Diffusion Models for Video Prediction and Infilling

Tobias Höppe *tobihoeppe@gmail.com*
KTH Stockholm

Arash Mehrjou[*]
MPI for Intelligent Systems & ETH Zürich

Stefan Bauer[*]
KTH Stockholm

Didrik Nielsen[*]
Norwegian Computing Center

Andrea Dittadi[*] *adit@dtu.dk*
Technical University of Denmark & MPI for Intelligent Systems

Reviewed on OpenReview: *https://openreview.net/forum?id=lf0lr4AYM6*

Abstract

Predicting and anticipating future outcomes or reasoning about missing information in a sequence are critical skills for agents to be able to make intelligent decisions. This requires strong, temporally coherent generative capabilities. Diffusion models have shown remarkable success in several generative tasks, but have not been extensively explored in the video domain. We present Random-Mask Video Diffusion (RaMViD), which extends image diffusion models to videos using 3D convolutions, and introduces a new conditioning technique during training. By varying the mask we condition on, the model is able to perform video prediction, infilling, and upsampling. Due to our simple conditioning scheme, we can utilize the same architecture as used for unconditional training, which allows us to train the model in a conditional and unconditional fashion at the same time. We evaluate RaMViD on two benchmark datasets for video prediction, on which we achieve state-of-the-art results, and one for video generation. High-resolution videos are provided at `https://sites.google.com/view/video-diffusion-prediction`.

1 Introduction

Videos contain rich information about the world, and a vast amount of diverse video data is available. Training models that understand this data can be crucial for developing agents that interact with the surrounding world effectively. In particular, video prediction plays an increasingly important role: autonomous driving (Hu et al., 2020), anticipating events (Zeng et al., 2017), planning (Finn & Levine, 2017) and reinforcement learning (Hafner et al., 2019) are applications which can benefit from increasing performance of prediction models. On the other hand, video infilling—i.e., observing a part of a video and generating missing frames—can be used for example in planning, estimating trajectories, and video processing. In addition, video models can be valuable for downstream tasks such as action recognition (Kong & Fu, 2018) and pose estimation (Sahin et al., 2020). However, there has not been extensive research on video infilling and most research is focusing on generation or prediction.

Most recent approaches to video prediction are based on variational autoencoders (Babaeizadeh et al., 2021; Saxena et al., 2021) or GANs (Clark et al., 2019; Luc et al., 2020). Diffusion models (Sohl-Dickstein et al.,

[*]Equal advising.

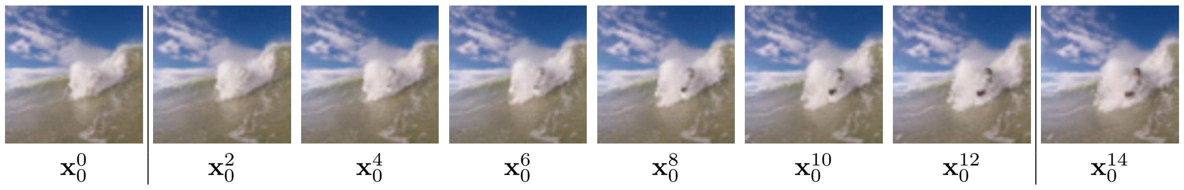

Figure 1: The first two and last two frames of a video are given and our model does fill in the missing frames very accurate and with much detail.

2015; Ho et al., 2020; Nichol & Dhariwal, 2021; Song et al., 2021; Abstreiter et al., 2021; Mittal et al., 2022; Dockhorn et al., 2021) have recently seen tremendous progress on static visual data, even outperforming GANs in image synthesis (Dhariwal & Nichol, 2021), but have not yet been extensively studied for videos. Considering their impressive performance on images, it is reasonable to believe that diffusion models may also be useful for tasks in the video domain.

In this paper, we extend diffusion models to the video domain via several technical contributions. We use 3D convolutions and a new conditioning procedure incorporating randomness. Our model is not only able to predict future frames of a video but also fill in missing frames at arbitrary positions in the sequence (see Fig. 1). Therefore, our Random-Mask Video Diffusion (RaMViD) can be used for several video completion tasks. We summarize our technical contributions as follows:[1]

- A novel diffusion-based architecture for video prediction and infilling.
- Competitive performance with recent approaches across multiple datasets.
- Introduce a schedule for the random masking.

The remainder of this paper is organized as follows: In Section 2, we provide the necessary background on diffusion models and video prediction and outline relevant related work. Section 3 describes Random-Mask Video Diffusion (RaMViD). In Section 4, we present and discuss extensive experiments on several benchmark datasets. We finally conclude with a discussion in Section 5.

2 Background and related work

Diffusion models. Diffusion-based models generally refer to the class of machine learning algorithms that consist of gradually transforming a complex distribution into unstructured noise and learning to reverse this process to recover the data generating distribution. They have attracted a great deal of attention after being successfully applied to a diverse range of tasks such as image generation (Song et al., 2021; Niu et al., 2020), audio (Chen et al., 2021), graph and shape generation (Cai et al., 2020). The essence of these models is two stochastic (diffusion) processes implemented by Stochastic Differential Equations (SDEs), a forward and a backward one. We explain the formulation in the abstract domain here and specialize it later according to the application of this work.

Let $\mathbf{x}_0 \in \mathbb{R}^d$ be a sample from the empirical data distribution, i.e., $\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x}_0)$ and d be the data dimension. The forward diffusion process takes $\mathbf{x}_0$ as the starting point and creates the random trajectory $\mathbf{x}_{[0,T]}$ from $t = 0$ to the final time $t = T$. The forward process is designed such that $p(\mathbf{x}_T \mid \mathbf{x}_0)$ has a simple unstructured distribution. One example of such SDEs is

$$d\mathbf{x}_t = f(\mathbf{x}_t, t)dt + g(t)dw := \sqrt{\frac{d[\sigma^2(t)]}{dt}}dw \ , \tag{1}$$

where w is the Brownian motion. A desirable property of this process is the fact that the conditional distribution $p(\mathbf{x}_t \mid \mathbf{x}_0)$ takes a simple analytical form:

$$p(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \mathbf{x}_0, \left(\sigma^2(t) - \sigma^2(0)\right)\mathbf{I}\right) . \tag{2}$$

[1]Code is available at `https://github.com/Tobi-r9/RaMViD`.

Upon learning the gradient of $p(\mathbf{x}_t)$ for each t, one can reverse the above process and obtain the complex data distribution from pure noise as

$$d\mathbf{x}_t = [f(\mathbf{x}_t, t) - g^2(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}_t)]dt + g(t)dw' \ , \tag{3}$$

where w' is a Brownian motion independent of the one in the forward direction. Hence, generating samples from the data distribution boils down to learning $\nabla_{\mathbf{x}} \log p(\mathbf{x})$.

The original score matching objective (Hyvärinen & Dayan, 2005):

$$\mathbb{E}_{\mathbf{x}_t} \left[\| s_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) \|_2^2 \right] \tag{4}$$

is the most intuitive way to learn the score function, but is unfortunately intractable. Denoising Score Matching (DSM) provides a tractable alternative objective function:

$$J_t^{DSM}(\theta) = \mathbb{E}_{\mathbf{x}_0} \mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0} \left[\| s_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t \mid \mathbf{x}_0) \|_2^2 \right] \tag{5}$$

whose equivalence with the original score matching objective was shown by Vincent (2011) and used to train energy models by Saremi et al. (2018). Similarly to many recent works, we use the DSM formulation of score matching to learn the score function.

Video prediction and infilling. Research in video prediction has received more attention in the previous years, as the ability to predict videos can be used for several downstream tasks (Oprea et al., 2020). Video prediction can be modeled in a deterministic or stochastic form. Deterministic modeling (Walker et al., 2015; Vondrick & Torralba, 2017; Terwilliger et al., 2019; Sun et al., 2019) tries to predict the most likely future, but this often leads to averaging the future states (Li et al., 2019). Due to the stochastic nature of the future, generative models have lately shown to be more successful in capturing the underlying dynamics. For this approach, variational models are often used by modeling the stochastic content in a latent variable (Babaeizadeh et al., 2018; Saxena et al., 2021; Denton & Fergus, 2018; Wu et al., 2021a). However, this often leads to blurry predictions due to underfitting, and Babaeizadeh et al. (2021) have overcome this problem via architectural novelties. Blurry prediction is a less serious problem in GANs and promising results have been achieved especially on large datasets (Clark et al., 2019; Luc et al., 2020). On the other hand, the body of research on video infilling is significantly more scarce, with most works in this area focusing on frame interpolation (Jiang et al., 2018). However, Xu et al. (2018) have shown interesting results in infilling, by modeling the video as a stochastic generation process.

Concurrent work. Yang et al. (2022) is the only work so far that has used diffusion models for autoregressive video prediction, by modeling residuals for a predicted frame. However, since their evaluation procedure and datasets are different, a comparison with their work is not possible. A few concurrent works have recently considered diffusion models for video generation. Ho et al. (2022) focus on unconditional video generation, Harvey et al. (2022) use diffusion models to predict long videos, and Voleti et al. (2022), the most similar to our work, also consider video prediction and infilling.

3 Random-Mask Video Diffusion

Our method, Random-Mask Video Diffusion (RaMViD), consists of two main features. First, the way we introduce conditional information is different from prior work. Second, by randomizing the mask, we can directly use the same approach for video prediction and video completion (infilling). In the following, we detail each of these aspects of RaMViD.

