Generating Coherent Timelapse Videos from a Semi-Structured Latent Space

Rory Little
Western Washington University
516 High st. Bellingham, WA
littler4@wwu.edu

Joshua Kovac
kovacj@wwu.edu

Dalton Lange
langed@wwu.edu

Finn Eitreim
eitreif@wwu.edu

Avery Le
len22@wwu.edu

Scott Wehrwein
scott.wehrwein@wwu.edu

Abstract

Timelapse videos often bear problems such as jittering, sudden dramatic changes without transition, and at the highest speeds, periodic flashing. All of these undesirable artifacts are due to the sampling method in which timelapse videos are traditionally constructed. Our goal is to develop video generation techniques to synthesize idealized timelapse videos that are free from these artifacts. Given a long term video capture source, we propose a model which can generate coherent timelapse videos of that source while displaying content from only a chosen set of timescales. Current video generation techniques use RNNs or similar architectures to perform an iterative, next-frame predictive approach to video generation. Our model instead uses contrastive learning to build a semi-structured latent space organized by timescales for each input video. Interpolation in the latent space then allows us to generate videos, while masking in the latent space allows us to filter undesirable frequencies for the videos.

1. Introduction

Timelapse videos are a nice way to visualize the passage of time in a scene. A key desire when viewing these videos is to see changes on a faster timescale than they would normally occur. However, the quality of this visualization depends heavily on the method used to generate the timelapse.

A classic method for generating timelapses is to simply sub-sample a source video at a fixed rate. While this does achieve the desired result of seeing longer term changes occur at a faster rate, the faster changes occurring in the video still exist in the timelapse, and having only been sped up. The result is a jittery experience, where changes occurring at higher frequencies still dominate the visual space of the timelapse. What we implicitly would like to see when creating and viewing timelapse videos is lower frequency changes in the source video scaled up to a higher, more human scale frequency, with the high frequencies from the source video being filtered out, rather than being up-sampled as well.

In the context of generating variable scale timelapse videos, current methods tend to fall short in a few key areas. Generally, we would like our generated videos to satisfy the following criteria:

1. The model should be able to generate arbitrary frame-to-frame time steps, as the time between frames of a lapse video is variable and depends on the scale of the timelapse, length of the video and frame rate of the video.

2. The model should be able to follow a coherent sequence of events from beginning to end (ie. when the sky gets dark, street lights should turn on).

3. Changes that occur at frequencies too high relative to the scale of the timelapse should not be included in the video (ie. changes at the scale of people walking or cars driving should not be seen by a quick timelapse video which represents several hours of footage).

Our proposed model satisfies these criteria, and is able to generate plausible timelapses from a variety of fixed camera scenes. In addition, we propose a variety of editing methods for changes at differing timescales, such as generating videos containing multiple timescales normalized to a human viewable frequency, and the insertion of objects into times that they were not originally a part of.

Our method utilizes recent advancements in deep learning in applying structure to an embedding space in order to facilitate our editing controls. In our embedding space, timescales become "disentangled" into separated bins, allowing us to independently manipulate objects that change in those timescales. We limit ourselves to source material from fixed-camera, long-term capture videos. These types

of videos are ideal for visualization of change via timelapses because the only changes that occur in the video are due to changes in the scene, not due to changes in perspective or movement of the camera.

2. Background and Related Work

2.1. Temporal Gaussian Pyramids

Swift et al. developed a method to visualize changes at various timescales in fixed-camera, long-term capture videos [10]. In this work, a sequence of *temporal gaussian pyramids* are generated by blurring subsequent frames across the time dimension, thereby filtering out high frequency changes by smearing the changes across many frames. This blurred result is then subsampled to generate the next layer of the pyramid. In their work, they found that a similar supersampling method can be used to generate timelapse videos without frequencies past a specific threshold. However, there are several limitation for this technique.

For one, the method requires quite a large amount of data and computation time to generate a single video, as generating a single frame requires the computation of an exponentially growing number of frames from lower temporal gaussian pyramids. The method also, due to its temporal blurring, contains some artifacts at larger timescales, such as a permanent twilight look due to the averaging of day and night frames. A further limitation from this method is that all timelapse videos that it can generate are derived directly from a Gaussian pyramid; novel compositions within the scene are not able to be synthesized, as each timelapse frame is firmly derived from existing frames. Due to these limitations, and the very repetitive nature of the fixed-camera videos, it should be possible to both represent the changes with much less data, and learn to perform a similar task using deep learning.

