

ABSTRACT

Title of Dissertation: Advance Video Modeling Techniques
for Video Generation and Enhancement Tasks

Gaurav Shrivastava
Doctor of Philosophy, 2024

Dissertation Directed by: Professor Abhinav Shrivastava
Department of Computer Science
University of Maryland, College Park

This thesis investigates advanced techniques that are useful in video modeling for generation and enhancement tasks.

In the first part of the thesis, we explore generative modeling that exploits the external corpus for learning priors. The task here is of video prediction, i.e., to extrapolate future sequences given a few context frames. In a followup work we also demonstrate how can we reduce the inference time further and make the video prediction model more efficient. Additionally, we demonstrate that we are not only able to extrapolate one future sequence from a given context frame but multiple sequences given context frames.

In the second part, we explore the methods that exploit internal statistics of videos to perform various restoration and enhancement tasks. Here, we show how robustly they perform the restoration tasks like denoising, super-resolution, frame interpolation, and object removal tasks. Furthermore, in a follow-up work, we utilize the inherent compositionality of videos and internal statistics to perform a wider

variety of enhancement tasks such as relighting, dehazing, and foreground/background manipulations.

Lastly, we provide insight into our future work on how data-free enhancement techniques could be improved. Additionally, we provide further insights on how multisteps video prediction techniques can be improved.

Advance Video Modeling Techniques
for Video Generation and Enhancement Tasks

by

Gaurav Shrivastava

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctoral of Science
2024

Advisory Committee:

Professor Abhinav Shrivastava, Chair/Advisor

Professor Carol Y. Espy-Wilson, Dean's Representative

Professor Chris Metzler, Committee Member

Professor Tianyi Zhou, Committee Member

Professor David Jacobs, Committee Member

© Copyright by
Gaurav Shrivastava
2024

With profound gratitude, I thank my family and friends for their unyielding and unconditional support. Their constant encouragement, love, and belief in me have been the bedrock of my journey. Every achievement I've reached is a testament to their unwavering presence in my life, and I am forever grateful for the strength they have given me throughout this process.

Acknowledgments

I owe my gratitude to all the people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost, I'd like to thank my advisor, Professor Abhinav Shrivastava, for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past five years. He has always made himself available for help and advice, and there has never been an occasion when I've knocked on his door and he hasn't given me time. It has been a pleasure to work with and learn from such an extraordinary individual. From helping me structure my experiments and guiding my writing of the first paper, to encouraging me to pursue bold ideas, Professor Shrivastava has been an exceptional mentor. Beyond academics, he was the best advisor, supporting me as I navigated both my personal and professional life over these five years.

I am also deeply grateful to Professor Carol Y. Espy-Wilson, Professor Chris Metzler, Professor Tianyi Zhou, and Professor David Jacobs for agreeing to serve on my thesis defense committee and for dedicating their valuable time to review this manuscript. Your feedback and insights have been invaluable in shaping the final version of this work.

To my wife, Devanshi, I owe the deepest thanks. You stood by me as a pillar of unwavering support throughout my PhD journey, giving me strength when I needed

it most. Your love, patience, and encouragement have been a constant source of motivation, and I could not have accomplished this without you by my side.

I also want to extend my heartfelt appreciation to my labmates at the PI lab. Without you, this journey would not have been nearly as fulfilling or enjoyable. The engaging discussions, collaborative problem-solving, and the sense of community we built together have been essential to my growth as a researcher. From tackling complex technical challenges to sharing lighter moments, your camaraderie made every day, in this journey, something to look forward to.

I must also acknowledge my friends outside of academia, who provided me with emotional support and much-needed balance during this intense period. Your encouragement, humor, and friendship helped me stay grounded and reminded me of the world beyond my research. I am deeply grateful for the joy and perspective you brought into my life.

To my parents and especially my brother Harsh, who have always believed in my abilities and have supported me unconditionally—thank you. Your sacrifices and encouragement throughout my life have been the foundation for all that I have accomplished. This achievement would not have been possible without your support and guidance.

I am also thankful to the multiple industry and academic mentors I have had, who have helped shape the way I think and tackle problems. Your guidance and perspectives have played an instrumental role in my development as a researcher and a thinker.

Lastly, I would like to acknowledge the funding agencies (Office of Naval

Research (N000141612713), Defense Advanced Research Projects Agency (DARPA) SAIL-ON program (W911NF2020009), NSF CAREER Award (2238769) to AS, independent grants from Facebook AI and Dean's Fellowship), Industry support through internship(Google, Meta, ByteDance) and institutional support (UMD - TAsip) that made this work possible. The resources and infrastructure provided by the university were critical in conducting my research, and I am deeply appreciative of the opportunities I've been given to explore, learn, and grow.

To everyone who has played a role in this journey, whether big or small, I am eternally grateful. This accomplishment is as much yours as it is mine. Thank you.

Lastly, I like to thank everyone who helped me through the journey!

Table of Contents

Dedication	ii
Acknowledgements	iii
Table of Contents	vi
List of Tables	x
List of Figures	xv
Chapter 1: Introduction	1
1.1 Internally Learnt Prior	2
1.2 Externally Learnt Prior	3
Chapter 1: Video Prediction by Modeling Videos as Continuous Multi-Dimensional Processes(CVP)	5
1.1 Introduction	5
1.2 Related Works	7
1.3 Method	9
1.4 Experiments	16
1.4.1 Datasets	17
1.4.2 Metrics	18
1.5 Setup and Results	18
1.6 Limitation	25
1.7 Conclusion	27
1.8 Broader Impact	28
1.9 Appendix	29
1.9.1 Extended derivations of Eq. (1.21)	29
1.9.2 Extended derivation for Eq. (2.2)	31
1.9.3 Training Details	32
1.9.4 Ablation Studies	33
Chapter 2: Efficient Continuous Video Flow Model for Video Prediction	36
2.1 Introduction	36
2.2 Related Works	38
2.3 Method	42
2.3.1 Video Prediction Framework	42

2.3.2	Forward and Reverse Process	43
2.3.3	Likelihood and Variational Bound	44
2.3.4	Reverse Process	45
2.3.5	Training Objective	45
2.4	Experiments	46
2.4.1	Datasets	46
2.4.2	Metrics	47
2.5	Setup and Results	48
2.6	Limitation	54
2.7	Conclusion	56
Chapter 3: Diverse Video Generation using a Gaussian Process Trigger		57
3.1	Introduction	57
3.2	Related work	61
3.3	Review of Gaussian Process	63
3.3.1	Learning and Model Selection	64
3.3.2	Scalable $\mathcal{GP}$	64
3.4	Our Approach	66
3.4.1	Frame Auto-Encoder	66
3.4.2	LSTM Temporal Dynamics Encoder	67
3.4.3	$\mathcal{GP}$ Temporal Dynamics Encoder	68
3.4.4	Training Objective	69
3.4.5	Inference Model of Diverse Video Generator (DVG)	69
3.4.6	Trigger Switch Heuristics	70
3.5	Experiments	70
3.5.1	Baselines	72
3.5.2	Metrics	73
3.5.3	Results	75
3.6	Conclusion	78
3.7	Appendix	78
3.7.1	Ablation Studies for Temporal Dynamics Encoder	78
3.7.2	Analysis: Changes in action after GP triggering	79
3.7.3	SSIM and PSNR Results	82
3.7.4	Qualitative Results	82
3.7.5	Gaussian Layer Specifics	85
3.7.6	I3D Network architecture for Action Classifier	85
Chapter 4: Video Dynamics Prior: An Internal Learning Approach for Robust Video Enhancements		87
4.1	Introduction	87
4.2	Related Works	90
4.3	Model Overview	93
4.3.1	Latent Frame predictor Network (LFPNet)	93
4.3.2	Frame Decoder Network (FDNet)	94
4.3.3	Formulation of Video Dynamics Prior	94

4.3.4	Neural Module Weights Optimization	95
4.3.5	Robustness of Video Dynamics Prior	96
4.4	Video Denoising	100
4.5	Video Frame Interpolation	103
4.6	Video Super Resolution	104
4.7	Video Object Removal	106
4.8	Limitations	107
4.9	Conclusion	108
4.10	Broader Impact	108
4.11	Appendix	109
4.11.1	Comparison with cascaded models	109
4.11.2	Ablation study	110
4.11.3	Processing time	114
4.11.4	Optimization Details	115
4.11.5	Additional Details - FDNet	117
4.11.6	Dataset Descriptions	117
Chapter 5:	Video Decomposition Prior: Editing Videos Layer by Layer	119
5.1	Introduction	119
5.2	Related Works	122
5.3	Video Decomposition Prior Framework	123
5.3.1	Modules Overview	124
5.3.2	VDP: Optimization objective for generic video decomposition	125
5.4	Video Relighting	126
5.5	Unsupervised Video Object Segmentation	129
5.5.1	Coherent Edits Propagation	134
5.6	Video Dehazing	135
5.7	Limitations of our approach	137
5.8	Conclusion	138
5.9	Appendix	138
5.9.1	Network Architecture for RGBnet and α -net	138
5.9.2	Optimization Details	139
5.9.3	UVOS	141
5.9.4	Intuition for Dehazing/Relighting Tasks	145
5.9.5	Ablation for Dehazing and Relighting tasks	146
5.9.6	Edit propagation	148
5.9.7	Limitations of our approach	149
5.9.8	Optical Flow	150
5.9.9	Optical Flow Warping Operation	150
5.9.10	Optical Flow RGB Representation	151
Chapter 6:	Future Work and Conclusion	152
6.1	Future Work	152
6.1.1	Data-Free Enhancement Techniques	152

6.1.2	Future Directions for Enhancing Temporal Coherence in Continuous Video Flow (CVF)	154
6.2	Conclusion	155
	Bibliography	157

List of Tables

1.1	Video prediction results on KTH (64×64), predicting 30 and 40 frames using models trained to predict k frames at a time. All models condition on 10 past frames on 256 test videos.	20
1.2	<i>BAIR</i> dataset evaluation. Video prediction results on BAIR (64×64) conditioning on p past frames and predicting $pred$ frames in the future, using models trained to predict k frames at a time. The common way to compute the FVD is to compare 100×256 generated sequences to 256 randomly sampled test videos. Best results are marked in <i>bold</i>	23
1.3	Quantitative comparisons on the Human3.6M dataset. The best results under each metric are marked in bold.	25
1.4	Video prediction results on UCF (128×128), predicting 16 frames. All models are conditioned on 5 past frames.	25
1.5	U-NET: We utilize Hugging face diffusers library for our U-Net implementation. We utilize ‘positional’ type for timestep embeddings. We utilize 4 layers per block. The target resolution for KTH, BAIR and Human3.6M is kept at 64×64 and 128×128 for UCF101 dataset. Additionally, we keep the number of timesteps T as 100 given our compute resources. c denotes the number of channels present in the frame. n is the number of initial context frames based on which next frame is predicted, i.e., $\mathbf{x}^{0:n} \rightarrow \mathbf{x}^{1:n+1}$	33
1.6	Comparison with baselines on sampling steps and sampling time required for BAIR robot push dataset.	34
1.7	Ablation study: Video prediction results on KTH (64×64), predicting 30 frames. All models condition on 4 past frames on 256 test videos. The method with settings marked with $*$ is reported in the main paper.	35
2.1	Video prediction results on KTH (64×64), predicting 30 and 40 frames using models trained to predict k frames at a time. All models condition on 10 past frames on 256 test videos.	49
2.2	<i>BAIR</i> dataset evaluation. Video prediction results on BAIR (64×64) conditioning on p past frames and predicting $pred$ frames in the future, using models trained to predict k frames at a time. The common way to compute the FVD is to compare 100×256 generated sequences to 256 randomly sampled test videos. Best results are marked in <i>bold</i>	50

2.3	Comparison with baselines on number of parameters, sampling steps and sampling time required for BAIR robot push dataset.	52
2.4	Quantitative comparisons on the Human3.6M dataset. The best results under each metric are marked in bold.	54
2.5	Video prediction results on UCF (128×128), predicting 16 frames. All models are conditioned on 5 past frames.	54
3.1	Quantitative results on KTH, BAIR, Human3.6M datasets. For the FVD Score , all methods use the best matching sample out of 100 random samples and lower numbers are better. For the Diversity Score , we compute the score across 50 generated samples, for 500 starting sequences, and higher numbers are better.	74
3.2	Quantitative results on KTH, BAIR, Human3.6M datasets. For the FVD Score , all the ablation methods use the best matching sample out of 100 random samples and lower numbers are better. For the Diversity Score , we compute the score across 50 generated samples, for 500 starting sequences, and higher numbers are better.	79
4.1	Quantitative video denoising results: To evaluate our method, we calculate the Avg PSNR score between the generated noise-free video and ground truth video. The σ denotes the standard deviation for the added Gaussian noise to the frames. While the λ denotes the intensity of additive Poisson noise. For all baselines and our method, the score is calculated on the DAVIS dataset [1]. Also, blue (bold) denotes the best score. Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.	100
4.2	Quantitative video frame interpolation results: For the evaluation of our method we calculate the Avg PSNR and SSIM score between the generated interpolated video frames and ground truth video. For all baselines and our method, the score is calculated on the UCF101 dataset [2]. The color blue (bold) denotes the best performance. Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.	100
4.3	Quantitative video super-resolution results: For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the generated super-resolution video and ground truth video for all baselines along with our method. The score is calculated on the standard VIMEO-90K-T dataset. Best results in the table are shown in blue (bold) . Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.	102

4.4	Quantitative Object Removal results: We compare the average PSNR and SSIM scores of our method with the existing baselines on the DAVIS dataset. The best performance is denoted by the score in blue (bold) . Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.	106
4.5	Quantitative cascaded video super-resolution(4×) results: For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the generated super-resolution video and ground truth video for all baselines along with our method. The score is calculated on the standard VIMEO-90K-T dataset. Best results in the table are shown in blue (bold) . Please note we use VSR baselines by cascading it with the video denoiser model FastDVDnet [3]. The baselines marked with * denote a data-driven baseline.	111
4.6	Quantitative cascaded video frame interpolation results: For the evaluation of our method, we calculate the Avg PSNR and SSIM score between the generated interpolated video frames and ground truth video. For all baselines and our method, the score is calculated on the UCF101 triplet dataset. Please note we use VFI baselines by cascading it with the video denoiser model FastDVDnet [3]. The color blue (bold) denotes the best performance. The baselines marked with * denote a data-driven baseline.	111
4.7	Quantitative ablations results: To evaluate our method, we calculate the Avg PSNR score between the generated noise-free video and ground truth video. blue (bold) denotes the best score.	114
4.8	Training and inference time comparison with the best baselines. Note that the inference time in the table is given as the time required to process per frame of a video.	116
4.9	We include LPIPS and FVD [4] metrics for completeness of quantitative evaluation of enhancement tasks (video denoising, object removal, and video interpolation). FVD is a deep neural network based metric that evaluates both the spatial and temporal quality of the processed video with the ground truth video. FVD requires minimum of 16 frames in a sequence for evaluation. Hence, we were unable to evaluate FVD for task like VFI and VSR on UCF101 triplet and Vimeo90KT datasets.	118
5.1	Quantitative evaluation on Video Relighting benchmarks: For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the re-lit video and ground truth video. We use the pretrain model for the baselines that are publicly made available for calculating the PSNR and SSIM scores. For all baselines and our method, the score is calculated on the SDS dataset (5). Also, blue (bold) denotes the best score.	127

5.2	Quantitative evaluation on VOS benchmarks: For the evaluation of our method, we calculate the IOU score ($\mathcal{J}$) between the generated masks and the ground truth mask. Also, blue (bold) denotes the best score. Please note that the DyStab is trained using a video dataset, while all the other methods require only a test time video sequence to segment the object from the video.	131
5.3	Quantitative evaluation on Video Dehazing benchmarks: To evaluate our method, we calculate the Avg PSNR and SSIM scores between the generated haze-free video and ground truth video. For all baselines and our method, the score is calculated on the REVIDE dataset (6).	136
5.4	RGBnet: Each convolution is followed by batch normalization and a leaky rectified linear unit (LeakyReLU; negative slope = 0.2), except for the last layer of the encoder and decoder. In the last layer of the encoder and decoder, we use the sigmoid activation function. We utilize Pytorch for our implementation.	139
5.5	α-net: Each convolution layer is followed by batch normalization and a leaky rectified linear unit (LeakyReLU; negative slope = 0.2), except for the last layer of the encoder and decoder. In the last layer of the encoder and decoder, we use the sigmoid activation function. We utilize Pytorch for our implementation.	140
5.6	Ablation Study: We utilize the DAVIS-16 dataset for performing the ablation study. We calculate the IOU score ($\mathcal{J}$) between the generated mask and the ground truth mask.	145
5.7	Quantitative evaluation on Dehazing/Relighting ablations: This Table presents the results of quantitative evaluations conducted on dehazing and relighting tasks using three different settings of our method. To assess the performance of our approach, we measured the average PSNR and SSIM scores between the processed videos and ground truth videos. For dehazing evaluations, we used the REVIDE [6] dataset, and for relighting evaluations, we used the SDSD [5] dataset.	145
5.8	Quantitative evaluation on VOS benchmarks: This Table presents the results of a quantitative evaluation of our method on VOS benchmarks. To assess the performance of our approach, we calculated the IOU score ($\mathcal{J}$) between the generated masks and the ground truth mask on the DAVIS dataset [1]. The scores were computed for all baselines and our method. It is important to note that the methods listed above the line require no annotated data for performing the VOS task, while the methods mentioned below the line require annotated data for training. The best score in each category is highlighted in blue (bold)	147

5.9	Quantitative evaluation of Video Edits Propagation:	For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the reconstructed videos and ground truth video sampled from the DAVIS-16 dataset 1. We use the code for the baselines that are publicly available to calculate the PSNR and SSIM scores. For all baselines and our method, the score is calculated by averaging the PSNR and SSIM value over a sequence length of 15 frames (frames: 2-16).	148
-----	--	--	-----

List of Figures

1.1	The figure is divided into two parts. The top portion of the figure illustrates the intermediate frames $\mathbf{x}_t$ between two consecutive frames. $\mathbf{x}, \mathbf{y}$ represents consecutive frames from a video sequence where $\mathbf{y} = \mathbf{x}^{j+1}$ and $\mathbf{x} = \mathbf{x}^j$. $\mathbf{x}^j$ denotes some frame at timestep j in the video sequence $\mathcal{V} = \{\mathbf{x}^i\}_{i=1}^N$. $\mathbf{z}$ denotes the white noise. The lower portion of the figure represents the directed graphical model considered in this work to represent the continuous video process.	6
1.2	Fig. (a) demonstrates the methodology for estimating $\mathbf{x}_t$ in a single step, showcasing the specific computational process involved. Fig. (b) details the training pipeline of our Continuous Video Process (CVP) model, where $\mathbf{x}_t$ and t are fed as inputs to the U-Net architecture, and the anticipated output is $\hat{\mathbf{y}}$, with $\hat{\mathbf{y}} = \mathbf{x}^{1:k+1}$ in this scenario. Fig. (c) provides an overview of the sampling pipeline utilized in our CVP method, illustrating the sequential steps to predict the next frame of the video sequence given the context frames.	10
1.3	Figure represents qualitative results of our CVP model on the KTH dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4^{th} predicted future frame is shown in the figure.	19
1.4	Figure represents qualitative results of our CVP model on the BAIR dataset. The number of context frames used in the above setting is two for both sequences. Every 6^{th} predicted future frame is shown in the figure.	22
1.5	Figure represents qualitative results of our CVP model on the Human3.6M dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4^{th} predicted future frame is shown in the figure.	24
1.6	Figure represents qualitative results of our CVP model on the UCF dataset. The number of context frames used in the above setting is 5 for all three sequences. Every 4^{th} predicted future frame is shown in the figure.	26

2.1	Fig. (a) represents a naive adaptation of the diffusion model for the video prediction task. Here, the sampling process always starts from a Gaussian distribution, and sampling steps are taken in the direction of conditional distribution given by $X_{j+1} X_j$. Here, X_{j+1} denotes frame at time $j + 1$. In contrast, Fig. (b) introduces our Continuous Video Flow (CVF) approach, which reimagines the problem by treating video not as a discrete sequence of frames but as a continuously evolving process. Instead of starting from a static Gaussian distribution for each sampling step, CVF models the underlying dynamics of the entire video, learning to predict changes smoothly over time. This continuous framework allows the model to better capture temporal coherence and evolution, leading to more accurate and fluid video predictions.	38
2.2	Overview of the Continuous Video Flow (CVF) framework. (a) Stage 1 depicts the auto-encoding process where input video frames X_j are passed through an encoder (Enc) and decoder (Dec) to reconstruct $\hat{X}_j$. (b) Stage 2 illustrates the forward and reverse processes in the latent space. In the forward process, latent embeddings $z_T, \dots, z_1$ are generated through a fixed process given by Eqn 2.2. The reverse process involves sampling z_{j+1} from z_j through the learned process p_θ for continuous video frame prediction. (c) The full pipeline of CVF, showing how frame X_j is passed through the encoder to obtain latent embedding z_j and $\hat{z}_{j+1}$, which is then used for decoding to obtain $\hat{X}_{j+1}$ frame.	41
2.3	Figure represents qualitative results of our CVF model on the KTH dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4 th predicted future frame is shown in the figure.	48
2.4	Figure represents qualitative results of our CVF model on the BAIR dataset. The number of context frames used in the above setting is two for both sequences. Every 6 th predicted future frame is shown in the figure.	52
2.5	Figure represents qualitative results of our CVF model on the Human3.6M dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4 th predicted future frame is shown in the figure.	53
2.6	Figure represents qualitative results of our CVF model on the UCF dataset. The number of context frames used in the above setting is 5 for all three sequences. Every 4 th predicted future frame is shown in the figure.	55
3.1	Given “person holding cup,” humans can often predict multiple possible futures (<i>e.g.</i> , “drinking from the cup” or “keeping the cup on the table.”).	58
3.2	An illustration of using $\mathcal{GP}$ variance to control sampling on-going actions <i>vs.</i> new actions.	61

3.3	An overview of the proposed Diverse Video Generator (DVG).	67
3.4	LPIPS Quantitative Results on KTH, BAIR, and Human3.6M datasets. All methods use the best matching sample out of 100 random samples. We used fixed trigger heuristic to keep trigger point for each sample the same for our approach.	72
3.5	Qualitative Results on BAIR (left), KTH (center), Human3.6M (right, top), and UCF (right, bottom) datasets. First row is the ground-truth video in each figure (with the last frame of the provided 5 frames is shown as ‘GT’). Every 5 th frame is shown.	77
3.6	Ablation results on KTH, Human3.6M and BAIR dataset using variants of temporal dynamics model in our method. We report best LPIPS metric. All methods use the best matching sample out of 100 random samples. We used fixed trigger to keep trigger point for each sample the same. On KTH, all temporal dynamics models have similar performance; and on BAIR, our LSTM model have best performance.	79
3.7	Changes in action from past frames to future frames on KTH dataset. Total of 25,000 generated videos were used to calculate percentage change shown in the above figure.	80
3.8	Quantitative results on KTH, BAIR and Human3.6M dataset. We report average SSIM and PSNR metrics. All methods use the best matching sample out of 100 random samples. We used fixed trigger to keep trigger point for each sample the same.	83
3.9	KTH dataset: Qualitative comparison of the generated video sequences (every 5 th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)	84
3.10	Qualitative results on BAIR dataset. We show the best LPIPS samples out of 100 samples for all methods.	84
3.11	Qualitative results on BAIR dataset. We show the best LPIPS samples out of 100 samples for all methods.	84
3.12	Human3.6M dataset: Qualitative comparison of the generated video sequences (every 5 th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)	84
3.13	Human3.6M dataset: Qualitative comparison of the generated video sequences (every 5 th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)	84
3.14	KTH dataset: Qualitative comparison of the generated video sequences (every 5 th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)	85

4.1	Model pipeline: We are given $X_1, \dots, X_t$ frames of a person exercising (depicted in the Fig). Here, we start by sampling $z_0 \sim \mathcal{N}(0, I)$ as an initial latent representation. Our model comprises two key modules: (i) the Frame Decoder Network (FDNet) and (ii) the Latent Frame Predictor Network (LFPNet), depicted in the figure. These modules, together form the pipeline of our VDP model, as shown in (iii). We optimize the weights of these neural modules on this specific input sequence by doing backpropagation over the loss pertaining to the task at hand.	92
4.2	(i) Video B is generated by replacing the second frame of video A. (ii) Top row depicts the video obtained during the early round of optimization of the objective given by Eq. 4.4 for video B. Further, the average Mutual information [7] of video(intermediate) is much closer to video A. (iii) The loss plot depicts the convergence of our VDP model for in five different settings.	97
4.3	Video Denoising: Frame from a video in the DAVIS dataset denoised using different approaches. (a) Frame corrupted with additive poisson noise of intensity 25 (relative to intensity range [0-255]). Figure (b) depicts the denoised frames generated by FastDVDnet [3] method. Figure (c) depicts the denoised frame generated by UDVD [8] method. Figure (d) depicts the denoised frame generated by M2F2 [9] method. Figure (e) depicts the denoised frames generated by our method. Figure (f) shows the clean ground truth frames.	99
4.4	Video Frame Interpolation: In order to compare our method to existing baseline methods, we conducted visual comparisons using three frames for each method. We introduced Gaussian noise ($\sigma = 15$) in the second frame and tasked all methods with interpolating frames between them. It can be observed that our method not only denoises the second noisy input frame but also generates denoised intermediate frames. This is in contrast to other baseline methods that interpolate noise along with the clean signal.	101
4.5	Video Super Resolution: Comparison of results of the ‘Cameraman’ sequence from VIMEO-90K-T dataset. Gaussian noise of std $\sigma = 5$ is added to the lower-resolution input frames. (a) High-resolution noise-free ground truth frame. (b) Patch from higher resolution frame generated by BiCubic method. (c) Patch from higher resolution generated by BasicVSR++ [10]. (d) Patch from higher resolution frame generated by EDVR [11] method. (e) Patch from the higher resolution by Ours VDP method. (f) Patch from higher resolution ground truth frame. Note the noise suppression and details of the camera in the patch generated by our method compared to baselines.	102

4.6	Video Object Removal. In the above figure, we describe a video object removal task. The input in the figure represents the original video sequence frame without any alterations. The InterVI [12], FGVC [13], and LongVI [14] in the figure represent the frames generated by removing the object “car” from the video sequence. The rightmost frame depicts the removal of the object from the video frame by our method. It can be observed that our method do not create hallucinations. . . .	105
4.7	Video Super Resolution: Comparison of results of the ‘Cameraman’ sequence from VIMEO-90K-T dataset. Gaussian noise of std $\sigma = 5$ is added to the lower-resolution input frames. (a) Bicubic extrapolation of noisy low-resolution frame (b) Bicubic extrapolation of denoised low-resolution frame. (c) Patch from higher resolution generated by BasicVSR++ [10] over the denoised low-resolution frame. (d) Patch from higher resolution frame generated by EDVR [11] over the denoised low-resolution frame. (e) Patch from the higher resolution by Ours VDP method. (f) Patch from higher resolution ground truth frame. We use standard video denoiser FastDVDNet [3] to denoise the input low-resolution video frames. Based on the figure, it appears that the denoiser used in the first stage of the model removes vital information from the lower-resolution frames, which leads to errors in the second stage. Our VDP model, on the other hand, preserves more details compared to the baselines.	111
4.8	Video Denoising (Ablation): Frame from a video in the DAVIS dataset denoised using loss functions. (a) Frame corrupted with additive poisson noise of intensity 25 (relative to intensity range [0-255]). Figure (b) depicts the denoised frames generated by only $\mathcal{L}_{\text{rec}}$. Figure (c) depicts the denoised frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$. Figure (d) depicts the denoised frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$. Figure (e) depicts the denoised frames generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{var}}$. Figure (f) shows the clean ground truth frames.	113
4.9	Video Super Resolution (Ablation): Comparison of results of the ‘Cameraman’ sequence from VIMEO-90K-T dataset. (a) Bicubic extrapolation of low-resolution frame. (b) Patch from higher resolution generated by utilizing only the reconstruction loss given by Eqn. 1. (c) Patch from higher resolution frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$ (d) Patch from the higher resolution frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$. (e) Patch from the higher resolution frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{var}}$ (f) Patch from higher resolution ground truth frame.	114
4.10	Video Frame Interpolation (Ablation): In order to compare our method with different loss functions, we conducted a visual comparison. (a) Patch generated by utilizing only the reconstruction loss given by Eqn. 1. (b) Patch generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$ (c) Patch generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$. (d) Patch generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{var}}$	115

4.11	Video Object Removal (Ablation). In the above figure, we describe a video object removal task. The GT in the figure represents the original video sequence frame without any alterations. (a) Patch generated by utilizing only the reconstruction loss given by Eqn. 1. (b) Patch generated utilizing the loss given by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$ (c) Patch generated by utilizing the loss given by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$. (d) Patch generated by utilizing the loss given by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{var}}$	115
5.1	Visual representation of video edits obtained using VDP. The first row demonstrates a foreground manipulation example that leverages an object mask as the decomposition guide. This object mask is obtained by performing Unsupervised Video Object Segmentation (UVOS), which is a downstream task of video decomposition and is achieved using our proposed framework. Our approach effectively separates the foreground objects and background in the video sequence, enabling us to perform object manipulation. The second row shows the result of our approach for video dehazing, where our method effectively removes the haze from the scene. Finally, the third row showcases the effectiveness of our approach for video relighting, where our method effectively changes the lighting of the scene. Our proposed approach outperforms the state-of-the-art methods for the latter two tasks, highlighting the efficacy of our unified framework for video decomposition.	121
5.2	VDP for relighting a video sequence. In this pipeline, the input video frame t is fed into a shallow U-Net denoted by $f_{\text{RGB}}^{(1)}$. While the flow-RGB is given as input to a separate shallow U-Net network denoted by $f_{\alpha}^{(1)}$. The intermediate output of $f_{\text{RGB}}^{(1)}$ is a re-lit version of the input frame t. While the output of $f_{\alpha}^{(1)}$ is transmission maps or Tmap denoted by $1/A_t$. It is important to note that γ^{-1} is also treated as a trainable parameter in the pipeline. After obtaining the reconstructed frame t , we apply reconstruction and warping losses respectively to optimize $f_{\text{RGB}}^{(1)}$, $f_{\alpha}^{(1)}$, γ^{-1}	124
5.3	Qualitative evaluation on Video Relighting benchmark: We compare the re-lit result of our method with the baselines on the SDS (5) dataset. We compare our method against both the image and video baselines. ZeroDCE++ (15) method is a image-based baseline while SDS (5) and Stablelve (16) are video baselines. Please note that we have utilized the pretrained models of all the baselines to obtain qualitative results. Methods SDS and Stablelve require training on the external dataset, while our approach operates directly on the low-light video sequence.	128

- 5.4 **Our Framework:** In this figure, we present the pipeline for the task of decomposing the video into two different components. For the above configuration, the input video frame t is fed into two different shallow U-Nets denoted by $f_{\text{RGB}}^{(1)}$ and $f_{\text{RGB}}^{(2)}$ respectively. We obtain two different intermediate components of the input frame t in the form of layer 1 and layer 2. Additionally, we process forward flow-RGBs using a separate shallow U-Net network denoted by $f_{\alpha}^{(1)}$. This network outputs α compositing maps, which are then used for blending both layers respectively to reconstruct the original input frame t . After the compositing, we apply various losses (reconstruction loss and regularization loss) depending on the task at hand. **(Right) Illustration of Flow Similarity loss:** we define this loss over flow-RGB image (F^{RGB}). This loss ensures that pixels with similar motions are grouped together. **Illustration of Reconstruction layer loss:** We define this loss over the masked object layers and masked input frame X_t . This loss ensures that the masked object layer such as $M_t f_{\text{RGB}}^{(1)}$ has the same appearance as the masked input frame $M_t X_t$. . 129
- 5.5 **Qualitative evaluation on VOS benchmark:** We compare the mask from our method with the baselines on DAVIS-16 (1) dataset. We compare our method against the baselines DS (17) and DyStab (18). Please note the above masks (exception: DoubleDIP) for baselines are pre-computed masks provided by the authors of the papers. . . . 131
- 5.6 **Edits propagation:** First column in the figure represents the input frame to be edited. Second column represents the edits performed. Next columns in Fig. (a) represent the qualitative comparison of propagated edits with baselines, while Fig. (b) represents propagated edits throughout the video. From Row 1, 2 - it can be observed that the quality of our framework's propagated edited decals is much better as compared to the baselines. The legs of the camel(Row 1) and parachute bag shape(Row 2) are preserved in our method's edit propagation. Row 3 - It can be observed that consistent edits in a particular layer can be performed throughout the video. In this case, we show an example of style transfer being performed on the swan. Row 4 - we can selectively replace the background layer and propagate the edits through the video. 133

5.7	Qualitative evaluation on Video Dehazing benchmark: We compare the dehazed result of our method with the baselines on the REVIDE (6) dataset. We compare our method against both the image and video baselines. MSBDN (19) method is an image-based baseline while EDVR (11), DoubleDIP (20) and CG-IDN (6) are video baselines. Please note that both the EDVR and CG-IDN methods are trained using the training data of the REVIDE dataset. In contrast, our approach operates directly on the hazy video sequence. As can be seen from the figure, our approach to dehazing produces a color palette in the dehazed frame that closely resembles the color palette in the ground truth frame.	137
5.8	Comparison with pretrained image segmentation model: The examples presented in the figure above are samples from the MoCA dataset [21]. In comparison to MaskRCNN [22], a pretrained image segmentation model trained on the COCO dataset, our model is more robust in handling unfamiliar objects in the scene. The figure shows some examples where MaskRCNN fails to segment the unfamiliar object correctly, while our approach produces much better segmentation results.	141
5.9	(a) Segmentation in variable lighting conditions throughout the scene. We compare the mask from our method with the baselines Deformable sprites [17] and MG [23] on frames of three video sequences. It can be observed from the above figure methods such as Deformable-sprites and MG are more prone to lighting changes as opposed to our method. (b) Decomposition into multiple layers We demonstrate an example where the number of layers L is set to both $L = 2$ and $L = 3$. $L = 2$ results in foreground and background layers separation, while $L = 3$ results in the separation of individual person in different object layers.	143
5.10	Ablation Study: We study the effect of using only a subset of the loss function for the video object segmentation task.	144
5.11	Ablation Study: This figure presents the results of an ablation study where we analyzed the impact of using FlowRGB, Gaussian Noise, and RGB frame as input to the α -net. Row (a) shows a qualitative comparison for relighting tasks, where we evaluated three different settings for the α -net. Row (b) presents a comparison for the dehazing task.	144

Chapter 1: Introduction

The phrase coined by Henrik Ibsen "a picture is worth a thousand words" suggests that a single image can convey a lot of information. If we extrapolate this idea to videos, which are essentially moving images, a video could be worth significantly more than a thousand words! In fact, this definition can be extrapolated for videos which suggests a one minute video ($1000 \times 60 \times 24$ for a 24 fps video) is equivalent to around 1.5 million words! This is because videos can convey complex information, emotions, and stories in a more engaging and dynamic way than static images or text.

Recently, there has been a significant uptick in the generation of video content, especially following the advent of short-form content. This surge can be attributed to the availability of high-quality cameras and storage on handheld devices, as well as the push from companies such as Meta, TikTok, YouTube, and Netflix for content creators to generate new content for their platforms. However, capturing high-quality video poses a significant challenge due to various factors. For example, noise, haze, or low lighting conditions may be introduced due to non-ideal settings for capturing a scene. Additionally, sometimes cameras may not be of sufficient quality to capture a scene at the required resolution or frame rate. Furthermore, creating or editing

existing videos is a professional task that requires artistic levels of expertise, which can be difficult to acquire.

To propose a solution for such ill-posed tasks (there is insufficient information to uniquely determine a solution) priors play a pivotal role. When these priors are derived from a internal statistics of a video we termed them as internally learnt prior. Alternatively when these priors are learned using external data corpus we termed them as externally learnt priors. In our proposed direction we explore ways to exploit both these type of priors for different tasks and model videos to perfection. We propose four different types of priors namely 1st) video dynamics prior 2nd) video decomposition prior and 3rd) diverse video generation prior and 4th) continuous video process prior.

1.1 Internally Learnt Prior

In the realm of video modeling, internally learnt priors represent a fundamental concept where prior knowledge or assumptions are derived solely from the inherent statistical properties of a single video, without reliance on external sources or supplementary training data. These priors play a crucial role in identifying significant patterns, structures, and regularities intrinsic to the video data, encompassing phenomena such as spatio-temporal coherence between frames.

The appeal of internally learnt priors lies in their effectiveness in scenarios characterized by scarcity or unavailability of external data such as for the task of video dehazing/relighting. However, it is essential to recognize the potential limitations of

internally learnt priors, particularly regarding their ability to generalize to unseen data beyond the explicit scope of a single video. Hence, careful consideration of these factors is essential in the utilization of internally learnt priors within the dynamic landscape of video modeling. Two of our proposed approaches namely, video dynamics prior and video decomposition prior are internally learnt prior.

1.2 Externally Learnt Prior

Externally learnt priors in the context of video modeling refer to prior knowledge or assumptions about the dataset that are derived from external sources or supplementary training data. Unlike internally learnt priors, which rely solely on the intrinsic statistical properties of the query video, externally learnt priors draw upon a broader array of data sources to inform the modeling process.

In the realm of generative modeling, externally learnt priors hold significant promise and utility. One of their key advantages lies in their ability to generalize beyond the statistics of the query video. By leveraging information gleaned from diverse external datasets [2, 24, 25, 26], these priors can capture a more comprehensive understanding of the underlying dynamics and characteristics of the video domain.

Furthermore, externally learnt priors can facilitate the synthesis of novel content and open up new avenues for artistic expression, storytelling, and content creation within the realm of video generation. Moreover, externally learnt priors offer valuable insights and guidance in scenarios where the query video may be limited in size, scope, or diversity. By supplementing the inherent information present in the query

video with knowledge acquired from external sources, these priors can enrich the modeling process and enhance the generalization capabilities of generative models.

In summary, externally learnt priors play a pivotal role in generative modeling by harnessing knowledge from external datasets to inform and enrich the modeling process. Their ability to generalize beyond the statistics of the query video enables models to produce outputs that are diverse, creative, and expressive, opening up new possibilities for content generation and artistic exploration in the realm of video modeling.

Chapter 1: Video Prediction by Modeling Videos as Continuous Multi-Dimensional Processes(CVP)

1.1 Introduction

In the evolving landscape of machine learning and generative models, particularly in the domain of video representation, there exists a pivotal challenge in adequately capturing the dynamic transitions between consecutive frames. In this paper, we introduce a novel approach to video representation that treats the video as a continuous process in multi-dimensions. This methodology is anchored in the observation that transitions between consecutive frames in a video do not uniformly contain the same amount of motion. Modeling these transitions with a single-step process often leads to suboptimal quality in sampling. Our method, therefore, involves multiple predefined steps between two consecutive frames, drawing inspiration from recent advancements in diffusion models for image data. This multi-step diffusion process has been instrumental in better modeling image data, and we aim to extend this success to video data.