3.1 Conditional training

Let $\mathbf{x}_0 \in \mathbb{R}^{L,W,H,C}$ be a video with length L. We partition the video $\mathbf{x}_0$ into two parts: the unknown frames $\mathbf{x}_0^{\mathcal{U}} \in \mathbb{R}^{L-k,W,H,C}$ and the conditioning frames $\mathbf{x}_0^{\mathcal{C}} \in \mathbb{R}^{k,W,H,C}$, where $\mathcal{U}$ and $\mathcal{C}$ are sets of indices such that $\mathcal{U} \cap \mathcal{C} = \varnothing$ and $\mathcal{U} \cup \mathcal{C} = \{0, 1, \ldots, L-1\}$. We write $\mathbf{x}_0 = \mathbf{x}_0^{\mathcal{U}} \oplus \mathbf{x}_0^{\mathcal{C}}$ with the following definition for $\oplus$:

$$(\mathbf{a}^{\mathcal{U}} \oplus \mathbf{b}^{\mathcal{C}})^i := \begin{cases} \mathbf{a}^i \text{ if } i \in \mathcal{U} \\ \mathbf{b}^i \text{ if } i \in \mathcal{C} \end{cases} \tag{6}$$

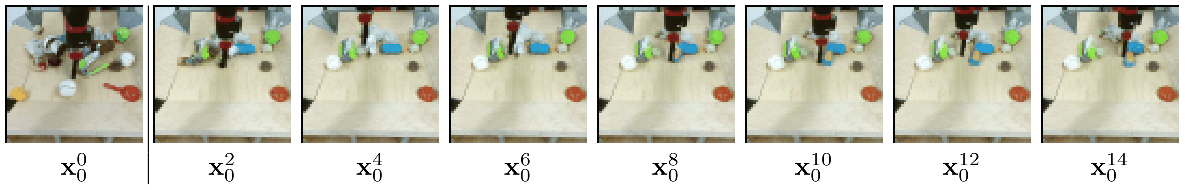

Figure 2: An unconditionally trained model is used to predict 15 frames given one frame. Even with re-sampling we can see, that objects in the background are not harmonized between the predicted and conditioned frames.

where the superscript i indicates tensor indexing and in our case corresponds to selecting a frame from a video. Here, t indicates the diffusion step, with $t = 0$ corresponding to the data and $t = T$ to the prior Gaussian distribution.

If we use an unconditionally trained model, we find that the predicted unknown frames $\mathbf{x}_0^{\mathcal{U}}$ do not harmonize well with the conditioning frames $\mathbf{x}_0^{\mathcal{C}}$, as shown in Fig. 2. One solution for this would be re-sampling, as proposed by Lugmayr et al. (2022). In re-sampling, we take one step in the learned reversed diffusion (denoising) process and then go back by taking a step in the forward diffusion process (i.e., adding noise again). This is repeated several times for each diffusion step, to make sure the predicted and conditioning frames are harmonized. However, this becomes computationally too expensive for videos, especially when using very few conditioning frames, as the number of re-sampling steps needs to be increased. To mitigate this issue, we propose to train the model conditionally with *randomized masking*.

Conditional diffusion models usually optimize

$$\mathbb{E}_{\mathbf{x}_0}\left\{\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\left[\left\|s_\theta(\mathbf{x}_t,\mathbf{x}_0^{\mathcal{C}},t)-\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t\,|\,\mathbf{x}_0)\right\|_2^2\right]\right\} \tag{7}$$

where $\mathbf{x}_0^{\mathcal{C}}$ is typically given as a separate input through an additional layer (Chen et al., 2021) or it is concatenated with the input (Saharia et al., 2021b; Batzolis et al., 2021; Saharia et al., 2021a). On the other hand, we feed the entire sequence to the network s_θ but only add noise to the unmasked frames: $\mathbf{x}_t^{\mathcal{U}} \sim \mathcal{N}\left(\mathbf{x}_0^{\mathcal{U}}, \left(\sigma^2(t)-\sigma^2(0)\right)\mathbf{I}\right)$. The input to the network is then a video where some frames are noisy and some are clean: $\mathbf{x}_t = \mathbf{x}_t^{\mathcal{U}} \oplus \mathbf{x}_0^{\mathcal{C}}$ (see Fig. 3). The loss is computed only with respect to $\mathbf{x}_t^{\mathcal{U}}$:

$$J_t^{\mathrm{RaMViD}}(\theta)=\mathbb{E}_{\mathbf{x}_0}\left\{\mathbb{E}_{\mathbf{x}_t^{\mathcal{U}}|\mathbf{x}_0}\left[\left\|s_\theta(\mathbf{x}_t,t)^{\mathcal{U}}-\nabla_{\mathbf{x}_t^{\mathcal{U}}}\log p(\mathbf{x}_t^{\mathcal{U}}\,|\,\mathbf{x}_0)\right\|_2^2\right]\right\}. \tag{8}$$

where $s_\theta(\mathbf{x}_t,t)^{\mathcal{U}}$ represents the output of the model with indices in $\mathcal{U}$. Note that the score function $\nabla_{\mathbf{x}_t^{\mathcal{U}}}\log p(\mathbf{x}_t^{\mathcal{U}}\,|\,\mathbf{x}_0)$ has the same dimension as $\mathbf{x}_t^{\mathcal{U}}$, whereas in Eq. (7) it has the dimension of the entire video $\mathbf{x}_t$. This leads to the forward diffusion process:

$$d\mathbf{x}_t^{\mathcal{U}} = f(\mathbf{x}_t^{\mathcal{U}},t)dt + g(t)dw \tag{9}$$

and the reversed diffusion process then becomes:

$$d\mathbf{x}_t^{\mathcal{U}} = [f(\mathbf{x}_t^{\mathcal{U}},t) - g^2(t)\nabla_{\mathbf{x}_t^{\mathcal{U}}}\log p(\mathbf{x}_t^{\mathcal{U}}|\mathbf{x}_0^{\mathcal{C}})]dt + g(t)dw' \;. \tag{10}$$

Similarly, Tashiro et al. (2021) compute the loss only on the unknown input. However, they also use concatenation and zero-padding to bring $\mathbf{x}_t^{\mathcal{C}}$ and $\mathbf{x}_t$ to the same dimension. For a more detailed schematic, see Appendix A. In our implementation, we used a discrete diffusion process with $t \in \{0, 1, \dots, T-1, T\}$.

3.2 Randomization

As previously mentioned, the proposed model is able to perform several tasks. We achieve this by sampling $\mathcal{C}$ at random. At each training step, we first choose the number of conditioning frames $|\mathcal{C}| = k \in \{1, \dots, K\}$, where K is a chosen hyperparameter. Then we define $\mathcal{C}$ by selecting k random indices from $\{0, \dots, L-1\}$, and we refrain from applying the diffusion process to the corresponding frames. Since the videos now consist

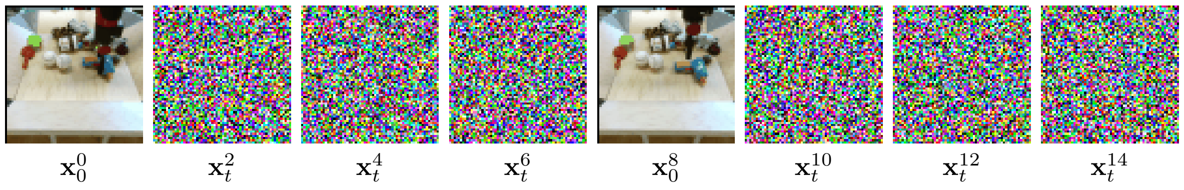

Figure 3: Example input of the network with $\mathcal{C} = \{0, 8\}$.

of original and noisy frames in varying positions, the model has to learn to distinguish between them in order to use the frames $\mathbf{x}_0^{\mathcal{C}}$ as information for the reversed diffusion process. After training, we can use RaMViD by fixing $\mathcal{C}$ to the set of indices of the known frames ($\mathcal{C}$ can be any arbitrary subset of $\{0, \dots, L-1\}$) and generating the unknown frames (those with indices in $\mathcal{U}$).

Our approach allows us to use the exact same architecture of unconditionally trained models, thus enabling *mixed training*, where we train the model conditionally and unconditionally at the same time. We set $\mathcal{C} = \varnothing$ (i.e., the model does not have any conditional information $\mathbf{x}_t^{\mathcal{C}}$) with probability p_U, which is a fixed hyperparameter. If $\mathcal{C} = \varnothing$, our objective in Eq. (8) becomes the same as the objective in Eq. (5) used for unconditional training. The pseudocode for RaMViD is shown in Algorithm 1.