2.2. Generative Video Models

Recent advances in generative image models ([6], [12]) have led to remarkable results in many sub-domains, such as video generation([13], [5]). While these models have stunning results to look at, they tend to be limited for generating timelapse videos by a few pitfalls. Some models rely on a next-frame predictive architecture, where a single or limited sequence of frames is generated from a window of pre-existing context. A common problem with these models is that they have no overarching direction and struggle to generate sequences over longer periods of time.

A known solution to the above problem, used by other video generation models, is to first select several guiding points throughout the video, in order to add a "plot" to the generated video ([9], [11]). While this approach has seen success at generating arbitrary length videos, there tends to be a disconnect between the overall guiding structure of the video and the local frame generation, leading to abrupt, non-coherent changes when analyzing the overall trajectory of the video.

2.3. Timescale Disentanglement

The idea for and method of disentangling the representations of various timescales in some latent space is a recent innovation by Kovac et al. [7] [1]. We build on this work and its applications, and give a brief overview here.

Kovac et al. train an autoencoder to encode frames from some input video into a sequence of bins so that, if an object in a frame changes in the original video at frequency f, the information in the encoded form of that frame needed to reconstruct that object should be located in bin_s, where $s \leq \log f < s + 1$. In other words, bin s of the encoded form should represent information relevant to frequencies from $\exp(s)$ to $\exp(s + 1)$. The current method to create this encoded structure relies on two separate terms in the loss function of the autoencoder. The structure of the autoencoder is such that an image x is encoded into a compressed form w, a set of M bins, each retaining two spatial dimensions and a channel dimension.

The first term of the loss function is designed to enforce separation of frequencies into bins via a contrastive[1] loss. The general idea for this term is that we would like the embedded forms of images that are some temporal distance apart to be similar in specific bins. These specific bins are those that encode frequencies greater than that temporal distance between the images. To compute this loss, we sweep across all binned frequencies, specifically sampling positive examples for each frequency, and considering all other example images to be negative examples for that frequency.

First, we need a metric for measuring the similarity of two latent codes w_i and w_j derived from images x_i and x_j, judging them by how similar their encodings are of frequencies at or below some threshold f. As such, let w^f be the embedded code w masked to zero out all bins representing frequencies higher than f. The metric for comparing the similarities of two embedded forms is

$$\text{sim}(w_i, w_j, f) = \frac{w_i^f \cdot w_j^f}{\|w_i^f\| \|w_j^f\|} \tag{1}$$

the cosine similarity of the masked latent codes. Cosine similarity is chosen so that variance in the magnitude of the expression of different features is not punished, as it is only necessary that the features are expressed in a similar way.

This similarity measure is used to apply a contrastive loss, with an attractive force given to bins whose timescales are similar, and a repulsive force given to those that are different. A positive pair, x_i and x_j are chosen such that $f \approx |t_i - t_j|$ for respective in-video timestamps t_i and t_j

[1] This paper is concurrently up for review by the graduate committee.

of the frames. While the positive example is explicitly sampled, all other images in the mini-batch of N images are considered to be negative examples when compared to x_i. The contrastive loss term for a mini-batch of N items at a frequency f is then

$$\mathcal{L}_c(w, i, j, f) = -\log \frac{\exp(\text{sim}(w_i, w_j, f)/\tau)}{\sum_{k=1}^{N} \mathbb{1}_{k \neq i} \exp(\text{sim}(w_i, w_k, f)/\tau)} \tag{2}$$

A second term in the loss function encourages similarity between input images x^f from some frequency level of a temporal Gaussian pyramid, and reconstructed images $\hat{x}^f$, which are derived from masked latent codes w^f. This second term in the loss,

$$\mathcal{L}_r(x_i, f) = \lambda_f \Big(\alpha(1 - \text{MS-SSIM}(\hat{x}_i^f, x_i^f)) + (1 - \alpha)|\hat{x}_i^f - x_i^f| \Big) \tag{3}$$

both drives actual reconstruction of the input images (a typical autoencoder task), and also ensures that enough information to reconstruct some limited form of the scene is stored in a subset of the latent bins. Both L_1 and multi-scale structural similarity[14] are used as distance metrics in this term, to enforce both pixel-to-pixel and structural reconstructive accuracy. The mix between these two evaluations of image distance is controlled by the hyperparameter α. A sequence λ_f is also used to control how much each frequency contributes to the overall loss, with more weight being placed on reconstructions of higher frequencies.