Previous efforts in video modeling with diffusion models have tended to approach videos as a series of images, generating separate volumes of video frame

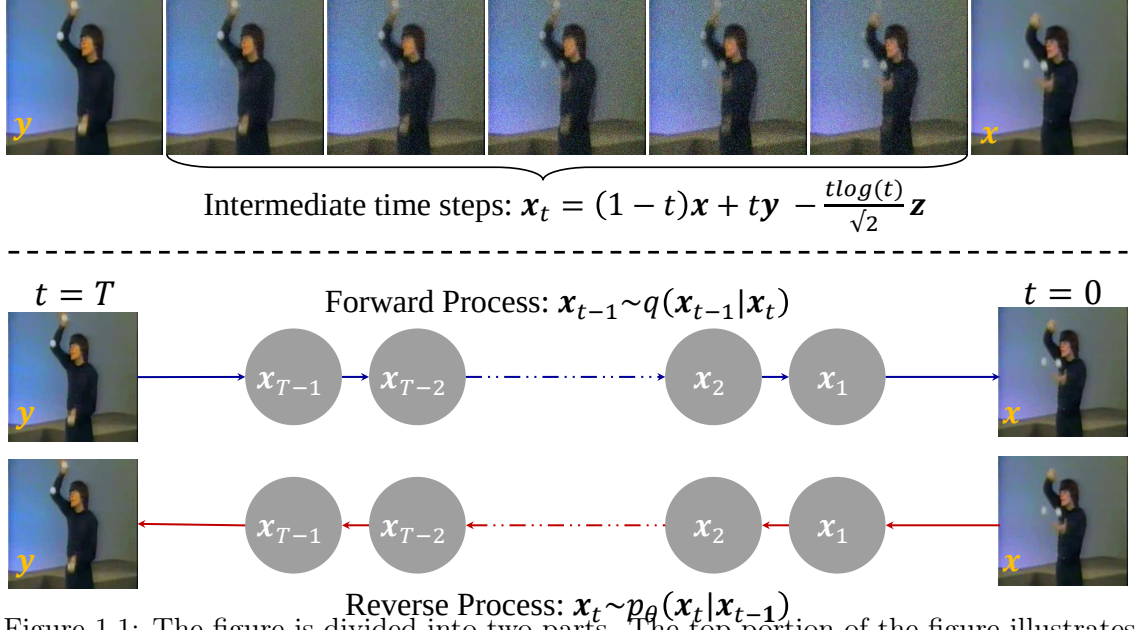


Figure 1.1: The figure is divided into two parts. The top portion of the figure illustrates the intermediate frames $\mathbf{x}_t$ between two consecutive frames. $\mathbf{x}, \mathbf{y}$ represents consecutive frames from a video sequence where $\mathbf{y} = \mathbf{x}^{j+1}$ and $\mathbf{x} = \mathbf{x}^j$. $\mathbf{x}^j$ denotes some frame at timestep j in the video sequence $\mathcal{V} = \{\mathbf{x}^i\}_{i=1}^N$. $\mathbf{z}$ denotes the white noise. The lower portion of the figure represents the directed graphical model considered in this work to represent the continuous video process.

sequences and applying external constraints such as applying temporal attention to maintain the temporal coherence. We argue that this approach overlooks the inherent continuity in video data, which can be more naturally conceptualized as a continuous multi-dimensional process. Our proposed method defines this continuous process, beginning with two consecutive frames from a video sequence as endpoints this can be observed in Fig. 5.1. We delineate the forward process through interpolation between these endpoints, with a predefined number of steps guiding the transition from one point to another. To ensure the existence of $p(\mathbf{x}_t)$ at all points, we introduce a novel noise schedule that applies zero noise at both endpoints.

We approximate each step between these endpoints using a Gaussian distribution, following the assumptions made in diffusion models for images by the

paper [27, 28, 29, 30]. In defining this forward process, we also lay the groundwork for estimating a reverse process. This paper presents a novel lower variational bound for estimating this reverse process.

To summarize, our contribution in this work is as follows:

- We introduce a novel approach for representing videos as multi-dimensional continuous processes.
- We derive a novel variational bound that efficiently estimates the reverse process in our proposed ‘Continuous Video Process (CVP)’ model.
- Our method employs a unique noise schedule for the continuous video process, characterized by zero noise at both endpoints, ensuring the existence of $p(\mathbf{x}_t)$ at all intermediate timesteps.
- We demonstrate the efficacy of our approach through state-of-the-art results in video prediction tasks across four different datasets namely, KTH action recognition, BAIR robot push, Human3.6M, and UCF101 datasets.

1.2 Related Works

Understanding and predicting future states based on observed past events is a cornerstone challenge in the domain of video understanding, crucial for applications where capturing the inherent multi-modality of future states is vital, such as in autonomous vehicles. Early methods in this field, as noted by Yuen et al.[31] and Walker et al.[32], primarily focused on matching past frames within datasets to

extrapolate future states, although these predictions were constrained to either symbolic trajectories or directly retrieved future frames. The advent of deep learning has significantly propelled advancements in this area. One of the seminal works by Srivastava et al.[33] leveraged a multi-layer LSTM network for deterministic representation learning of video sequences. Subsequent studies [34, 35, 36, 37, 38, 39, 40], have expanded the scope of this research by constructing models that account for the stochastic nature of future states, marking a notable shift from earlier deterministic approaches.

Recent research in this domain has explored both implicit and explicit probabilistic modeling approaches. Implicit probabilistic modeling, typified by GAN [41]-based models, has a substantial history. Nonetheless, these models [42, 43, 44] often grapple with training stability issues and mode collapse (where model only focuses on a few modes in the dataset) issues. On the other hand, explicit probabilistic modeling for video prediction encompasses a range of methodologies, including Variational Autoencoders (VAEs) [45], Gaussian processes, and Diffusion models. VAE-based video prediction methods [42, 46, 47] tend to average results to align with all potential future scenarios, which undermines the fidelity of predictions. Gaussian process-based models [48, 49] exhibit proficiency with smaller datasets but encounter scalability issues owing to matrix inversion limitations when calculating training likelihood. While workarounds exist, they tend to compromise result fidelity.

Recent advancements in diffusion models [50, 51, 52, 53] have positioned them as the preferred choice for video prediction tasks. These multi-step models offer superior sample quality and are resilient to mode collapse. However, even with such

lucrative advantages, modeling videos with these models tends to have downsides. Majorly methods falling under this category enforce temporal consistency using artificial external constraints such as the introduction of temporal attention blocks. This might be effective but comes at a cost of significant computing power.

Another class of popular video prediction models is hierarchical prediction [54, 55, 56, 57, 58] models. These models are multistage models that decompose the problems into two stages. They first predict a high-level structure of a video, like a human pose, and then leverage that structure to make predictions at the pixel level. These models generally require additional annotation for the high-level structure for training, unlike ours that predicts future frames utilizing only the pixel-level information of context frames.

We also want to highlight some very recent works like InDI [59], and Cold diffusion [60] that provide an alternate approach to denoising diffusion models that is similar to our approach. However, their works only explored such formulation for image-based computational photography and image generation tasks.

1.3 Method

Instead of introducing noise iteratively to the frames until they conform to a Gaussian distribution, and adopting a reverse process such as denoising diffusion, a commonly employed technique for video prediction, we introduce a novel model category designed to depict videos as continuous processes. This section delves into the modeling of this continuous video process.

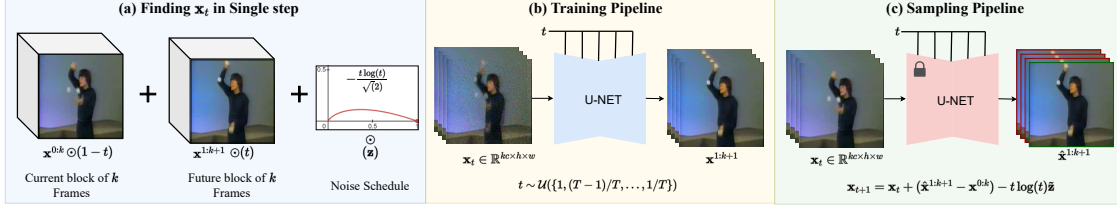


Figure 1.2: Fig. (a) demonstrates the methodology for estimating $\mathbf{x}_t$ in a single step, showcasing the specific computational process involved. Fig. (b) details the training pipeline of our Continuous Video Process (CVP) model, where $\mathbf{x}_t$ and t are fed as inputs to the U-Net architecture, and the anticipated output is $\hat{\mathbf{y}}$, with $\hat{\mathbf{y}} = \mathbf{x}^{1:k+1}$ in this scenario. Fig. (c) provides an overview of the sampling pipeline utilized in our CVP method, illustrating the sequential steps to predict the next frame of the video sequence given the context frames.

Suppose we have a video sequence denoted by $\mathcal{V} = \{\mathbf{x}^t\}_1^N$ where $\mathbf{x}^j \in \mathbb{R}^{c \times h \times w}$ is the frame at the timestep j . We represent this video sequence as a continuous process. The intermediate frames between $\mathbf{x} = \mathbf{x}^j$ and $\mathbf{y} = \mathbf{x}^{j+1}$ are given by the following equation.

$$\mathbf{x}_t = (1-t)\mathbf{x} + t\mathbf{y} - \frac{t \log(t)}{\sqrt{2}}\mathbf{z} \quad (1.1)$$

Here, $\mathbf{z} \sim \mathcal{N}(0, I)$ denotes the white noise. From the above Eqn, it can be seen that at $t = 0$, we get the frame $\mathbf{x}^j$ and at $t = 1$, we get the frame $\mathbf{x}^{j+1}$. We utilize this continuous process of evolving $\mathbf{x}^j \rightarrow \mathbf{x}^{j+1}$ given by Eqn. 2.1 and derive both the forward and reverse processes. For defining the forward process, we take steps in the direction $t : T \rightarrow 0$ instead of the other way, which happens in denoising diffusion process [27]. The reason for this is we want the reverse process to start from past frame $\mathbf{x}$ and according to the Eqn. 2.1 $\mathbf{x}_t = \mathbf{x}$ at $t = 0$.

We can write the forward process, i.e., going from the start point $\mathbf{y}$ at $t = T$ to endpoint $\mathbf{x}$ at $t = 0$,

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + (\mathbf{y} - \mathbf{x})\Delta t - t \log(t)\mathbf{z} \quad (1.2)$$

From the above equation, we can write the posterior for the forward process as $q(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{x}, \mathbf{y}) = \mathcal{N}(\mathbf{x}_{t+1} : \tilde{\mu}(\mathbf{x}_t, \mathbf{x}, \mathbf{y}), g^2(t)I)$. Where $g(t) = -t \log t$. The whole derivation is provided in the appendix.

For modeling our video diffusion process, we like to model the likelihood function $p_\theta(\mathbf{x}_T) := \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{0:T-1}$ and minimize the negative log-likelihood to obtain the best fit for our model. Here, $p_\theta(\mathbf{x}_{0:T})$ is the probability of the reverse process, and it is defined as a Markov chain with learned Gaussian transitions starting at $p(\mathbf{x}_0) = p_{data}(\mathbf{x})$. Important note about the notations $\mathbf{x}_0, \mathbf{x}_T$, unless specified consider $\mathbf{x}_0 = \mathbf{x}$ and $\mathbf{x}_T = \mathbf{y}$ where $\mathbf{x}$ is the frame in the video sequence at j^{th} position and $\mathbf{y}$ is the frame at $(j+1)^{th}$ position. One important assumption about the continuous video process is we assume the transition between the frames $\mathbf{x}$ and $\mathbf{y}$ to follow Markov chain, i.e., the current state at timestep t only depends on the previous state at timestep $t-1$. Leveraging this assumption we can define the reverse process as follows,

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_0) \prod_{t=1}^T p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad (1.3)$$

where, $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_\theta(\mathbf{x}_{t-1}, t-1), \boldsymbol{\Sigma}_\theta(\mathbf{x}_{t-1}, t-1))$. We are interested in

learning the reverse process to perform our video prediction task.

The forward process or the diffusion process is a fixed Markov chain that gradually transforms the frame $\mathbf{y}$ to frame $\mathbf{x}$.

$$q(\mathbf{x}_{0:T-1}|\mathbf{x}_T) := \prod_{t=1}^T q(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad (1.4)$$

Training is performed by minimizing the variational bound on the negative log-likelihood.

$$\mathbb{E}[-\log p_\theta(\mathbf{x}_T)] \leq \mathbb{E}_q \left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{0:T-1}|\mathbf{x}_T)} \right] \quad (1.5)$$

$$\leq \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t)} \right] \quad (1.6)$$

$$=: L(\theta) \quad (1.7)$$

This variational bound can be simplified to the following,

$$L = \mathbb{E}_q \left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{0:T-1}|\mathbf{x}_T)} \right] \quad (1.8)$$

$$= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t)} \right] \quad (1.9)$$

$$= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) \right] \quad (1.10)$$

$$- \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0, \mathbf{x}_T)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_T)} \right] \quad (1.11)$$

$$= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right] \quad (1.12)$$

$$- \log \frac{q(\mathbf{x}_T|\mathbf{x}_0, \mathbf{x}_T)}{q(\mathbf{x}_0|\mathbf{x}_0, \mathbf{x}_T)} \right] \quad (1.13)$$

$$= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right] \quad (1.14)$$

$$= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) \right] \quad (1.15)$$

$$- \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \cdot \frac{p(\mathbf{x}_0|\mathbf{x}_{t-1})}{p(\mathbf{x}_0|\mathbf{x}_t)} \right] \quad (1.16)$$

$$= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right] \quad (1.17)$$

$$- \log \frac{p(\mathbf{x}_0|\mathbf{x}_0)}{p(\mathbf{x}_0|\mathbf{x}_T)} \right] \quad (1.18)$$

$$= \mathbb{E}_q \left[-\log \frac{p(\mathbf{x}_0)}{p(\mathbf{x}_0|\mathbf{x}_T)} - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right] \quad (1.19)$$

$$(1.20)$$

Both $\mathbf{x}_0$ and $\mathbf{x}_T$ are observed variable hence, we ignore the first term in the RHS. We focus on the second term for training the parameters for our CVP models. Therefore,

the resulting loss function becomes,

$$L(\theta) =: \sum_{t \geq 1} D_{\text{KL}}(q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}, \mathbf{y}) \parallel p_{\theta}(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0)). \quad (1.21)$$

In the above Eqn, the KL divergence term utilizes the comparison of $p_{\theta}(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x})$ with forward process posterior term, which is tractable under the process given by Eqn. 2.2. The forward process posterior term is given by

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}, \mathbf{y}) = \mathcal{N}(\mathbf{x}_t : \tilde{\mu}(\mathbf{x}_{t-1}, \mathbf{x}, \mathbf{y}), g^2(t)I) \quad (1.22)$$

where, $\tilde{\mu}(\mathbf{x}_t, \mathbf{x}, \mathbf{y}) = \mathbf{x}_t + (\mathbf{y} - \mathbf{x})$ and $g(t) = -t \log(t)$. Consequently, all KL divergences in Eqn. 1.21 are comparisons between Gaussians, so they can be calculated in a Rao-Blackwellized fashion with closed-form expressions instead of high-variance Monte Carlo estimates. It is important to note while deriving the Eqn. 1.21, we ignore some terms that purely involve the forward process posteriors as q has no learnable parameters, so such terms are constants during training.

Now we discuss our choices in $p_{\theta}(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}) = \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_{\theta}(\mathbf{x}_{t-1}, t-1, \mathbf{x}), \boldsymbol{\Sigma}_{\theta}(\mathbf{x}_{t-1}, t-1, \mathbf{x}))$ for $1 < t \leq T$. First, we set $\boldsymbol{\Sigma}_{\theta}(\mathbf{x}_{t-1}, t-1) = g^2(t)I$ to untrained time dependent constants. Experimentally, the choice of $g(t) = -t \log(t)$ works the best. This noise function has an interesting property that noise is absent both at the start and end points, i.e., $g(t) = 0 \quad \forall t = \{0, 1\}$.

Second, to represent the mean $\boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t, \mathbf{x})$, we propose a specific parameterization motivated by the forward process posterior given by Eqn. 1.22. With

$p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}) = \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_\theta(\mathbf{x}_{t-1}, t-1, \mathbf{x}), g^2(t)\mathbf{I})$, we can write:

$$L(\theta) := \mathbb{E}_q \left[\frac{1}{2g^2(t)} \|\tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}, \mathbf{y}) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t, \mathbf{x})\|^2 \right] + C \quad (1.23)$$

where C is a constant that does not depend on θ . So, we see that the most straightforward parameterization of $\boldsymbol{\mu}_\theta$ is a model that predicts $\tilde{\boldsymbol{\mu}}_t$, the forward process posterior mean.

However, we can simplify Eqn. 1.23 further and obtain a very simple training loss objective by delving in the term $\tilde{\boldsymbol{\mu}}$. We further parameterize the term $\boldsymbol{\mu}_\theta$ as follows,

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t, \mathbf{x}) = \mathbf{x}_t + (\mathbf{y}_\theta(\mathbf{x}_t) - \mathbf{x}) \quad (1.24)$$

When we substitute this $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t, \mathbf{x})$ parameterization in the Eqn. 1.23 we get the simplified version of the loss $L(\theta)$ as follows,

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t, \mathbf{x}_t} \left[\frac{1}{2g^2(t)} \|\mathbf{y} - \mathbf{y}_\theta((\mathbf{x}_t, t))\|^2 \right] \quad (1.25)$$

For training the video prediction model utilizing the above Eqn. 2.9 we obtain the $\mathbf{x}_t$ as a function of t by leveraging the Eqn. 2.1. The following equation gives a more generic form of the final loss function utilized to train the video prediction model,

$${}_\theta \mathbb{E}_{t, \mathbf{x}, \mathbf{y}} \left[\frac{1}{2g^2(t)} \left\| \mathbf{y} - \mathbf{y}_\theta((1-t)\mathbf{x} + t\mathbf{y} + \frac{g(t)}{\sqrt{2}}\mathbf{z}, t) \right\|^2 \right] \quad (1.26)$$

The whole training and sampling pipeline is described in the training Alg. 4, sampling Alg. 3 and depicted in Fig. 2.2.

Algorithm 1 Training of CVP model

```

1: repeat
2:    $\mathbf{x}, \mathbf{y} \sim q_{\text{data}}(\mathbf{x}, \mathbf{y})$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
        $\nabla_{\theta} \frac{1}{2g^2(t)} \|\mathbf{y} - \mathbf{y}_{\theta}((1-t)\mathbf{x} + t\mathbf{y} - (t \log(t)/\sqrt{2})\mathbf{z}, t)\|^2$ 
6: until converged

```

Algorithm 2 Sampling Algorithm

```

1:  $\mathbf{x} \sim q_{\text{data}}(\mathbf{x})$ 
2:  $\mathbf{x}_0 = \mathbf{x}$ 
3:  $d = \frac{1}{N}$ , Here  $N$  denotes number of steps.
4: for  $t = 1, \dots, N$  do
5:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}d)$  if  $t > 1$ , else  $\mathbf{z} = \mathbf{0}$ 
6:    $\mathbf{x}_{t+1} = \mathbf{x}_t + (\hat{y}(\mathbf{x}_t, t) - \mathbf{x})d - t \log(t)\mathbf{z}$ 
7: end for
8: return  $\mathbf{x}_T$ 

```

1.4 Experiments

Video prediction task can be defined as given a few context frames, the model has to predict the subsequent future frames. In this section, we empirically demonstrate that our approach yields superior results in modeling the video prediction task.

1.4.1 Datasets

We chose 4 different types of datasets to demonstrate the efficacy of our approach. These are standard benchmarks for video prediction tasks. Dataset lists include KTH action recognition dataset [61], BAIR robot pushing dataset [62], Human3.6M [63] and UCF101 [2] datasets. Training and architecture-specific details about the approach are included in the appendix.

KTH Action Recognition Dataset. The KTH action dataset [61] consists of video sequences of 25 people performing six different actions: walking, jogging, running, boxing, hand-waving, and hand-clapping. The background is uniform, and a single person is performing actions in the foreground. The foreground motion of the person in the frame is fairly regular. The frames in the video for this dataset consist of a single channel. The spatial resolution of the frames in the video is downsampled to the size of 64×64 .

BAIR pushing Dataset. The BAIR robot pushing dataset [62] contains the videos of table mounted sawyer robotic arm pushing various objects around. The BAIR dataset consists of different actions given to the robotic arm to perform. The spatial resolution of the frames in the video is kept to be 64×64 .

Human3.6M Dataset. Human3.6M [63] dataset consists of 10 subjects performing 15 different actions. The pose information from the dataset was not used in predicting next frame. The background is uniform, and a single person is performing actions in the foreground. The foreground motion of the person in the frame is fairly regular. The frames in the video for this dataset consist of ‘RGB’ channels. The spatial

resolution of the frames in the video is downsampled to the size of 64×64 .

UCF101 Dataset. This dataset [2] consists of 13,320 videos belonging to 101 different action classes. The video seems to have a variety of backgrounds and the frames of the video have three channels, namely ‘RGB’. We reshape the resolution of frames from the original size of 320×240 down to 128×128 for our video prediction tasks. The downsampling is done utilizing the bicubic downsampling.

1.4.2 Metrics

We primarily use the FVD [64] metric to determine the best-performing baseline when evaluating a video prediction task. FVD metric evaluates a baseline on both terms, the reconstruction quality and diversity of the generated samples. FVD is calculated as the frechet distance between the I3D embeddings of generated video samples and real samples. The I3D network used for obtaining the embeddings for real and generated video is trained on the Kinetics-400 dataset.

1.5 Setup and Results

Below, we describe in detail how the setup for our experiment looks compared to baselines. We also showcase our findings about the performance of our method and comparison to baselines in this section.

KTH action recognition dataset: For this dataset, we adhered to the baseline setup [50], which utilizes the first 10 frames as context frames. In baseline setup, these 10 frames are utilized to predict the subsequent 30 and 40 frames. A notable

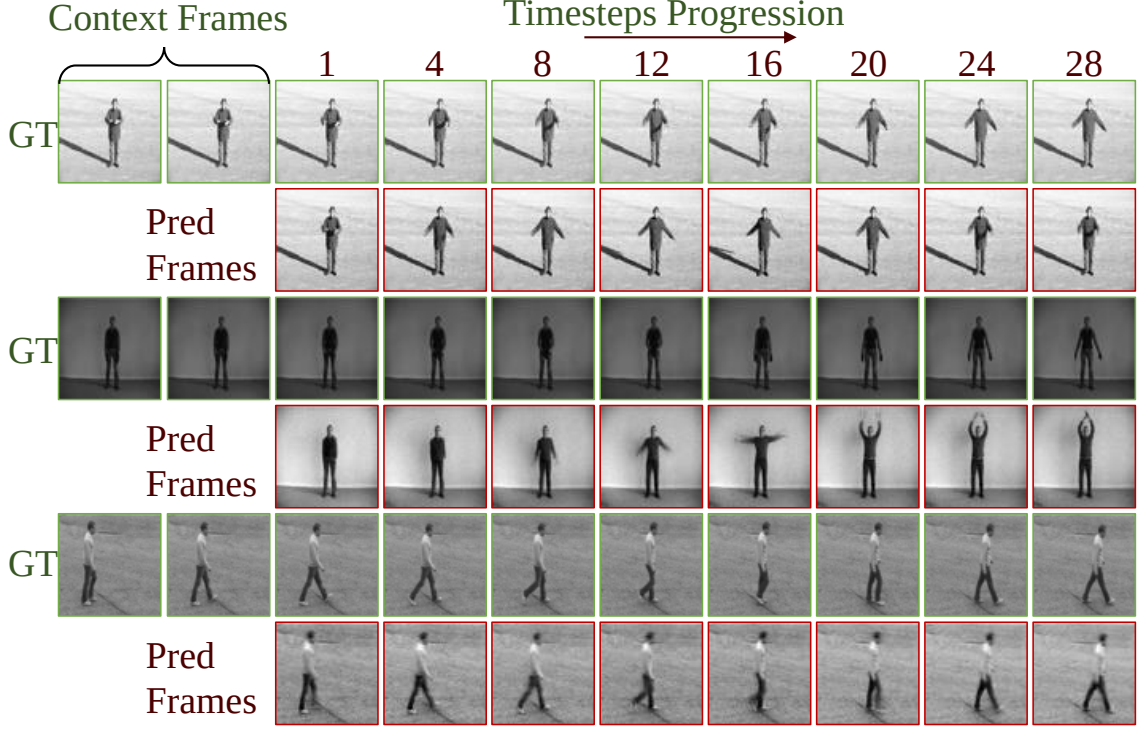


Figure 1.3: Figure represents qualitative results of our CVP model on the KTH dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4th predicted future frame is shown in the figure.

aspect of our experiment is we only used the last 4 frames from this sequence of 10 frames as context frames in our CVP model, while disregarding the information in the remaining 6 frames. This decision was taken to maintain consistency with the experimental setups used in prior baseline methodologies. The outcomes of this evaluation are summarized in Table. 2.1.

It can be observed from the Table. 2.1, our model’s unique approach requires a significantly reduced number of frames for training. Contrary to other methods that train on an additional set of k frames ($10[\text{context frames}] + k[\text{future frames}]$), our model uses just one frame (effectively $4[\text{context frames}] + 1[\text{future frames}]$). We employ the 4 context frames to predict the immediate next frame and then autoregressively generate either 30 or 40 frames, depending on the evaluation requirement. This

KTH [10 $\rightarrow$ #pred; trained on k]	k	#pred	FVD $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$
SVG-LP [46]	10	30	377	28.1	0.844
SAVP [42]	10	30	374	26.5	0.756
MCVD [50]	5	30	323	27.5	0.835
SLAMP [65]	10	30	228	29.4	0.865
SRVP [66]	10	30	222	29.7	0.870
RIVER [51]	10	30	180	30.4	0.86
CVP (Ours)	1	30	140.6	29.8	0.872
Struct-vRNN [67]	10	40	395.0	24.29	0.766
SVG-LP [46]	10	40	157.9	23.91	0.800
MCVD [50]	5	40	276.7	26.40	0.812
SAVP-VAE [42]	10	40	145.7	26.00	0.806
Grid-keypoints [68]	10	40	144.2	27.11	0.837
RIVER [51]	10	40	170.5	29.0	0.82
CVP (Ours)	1	40	120.1	29.2	0.841

Table 1.1: Video prediction results on KTH (64×64), predicting 30 and 40 frames using models trained to predict k frames at a time. All models condition on 10 past frames on 256 test videos.

methodology is supported by our model’s efficient handling of video sequences as continuous processes, which eliminates the need for external artificial constraints, such as temporal attention mechanisms.

The results, as shown in Table 2.1, clearly indicate that our method delivers state-of-the-art performance when compared to other baseline models. Additionally, the qualitative results for our CVP model on the KTH dataset can be observed in Fig. 2.3.

BAIR Robot Push dataset: The BAIR Robot Push dataset is characterized by highly stochastic video sequences. In our study, we adhered to a baseline setup [50] with three main experimental settings: 1) using only one context frame to predict the next 15 frames, 2) employing two context frames to predict 14 future frames, and 3) utilizing two context frames to forecast the next 28 frames. The outcomes of these approaches are summarized in Table 2.2.

As observed in Table 2.2, a trend emerges where increasing the number of frames predicted at a time concurrently results in a degradation of prediction quality.

This phenomenon is hypothesized to stem from an augmented disparity between the blocks of context frames and predicted future frames. Specifically, consider the scenario where two context frames are designated as $\mathbf{x}^{0:2}$, corresponding to $\mathbf{x}$ in the context of Eqn.2.1. Under the first experimental condition, where the model predicts a single frame at a time, the future frame prediction block is represented as $\mathbf{x}^{1:3}$, analogous to $\mathbf{y}$ in Eqn.2.1. Conversely, in the second condition, where two frames are predicted simultaneously, the future frame block extends to $\mathbf{x}^{2:4}$, again paralleling $\mathbf{y}$ in the equation. This setup implies that in the former setting, interpolation occurs between adjacent frames (i.e., the transition from $\mathbf{x}^0 \rightarrow \mathbf{x}^1$ and $\mathbf{x}^1 \rightarrow \mathbf{x}^2$), while in the latter, interpolation spans a two-frame interval (i.e., the transition from $\mathbf{x}^0 \rightarrow \mathbf{x}^2$ and from $\mathbf{x}^1 \rightarrow \mathbf{x}^3$). The expanded interval in the second scenario is posited as the causative factor for the observed reduction in predictive performance, particularly in configurations where $k = 2$ and $p = 2$.

The results, as shown in Table 2.2, clearly indicate that our method delivers state-of-the-art performance compared to other baseline models. Additionally, the qualitative results for our CVP model on the BAIR dataset can be observed in Fig. 2.4.

Human3.6M dataset: Similar to the KTH dataset, the Human3.6M dataset features actors performing distinct actions against a static background. However, the Human3.6M dataset distinguishes itself by offering a greater variety of distinct actions within its videos and providing three-channel video frames, in contrast to the single-channel frames of the KTH dataset. For evaluating the Human3.6M dataset, we employed a similar setup to that used for the KTH dataset, where 5 frames are

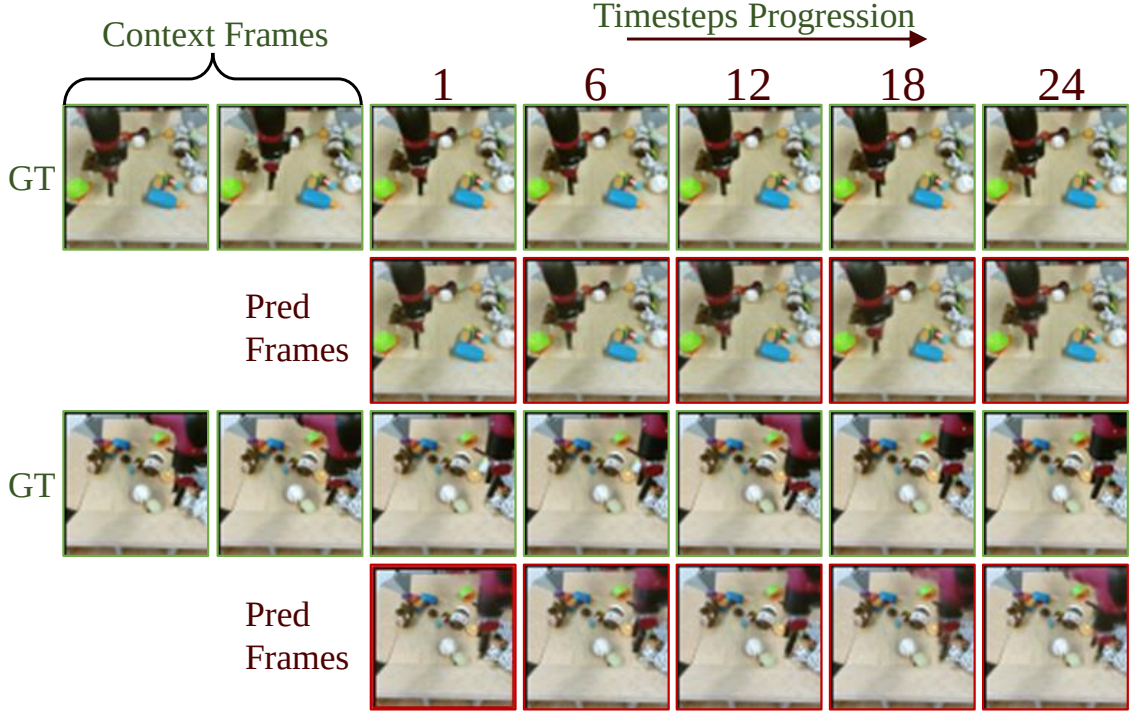


Figure 1.4: Figure represents qualitative results of our CVP model on the BAIR dataset. The number of context frames used in the above setting is two for both sequences. Every 6th predicted future frame is shown in the figure.

provided as context, and the model predicts the subsequent 30 frames based on these context frames. The results of this evaluation are summarized in Table 2.4.

An analysis of Table 2.4 reveals that our model, with its unique approach, requires a significantly lower number of frames for training, needing only a total of 6 frames per block to yield results that are considerably better than those of the baselines.

The results, as presented in Table 2.4, unequivocally demonstrate that our method outperforms other baseline models, establishing a new state-of-the-art on the Human3.6M dataset. Furthermore, the qualitative efficacy of our CVP model on the Human3.6M dataset is illustrated in Fig. 2.5, showcasing the model’s ability to effectively capture and predict the dataset’s varied actions.

BAIR (64×64)	p	k	#pred	FVD↓
LVT [69]	1	15	15	125.8
DVD-GAN-FP [43]	1	15	15	109.8
TrIVD-GAN-FP [44]	1	15	15	103.3
VideoGPT [70]	1	15	15	103.3
CCVS [71]	1	15	15	99.0
FitVid [72]	1	15	15	93.6
MCVD [50]	1	5	15	89.5
NÜWA [73]	1	15	15	86.9
RaMViD [53]	1	15	15	84.2
VDM [52]	1	15	15	66.9
RIVER [51]	1	15	15	73.5
CVP (Ours)	1	1	15	70.1
DVG [48]	2	14	14	120.0
SAVP [42]	2	14	14	116.4
MCVD [50]	2	5	14	87.9
CVP (Ours)	2	2	14	68.2
CVP (Ours)	2	1	14	65.1
SAVP [42]	2	10	28	143.4
Hier-vRNN [47]	2	10	28	143.4
MCVD [50]	2	5	28	118.4
CVP (Ours)	2	2	28	95.1
CVP (Ours)	2	1	28	85.1

Table 1.2: *BAIR* dataset evaluation. Video prediction results on BAIR (64×64) conditioning on p past frames and predicting $pred$ frames in the future, using models trained to predict k frames at a time. The common way to compute the FVD is to compare 100×256 generated sequences to 256 randomly sampled test videos. Best results are marked in *bold*.

UCF101 dataset: The UCF101 dataset presents a greater level of complexity compared to the KTH or Human3.6M datasets, owing to its substantially higher number of action categories, diverse backgrounds, and significant camera movements. Notably, we only use information from the context frames for our prediction task. No extra information, like class labels, was used for the prediction task. In evaluating the UCF101 dataset, we adopted an approach similar to that used for the Human3.6M dataset, where 5 context frames are provided, and the model is tasked with predicting

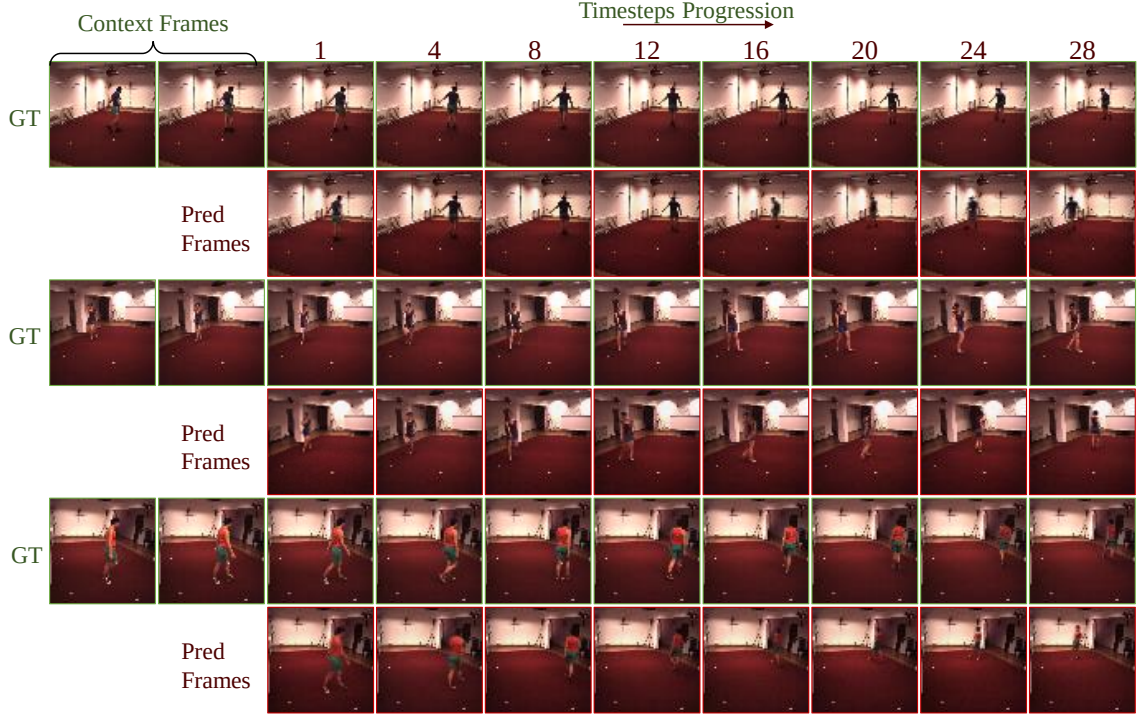


Figure 1.5: Figure represents qualitative results of our CVP model on the Human3.6M dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4th predicted future frame is shown in the figure.

the next 16 frames based on these. The outcomes of this evaluation are detailed in Table. 2.5.

An examination of Table. 2.5 reveals that our CVP model surpasses the performance of other baseline models, thereby setting a new benchmark for the UCF101 dataset. Additionally, the qualitative performance of our CVP model on the UCF101 dataset is depicted in Fig. 2.6. This illustration showcases the model’s proficiency in accurately capturing and predicting the diverse range of actions featured in the dataset.

Human3.6M	p	k	#pred	FVD↓
SVG-LP [46]	5	10	30	718
Struct-VRNN [67]	5	10	30	523.4
DVG [48]	5	10	30	479.5
SRVP [66]	5	10	30	416.5
Grid keypoint [68]	8	8	30	166.1
CVP (Ours)	5	1	30	144.5

Table 1.3: Quantitative comparisons on the Human3.6M dataset. The best results under each metric are marked in bold.

UCF101 [5 → 16]	p	k	#pred	FVD↓
SVG-LP [46]	5	10	16	1248
CCVS [71]	5	16	16	409
MCVD [50]	5	5	16	387
RaMViD [53]	5	4	16	356
CVP (Ours)	5	1	16	245.2

Table 1.4: Video prediction results on UCF (128×128), predicting 16 frames. All models are conditioned on 5 past frames.

1.6 Limitation

While our method demonstrates promising results in video prediction, it is important to acknowledge its limitations to guide future research and application development.