Algorithm 1 RaMViD.

```
Initialize model $\sim s_\theta$
$T$ = Number of diffusion steps
$K$ = Max number of frames to condition on
$L$ = Length of the video
while not converged do
    $\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x}_0)$
    $t \sim \text{Uniform}(\{0, \dots, T\})$
    $b \sim \text{Bernoulli}(p_U)$
    if $b$ then
        $\mathcal{C} = \varnothing$
    else
        $k \sim \text{Uniform}(\{1, \dots, K\})$
        $\mathcal{C} \sim \text{Uniform}\left(\{S \subseteq \{0, \dots, L-1\} \,:\, |S| = k\}\right)$
    end if
    $\mathcal{U} = \{0, \dots, L-1\} \setminus \mathcal{C}$
    $\mathbf{x}_t^{\mathcal{U}} \sim \mathcal{N}\left(\mathbf{x}_t^{\mathcal{U}}; \mathbf{x}_0^{\mathcal{U}}, \left(\sigma^2(t) - \sigma^2(0)\right)\mathbf{I}\right)$
    $\mathbf{x}_t = \mathbf{x}_t^{\mathcal{U}} \oplus \mathbf{x}_0^{\mathcal{C}}$
    Take a gradient step on: $\nabla_\theta \mathbb{E}_{\mathbf{x}_0}\left\{\mathbb{E}_{\mathbf{x}_t^{\mathcal{U}}|\mathbf{x}_0}\left[\left\| s_\theta(\mathbf{x}_t, t)^{\mathcal{U}} - \nabla_{\mathbf{x}_t^{\mathcal{U}}} \log p(\mathbf{x}_t^{\mathcal{U}} \,|\, \mathbf{x}_0)\right\|_2^2\right]\right\}$
end while
```

4 Experiments

4.1 Experimental setup

Implementation details. Our implementation relies on the official code of Nichol & Dhariwal (2021),[2] adapted to video data by using 3D convolutions. Even though most previous work uses the cosine noise schedule, we found that the linear noise schedule works better when training the model conditionally. Therefore, we use a linear diffusion schedule for our experiments. For the architecture, we also use the same as proposed by Nichol & Dhariwal (2021): a U-Net with self-attention at the resolutions 16 and 8. We do not encode the time dimension. We use two ResNet blocks per resolution for the BAIR dataset, and three blocks

[2]`https://github.com/openai/improved-diffusion`

for Kinetics-600 and UCF-101. We set the learning rate for all our experiments to 2e-5, use a batch size of 32 for BAIR and 64 for Kinetics-600 and UCF-101, and fix $T = 1000$. We found, especially on the more diverse datasets like Kinetics-600 and UCF-101, that larger batch sizes produce better results. Therefore, to increase the batch size, we use gradient accumulation by computing the gradients for micro-batches of size 2 and accumulate for several steps before doing back-propagation.

Datasets and evaluations. To compare our model to prior work, we train it on the BAIR robot pushing dataset (Ebert et al., 2017). The dataset consists of short videos, with 64×64 resolution, of a robot arm manipulating different objects. For evaluation, we use the same setting as Rakhimov et al. (2020), which is to predict the next 15 frames given one observed frame. We train on videos of length 20.

Additionally, we evaluate our model on the Kinetics-600 dataset (Carreira et al., 2018), which consists of roughly 500,000 10-second YouTube clips, also at 64×64 resolution, from 600 classes. The size and the diversity of this dataset make it a perfect task to investigate if the model captures the underlying real-world dynamics. For downloading and preprocessing we use the dataset's public repository.[3] On Kinetics-600, we compare our model to concurrent work by predicting 11 frames when conditioned on 5 frames (Luc et al., 2020). We additionally perform several ablation studies on video completion. We train on 16 frames and choose again $K = 4$.

To quantitatively evaluate the unconditional generation performance when using $p_U > 0$, we also train on UCF-101 (Soomro et al., 2012), a common benchmark for unconditional video generation. It consists of 13,320 videos from 101 human action classes. We also rescale this dataset to 64×64 and train with $K = 4$.

To quantitatively evaluate prediction, we use the Fréchet Video Distance (FVD) (Unterthiner et al., 2018),[4] which captures semantic similarity and temporal coherence between videos by comparing statistics in the latent space of a Inflated 3D ConvNet (I3D) trained on Kinetics-400. To evaluate unconditional generation, we use the Inception Score (IS) (Salimans et al., 2016) to measure the quality and diversity of the generated videos. As we have to adapt the score to videos, we use the public repository from Saito et al. (2020).[5]

4.2 BAIR

We train four models on the BAIR dataset with $p_U \in \{0, 0.25, 0.5, 0.75\}$ respectively. The models are trained for 250,000 iterations with a batch size of 32 on 8 GPUs.

First, we test our method with the typical evaluation protocol for BAIR (predicting 15 frames, given one conditional frame). With all values of p_U, we can achieve state-of-the-art performance, as shown in Table 1. By using $p_U > 0$, we can even increase the performance of our method. However, it seems that there is a tipping point after which the increasing unconditional rate hurts the prediction performance of the model. Interestingly, we find that also the model trained with $p_U = 0.75$ overcomes the harmonization problem described in Section 3.1. We have trained more models with $p_U = 0.25$ but varying K. When training with $K = 2$ we found a slight drop in performance but with $K = 8$ a slight increase. Furthermore, we experiment with a task-specific model for prediction (i.e. setting $\mathcal{C} = 0$ for all training steps), which we will call *RaMViD fixed*. However, we find that this does not improve performance and does not appear to work reliably on other video completion tasks.

Since we train with randomized masking, we can also perform video infilling with the same models, without retraining. We condition on the first and last frame (i.e., set $\mathcal{C} = \{0, 15\}$ for sampling) and compute the FVD of the 14 generated frames. Again we find that the performance is very similar for different values of p_U (see Table 2), and only a slight change in results when using different values for K. Interestingly, RaMViD is also able to perform unconditional generation on BAIR for all considered values of p_U, as shown in Appendix B.1.

So far, we have shown that our method works very well for prediction and infilling. However, since the BAIR dataset is arguably rather simple and not very diverse, we will now evaluate RaMViD on the significantly

[3]https://github.com/cvdfoundation/kinetics-dataset

[4]https://github.com/google-research/google-research/tree/master/frechet_video_distance

[5]https://github.com/pfnet-research/tgan2

Table 1: Prediction performance on BAIR. The values are taken from Babaeizadeh et al. (2021) after inquiring about the evaluation procedure. Parameter counts were obtained either directly from the papers or by contacting the authors. Since our computational constraints did not allow us to do several runs for each method, we only give error bars for RaMViD ($p_U = 0.25$)

Method	**FVD** ($\downarrow$)	**# parameters**
Latent Video Transformer (Rakhimov et al., 2020)	125.8	
SAVP (Lee et al., 2018)	116.4	
DVD-GAN-FP (Clark et al., 2019)	109.8	
TrIVD-GAN-FP (Luc et al., 2020)	103.3	
VideoGPT (Yan et al., 2021)	103.3	40M
Video Transfomer (Weissenborn et al., 2020)	94.0	373M
FitVid (Babaeizadeh et al., 2021)	93.6	302M
MCVD (Voleti et al., 2022)	89.5	251.2M
NÜWA (Wu et al., 2021b)	86.9	
RaMViD ($p_U = 0, K = 4$)	86.41	235M
RaMViD ($p_U = 0.25, K = 4$)	85.3 ± 1.8	235M
RaMViD ($p_U = 0.5, K = 4$)	85.03	235M
RaMViD ($p_U = 0.75, K = 4$)	86.05	235M
RaMViD ($p_U = 0.25, K = 2$)	87.39	235M
RaMViD ($p_U = 0.25, K = 8$)	**82.64**	235M
RaMViD ($p_U = 0.25,$ *fixed*)	89.01	235M

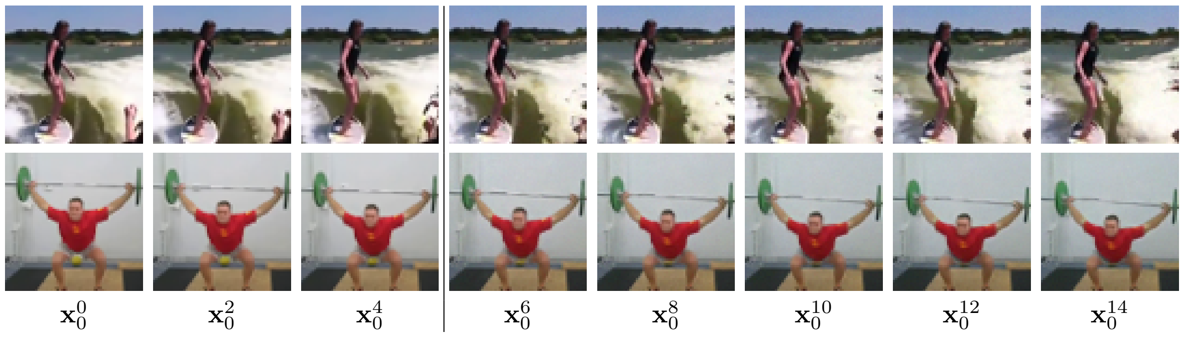

Figure 4: Prediction of 11 frames given the first 5 frames on Kinetics-600 with RaMViD ($p_U = 0.25$).

more complex Kinetics-600 dataset. Since using $K = 4$ appears to lead to stable performance on all tasks we evaluated our model on, we will use $K = 4$ for our experiments on Kinetics-600 and UCF-101.

Table 2: Infilling performance on BAIR.

Method	**FVD** ($\downarrow$)
RaMViD ($p_U = 0, K = 4$)	85.68
RaMViD ($p_U = 0.25, K = 4$)	85.02
RaMViD ($p_U = 0.5, K = 4$)	87.04
RaMViD ($p_U = 0.75, K = 4$)	87.85
RaMViD ($p_U = 0.25, K = 2$)	**83.83**
RaMViD ($p_U = 0.25, K = 8$)	87.11
RaMViD ($p_U = 0.25,$ *fixed*)	119.76

Table 3: Prediction performance on Kinetics-600. Values are taken from Moing et al. (2021) after inquiring about the evaluation procedure. Parameter counts were obtained either directly from the papers or by contacting the authors. Since our computational constraints did not allow us to do several runs for each method, we only give error bars for RaMViD ($p_U = 0.25$)

Method	**FVD** (↓)	**# parameters**
Video Transfomer (Weissenborn et al., 2020)	170 ± 5	373M
DVD-GAN-FP (Clark et al., 2019)	69 ± 1	
CCVS (Moing et al., 2021)	55 ± 1	366M
TrIVD-GAN-FP (Luc et al., 2020)	26 ± 1	
RaMViD ($p_U = 0$)	18.69	308M
RaMViD ($p_U = 0.25$)	**17.53** ± 1.07	308M
RaMViD ($p_U = 0.5$)	17.61	308M
RaMViD ($p_U = 0.75$)	27.64	308M

Table 4: Performance of RaMViD on Kinetics-600, when conditioning on different frames.

