Strumming to the Beat: Audio-Conditioned Contrastive Video Textures

Medhini Narasimhan[1] Shiry Ginosar[1] Andrew Owens[2] Alexei Efros[1] Trevor Darrell[1]
[1]University of California, Berkeley [2]University of Michigan
{medhini, shiry}@berkeley.edu, ahowens@umich.edu, {aaefros, trevordarrell}@berkeley.edu
https://medhini.github.io/audio_video_textures

Abstract

We introduce a non-parametric approach for infinite video texture synthesis using a representation learned via contrastive learning. We take inspiration from work in Video Textures [13], which showed that plausible new videos could be generated from a single one by stitching its frames together in a novel yet consistent order. This classic work, however, was constrained by its use of hand-designed distance metrics, limiting its use to simple, repetitive videos. We draw on recent techniques from self-supervised learning to learn this distance metric, allowing us to compare frames in a manner that scales to more challenging dynamics, and to condition on other data, such as audio. We learn representations for video frames and frame-to-frame transition probabilities by fitting a video-specific bi-gram model trained using contrastive learning. To synthesize a texture, we randomly sample frames with high transition probabilities to generate diverse temporally smooth videos with novel sequences and transitions. The model naturally extends to an audio-conditioned setting without requiring any finetuning. Our model outperforms baselines on human perceptual scores, can handle a diverse range of input videos, and can combine semantic and audio-visual cues in order to synthesize videos that synchronize well with an audio signal.

1. Introduction

We revisit Video Textures [13], a classic non-parametric video synthesis method which converts a single input video into an infinitely long and continuously varying video sequence. Video textures have been used to create dynamic backdrops for special effects and games, 3D portraits, dynamic scenes on web pages, and the interactive control of video-based animation [11, 12]. In these models, a new plausible video texture is generated by stitching together snippets of an existing video. Classic video texture methods have been very successful on simple videos with a high degree of regularity, such as a swinging pendulum. However, their reliance on Euclidean pixel distance as a similarity metric between frames makes them brittle to irregularities and chaotic movements, such as dances or performance of a musical instrument. They are also sensitive to subtle changes in brightness and often produce jarring transitions.

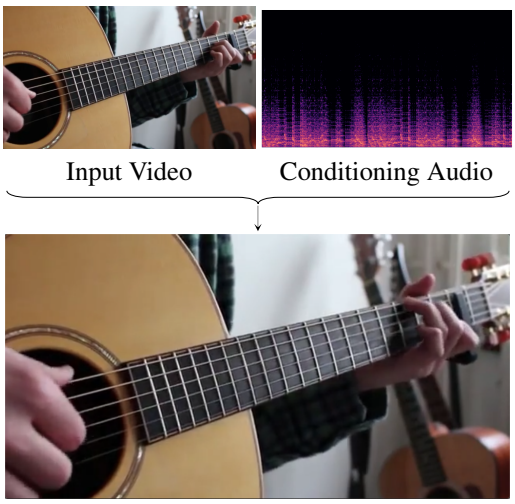

Figure 1: **Strumming to the Beat.** Click on each image to play the video/audio. We introduce Contrastive Video Textures, a learning-based approach for video texture synthesis. Given an input video and a conditioning audio, we extend our Contrastive model to synthesize a video texture that matches the conditioning audio.

Representation-learning methods have made significant advances in the past decade and offer a potential solution to the limitations of classic video texture approaches. A natural approach may be to use Generative Adversarial Networks (GANs) [5] and/or Variational Autoencoders (VAEs) [6] which have achieved great success in generating images "from scratch". Yet while video generation [8, 9, 14–17] has shown some success, videos produced using such methods are unable to match the realism of actual videos. Current generative video methods fail to capture the typical temporal dynamics of real video and as a result fail on our task of synthesizing long and diverse video sequences conditioned on a single source video. In this work, we investigate contrastive learning [1–3] approaches to graph-based sequence