A primary limitation of our approach is its reliance on a limited context frame window for predicting the next frame. Specifically, when a context vector, denoted as $\mathbf{x}^{0:4}$, comprising 4 video frames is used, the prediction of the subsequent frame is entirely dependent on this four-frame window. This model architecture performs adequately in scenarios involving uniform video sequences. However, its efficacy diminishes in a setting that requires more context to predict the future frame. Addressing this limitation requires a more adaptive approach that can handle varying

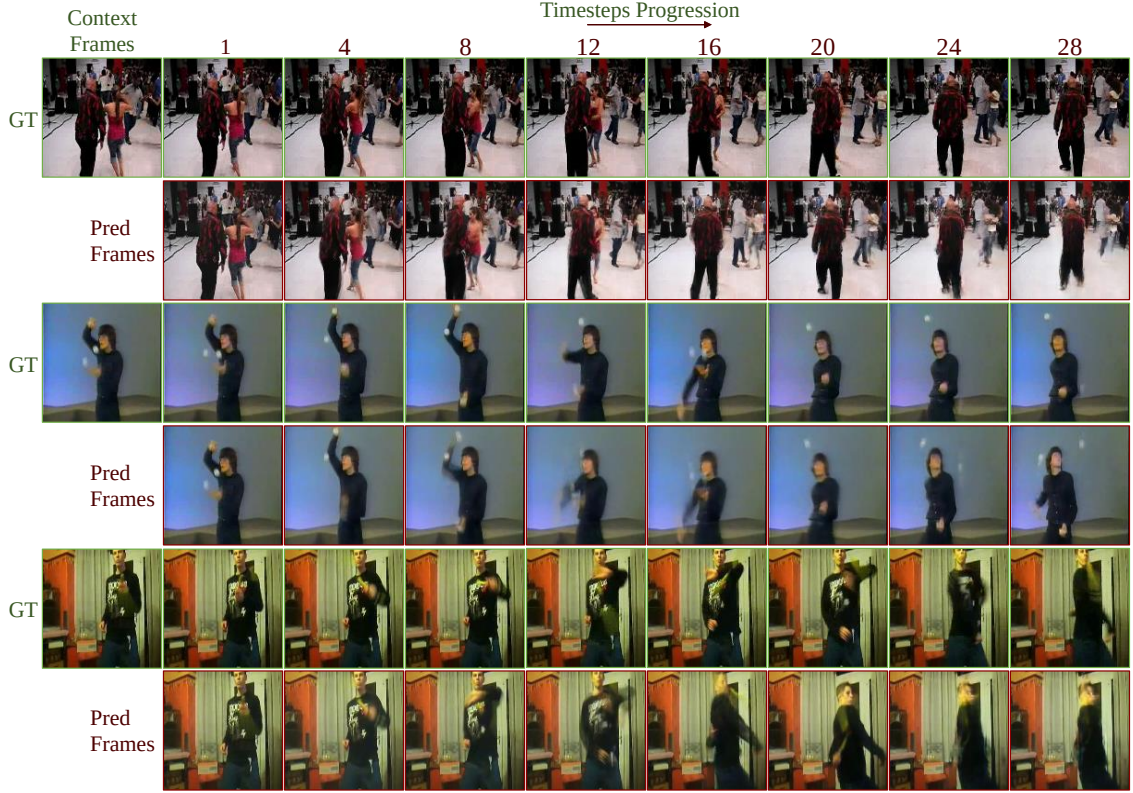


Figure 1.6: Figure represents qualitative results of our CVP model on the UCF dataset. The number of context frames used in the above setting is 5 for all three sequences. Every 4th predicted future frame is shown in the figure.

contextual information, a challenge we have earmarked for future research.

Another constraint lies in the computational efficiency of our model. Currently, it necessitates multiple steps to sample a single frame, which could become a significant bottleneck, especially when a larger number of frame predictions are required. Although our method is more efficient in terms of the number of steps needed for frame sampling compared to diffusion-based counterparts, further optimization is necessary to reduce the computational overhead associated with this process.

Additionally, our experimental setup was constrained by the computational resources available to us. The model was developed and tested using just two A6000 GPUs. This limitation raises questions about the potential improvements

that could be achieved with a more powerful computational setup. A larger model with an increased number of parameters, trained on more advanced hardware, could potentially unveil further advancements in video prediction capabilities. We recognize this as an important area for investigation and encourage labs with more substantial resources to explore this avenue.

In summary, while our model represents a significant step forward in video prediction, these limitations highlight crucial areas for future research and development, paving the way for more robust and versatile video prediction models.

1.7 Conclusion

In this work, we have presented a novel model class designed specifically for video representation, marking a significant advancement in the field of video prediction tasks. Our comprehensive experimental evaluations across various datasets, including KTH, BAIR, Human3.6M, and UCF101, have not only validated the effectiveness of our model but also established new benchmarks in state-of-the-art performance for video prediction tasks.

A notable aspect of our approach is its efficiency in terms of the required number of context and future frames for training. Moreover, our model’s continuous video process capability uniquely operates without the need for additional constraints such as temporal attention, which are typically employed to ensure temporal consistency. This aspect of our model underscores its inherent ability to maintain temporal coherence, further simplifying the video prediction process while enhancing its

effectiveness.

In conclusion, the innovations introduced in our model offer promising directions for future research in video representation and prediction. The achievements demonstrated in this paper not only contribute to the advancement of video prediction methodologies but also open avenues for exploring more efficient and effective ways of video representation in various real-world applications.

1.8 Broader Impact

We used this method for video prediction; however, such modeling can make a major impact on many computational photography tasks. Here, one end of the CVP can be a corrupted image and the other end be a clean ground truth image. Additionally, a larger model with an increased number of parameters, trained on more advanced hardware, could potentially have advanced video prediction capabilities. This can lead to a significant increase in the creation of high-quality artificially generated content, further compounding the problems of fake content. However, a positive contribution of this approach can help with its application in autonomous driving.

1.9 Appendix

1.9.1 Extended derivations of Eq. (1.21)

Below is a derivation of Eq. (1.21), the reduced variance variational bound for our CVP models.

$$\begin{aligned}
L &= \mathbb{E}_q \left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{0:T-1}|\mathbf{x}_T)} \right] \\
&= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t)} \right] \\
&= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) \right. \\
&\quad \left. - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0, \mathbf{x}_T)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_T)} \right] \\
&= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right. \\
&\quad \left. - \log \frac{q(\mathbf{x}_T|\mathbf{x}_0, \mathbf{x}_T)}{q(\mathbf{x}_0|\mathbf{x}_0, \mathbf{x}_T)} \right] \\
&= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right] \\
&= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) \right. \\
&\quad \left. - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \cdot \frac{p(\mathbf{x}_0|\mathbf{x}_{t-1})}{p(\mathbf{x}_0|\mathbf{x}_t)} \right] \\
&= \mathbb{E}_q \left[-\log p(\mathbf{x}_0) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right. \\
&\quad \left. - \log \frac{p(\mathbf{x}_0|\mathbf{x}_0)}{p(\mathbf{x}_0|\mathbf{x}_T)} \right] \\
&= \mathbb{E}_q \left[-\log \frac{p(\mathbf{x}_0)}{p(\mathbf{x}_0|\mathbf{x}_T)} - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0, \mathbf{x}_T)} \right]
\end{aligned}$$

Both $\mathbf{x}_0$ and $\mathbf{x}_T$ are observed variable hence, we ignore the first term in the RHS. We focus on the second term for training the parameters for our CVP models. Therefore,

the resulting loss function becomes,

$$L(\theta) =: \sum_{t \geq 1} D_{\text{KL}}(q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}, \mathbf{y}) \parallel p_{\theta}(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0)).$$

1.9.2 Extended derivation for Eq. (2.2)

Using Eq. (2.2) we can write the term $\mathbf{x}_{t+\Delta t}$ as follows,

$$\begin{aligned} \mathbf{x}_{t+\Delta t} &= (1 - (t + \Delta t))\mathbf{x} + (t + \Delta t)\mathbf{y} \\ &\quad - \frac{(t + \Delta t) \log(t + \Delta t)}{\sqrt{2}} \mathbf{z}_{t+\Delta t} \end{aligned}$$

Considering the term $(t + \Delta t) \log(t + \Delta t)$ we simplify further,

$$(t + \Delta t) \log(t + \Delta t) = t \left(1 + \frac{\Delta t}{t}\right) \log t \left(1 + \frac{\Delta t}{t}\right). \quad (1.27)$$

if Δt is infinitesimally small we can write $\left(1 + \frac{\Delta t}{t}\right) \rightarrow 1$. Using this property we can

rewrite $\mathbf{x}_{t+\Delta t}$ as,

$$\mathbf{x}_{t+\Delta t} = (1 - (t + \Delta t))\mathbf{x} + (t + \Delta t)\mathbf{y} - \frac{t \log(t)}{\sqrt{2}} \mathbf{z}_{t+\Delta t} \quad (1.28)$$

$$(1.29)$$

Now, Subtracting $\mathbf{x}_{t+\Delta t}$ (Eq. (1.29)) and $\mathbf{x}_t$ (Eq. (2.1)) we get,

$$\mathbf{x}_{t+\Delta t} - \mathbf{x}_t = (y - x)\Delta t - \frac{t \log(t)}{\sqrt{2}}(\mathbf{z}_{t+\Delta t} - \mathbf{z}_t) \quad (1.30)$$

Focusing on the term $(\mathbf{z}_{t+\Delta t} - \mathbf{z}_t)$. Here, $\mathbf{z}_t, \mathbf{z}_{t+\Delta t} \sim \mathcal{N}(0, I)$. Hence, we can write,

$$(\mathbf{z}_{t+\Delta t} - \mathbf{z}_t) = \sqrt{2}\mathbf{z} \quad \text{where, } \mathbf{z} \sim \mathcal{N}(0, I) \quad (1.31)$$

Substituting this result back to Eq. (1.30) we get the following,

$$\mathbf{x}_{t+\Delta t} - \mathbf{x}_t = (y - x)\Delta t - t \log(t)\mathbf{z}. \quad (1.32)$$

Rearranging the terms we get the Eq. (2.2).

1.9.3 Training Details

For the optimization of our model, we harnessed the compute of two Nvidia A6000 GPUs, each equipped with 48GB of memory, to train our CVP model effectively. We adopted a batch size of 64 and conducted training for a total of 500,000 iterations. To optimize the model parameters, we employed the AdamW optimizer. Additionally, we incorporated a cosine decay schedule for learning rate adjustment, with warm-up steps set at 10,000 iterations. The maximum learning rate (Max LR) utilized during training was 5e-5.

Module	Type	Num Inputs	Num Outputs
Encoder	Conv2D	$n \times c$	128
	DownBlock2D	128	128
	DownBlock2D	128	128
	DownBlock2D	128	256
	DownBlock2D	256	256
	AttnDownBlock2D	256	512
	ResnetDownsampleBlock2D	512	512
Decoder	ResnetUpsampleBlock2D	512	512
	AttnUpBlock2D	512	512
	UpBlock2D	512	256
	UpBlock2D	256	256
	UpBlock2D	256	128
	UpBlock2D	128	128
	Conv2d	128	$n \times c$

Table 1.5: **U-NET**: We utilize Hugging face diffusers library for our U-Net implementation. We utilize ‘positional’ type for timestep embeddings. We utilize 4 layers per block. The target resolution for KTH, BAIR and Human3.6M is kept at 64×64 and 128×128 for UCF101 dataset. Additionally, we keep the number of timesteps T as 100 given our compute resources. c denotes the number of channels present in the frame. n is the number of initial context frames based on which next frame is predicted, i.e., $\mathbf{x}^{0:n} \rightarrow \mathbf{x}^{1:n+1}$.

1.9.4 Ablation Studies

In this section, we present a series of ablation studies conducted to ascertain the impact of various components in our proposed methodology. These studies focus on three primary aspects: the modification of the noise schedule denoted as $g(t)$, the variation in the number of sampling steps, and the exploration of different strategies for sampling the timestep t . Our experimental framework utilizes the KTH dataset for these evaluations.

The outcomes of these experiments are systematically tabulated in Table. 1.7, offering a comprehensive view of the results. The key insights derived from these

Method	Sampling(Steps/Frame)	Time Taken(hrs)
MCVD	100	2
RaMVID	500	7.2
Ours	25	0.45

Table 1.6: Comparison with baselines on sampling steps and sampling time required for BAIR robot push dataset.

ablation studies are threefold. Firstly, our analysis underscores the criticality of sampling the timestep t from a uniform square root distribution, specifically $t \sim \sqrt{\mathcal{U}[0, 1]}$. This approach appears to significantly influence the model’s performance.

Secondly, regarding the noise schedule $g(t)$, we find that the optimal formulation for the task of video prediction is given by $g(t) = \frac{-t \log(t)}{\sqrt{2}}$. This particular noise schedule is characterized by a zero initial and final noise level, with a peak near $t = 0$. Such a configuration is advantageous for our application.

Thirdly, our results, as detailed in Table 1.7, indicate that an increase in the number of sampling steps beyond 25 does not substantially improve the outcome. Our method outperforms MCVD by producing higher-quality frames in just 25 sampling steps, a 75% reduction compared to its 100 steps. This efficiency is attributed to our CVP method, which retains information from preceding frames, eliminating the need for regeneration from a Gaussian noise vector. Refer to the Table 1.6 for more details.

In summary, these ablation studies provide valuable insights into the dynamics of our model under varying conditions, highlighting the importance of specific parameter settings and offering guidance for future research directions.

KTH	Noise Schedule($\sqrt{2}g(t)$)	Sampling steps	t Distrbution	FVD↓
CVP Model Ablations	-	25	$\mathcal{U}[0, 1]$	348.2
	$\sin(\pi t)$	25	$\mathcal{U}[0, 1]$	278.2
	$\sin(\pi t)$	25	$\sqrt{\mathcal{U}[0, 1]}$	237.7
	$t \sin(\pi t)$	25	$\mathcal{U}[0, 1]$	240.7
	$t \sin(\pi t)$	25	$\sqrt{\mathcal{U}[0, 1]}$	208.4
	$\sqrt{t(1-t)}$	25	$\mathcal{U}[0, 1]$	209.6
	$\sqrt{t(1-t)}$	25	$\sqrt{\mathcal{U}[0, 1]}$	187.8
	$-t \log(t)$	25	$\mathcal{U}[0, 1]$	190.4
	$-t \log(t)*$	25*	$\sqrt{\mathcal{U}[0, 1]}*$	140.6*
	$-t \log(t)$	5	$\sqrt{\mathcal{U}[0, 1]}$	165.7
	$-t \log(t)$	10	$\sqrt{\mathcal{U}[0, 1]}$	144.3
	$-t \log(t)$	50	$\sqrt{\mathcal{U}[0, 1]}$	139.4

Table 1.7: **Ablation study:** Video prediction results on KTH (64×64), predicting 30 frames. All models condition on 4 past frames on 256 test videos. The method with settings marked with * is reported in the main paper.

Chapter 2: Efficient Continuous Video Flow Model for Video Prediction

2.1 Introduction

Videos serve as digital representations of the continuous real world. However, due to inherent camera limitations, particularly fixed framerate constraints, these continuous signals are discretized in the temporal domain when captured. This temporal discretization, while often overlooked in generative video modeling, presents significant challenges. Current approaches, such as diffusion-based models, generally focus on generating the next frame in a sequence based on a given context frame, neglecting intermediate moments between frames (e.g., predicting frames at $T+0.5$ or $T+0.25$ given a context frame at T). In this work, we aim to address the limitations imposed by such discretization during the generative modeling of videos.

Diffusion models have emerged as a state-of-the-art technique for video generation, but they face computational challenges, particularly in inference time. High fidelity inferences from diffusion models require hundreds or even thousands of sampling steps, which may be feasible for single-image generation but becomes prohibitive for video generation, where thousands of frames must be generated for even a short video. The naive adaptation of diffusion models for video tasks results in significant computational bottlenecks during inference, especially in production settings. To

address this issue, we propose a method that reduces the need for extensive sampling by starting the process from previous context frames rather than from an analytical distribution. This continuous modeling approach not only reduces the number of sampling steps but also enhances fidelity and reduces the model’s parameter count.

Our proposed method introduces a continuous process that treats video frames as multi-dimensional endpoints. Starting with two consecutive frames, we pass them through an encoder network to obtain their latent embeddings. We then interpolate between these endpoints using a noise schedule that applies zero noise at the boundaries, thus ensuring the existence of $p(\mathbf{x}_t)$ at all points. This continuous framework, grounded in Gaussian approximations between frames, builds on existing diffusion models for images while extending their applicability to videos.

In summary, our contributions are as follows:

- We introduce a novel approach for representing videos as multi-dimensional continuous processes in latent space.
- We empirically demonstrate that our model requires fewer sampling steps, reducing inference-time computational overhead without compromising result fidelity.
- By reducing the memory footprint of frames in latent space, our model can predict future frames over a longer context sequence.
- Our approach requires fewer parameters compared to state-of-the-art video prediction models.

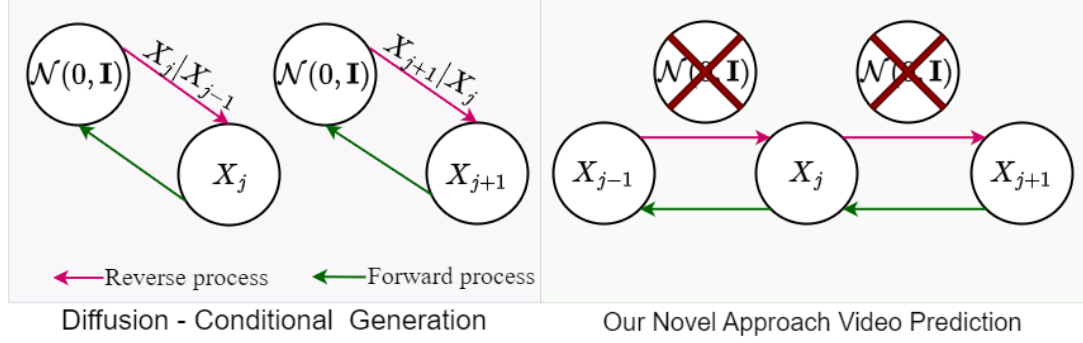


Figure 2.1: Fig. (a) represents a naive adaptation of the diffusion model for the video prediction task. Here, the sampling process always starts from a Gaussian distribution, and sampling steps are taken in the direction of conditional distribution given by $X_{j+1}|X_j$. Here, X_{j+1} denotes frame at time $j + 1$. In contrast, Fig. (b) introduces our Continuous Video Flow (CVF) approach, which reimagines the problem by treating video not as a discrete sequence of frames but as a continuously evolving process. Instead of starting from a static Gaussian distribution for each sampling step, CVF models the underlying dynamics of the entire video, learning to predict changes smoothly over time. This continuous framework allows the model to better capture temporal coherence and evolution, leading to more accurate and fluid video predictions.

- We demonstrate state-of-the-art performance across several video prediction benchmarks, including KTH action recognition, BAIR robot push, Human3.6M, and UCF101 datasets.

2.2 Related Works

Predicting future frames from past observations is a fundamental task in video understanding, critical for domains requiring accurate modeling of multi-modal future outcomes, such as autonomous driving. Early approaches, such as those by Yuen et al. [31] and Walker et al. [32], tackled this problem by matching observed frames to predict future states, primarily relying on symbolic trajectories or directly retrieved future frames from datasets. While these methods laid important groundwork, their

predictions were constrained by the deterministic nature of their models, limiting their ability to capture the inherent uncertainty of future frames.

The introduction of deep learning marked a significant advancement in this area. Srivastava et al. [33] pioneered the use of multi-layer LSTM networks, which focused on deterministic representation learning for video sequences. Building upon this, subsequent work explored more complex and stochastic methods. For instance, studies such as [34, 35, 36, 37, 38, 39, 40] shifted towards incorporating stochastic processes to better model the uncertain nature of future frame predictions. This transition represents a growing recognition in the field: capturing uncertainty is essential for more accurate and robust future frame prediction.

Research in video prediction has advanced along two primary directions: implicit and explicit probabilistic models. Implicit models, particularly those rooted in Generative Adversarial Networks (GANs) [41], have been widely explored but frequently encounter difficulties with training stability and mode collapse—where the model disproportionately focuses on a limited subset of data modes [42, 43, 44]. These limitations have motivated increased interest in explicit probabilistic methods, which offer a more controlled approach to video prediction.

Explicit methods cover a spectrum of techniques, such as Variational Autoencoders (VAEs) [45], Gaussian processes, and diffusion models. VAE-based approaches to video prediction [42, 46, 47] often produce results that average multiple potential outcomes, resulting in lower-quality predictions. Gaussian process-based models [48, 49], while effective on small datasets, struggle with scalability due to the computational expense of matrix inversions required for training likelihood estima-

tions. Attempts to circumvent this limitation usually result in reduced predictive accuracy.

Diffusion models [50, 51, 52, 53] have emerged as a powerful alternative, providing high-quality samples and mitigating issues like mode collapse. These models leverage multi-step denoising, ensuring more consistent predictions. However, maintaining temporal coherence across frames requires additional constraints, such as temporal attention blocks, which can significantly increase computational demands.

An emerging method, the continuous video process [74], conceptualizes videos as continuously evolving data streams. This approach efficiently utilizes redundant information in video frames to reduce the time needed for inference. Despite its promise, this method relies on pixel-space blending between consecutive frames, leading to suboptimal redundancy exploitation. Our approach [75] instead performs blending in the latent space, facilitating improved semantic interpolation between frames, as evidenced by previous work [74].

Finally, recent advancements such as InDI [59], rectified flow methods [76, 77], and Cold Diffusion [60] propose alternative diffusion-based strategies, though their focus has predominantly been on image generation and computational photography. In contrast, our approach extends these ideas to video prediction, achieving a balance between computational efficiency and temporal consistency.

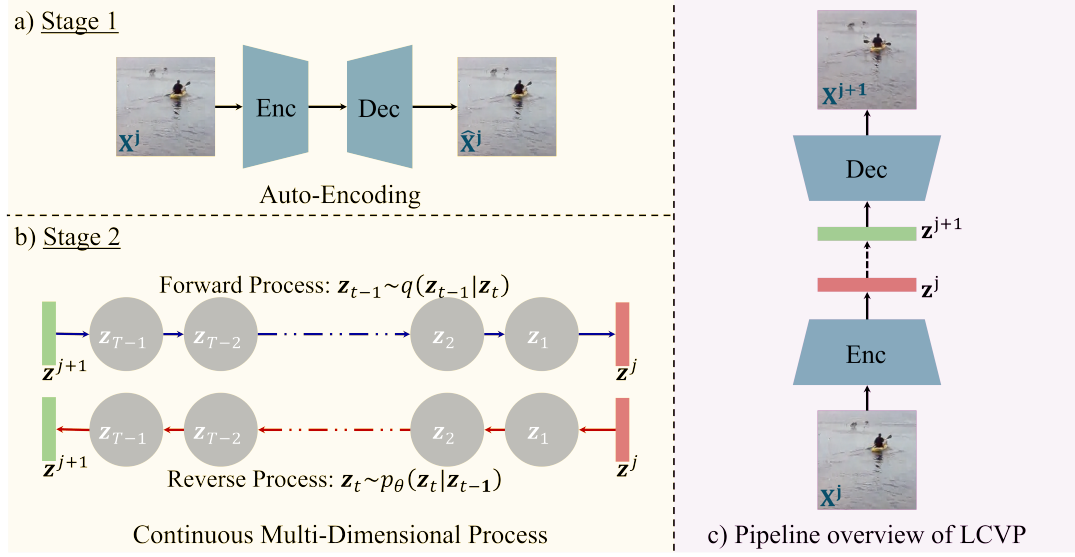


Figure 2.2: **Overview of the Continuous Video Flow (CVF) framework.** (a) Stage 1 depicts the auto-encoding process where input video frames X_j are passed through an encoder (Enc) and decoder (Dec) to reconstruct $\hat{X}_j$. (b) Stage 2 illustrates the forward and reverse processes in the latent space. In the forward process, latent embeddings $z_T, \dots, z_1$ are generated through a fixed process given by Eqn 2.2. The reverse process involves sampling z_{j+1} from z_j through the learned process p_θ for continuous video frame prediction. (c) The full pipeline of CVF, showing how frame X_j is passed through the encoder to obtain latent embedding z_j and $\hat{z}_{j+1}$, which is then used for decoding to obtain $\hat{X}_{j+1}$ frame.

2.3 Method

Videos contain significant redundancy at the pixel level, and directly interpolating between frames in pixel space often results in blurry intermediate frames. To address both the redundancy and interpolation issues, we encode the frames into a latent space. This encoding offers two key benefits - First, **Compression:** Reduces irrelevant pixel information. Second, **Semantic Interpolation:** Ensures that interpolation in the latent space corresponds to more meaningful, semantic transitions between frames [78]. In addition, encoding frames into the latent space allows for a larger context window when performing video prediction tasks.

2.3.1 Video Prediction Framework

Let $\mathcal{V} = \{\mathbf{x}^t\}_{t=1}^N$ be a video sequence where each frame $\mathbf{x}^j \in \mathbb{R}^{c \times h \times w}$ is a tensor representing the frame at timestep j . Our video prediction framework consists of two main stages:

Stage 1: Encoding in Latent Space. Each frame in the sequence is encoded into the latent space using a pre-trained autoencoder [79]. The encoder processes the video sequence $\mathcal{V}$ and produces a corresponding sequence of latent representations, frame by frame.

Stage 2: Continuous Process in Latent Space. Once the frames are encoded in latent space, we model the sequence as a continuous process. Specifically, the interpolation

between consecutive latent frames $\mathbf{z}^j$ and $\mathbf{z}^{j+1}$ is given by:

$$\mathbf{z}_t = (1 - t)\mathbf{z}_x + t\mathbf{z}_y - \frac{t \log(t)}{\sqrt{2}}\boldsymbol{\epsilon} \quad (2.1)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$ denotes white noise. This process ensures smooth transitions between frames in latent space. At $t = 0$, we retrieve the latent $\mathbf{z}^j$, and at $t = 1$, we retrieve the latent $\mathbf{z}^{j+1}$.

2.3.2 Forward and Reverse Process

We utilize a forward process to transition from latent $\mathbf{z}_x$ to latent $\mathbf{z}_y$ over time t . The forward process starts at $t = T$ with latent $\mathbf{z}_y$ and moves to $t = 0$ for latent $\mathbf{z}_x$, as described by:

$$\mathbf{z}_{t+\Delta t} = \mathbf{z}_t + (\mathbf{z}_y - \mathbf{z}_x)\Delta t - t \log(t)\boldsymbol{\epsilon} \quad (2.2)$$

From this, the forward process posterior is:

$$q(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{z}_x, \mathbf{z}_y) = \mathcal{N}(\mathbf{z}_{t+1}; \tilde{\boldsymbol{\mu}}(\mathbf{z}_t, \mathbf{z}_x, \mathbf{z}_y), g^2(t)I) \quad (2.3)$$

where $g(t) = -t \log(t)$ and $\tilde{\boldsymbol{\mu}}(\mathbf{z}_t, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{z}_t + (\mathbf{z}_y - \mathbf{z}_x)$.

2.3.3 Likelihood and Variational Bound

We model the continuous process in latent space through the following likelihood function:

$$p_\theta(\mathbf{z}_T) := \int p_\theta(\mathbf{z}_{0:T}) d\mathbf{z}_{0:T-1} \quad (2.4)$$

To train the model, we minimize the negative log-likelihood. The training objective involves minimizing the following variational bound:

$$\mathbb{E}[-\log p_\theta(\mathbf{z}_T)] \leq \mathbb{E}_q \left[-\log \frac{p_\theta(\mathbf{z}_{0:T})}{q(\mathbf{z}_{0:T-1}|\mathbf{z}_T)} \right] \quad (2.5)$$

which can be simplified following the paper [74]:

$$L(\theta) = \sum_{t \geq 1} \text{KL}(q(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{z}_x, \mathbf{z}_y) || p_\theta(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{z}_x)) \quad (2.6)$$

Here, the KL divergence compares the forward process posterior with the reverse process model.

2.3.4 Reverse Process

We assume a Markov chain structure for the reverse process, meaning that the current state depends only on the previous timestep:

$$p_{\theta}(\mathbf{z}_{0:T}) = p(\mathbf{z}_0) \prod_{t=1}^T p_{\theta}(\mathbf{z}_t | \mathbf{z}_{t-1}) \quad (2.7)$$

The reverse process is modeled as a Gaussian distribution:

$$p_{\theta}(\mathbf{z}_{t-1} | \mathbf{z}_t) = \mathcal{N}(\mathbf{z}_t; \boldsymbol{\mu}_{\theta}(\mathbf{z}_{t-1}, t-1), g^2(t)\mathbf{I}) \quad (2.8)$$

2.3.5 Training Objective

The training objective simplifies to:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t, \mathbf{z}_t} \left[\frac{1}{2g^2(t)} \|\mathbf{z}_y - \mathbf{z}_{\theta}(\mathbf{z}_t, t)\|^2 \right] \quad (2.9)$$

where $g(t) = -t \log(t)$ controls the noise added at each timestep. We use the interpolation function in Eqn. 2.1 to model the intermediate frames during training.

Final Loss Function. The final training objective to optimize the video prediction model is given by:

$$_{\theta} \mathbb{E}_{t, \mathbf{z}_x, \mathbf{z}_y} \left[\frac{1}{2g^2(t)} \left\| \mathbf{z}_y - \mathbf{z}_{\theta}((1-t)\mathbf{z}_x + t\mathbf{z}_y + \frac{g(t)}{\sqrt{2}}\boldsymbol{\epsilon}, t) \right\|^2 \right] \quad (2.10)$$

Algorithm 3 Sampling Algorithm

```

1:  $\mathbf{z}_x \sim q_{\text{data}}(\mathbf{z}_x)$ 
2:  $\mathbf{z}_0 = \text{Enc}(\mathbf{x})$ 
3:  $d = \frac{1}{N}$ , Here  $N$  denotes number of steps.
4: for  $t = 1, \dots, N$  do
5:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{Id})$  if  $t > 1$ , else  $\epsilon = \mathbf{0}$ 
6:    $\mathbf{z}_{t+1} = \mathbf{z}_t + (\hat{y}(\mathbf{z}_t, t) - \mathbf{z}_x)d - t \log(t)\epsilon$ 
7: end for
8: return  $\text{Dec}(\mathbf{z}_T)$ 

```

Algorithm 4 Training of CVF model

```

1: repeat
2:    $\mathbf{x}, \mathbf{y} \sim q_{\text{data}}(\mathbf{x}, \mathbf{y})$ 
3:    $\mathbf{z}_x, \mathbf{z}_y \sim \text{Enc}(\mathbf{x}), \text{Enc}(\mathbf{y})$ 
4:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
5:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
6:   Take gradient descent step on
        $\nabla_{\theta} \frac{1}{2g^2(t)} \|\mathbf{z}_y - \mathbf{z}_{\theta}((1-t)\mathbf{z}_x + t\mathbf{z}_y - (t \log(t)/\sqrt{2})\epsilon, t)\|^2$ 
7: until converged

```

The entire training and sampling pipeline is described in Algorithm 4 and Algorithm 3, and visualized in Figure 2.2.

2.4 Experiments

The video prediction task is defined as predicting future frames from a given sequence of context frames. In this section, we empirically validate the performance of our proposed method, demonstrating its superior results in modeling video prediction tasks across diverse datasets.

2.4.1 Datasets

We employ four standard benchmarks to evaluate the efficacy of our approach: KTH Action Recognition, BAIR Robot Pushing, Human3.6M, and UCF101. Each of these datasets presents unique challenges for video prediction, allowing for a comprehensive assessment of our method’s robustness. Training protocols and architecture specifics are provided in the appendix.

KTH Action Recognition Dataset: This dataset [61] contains video sequences

of 25 individuals performing six actions—walking, jogging, running, boxing, hand-waving, and hand-clapping. The videos feature a uniform background with a single person performing the action in the foreground, offering relatively regular motion patterns. Each video frame is downsampled to a spatial resolution of 64×64 and is represented as a single-channel image.

BAIR Robot Pushing Dataset: The BAIR dataset [62] captures a Sawyer robotic arm pushing various objects on a table. It includes different robotic actions, providing a dynamic and controlled environment. The resolution of the video frames is also 64×64 .

Human3.6M Dataset: Human3.6M [63] features 10 subjects performing 15 different actions. Unlike other works, we exclude pose information from the prediction task, focusing solely on video frame data. The dataset offers regular foreground motion against a uniform background, with video frames in RGB format and downsampled to 64×64 .

UCF101 Dataset: UCF101 [2] consists of 13,320 videos spanning 101 action classes. This dataset introduces a diverse range of backgrounds and actions. Each frame is resized from the original resolution of 320×240 to 128×128 using bicubic downsampling, retaining all three RGB channels.

2.4.2 Metrics

For performance evaluation, we use the Fréchet Video Distance (FVD) [64] metric, which assesses both reconstruction quality and diversity in generated samples.

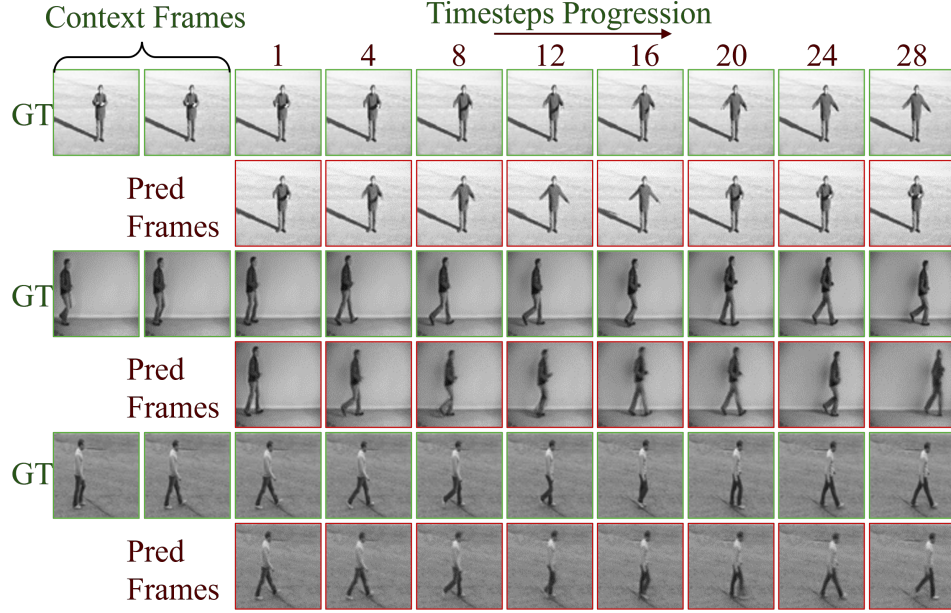


Figure 2.3: Figure represents qualitative results of our CVF model on the KTH dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4th predicted future frame is shown in the figure.

The FVD computes the Fréchet distance between I3D embeddings of generated and real video samples. The I3D network is pretrained on the Kinetics-400 dataset to provide reliable embeddings for this metric.

2.5 Setup and Results

In this section, we detail the experimental setup, comparing our method to existing baselines. We then present the performance of our approach across all datasets, analyzing both quantitative metrics and qualitative results.

KTH Action Recognition Dataset: For this dataset, we followed the baseline setup from [74], which uses the first 10 frames as context to predict the subsequent 30 or 40 future frames. A key distinction in our experiment is that, instead of utilizing

KTH [10 $\rightarrow$ #pred; trained on k]	k	#pred	FVD $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$
SVG-LP [46]	10	30	377	28.1	0.844
SAVP [42]	10	30	374	26.5	0.756
MCVD [50]	5	30	323	27.5	0.835
SLAMP [65]	10	30	228	29.4	0.865
SRVP [66]	10	30	222	29.7	0.870
RIVER [51]	10	30	180	30.4	0.86
CVP [74]	1	30	140.6	29.8	0.872
CVF (Ours)	1	30	108.6	30.6	0.891
Struct-vRNN [67]	10	40	395.0	24.29	0.766
SVG-LP [46]	10	40	157.9	23.91	0.800
MCVD [50]	5	40	276.7	26.40	0.812
SAVP-VAE [42]	10	40	145.7	26.00	0.806
Grid-keypoints [68]	10	40	144.2	27.11	0.837
RIVER [51]	10	40	170.5	29.0	0.82
CVP [74]	1	40	120.1	29.2	0.841
CVF (Ours)	1	40	100.8	29.7	0.852

Table 2.1: Video prediction results on KTH (64×64), predicting 30 and 40 frames using models trained to predict k frames at a time. All models condition on 10 past frames on 256 test videos.

all 10 context frames, we only use the last 4 frames as context in our CVF model, deliberately discarding the first 6 frames. This choice aligns with prior methodologies and allows a fair comparison. The results of our evaluation are summarized in Table 2.1.

From Table 2.1, it is evident that our model achieves superior performance while requiring fewer training frames. Unlike other approaches that rely on a larger number of frames for both context and future predictions (e.g., 10 context frames plus k future frames), our model operates effectively with just 4 context frames and 1 future frame. Specifically, we predict the immediate next frame using 4 context frames and then autoregressively generate 30 or 40 future frames, depending on the evaluation setting. This efficiency stems from our model’s continuous sequence processing capabilities, which avoid the need for explicit temporal attention mechanisms or

artificial constraints.

The results, as presented in Table 2.1, confirm that our method delivers state-of-the-art performance relative to baseline models. Qualitative results of our CVF model on the KTH dataset are shown in Fig. 2.3.

BAIR (64×64)	p	k	#pred	FVD↓
LVT [69]	1	15	15	125.8
DVD-GAN-FP [43]	1	15	15	109.8
TrIVD-GAN-FP [44]	1	15	15	103.3
VideoGPT [70]	1	15	15	103.3
CCVS [71]	1	15	15	99.0
FitVid [72]	1	15	15	93.6
MCVD [50]	1	5	15	89.5
NÜWA [73]	1	15	15	86.9
RaMViD [53]	1	15	15	84.2
VDM [52]	1	15	15	66.9
RIVER [51]	1	15	15	73.5
CVP [74]	1	1	15	70.1
CVF (Ours)	1	1	15	65.8
DVG [48]	2	14	14	120.0
SAVP [42]	2	14	14	116.4
MCVD [50]	2	5	14	87.9
CVP [74]	2	1	14	65.1
CVF(Ours)	2	1	14	61.2
SAVP [42]	2	10	28	143.4
Hier-vRNN [47]	2	10	28	143.4
MCVD [50]	2	5	28	118.4
CVP [74]	2	1	28	85.1
CVF (Ours)	2	1	28	78.5

Table 2.2: *BAIR* dataset evaluation. Video prediction results on BAIR (64×64) conditioning on p past frames and predicting $pred$ frames in the future, using models trained to predict k frames at a time. The common way to compute the FVD is to compare 100×256 generated sequences to 256 randomly sampled test videos. Best results are marked in *bold*.

BAIR Robot Push Dataset: The BAIR Robot Push dataset is known for its highly stochastic video sequences. In line with previous studies [74], we experimented with three different setups: 1) using one context frame to predict the next 15 frames,

2) using two context frames to predict 14 future frames, and 3) using two context frames to predict 28 future frames. The results for these settings are detailed in Table 2.2.