Combined, and appropriately averaged, these two terms craft a loss function which can properly disentangle the timescales of a video, and in the process enforcing a structure on the latent space of the autoencoder.

3. Timelapse Video Generation

Once disentangled, the latent space w now has a few key properties desirable for generating timelapse videos. For one, most animated objects in a scene now are categorized in their latent representations by the frequencies at which they change. This means that two new editing controls are available for individual image reconstructions:

1. Specific frequencies can be masked out of an image at will, yielding the option to remove arbitrary objects from a scene.

2. Specific frequencies can be substituted into a scene, yielding the option to populate a frame with objects taken from a different frame of the source video.

Additionally, Kovac et al. [7] also note that the disentangled latent space is much smoother in the path drawn by encoding sequential frames from a source video. Due to this, there is less error when approximating intermediate frames between a start and end goal than there would be in an unstructured model.

While these controls only edit individual frames at a time, we find that they can be applied to sequences of latent codes to generate coherent video editing controls. The key to creating a timelapse, then, is finding a sequence of latent codes to use.

3.1. Approximating Gaussian Pyramids

Due to equation 2, we can directly attempt to reconstruct the timelapse results of [10] with a fraction of the data and computational cost. We can use the following simple procedure: subsample frames uniformly from a source video, encode those subsampled frames through the timescale disentanglement autoencoder, mask undesirable frequencies from the latent codes, and finally reconstruct the masked frames. This method is the most "honest" - frames in the generated timelapse are the direct result of real intermediate frames between some start and end points in the source video. However, we are able to use additional benefits of the smoothness of the latent space to generate a sequence of intermediate frames without needing to directly sample those intermediate frames.

3.2. Latent Interpolation

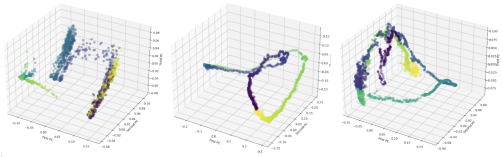

Figure 1. PCA projection of sequential w representations to three dimensions of our (1) Bryant Park, (2) Rane, and (3) Geiranger datasets. Points are sampled uniformly along a 24 hour time span, with color indicating the progression of time.

Interpolating between two points w_1 and w_2 in a generative model's latent space is known to have interesting and useful results [3]. Using this technique, we can generate intermediate latent points after only retrieving start and end points using the encoder (see figure 2). This does, however, leave us with artifacts related to the crude interpolation technique.

There are still artifacts familiar to standard autoencoder design, such a crossfading effect of objects that change at higher frequencies. While these artifacts can be somewhat hidden through latent masking, the use of this technique is still limited, as intermediate frames in even mid to low frequencies do not necessarily match their real-video-sampled counterparts. This is due to the nonlinear nature of the arrow of time in the latent space, and the complex paths that a forward temporal motion weaves.