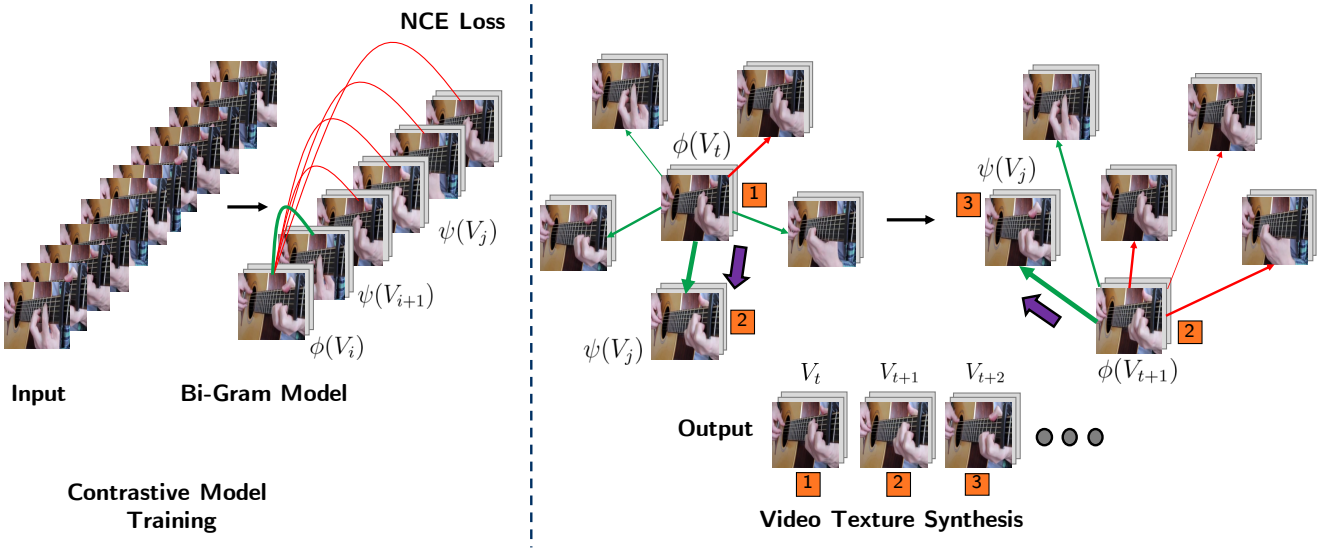

Figure 2: **Contrastive Video Textures.** We extract overlapping segments from the video and fit a bi-gram model trained using NCE loss (Eq. 2) which learns representations for query/target pairs such that given a query segment V_i, $\phi(V_i)$ is similar to positive segment $\psi(V_{i+1})$ and dissimilar to negative segments $\psi(V_j)$ where $j \in [1, ...N]$ and $j \neq i, i+1$. **Video Texture Synthesis.** During inference, we start with a random segment V_t shown by 1, compute $\phi(V_t)$ and $\psi(V_j)$ $\forall$ $j \in [1, ...N]$ and calculate the edge weights as similarity between $\phi(V_t)$ and $\psi(V_j)$. We denote higher weight edges in green and lower weighted edges in red and the thickness correlates with the probability. We randomly traverse (purple arrow) along one of the higher weighted edges to reach 2. 1 and 2 are appended to the output and the process is repeated with 2 as the query.

generation, conditional and unconditional, and demonstrate the ability of learned visual texture representations to render compelling video textures.

We propose Contrastive Video Textures, a non-parametric learning-based approach for video texture synthesis that overcomes the aforementioned limitations. As in [13], we synthesize textures by resampling frames from the input video. However, as opposed to using pixel similarity, we *learn* feature representations and a distance metric to compare frames by training a deep model on *a single input video*. The network is trained using contrastive learning to fit an example-specific bi-gram model (*i.e.* a Markov chain). This allows us to learn features that are spatially and temporally best suited to the input video.

To synthesize the video texture, we use the video-specific model to compute probabilities of transitioning between frames of the same video. We represent the video as a graph where the individual frames are nodes and the edges represent transition probabilities predicted by our video-specific model. We generate output videos by randomly traversing edges with high transition probabilities. Our proposed method is able to synthesize realistic, smooth, and diverse output textures on a variety of dance and music videos as shown at this website.

Learning the feature representations allows us to easily extend our model to an audio-conditioned video synthesis task as seen in Fig. 1. Given a source video with associated audio and a new *conditioning* audio not in the source, we synthesize a new video that matches the conditioning audio. A demonstration of this task where the guitarist is "strumming to the beats" of a new song is included in the Supp. We modify the inference algorithm to include an additional constraint that the predicted frame's audio should match the conditioning audio. We trade off between temporal coherence (frames predicted by the constrastive video texture model) and audio similarity (frames predicted by the audio matching algorithm) to generate videos that are temporally smooth and also align well with the conditioning audio.