As shown in Table 2.2, there is a clear trend where increasing the number of predicted frames leads to a gradual decline in prediction quality. This performance drop is likely due to an increasing mismatch between the context frames and the predicted future frames. For instance, when using two context frames, denoted as $\mathbf{x}^{0:2}$, to predict a single future frame, the prediction block is represented as $\mathbf{z}^{1:3}$, aligning with the setup in Eqn.2.1. In the second scenario, where two frames are predicted, the future block extends to $\mathbf{x}^{2:4}$. This setup means that in the first condition, interpolation occurs between adjacent frames (i.e., from $\mathbf{z}^0 \rightarrow \mathbf{z}^1$ and $\mathbf{z}^1 \rightarrow \mathbf{z}^2$), while in the second condition, interpolation spans a larger gap (e.g., $\mathbf{z}^0 \rightarrow \mathbf{z}^2$ and $\mathbf{z}^1 \rightarrow \mathbf{z}^3$). This extended gap likely contributes to the observed decrease in predictive performance, particularly when $k = 2$ and $p = 2$.

As indicated in Table 2.2, our method consistently outperforms baseline models. Qualitative results of our CVF model on the BAIR dataset can be seen in Fig. 2.4.

Table 2.3 highlights the superior efficiency of the proposed CVF model compared to diffusion based baselines across key metrics. CVF has the fewest parameters and requires only 5 sampling steps per frame. This makes CVF highly efficient and practical for video prediction tasks, where speed and resource efficiency are critical.

Human3.6M Dataset: Like the KTH dataset, the Human3.6M dataset features actors performing distinct actions against a static background. However, Human3.6M provides a broader range of actions and includes three-channel RGB video frames,

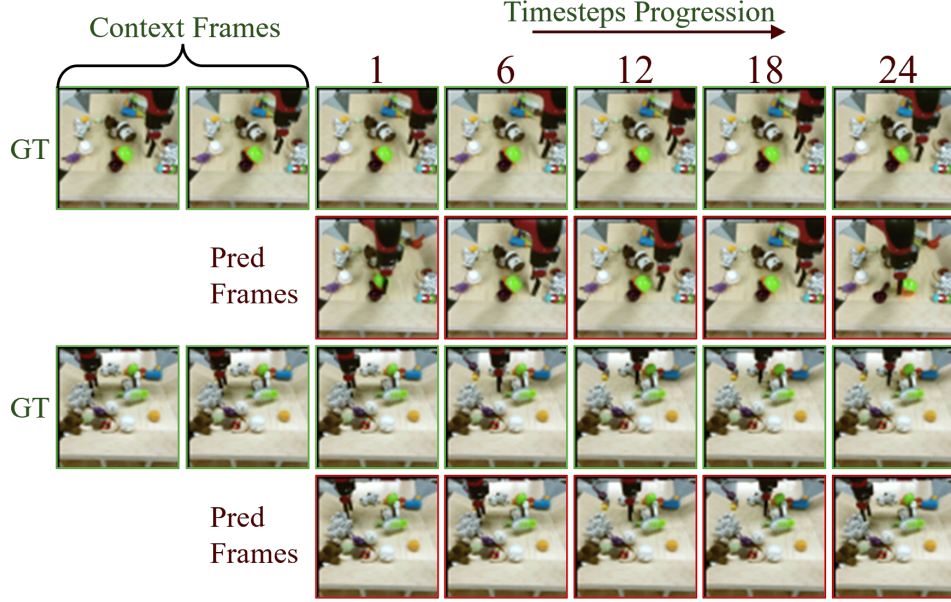


Figure 2.4: Figure represents qualitative results of our CVF model on the BAIR dataset. The number of context frames used in the above setting is two for both sequences. Every 6th predicted future frame is shown in the figure.

BAIR	# Params	# Sampling(Steps/Frame)	Time Taken(in hrs)
MCVD [50]	251M	100	2
RaMViD [53]	235M	500	7.2
CVP [74]	118M	25	0.45
CVF (Ours)	40M	5	0.112

Table 2.3: Comparison with baselines on number of parameters, sampling steps and sampling time required for BAIR robot push dataset.

whereas KTH offers single-channel frames. For our evaluation, we adopted a setup similar to that used for KTH, providing 5 context frames and predicting the subsequent 30 future frames. The results of this evaluation are summarized in Table 2.4.

From Table 2.4, it is clear that our model requires significantly fewer training frames to achieve superior results. Specifically, it only uses 6 frames per block (5 context frames and 1 future frame), yet delivers performance that surpasses the baseline methods.

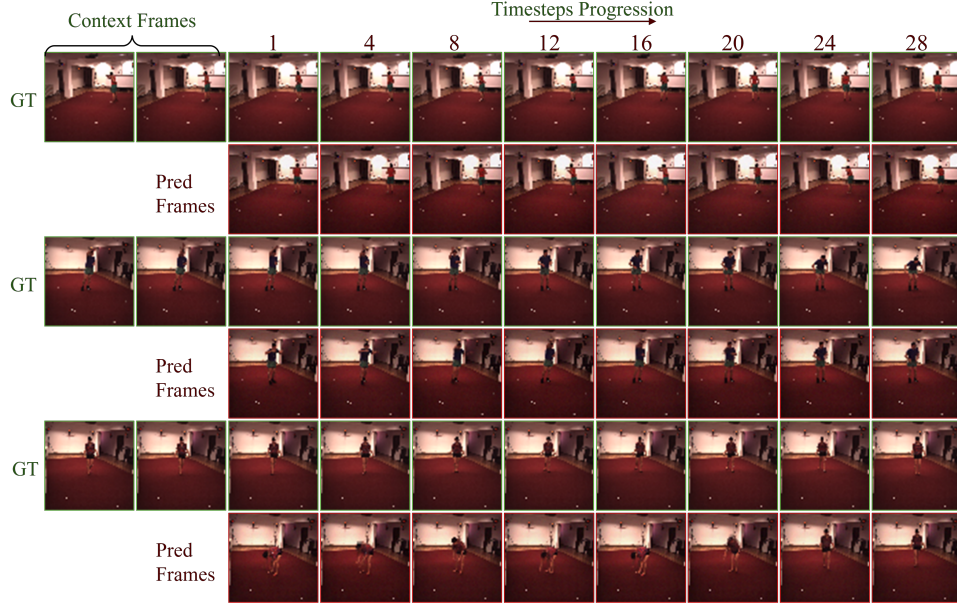


Figure 2.5: Figure represents qualitative results of our CVF model on the Human3.6M dataset. The number of context frames used in the above setting is 4 for all three sequences. Every 4th predicted future frame is shown in the figure.

The results, as shown in Table 2.4, demonstrate that our approach outperforms existing models, establishing a new state-of-the-art on the Human3.6M dataset. Additionally, the qualitative results in Fig. 2.5 highlight our CVF model’s ability to accurately capture and predict the diverse actions within the dataset, further showcasing its effectiveness.

UCF101 Dataset: The UCF101 dataset introduces a significantly higher level of complexity compared to the KTH and Human3.6M datasets, primarily due to its large variety of action categories, diverse backgrounds, and pronounced camera movements. For our frame-conditional generation task, we strictly utilize information from the context frames, without leveraging any additional data, such as class labels, for the prediction task. Our evaluation setup mirrors that used for the Human3.6M dataset, where 5 context frames are provided, and the model is tasked with predicting

Human3.6M	p	k	#pred	FVD↓
SVG-LP [46]	5	10	30	718
Struct-VRNN [67]	5	10	30	523.4
DVG [48]	5	10	30	479.5
SRVP [66]	5	10	30	416.5
Grid keypoint [68]	8	8	30	166.1
CVP [74]	5	1	30	144.5
CVF(Ours)	5	1	30	120.2

Table 2.4: Quantitative comparisons on the Human3.6M dataset. The best results under each metric are marked in bold.

UCF101 [5 $\rightarrow$ 16]	p	k	#pred	FVD↓
SVG-LP [46]	5	10	16	1248
CCVS [71]	5	16	16	409
MCVD [50]	5	5	16	387
RaMViD [53]	5	4	16	356
CVP [74]	5	1	16	245.2
CVF(Ours)	5	1	16	221.7

Table 2.5: Video prediction results on UCF (128×128), predicting 16 frames. All models are conditioned on 5 past frames.

the subsequent 16 frames. The results of this evaluation are presented in Table 2.5.

Upon analyzing Table 2.5, it is evident that our CVF model outperforms all baseline models, setting a new state-of-the-art benchmark for the UCF101 dataset. Furthermore, the qualitative results shown in Fig. 2.6 demonstrate the model’s ability to effectively capture and predict the diverse actions present in this challenging dataset, even without the use of external information like class labels.

2.6 Limitation

While our method demonstrates strong performance in video prediction, it is essential to acknowledge certain limitations that point toward avenues for future

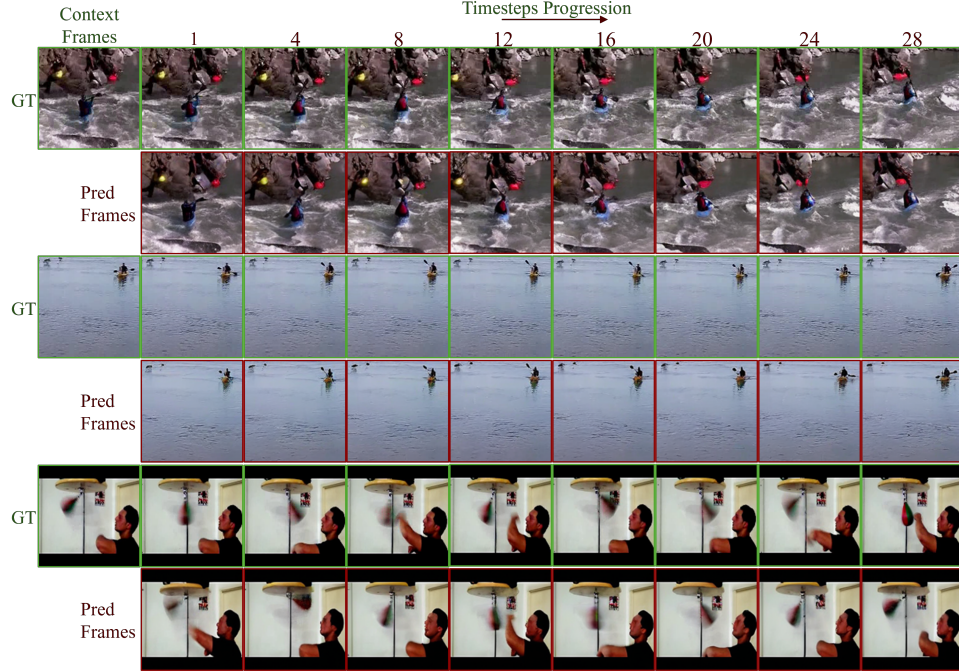


Figure 2.6: Figure represents qualitative results of our CVF model on the UCF dataset. The number of context frames used in the above setting is 5 for all three sequences. Every 4th predicted future frame is shown in the figure.

work.

First, a key limitation lies in computational efficiency. Although our approach requires fewer sampling steps compared to traditional diffusion-based models, generating each frame still demands a sequential process that can become a bottleneck when scaling to longer video sequences or real-time applications. Further optimization, particularly in reducing the number of sampling steps and computational overhead, remains an open challenge.

Second, our experiments were constrained by computational resources, utilizing only two A6000 GPUs. With access to more powerful hardware or distributed computing, there may be potential for significant gains in both model complexity and performance. We encourage future research to investigate the model’s behavior on

larger datasets and with more substantial computational resources, as these factors could reveal additional improvements in video prediction quality.

2.7 Conclusion

In this work, we introduced a novel model architecture designed to more effectively leverage video representations, marking a significant contribution to video prediction tasks. Through extensive experimental evaluation on diverse datasets, including KTH, BAIR, Human3.6M, and UCF101, our approach consistently demonstrated superior performance, setting new benchmarks in state-of-the-art video prediction.

A key strength of our model is its efficient utilization of parameters and significantly lower number of sampling steps during inference compared to existing methods. Notably, our model’s ability to treat video as a continuous process eliminates the need for additional constraints, such as temporal attention mechanisms, which are often used to enforce temporal consistency. This highlights the model’s inherent capacity to maintain temporal coherence naturally, streamlining the video prediction process while improving both efficiency and predictive accuracy.

Chapter 3: Diverse Video Generation using a Gaussian Process Trigger

3.1 Introduction

Humans are often able to imagine multiple possible ways that the scene can change over time. Modeling and generating diverse futures is an incredibly challenging problem. The challenge stems from the inherent multi-modality of the task, *i.e.*, given a sequence of past frames, there can be multiple possible outcomes of the future frames. For example, given the image of a “person holding a cup” in Figure. 3.1, most would predict that the next few frames correspond to either the action “drinking from the cup” or “keeping the cup on the table.” This challenge is exacerbated by the lack of real training data with diverse outputs – all real-world training videos come with a single real future and no “other” potential futures. Similar looking past frames can have completely different futures (*e.g.*, Figure. 3.1). In the absence of any priors or explicit supervision, the current methods struggle with modeling this diversity. Given similar looking past frames, with different futures in the training data, variational methods, which commonly utilize [45], tend to average the results to better match to *all* different futures [34, 42, 46, 80, 81]. We hypothesize that explicit modeling of future diversity is essential for high-quality, diverse future frame generation.



Figure 3.1: Given “person holding cup,” humans can often predict multiple possible futures (e.g., “drinking from the cup” or “keeping the cup on the table.”).

In this paper, we model the diversity of the future states, given past context, using Gaussian Processes ($\mathcal{GP}$) [82], which have several desirable properties. They learn a prior on potential future given past context, in a Bayesian formulation. This allows us to update the distribution of possible futures as more context frames are provided as evidence and maintain a list of potential futures (underlying *functions* in $\mathcal{GP}$). Finally, our formulation provides an interesting property that is crucial to generating future frames – the ability to estimate *when* to generate a diverse output *vs. continue* an on-going action, and a way to *control* the predicted futures.

In particular, we utilize the variance of the $\mathcal{GP}$ at any specific time step as an indicator of whether an action sequence is on-going or finished. An illustration of this mechanism is presented in Figure. 3.2. When we observe a frame (say at time t) that can have several possible futures, the variance of the $\mathcal{GP}$ model is high (Figure. 3.2 (left)). Different functions represent potential action sequences that can be generated, starting from this particular frame. Once we select the next frame (at $t+2$), the $\mathcal{GP}$ variance of the future states is relatively low (Figure. 3.2 (center)), indicating that an action sequence is on-going, and the model should continue it as opposed to trying to sample a diverse sample. After the completion of the on-going sequence, the $\mathcal{GP}$ variance over potential future states becomes high again. This

implies that we can continue this action (*i.e.*, pick the mean function represented by the black line in Figure. 3.2 (center)) or try and sample a potentially diverse sample (*i.e.*, one of the functions that contributes to high-variance). This illustrates how we can use $\mathcal{GP}$ to decide when to trigger diverse actions. An example of using $\mathcal{GP}$ trigger is shown in Figure. 3.2 (right), where after every few frames, we trigger a different action.

Now that we have a good way to model diversity, the next step is to generate future frames. Even after tremendous advances in the field of generative models for image synthesis [42, 46, 80, 83, 84, 85, 86, 87, 88], the task of generating future frames (not necessarily diverse) conditioned on past frames is still hard. As opposed to independent images, the future frames need to obey potential video dynamics that might be on-going in the past frames, follow world knowledge (*e.g.*, how humans and objects interact), *etc.*. We utilize a fairly straightforward process to generate future frames, which utilizes two modules: a frame auto-encoder and a dynamics encoder. The frame auto-encoder learns to encode a frame in a latent representation and utilizes it to generate the original frame back. The dynamics encoder learns to model dynamics between past and future frames. We learn two independent dynamics encoders: an LSTM encoder, utilized to model on-going actions and the $\mathcal{GP}$ encoder (similar to [33]), and a $\mathcal{GP}$ encoder, utilized to model transitions to new actions. The variance of this $\mathcal{GP}$ encoder can be used as a trigger to decide when to sample new actions. We train this framework end-to-end. We first provide an overview of $\mathcal{GP}$ formulation and scalable training techniques in §3.3, and then describe our approach in §3.4.

Comprehensively evaluating diverse future frames generation is still an open research problem. Following recent state-of-the-art, we will evaluate different aspects of the approach independently. The quality of generated frames is quantified using image synthesis/reconstruction per-frame metrics: SSIM [39, 89], PSNR, and LPIPS [90, 91, 92]. The temporal coherence and quality of a short video clip (16 neighboring frames) are jointly evaluated using the FVD [64] metric. However, high-quality, temporarily coherent frame synthesis does not evaluate diversity in predicted frames. Therefore, to evaluate diversity, since there are no multiple ground-truth futures, we propose an alternative evaluation strategy, inspired by [55]: utilizing action classifiers to evaluate whether an *action switch* has occurred or not. A change in action indicates that the method was able to sample a diverse future. Together, these metrics can evaluate if an approach can generate multiple high-quality frames that temporally coherent and diverse. Details of these metrics and baselines, and extensive quantitative and qualitative results are provided in §3.5.

To summarize, our contributions are: (a) modeling the diversity of future states using a $\mathcal{GP}$, which maintains priors on future states given the past frames using a Bayesian formulation (b) leveraging the changing $\mathcal{GP}$ distribution over time (given new observed evidence) to estimate when an on-going action sequence completes and using $\mathcal{GP}$ variance to *control* the triggering of a diverse future state. This results in state-of-the-art results on future frame generation. We also quantify the diversity of the generated sequences using action classifiers as a proxy metric.

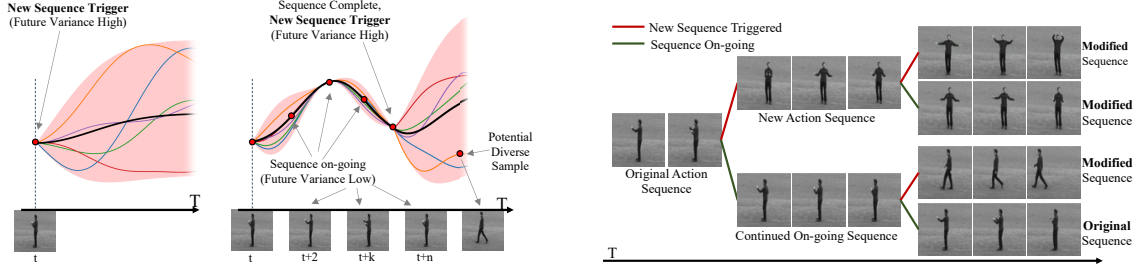


Figure 3.2: An illustration of using $\mathcal{GP}$ variance to control sampling on-going actions *vs.* new actions.

3.2 Related work

Understanding and predicting the future, given the observed past, is a fundamental problem in video understanding. The future states are inherently multi-modal and capturing their diversity finds direct applications in many safety-critical applications (*e.g.*, autonomous vehicles), where it is critical to model different future modes. Earlier techniques for future prediction [31, 32] relied on searching for matches of past frames in a given dataset and transferring the future states from these matches. However, the predictions were limited to symbolic trajectories or retrieved future frames. Given the modeling capabilities of deep representations, the field of future frame prediction tremendous progress in recent years. One of the first works on video generation [33] used a multi-layer LSTM network to learn representations of video sequences in a deterministic way. Since then, a wide range of papers [34, 35, 36, 37, 38, 39, 40] have built models that try to incorporate stochasticity of future states. Generally, this stochasticity lacks diverse high-level semantic actions.

Recently, several video generation models have utilized generative models,

like variational auto-encoders (VAEs) [45] and generative adversarial networks (GANs) [41], for this task. One of the first works by Xue et al. [93] utilized a conditional VAE (cVAE) formulation to learn video dynamics. Similar to our approach, their goal was to model the frame prediction problem in a probabilistic way and synthesizing many possible future frames from a single image. Since then, several works have utilized the cVAE for future generation [46, 80]. The major drawback of using the cVAE approach is that its objective function marginalizes over the multi-modal future, limiting the diversity in the generated samples [94]. GAN-based models are another important class of synthesis models used for future frame prediction or video generation [83, 84, 85, 86, 87, 88]. However, these models are very susceptible to mode collapse [95], *i.e.*, the model outputs only one or a few modes instead of generating a wide range of diverse output. The problem of mode collapse is quite severe for conditional settings, as demonstrated by [96, 97, 98]. This problem is worse in the case of diverse future frame generation due to the inherent multi-modality of the output space and lack of training data.

Another class of popular video generation models is hierarchical prediction [54, 55, 56, 57]. These models decompose the problems into two steps. They first predict a high-level structure of a video, like a human pose, and then leverage that structure to make predictions at the pixel level. These models generally require additional annotation for the high-level structure for training.

Unlike these approaches, Our approach explicitly focuses on learning the distribution of diverse futures using a $\mathcal{GP}$ prior on the future states using a Bayesian formulation. Moreover, such $\mathcal{GP}$ approaches have been used in the past for modeling

the human dynamics as demonstrated by [99, 100, 101]. However, due to the scalability issues in $\mathcal{GP}$, these models were limited to handling low dimensional data, like human pose, lane switching, path planning, etc. To the best of our knowledge, ours is the first approach that can process video sequences to predict when an on-going action sequence completes and control the generation of a diverse state.

3.3 Review of Gaussian Process

A Gaussian process ($\mathcal{GP}$) [82] is a Bayesian non-parametric approach that learns a joint distribution over functions that are sampled from a multi-variate normal distribution. Consider a data set consisting of n data-points $\{\text{inputs}, \text{targets}\}_1^n$, abbreviated as $\{X, Y\}_1^n$, where the inputs are denoted by $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, and targets by $Y = \{\mathbf{y}_1, \dots, \mathbf{y}_n\}$. The goal of $\mathcal{GP}$ is to learn an unknown function f that maps elements from input space to a target space. A $\mathcal{GP}$ regression formulates the functional form of f by drawing random variables from a multi-variate normal distribution given by $[f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_n)] \sim \mathcal{N}(\mu, K_{X,X})$, with mean μ , such that $\mu_i = \mu(\mathbf{x}_i)$, and $K_{X,X}$ is a covariance matrix. $(K_{X,X})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, where $k(\cdot)$ is a kernel function of the $\mathcal{GP}$. Assuming the $\mathcal{GP}$ prior on $f(X)$ with some additive Gaussian white noise $\mathbf{y}_i = f(\mathbf{x}_i) + \epsilon$, the conditional distribution at any unseen

points X_* is given by:

$$\mathbf{f}_*|X_*, X, Y \sim \mathcal{N}(E[\mathbf{f}_*], \text{Cov}[\mathbf{f}_*]), \text{ where} \quad (3.1)$$

$$E[\mathbf{f}_*] = \mu_{X_*} + K_{X_*,X}[K_{X,X} + \sigma^2 I]^{-1}Y$$

$$\text{Cov}[\mathbf{f}_*] = K_{X_*,X_*} - K_{X_*,X}[K_{X,X} + \sigma^2 I]^{-1}K_{X,X_*}$$

3.3.1 Learning and Model Selection

We can derive the marginal likelihood for the $\mathcal{GP}$ by integrating out the $f(x)$ as a function of kernel parameters alone. Its logarithm can be defined analytically as:

$$\log(p(Y|X)) = -\frac{1}{2} \left(Y^T (K_\theta + \sigma^2 I)^{-1} Y + \log |K_\theta + \sigma^2 I| \right) + \text{const}, \quad (3.2)$$

where θ denotes the parameters of the covariance function of kernel $K_{X,X}$. Notice that the marginal likelihood involves a matrix inversion and evaluating a determinant for $n \times n$ matrix. A naïve implementation would require cubic order of computations $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ of storage, which hinders the use of $\mathcal{GP}$ for a large dataset. However, recent researches have tried to ease these constraints under some assumptions.

3.3.2 Scalable $\mathcal{GP}$

The model selection and inference of $\mathcal{GP}$ requires a cubic order of computations $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ of storage which hinders the use of $\mathcal{GP}$ for a large dataset. Titsias

[102] proposed a new variational approach for sparse approximation of the standard $\mathcal{GP}$ which jointly infers the inducing inputs and kernel hyperparameters by optimizing a lower bound of the true log marginal likelihood, resulting in $\mathcal{O}(nm^2)$ computation, where $m < n$. Hensman et al. [103] proposed a new variational formulation of true log marginal likelihood that resulted in a tighter bound. Another advantage of this formulation is that it can be optimized in a stochastic [103] or distributed [104, 105] manner, which is well suited for our frameworks which use stochastic gradient descent. Further, recent works [106, 107, 108] have improved the scalability by reducing the learning to $\mathcal{O}(n)$ and test prediction to $\mathcal{O}(1)$ under some assumptions.

In this work, for scalability, we leverage the SVGP approach proposed by Hensman et al. [103]. It proposes a tighter bound for the sparse GP introduced by Titsias [102], which uses pseudo inputs $\mathbf{u}$ to lower bound the log joint probability over targets and pseudo inputs. SVGP introduces a multivariate normal variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$, where the parameters $\mathbf{m}$ and $\mathbf{S}$ are optimized using the evidence lower bound or ELBO (eq. 3.3) of true marginal likelihood (eq. 3.2). The pseudo inputs, $\mathbf{u}$, depend on variational parameters $\{\mathbf{z}_m\}_{m=1}^M$, where $M = \dim(\mathbf{u}) \ll N$. Therefore, the ELBO for SVGP is

$$\mathcal{L}_{\text{svgp}}(Y, X) = \mathbb{E}_{q(\mathbf{u})} [\log p(Y, \mathbf{u}|X, Z)] + H[q(\mathbf{u})], \quad (3.3)$$

where the first term was proposed by Titsias [102], and the modification by Hensman et al. [103] introduces the second term. For details about the pseudo inputs $\mathbf{u}$ and variational parameters $\mathbf{z}_i$, we refer the readers to [102, 103].

In this work, we build on the sparse $\mathcal{GP}$ approach from GPytorch [109], which implements [103].

3.4 Our Approach

Given a set of observed frames, our goal is to generate a diverse set of future frames. Our model has three modules: (a) a frame auto-encoder (or encoder-generator), (b) an LSTM temporal dynamics encoder, and (c) a $\mathcal{GP}$ temporal dynamics encoder to model priors and probabilities over diverse potential future states.

The frame encoder maps the frames onto a latent space, that is later utilized by temporal dynamics encoders and frame generator to synthesize the future frames. For inference, we use all three modules together to generate future frames, and use the $\mathcal{GP}$ as a trigger to switch to diverse future states. Below we describe each module in detail.

3.4.1 Frame Auto-Encoder

The frame encoder network is a convolution encoder which takes frame $\mathbf{z}_{xt} \in \mathbb{R}^{H \times W}$ and maps them to the latent space $\mathbf{z}_t \in \mathbb{R}^d$, where $H \times W$ is the input frame size and d is the dimension of latent space respectively. Therefore, $f_{\text{enc}} : \mathbb{R}^{H \times W} \rightarrow \mathbb{R}^d$, *i.e.*, $\mathbf{z}_t = f_{\text{enc}}(\mathbf{z}_{xt})$. Similarly, the decoder or generator network, utilizes the latent feature to generate the image. Therefore, $f_{\text{gen}} : \mathbb{R}^d \rightarrow \mathbb{R}^{H \times W}$, *i.e.*, $\hat{\mathbf{z}}_{xt} = f_{\text{gen}}(\mathbf{z}_t)$. We borrow the architectures for both encoder and generator networks from [46],

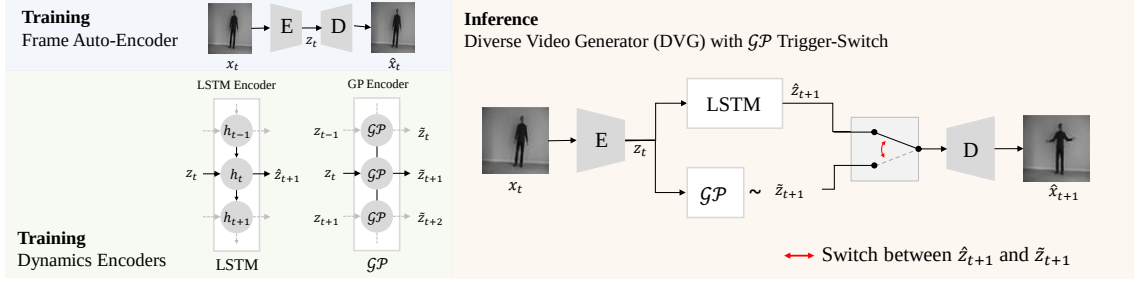


Figure 3.3: An overview of the proposed Diverse Video Generator (**DVG**).

where the frame encoder is convolutional layers from VGG16 network [110] and the generator is a mirrored version of the encoder with pooling layers replaced with spatial up-sampling, a sigmoid output layer, and skip connections from the encoder network to reconstruct image. $\mathcal{L}_{\text{gen}}(\mathbf{z}_{xt}, \hat{\mathbf{z}}_{xt}) = \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|^2$ is the reconstruction loss for frame auto-encoder. This auto-encoder is illustrated in Figure. 3.3 (stage 1).

3.4.2 LSTM Temporal Dynamics Encoder

The first dynamics we want to encode is of an on-going action sequence, *i.e.*, if an action sequence is not completed yet, we want to continue generating future frames of the same sequence till it finishes. This module $f_{\text{LSTM}} : \mathbb{R}^d \rightarrow \mathbb{R}^d$, has a fully-connected layer, followed by two LSTM layers with 256 cells each, and a final output fully-connected layer. The output fully-connected layer takes the last hidden state from LSTM ($\mathbf{h}_{t+1}$) and outputs $\hat{\mathbf{z}}_{t+1}$ after $\tanh(\cdot)$ activation. Therefore, $\hat{\mathbf{z}}_{t+1} = f_{\text{LSTM}}(\mathbf{z}_t)$. The training loss is given by $\mathcal{L}_{\text{LSTM}} = \sum_{t=1}^T \|\mathbf{z}_{t+1} - \hat{\mathbf{z}}_{t+1}\|^2$, where T are the total number of frames (both past and future) used for training. This simple dynamics encoder, inspired by [33] and illustrated in Figure. 3.3 (stage 2), is effective and performs well on standard metrics.

3.4.3 $\mathcal{GP}$ Temporal Dynamics Encoder

Next, we want to learn the priors for potential future states by modeling the correlation between past and future states using a $\mathcal{GP}$ layer. Given a past state, this temporal dynamics encoder captures the distribution over future states. This enables us to use the predictive variance to decide when to sample diverse outputs, and provides us with a mechanism to sample diverse future states. The input to the $\mathcal{GP}$ layer is $\mathbf{z}_t$ and the output is a mean and variance, which can be used to sample $\tilde{\mathbf{z}}_{t+1}$. The loss function for the $\mathcal{GP}$ dynamics encoder follows from eq. 3.3, $\mathcal{L}_{\text{GP}} = -\mathcal{L}_{\text{svgp}}(\mathbf{z}_{t+1}, \mathbf{z}_t)$.

Unlike LSTM, the $\mathcal{GP}$ layer does not have hidden or transition states; it only models pair-wise correlations between past and future frames (illustrated in Figure. 3.3 (stage 2)). In this work, we use the automatic relevance determination (ARD) kernel, denoted by $k(\mathbf{z}, \mathbf{z}') = \sigma_{\text{ARD}}^2 \exp\left(-0.5 \sum_{j=1}^d \omega_j (z_j - z'_j)^2\right)$, parameterized by learnable parameters σ_{ARD} and $\{\omega_1, \dots, \omega_d\}$. This $\mathcal{GP}$ layer is implemented using GPyTorch [109] (refer to §3.3).

3.4.4 Training Objective

All three modules, frame auto-encoder and the LSTM and $\mathcal{GP}$ temporal encoders, are jointly trained using the following objective function:

$$\mathcal{L}_{\text{DVG}} = \sum_{t=1}^T \left(\underbrace{\lambda_1 \mathcal{L}_{\text{gen}}(\mathbf{z}_{xt}, \hat{\mathbf{z}}_{xt})}_{\text{Frame Auto-Encoder}} + \underbrace{\lambda_2 \mathcal{L}_{\text{gen}}(\mathbf{z}_{xt}, f_{\text{gen}}(\hat{\mathbf{z}}_t))}_{\text{LSTM Frame Generation}} + \underbrace{\lambda_3 \mathcal{L}_{\text{gen}}(\mathbf{z}_{xt}, f_{\text{gen}}(\tilde{\mathbf{z}}_t))}_{\text{\mathcal{GP} Frame Generation}} + \right. \\ \left. \underbrace{\lambda_4 \mathcal{L}_{\text{LSTM}}(\mathbf{z}_{t+1}, \hat{\mathbf{z}}_{t+1})}_{\text{LSTM Dynamics Encoder}} + \underbrace{\lambda_5 \mathcal{L}_{\text{GP}}(\mathbf{z}_{t+1}, \mathbf{z}_t)}_{\text{\mathcal{GP} Dynamics Encoder}} \right) \quad (3.4)$$

where $[\lambda_1, \dots, \lambda_5]$ are hyperparameters. There are three frame generation losses, each utilizing different latent code: $\mathbf{z}_t$ from frame encoder, $\hat{\mathbf{z}}_t$ from LSTM encoder, and $\tilde{\mathbf{z}}_t$ from $\mathcal{GP}$ encoder. In addition, there are two dynamics encoder losses, one each for LSTM and $\mathcal{GP}$ modules. Empirically, we observed that the model trains better with higher values for $\lambda_1, \lambda_2, \lambda_4$, possibly because $\mathcal{GP}$ is only used to sample a diverse state and all other states are sampled from the LSTM encoder.

3.4.5 Inference Model of Diverse Video Generator (DVG)

During inference, we put together the three modules described above as follows. The output of the frame encoder $\mathbf{z}_t$ is given as input to both LSTM and $\mathcal{GP}$ encoders. The LSTM outputs $\hat{\mathbf{z}}_{t+1}$ and the $\mathcal{GP}$ outputs a mean and a variance. The variance of $\mathcal{GP}$ can be used to decide if we want to continue an on-going action or generate new diverse output, a process we call **trigger switch**. If we decide to stay with the

on-going action, LSTM’s output $\hat{\mathbf{z}}_{t+1}$ is provided to the decoder to generate the next frame. If we decide to *switch*, we sample $\tilde{\mathbf{z}}_{t+1}$ from the $\mathcal{GP}$ and provide that as input to the decoder. This process is illustrated in Figure. 3.3 (stage 3). The generated future frame is used as input to the encoder to output the next $\mathbf{z}_{t+1}$; this process is repeated till we want to generate frames.

3.4.6 Trigger Switch Heuristics

An important decision for a diverse future generation is when to continue the current action and when to switch to a new action. We use the $\mathcal{GP}$ to switch to new states. We use two heuristics to decide when to generate diverse actions: a deterministic switch and a $\mathcal{GP}$ trigger switch. For the deterministic switch, we do not use the variance of the $\mathcal{GP}$ as a trigger, and switch every 15 frames. Each switch uses the sample from $\mathcal{GP}$ as the next future state. This enables us to have a consistent sampling strategy across generated samples. For the $\mathcal{GP}$ trigger switch, we compare the current $\mathcal{GP}$ variance with the mean of the variance of the last 10 states. If the current variance is larger than two standard deviations, we trigger a switch. This variable threshold allows the diverse video generator to trigger switches based on evidence, which can vary widely across samples.

3.5 Experiments

Next, we describe the experimental setup, datasets we use, qualitative and quantitative results. We evaluate our models on four datasets and compare it with

the state-of-the-art baselines. All models use 5 frames as context (past) during training and learn to predict the next 10 frames. However, our model is not limited to generating just 10 frames. All our models are trained using Adam optimizer.

KTH Action Recognition Dataset. The KTH action dataset [61] consists of video sequences of 25 people performing six different actions: walking, jogging, running, boxing, hand-waving, and hand-clapping. The background is uniform, and a single person is performing actions in the foreground. Foreground motion of the person in the frame is fairly regular.

BAIR pushing Dataset. The BAIR robot pushing dataset [62] contains the videos of table mounted sawyer robotic arm pushing various objects around. The BAIR dataset consists of different actions given to the robotic arm to perform.

Human3.6M Dataset. Human3.6M [111] dataset consists of 10 subjects performing 15 different actions. We did not use the pose information from the dataset.

UCF Dataset. This dataset [2] consists of 13,320 videos belonging to 101 different action classes. We sub-sample a small dataset for qualitative evaluation on complex videos. Our subset consists of 7 classes: Bench press, Bodyweight squats, Clean and Jerk, Pull-ups, Push-ups, Shotput, Tennis-Swing, Lunges and Fencing. The background of this dataset can bias our diversity evaluation metric. Therefore, we only include this dataset only for qualitative evaluation.

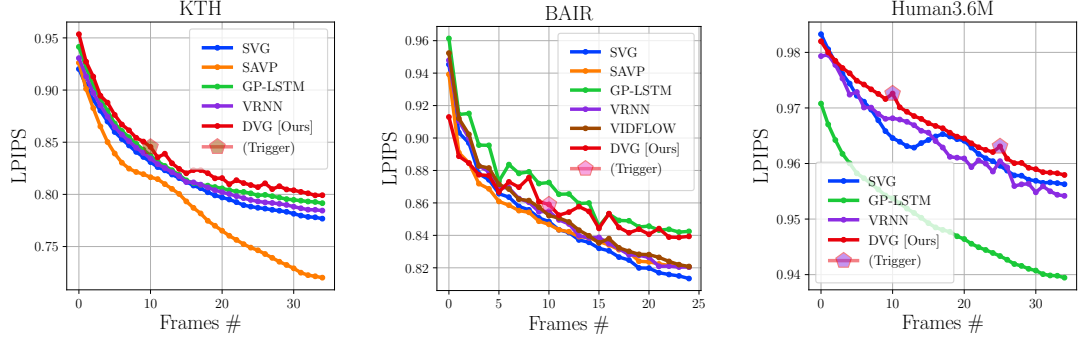


Figure 3.4: **LPIPS Quantitative Results** on KTH, BAIR, and Human3.6M datasets. All methods use the best matching sample out of 100 random samples. We used fixed trigger heuristic to keep trigger point for each sample the same for our approach.

3.5.1 Baselines

We compared our method with the following prior works. Further, wherever available, we used either the official implementation or the pre-trained models for the baselines uploaded by their respective authors.

SVG-LP [46]: Stochastic Video Generation with a Learned Prior (SVG-LP) is a VAE-based method that uses a latent space prior for generating video sequences. It outperforms other VAE-based approaches (*e.g.*, SV2P [80]). It also uses an LSTM-based dynamics model. This model similar to ours except that we use a $\mathcal{GP}$ to model the prior on future states to aid with multi-modal outputs.

SAVP [42]: Stochastic Adversarial Video Prediction (SAVP) is a VAE-GAN based approach that combines the best of both families of approaches. It also uses the LSTM dynamics model.

Conditional VRNN [40]: Condition variational RNN leverages the flexibility of hierarchical latent variable models to increase the expressiveness of the latent space.

VidFlow [112]: VideoFlow model uses a normalizing flow approach that enables direct optimization of the data likelihood.