Method	$\mathcal{C} = \{0, 1, 14, 15\}$	$\mathcal{C} = \{0, 5, 10, 15\}$
RaMViD ($p_U = 0$)	**10.68**	6.28
RaMViD ($p_U = 0.25$)	10.85	**4.91**
RaMViD ($p_U = 0.5$)	10.86	5.90
RaMViD ($p_U = 0.75$)	17.33	7.29

4.3 Kinetics-600

For the Kinetics-600 dataset, we increase the batch size to 64 and train for 500,000 iterations on 8 GPUs. First, we evaluate the model on prediction with the setting described in Section 4.1 (predict 11 frames given 5 frames). When comparing our models to concurrent work, we find that RaMViD achieves state-of-the-art results by a significant margin (see Table 3). In Fig. 4, we can see that the model produces temporally coherent outputs and is able to model details, especially in the background, such as clouds and patterns in the water. Nevertheless, it struggles with fast movements: objects moving quickly often get deformed, as can be observed in Appendix B.2. Similar to what we have seen in Table 1, having an unconditional rate $p_U > 0$ increases the performance up to a tipping point. However, differently from the model trained on BAIR, the FVD score now drops significantly with $p_U = 0.75$. We conjecture that this drop in performance is due to the complexity of the data distribution. In BAIR, the conditional and unconditional distributions are rather similar, while this is not true for Kinetics-600.

We also evaluate RaMViD on two video completion tasks on Kinetics-600. The first task is to fill in a video given the two first and last frames (i.e., $\mathcal{C} = \{0, 1, 14, 15\}$): the challenge here is to harmonize the observed movement at the beginning with the movement observed at the end. In the second task, the conditioning frames are distributed evenly over the sequence (i.e., $\mathcal{C} = \{0, 5, 10, 15\}$), hence the model has to infer the movement from the static frames and harmonize them into one realistic video. RaMViD excels on both tasks, as shown quantitatively in Table 4 and qualitatively in Figs. 11 and 12. Especially when setting $\mathcal{C} = \{0, 5, 10, 15\}$ RaMViD is able to fill the missing frames with very high quality and coherence. This setting can be easily applied to upsampling by training a model on high-FPS videos and then sampling a sequence conditioned on a low-FPS video.

We find that only RaMViD ($p_U = 0.5$) and RaMViD ($p_U = 0.75$) can generate unconditional videos on Kinetics-600. To quantify RaMViD's unconditional generation, we will evaluate these models on the UCF-101 dataset and compare it to other work.

Table 5: Generative performance of RaMViD on UCF-101. Note that the methods TGAN-F, VideoGPT and DVD-GAN in Table 5 are trained with 128 × 128 resolution, which gives them a slight advantage, as the IS score is computed with 112 × 112 resolution.

Method	IS (↑)	# parameters	resolution
VGAN (Vondrick et al., 2016)	8.31 ± 0.09		64 × 64
MoCoGAN (Tulyakov et al., 2018)	12.42 ± 0.03	3.3M	64 × 64
TGAN-F (Kahembwe & Ramamoorthy, 2020)	13.62 ± 0.06	17.5M	64 × 64
progressive VGAN (Acharya et al., 2018)	14.56 ± 0.05		64 × 64
TGAN-F (Kahembwe & Ramamoorthy, 2020)	22.91 ± 0.19	70M	128 × 128
VideoGPT (Yan et al., 2021)	24.69 ± 0.3	200M	128 × 128
TGANv2 (Saito et al., 2020)	26.60 ± 0.47	200M	64 × 64
DVD-GAN (Clark et al., 2019)	**32.97** ± 1.7		128 × 128
RaMViD ($p_U = 0.5$)	20.84 ± 0.08	308M	64 × 64
RaMViD ($p_U = 0.75$)	21.71 ± 0.21	308M	64 × 64

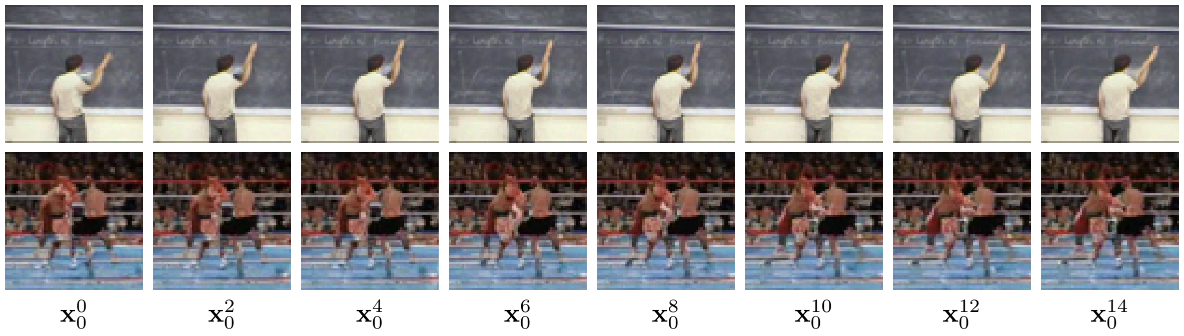

Figure 5: Unconditional generation on the UCF-101 dataset. The first generation does not have much movement and is generated very realistically. In the second video, we the background is generated properly, but we see that the fast-moving people are unrealistically deformed.

4.4 UCF-101

We train RaMViD on UCF-101 with the same setting as used for Kinetics-600 but for 450,000 iterations. Table 5 shows that our model achieves competitive performance on unconditional video generation, although it does not reach state-of-the-art. The trained models can successfully generate scenes with a static background and a human performing an action in the foreground, consistent with the training dataset (see Figs. 5 and 13). However, the actions are not always coherent and moving objects can deform over time. Note that UCF-101 is a very small dataset given its complexity. Therefore we do observe some overfitting. Since for each action we only have around 25 different settings, our model does not learn to combine those but generates very similar videos to the training set. Due to the characteristics of this dataset we think with more extensive hyperparameter tuning, one can achieve better results with RaMViD in unconditional generation. But our focus does not lie on this.

4.5 Autoregressive sampling

While we train our models only on 16 (Kinetics-600) or 20 (BAIR) frames, it is still possible to sample longer sequences autoregressively. By conditioning on the latest sampled frames, one can sample the next sequence and therefore generate arbitrarily long videos. In Fig. 6, we show examples of this autoregressive sampling with RaMViD ($p_U = 0.25$) trained on Kinetics-600. However, we found that this is rather challenging because, at each autoregressive step, the quality of the generated sequence slightly deteriorates. This amplifies over time, often resulting in poor quality after about 30 frames.

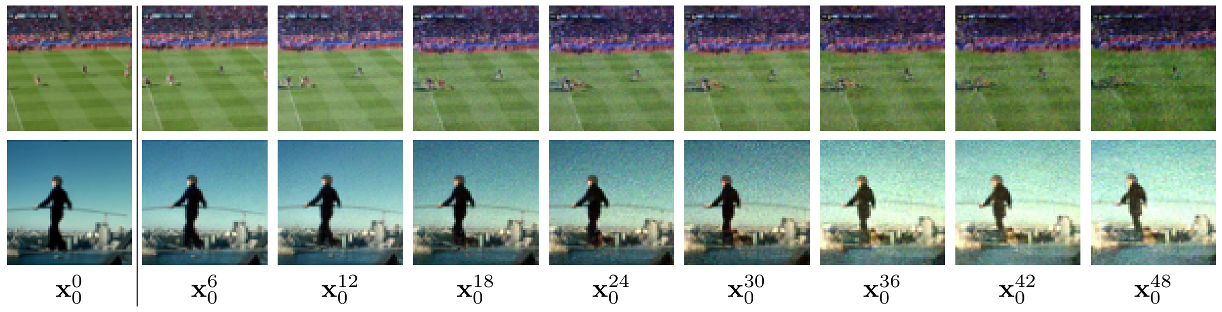

Figure 6: Autoregressive prediction of 45 frames conditioned on 5 frames with RaMViD ($p_U = 0.25$) trained on Kinetics-600.

5 Conclusion

We have shown that diffusion models, which have been demonstrated to be remarkably powerful for image generation, can be extended to videos and used for several video completion tasks. The way we introduce conditioning information is novel, simple, and does not require any major modification to the architecture of existing diffusion models, but it is nonetheless surprisingly effective. Although the proposed method targets conditional video generation, we also introduce an alternative masking schedule in an attempt to improve the unconditional generation performance without sacrificing performance on conditional generation tasks.

Since we have observed varying performance in different tasks using different masking schemes, an interesting direction for future research is to investigate which masking schedules are more suitable for each task. It would also be interesting to explore in future work whether our conditioning technique is also effective for completion on other data domains. Finally, the focus of this work has been on the diffusion-based algorithm for videos rather than on optimizing the quality of each frame. It has been shown in concurrent works that including super-resolution modules helps create high-resolution videos. Adding a super-resolution module to RaMViD would be a relevant direction for future work.

Broader impact statement

Generative models have been a cornerstone of AI research for many years. While some of these models have been used for the benefit of society e.g., in wildlife conservation, they can likewise be used with malicious intent such as for creating deepfakes. Similarly to other generative models, the capabilities of our approach for filling and predicting videos can be used for the benefit of all as well as unfortunately the opposite. In this work, we can not see any specific negative impact beyond the general possibility of malicious users of AI algorithms. In addition, our models require large video datasets for training. For some of these curated datasets, the underlying distribution of samples across various groups might not be uniform and before deploying any machine learning model trained on these datasets these aspects need to be carefully evaluated.