2. Contrastive Video Textures

An overview of our method is provided in Fig. 2. We propose a non-parametric learning-based approach for video texture synthesis. At a high-level, we fit an example-specific bi-gram model (*i.e.* a Markov chain) and use it to re-sample input frames, producing a diverse and temporally coherent video. In the following, we first define the bi-gram model, and then describe how to train and sample from it.

Given an input video, we extract N overlapping segments denoted by V_i where $i \in [1, ...N]$, with a sliding window of length W and stride s. Consider these segments to be the states of a Markov chain, where the probability of transition is computed by a deep similarity function parameterized by encoders ϕ and ψ:

$$P(V_{i+1}|V_i) \propto \exp(\text{sim}(\phi(V_i), \psi(V_{i+1}))/\tau) \quad (1)$$

We use two separate encoder heads ϕ and ψ for the query and target, respectively, to break the symmetry between the two embeddings. This ensures $\text{sim}(V_i, V_{i+1}) \neq \text{sim}(V_{i+1}, V_i)$, which allows the model to learn the arrow of time. Fitting the transition probabilities amounts to fitting the parameters of ϕ and ψ, which here will take form of a 3D convolutional network. The model is trained using temperature-scaled and normalized NCE Loss [10]:

$$\begin{aligned}\mathcal{L}(V, \phi) &= \sum_{i=1}^{N} -\log P(V_{i+1}|V_i) \\ &= \sum_{i=1}^{N} -\log \frac{\exp(S(V_i, V_{i+1})/\tau)}{\sum_{j=1}^{N} \mathbb{1}_{[j \notin \{i, i+1\}]} \exp(S(V_i, V_j)/\tau)} \\ &\text{where, } S(V_i, V_j) = \text{sim}(\phi(V_i), \psi(V_j)) \end{aligned} \quad (2)$$

where τ denotes a temperature term that modulates the sharpness of the softmax distribution. As the complexity increases with number of negatives in the denominator, for efficiency, we use negative sampling [10] to approximate the denominator in Eq 2. Fitting the encoder in this manner amounts to learning a video representation by contrastive learning, where the positive is the segment that follows, and negatives are sampled from the set of all other segments. The encoder thus learns features useful for predicting the dynamics of phenomena specific to the input video.

Video Texture Synthesis. To synthesize the texture, we represent the video as a graph, with nodes as segments and edges indicating the transition probabilities computed by our Contrastive model as shown in Fig. 2. We randomly select a query segment V_t among the segments of the video and set the output sequence to all the W frames in V_t. Next, our model computes $\phi(V_t)$ and $\psi(V_j)$ for all target segments in the video and updates the edges of the graph with the transition probabilities, given by $\text{sim}(\phi(V_t), \psi(V_j))$.

Given that we fit the model on a single video, it is important that we ensure there is enough entropy in the transition distribution in order to ensure diversity in samples synthesized during inference. Always selecting the target segment with the highest transition probability would regurgitate the original sequence, as the model was trained to predict V_{j+1} as the positive segment given V_j as the query. Thus, given the current segment V_j, while we could transition to the very next segment V_{j+1}, we want to encourage the model to transition to other segments similar to V_{j+1}. While we assume that our input video sequence exhibits sufficient hierarchical, periodic structure to ensure repetition and multi-modality, we can also directly adjust the conditional entropy of the model through the softmax temperature term τ. A lower temperature would flatten the transition probabilities (*i.e.* increase the entropy) and reduce the difference in probabilities of the positive segment and segments similar to it. To avoid abrupt and noisy transitions, we set all transition probabilities below a certain threshold to zero. The threshold is set to be $t\%$ of the maximum transition probability connecting V_j to any other node V_t. We compute t heuristically.

Next, we randomly select a positive segment to transition to from the edges with non-zero probabilities. This introduces variance in the generated textures and also ensures that the transitions are smooth and coherent. We then append the last s number of frames in the positive segment to the output. This predicted positive segment V_{t+1} is again fed into the network as the query and this is repeated to generate the whole output in an autoregressive fashion.