GP-LSTM: We train a model inspired by [113], where the dynamics model is a $\mathcal{GP}$, which uses recurrent kernels modeled by an LSTM. This method is closest to ours since it utilizes the constructs of both $\mathcal{GP}$ and LSTM. However, they train a single dynamics model and have no way to *control* the generation of future states.

We refer to our model as **Diverse Video Generator (DVG)**, which uses a $\mathcal{GP}$ trigger switch. We also study heuristic switching at **15** and **35** frames for ablation analysis.

3.5.2 Metrics

Evaluation of generated videos in itself is an open research problem with new emerging metrics. In this work, we tried our best to cover all published metrics which have been used for evaluating future frame generation models.

Accuracy of Reconstruction. One way to evaluate a video generation model is to check how close the generated frames are to the ground-truth. Since the models are intended to be stochastic or diverse for variety in prediction, this is achieved by sampling a finite number of future sequences from a model and evaluating the similarity between the best matching sequence to the ground-truth and the ground truth sequence. Previous works [42, 46, 80] used traditional image reconstruction metrics, SSIM and PSNR, to measure the similarity between the generated samples and ground-truth. As shown by [64, 90, 91, 92], these metrics are not suited for

Model	Trigger	FVD Score ($\downarrow$)			Diversity Score ($\uparrow$) (frames: [10,25])		Diversity Score ($\uparrow$) (frames: [25,40])	
		KTH	BAIR	Human3.6M	KTH	Human3.6M	KTH	Human3.6M
SVG-LP	-	156.35	270.04	718.04	20.10	4.8	21.20	4.6
SAVP	-	65.98	126.75	-	26.60	-	24.50	-
GP-LSTM	-	92.34	197.49	604.75	31.40	5.4	30.90	6.0
VidFlow	-	-	124.81	-	-	-	-	-
VRNN	-	67.26	134.81	523.45	32.50	5.6	31.80	5.9
DVG [ours]	@15,35	65.69	123.08	479.43	48.30	9.3	46.20	9.0
DVG [ours]	$\mathcal{GP}$	69.63	120.03	496.89	47.71	10.8	48.10	10.1

Table 3.1: **Quantitative results** on KTH, BAIR, Human3.6M datasets. For the **FVD Score**, all methods use the best matching sample out of 100 random samples and lower numbers are better. For the **Diversity Score**, we compute the score across 50 generated samples, for 500 starting sequences, and higher numbers are better.

video evaluation because of their susceptibility towards perturbation like blurring, structural distortion, *etc.*. We evaluate the similarity of our generated sequences using recently proposed perceptual metrics, VGG cosine similarity (LPIPS), and Frechet Video Distance (FVD) score. **LPIPS** or Learned Perceptual Image Patch Similarity is a metric developed to quantify the perceptual distance between two images using deep features. Several works [90, 91, 92] show that this metric is much more robust to perturbation like distortion, blurriness, warping, color shift, lightness shift, *etc.* **Frechet Video Distance** (FVD score) [64] is a deep metric designed to evaluate the generated video sequences. As is standard practice, all methods use the best matching sample out of 100 randomly generated samples.

Diversity of Sequences. Reconstruction accuracy of generated samples only implies that there is at least one generated sequence close to the ground-truth. However, these metrics do not capture the inherent multi-modal nature of the task. Aside from being able to generate samples close to ground truth, a video generation model

should be able to represent diversity in its generated video sequences. Therefore, we propose a metric inspired by [55] that utilizes a video classifier to quantify the diversity of generated sequences. The action classifier is trained on the respective datasets (KTH and Humans3.6M). Note that we cannot utilize this metric for the BAIR dataset since we do not have corresponding action labels. For the diversity metric, we use 500 starting sequences of 5 frames, sample 50 future sequences of 40 frames. We ignore the first 5 generated frames since they are likely correlated with the ground-truth and correspond to the on-going sequences. Then, we evaluate the next two clips, made of frames $[10, 25]$ and $[25, 40]$. Ideally, a method that can generate diverse sequences will generate more diverse clips as time progresses. For the **Diversity Score**, we compute the mean number of generated clips that changed from the on-going action as classified by the classifier. More concretely, if N is the total number of generated clips ($N = 25000$ for us), c_i is the ground-truth class, $\hat{c}_i$ is the predicted class, and $\mathbb{1}(\cdot)$ is the indicator function if the parameter is correct, then Diversity Score = $\frac{1}{N} \sum_N \mathbb{1}(c_i \neq \hat{c}_i)$.

3.5.3 Results

3.5.3.1 Quantitative Results (Reconstruction)

We report the quantitative results on KTH, BAIR, and Human3.6M datasets using the LPIPS metric in Figure. 3.4, and FVD metric in Table 3.1. For comparisons with baselines in Figure. 3.4, we see that on the KTH and Human3.6M dataset, our approach generally performs on-par or better than the baselines. In fact, except

for SAVP, all methods are very close to each other. For the Human3.6M dataset, the GP-LSTM baseline performs poorly, and all other methods are similar, with ours being better than others. On the BAIR dataset, we notice that our GP-LSTM baseline performs better, closely followed by our approach. Again, SAVP performs worse on all metrics. For the FVD metric (Table 3.1), variants of our approach achieve state-of-the-art results on all datasets. Note that using a fixed trigger at frames **15** and **35** leads to better FVD scores for KTH and Human3.6M dataset, while $\mathcal{GP}$ trigger performs better for the BAIR dataset. All settings, except one, of our approach perform better than the baselines.

3.5.3.2 Quantitative Results (Diversity)

We report the quantitative results using the proposed diversity score in Table 3.1. We notice for the KTH dataset that SVG-LP/SAVP baseline change actions 20.1%/26.6% of the time in the first clip and 21.2%/24.5% of the time in the second clip. In comparison, our approach gives the diversity score of 48.53% and 48.10% for the first and second clips, respectively. As can be observed, the $\mathcal{GP}$ trigger results in considerably higher diversity as the sequence progresses. On the Human3.6M dataset, the difference between the scores of baselines and our methods is $\sim 6\%$. The overall score drop between the KTH and the Human3.6M datasets on diversity metric can be accounted to actions performed in the videos are very distinct. Besides, cameras are placed far off from the person performing actions making it harder for the models to generate diverse samples.

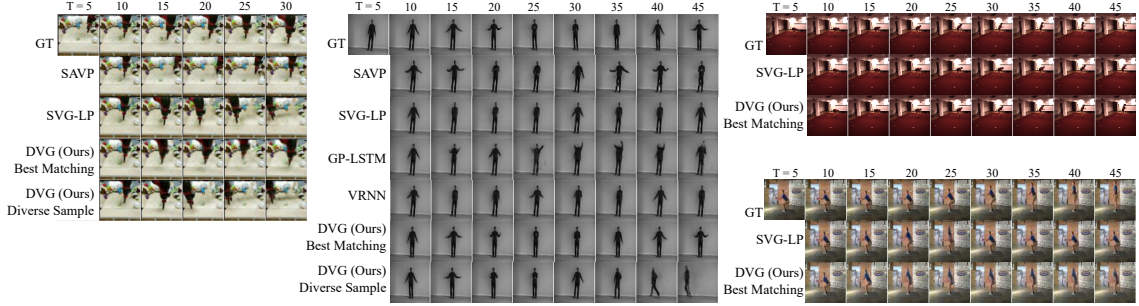


Figure 3.5: **Qualitative Results** on BAIR (left), KTH (center), Human3.6M (right, top), and UCF (right, bottom) datasets. First row is the ground-truth video in each figure (with the last frame of the provided 5 frames is shown as ‘GT’). Every 5th frame is shown.

We further analyze common action triggers for the $\mathcal{GP}$ trigger and notice that it separates the moving (walk, jog, run) and still (clap, wave, box) actions, and common action switches are within each cluster; *e.g.*, walk $\leftrightarrow$ jog, wave $\leftrightarrow$ clap.

3.5.3.3 Qualitative Results

Qualitative Results are shown in Figure. 3.5. For KTH results in Figure. 3.5, we plot a randomly selected sample for all methods. As we can see, SAVP and SVG-LP output average or blurry images after 20-30 frames, and our method is able to switch between action classes (for diverse sample using $\mathcal{GP}$ trigger). For the BAIR dataset, we show the best LPIPS results for all approaches; where we can see that our method generates samples much closer to the ground-truth. We also included a random sample with a fixed trigger at 15th frame, where we can see a change in the action. For the Human3.6M dataset (after digital zoom), we can see that our best LPIPS sample matches the ground-truth person’s pose closely as opposed to SVG-LP, demonstrating the effectiveness of our approach. Similar results are observed for the UCF example.

3.6 Conclusion

We propose a method for diverse future video generation. We model the diversity in the potential future states using a $\mathcal{GP}$, which maintains priors on future states given the past and a probability distribution over possible futures given a particular sample. Since this distribution changes with more evidence, we leverage its variance to estimate when to generate from an on-going action and when to switch to a new and diverse future state. We achieve state-of-the-art results for both reconstruction quality and diversity of generated sequences.

3.7 Appendix

3.7.1 Ablation Studies for Temporal Dynamics Encoder

We perform ablation studies on our model by trying different variants of recurrent modules for our temporal dynamics encoder networks. These models are: **DVG-RNN**, our model with an RNN dynamics encoder; **DVG-GRU**, with an RNN dynamics encoder with GRU units.

Figure. 3.6 shows ablation analysis for different variants of our approach. On the KTH dataset, different dynamics models (RNN, GRU, LSTM) all perform the same. On the BAIR dataset, RNN perform poorly and LSTM performs the best among the three. On Human3.6M dataset RNN performs higher than our LSTM and GRU models. On the FVD metric in Table 3.2, all variants of our approach perform better than all baselines. In approaches, GRU dynamics model performs

better on KTH and LSTM performs better on Human3.6M and BAIR dataset.

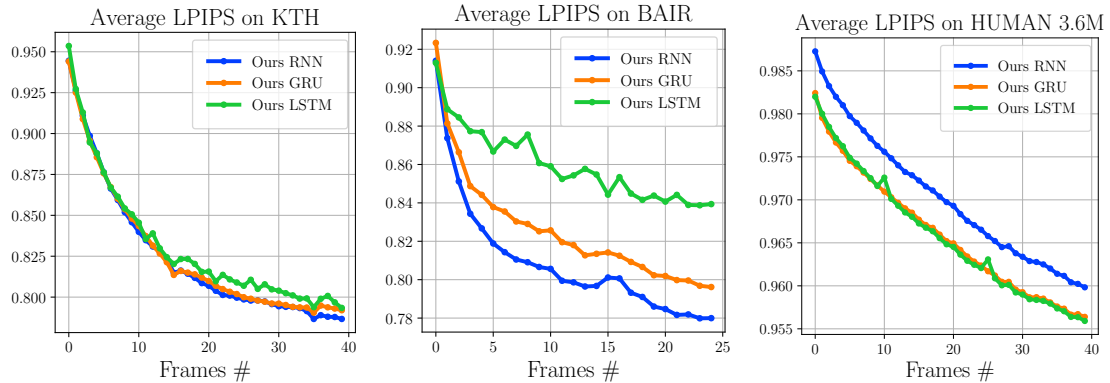


Figure 3.6: **Ablation results** on KTH, Human3.6M and BAIR dataset using variants of temporal dynamics model in our method. We report best LPIPS metric. All methods use the best matching sample out of 100 random samples. We used fixed trigger to keep trigger point for each sample the same. On KTH, all temporal dynamics models have similar performance; and on BAIR, our LSTM model have best performance.

Model	Dynamics	Trigger	FVD Score ($\downarrow$)			Diversity Score ($\uparrow$) (frames: [10,25])		Diversity Score ($\uparrow$) (frames: [25,40])	
			KTH	BAIR	Human3.6M	KTH	Human3.6M	KTH	Human3.6M
DVG [ours]	LSTM	@15,35	65.69	123.08	479.43	48.30	9.3	46.20	9.0
DVG [ours]	GRU	@15,35	64.89	124.38	485.96	48.53	8.5	44.23	9.1
DVG [ours]	RNN	@15,35	66.84	126.07	503.64	46.60	7.6	41.50	8.2

Table 3.2: **Quantitative results** on KTH, BAIR, Human3.6M datasets. For the **FVD Score**, all the ablation methods use the best matching sample out of 100 random samples and lower numbers are better. For the **Diversity Score**, we compute the score across 50 generated samples, for 500 starting sequences, and higher numbers are better.

3.7.2 Analysis: Changes in action after GP triggering

KTH action dataset comprises of 6 action classes: walking, running, jogging, waving, clapping, and boxing. On an abstract level, we can cluster these actions into two categories moving actions and still actions. From the Figure. 3.7, it is interesting to observe that our GP triggering model captures the future trajectories of the videos and clusters them into these two categories moving actions and still

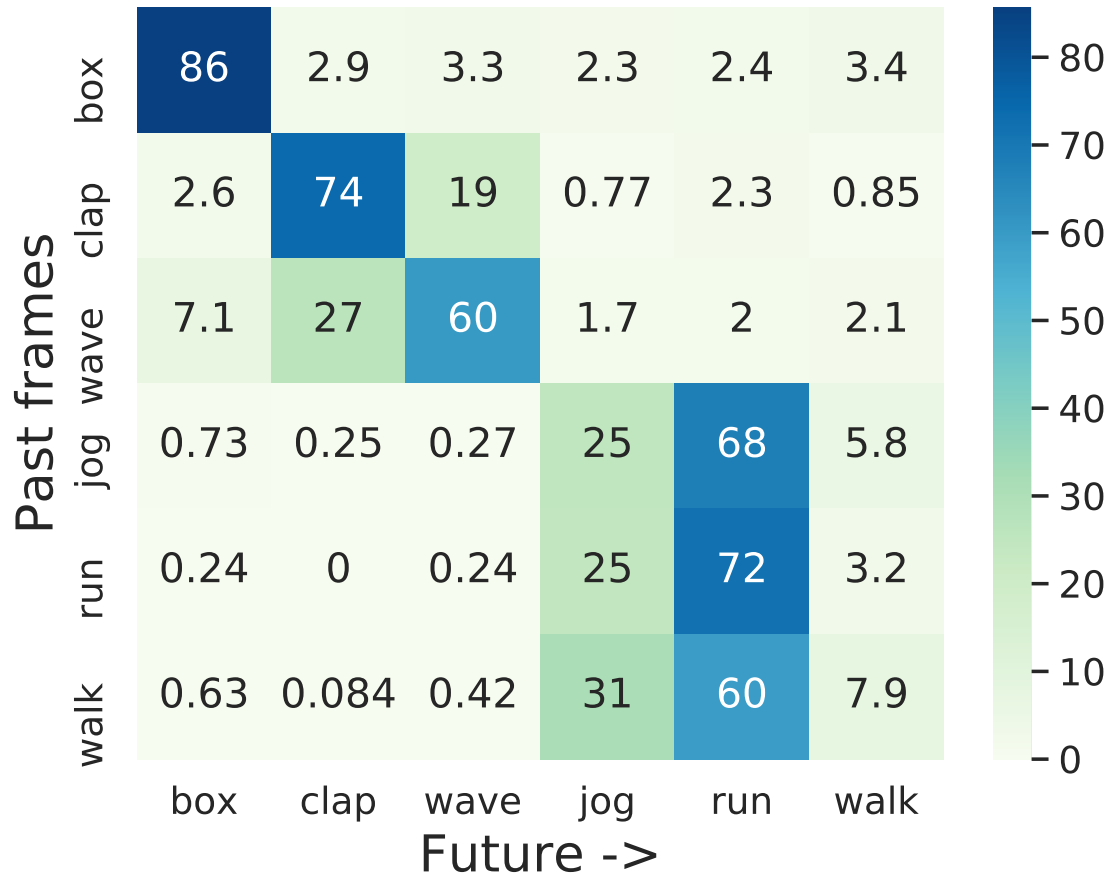


Figure 3.7: **Changes in action from past frames to future frames** on KTH dataset. Total of 25,000 generated videos were used to calculate percentage change shown in the above figure.

actions. Common action switches that are to be expected can be observed from the Figure. 3.7; for example, walk and jog, wave and clap interchange frequently after triggering. Still actions seldom change to moving actions.

3.7.3 SSIM and PSNR Results

We evaluated our generated video sequences using the tradition metrics like structural similarity index (SSIM) and peak signal-to-noise ratio (PSNR) for comparison with previous baselines which reported these metrics. We trained all models on 64×64 -size frames from the KTH, Human3.6M, and BAIR datasets. We used the standard training practice of using 5 frames as context (or past) and the model have to predict the next 10 frames. For all methods, SSIM and PSNR is computed by drawing 100 samples from the model for each test sequence and picking the best score with respect to the ground truth. We emphasize that these results are only for completeness and we hope that the community will stop relying on such reconstruction metrics for video prediction.

Results are reported in Figure. 3.8 represent the evaluation plots for traditional metrics on KTH, BAIR, and Human3.6M dataset. We follow the experimental setups from the baseline papers.

3.7.4 Qualitative Results

It can be observed from Figure. 3.9 that after 15th frame SVG-LP is stuck in the same pose while after 35th frame SAVP starts distorting the human. However, our method (DVG) consistently generates frames that are diverse and distortion free for longer period of time. Similarly, in Figure. 3.14 it can be observed that after 30th frame SVG-LP and SAVP start generating subpar frames while our method is able to generate visually acceptable sequences for longer term. Few additional

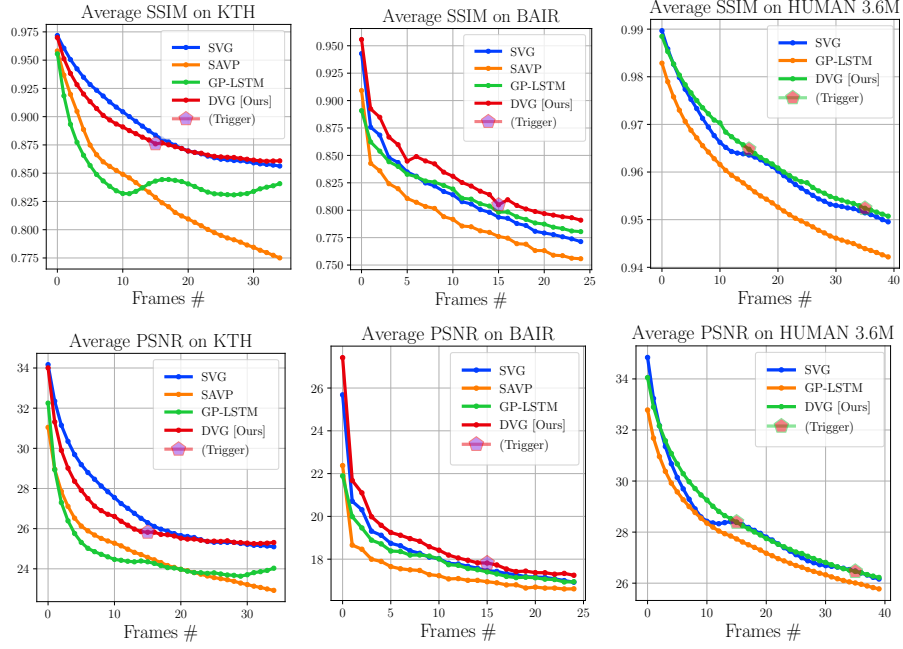


Figure 3.8: **Quantitative results** on KTH, BAIR and Human3.6M dataset. We report average SSIM and PSNR metrics. All methods use the best matching sample out of 100 random samples. We used fixed trigger to keep trigger point for each sample the same.

qualitative results on the BAIR dataset are provided in Figures. 3.10-3.11, and on the Human3.6M dataset in Figures. 3.12-3.13.

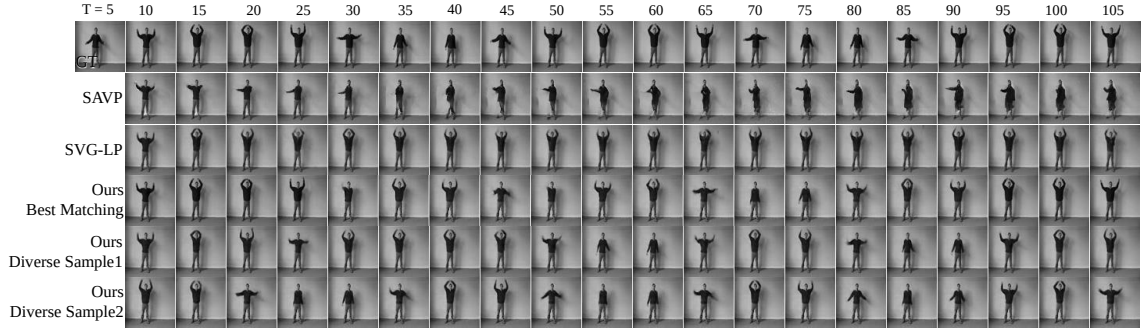


Figure 3.9: **KTH dataset:** Qualitative comparison of the generated video sequences (every 5th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)

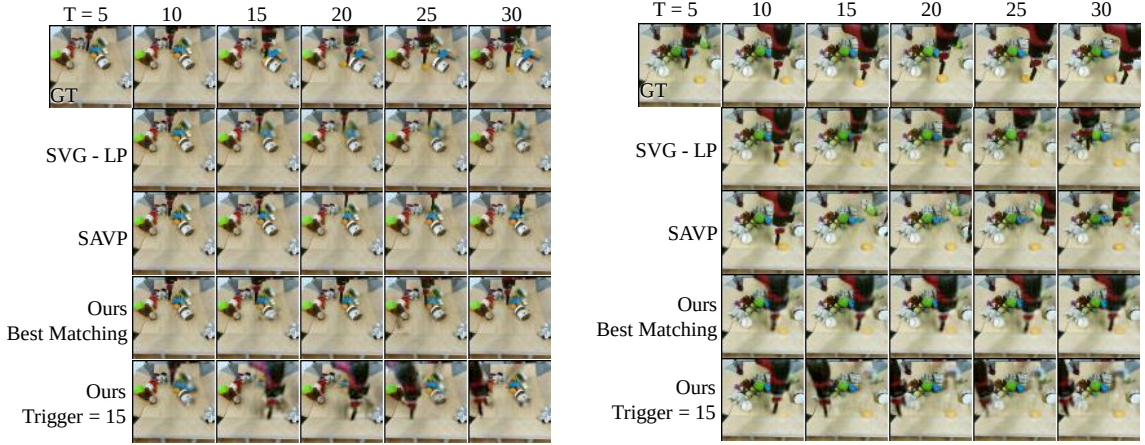


Figure 3.11: **Qualitative results** on BAIR dataset. We show the best LPIPS samples out of 100 samples for all methods.

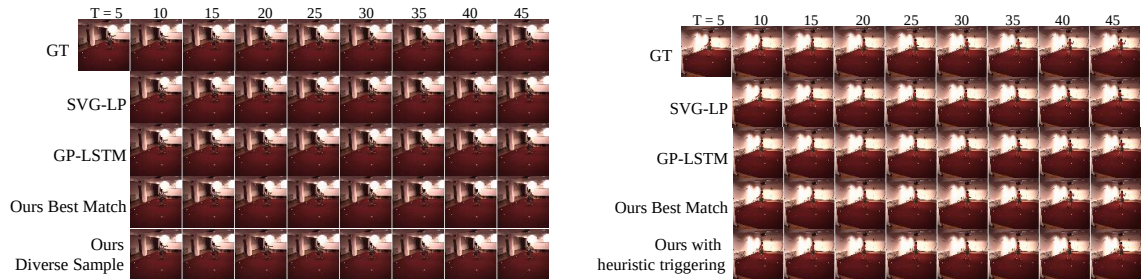


Figure 3.13: **Human3.6M dataset:** Qualitative comparison of the generated video sequences (every 5th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)

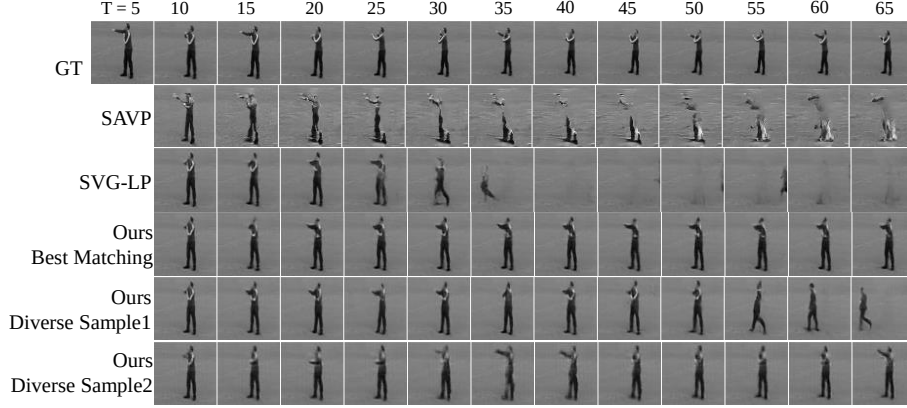


Figure 3.14: **KTH dataset:** Qualitative comparison of the generated video sequences (every 5th frame shown). First row is the ground-truth video (with last frame of the provided 5 frames is shown)

3.7.5 Gaussian Layer Specifics

As mentioned in the paper, GPytorch was used for our GP layer implementation. We utilized a large-scale variational GP implementation of GPytorch for our multi-dimensional GP regression problem of learning to predict the variance over the future frames in the latent space. For variational GP implementation, 40 inducing points were randomly initialized and learned during the training of GP. We used a RBF kernel along with gaussian likelihood for our GP layer. For optimization of our GPLayer, we employed stochastic optimization technique (Adam optimizer) to minimize the variational ELBO for a GP.

3.7.6 I3D Network architecture for Action Classifier

For our diversity metric mentioned in §3.5, we utilized the standard kinetics-pretrained I3D action recognition classifier. The input to the action classifier is a 15 frames clip and each frame has a size of 64×64 . The action classifier attains

accuracy close to 100% for KTH dataset and is above 90% accuracy for human3.6m dataset.

Chapter 4: Video Dynamics Prior: An Internal Learning Approach for Robust Video Enhancements

4.1 Introduction

Recently, video enhancement [3, 11, 114, 115, 116, 117, 118, 119] and editing [12, 13, 14, 120] tasks have attracted much attention in the computer vision community. Most of these tasks are ill-posed problems, i.e., they do not have a unique solution. For example, enhancement tasks such as video frame interpolation can have infinitely many plausible solutions. Therefore, the aim is to find a visually realistic solution coherent in both the space and time domain for such tasks.

Priors play a critical role in finding reasonable solutions to these ill-posed problems by learning spatio-temporal constraints. In recent years, deep learning has emerged as the most promising technique [3, 11, 48, 58, 114, 121, 122, 123, 124] for leveraging data to create priors that enforce these spatio-temporal constraints. However, the generalization of these data-driven priors mainly relies on data distribution at training time.

In the current era of short video platforms such as TikTok, Instagram reels, YouTube shorts, and Snap, millions of new artistic elements (e.g., new filters, trends,

etc.) are introduced daily. This trend causes substantial variations between the train and the test sets, which results in diminished effectiveness of the learned prior. Moreover, frequent fine-tuning or frequent training to improve the efficacy of the prior significantly hamper the scalability of such models.

Furthermore, the existing video enhancement approaches assume clean and sharp input frames, which may not always be the case due to factors like bad lighting conditions during the scene capture or video compression for internet transmission. These factors introduce noise in the clean video signal, and feeding such noisy input frames to video enhancement approaches results in suboptimal enhancements.

To address these challenges, we propose a novel Video Dynamics Prior (VDP) that leverages the internal statistics of a query video sequence to perform video-specific enhancement tasks. By utilizing a video’s internal statistics, our approach eliminates the need for curated training data. Thus, removing the constraints with regard to encountering test time variations. Additionally, we introduce a spatial pyramid loss that enhances robustness in the temporal domain, enabling our method to handle noisy input frames encountered at test time.

VDP utilizes two fundamental properties of a video sequence: (1) Spatio-temporal patch (ST patch) recurrence and (2) Spatio-temporal consistency. ST patch recurrence is a fundamental finding [125] that reveals significant recurrence of small ST patches (e.g., $5 \times 5 \times 3$ or $7 \times 7 \times 3$) throughout a natural video sequence. To leverage this strong prior, we employ an LSTM-Convolution-based architecture, which is detailed in Sec. 4.3. Moreover, to leverage the ST patch recurrence property across scales in a video, we introduced a novel spatial pyramid loss. In Sec. 4.3.5, we

demonstrate that our novel spatial pyramid loss helps the VDP model to be more robust to unstructured noise in both spatial and temporal domains.

Further, our approach considers the next frame prediction task to initialize our recurrent architecture. The task aims to auto-regressively predict the next frame, taking the current frame and previous frames information as context. We leveraged architecture from such a task as it promotes the spatio-temporal consistency in the processed video frames and models the complex temporal dynamics required for various downstream tasks. In our approach, we first formulate a next-frame prediction model with unknown parameters. Then depending on the downstream task, we maximize the observation likelihood with slight modification in the objective function.

This paper presents three key contributions to the field of video processing. Firstly, we introduce a novel inference time optimization technique that eliminates the need for training data collection to learn neural module weights for performing tasks. Secondly, we propose a novel spatial pyramid loss and demonstrate its effectiveness in providing robustness to spatial and temporal noise in videos. Lastly, we showcase the superior performance of our approach in multiple downstream video processing tasks, including video denoising, video frame interpolation, and video object removal. Our method outperforms current baselines and exhibits robustness in case noisy frames are provided as input for enhancement.

The rest of the paper is organized as follows. In Sec. 4.2, we briefly introduce the background work done in the area of video enhancement tasks. In Sec. 4.3, we discuss our methodology and key components of our architecture. Sec. 4-7, we briefly

describe the video processing tasks and accompanying modifications to the objective function for dealing with different tasks at hand. Finally, we conclude our work in Sec. 5.8.

4.2 Related Works

Over the past decades, many research efforts have been made to increase the efficacy of video enhancements. Traditional methods like [126, 127, 128] utilized affine-estimation based motion approach to perform video enhancement tasks. However, these methods were very limited in their capability of learning a good representation of videos to generalize well on various scenes. All of this changed with the recent developments in deep learning techniques that considerably improve the representation of videos and deliver high performance gains on video enhancement tasks.

Internal Learning. In the recent past, a considerable number of deep internal learning approaches were put forth to perform image enhancement tasks using just a single test image [20, 129, 130, 131, 132, 133, 134, 135]. Such approaches utilize patch recurrence property in a natural image as a strong prior to deal with inverse problems such as image denoising [20, 130, 136], super-resolution [129, 130, 134], and object removal [130]. However, when such internal learning approaches are directly applied to videos for enhancement tasks, it creates flickering artifacts in the processed video [137, 138]. These artifacts arise as image-based methods fail to maintain a video’s temporal consistency, a critical component of a video signal.

Extending the deep internal-learning based approach to videos remains a challenging problem. There are very few works proposed in the area that deal with videos, and all of these works are tailored towards a specific video enhancement task [12, 139]. In this paper, we extend a deep internal-learning based framework to the video space that effectively explores various video enhancement tasks such as denoising, super-resolution, frame interpolation and inpainting, utilizing a single structure of our VDP model.

Flow Guided Methods. In the image processing tasks, most of the methods utilize the 2D CNN modules. However, if we use such models in standalone settings, we often find that they are inept at processing the additional temporal dimension present in a video sequence. In order to overcome the temporal consistency problem, some researchers have opted for an optical-flow based approach to capture the motion occurring in the videos [114, 115, 117, 140, 141, 142, 143]. Essentially, they model the flow of the video sequences by doing explicit motion estimation. Then they perform warping operations (motion compensation) to generate enhanced video frames. These optical-flow based approaches have found plenty of applications in tasks such as video denoising [114, 115], video super-resolution [10, 114, 117, 144], video object removal [13, 145], video frame interpolation [114, 123, 146, 147, 148]. However, accurate motion estimation using optical flow is inherently challenging and computationally expensive. Therefore, deep neural networks such as RAFT [149] and SPyNet [150] are employed to estimate optical flow. These networks rely on lots of training data to learn to predict the optical flow. This double reliance on data by both the video enhancement approach and optical flow prediction networks can

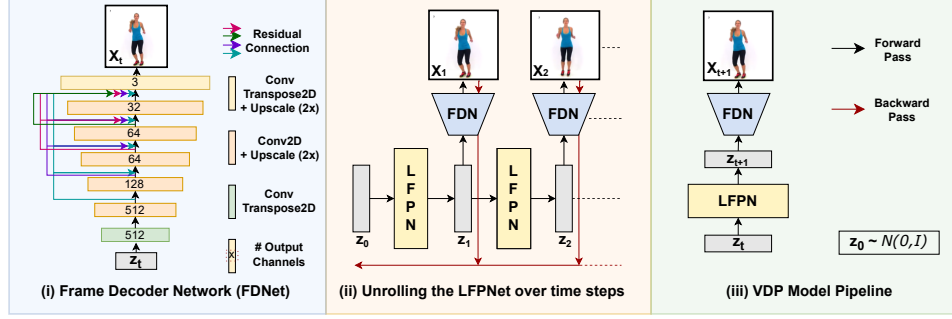


Figure 4.1: Model pipeline: We are given $X_1, \dots, X_t$ frames of a person exercising (depicted in the Fig). Here, we start by sampling $z_0 \sim \mathcal{N}(0, I)$ as an initial latent representation. Our model comprises two key modules: (i) the Frame Decoder Network (FDNet) and (ii) the Latent Frame Predictor Network (LFPNet), depicted in the figure. These modules, together form the pipeline of our VDP model, as shown in (iii). We optimize the weights of these neural modules on this specific input sequence by doing backpropagation over the loss pertaining to the task at hand.

introduce dataset biases. Further, we showcase that the flowbased models for video super-resolution and frame interpolation tasks are not robust to noise or artifacts arising from capturing the scene in low-lighting or bad lighting conditions. In our work, instead of explicitly performing motion estimation and motion compensation, we utilize LSTM-Conv decoder networks represented in Fig. 4.1 to capture the temporal information implicitly. Furthermore, our method is able to infer context directly from the test video sequence itself. Thus, mitigating the problems arising from dataset biases. Additionally, we empirically demonstrate that our approach is stable for tasks like super-resolution and frame interpolation, even in the presence of noisy input frames sequence.

4.3 Model Overview

Given a video sequence as context, we perform various video-specific processing tasks on this sequence of frames. In traditional settings, the training data is used to learn good representations that capture the patterns in the video sequences. However, in our case of no data, we rely on a proper choice of a deep neural network that can capture the ST patch recurrence property of a video as an informative prior for the video processing task at hand. Our model for performing all video-specific tasks is a combination of two modules. (1) The latent frame predictor network and (2) The frame decoder network as depicted in Fig. 4.1.(iii).

4.3.1 Latent Frame predictor Network (LFPNet)

The task of the latent frame predictor network is to maintain the spatio-temporal consistency of the video. It learns the dynamics of a video sequence in latent space, i.e., it models the temporal relations between the frames in latent space. We use LSTM cells as the building block for this module as they implicitly capture the temporal relation between the latent frames. This LFPNet module (is inspired by [46, 48]) $g(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$, has a fully-connected layer, followed by four LSTM layers with 1024 cells each, and a final output fully-connected layer. The output fully-connected layer takes the last hidden state from LSTM ($\mathbf{h}_{t+1}$) and outputs $\hat{\mathbf{z}}_{t+1}$ after $\tanh(\cdot)$ activation.

4.3.2 Frame Decoder Network (FDNet)

This network maps the frame embeddings from the latent space to the image space, i.e., $f(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^{c \times h \times w}$. The architecture for our VDP’s frame decoder network is depicted in the Fig. 4.1.(i). The residual skip connections in our architecture play a crucial role in the performance of our module FDNet by addressing the vanishing gradient problem and facilitating faster convergence.

4.3.3 Formulation of Video Dynamics Prior

For our proposed Video Dynamics Prior or VDP prior we only rely on the two modules namely LFPNet and FDNet. We start by sampling an initial latent noise vector $z_0 \sim \mathcal{N}(0, I)$ where $z_0 \in \mathbb{R}^D$. Here, D is the dimension of latent space. For our experiments, we keep the dimension D of latent space as 1024. Now, as illustrated in Fig. 4.1, we pass this z_0 embedding through the ‘LFPNet’ and obtain the next frame embedding z_1 . We then pass z_1 through the frame decoder network to obtain the next image frame $\hat{X}_1$. Mathematically, we can write the expression for the forward pass to obtain the future frame as $\hat{X}_{t+1} = f(g(z_t))$. Where, $f : \mathbb{R}^D \rightarrow \mathbb{R}^{c \times h \times w}$ denotes the frame decoder network (§4.3.2) that maps the latent space to image space i.e. $X_{t+1} = f(z_{t+1})$. Function $g(\cdot)$ denotes the latent frame predictor network. This network takes the latent embedding at the current step as the input and predicts the latent embedding at the next step as the output, i.e., $z_{t+1} = g(z_t)$. Leveraging future frame prediction task helps in initializing the LFPNet in the latent space.

4.3.4 Neural Module Weights Optimization

Since we optimize the weights of neural modules (LFPNet and FDNet) using a corrupted sequence, reconstruction loss alone is not enough to find a visually plausible enhancement of the input video. Hence, we introduce a few other regularization losses.

Reconstruction Loss: We use a combination of L1 and perceptual loss to formulate the reconstruction loss. It is given by Eqn. 5.2,

$$\mathcal{L}_{\text{rec}} = \|X_{t+1} - f(g(z_t))\| + \|\phi(X_{t+1}) - \phi(f(g(z_t)))\|. \quad (4.1)$$

Here, $\phi(\cdot)$ in the Eqn. 5.2 denotes the pre-trained VGG network on ImageNet [151]. $X_t, \hat{X}_t$ denotes the input and reconstructed video frames at timestep t, respectively.

Spatial Pyramid Loss: It has been well established that small ST patches of size $5 \times 5 \times 3$ recur within and across the scales for a video. Additionally, bicubic downscaling of a video results in coarsening of the edges in the downscaled video. This results in the downscaled video to have less motion speed and less motion blur. Both of these properties are extremely useful for video enhancement tasks. Hence, to utilize such lucrative properties of a video, we introduced spatial pyramid loss. For this loss, we scale down the input image to 3 levels ($\frac{h,w}{k^2}$ where k is 2,4,8) using a downsampler and calculate the reconstruction loss for each level. It is given by Eqn. 4.2,

$$\mathcal{L}_{\text{spl}} = \sum_{i=2,4,8} \|d_i(X_{t+1}) - d_i(f(g(z_t)))\|. \quad (4.2)$$

Here, X_{t+1} denotes the input video frames at timestep $t + 1$. $d_i(\cdot)$ in the Eqn. 4.2 denotes the downsampler network and i denotes scale of downsampler. For example, $d_2(\cdot) : \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R}^{c \times h/2 \times w/2}$.