Acknowledgments

This project was enabled by the Berzelius cluster at the Swedish National Supercomputer Center (NSC). We thank our anonymous reviewers for their constructive feedback.

References

Korbinian Abstreiter, Stefan Bauer, Bernhard Schölkopf, and Arash Mehrjou. Diffusion-based representation learning. *arXiv preprint arXiv:2105.14257*, 2021.

Dinesh Acharya, Zhiwu Huang, Danda Pani Paudel, and Luc Van Gool. Towards high resolution video generation with progressive growing of sliced wasserstein gans, 2018.

Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction. In *International Conference on Learning Representations*, 2018.

Mohammad Babaeizadeh, Mohammad Taghi Saffar, Suraj Nair, Sergey Levine, Chelsea Finn, and Dumitru Erhan. Fitvid: Overfitting in pixel-level video prediction. *arXiv preprint arXiv:2106.13195*, 2021.

Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. Conditional image generation with score-based diffusion models, 2021.

Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation, 2020.

Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600, 2018.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations*, 2021.

Aidan Clark, Jeff Donahue, and Karen Simonyan. Efficient video generation on complex datasets. *CoRR*, abs/1907.06571, 2019. URL `http://arxiv.org/abs/1907.06571`.

Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1174–1183. PMLR, 7 2018.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. *arXiv preprint arXiv:2112.07068*, 2021.

Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg (eds.), *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pp. 344–356. PMLR, 11 2017.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2786–2793, 2017. doi: 10.1109/ICRA.2017.7989324.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 6 2019.

William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models, 2022.

Anthony Hu, Fergal Cotter, Nikhil Mohan, Corina Gurau, and Alex Kendall. Probabilistic future prediction for video scene understanding, 2020.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9000–9008, 2018.

Emmanuel Kahembwe and Subramanian Ramamoorthy. Lower dimensional kernels for video discriminators. *Neural Networks*, 132:506–520, 2020. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2020.09.016.

Yu Kong and Yun Fu. Human action recognition and prediction: A survey, 2018.

Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction, 2018.

Maomao Li, Chun Yuan, Zhihui Lin, Zhuobin Zheng, and Yangyang Cheng. Stochastic video generation with disentangled representations. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 224–229, 2019.

Pauline Luc, Aidan Clark, Sander Dieleman, Diego de Las Casas, Yotam Doron, Albin Cassirer, and Karen Simonyan. Transformation-based adversarial video prediction on large-scale data, 2020.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models, 2022.

Sarthak Mittal, Guillaume Lajoie, Stefan Bauer, and Arash Mehrjou. From points to functions: Infinite-dimensional representations in diffusion models. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022.

Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. CCVS: Context-aware controllable video synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171. PMLR, 7 2021.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling, 2020.

Sergiu Oprea, Pablo Martinez-Gonzalez, Alberto Garcia-Garcia, John Alejandro Castro-Vargas, Sergio Orts-Escolano, Jose Garcia-Rodriguez, and Antonis Argyros. A review on deep learning techniques for video prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Ruslan Rakhimov, Denis Volkhonskiy, Alexey Artemov, Denis Zorin, and Evgeny Burnaev. Latent video transformer, 2020.

Chitwan Saharia, William Chan, Huiwen Chang, Chris A. Lee, Jonathan Ho, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models, 2021a.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement, 2021b.

Caner Sahin, Guillermo Garcia-Hernando, Juil Sock, and Tae-Kyun Kim. A review on object pose recovery: from 3d bounding box detectors to full 6d pose estimators, 2020.

Masaki Saito, Shunta Saito, Masanori Koyama, and Sosuke Kobayashi. Train sparsely, generate densely: Memory-efficient unsupervised training of high-resolution temporal GAN. *International Journal of Computer Vision*, 128(10-11):2586–2606, 5 2020. doi: 10.1007/s11263-020-01333-y.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved Techniques for Training GANs. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Saeed Saremi, Arash Mehrjou, Bernhard Schölkopf, and Aapo Hyvärinen. Deep energy estimator networks. *arXiv preprint arXiv:1805.08306*, 2018.

Vaibhav Saxena, Jimmy Ba, and Danijar Hafner. Clockwork variational autoencoders. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 7 2015. PMLR.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild, 2012.

Jiangxin Sun, Jiafeng Xie, Jian-Fang Hu, Zihang Lin, Jianhuang Lai, Wenjun Zeng, and Wei-shi Zheng. Predicting future instance segmentation with contextual pyramid convlstms. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, pp. 2043–2051, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450368896. doi: 10.1145/3343031.3350949.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Adam Terwilliger, Garrick Brazil, and Xiaoming Liu. Recurrent flow-guided semantic forecasting. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1703–1712, 2019. doi: 10.1109/WACV.2019.00186.

Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6 2018.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *CoRR*, abs/1812.01717, 2018.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23 (7):1661–1674, 2011. doi: 10.1162/NECO_a_00142.

Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. Mcvd: Masked conditional video diffusion for prediction, generation, and interpolation, 2022.

Carl Vondrick and Antonio Torralba. Generating the future with adversarial transformers. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2992–3000, 2017. doi: 10.1109/CVPR.2017.319.

Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Jacob Walker, Abhinav Gupta, and Martial Hebert. Dense optical flow prediction from a static image. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2443–2451, 2015. doi: 10.1109/ICCV.2015.281.

Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. In *International Conference on Learning Representations*, 2020.

Bohan Wu, Suraj Nair, Roberto Martín-Martín, Li Fei-Fei, and Chelsea Finn. Greedy hierarchical variational autoencoders for large-scale video prediction. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2318–2328, 2021a. doi: 10.1109/CVPR46437.2021.00235.

Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. NÜwa: Visual synthesis pre-training for neural visual world creation, 2021b.

Qiangeng Xu, Hanwang Zhang, Weiyue Wang, Peter N. Belhumeur, and Ulrich Neumann. Stochastic dynamics for video infilling, 2018.

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers, 2021.

Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation, 2022.

Kuo-Hao Zeng, William B. Shen, De-An Huang, Min Sun, and Juan Carlos Niebles. Visual forecasting by imitating dynamics in natural sequences. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3018–3027, 2017. doi: 10.1109/ICCV.2017.326.

A Implementation details

Fig. 7 presents a sketch of RaMViD's architecture. Thanks to the way we introduce conditioning frames, the architecture does not need to be different from the one in unconditional models.

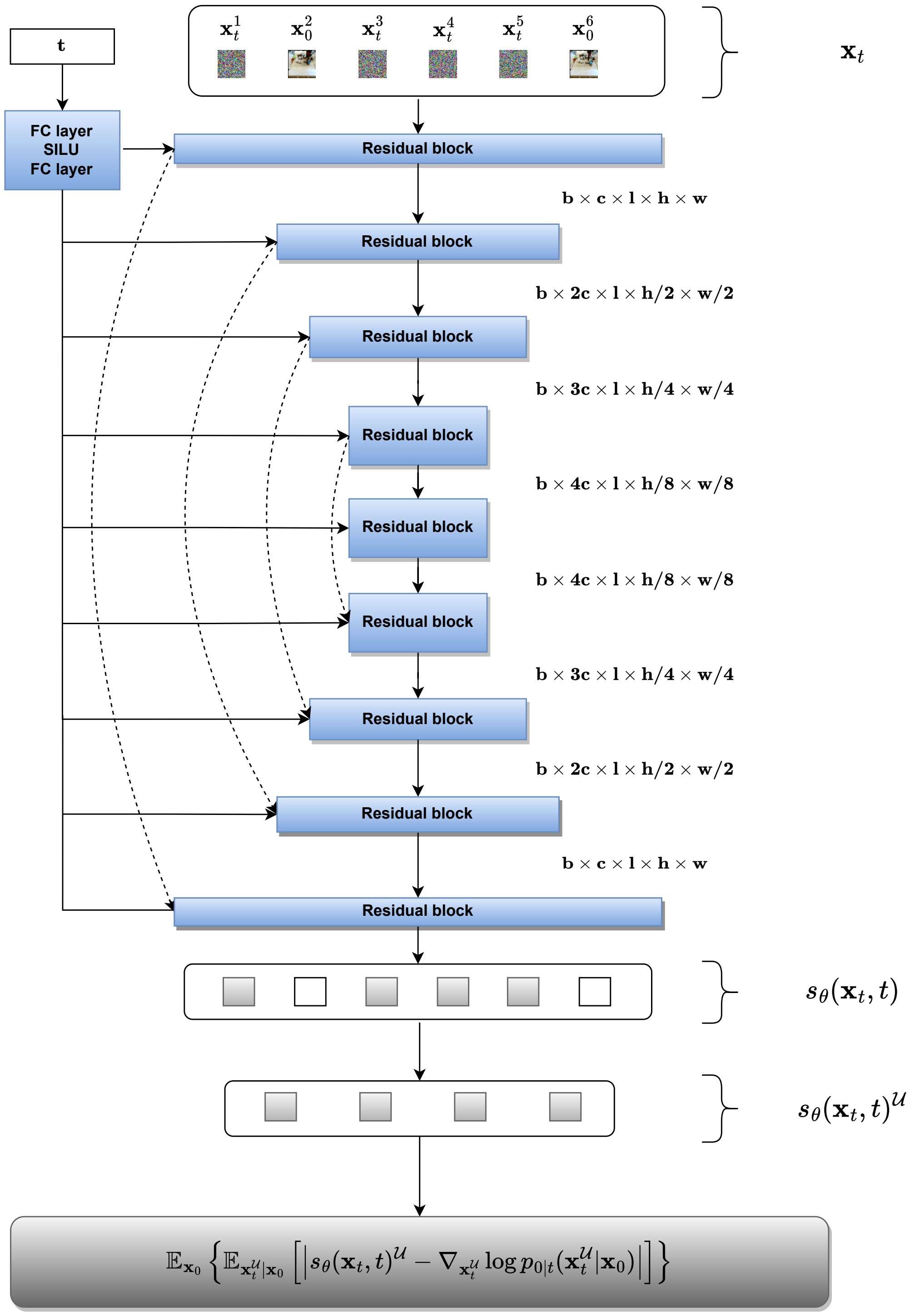
t
FC layer
SILU
FC layer
$\mathbf{x}_t^1$
$\mathbf{x}_0^2$
$\mathbf{x}_t^3$
$\mathbf{x}_t^4$
$\mathbf{x}_t^5$
$\mathbf{x}_0^6$
$\mathbf{x}_t$
Residual block
$\mathbf{b} \times \mathbf{c} \times \mathbf{l} \times \mathbf{h} \times \mathbf{w}$
$\mathbf{b} \times \mathbf{2c} \times \mathbf{l} \times \mathbf{h/2} \times \mathbf{w/2}$
$\mathbf{b} \times \mathbf{3c} \times \mathbf{l} \times \mathbf{h/4} \times \mathbf{w/4}$
$\mathbf{b} \times \mathbf{4c} \times \mathbf{l} \times \mathbf{h/8} \times \mathbf{w/8}$
$s_\theta(\mathbf{x}_t, t)$
$s_\theta(\mathbf{x}_t, t)^{\mathcal{U}}$
$\mathbb{E}_{\mathbf{x}_0}\left\{\mathbb{E}_{\mathbf{x}_t^{\mathcal{U}}|\mathbf{x}_0}\left[\left|s_\theta(\mathbf{x}_t,t)^{\mathcal{U}} - \nabla_{\mathbf{x}_t^{\mathcal{U}}}\log p_{0|t}(\mathbf{x}_t^{\mathcal{U}}|\mathbf{x}_0)\right|\right]\right\}$