Audio-Conditioned Video Textures. We show that it is easy to extend our Contrastive Video Textures algorithm to synthesize videos that match a conditioning audio signal. Given an input video with corresponding audio A^s and an external conditioning audio A^c, we synthesize a new video that is synchronized with the conditioning audio. We extract N overlapping segments from the input and conditioning audio, as before. We compute the similarity of the input audio segments A^s to the conditioning audio segment A^c by projecting them into a common embedding space. We construct a transition probability matrix T_a in the audio space as, $T_a(i, j) = \text{sim}(\varphi(A^c_i), \varphi(A^s_j))$

Note that, unlike video segments in Eq. 1, the audio segments come from two separate audio signals. Hence, there's no need to have two separate subnetworks as there's no symmetry and we use the same audio encoder φ for both. We compute the transition probabilities T_v for the target video

Method	**Preference %**
Classic	3.33 ± 2.42 %
Classic Deep	6.66 ± 3.37 %
Classic+	10.95 ± 4.22 %
Classic++	9.52 ± 3.97 %
Any Classic	30.48 ± 6.22 %
Contrastive	**69.52** ± 6.22 %

Table 1: **Perceptual Studies for Unconditional Video Textures**. We show MTurk evaluators textures synthesized by all 5 methods and ask them to pick the most realistic one. We also report the chance evaluators chose *any* of the variation of the classic model.

Method	**Real vs. Fake**
Classic++	$11.4 \pm 4.30\%$
Classic+	15.7 ± 4.92 %
Contrastive	**25.7** $\pm 4.30\%$

Table 2: **Unconditional: Real vs. Fake study.** We show evaluators a pair of videos (generated and real video) without labels, ask them to pick the real one. Our method fools evaluators more times than Classic.

Method	Real vs Fake
Random Clip	$15.33 \pm 5.76\%$
Audio NN	$20.4 \pm 6.63\%$
Contrastive	**26.74** $\pm 6.14\%$

Table 3: **Conditional: Real vs. Fake study.** We show evaluators a pair of videos (generated and real video) without labels and ask them to pick the real one. Our method fooled evaluators more often than the baselines.

segments given the previous predicted segment using the Contrastive video textures model (Eq. 2). The joint transition probabilities for a segment are formulated as a trade-off between the audio-conditioning signal and the temporal coherence constraint as, $T = \alpha T_v + (1 - \alpha)T_a$

3. Experiments

We curate a dataset of 70 videos from different domains such as dance and musical instruments including piano, guitar, suitar, drums, flute, ukelele, and harmonium. A subset of these videos were randomly sampled from the PianoYT dataset [7] and the rest were downloaded from YouTube. We used 40 of the 70 videos to tune our hyperparameters and tested on the remaining 30 without any tuning. We conduct perceptual evaluations on Amazon MTurk to qualitatively compare the results from our method to different baselines. We also introduce and report results on a new metric,diversity score, which measures the diversity of the textures.

In Tab. 1, we compare our results to the classic video textures algorithm [13] and to three variations of the algorithm. Classic+ appends multiple frames to the output sequence instead of a single frame, Classic++ adds a stride while filtering the distance matrix and Classic Deep uses ImageNet pretrained ResNet features instead of raw pixel values. Our contrastive model surpasses all baselines by a large margin and was chosen 69.52% of the time. Additionally, we report results of real vs. fake studies in Tab. 2.

Diversity. For a fair comparison, we set the temperature for both Contrastive and Classic+ methods such that the resulting videos have approximately the same number of transitions (9±2). Evaluators were shown textures from both methods and asked to pick the one they found more realistic. Contrastive videos were preferred 76.6% of the time, comparable to the result in Tab. 1, indicating that our method finds better transitions. Additionally, we measure diversity score (DS) as the number of new transitions in every 30 seconds of the synthesized video, averaged over all videos. A transition is considered *new* if it hasn't occurred in the 30 second time-frame. Our method achieves a DS of 7.78, indicating that our textures are diverse and contain, on an average, 7 (of 9) new transitions every 30 seconds. Classic+ achieves a DS of 2.3 indicating that the video texture loops