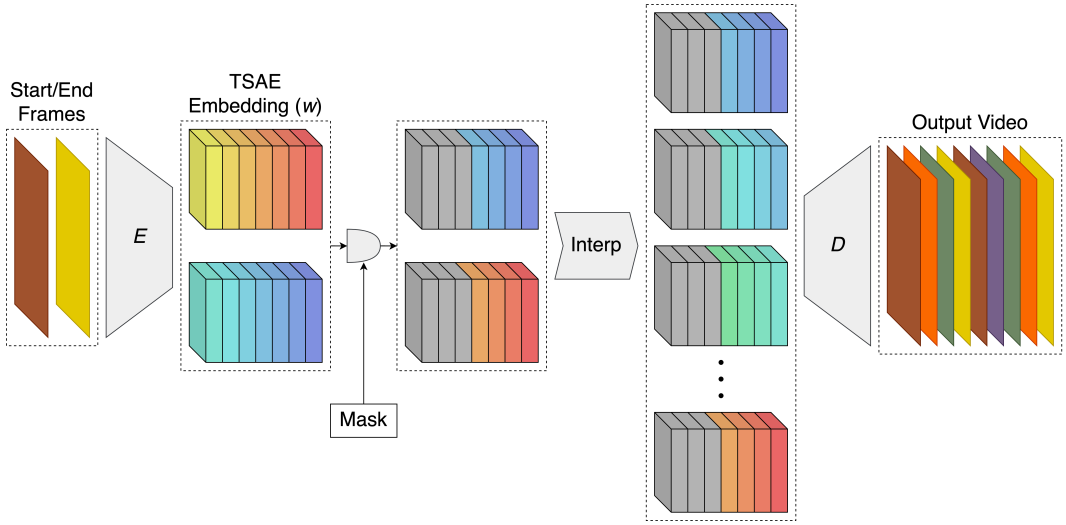

Figure 2. Our model for synthesis of intermediate frames via interpolation. A pair of input frames are encoded into a disentangled embedding space w, before being masked to remove high frequencies. The masked end points are then interpolated and pointwise decoded to produce a sequence of frames for a timelapse video.

In figure 1 we plot the results of encoding sequences of frames from our videos into the latent space. It is clear that the forward direction of time does not correspond directly to a linear change in the space. Instead, points in the latent space tend to meander in circular motions around the origin. These paths are complex and sometimes seemingly random, indicating that further nonlinear transformations of the latent would be required in order to create a space ready for a nice interpolation. While we are not able to perfectly mimic these paths right now, we can create better approximations of the paths by using spherical interpolations, rather than the linear interpolations used in standard generative examples. We also justify the use of spherical interpolation due to how it better matches the similarity metric from equation 1.

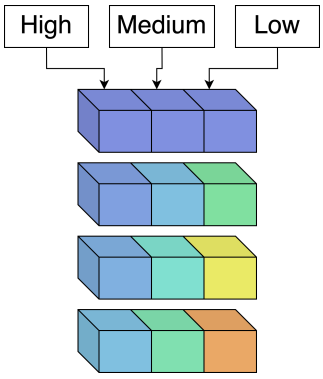

Figure 3. Different frequencies sampled at different scales in the latent code, with higher frequencies being sampled at higher rates, and lower frequencies being sampled at lower rates. The direction of time between the latent codes here is down, and colors indicate the flow of time from the source material.

3.3. Multiscale Video Generation

We can also perform summarizations of a scene in the style of [15], due to our ability to swap binned frequencies. The procedure here is also fairly simple - we perform our latent subsampling to create sequences at different time scales - sampling the high frequencies at a normal rate, but sampling other frequencies at lower rates to bring their energy up to match the highest frequencies, as seen in figure 3. In this way, we can display multiple timescales in a single timelapse.

We find that for this method, it is necessary to perform the high frequency sampling using subsampling from the source video, as described in section 3.1. While interpolation does perform well at tracking lower frequency changes, the most local structure of the passage of time in the latent is still extremely jittery, and so relying only on interpolation fails to yield coherent results.

4. Results

Our method of timelapse video generation is able to successfully generate coherent timelapse which fit our earlier criteria. Both methods for sampling intermediate latent codes generate good-looking videos, although a few key differences should be noted. The main difference between the two is the difficulty of interpolation at capturing complex high frequency motions. However, another difference exists in how scenes gradually change over time. For example, in figure 5, a sunset period is skipped, as day fades directly into night. A clear juxtaposition of the differences between the two methods is in figure 4, where the video-sampled frames differ significantly from the interpolated

Figure 4. Three techniques for creating timelapse videos.
Top: Traditional subsampled frames from a source video.
Middle: Same frames as the above, but encoded, masked to eliminate high frequencies, and reconstructed.
Bottom: Frames interpolated between start and end points.

frames. However, these differences only are clear when viewing the methods two side by side, and in both dataset examples, the timelapse generated by interpolation would be considered coherent on its own.

We do run into some problems with reconstructing movements at high temporal frequencies, such as the motion of people walking, or cars driving by. This is due to numerous issues, such as the bottleneck of information in the latent space and limitations of the autoencoder. The main problem is with generating high frequency changes, as noted earlier, is that at the smallest scales, the latent space still is not very smooth, and so simple interpolation methods fail to capture natural looking changes. However, mid-frequency movements, such as a boat turning, or low frequency movements, such as a day/night cycle, are reconstructed well by our method.

Figure 5. Interpolation transitioning from day to night in Rane.

