2, 4
69. Zablotskaia, P., Siarohin, A., Zhao, B., Sigal, L.: Dwnet: Dense warp-based network for pose-guided human video generation. In: Brit. Mach. Vis. Conf. (2019) 2, 4
70. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: Int. Conf. Comput. Vis. (2017) 1
71. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: IEEE Conf. Comput. Vis. Pattern Recog. (2018) 11
72. Zhao, B., Meng, L., Yin, W., Sigal, L.: Image generation from layout. In: IEEE Conf. Comput. Vis. Pattern Recog. (2019) 1