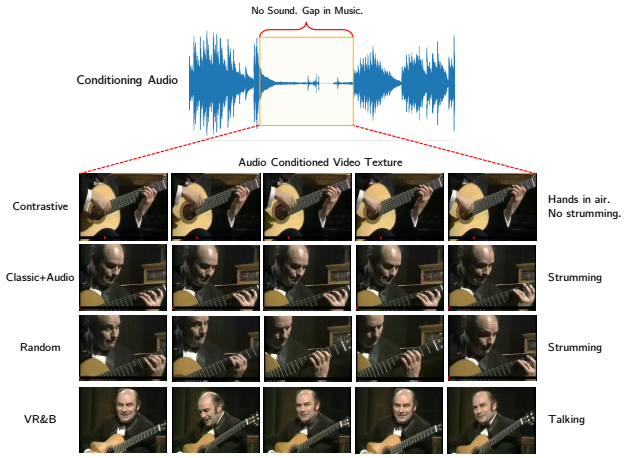

Figure 3: Qualitative comparison of audio conditioned video textures synthesized by Classic+Audio, Random Clip, Visual Rhythm and Beat (VRB) and our Contrastive model. The conditioning audio waveform shows a gap in the audio where no music is being played. Our model is able to pick up on that and the corresponding video that is synthesized has hands in the air and no strumming. However both Random Clip and Classic+Audio show strumming and VRB shows the person talking.

over the same part of the input video.

For audio conditioned video synthesis, we compare audio conditioned video textures synthesized by our method to four baselines; (1) Random Replay: we randomly choose a portion of the source video to match the conditioning audio. (2) Classic+Audio: we add audio conditioning to classic video textures. (3) Visual Rhythm and Beat (VRB) [4] (4) Audio Nearest Neighbours: for each conditioning audio segment, we pick the nearest neighbour source audio segment and the corresponding video segment.

We conduct perceptual studies comparing the audio conditioned video textures synthesized by our contrastive model and all of the baselines. The evaluators were shown two videos with the same conditioning audio, one synthesized by our method and the other by the baseline. They were asked to pick the video that they felt was more in sync with the audio. Our method was chosen 92% of the time when compared with Classic+Audio, 84% of the time when compared with VRB, 70% of the time when compared with Random Replay and 66% of the time when compared with Audio NN. As shown in Tab. 3, we conducted a real vs. fake study comparing the ground truth videos with the synthesized videos from contrastive and the two best baselines (Random Clip and Audio NN). While Random Clip and Audio NN beat the ground truth only 15.33% and 20.4% respectively, our method was able to fool evaluators 26.74% of the time.

We show qualitative results comparing our method to the five baselines described above in Fig. 3 and also include additional results at this website.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 4321

[2] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[3] Yang Chen, Yingwei Pan, Ting Yao, Xinmei Tian, and Tao Mei. Mocycle-gan: Unpaired video-to-video translation. 2019. 4321

[4] Abe Davis and Maneesh Agrawala. Visual rhythm and beat. *ACM Trans. Graph.*, 2018. 4324

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *Adv. Neural Inform. Process. Syst.*, 2014. 4321

[6] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *Int. Conf. Learn. Represent.*, 2013. 4321

[7] A Sophia Koepke, Olivia Wiles, Yael Moses, and Andrew Zisserman. Sight to sound: An end-to-end approach for visual piano transcription. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020. 4324

[8] Hsin-Ying Lee, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. Dancing to music. In *Adv. Neural Inform. Process. Syst.*, 2019. 4321

[9] Arun Mallya, Ting-Chun Wang, Karan Sapra, and Ming-Yu Liu. World-consistent video-to-video synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 4321

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Adv. Neural Inform. Process. Syst.*, 2013. 4323

[11] Arno Schödl and Irfan A Essa. Machine learning for video-based rendering. In *Adv. Neural Inform. Process. Syst.*, 2001. 4321

[12] Arno Schödl and Irfan A Essa. Controlled animation of video sprites. 2002. 4321

[13] Arno Schödl, Richard Szeliski, David H Salesin, and Irfan Essa. Video textures. 2000. 4321, 4322, 4324

[14] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 4321

[15] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Adv. Neural Inform. Process. Syst.*, 2016.

[16] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *Adv. Neural Inform. Process. Syst.*, 2018.

[17] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *Adv. Neural Inform. Process. Syst.*, 2018. 4321