Variation Loss: This is a variant of Total variational loss introduced in the paper [152]. Variation loss smoothens the frames while preserving the edge details in the frame. It is given by Eqn. 4.3,

$$\mathcal{L}_{\text{var}}(\hat{X}_t) = \sum_{i=1}^{H-1} \sum_{j=1}^{W-1} \|\hat{x}_{ij}^t - \hat{x}_{(i-1)j}^t\| + \|\hat{x}_{ij}^t - \hat{x}_{i(j-1)}^t\|. \quad (4.3)$$

Here, $\hat{x}_{ij}^t$ denotes the pixel of the processed video frames ($\hat{X}_t = f(g(z_{t-1}))$) at timestep t and location $\hat{X}_t[i, j]$.

Final Loss: We combine all the losses together into a final loss term as follows,

$$\mathcal{L}_{\text{final}} = \sum_{t=0}^{T-1} \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{spl}} \mathcal{L}_{\text{spl}} + \lambda_{\text{var}} \mathcal{L}_{\text{var}}. \quad (4.4)$$

4.3.5 Robustness of Video Dynamics Prior

In this section, we explore the internal learning properties of our model. For thorough exploration, we perform various experiments to explore the spatio-temporal patch recurrence [125] in natural videos using our model. In the past, authors of the

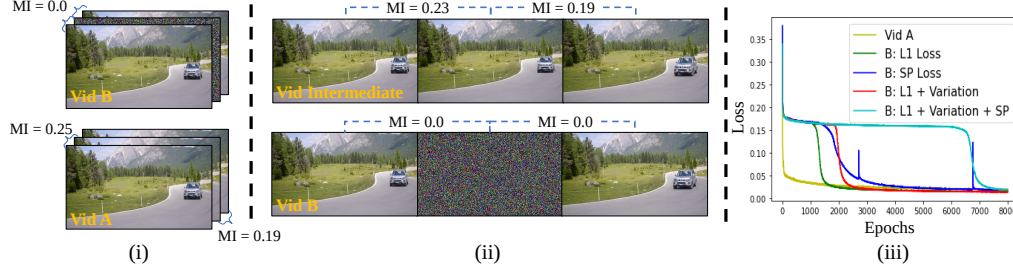


Figure 4.2: (i) Video B is generated by replacing the second frame of video A. (ii) Top row depicts the video obtained during the early round of optimization of the objective given by Eq. 4.4 for video B. Further, the average Mutual information [7] of video(intermediate) is much closer to video A. (iii) The loss plot depicts the convergence of our VDP model for in five different settings.

papers [20, 130] have explored the utilization of the patch recurrence property in images by the deep learning models to perform various enhancement tasks. Following these prior works, we extend such understanding to the non-trivial spatio-temporal patch (or ST-patch) recurrence phenomenon exhibited by our model.

We design a simple experiment where we have a natural video sequence A denoted as $V_a = \{X_1, X_2, X_3\}$ and depicted in Fig. 4.2.(i). We create another sequence V_b by replacing the frame X_2 with a random noisy frame X_{nf} resulting in $V_b = \{X_1, X_{nf}, X_3\}$. As shown in Fig. 4.2. (ii), the noisy frame is fairly independent of frames X_1 or X_3 hence, we get mutual information between these frames close to zero. This replacement of frame X_2 with X_{nf} results in higher entropy [153] of the video sequence V_b (larger diversity in temporal dimension $\rightarrow$ lower ST patch recurrence $\rightarrow$ lower self-similarity $\rightarrow$ higher entropy of sequence) in comparison to original sequence V_a (lower diversity in temporal dimension $\rightarrow$ higher ST patch recurrence $\rightarrow$ higher self-similarity $\rightarrow$ lower entropy of sequence). Hence when the VDP model is optimized over both the sequences V_a and V_b , it requires a longer duration for loss convergence as depicted by Fig. 4.2.(iii). During the optimization

over the video sequence V_b , our model obtains the sequence resembling V_a (given by $V_{\text{intermediate}}$ in Fig. 4.2.(ii)) in early rounds of optimization, near the epochs where we observe the first dip in the convergence curve and before ending up overfitting sequence V_b .

We conducted such experiments on our model under five different settings. 1) Using only L1 reconstruction loss (given by Eqn. 5.2) on video V_a . 2) Using only L1 reconstruction loss on video V_b . 3) Using only L1 reconstruction loss and spatial pyramid loss (given by Eqn. 4.2) on video V_b . 4) Using only L1 reconstruction loss and variation loss (given by Eqn. 4.3) on video V_b . 5) Using L1 reconstruction loss, spatial pyramid loss, and variation loss on video V_b .

The energy plot in Fig. 4.2(iii) represents the trend in MSE loss for the aforementioned five settings. It can be observed that in the absence of any noise in the video signal (yellow), the model fits the input quickly. The quick convergence can be attributed to higher ST patch recurrence in video sequence A. The VDP model optimized over the video sequence V_b using only the L1 reconstruction loss (green) provides some impedance to the temporal noise but quickly overfits the noise after a couple of iterations. Further, the VDP model optimized under setting 3 (red) or 4 (blue) provides higher impedance to noise than setting 2 but still converges after a few hundred epochs. However, when the VDP model is optimized using setting 5 (cyan), it provides considerably higher impedance to the noise in the temporal domain.

We observe a similar phenomenon when optimizing our model with a larger void filled with noisy frames. Additionally, when we optimize our VDP model over

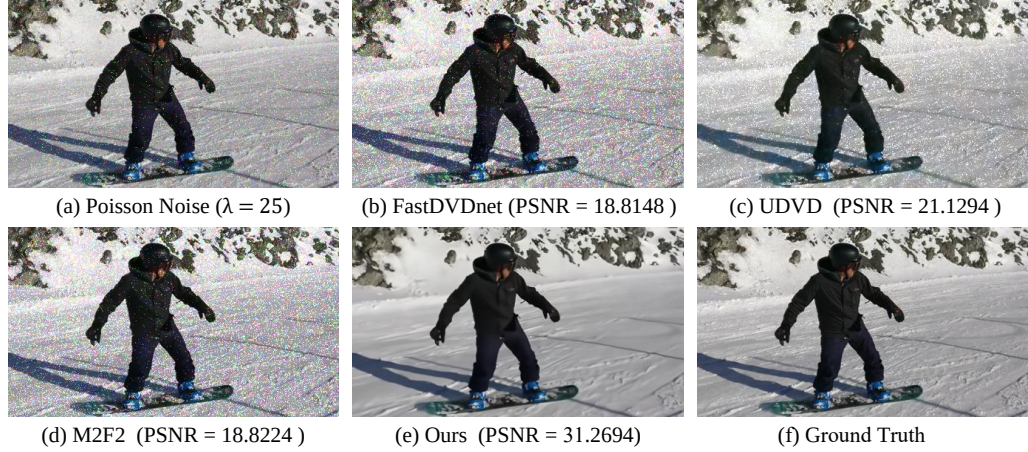


Figure 4.3: **Video Denoising:** Frame from a video in the DAVIS dataset denoised using different approaches. (a) Frame corrupted with additive poisson noise of intensity 25 (relative to intensity range $[0-255]$). Figure (b) depicts the denoised frames generated by FastDVDnet [3] method. Figure (c) depicts the denoised frame generated by UDVD [8] method. Figure (d) depicts the denoised frame generated by M2F2 [9] method. Figure (e) depicts the denoised frames generated by our method. Figure (f) shows the clean ground truth frames.

a noisy video (here, every frame in the video sequence consists of Gaussian noise), we do not see convergence at all. We attribute this behavior of our VDP model to promoting self-similarity of the st-patches over the output video sequence. Moreover, incorporating the spatial pyramid loss introduces frame matching at multiple levels of coarse scale. This encourages a more structured signal in the output and provides a higher impedance to the unstructured noise. We performed the same experiment over 250 different video sequences (randomly sampled from different datasets like Vimeo-90K, DAVIS, and UCF101) and came up with similar findings as given by Fig. 4.2.(iii).

Metric	Additive Noise Type	Noise Level	M2F2 [9]*	FastDVDnet [3]*	UDVD [8]*	Ours
PSNR	Gaussian	($\sigma = 20$)	34.89	35.77	35.12	35.74
PSNR	Gaussian	($\sigma = 30$)	33.21	34.04	33.92	34.07
PSNR	Poisson	($\lambda = 25$)	18.19	19.02	22.08	31.96
PSNR	Poisson	($\lambda = 30$)	17.12	18.03	20.98	30.07

Table 4.1: **Quantitative video denoising results:** To evaluate our method, we calculate the Avg PSNR score between the generated noise-free video and ground truth video. The σ denotes the standard deviation for the added Gaussian noise to the frames. While the λ denotes the intensity of additive Poisson noise. For all baselines and our method, the score is calculated on the DAVIS dataset [1]. Also, **blue (bold)** denotes the best score. Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.

Metrics	# Noisy Frames	Noise level	ToFlow [114]*	DAIN [123]*	SoftSplat [148]*	RIFE [147]*	Ours
PSNR	None	None	34.68	35.00	35.39	35.41	35.41
SSIM	None	None	0.9677	0.968	0.970	0.970	0.970
PSNR	2 nd Frame	$\sigma = 15$	20.30	20.87	18.25	20.22	34.92
SSIM	2 nd Frame	$\sigma = 15$	0.514	0.521	0.4012	0.512	0.918

Table 4.2: **Quantitative video frame interpolation results:** For the evaluation of our method we calculate the Avg PSNR and SSIM score between the generated interpolated video frames and ground truth video. For all baselines and our method, the score is calculated on the UCF101 dataset [2]. The color **blue (bold)** denotes the best performance. Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.

4.4 Video Denoising

In this section, we present our VDP model utilized for the video denoising task. A noisy video can be defined as the combination of a clean video signal and additive noise. As shown in the previous section, our VDP model exhibits higher impedance to the noise signal and lower impedance to the clean video signal. Exploiting this property, we propose a denoiser for video sequences. Unlike traditional approaches, our denoiser does not require data collection, as we demonstrate that the final loss function (Eqn. 4.4) alone is sufficient for effective video denoising.

We optimize our model parameters on the noisy test-time video sequence using

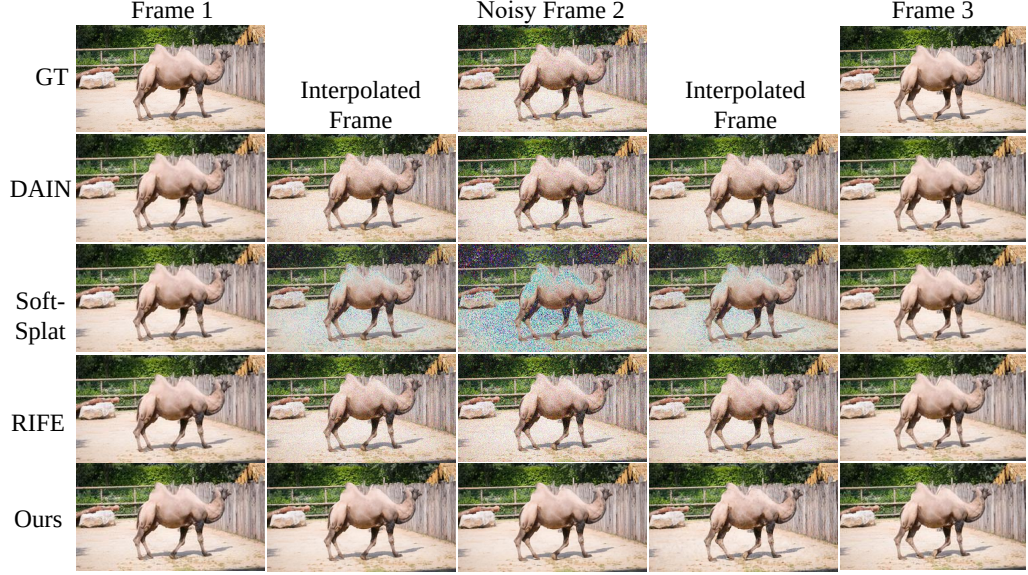


Figure 4.4: **Video Frame Interpolation:** In order to compare our method to existing baseline methods, we conducted visual comparisons using three frames for each method. We introduced Gaussian noise ($\sigma = 15$) in the second frame and tasked all methods with interpolating frames between them. It can be observed that our method not only denoises the second noisy input frame but also generates denoised intermediate frames. This is in contrast to other baseline methods that interpolate noise along with the clean signal.

the loss function given by Eqn. 4.4. We then use these optimized LFPNet & FDNet modules for generating high-quality clean video frames. The new enhanced video frames for each timestamp can be obtained by $\hat{X}_{t+1} = f(g(z_t))$.

The presence of noise in a video can be attributed to various factors, such as non-ideal video capture settings or lossy video streaming over the internet. While curating training datasets for baselines, it is common to assume that the additive noise follows a Gaussian distribution. However, at test time, when this additive noise is drawn from a Poisson distribution, the performance of baseline models deteriorates significantly. In contrast, our VDP model, optimized on the test sequence by minimizing the loss given in Eqn. 4.4, achieves highly effective denoising results. Fig. 4.3 provides a qualitative comparison between our method and the baselines,

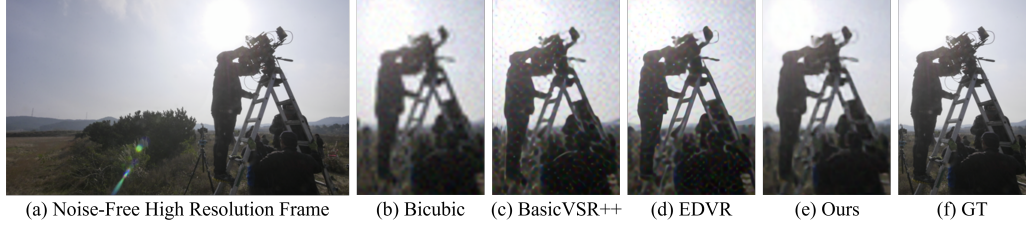


Figure 4.5: **Video Super Resolution:** Comparison of results of the ‘Cameraman’ sequence from VIMEO-90K-T dataset. Gaussian noise of std $\sigma = 5$ is added to the lower-resolution input frames. (a) High-resolution noise-free ground truth frame. (b) Patch from higher resolution frame generated by BiCubic method. (c) Patch from higher resolution frame generated by BasicVSR++ [10]. (d) Patch from higher resolution frame generated by EDVR [11] method. (e) Patch from the higher resolution by Ours VDP method. (f) Patch from higher resolution ground truth frame. Note the noise suppression and details of the camera in the patch generated by our method compared to baselines.

Metrics	Noisy Frames	Noisy Intensity	BiCubic	ToFlow [114]*	EDVR [11]*	BasicVSR++ [10]*	Ours
PSNR		-	31.12	33.20	35.79	35.95	35.70
SSIM		-	0.870	0.920	0.937	0.940	0.936
PSNR		$\sigma = 5$	29.90	30.20	32.02	31.58	33.87
SSIM		$\sigma = 5$	0.796	0.712	0.758	0.720	0.878

Table 4.3: **Quantitative video super-resolution results:** For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the generated super-resolution video and ground truth video for all baselines along with our method. The score is calculated on the standard VIMEO-90K-T dataset. Best results in the table are shown in **blue (bold)**. Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.

clearly demonstrating the superior performance of our approach.

To further evaluate the effectiveness of our method, we conduct experiments on the DAVIS dataset using different noise settings. We compare our model against state-of-the-art baselines and present the quantitative evaluation in Table. 4.1. The results highlight the superiority of our method in video denoising tasks, showcasing its robustness in handling diverse noise distributions.

4.5 Video Frame Interpolation

Frame interpolation is a challenging task that involves generating intermediate frames between the given frames in a video sequence. The primary objective is to synthesize high-quality and realistic frames that accurately capture the spatio-temporal coherence of the original video. However, existing baseline methods for frame interpolation assume that the input video consists of high-quality frames without any noise. In real-world settings, video sequences often contain noise due to non-ideal capturing conditions, which poses challenges and diminishes the performance of these baseline methods. Such limitations are addressed by our VDP model.

For this task, we utilize the latent embeddings to perform the frame interpolation. First, we optimize our VDP model parameters using the loss function given by Eqn. 4.4. Now, to generate intermediate frames, we linearly interpolate the latent embeddings of the video using the equation.

$$X_{\text{intermediate}} = f(\alpha z_{t+1} + (1 - \alpha)z_t) \quad \forall \alpha \in (0, 1). \quad (4.5)$$

Here, $X_{\text{intermediate}}$ is the interpolated frame between the frames X_t and X_{t+1} . For example, to perform a 4x frame interpolation, we select $\alpha = [0.25, 0.5, 0.75]$ in the Eqn. 4.5. To evaluate the performance of our method, we conduct experiments on the UCF101 (triplet) dataset [2, 123] under two conditions: 1) ideal triplet input sequence, 2) triplet input sequence containing one noisy frame. We make a qualitative

comparison between our method and the baselines depicted in Fig. 4.4. It can be observed from Fig. 4.4 that our method is more robust to noisy input and outputs the crisp intermediate frames without noise, unlike baseline methods DAIN [123], SoftSplat [148] and RIFE [147].

Table. 4.6 represents the quantitative evaluation of our approach on the UCF-101 dataset. It can be observed from the table that our method outperforms the existing baselines on PSNR and SSIM metrics in the video frame interpolation task, further validating the effectiveness and suitability of our approach for this task.

4.6 Video Super Resolution

Given our ever-increasing screen resolutions, there is a great demand for higher-resolution video content. In this task, we are given a lower-resolution video sequence as input. We are tasked to change this input to a higher-resolution video sequence.

For this task, we modify the loss objective given by Eqn. 4.4 as follows,

$$f, g \sum_{t=0}^{T-1} \lambda_{\text{rec}} \mathcal{L}_{\text{rec}}(X_{t+1}, d_{\text{sc}}(f(g(z_t)))) + \lambda_{\text{spl}} \mathcal{L}_{\text{spl}}(X_{t+1}, d_{\text{sc}}(f(g(z_t)))) + \lambda_{\text{var}} \mathcal{L}_{\text{var}}(f(g(z_t))). \quad (4.6)$$

The modified objective for the video super-resolution task is given by Eqn. 4.6. Here, function $d_{\text{sc}}(\cdot)$ denotes the downsampling function. For example, to perform 4x super-resolution, we would define downsampling function as $d_{\text{sc}} : \mathbb{R}^{c \times h \times w} \rightarrow \mathbb{R}^{c \times (h/4) \times (w/4)}$.

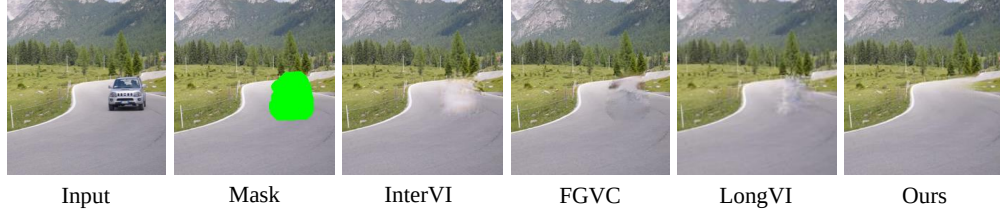


Figure 4.6: **Video Object Removal.** In the above figure, we describe a video object removal task. The input in the figure represents the original video sequence frame without any alterations. The InterVI [12], FGVC [13], and LongVI [14] in the figure represent the frames generated by removing the object “car” from the video sequence. The rightmost frame depicts the removal of the object from the video frame by our method. It can be observed that our method do not create hallucinations.

We evaluate our method for the video super-resolution task on the VIMEO90-K-T dataset and contrast our model against the state-of-the-art baseline methods. We evaluate our method on two different settings: 1) When sharp lower-resolution frames are provided, 2) When noisy lower-resolution frames are provided. We perform a $4\times$ super resolution of frames in the video sequences. We make a qualitative comparison between our method and the baselines for the setting when noisy frames are given as input frames, depicted in Fig. 4.7. It can be observed from Fig. 4.7 that our method outputs better and noise-free higher resolution frames with more details in comparison to the baseline methods. Table. 4.5 represents the quantitative evaluation of our approach. It can be observed from the table that there is a significant drop in the performance of baselines to our method when a slight amount of Gaussian noise is added to the input frames. We demonstrate empirically that our method is much more robust to noise as compared to other baseline methods.

Metrics	DIP [130]	DFGVI [154]*	FGVC [13]*	InterVI [12]	LongVI [14]	Ours
PSNR	26.03	27.57	31.92	30.86	31.80	32.06
SSIM	0.921	0.9289	0.9499	0.9350	0.9457	0.9512

Table 4.4: **Quantitative Object Removal results:** We compare the average PSNR and SSIM scores of our method with the existing baselines on the DAVIS dataset. The best performance is denoted by the score in **blue (bold)**. Please note we use pretrained models for baselines provided by the authors. The baselines marked with * denote a data-driven baseline.

4.7 Video Object Removal

Unwanted items in a video can grab the viewers’ attention from the main content. In this task, we remove such objects from the original video stream. For this task, we assume a mask $M = \{m_t\}_1^T \forall m_t \in \{0, 1\}^{c \times w \times h}$ is provided for each frame to remove an object from the video sequence. To perform the removal of an object, we modify the objective function given by Eqn. 4.4 as follows,

$$\sum_{f,g} \lambda_{\text{rec}} \mathcal{L}_{\text{rec}}(m_t X_t, m_t f(g(z_{t-1}))) + \lambda_{\text{spl}} \mathcal{L}_{\text{spl}}(m_t X_t, m_t f(g(z_{t-1}))) + \lambda_{\text{var}} \mathcal{L}_{\text{var}}(f(g(z_{t-1}))). \quad (4.7)$$

We evaluate our method for the video object removal task on the DAVIS dataset. A qualitative comparison between our method and the baselines can be seen in Fig. 4.6. It can be observed from Fig. 4.6 that our method performs better than the existing methods in removing the object from the video without creating any artifacts in the processed videos.

Table. 4.4 represents the quantitative evaluation of our approach. We perform the evaluation based on the stationary mask setting. In this setting, a stationary

mask is applied to all frames of the video sequence. This setting helps us evaluate the model’s ability to generate temporally coherent frames even if some information in the frames is consistently missing.

4.8 Limitations

One severe limitation of our approach is that it is an offline approach, i.e., it can not be used in real-time to perform video enhancements. One way to rectify this limitation is to work on reducing the parameters for the frame decoder network. One suggestion is modifying the residual skip connection from concatenation-based connections to additive connections. This would drastically reduce the number of parameters to be trained with minimal effects on the efficacy of the model.

The second limitation of our approach is that it is a recurrent approach and can not be scaled using multiple GPUs. This issue can be alleviated using a temporal attention-based approach that can have long-term memory and is scalable across multiple GPUs.

Our approach, in its current stage, requires no data and is only able to perform low-level vision tasks. However, to perform high-level vision tasks such as performing video segmentation or video manipulation, we conjecture that external knowledge would be required, like a collection of external datasets or a pretrained model on a such external dataset to aid our VDP method.

4.9 Conclusion

We conclude in this work that a video-specific approach is a way to go for a considerable number of video processing tasks. Our VDP method not only performed better than the task-specific approaches but also it is not hamstrung by the limited availability of either the computing power or the training data. We demonstrate that the introduction of a spatial pyramid loss enhances the robustness of our proposed VDP method. Leveraging this loss, we were able to achieve state-of-the-art results in all enhancement tasks, even in the presence of noise in the input frames. One limitation of our method is the long processing time compared to other methods at the test time inference. Nevertheless, it is important to note that we do not require training as the other baseline methods (requires multiple GPU hrs if not days). In the future, we will focus on improving efficiency to shorten the processing time for increased practical utility. One great advantage of our approach is removing the data collection requirement for such processing tasks. We hope that this step towards a data-free approach will reduce the massive data mining across the internet and provide better data privacy measures.

4.10 Broader Impact

Our approach offers a significant advantage by eliminating the need for data collection in processing tasks. This shift towards a data-free approach has the potential to mitigate the extensive data mining activities conducted across the

internet, thereby addressing concerns related to data privacy. By reducing the reliance on data collection, our approach promotes more ethical and responsible use of technology.

However, it is crucial to acknowledge that our approach also has some limitations. We believe that addressing these challenges through ongoing research and development will contribute to the continuous improvement and refinement of data-free approaches, thereby enhancing their broader impact.

4.11 Appendix

4.11.1 Comparison with cascaded models

Existing video enhancement approaches rely on clean and sharp input frames to perform the task at hand. However, this is not always the case depending on the lighting condition of scene capture or compressing a video stream to send it over the internet. All of these introduce noise in the clean video signal. Applying video enhancement approaches in such conditions results in sub-optimal enhancements, as demonstrated in the main paper. One other way of dealing with such conditions is the use of a two-stage cascaded approach. For example, we have to provide super-resolute frames for a noisy video sequence. We can get the solution by first denoising the frames and then processing these with super-resolution approaches. However, this also results in error propagation from one stage to another and results in artifacts.

To evaluate the aforementioned reasoning, we perform a two-stage cascaded approach for video super-resolution and frame interpolation tasks. In this cascaded

approach, we apply the frame denoiser first and then apply the VSR or VFI model. For video denoiser we use the FastDVDnet [3]. Table. 4.5 and 4.6 describe the quantitative evaluation of baselines(cascaded approach) with our VDP model for super-resolution and frame interpolation tasks. It can be observed that there is a deterioration in results for the super-resolution task for the cascaded approach than directly applying the baseline. This is primarily due to the fact that when the denoiser is applied in the first stage, it washes away important details from the lower-resolution frames. When these frames are passed through the video super-resolution baselines, the error from the first stage is propagated further. A qualitative visualization of this is depicted in Fig. 4.7. However, for the frame interpolation task, the case is reversed, and cascaded baseline approaches yield better results than only baseline approaches. It can be seen from Table. 4.6 that the cascaded baselines perform better than only baseline approaches but still perform worse than our VDP approach.

Also, we have shown in the main paper that denoisers like FastDVDnet [3] are also prone to variations in noise distribution. This would hurt the efficacy of the cascaded models if the noise is coming from a distribution other than the Gaussian distribution.

4.11.2 Ablation study

We performed ablation experiments to study the effect of losses for our video dynamics prior. We report the quantitative results for all the enhancement tasks in

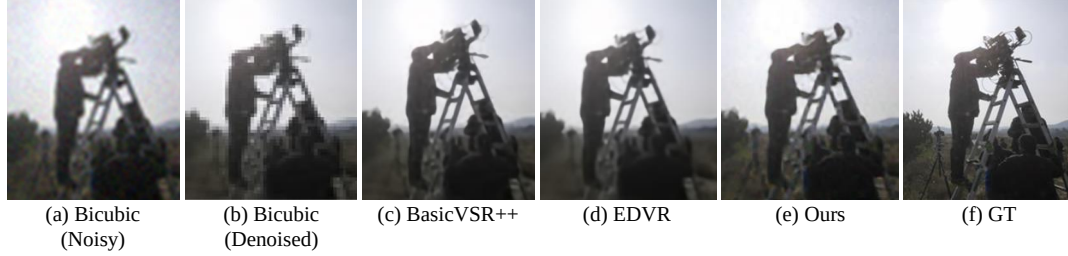


Figure 4.7: **Video Super Resolution:** Comparison of results of the ‘Cameraman’ sequence from VIMEO-90K-T dataset. Gaussian noise of std $\sigma = 5$ is added to the lower-resolution input frames. (a) Bicubic extrapolation of noisy low-resolution frame (b) Bicubic extrapolation of denoised low-resolution frame. (c) Patch from higher resolution generated by BasicVSR++ [10] over the denoised low-resolution frame. (d) Patch from higher resolution frame generated by EDVR [11] over the denoised low-resolution frame. (e) Patch from the higher resolution by Ours VDP method. (f) Patch from higher resolution ground truth frame. We use standard video denoiser FastDVDNet [3] to denoise the input low-resolution video frames. Based on the figure, it appears that the denoiser used in the first stage of the model removes vital information from the lower-resolution frames, which leads to errors in the second stage. Our VDP model, on the other hand, preserves more details compared to the baselines.

Metrics	Noisy Frames	Noisy Intensity	BiCubic	EDVR [11]*	BasicVSR++ [10]*	Ours
PSNR		$\sigma = 5$	27.78	31.56	31.03	33.87
SSIM		$\sigma = 5$	0.806	0.857	0.853	0.878

Table 4.5: **Quantitative cascaded video super-resolution(4×) results:** For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the generated super-resolution video and ground truth video for all baselines along with our method. The score is calculated on the standard VIMEO-90K-T dataset. Best results in the table are shown in blue (**bold**). Please note we use VSR baselines by cascading it with the video denoiser model FastDVDnet [3]. The baselines marked with * denote a data-driven baseline.

Metrics	# Noisy Frames	Noise level	ToFlow [114]*	DAIN [123]*	SoftSplat [148]*	RIFE [147]*	Ours
PSNR	2 nd Frame	$\sigma = 15$	33.31	33.51	34.20	33.95	34.92
SSIM	2 nd Frame	$\sigma = 15$	0.882	0.891	0.898	0.892	0.918

Table 4.6: **Quantitative cascaded video frame interpolation results:** For the evaluation of our method, we calculate the Avg PSNR and SSIM score between the generated interpolated video frames and ground truth video. For all baselines and our method, the score is calculated on the UCF101 triplet dataset. Please note we use VFI baselines by cascading it with the video denoiser model FastDVDnet [3]. The color blue (**bold**) denotes the best performance. The baselines marked with * denote a data-driven baseline.

Table. 4.7. It can be observed from the denoising section of the table that spatial pyramid loss $\mathcal{L}_{\text{spl}}$ (given by Eqn. 2) is very important to provide robustness to both the spatial and temporal domains. Additionally, it can be observed that variation loss $\mathcal{L}_{\text{var}}$ is very important for tasks like super-resolution, frame interpolation, and object removal. In Fig. 4.8-4.11, we show a qualitative comparison between VDP model optimized with different subsets of losses.

Video denoising: It can be observed from Fig. 4.8 that only using the reconstruction loss $\mathcal{L}_{\text{rec}}$ results in very suboptimal denoising. Optimizing for a high number of epochs using only $\mathcal{L}_{\text{rec}}$ results in the total reconstruction of the noisy signal. Adding the variation loss $\mathcal{L}_{\text{var}}$ to $\mathcal{L}_{\text{rec}}$ results in smoothening of the video frame as can be observed from the Fig. 4.8. However, adding the spatial pyramid loss $\mathcal{L}_{\text{spl}}$ to $\mathcal{L}_{\text{rec}}$ results in a very strong denoiser. This can also be observed in the quantitative evaluation given by Table. 4.7.

Video Super Resolution: It can be observed from Fig. 4.9 that only using the reconstruction loss $\mathcal{L}_{\text{rec}}$ results in suboptimal quality of super resolute frames with edginess in the texture. The addition of the spatial pyramid loss $\mathcal{L}_{\text{spl}}$ to $\mathcal{L}_{\text{rec}}$ results in better texture in the high-resolution frames, yet some edginess in the texture remains. However, the addition of the variation loss $\mathcal{L}_{\text{var}}$ to $\mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{rec}}$ results in smoothening of the texture while retaining the edges. This can also be observed in the quantitative evaluation given by Table. 4.7.

Video Frame Interpolation: It can be observed from Fig. 4.10 that only using the reconstruction loss $\mathcal{L}_{\text{rec}}$ results in suboptimal quality of interpolated frames with rough textures and missing details. The addition of the spatial pyramid loss $\mathcal{L}_{\text{spl}}$ to

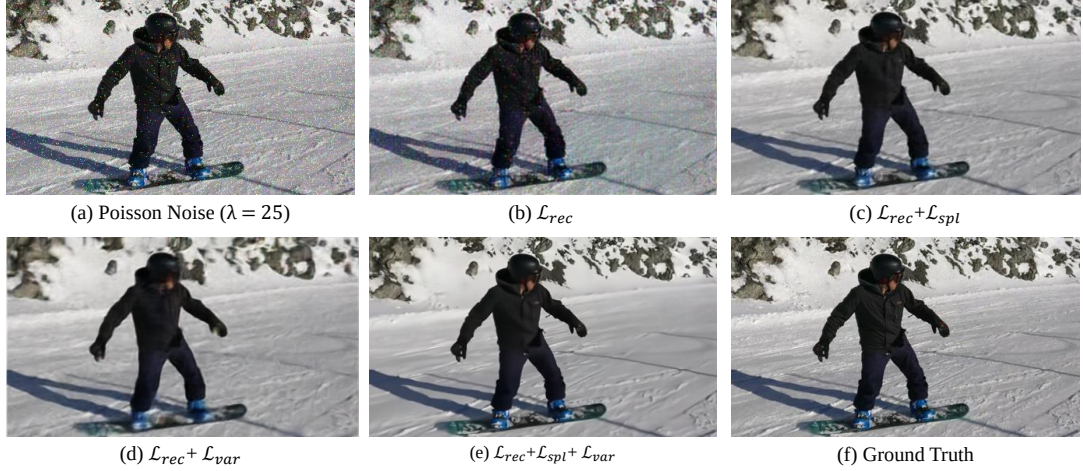


Figure 4.8: **Video Denoising (Ablation):** Frame from a video in the DAVIS dataset denoised using loss functions. (a) Frame corrupted with additive poisson noise of intensity 25 (relative to intensity range [0-255]). Figure (b) depicts the denoised frames generated by only $\mathcal{L}_{\text{rec}}$. Figure (c) depicts the denoised frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$. Figure (d) depicts the denoised frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$. Figure (e) depicts the denoised frames generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{var}}$. Figure (f) shows the clean ground truth frames.

$\mathcal{L}_{\text{rec}}$ results in better texture and increased details in the interpolated frames however, some edginess in the texture remains. However, the addition of the variation loss $\mathcal{L}_{\text{var}}$ to $\mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{rec}}$ results in smoothening of the texture while retaining the details. This can also be observed in the quantitative evaluation given by Table. 4.7.

Video Object Removal: It can be observed from Fig. 4.11 that only using the reconstruction loss $\mathcal{L}_{\text{rec}}$ results in suboptimal quality of interpolated region in the frames with rough textures and color jitter artifacts. The addition of the spatial pyramid loss $\mathcal{L}_{\text{spl}}$ to $\mathcal{L}_{\text{rec}}$ results in better texture and reduced jittering artifacts in the interpolated region in the frames. However, the addition of the variation loss $\mathcal{L}_{\text{var}}$ to $\mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{rec}}$ results in smoothening of the texture and merging of the inpainted region with the background. This can also be observed in the quantitative evaluation given by Table. 4.7.

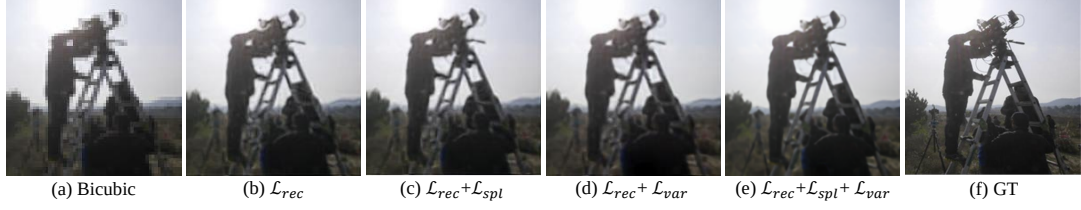


Figure 4.9: **Video Super Resolution (Ablation)**: Comparison of results of the ‘Cameraman’ sequence from VIMEO-90K-T dataset. (a) Bicubic extrapolation of low-resolution frame. (b) Patch from higher resolution generated by utilizing only the reconstruction loss given by Eqn. 1. (c) Patch from higher resolution frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$ (d) Patch from the higher resolution frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$. (e) Patch from the higher resolution frame generated by $\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}} + \mathcal{L}_{\text{var}}$ (f) Patch from higher resolution ground truth frame.

4.11.3 Processing time

We would like to point out here that if we take cumulative training and inference time, then our baseline would come out to be a better bargain in many scenarios, as depicted by Table. 4.8. To put it in perspective, a data-heavy baseline requires an ample amount of time to train, even when provided with perfect hyperparameters, making exploring different architectures quite expensive. In addition, the training dataset is limited in terms of representing real-world scenarios; this would further increase the spending time and effort in fine-tuning the network every time for a new setting. Please note that the inference time for our method has been calibrated on

Task	Dataset	Metric	Additive Noise Type	Noise Level	$\mathcal{L}_{\text{rec}}$	$\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}}$	$\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{spl}}$	$\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{var}} + \mathcal{L}_{\text{spl}}$
Denoising	DAVIS	PSNR	Poisson	$(\lambda = 25)$	20.58	24.63	30.14	31.96
		PSNR	Poisson	$(\lambda = 30)$	20.11	23.82	29.81	30.07
Super-Resolution	Vimeo-90KT	PSNR			31.23	33.86	32.18	35.70
		SSIM			0.890	0.924	0.915	0.936
Frame-interpolation	UCF-101	PSNR			33.29	34.13	33.88	35.41
		SSIM			0.943	0.945	0.963	0.970
Object Removal	DAVIS	PSNR			29.31	31.78	30.12	32.06
		SSIM			0.9125	0.9451	0.9352	0.9512

Table 4.7: **Quantitative ablations results**: To evaluate our method, we calculate the Avg PSNR score between the generated noise-free video and ground truth video. **blue (bold)** denotes the best score.

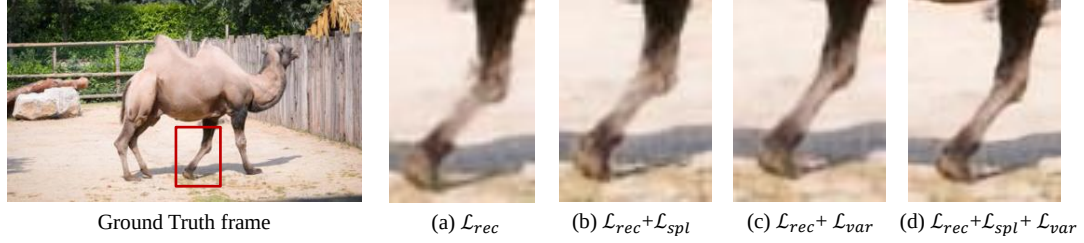


Figure 4.10: **Video Frame Interpolation (Ablation)**: In order to compare our method with different loss functions, we conducted a visual comparison. (a) Patch generated by utilizing only the reconstruction loss given by Eqn. 1. (b) Patch generated by $\mathcal{L}_{rec} + \mathcal{L}_{spl}$ (c) Patch generated by $\mathcal{L}_{rec} + \mathcal{L}_{var}$. (d) Patch generated by $\mathcal{L}_{rec} + \mathcal{L}_{spl} + \mathcal{L}_{var}$.