Figure 7: Sketch of our method. In the last step we only compute the loss with respect to the frames that were corrupted with noise. The number of channels **c** is 128, and **l** is the video length.

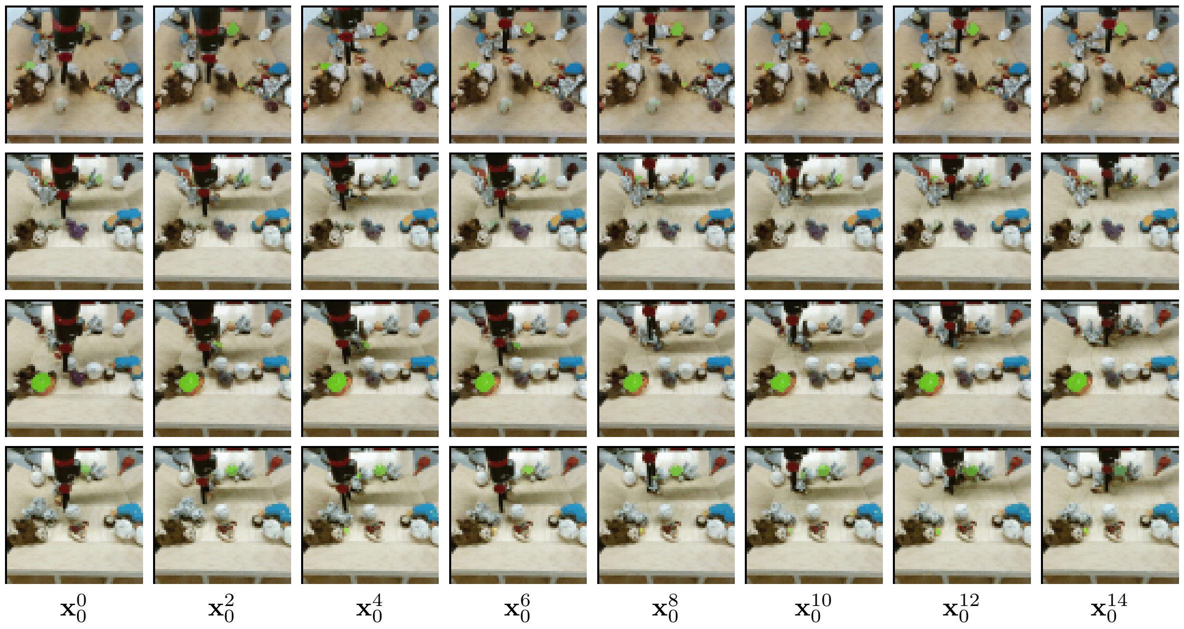

Figure 8: Unconditional generation on the BAIR dataset sampled from RaMViD for $p_U = 0$ (first row) until $p_U = 0.75$ (last row). Due to the low complexity of the dataset, we can generate reasonable unconditional videos even with $p_U = 0$. However, the quality of details increases with increasing p_U.

As mentioned, we use the linear noise schedule and the score-based ("simple") objective for all experiments. All models are trained with 1000 diffusion steps, for sampling we used 750 steps on BAIR, and 500 on Kinetics-600 and UCF-101.

As mentioned in Section 4, we use the code base from Nichol & Dhariwal (2021) (MIT license) and their proposed architecture, except that we use $3 \times 3 \times 3$ convolution kernels. In the encoder, we downsample only the spatial dimensions down to 8×8 in three steps. We use 128 channels for the first block and increase it by 128 for each downsampling step. As mentioned in Section 4, we use multi-head self-attention at the resolutions 16 and 8, each with 4 attention heads. For sampling, we found it to be more beneficial to sample from the exponential moving average (EMA) of the weights (Nichol & Dhariwal, 2021). We set the EMA rate to 0.9999.

B Additional results

B.1 Results on BAIR

Fig. 8 shows that all of the models are also able to do unconditional video generation (even RaMViD ($p_U = 0$), we assume that this is due to the low diversity of the dataset). Qualitatively, we can see in Fig. 8 that videos generated by models with higher p_U are better in generating details. While all models can generate the moving robot arm, only the models with $p_U \geq 0.25$ can properly generate the different objects in the box. However, we have no quantitative results on unconditional generation on BAIR.

B.2 Results on Kinetics-600

Kinetics-600 in practice appears to be the most difficult dataset among those considered here. While our results are state-of-the-art (see Table 3), we do observe failure cases. One of the most common failure cases is fast movement. In that case we often see a deformation of the moving object (see Fig. 9).

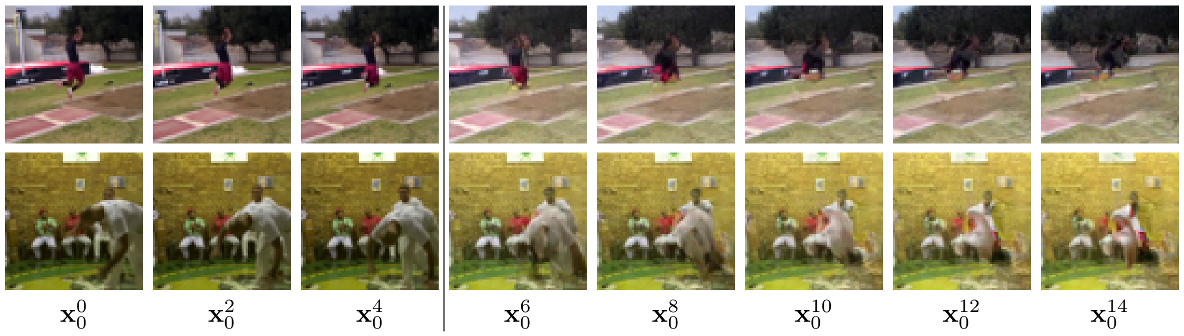

Figure 9: Prediction of 11 frames given the first 5 frames on Kinetics-600 when conditioning on fast-moving objects. We can see that the background does get preserved well, while the object itself gets unrealistically deformed.

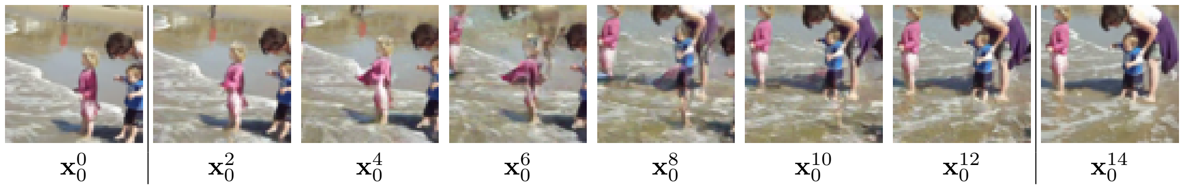

Figure 10: Infilling on Kinetics. The conditioned frames are $\mathcal{C} = \{0, 1, 14, 15\}$. The people in frames 0 and 1 are placed quite differently than in 14 and 15. The model is not able to generate the necessary camera movement and does simply interpolate between the frames.

Fast camera movement can also be a problem for infilling. If an object is placed very different between the first and last frames, the model does not generate a harmonized movement but makes the object disappear in the first and appear in the last frames Fig. 10.

C Qualitative comparison

The main qualitative improvement of RaMViD compared to other methods is the decrease in occurence of deformed objects in the predictions on Kinetics-600 (see Fig. 15). While the predictions of several other methods suffer from object deformations, RaMViD only suffers from this with fast-moving objects. On BAIR, on the other hand, our qualitative results are similar to the ones of recent methods, although RaMViD can predict details in the interactions in more detail (see Fig. 14).