The results of our multiscale video generation are less promising. While some success has been seen here, results do lack in fidelity due to some leakage of information between representations of timescales. For example, in figure 7, we see a man only partially reconstructed in the center of the frame, due to leaking of his representation into lower frequency sections of the latent space. In a better case, we try a similar experiment in Bryant Park in figure 6, attempting to show skaters moving at a normal speed while shifting day into night. While there is some success here, clearly the time of day has not fully transitioned. This is again due to leakage between frequency bins, this time where the high frequency portions of the latent code containing the ice skaters also encode information about the time of day, making it impossible to swap them fully into a night scene.

5. Future Work

While our method for timelapse video generation yields good results under certain conditions, there is more that could be done. Learning to generate the paths seen in figure 1 would yield substantially better results in generating low and mid frequency changes without sampling frames, and provide more insight into how the timescale autoencoder represents the linear passage of time. Additionally, further improvements in the autoencoder itself could greatly benefit the generation of videos using it. Particularly, improving the smoothness of the path through the latent space even more would assist greatly in correctly interpolating across high frequency changes, and could possible lead to insights for full blown video synthesis.

There is also space here for novel video generation. While the latent space of the autoencoder has a strict structure, and is not able to be randomly sampled directly, several recent architectures ([4], [2]) to enable the sampling of latent spaces of pre-trained autoencoders could be used to enable video synthesis without fixed start and end points.

We have attempted work on using a simplistic Variational Autoencoder [8] for these purposes, using an architecture where we encode the latent w space from the timescale disentanglement autoencoder into a second latent

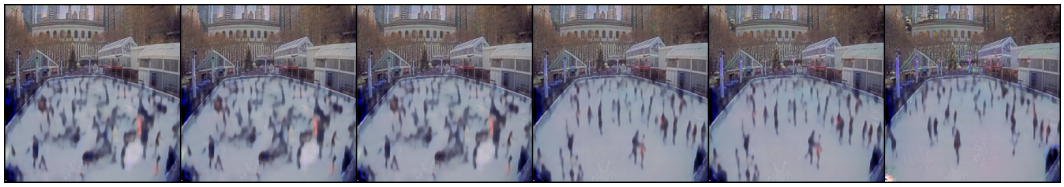

Figure 6. Skaters move at a human scale while day transitions into night.

Figure 7. Failure to swap a man into the center of the frame due to leakage between bins.

space (z), whose distribution when encoding and decode video frames from our video source is Gaussian. This architecture would allow for arbitrary sampling from the z latent space without the need for fixed start and end latent codes encoded from real video frames. In addition, the non-linear relationship between w and z spaces could have benefits for interpolation, as straight lines in z space may follow the arrow of time in w space more accurately. We would like to try a two tiered architecture, where coarse latent codes are sampled from an interpolation in z space, and are then further interpolated in w space to provide better control over fine details.

6. Conclusion

We have demonstrated the use of a semi-structured latent space in generating timelapse videos. Our method used the benefits of having disentanlged timescales in a latent space to enable the masking of unwanted frequencies to aid in creating better quality timelapse videos. We also put forward techniques for latent interpolation to further automate the video generation process with less input information. Our use of a semi-structured latent space to enable editing controls in the video generation process is novel in the field of computer vision, and is the first application of such a technique.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[2] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance, 2022.

[3] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *Advances in Neural Information Processing Systems*, 33:9841–9850, 2020.

[4] Yingqing He, Tianyu Yang, Yong Zhang, Ying Shan, and Qifeng Chen. Latent video diffusion models for high-fidelity video generation with arbitrary lengths. *arXiv preprint arXiv:2211.13221*, 2022.

[5] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models, 2022.

[6] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.

[7] Joshua Kovac, Rory Little, Dalton Lange, Finn Eitreim, Avery Le, and Scott Wehrwein. Timescale disentanglement in semi-structured embeddings via self-supervised autoencoder.

[8] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, 29, 2016.

[9] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3626–3636, 2022.

[10] Melissa E Swift, Wyatt Ayers, Sophie Pallanck, and Scott Wehrwein. Visualizing the passage of time with video temporal pyramids. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):171–181, 2022.

[11] 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*, pages 1526–1535, 2018.

[12] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[13] Jacob Walker, Ali Razavi, and Aäron van den Oord. Predicting video with vqvae. *arXiv preprint arXiv:2103.01950*, 2021.

[14] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

[15] Scott Wehrwein, Kavita Bala, and Noah Snavely. Scene summarization via motion normalization. *IEEE Transactions on Visualization and Computer Graphics*, 27(4):2495–2501, 2020.