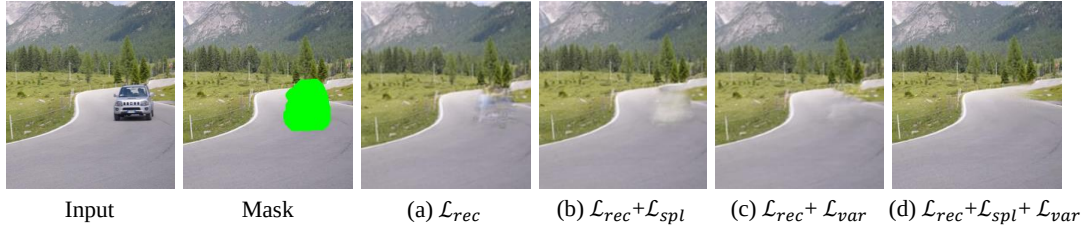


Figure 4.11: **Video Object Removal (Ablation)**. In the above figure, we describe a video object removal task. The GT in the figure represents the original video sequence frame without any alterations. (a) Patch generated by utilizing only the reconstruction loss given by Eqn. 1. (b) Patch generated utilizing the loss given by $\mathcal{L}_{rec} + \mathcal{L}_{spl}$ (c) Patch generated by utilizing the loss given by $\mathcal{L}_{rec} + \mathcal{L}_{var}$. (d) Patch generated by utilizing the loss given by $\mathcal{L}_{rec} + \mathcal{L}_{spl} + \mathcal{L}_{var}$.

one Nvidia A6000 GPU.

4.11.4 Optimization Details

To optimize our model, we use a single Nvidia A6000 GPU with 48G memory to process a single video at a time with 15-frame sequences of size 448x256. We optimize the modules LFPNet and FDNet weights using the entire test sequence with the *Adam optimizer* at a learning rate in the range of [0.0002, 0.002].

Video Denoising: For getting a good performance on the denoising task, we utilize the following weights for the different losses; $\lambda_{rec} = 1$, $\lambda_{spl} = 0.0001$ and $\lambda_{var} = 0.0001$.

Task	Dataset size	Resolution	Ours (Train + Infer)	Best Baseline (Train + Infer)
Super-Resolution	38,990 (7 fr/v)	$112 \times 64 \rightarrow 448 \times 256$	0 + 15s	300 hrs + 1s
Denoise	50 seq (3450 fr)	448×256	0 + 12s	200 hrs + 0.1s
Object Removal	50 seq (3450 fr)	448×256	0 + 8.5s	160 hrs + 8.3s
Frame Interpolation	3,782 triplets	448×256	0 + 10s	120hrs + 0.08s

Table 4.8: Training and inference time comparison with the best baselines. Note that the inference time in the table is given as the time required to process per frame of a video.

We observed that the performance of our VDP denoiser plateaued after 3600 epochs.

Video Frame Interpolation: For getting a good performance on the frame interpolation task, we utilize the following weights for the different losses; $\lambda_{\text{rec}} = 1$, $\lambda_{\text{spl}} = 0.0001$ and $\lambda_{\text{var}} = 0.0001$. We observed that the performance of our VDP frame interpolation model plateaued after 1800 epochs.

Video Super-Resolution : For getting a good performance on the VSR task, we utilize the following weights for the different losses; $\lambda_{\text{rec}} = 1$, $\lambda_{\text{spl}} = 0.01$ and $\lambda_{\text{var}} = 0.0001$. We observed that the performance of our VDP SR model plateaued after 4200 epochs.

Video Object removal: For getting a good performance on the object removal task, we utilize the following weights for the different losses; $\lambda_{\text{rec}} = 1$, $\lambda_{\text{spl}} = 0.01$ and $\lambda_{\text{var}} = 0.0001$. We observed that the performance of our VDP object removal model plateaued after 1800 epochs.

Additionally, we conducted our experiments using multiple random initializations of FDNet and LFPNet modules and observed consistent quantitative results across all runs.

4.11.5 Additional Details - FNet

Each convolution layer is followed by batch normalization and a leaky rectified linear unit (LeakyReLU; negative slope = 0.2), except for the last layer of the decoder. In the last layer of the decoder, we use the sigmoid activation function. We utilize Pytorch for our implementation.

4.11.6 Dataset Descriptions

Vimeo-90K-T [114] This dataset contains 7824 short clips downloaded from 'vimeo.com'. Each clip only contains 7 frames per clip. We used this dataset for a video super-resolution task. The lower resolution input frames have a size of $3 \times 112 \times 64$ while the higher resolution frames have a size of $3 \times 448 \times 256$.

DAVIS [1] This dataset contains 50 curated video clips, and each clip contains a unique object in the clip. The dataset has masking annotation present for this object for each clip. Each frame in each clip contains 1 mask annotation. There is 3450 total number of frames in this dataset. We used this dataset for video denoising and object removal task. We resized the resolution of each clip for the aforementioned tasks. The resolution of input frames for the task was of the size $3 \times 448 \times 256$.

UCF [2] The UCF101 dataset contains videos with a large variety of human actions. We utilized this dataset for the video frame interpolation task. We used this dataset under standard settings put forth by paper [123]. There are 379 triplets with a resolution of 256×256 pixels.

Task	Method	Dataset	Noisy Frames	Noise Intensity	FVD↓	LPIPS↓	PSNR↑	SSIM↑
Denoise (Poisson)	FastDVDnet	DAVIS		$\lambda = 25$	1743	0.65	19.02	-
	Ours				54	0.0056	31.96	-
Object Removal	FGVC	DAVIS		-	580	0.0556	31.92	0.9499
	Ours				445	0.0408	32.06	0.9512
Inter- -polation	RIFE	UCF101	2 nd Frame	$\sigma = 15$	-	0.412	20.22	0.512
	SoftSplat- $\mathcal{L}_F$				-	0.624	18.25	0.401
	Ours				-	0.102	34.92	0.918
VSR	EDVR	Vimeo90KT		$\sigma = 5$	-	0.5051	32.02	0.758
	BasicVSR++				-	0.5263	31.58	0.720
	Ours					0.181	33.87	0.878

Table 4.9: We include LPIPS and FVD [4] metrics for completeness of quantitative evaluation of enhancement tasks (video denoising, object removal, and video interpolation). FVD is a deep neural network based metric that evaluates both the spatial and temporal quality of the processed video with the ground truth video. FVD requires minimum of 16 frames in a sequence for evaluation. Hence, we were unable to evaluate FVD for task like VFI and VSR on UCF101 triplet and Vimeo90KT datasets.

Chapter 5: Video Decomposition Prior: Editing Videos Layer by Layer

5.1 Introduction

Deep learning techniques, prominently convolutional networks [155] and vision transformers [156], have emerged as the preferred choice for video-related tasks due to their superior results. Typically, these frameworks are trained using a standardized method: curate a dataset of input/corrupted and ground truth/clean sequences, design the model, and minimize the L2 loss between predictions and actual values. While this supervised approach sets state-of-the-art benchmarks, it grapples with two significant challenges. First, collecting extensive datasets with annotated pairs for every task is impractical and expensive. Second, as demonstrated by prior works like [157], the performance of pretrained models deteriorates when the test data slightly deviates from the training data.

To circumvent these challenges, emerging video editing methods [157, 158, 159, 160, 161] were developed and they offer three key advantages: they operate directly on test sequences (also known as inference-time optimization), thereby obviating the need for collecting input and ground truth pairs; they are immune to dataset biases and generalize well to unseen examples [157]; and they are resource-efficient as they do not have to optimize over an external training corpus instead they optimize only

on the given test input video. However, these techniques are not without limitations. They are often designed for very specific tasks, and they often perform inferior to their data-trained counterpart.

In this paper, we propose a inference-time optimization framework **VDP**, that does not rely on a collection of task-specific video input and ground truth sequence pairs. We took inspiration from professional video editing techniques – where sequences are broken down into multiple RGB layers. Each RGB layer is associated with a distinct opacity layer, and by manipulating these RGB layers and their associated opacity levels, desired outcomes are achieved. In this work, we emphasize that proper decomposition formulation of a video leads to visually realistic solutions for the tasks. Fig. 5.1 illustrates our approach for the 3 different tasks discussed. Our framework relies only on the input video and its derived optical flow to solve a wide variety of video-specific tasks.

From a technical standpoint, **VDP** consists of two integral modules: ‘RGB-Net’ that operates on the appearance aspect of the input video and ‘ α -Net’ that operates on the associated optical flow RGB of the input video. Both of these modules are designed using the convolutional U-Net architecture. Based on the tasks, we use the appropriate video decomposition formulation and appropriate regularization terms. We then jointly optimize these convolutional U-Nets for the same.

Our work makes three main contributions. 1) We propose a novel inference-time optimization framework that decomposes a video into multiple layers and can effectively perform tasks like video object segmentation, dehazing, and relighting. 2) We introduce a new formulation of the logarithmic decomposition of a video to

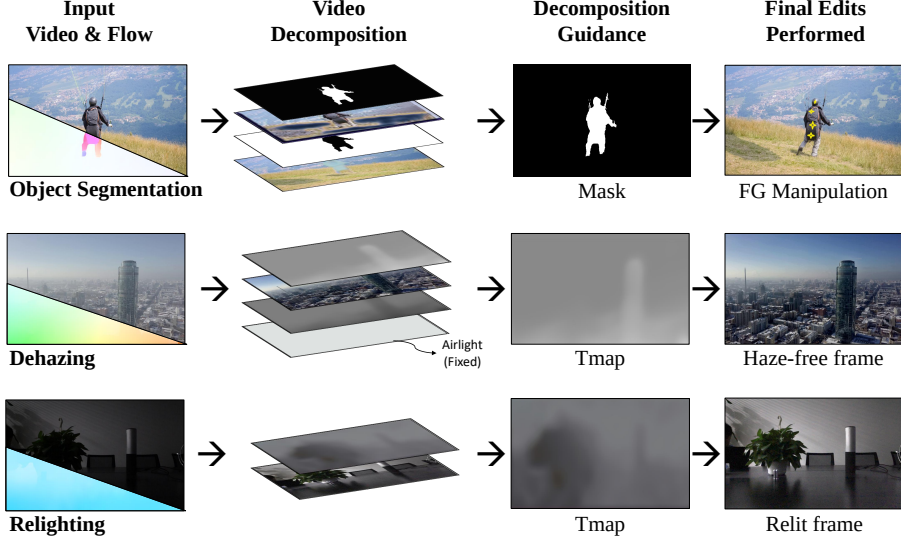


Figure 5.1: **Visual representation of video edits obtained using VDP.** The first row demonstrates a foreground manipulation example that leverages an object mask as the decomposition guide. This object mask is obtained by performing Unsupervised Video Object Segmentation (UVOS), which is a downstream task of video decomposition and is achieved using our proposed framework. Our approach effectively separates the foreground objects and background in the video sequence, enabling us to perform object manipulation. The second row shows the result of our approach for video dehazing, where our method effectively removes the haze from the scene. Finally, the third row showcases the effectiveness of our approach for video relighting, where our method effectively changes the lighting of the scene. Our proposed approach outperforms the state-of-the-art methods for the latter two tasks, highlighting the efficacy of our unified framework for video decomposition.

perform a relighting task. This new formulation leads to significant improvement in performance when compared to the current baseline methods. 3) We achieve state-of-the-art performances on relevant downstream applications of video decomposition, such as video dehazing and relighting. VDP outperforms the existing inference-time optimization techniques for video object segmentation.

5.2 Related Works

Traditionally, priors played a pivotal role in transforming an ‘ill-posed problem’ (one with multiple plausible solutions) to a well-posed problem. Past proposed priors [162, 163, 164, 165, 166] were very limited in their capability of learning a good representation of videos to generalize well on various scenes. All of this changed with the recent developments in deep learning techniques that considerably improve the representation of videos and deliver high-performance gains on video editing tasks.

Established works such as [130, 131, 157, 167, 168] are test time optimization techniques, i.e., they directly work on the corrupted input image/video sequences and do not require any ground truth for training. These approaches have highlighted the importance of formulating a loss function, combined with the optimization of neural network parameters, to achieve significant results for both image and video restoration problems. However, these insights predominantly catered to limited vision tasks such as denoising, super-resolution, and inpainting. Using our approach, we are able to address video relighting, dehazing, and unsupervised video object segmentation tasks under a unified framework. We demonstrate this breadth of applications by leveraging the compositionality properties inherent in videos.

DoubleDIP [20] employs a similar strategy, but their convolutional framework relies heavily on the appearance of the video when optimizing a task objective. This becomes a limitation when working with videos, as motion plays a critical role alongside appearance. Our method tackles this bottleneck by harnessing optical flow

as a motion cue. This inclusion of optical flow not only captures motion effectively but also permits warping operations vital for downstream tasks such as edit propagation, an ability absent in DoubleDIP’s decomposition framework.

Other inference-time optimization methodologies, such as [158, 160, 161, 169] are very task-specific and utilize a singular form of video composition formulation, i.e., α -blending. Techniques like these are heavily reliant on user-generated masks or image-centric baselines. In contrast to these methods, our proposed method can tackle a wide variety of tasks. In addition, our framework does not rely on pretrained image segmentation networks, thereby enhancing its robustness, especially when confronted with unfamiliar objects in videos. Method [17] is another inference-time optimization technique that avoids the use of pretrained image segmentation but encounters issues with objects whose appearances fluctuate over time, perhaps due to shifts in lighting. Such a limitation arises from their approach of encapsulating an object’s global appearance in videos into a single canonical texture layer. In contrast, our proposed methodology takes advantage of frame-by-frame appearance and motion cues, making our model significantly more resilient to variations in lighting.

5.3 Video Decomposition Prior Framework

Given a video sequence denoted as $\{X_t\}_1^T$, this section delves into our framework’s methodology to decompose the video into multiple layers. To facilitate this decomposition, our framework requires an associated forward optical flow of the

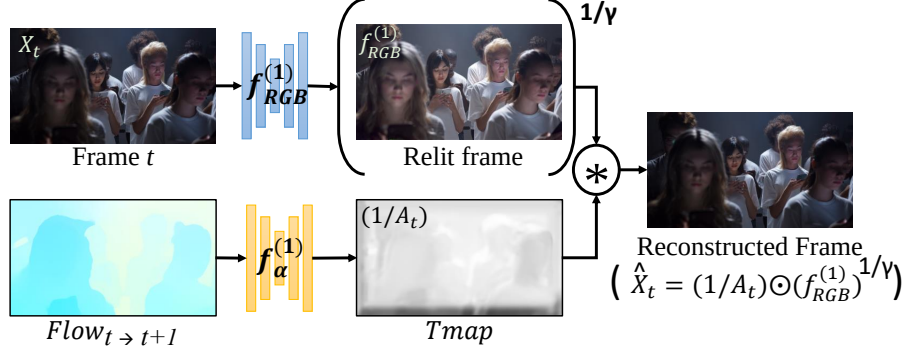


Figure 5.2: **VDP for relighting a video sequence.** In this pipeline, the input video frame t is fed into a shallow U-Net denoted by $f_{RGB}^{(1)}$. While the flow-RGB is given as input to a separate shallow U-Net network denoted by $f_{\alpha}^{(1)}$. **The intermediate output of $f_{RGB}^{(1)}$ is a re-lit version of the input frame t .** While the output of $f_{\alpha}^{(1)}$ is transmission maps or Tmap denoted by $1/A_t$. It is important to note that γ^{-1} is also treated as a trainable parameter in the pipeline. After obtaining the reconstructed frame t , we apply reconstruction and warping losses respectively to optimize $f_{RGB}^{(1)}$, $f_{\alpha}^{(1)}$, γ^{-1} .

input video, represented as $\{F_{t \rightarrow t+1}\}_2^T$. Subsequently, we will detail the design of the modules incorporated within VDP and outline the generic structure of our loss function.

5.3.1 Modules Overview

VDP framework consists of the following two modules.

(I) RGB layer predictor module (RGBnet): RGBnet (functional notation f_{RGB}) is a fully convolutional bottleneck network [170]. Given that we only optimize the weights over a single video, a shallow convolutional U-Net is sufficient for the task. RGBnet takes image frame X_t of the video sequence as input and outputs a RGB layer (a 3 channel layer of same resolution as the input frame X_t).

(II) α layer predictor module (α -net): α -net (functional notation f_{α}) architecture is similar to RGBnet, i.e., a fully convolutional bottleneck network. We utilize a

shallow convolutional U-Net for predicting the Tmaps or opacity layer. α -net takes flow-RGB (RGB representation of forward optical flow denoted as $F_{t \rightarrow t+1}^{\text{RGB}}$) feature maps of the video sequence as input and outputs the Tmap/opacity layer (a 1 channel layer of same resolution as the input frame X_t).

5.3.2 VDP: Optimization objective for generic video decomposition

VDP framework decomposes the input video into multiple layers and then composite these layers again to reconstructs the same video input. In this process, the **intermediate outputs** generated by VDP are of particular interest. In this section, we will discuss a generic structure of the loss function that will help ensure temporal consistency and faithful reconstruction of the original input video.

Loss Objective: The loss objective used to optimize our framework for different tasks is given by the following Eqn 5.1,

$$\mathcal{L}_{\text{final}} = \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{warp}} \mathcal{L}_{\text{warp}}. \quad (5.1)$$

Reconstruction Loss: We use a combination of L1 and perceptual loss [171] to facilitate faithful reconstruction of the original input video, which is given by,

$$\mathcal{L}_{\text{rec}} = \sum_t \|X_t - \hat{X}_t\|_1 + \|\phi(X_t) - \phi(\hat{X}_t)\|_1. \quad (5.2)$$

where, $\phi(\cdot)$ denotes embeddings obtained from the pre-trained VGG network trained on Imagenet [24].

Optical Flow Warp Loss: The intermediate output of interest X_t^o , should follow the same temporal coherence as the input video X_t at any given timestamp. This is ensured by the optical-flow warp loss,

$$\mathcal{L}_{\text{warp}} = \sum_t \|F_{t-1 \rightarrow t}(X_{t-1}^o) - X_t^o\| \quad (5.3)$$

where, $F_{t-1 \rightarrow t}$ denotes the forward optical flow between the frames X_{t-1} and X_t .

The subsequent sections will discuss various applications of our VDP framework.

5.4 Video Relighting

During the nighttime, less amount of luminescence leads to darker capture of a scene. Traditionally, gamma correction is used to relight such captured scene as gamma is what translates well between our eye’s light sensitivity and that of the camera. Mathematically, gamma correction can be defined as given by the Eqn. 5.4

$$X_t^{\text{out}} = A_t \odot (X_t^{\text{in}})^{\gamma} \quad \forall t \in (1, T], \quad (5.4)$$

where, X_t^{out} is the output of gamma correction at time step t . X_t^{in} is the input to which the gamma correction is applied. A_t and γ are the hyperparameters of gamma correction. An image frame is either darkened or lightened up based on the value of γ in Eqn. 5.4. For our model, we change the Eqn. 5.4 as follows,

$$\log(X_t^{\text{in}}) = \frac{1}{\gamma} (\log(1 \oslash A_t) + \log(X_t^{\text{out}})) \quad \forall t \in (1, T]. \quad (5.5)$$

Metrics	ZeroDCE++ (15)	SDSD (5)	StableLLVE (16)	Ours
PSNR	26.46	24.92	26.01	27.92
SSIM	0.85	0.73	0.84	0.90

Table 5.1: **Quantitative evaluation on Video Relighting benchmarks:** For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the re-lit video and ground truth video. We use the pretrain model for the baselines that are publicly made available for calculating the PSNR and SSIM scores. For all baselines and our method, the score is calculated on the SDSD dataset (5). Also, **blue (bold)** denotes the best score.

Where $\oslash$ operation denotes element-wise division.

The goal of the video relighting task is to recover a video $\{(X_t^{\text{out}})\}_1^T$ with good lighting conditions from its poorly lit counter-part video $\{(X_t^{\text{in}})\}_1^T$. We treat the relighting problem as a video decomposition problem, where one layer is the well-lit video. In addition, we model the transmission maps $1 \oslash A_t$ with our α -net model. Further, we model γ^{-1} as a trainable parameter that can take a value between (0,1). In summary, to find the well-lit video from the given input video, we initialize one RGBnet, one α -net, one trainable parameter γ^{-1} , and use the Eqn. 5.5 for reconstruction loss. This whole pipeline is given by Fig. 5.2. One advantage of utilizing the convolution nets to model RGBnet(F_{RGB}) is that convolutional nets provide a high impedance to the noise [130] when reconstructing a natural image. This property proves beneficial as merely applying gamma correction to a dark image can introduce noise into the relit image.

Our final loss objective to perform the relighting task is given by Eqn. 5.1.

We evaluate our method for the video relighting task on the SDSD dataset [5] and compare our model against the existing state-of-the-art baselines. It is important to note that the SDSD dataset consists of low-light captured RGB videos as the

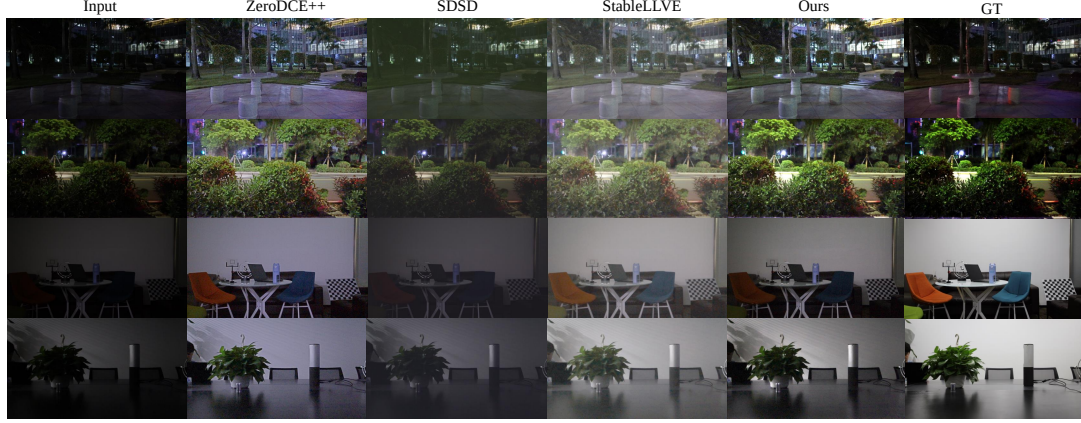


Figure 5.3: **Qualitative evaluation on Video Relighting benchmark:** We compare the re-lit result of our method with the baselines on the SDSD (5) dataset. We compare our method against both the image and video baselines. ZeroDCE++ (15) method is a image-based baseline while SDSD (5) and Stablellve (16) are video baselines. Please note that we have utilized the pretrained models of all the baselines to obtain qualitative results. Methods SDSD and Stablellve require training on the external dataset, while our approach operates directly on the low-light video sequence.

input. The SDSD dataset has a total of 150 sequence pairs (low-lit seq, gt seq). Each sequence contains 100-300 high-resolution frames of size 1920×1080 . We follow the same evaluation strategy for this dataset as the paper [5]. We choose the ZeroDCE++ [15], SDSD[5], and StableLLVE[16] approaches as baselines. SDSD and StableLLVE approaches are trained using the RGB video dataset; hence, these methods act as perfect baselines.

A qualitative evaluation can be seen in Fig. 5.3. Note that ours and the ZeroDCE++ approach do not require any training data. Our approach is optimized directly on the inference-time video sequence instead of other approaches that have to be trained using external data. Table. 5.1 represents the quantitative evaluation of our approach on the SDSD dataset. It can be observed from the table that our method outperforms the existing baselines by a significant margin in the video

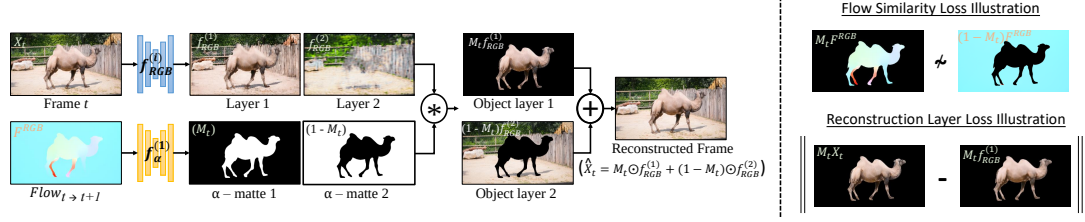


Figure 5.4: **Our Framework:** In this figure, we present the pipeline for the task of decomposing the video into two different components. For the above configuration, the input video frame t is fed into two different shallow U-Nets denoted by $f_{RGB}^{(1)}$ and $f_{RGB}^{(2)}$ respectively. We obtain two different intermediate components of the input frame t in the form of layer 1 and layer 2. Additionally, we process forward flow-RGBs using a separate shallow U-Net network denoted by $f_{\alpha}^{(1)}$. This network outputs α compositing maps, which are then used for blending both layers respectively to reconstruct the original input frame t . After the compositing, we apply various losses (reconstruction loss and regularization loss) depending on the task at hand. **(Right) Illustration of Flow Similarity loss:** we define this loss over flow-RGB image (F^{RGB}). This loss ensures that pixels with similar motions are grouped together. **Illustration of Reconstruction layer loss:** We define this loss over the masked object layers and masked input frame X_t . This loss ensures that the masked object layer such as $M_t f_{RGB}^{(1)}$ has the same appearance as the masked input frame $M_t X_t$.

relighting task.

5.5 Unsupervised Video Object Segmentation

Given a video sequence, the aim of unsupervised video object segmentation (UVOS) is to coherently track and segment the most salient object in a video without any human annotations. Several past works such as these (172, 173, 174, 175, 176, 177, 178, 179, 180) have made attempts to segment the primary object in a video without the use of any annotations. Unlike these works, our goal is not to detect semantic object categories but to find a decomposition that explains the scene motion and its appearance.

We start by leveraging the α -blending equation to write the reconstruction of

the input video,

$$X_t = \sum_{i=1}^L M_t^i \odot f_{\text{RGB}}^i(X_t) \quad \forall t \in (1, T]. \quad (5.6)$$

Here, M_t^i denotes the alpha map for the i^{th} object layer and t^{th} frame. Further, M_t^i is obtained using the following,

$$M_t^i = f_{\alpha}^i(F_{t-1 \rightarrow t}^{\text{RGB}}) \quad \forall t \in (1, n], \quad (5.7)$$

where, $F_{t-1 \rightarrow t}^{\text{RGB}}$ is the RGB image of flow estimate from frame $t - 1$ to t . Additionally, the following condition about the alpha maps should also satisfy

$$J_{h,w} = \sum_{i=1}^L M_t^i, \quad (5.8)$$

where, $J_{h,w}$ denotes the all-ones matrix of size $[h, w]$.

Since the problem of video decomposition is severely ill-posed, the loss objective given by Eqn. 5.1 alone is not enough to find a visually plausible layered decomposition of an input video. Additionally, to find the right decomposition we need the opacity layers to be binary masks. Hence, we introduce a few other regularization losses to mitigate such challenges.

Flow Similarity Loss: This loss ensures the motion of the layer i is uncorrelated with the rest of the video layers. We define the loss as cosine similarity between the VGG embeddings of the masked flow-RGB of layer i with the rest of the layers. The illustration for this loss is depicted in Fig. 5.4. We can mathematically write this

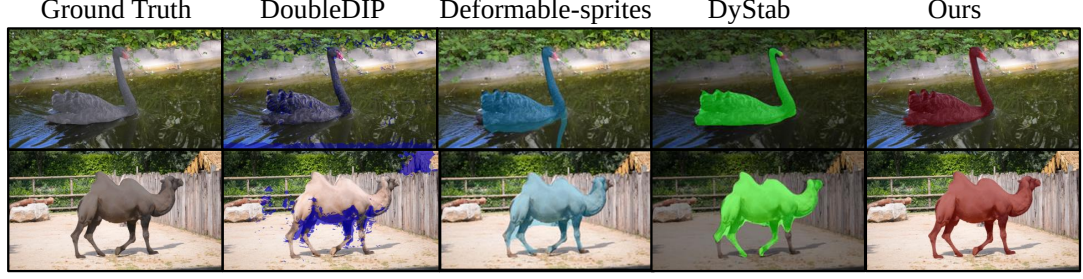


Figure 5.5: **Qualitative evaluation on VOS benchmark:** We compare the mask from our method with the baselines on DAVIS-16 (1) dataset. We compare our method against the baselines DS (17) and DyStab (18). Please note the above masks (exception: DoubleDIP) for baselines are pre-computed masks provided by the authors of the papers.

Model	Flow model	DAVIS (1)	FBMS (181)
DoubleDIP (20)	-	24.6	-
ARP (182)	CPMFlow (183)	76.2	59.8
CIS (184)	PWCNet (185)	71.5	63.5
MG (23)	RAFT (149)	68.3	53.1
DS (17)	RAFT (149)	79.1	71.8
DyStab* (18)	PWCNet (185)	80.0	73.2
Ours	RAFT (149)	81.1	73.2

Table 5.2: **Quantitative evaluation on VOS benchmarks:** For the evaluation of our method, we calculate the IOU score ($\mathcal{J}$) between the generated masks and the ground truth mask. Also, **blue (bold)** denotes the best score. Please note that the DyStab is trained using a video dataset, while all the other methods require only a test time video sequence to segment the object from the video.

loss as follows,

$$\mathcal{L}_{\text{Fsim}} = \frac{\phi(M_t^i \odot F^{\text{RGB}}) \cdot \phi((1 - M_t^i) \odot F^{\text{RGB}})}{\|\phi(M_t^i \odot F^{\text{RGB}})\| \|\phi((1 - M_t^i) \odot F^{\text{RGB}})\|}. \quad (5.9)$$

Here, $\phi(\cdot)$ in the Eqn. 5.9 denotes the embeddings obtained by the pre-trained VGG network trained on ImageNet. F^{RGB} is the RGB image of flow estimate from frame $t - 1$ to t .

Mask Loss: This loss ensures binarization in the generated layer mask. We utilized

the mask loss put forth by [20]. We can mathematically write this loss as follows,

$$\mathcal{L}_{\text{Mask}} = \sum_{L,T} (|M_t^i - 0.5|)^{-1}. \quad (5.10)$$

Reconstruction layer Loss: This loss ensures the appearance of the layer i is a segment of the original input video. We define the loss as an L1 loss between the masked RGB image prediction of layer i and the masked RGB image prediction of the original video. We can mathematically write this loss as follows,

$$\mathcal{L}_{\text{layer}} = \|M_t^i \odot X_t - M_t^i \odot \hat{X}_t\|. \quad (5.11)$$

Here, X_t is the RGB frame in the video sequence at time-step t . M_t^i is the alpha map for the i^{th} object layer and t^{th} frame.

Final Modified Loss: We combine all the losses together into a final loss term as follows,

$$\begin{aligned} \mathcal{L}_{\text{UVOS}} = & \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{Fsim}} \mathcal{L}_{\text{Fsim}} + \lambda_{\text{layer}} \mathcal{L}_{\text{layer}} \\ & + \lambda_{\text{warp}} \mathcal{L}_{\text{warp}} + \lambda_{\text{Mask}} \mathcal{L}_{\text{Mask}}. \end{aligned} \quad (5.12)$$

We leverage our final modified loss objective, given by Eqn. 5.12, to perform unsupervised video object segmentation. For this task, we want to split an input video into foreground and background layers. Hence, we initialize 2 RGBnets (f_{RGB}) and one α net (f_{α}). Weights for all these modules are optimized using the test sequence only. The illustration of this is shown in the Fig. 5.4. Because we are

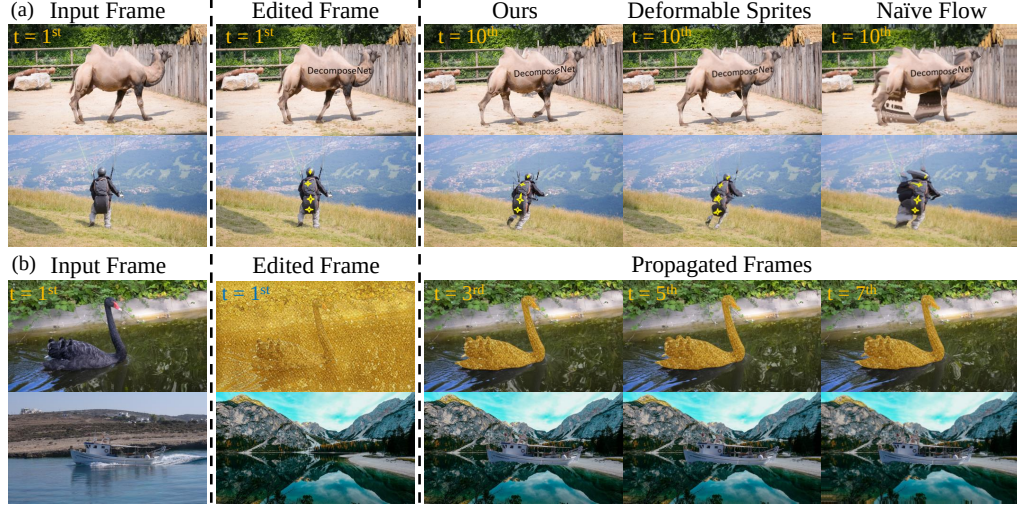


Figure 5.6: **Edits propagation:** First column in the figure represents the input frame to be edited. Second column represents the edits performed. Next columns in Fig. (a) represent the qualitative comparison of propagated edits with baselines, while Fig. (b) represents propagated edits throughout the video. From Row 1, 2 - it can be observed that the quality of our framework’s propagated edited decals is much better as compared to the baselines. The legs of the camel(Row 1) and parachute bag shape(Row 2) are preserved in our method’s edit propagation. Row 3 - It can be observed that consistent edits in a particular layer can be performed throughout the video. In this case, we show an example of style transfer being performed on the swan. Row 4 - we can selectively replace the background layer and propagate the edits through the video.

dealing with a decomposition of the input video into two layers, i.e., as $L = 2$, we modify Eqn. 5.6 as follows,

$$X_t = M_t \odot f_{\text{RGB}}^1(X_t) + (1 - M_t) \odot f_{\text{RGB}}^2(X_t) \quad \forall t \in (1, n]. \quad (5.13)$$

All the loss functions in the Eqn. 5.12 are modified with the value of $L = 2$. We fit our model parameters using loss function given by Eqn. 5.12. We then use the learned α -net to generate a segmentation mask for the input video. The predicted segmentation masks for each timestamp can be obtained by $M_t = f_{\alpha}(F_{t-1 \rightarrow t}^{\text{RGB}})$. where, $F_{t-1 \rightarrow t}^{\text{RGB}}$ is the RGB image of flow estimate from frame $t - 1$ to t .

We evaluate our method for unsupervised video object detection on the DAVIS-16 dataset [1] and compare our model against the existing baselines. A qualitative comparison between our approach and the baselines can be seen in Fig. 5.5. It can be observed from Fig. 5.5 that our method performs better than the existing methods in terms of segmenting the most salient object from the scene. Table. 5.2 presents a quantitative evaluation of our approach. Our method outperforms the existing inference-time optimized baselines in the unsupervised video object segmentation setting. DoubleDIP is the worst-performing baseline in our evaluation.

5.5.1 Coherent Edits Propagation

With our approach, we are able to perform downstream applications like coherently editing the foreground and background object layers in a scene. It can be observed from Fig. 5.6 the type of edits we can perform on one layer in some keyframe, which can then be propagated throughout the video. The distinct advantage of our method over other such methods is that the user can perform the edit on any keyframe which has a better view of the object. This makes it easier for the user to make the edit instead of making it in one global atlas for the object layer with a distorted single view. In Fig. 5.6, we perform different edits such as add decals or modify a keyframe with style transfer [186] or swap the background. These edits are then consistently propagated using a combination of flow warping operation and composition using the obtained masks.

5.6 Video Dehazing

Videos captured in outdoor settings are often degraded by a scattering medium (e.g., haze, fog, underwater scattering). The degradation is such that it grows with the depth of the scene. Typically, we can write a hazy video $\{X_t\}_1^T$ as put forth by papers [20, 166, 187, 188]:

$$X_t = \alpha \odot \text{Clr}(X_t) + (1 - \alpha) \odot A_t, \quad (5.14)$$

where A_t is the Airlight map (A-map), $\text{Clr}(X_t)$ is the haze-free image, and α here is the transmission (t-map). The goal of video dehazing is to recover the clear underlying video $\{\text{Clr}(X_t)\}_1^T$ from a hazy input video $\{X_t\}_1^T$, i.e., recover the input video that would have been captured on a clear day with good visibility conditions. We treat the dehazing problem as a video decomposition problem, where one layer is the haze-free video and the other layer is the Airlight map. This Airlight map is obtained using the standard methodology proposed in [189] and kept as a fixed (non-trainable) layer. Further, we model the transmission t-maps with our α -net model. In summary, to find the dehazed video from the given input video, we initialize one RGBnet, one α -net, and use the Eqn. 5.14 for reconstruction loss.

To optimize the model parameters, we optimize the objective function given by Eqn. 5.1.

We evaluate our method for the video dehazing task on the REVIDE dataset [6] and compare our model against the existing state-of-the-art baselines. It is important

Metrics	EDVR (11)	DoubleDIP (20)	MSBDN (19)	CG-IDN (6)	Ours
PSNR	21.22	17.32	22.01	23.21	24.83
SSIM	0.8707	0.7942	0.8759	0.8836	0.9164

Table 5.3: **Quantitative evaluation on Video Dehazing benchmarks:** To evaluate our method, we calculate the Avg PSNR and SSIM scores between the generated haze-free video and ground truth video. For all baselines and our method, the score is calculated on the REVIDE dataset (6).

to note that the REVIDE dataset consists of real-world captured hazy RGB video and clean video pairs. REVIDE dataset has a total of 48 sequence pairs (hazy seq, gt seq). There are 42 sequence pairs for the train set and the remaining 6 sequence pairs for the test set. Each sequence contains 10-60 high-resolution frames of size 2708×1800 . We choose both image-based and video-based approaches as baselines. MSBDN [19] is an image based dehazing baseline, while DoubleDIP [20], EDVR [11] and CG-IDN [6] are the video based baseline. EDVR and CG-IDN approaches are trained using the videos of the REVIDE dataset; hence, these methods act as perfect baselines.

A qualitative evaluation can be seen in Fig. 5.7. Note that only ours and DoubleDIP approaches do not require any task-specific training data. Both approaches are optimized directly on the inference-time video sequence instead of other approaches that have to be trained using external data. Table. 5.3 represents the quantitative evaluation of our approach. It can be observed from the table that our method outperforms the existing baselines by a significant margin in the video dehazing task. DoubleDIP performs worst among the baselines in our quantitative evaluation.



Figure 5.7: **Qualitative evaluation on Video Dehazing benchmark:** We compare the dehazed result of our method with the baselines on the REVIDE (6) dataset. We compare our method against both the image and video baselines. MSBDN (19) method is an image-based baseline while EDVR (11), DoubleDIP (20) and CG-IDN (6) are video baselines. Please note that both the EDVR and CG-IDN methods are trained using the training data of the REVIDE dataset. In contrast, our approach operates directly on the hazy video sequence. As can be seen from the figure, our approach to dehazing produces a color palette in the dehazed frame that closely resembles the color palette in the ground truth frame.

5.7 Limitations of our approach

Our method has four limitations that must be considered. First, the accuracy of the decomposition is dependent on the effectiveness of the optical flow estimator network (RAFT). This means that any inaccuracies or limitations of the network will directly impact the quality of the results. Second, the number of separable layers is a hyperparameter that must be determined prior to implementation. Third, If the number of separable layers is set to N , then $2N$ independent f_{RGB} and f_{α} need to be initialized, which can significantly increase computational costs. Finally, it’s worth noting that our method is a test-time optimization technique and may be slower than its trained counterparts. However, as mentioned earlier, our model can be sped up by resizing the frame size for the VOS task or by utilizing multiple GPUs for optimization. These limitations must be taken into consideration when implementing our method in practice.