D Compute

Each model is trained on 8 NVIDIA A100 GPUs with 40 GB of memory. The models on BAIR are trained with a batch size of 32 and a micro-batch size of 16 for 250k iterations (~3 days). All other models on Kinetics-600 and UCF-101 are trained with a batch size of 64 and micro-batch size of 16. The models are trained for 500k iterations on Kinetics-600 (~10 days) and for 450k iterations on UCF-101 (~9 days).

E Datasets

The videos in all datasets have more frames than we train on. Therefore we choose random sub-sequences of the desired length during training.

BAIR robot pushing. The BAIR robot pushing dataset can be used under an MIT license. We use the low resolution dataset (64 × 64). Since the data is already in the correct size, no prepossessing is necessary. For evaluation we predict one sequence for each of the 256 test videos and compare the FVD to the ground

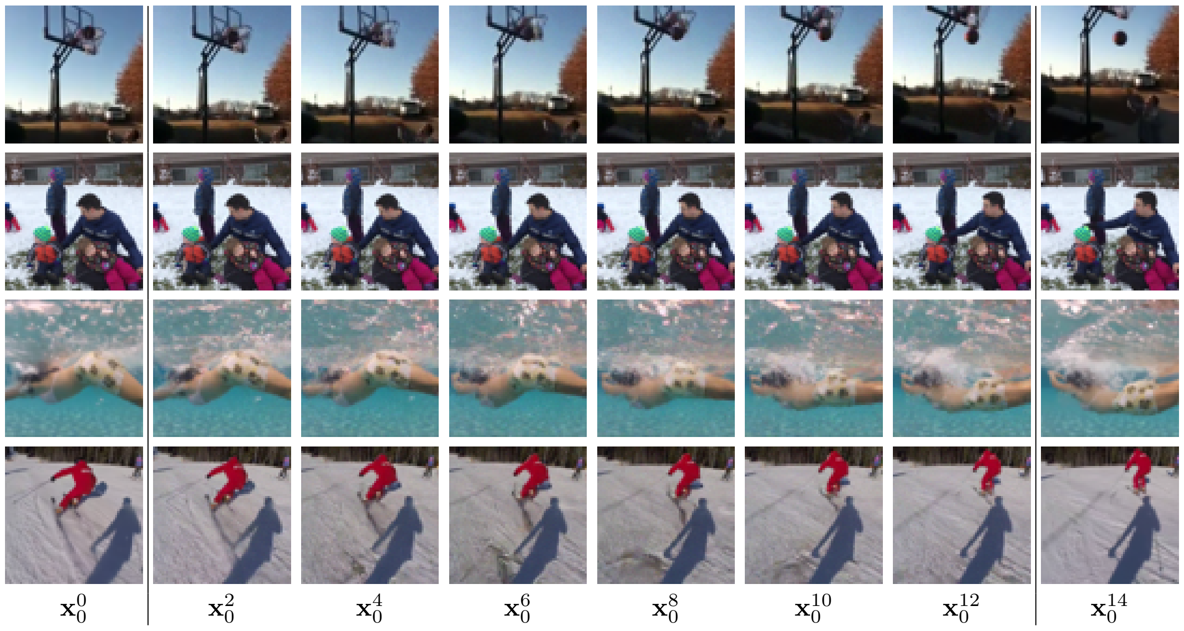

Figure 11: Video infilling ($\mathcal{C} = \{0, 1, 14, 15\}$) on Kinetics-600 with RaMViD ($p_U = 0.25$).

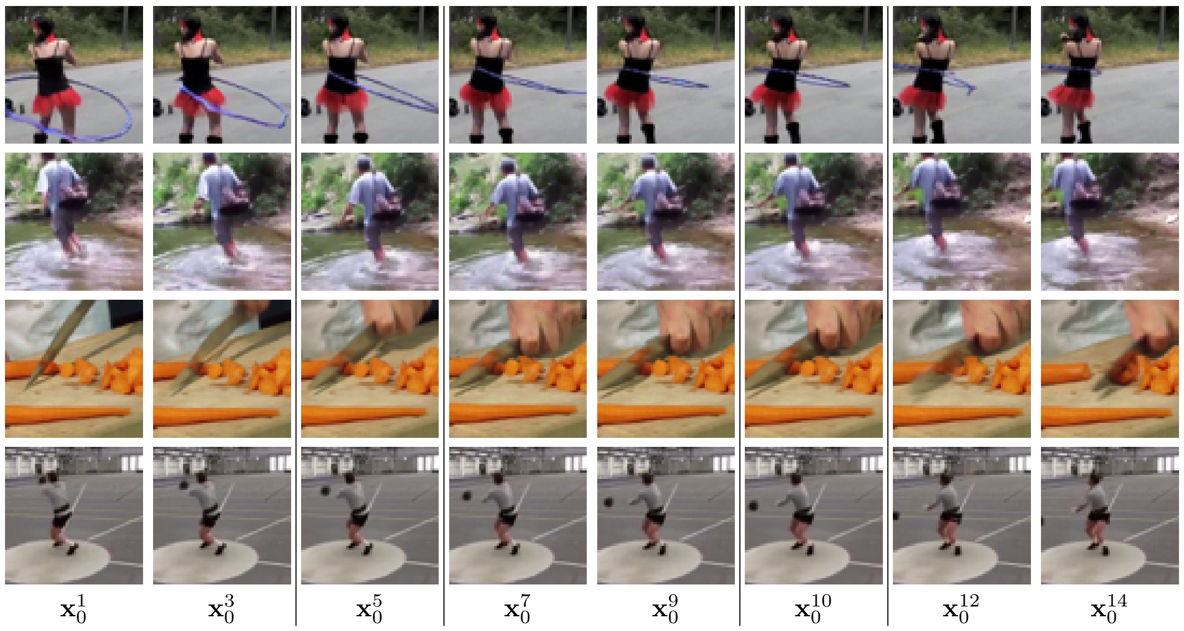

Figure 12: Video completion ($\mathcal{C} = \{0, 5, 10, 15\}$) on Kinetics-600 with RaMViD ($p_U = 0.25$).

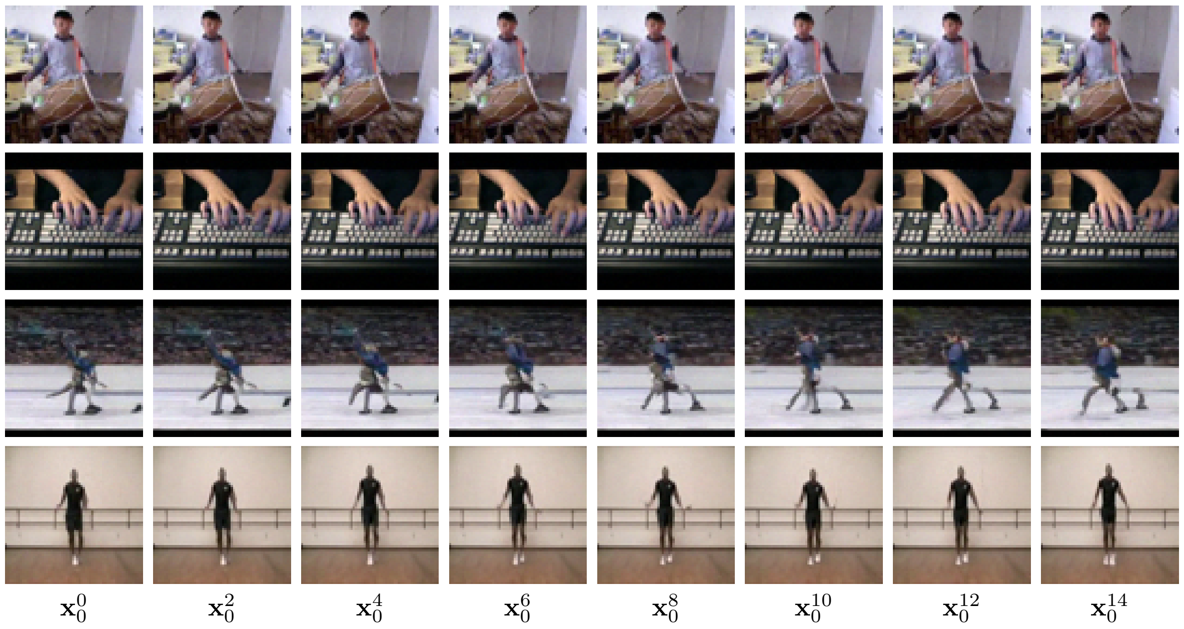

Figure 13: Qualitative results of unconditional generation on UCF-101. Scenes with less movement are generated well, but are often close to the training set.

truth. To get a proper evaluation score, we do this 100 times and the final FVD score is the average over all 100 runs. We train on a sequence length of 20.

Kinetics-600. The Kinetics-600 dataset has a Creative Commons Attribution 4.0 International License. The videos have different resolutions, which is why we reshape and center crop them to a 64 $\times$ 64 resolution. For evaluation we take 50,000 videos from the test set and predict a sequence for each of the videos. We then compute the statistics for the ground truth and the predicted videos to obtain the FVD score. We train on a sequence length of 16.

UCF-101. We could not find a license for the UCF-101 dataset. The original frames have a resolution of 160 $\times$ 120, therefore we resize and center crop the videos to a 64 $\times$ 64 resolution. We train on the entire dataset of 13,320 videos. To evaluate the generative performance, we sample 10000 videos unconditionally and compute the Inception Score (IS).[6] This is repeated three times.

F Sampling Speed

Since sampling 10,000 videos takes about 9 hours with the large model trained on Kinetics-600 and UCF-101, and 7 hours with the smaller model trained on BAIR using 500 sampling steps, it is crucial to know how many sampling steps are necessary to achieve satisfactory performance. We found that by using only 250 sampling steps for RaMViD (p = 0.25) trained on BAIR, our results drop significantly (to 101.54). However, when using 500 sampling steps, we achieve an FVD of 85.07 which is very similar to using 750 steps (84.20). We observe the same behaviour on Kinetics-600, where we achieve an FVD of 14 / 16 with 750 / 500 sampling steps but only an FVD of 49 with 250 steps. Therefore, when using the DDPM sampler with RaMViD, we recommend using a minimum of 500 sampling steps for video generation.

[6]`https://github.com/pfnet-research/tgan2`

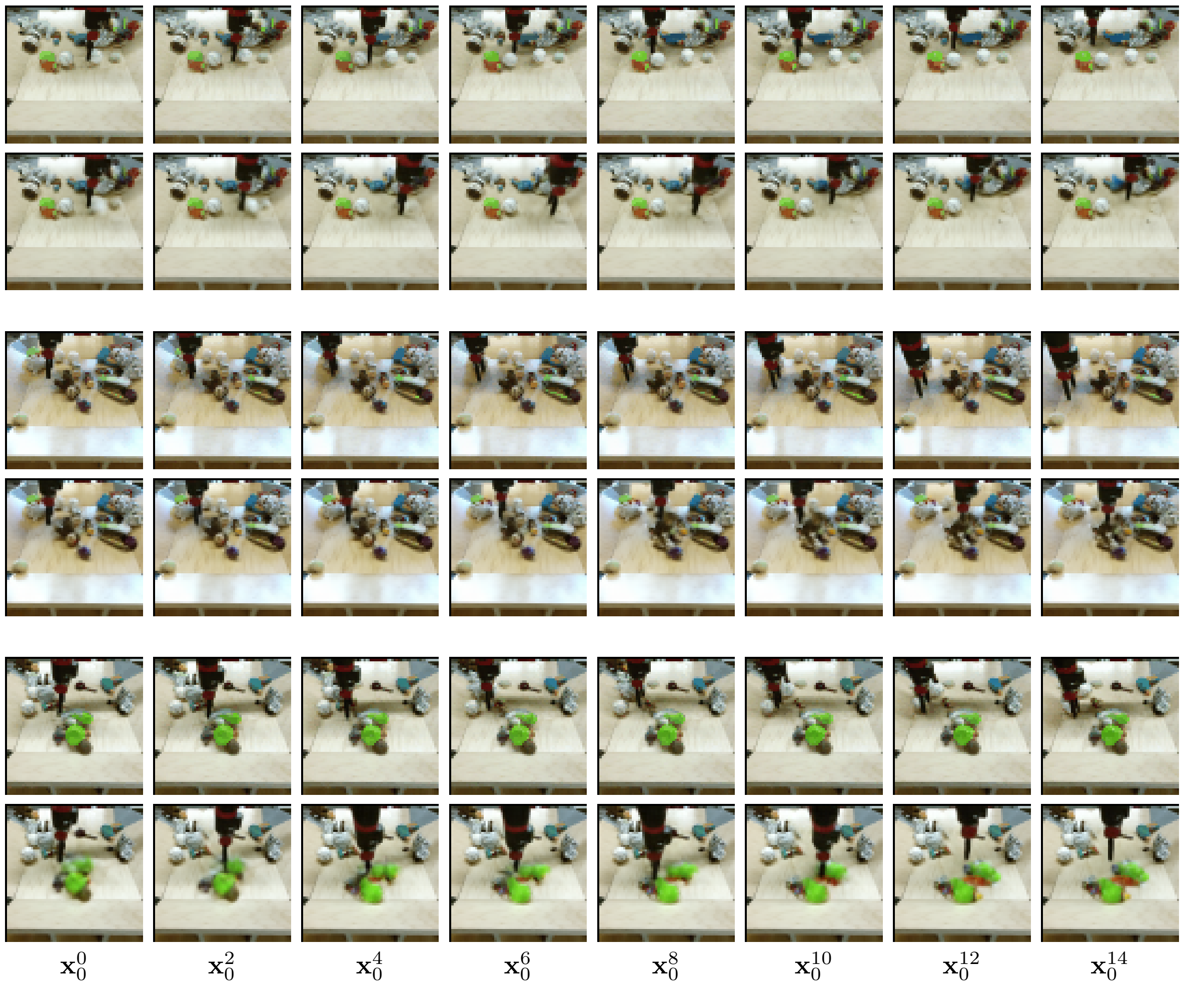

Figure 14: A comparison between RaMViD ($p_U = 0.25, K = 8$) and VideoGPT (Yan et al., 2021). The first row shows our predictions and the second VideoGPT respectively. Visually, the difference is not significant, but we found that VideoGPT produces artefacts, especially when there are interacting objects.

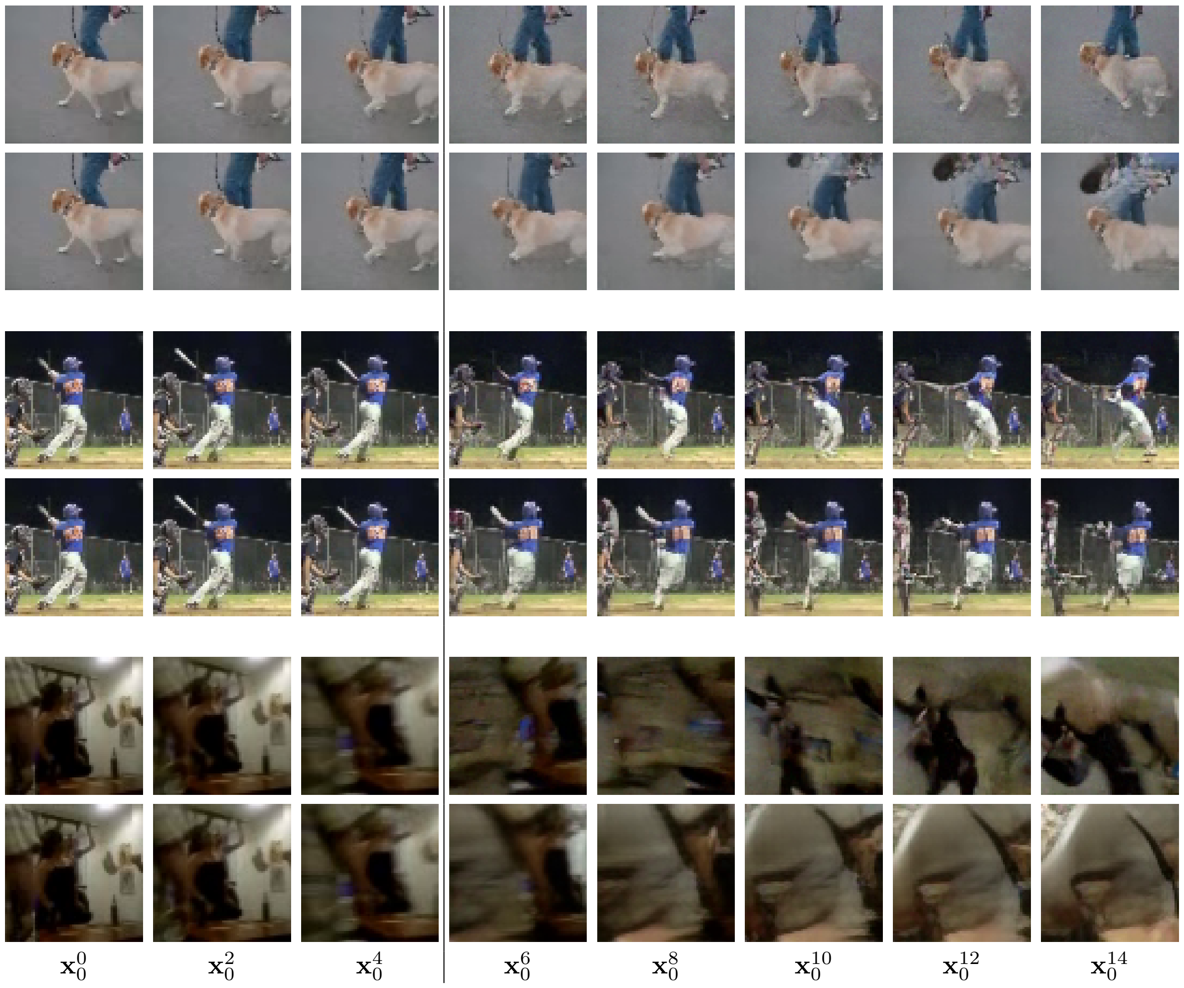

Figure 15: Predicted videos on Kinetics-600 from RaMViD ($p_U = 0.25$) in the top row, and CCVS in the bottom row. We found that our model is better in modelling motion, whereas for more static videos, the difference is visually not significant. Nevertheless, we also found that both models perform equally bad for fast object/camera movement.

G Concurrent work

As mentioned in Section 2, three concurrent works on diffusion models for videos were recently made public. Only Ho et al. (2022) and Voleti et al. (2022) consider similar tasks as we do. Ho et al. (2022) appears to outperform RaMViD on unconditional video generation on UCF-101, which is not surprising, as we train with the mixed method and therefore the models are mostly trained for conditional generation. Voleti et al. (2022) evaluate their method on BAIR with the same procedure we used, and the results reported in their publication suggest that RaMViD outperforms their proposed method.