5.8 Conclusion

We have demonstrated in the paper how the layered decomposition could be applied to a wide variety of downstream tasks. It is important to note that our model does not require any explicit training, i.e., we optimize the parameters using the test sequence itself. Additionally, in this paper, we achieve state-of-the-art performances in downstream tasks such as UVOS (among the other test time optimization techniques), dehazing, and relighting. We also demonstrated that the targeted edits propagation can also be performed as a downstream task of UVOS. With the help of semantic or perception cues, we believe our framework can also be extended for monocular depth estimation, semantic segmentation, and other high-level vision tasks.

5.9 Appendix

5.9.1 Network Architecture for RGBnet and α -net

RGBnet is a fully convolutional network. It is designed in such a way as to handle frames of all resolutions. The input to this network is RGB frame at time t . We utilize a U-net architecture for this network without the skip connections. More details about the architecture can be found in Table. 5.4. Similarly, more details about the α -net can be found in Table. 5.5.

Module	Type	Kernel Size	Num Inputs	Num Outputs	Stride	Padding
Encoder	Conv2D	3×3	3	64	1	1
	Conv2D	3×3	64	64	1	1
	Maxpool2D	2×2	64	64	2	0
	Conv2D	3×3	64	96	1	1
	Conv2D	3×3	96	128	1	1
	Maxpool2D	2×2	128	128	2	0
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Maxpool2D	2×2	128	128	2	0
	Conv2D	4×4	128	96	1	0
Decoder	ConvTranspose2d	4×4	96	128	1	0
	Bicubic upsample	(scale =2)	-	-	-	-
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Bicubic upsample	(scale =2)	-	-	-	-
	Conv2D	3×3	128	96	1	1
	Conv2D	3×3	96	64	1	1
	Bicubic upsample	(scale =2)	-	-	-	-
	Conv2D	3×3	64	64	1	1
	ConvTranspose2D	3×3	64	3	1	1

Table 5.4: **RGBnet**: Each convolution is followed by batch normalization and a leaky rectified linear unit (LeakyReLU; negative slope = 0.2), except for the last layer of the encoder and decoder. In the last layer of the encoder and decoder, we use the sigmoid activation function. We utilize Pytorch for our implementation.

5.9.2 Optimization Details

To optimize our model, we use a single Nvidia A6000 GPU with 48G memory to process a single video at a time of resolution 856x480. We optimize the module ($f_{\text{RGB}}(\cdot)$ and $f_{\alpha}(\cdot)$) weights using the entire test sequence with the *Adam optimizer* at a learning rate in the range of $[0.00002, 0.002]$. Because our approach is non-recurrent, it can be scaled up by parallelizing over multiple GPUs, which speeds up the process for longer sequences. We achieve a processing rate of 60 frames/min for dehazing and relighting and 40 frames/min for VOS with one salient object. However, we can speed up the VOS task by reducing the dimension from 856x480 to 224x128

Module	Type	Kernel Size	Num Inputs	Num Outputs	Stride	Padding
Encoder	Conv2D	3×3	3	64	1	1
	Conv2D	3×3	64	64	1	1
	Maxpool2D	2×2	64	64	2	0
	Conv2D	3×3	64	96	1	1
	Conv2D	3×3	96	128	1	1
	Maxpool2D	2×2	128	128	2	0
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Maxpool2D	2×2	128	128	2	0
	Conv2D	4×4	128	96	1	0
Decoder	ConvTranspose2d	4×4	96	128	1	0
	Bicubic upsample	(scale =2)	-	-	-	-
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Conv2D	3×3	128	128	1	1
	Bicubic upsample	(scale =2)	-	-	-	-
	Conv2D	3×3	128	96	1	1
	Conv2D	3×3	96	64	1	1
	Bicubic upsample	(scale =2)	-	-	-	-
	Conv2D	3×3	64	64	1	1
	ConvTranspose2D	3×3	64	1	1	1

Table 5.5: **α -net:** Each convolution layer is followed by batch normalization and a leaky rectified linear unit (LeakyReLU; negative slope = 0.2), except for the last layer of the encoder and decoder. In the last layer of the encoder and decoder, we use the sigmoid activation function. We utilize Pytorch for our implementation.

and then running the VOS task pipeline. This results in a 16x speed-up of the pipeline, increasing the processing rate to 640 frames/min without affecting the efficacy of our method. We use 100 epochs for a 60-frame sequence in VOS and 60 epochs for dehazing and relighting. Additionally, we conducted our experiments using multiple random initializations of $f_{\text{RGB}}(\cdot)$ and $f_{\alpha}(\cdot)$ modules and observed consistent quantitative results across all runs.

5.9.2.1 Hyper parameter setting

Unsupervised Video Object Segmentation: For getting a good performance on the UVOS task we utilize the following weights for the different losses; $\lambda_{\text{rec}} = 1$,

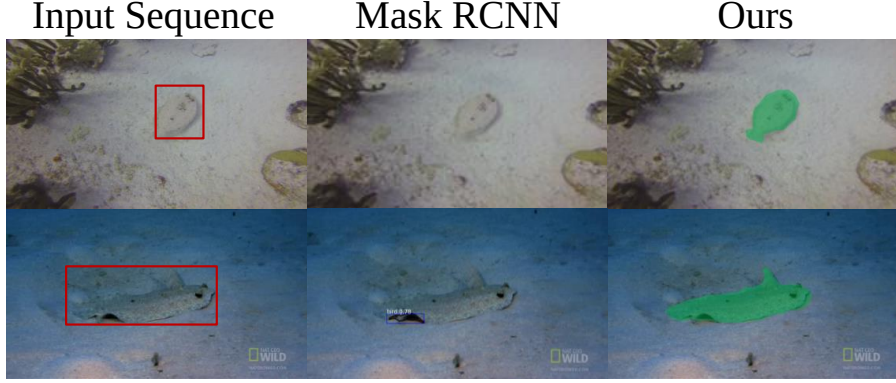


Figure 5.8: **Comparison with pretrained image segmentation model:** The examples presented in the figure above are samples from the MoCA dataset [21]. In comparison to MaskRCNN [22], a pretrained image segmentation model trained on the COCO dataset, our model is more robust in handling unfamiliar objects in the scene. The figure shows some examples where MaskRCNN fails to segment the unfamiliar object correctly, while our approach produces much better segmentation results.

$\lambda_{\text{Fsim}} = 0.001$, $\lambda_{\text{layer}} = 1$, $\lambda_{\text{warp}} = 0.01$ and $\lambda_{\text{Mask}} = 0.01$.

Unsupervised Video Dehazing: For getting a good performance on the Dehazing task we utilize the following weights for the different losses; $\lambda_{\text{rec}} = 1$ and $\lambda_{\text{warp}} = 0.02$.

Unsupervised Video Relighting: For getting a good performance on the Relighting task we utilize the loss weights similar to the Dehazing task, i.e., $\lambda_{\text{rec}} = 1$ and $\lambda_{\text{warp}} = 0.02$.

5.9.3 UVOS

5.9.3.1 UVOS - Scene with Irregular Objects

In Fig. 5.8, we demonstrate that our method doesn't mandate the object to belong to a common category to achieve the correct decomposition. This is in stark contrast to other methods like that of [161], which heavily relies on MaskRCNN or a user-input mask for video decomposition. This flexibility provides our approach

with an advantage, allowing for precise video foreground (FG) and background (BG) edits and ensuring seamless propagation throughout the sequence.

5.9.3.2 UVOS - Lighting Changes

It's notable that the efficacy of deformable sprites [17] tends to deteriorate in video sequences where the appearance of an object undergoes alterations due to shifts in lighting. This decrement in performance emerges primarily because deformable sprites attempt to encapsulate an object's universal appearance using a singular canonical RGB texture. In contrast, our framework operates by modeling video decomposition at an individual frame level. Such an approach amplifies the resilience of the object segmentation process against lighting modifications, a fact that is evident from Fig. 5.9.(a).

5.9.3.3 UVOS - Decomposition into Multiple Layers

Our model's capabilities are not confined to a binary layer decomposition (Foreground layer and Background layer). As depicted in Fig. 5.9.(b), upon configuring the layer parameter to 2, our model precisely segregates the foreground from the background. Intriguingly, when the layer parameter is adjusted to 3, the model successfully isolates each person into their distinct layer while maintaining a separate layer for the background.

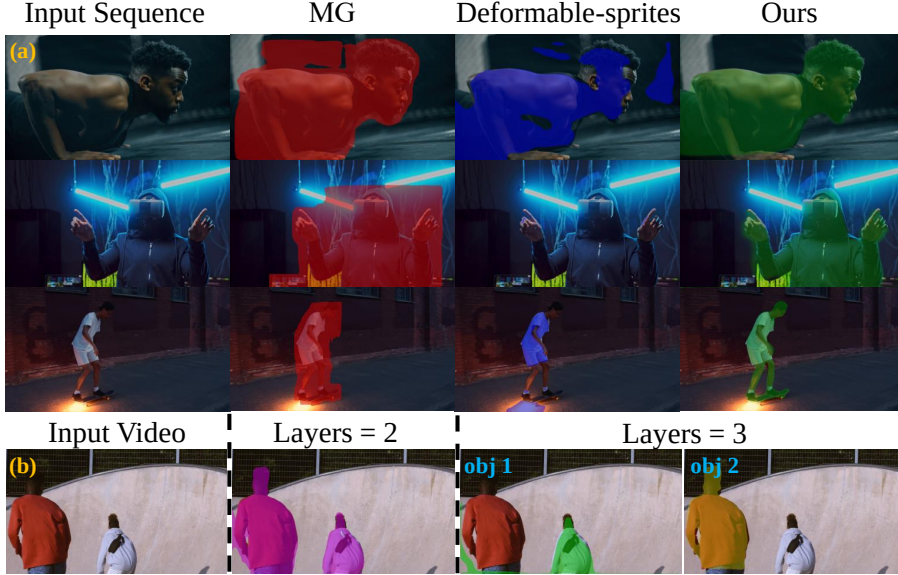


Figure 5.9: **(a) Segmentation in variable lighting conditions throughout the scene.** We compare the mask from our method with the baselines Deformable sprites [17] and MG [23] on frames of three video sequences. It can be observed from the above figure methods such as Deformable-sprites and MG are more prone to lighting changes as opposed to our method. **(b) Decomposition into multiple layers** We demonstrate an example where the number of layers L is set to both $L = 2$ and $L = 3$. $L = 2$ results in foreground and background layers separation, while $L = 3$ results in the separation of individual person in different object layers.

5.9.3.4 UVOS - Ablation Study

We performed ablation experiments to study the effect of losses for our unsupervised video object segmentation method. We report the quantitative results on the DAVIS-16 dataset in Table. 5.6. In Fig. 5.10, we show a qualitative comparison between architectures optimized with different subsets of losses given by Eqn. 12. It can be observed from Fig. 5.10 that removing the reconstruction loss drastically reduces the ability of our method to segment the object of interest in the scene. This is also reflected in the quantitative evaluation given by Table. 5.6. Dropping the $\mathcal{L}_{\text{Fsim}}$ reduces our model’s ability to separate the motion of the foreground object from the



Figure 5.10: **Ablation Study:** We study the effect of using only a subset of the loss function for the video object segmentation task.

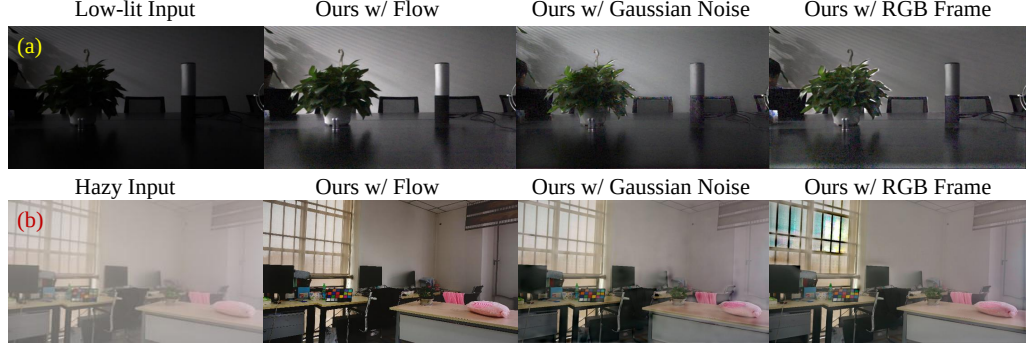


Figure 5.11: **Ablation Study:** This figure presents the results of an ablation study where we analyzed the impact of using FlowRGB, Gaussian Noise, and RGB frame as input to the α -net. Row (a) shows a qualitative comparison for relighting tasks, where we evaluated three different settings for the α -net. Row (b) presents a comparison for the dehazing task.

background scene. Optimizing Eqn. 12 without the $\mathcal{L}_{\text{layer}}$ results in a performance drop as the obtained object mask lacks the appearance grouping provided by the addition of the layer loss. Lastly, dropping the $\mathcal{L}_{\text{warp}}$ reduces the temporal consistency of the masks throughout the sequence.

5.9.3.5 Complete VOS Evaluation for DAVIS-16

We include the supervised VOS approaches along with the unsupervised approaches for the completeness of the evaluation. It is important to note that unsupervised in the context of this paper implies that no human annotation is required either for the training or testing process of the approach. Table. 5.8 reflects

	w/o $\mathcal{L}_{\text{rec}}$	w/o $\mathcal{L}_{\text{Fsim}}$	w/o $\mathcal{L}_{\text{layer}}$	w/o $\mathcal{L}_{\text{warp}}$	All Losses
Davis	52.6	67.3	72.8	78.5	81.1

Table 5.6: **Ablation Study:** We utilize the DAVIS-16 dataset for performing the ablation study. We calculate the IOU score ($\mathcal{J}$) between the generated mask and the ground truth mask.

Metrics	Task	Ours w/ FlowRGB	Ours w/ Gaussian Noise	Ours w/ RGB
PSNR	Dehazing	24.83	16.79	16.84
SSIM	Dehazing	0.9164	0.7768	0.7721
PSNR	Relighting	27.92	23.74	24.52
SSIM	Relighting	0.9046	0.7221	0.7342

Table 5.7: **Quantitative evaluation on Dehazing/Relighting ablations:** This Table presents the results of quantitative evaluations conducted on dehazing and relighting tasks using three different settings of our method. To assess the performance of our approach, we measured the average PSNR and SSIM scores between the processed videos and ground truth videos. For dehazing evaluations, we used the REVIDE [6] dataset, and for relighting evaluations, we used the SDS D [5] dataset.

the evaluation of the baselines on the DAVIS-16 dataset for VOS task.

5.9.4 Intuition for Dehazing/Relighting Tasks

Dehazing: Previous studies, such as [189], have shown that small image patches are recurrent in natural images, but this recurrence is disrupted in non-ideal conditions such as haze/noise. [20] established this patch recurrence property could be exploited using a coupled DIP [130]. When a combination of multiple DIPs is used to reconstruct a frame, those DIPs tend to “split” a frame such that the patch distribution of each DIP output follows high patch recurrence. Hence, when we use a coupled version of U-nets in our framework to reconstruct the haze frame, it always converges into a natural image with repetitive patches (termed as a haze-free image) and a non-linear airlight map.

For the relighting task, we begin by defining a manifold for the relit frame, which is based on the gamma correction formulation as delineated in Eqn. 5. Our architectural choice incorporates convolutional U-Nets, leveraging their inherent property to enhance self-similarity in the produced outputs. Notably, natural images, particularly those captured in optimal conditions, consistently exhibit the patch recurrence property. When we employ the U-Net for optimization, its output (termed as ‘relit’ frame) reliably gravitates towards a natural image marked by repetitive patches. Concurrently, the system fine-tunes and selects the appropriate parameters for γ^{-1} and Tmaps ($1/A_t$). A salient feature of the U-Net’s output is its pronounced patch recurrence, which, in turn, translates to a substantial reduction in noise levels for the relit frame. This inherent noise reduction alleviates the need for a subsequent, explicit denoising process.

5.9.5 Ablation for Dehazing and Relighting tasks

We perform ablation studies of our framework for dehazing and relighting tasks. We evaluate our method utilizing three different settings, namely

- First setting, where we provide flow-RGBs as an input to α -net (ours VDP prior setting).
- Second setting, here, we provide a Gaussian Noise as an input to α -net.
- Third setting, in which we provide an RGB frame as an input to α -net.

We report the quantitative evaluation on REVIDE dataset for dehazing and SDSD dataset for relighting tasks. Please refer to Table. 5.7 and Fig. 5.11 for the findings.

Model	Annotations	No. of Annotations	DAVIS [1]	FBMS [181]
DoubleDIP [20]	-	-	24.6	-
ARP [182]	-	-	76.2	59.8
CIS [184]	-	-	71.5	63.5
MG [23]	-	-	68.3	53.1
DS [17]	-	-	79.1	71.8
DyStab* [18]	-	-	80.0	73.2
Ours	-	-	81.1	73.2
COSNet [159]	✓	17,000	80.5	75.6
AnDiff [190]	✓	2,000	81.7	-
EPONet [191]	✓	2,000+	80.6	-
PDB [192]	✓	17,000	77.2	74.0
LVO [193]	✓	2,000+	75.9	65.1
MATNet [194]	✓	14,000	82.4	76.1
DyStab [18]	✓	2,000	82.8	73.2

Table 5.8: **Quantitative evaluation on VOS benchmarks:** This Table presents the results of a quantitative evaluation of our method on VOS benchmarks. To assess the performance of our approach, we calculated the IOU score ($\mathcal{J}$) between the generated masks and the ground truth mask on the DAVIS dataset [1]. The scores were computed for all baselines and our method. It is important to note that the methods listed above the line require no annotated data for performing the VOS task, while the methods mentioned below the line require annotated data for training. The best score in each category is highlighted in **blue (bold)**.

It can be observed that using flow-RGBs as input to α -net provides significant gain over the other settings. We attribute this behavior to information obtained from neighboring frames that help the model to refine the Tmaps further and produce results with higher fidelity.

It is important to note that in our experiments, we initially attempted to employ raw (x,y) offsets as feature maps for input to the α -Net. However, this approach exhibited suboptimal normalization performance. Conversely, the colorization of optical flow effectively normalizes flow-rgb values within the range [0,1], proving to be a more suitable input for the α -Net.

Metrics	Naive Flow	Deformable Sprites [17]	Ours
PSNR	22.54	27.50	29.73
SSIM	0.4527	0.9543	0.9726

Table 5.9: **Quantitative evaluation of Video Edits Propagation:** For the evaluation of our method, we calculate the Avg PSNR and SSIM scores between the reconstructed videos and ground truth video sampled from the DAVIS-16 dataset 1. We use the code for the baselines that are publicly available to calculate the PSNR and SSIM scores. For all baselines and our method, the score is calculated by averaging the PSNR and SSIM value over a sequence length of 15 frames (frames: 2-16).

5.9.6 Edit propagation

In the case of Salient Object Stylization/Background swapping, we utilize X_0 as the edited frame. The forward optical flow $F_{t \rightarrow t+1}$ is obtained using the original input video. Given a frame X_t at time t and the forward flow $F_{t \rightarrow t+1}$ from frame t to frame $t + 1$, we can define the warped frame X'_{t+1} using the following relation:

$$X'_{t+1}(u, v) = X_t(u + \Delta u, v + \Delta v) \quad (5.15)$$

Where Δu and Δv are the horizontal and vertical displacements, respectively, from the forward flow $F_{t \rightarrow t+1}$ at pixel location (u, v) .

Furthermore, let M_{t+1} be a binary mask at time $t + 1$ where its values are either 0 or 1. The prediction $\hat{X}_{t+1}$ for a pixel at location (u, v) is then obtained by:

$$\hat{X}_{t+1}(u, v) = M_{t+1}(u, v) \odot X_{t+1}(u, v) + (1 - M_{t+1}(u, v)) \odot X'_{t+1}(u, v) \quad (5.16)$$

This formulation suggests that the prediction $\hat{X}_{t+1}$ is a weighted combination of the actual frame X_{t+1} and the warped frame X'_{t+1} , where the weighting is determined by the mask M_{t+1} .

5.9.6.1 Edits - Evaluation

To quantitatively evaluate the effectiveness of our approach for propagating coherent edits, we conducted a proxy experiment. Rather than editing the frame, we applied the warping operations without modifying the original frame. We then evaluated the quality of the reconstructed warped frames and compared our results against the baselines. The findings of this experiment are presented in Table. 5.9, which shows that our method outperformed all of the other baselines.

5.9.7 Limitations of our approach

Our method has four limitations that must be considered. First, the accuracy of the decomposition is dependent on the effectiveness of the optical flow estimator network (RAFT). This means that any inaccuracies or limitations of the network will directly impact the quality of the results. Second, the number of separable layers is a hyperparameter that must be determined prior to implementation. Third, If the number of separable layers is set to N , then $2N$ independent f_{RGB} and f_{α} need to be initialized, which can significantly increase computational costs. Finally, it's worth noting that our method is a test-time optimization technique and may be slower than its trained counterparts. However, as mentioned earlier, our model can be sped

up by resizing the frame size for the VOS task or by utilizing multiple GPUs for optimization. These limitations must be taken into consideration when implementing our method in practice.

5.9.8 Optical Flow

Optical flow pertains to the pattern of visible motion of objects, surfaces, and edges within a visual scene. It provides an estimation of the motion of objects between two consecutive frames, which results from the relative movement between the object and the camera. **Unless mentioned otherwise we use pretrained RAFT [149] model to predict the optical-flow between the two consecutive frames.**

Given two successive frames $X_t(u, v)$ and $X_{t+1}(u, v)$, the goal of optical flow is to identify a displacement vector $(\Delta u, \Delta v)$ for each pixel (u, v) such that:

$$X_t(u, v) = X_{t+1}(u + \Delta u, v + \Delta v) \quad (5.17)$$

Here, Δu and Δv represent the horizontal and vertical displacements, respectively.

5.9.9 Optical Flow Warping Operation

Optical flow warping denotes the application of the computed optical flow vectors to "warp" or "remap" one image in order to predict the subsequent image in the sequence. For each pixel in the present frame, its new position in the next frame is forecasted via the optical flow, transferring the intensity to this new location.

Given an frame X_t and its optical flow $F_{t \rightarrow t+1} = (\Delta u, \Delta v)$, the warped image X'_t for the next time frame $t + 1$ can be expressed as:

$$X'_t(u + \Delta u, v + \Delta v) = I_t(u, v) \quad (5.18)$$

5.9.10 Optical Flow RGB Representation

To visualize optical flow vectors, one can represent them in an RGB format. The 2D optical flow vectors $(\Delta u, \Delta v)$ are first converted into polar coordinates (r, θ) . The hue in the HSV color space corresponds to the motion direction, while the intensity or value indicates the motion magnitude.

To transition from $(\Delta u, \Delta v)$ to (r, θ) :

$$\theta = \arctan 2(\Delta v, \Delta u) \quad (5.19)$$

$$r = \sqrt{\Delta u^2 + \Delta v^2} \quad (5.20)$$

Subsequently, θ and r are proportionally scaled and mapped to their respective HSV values. Afterward, this HSV image can be converted into RGB for effective visualization. Note that to enhance visualization, there's often an imposed limit on the maximum value of r . Thus, any motion exceeding this threshold is typically visualized at maximum brightness.

Chapter 6: Future Work and Conclusion

6.1 Future Work

There are several promising research directions that build upon our work and offer exciting opportunities for further exploration. In future work, we aim to pursue these avenues, which we outline and discuss in detail in the following sections.

While our works video dynamics prior (VDP), video decomposition prior (VDP), and continuous video flow (CVF) models have demonstrated considerable promise across multiple video processing tasks, there are several areas that warrant further exploration. Future research directions should focus on both improving the efficiency of these models and extending their applicability to a broader set of vision problems.

6.1.1 Data-Free Enhancement Techniques

For Video Dynamics Prior [78], one key limitation remains the longer processing time during test-time inference, which stems from its sequential nature. We aim to explore more efficient architectures that reduce the time complexity while maintaining the state-of-the-art performance demonstrated by our spatial pyramid loss. Techniques such as parallelization, model pruning, or leveraging advances in

hardware acceleration could help mitigate the current computational bottlenecks. Moreover, enhancing the model’s capacity to handle real-time video tasks without sacrificing accuracy is an essential next step.

For the Video Decomposition Prior (VDP), we have demonstrated its potential across a variety of downstream tasks. Future work could investigate how layered decomposition techniques can be generalized to higher-level vision tasks such as monocular depth estimation and semantic segmentation. These tasks, often requiring rich temporal and spatial cues, stand to benefit from the intrinsic structure our model provides. Furthermore, multi-task learning frameworks could be explored to unify various vision tasks under a single model architecture, optimizing both time and resource efficiency.

Finally, we believe that future research direction could focus on integrating semantic cues such as depth into video processing models. This would enable tasks such as targeted edits propagation to become more robust, especially in dynamic scenes with occlusion or complex object interactions. The inclusion of perception-driven information can bridge the gap between low-level video enhancement tasks and higher-level vision applications, potentially extending our framework’s use case to real-world scenarios, such as autonomous driving or augmented reality.

6.1.2 Future Directions for Enhancing Temporal Coherence in Continuous Video Flow (CVF)

In the case of Continuous Video Flow (CVF), one future research direction could focus on further enhancing the model’s temporal coherence by leveraging advanced methods for continuous representations, such as neural ordinary differential equations (ODEs). Incorporating ODE-based solvers into the CVF framework could help smooth out temporal inconsistencies and better capture long-range dependencies in videos. Additionally, exploring spatio-temporal transformers could improve the model’s ability to handle complex, multi-dimensional relationships across frames. This could enable CVF to achieve even higher accuracy while remaining computationally efficient.

Another direction that could be used to improve the performance of CVF is to use implicit neural representation(INRs) to perform the encoding of frames. Generative training in the parameter space of the implicit model, can enable the synthesis process to operate within a structured, lower-dimensional space that inherently respects spatio-temporal constraints. This paradigm shift in video synthesis can have several critical advantages. First, it can enhance the spatio-temporal consistency by leveraging the continuous nature of implicit representations, which allows for smoother transitions across frames. Second, it could offer efficient video representations by encoding videos compactly and flexibly in the implicit space, making it scalable across various framerates and resolutions. Lastly, the method’s inherent flexibility will allow for adaptability to any framerate or resolution, addressing a

significant limitation in current generative models, which often struggle to generalize across different video specifications.

6.2 Conclusion

Through this thesis we have looked at various video generation and enhancement techniques commonly used across the generation landscapes. Starting with Diverse Video Generation [48] we look at how a gaussian process prior can help generate diverse video trajectories from the same given context. We also proposed a diversity metric that helped quantitatively evaluate the diversity in the generated sequences.

Video Dynamics Prior [78] where we introduced a test-time optimization technique to perform robust video enhancement tasks. In this paper we demonstrated that we can perform enhancement task such as denoising, frame interpolation, super-resolution and inpainting without requiring any external data corpus. We also demonstrated that because the technique does not rely on data it is free of biases arising because of training dataset. We also showed that the technique is more robust than any data trained technique. For example the performance of our model does not go down if it is dealing with gaussian noise/poisson noise which is not the case with data trained model.

Video Decomposition Prior [195] is a work where we introduce a unified framework where we are able to bring seemingly very unrelated video task such as dehazing, relighting, unsupervised video object segmentation and edits propagation into single framework. A primary contribution in this framework is it does not utilize any

external training corpus but utilizes only the information present in the test video in the form of motion and appearance. We demonstrated state of the art results for all mentioned video tasks.

Continuous Video Process or CVP [74] is a very important work. In this work, we introduce the idea of treating videos as a continuous evolving process given that it is a representation of our continuous real world. The idea is advancement of diffusion [27] methodology which got extremely famous for its success in image generation tasks. We came up with novel forward process and derived the reverse process that maximizes the likelihood of underlying video data distribution. We showed that our methodology not only achieve state-of-the-art results in video prediction tasks but also reduces compute overhead involved during the inference time. In the follow-up work to this which is Continuous video flow work we demonstrate further improvement in generation fidelity, more efficient model and even less compute requirement for video prediction tasks.

While these works cover a wide breadth of generation and enhancement tasks for videos, we in no way touch all existing directions but are positive our insights would be helpful for future research along the directions we explored.

Bibliography

- [1] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.
- [2] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild, 2012.
- [3] Matias Tassano, Julie Delon, and Thomas Veit. Fastdvdnet: Towards real-time deep video denoising without flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [4] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.
- [5] Ruixing Wang, Xiaogang Xu, Chi-Wing Fu, Jiangbo Lu, Bei Yu, and Jiaya Jia. Seeing dynamic scene in the dark: A high-quality video dataset with mechatronic alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9700–9709, 2021.
- [6] Xinyi Zhang, Hang Dong, Jinshan Pan, Chao Zhu, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Fei Wang. Learning to restore hazy video: A new real-world dataset and a new method. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9239–9248, 2021.
- [7] Colin Studholme, David John Hawkes, and Derek LG Hill. Normalized entropy measure for multimodality image alignment. In *Medical imaging 1998: image processing*, volume 3338, pages 132–143. SPIE, 1998.
- [8] Dev Yashpal Sheth, Sreyas Mohan, Joshua L Vincent, Ramon Manzorro, Peter A Crozier, Mitesh M Khapra, Eero P Simoncelli, and Carlos Fernandez-Granda. Unsupervised deep video denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1759–1768, 2021.

- [9] Valéry Dewil, Jérémy Anger, Axel Davy, Thibaud Ehret, Gabriele Facciolo, and Pablo Arias. Self-supervised training for blind multi-frame video denoising. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2724–2734, 2021.
- [10] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5972–5981, 2022.
- [11] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
- [12] Haotian Zhang, Long Mai, Ning Xu, Zhaowen Wang, John Collomosse, and Hailin Jin. An internal learning approach to video inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2720–2729, 2019.
- [13] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf. Flow-edge guided video completion, 2020.
- [14] Hao Ouyang, Tengfei Wang, and Qifeng Chen. Internal video inpainting by implicit long-range propagation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14579–14588, 2021.
- [15] Chongyi Li, Chunle Guo, and Chen Change Loy. Learning to enhance low-light image via zero-reference deep curve estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. doi: 10.1109/TPAMI.2021.3063604.
- [16] Fan Zhang, Yu Li, Shaodi You, and Ying Fu. Learning temporal consistency for low light video enhancement from single images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4967–4976, 2021.
- [17] Vickie Ye, Zhengqi Li, Richard Tucker, Angjoo Kanazawa, and Noah Snavely. Deformable sprites for unsupervised video decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2657–2666, 2022.
- [18] Yanchao Yang, Brian Lai, and Stefano Soatto. Dystab: Unsupervised object segmentation via dynamic-static bootstrapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2826–2836, 2021.
- [19] Hang Dong, Jinshan Pan, Lei Xiang, Zhe Hu, Xinyi Zhang, Fei Wang, and Ming-Hsuan Yang. Multi-scale boosted dehazing network with dense feature

- fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2157–2167, 2020.
- [20] Yosef Gandelsman, Assaf Shocher, and Michal Irani. "double-dip": Unsupervised image decomposition via coupled deep-image-priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11026–11035, 2019.
 - [21] Hala Lamdouar, Charig Yang, Weidi Xie, and Andrew Zisserman. Betrayed by motion: Camouflaged object discovery via motion segmentation. *Asian Conference on Computer Vision*, 2020.
 - [22] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
 - [23] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7177–7188, 2021.
 - [24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
 - [25] Jacques Fize, Gaurav Shrivastava, and Pierre André Ménard. Geodict: an integrated gazetteer. In *Proceedings of Language, Ontology, Terminology and Knowledge Structures Workshop (LOTKS 2017)*, 2017.
 - [26] Mathieu Roche, Maguelonne Teisseire, and Gaurav Shrivastava. Valorcarn-tetis: Terms extracted with biotex. 2017.
 - [27] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.
 - [28] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021.
 - [29] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*, 2021. URL [arXivpreprintarXiv:2011.13456](https://arxiv.org/abs/2011.13456).
 - [30] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *International Conference on Learning Representations*, 2020. URL [arXivpreprintarXiv:2010.02502](https://arxiv.org/abs/2010.02502).
 - [31] Jenny Yuen and Antonio Torralba. A data-driven approach for event prediction. In *European Conference on Computer Vision*, pages 707–720. Springer, 2010.

- [32] Jacob Walker, Abhinav Gupta, and Martial Hebert. Patch to the future: Unsupervised visual prediction. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3302–3309, 2014.
- [33] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML’15, pages 843–852. JMLR.org, 2015. URL <http://dl.acm.org/citation.cfm?id=3045118.3045209>.
- [34] Marc Oliu, Javier Selva, and Sergio Escalera. Folded recurrent neural networks for future video prediction. *CoRR*, abs/1712.00311, 2017. URL <http://arxiv.org/abs/1712.00311>.
- [35] Francesco Cricri, Xingyang Ni, Mikko Honkala, Emre Aksu, and Moncef Gabbouj. Video ladder networks. *CoRR*, abs/1612.01756, 2016. URL <http://arxiv.org/abs/1612.01756>.
- [36] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *CoRR*, abs/1706.08033, 2017. URL <http://arxiv.org/abs/1706.08033>.
- [37] N. Elsayed, A. S. Maida, and M. Bayoumi. Reduced-gate convolutional lstm architecture for next-frame video prediction using predictive coding. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, July 2019. doi: 10.1109/IJCNN.2019.8852480.
- [38] Ruben Villegas, Arkanath Pathak, Harini Kannan, Dumitru Erhan, Quoc V. Le, and Honglak Lee. High fidelity video prediction with large stochastic recurrent neural networks, 2019.
- [39] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3d LSTM: A model for video prediction and beyond. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=B1lKS2AqtX>.
- [40] Lluís Castrejón, Nicolas Ballas, and Aaron C. Courville. Improved conditional vrns for video prediction. *CoRR*, abs/1904.12165, 2019. URL <http://arxiv.org/abs/1904.12165>.
- [41] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.
- [42] Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

- [43] Aidan Clark, Jeff Donahue, and Karen Simonyan. Adversarial video generation on complex datasets. *arXiv preprint arXiv:1907.06571*, 2019.
- [44] 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. *arXiv preprint arXiv:2003.04035*, 2020.
- [45] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.
- [46] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior, 2018.
- [47] Lluís Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnnns for video prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7608–7617, 2019.
- [48] Gaurav Shrivastava and Abhinav Shrivastava. Diverse video generation using a gaussian process trigger. *arXiv preprint arXiv:2107.04619*, 2021.
- [49] Sarthak Bhagat, Shagun Uppal, Zhuyun Yin, and Nengli Lim. Disentangling multiple features in video sequences using gaussian processes in variational autoencoders, 2020.
- [50] Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. *Advances in Neural Information Processing Systems*, 35:23371–23385, 2022.
- [51] Aram Davtyan, Sepehr Sameni, and Paolo Favaro. Efficient video prediction via sparsely conditioned flow matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23263–23274, 2023.
- [52] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. 2022.
- [53] Tobias Höppe, Arash Mehrjou, Stefan Bauer, Didrik Nielsen, and Andrea Dittadi. Diffusion models for video prediction and infilling, 2022.
- [54] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *International Conference on Computer Vision*, 2017.
- [55] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to generate long-term future via hierarchical prediction, 2017.
- [56] Nevan Wichers, Ruben Villegas, Dumitru Erhan, and Honglak Lee. Hierarchical long-term video prediction without supervision, 2018.

- [57] Haoye Cai, Chunyan Bai, Yu-Wing Tai, and Chi-Keung Tang. Deep video generation, prediction and completion of human action sequences. *Lecture Notes in Computer Science*, page 374–390, 2018. ISSN 1611-3349.
- [58] Navaneeth Bodla, Gaurav Shrivastava, Rama Chellappa, and Abhinav Shrivastava. Hierarchical video prediction using relational layouts for human-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12146–12155, 2021.
- [59] Mauricio Delbracio and Peyman Milanfar. Inversion by direct iteration: An alternative to denoising diffusion for image restoration. *arXiv preprint arXiv:2303.11435*, 2023.
- [60] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. *arXiv preprint arXiv:2208.09392*, 2022.
- [61] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 32–36 Vol.3, Aug 2004. doi: 10.1109/ICPR.2004.1334462.
- [62] Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections, 2017.
- [63] Cristian Sminchisescu Catalin Ionescu, Fuxin Li. Latent structured models for human pose estimation. In *International Conference on Computer Vision*, 2011.
- [64] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric and challenges, 2018.
- [65] Adil Kaan Akan, Erkut Erdem, Aykut Erdem, and Fatma Güney. Slamp: Stochastic latent appearance and motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14728–14737, 2021.
- [66] Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari. Stochastic latent residual video prediction. In *International Conference on Machine Learning*, pages 3233–3246. PMLR, 2020.
- [67] Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin P Murphy, and Honglak Lee. Unsupervised learning of object structure and dynamics from videos. *Advances in Neural Information Processing Systems*, 32, 2019.

- [68] Xiaojie Gao, Yueming Jin, Qi Dou, Chi-Wing Fu, and Pheng-Ann Heng. Accurate grid keypoint learning for efficient video prediction. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5908–5915. IEEE, 2021.
- [69] Ruslan Rakhimov, Denis Volkhonskiy, Alexey Artemov, Denis Zorin, and Evgeny Burnaev. Latent video transformer. *arXiv preprint arXiv:2006.10704*, 2020.
- [70] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021.
- [71] Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. Ccvs: context-aware controllable video synthesis. *Advances in Neural Information Processing Systems*, 34:14042–14055, 2021.
- [72] 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.
- [73] Jian Liang, Chenfei Wu, Xiaowei Hu, Zhe Gan, Jianfeng Wang, Lijuan Wang, Zicheng Liu, Yuejian Fang, and Nan Duan. Nuwa-infinity: Autoregressive over autoregressive generation for infinite visual synthesis. *Advances in Neural Information Processing Systems*, 35:15420–15432, 2022.
- [74] Gaurav Shrivastava and Abhinav Shrivastava. Video prediction by modeling videos as continuous multi-dimensional processes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7236–7245, 2024.
- [75] Gaurav Shrivastava and Abhinav Shrivastava. Efficient continuous video flow model for video prediction. In *Tech Report*, pages 1–8, 2025.
- [76] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.
- [77] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow, 2022. URL <https://arxiv.org/abs/2209.03003>.
- [78] Gaurav Shrivastava, Ser-Nam Lim, and Abhinav Shrivastava. Video dynamics prior: An internal learning approach for robust video enhancements. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=CCq73CGMyV>.

- [79] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.
- [80] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction, 2017.
- [81] Hang Gao, Huazhe Xu, Qi-Zhi Cai, Ruth Wang, Fisher Yu, and Trevor Darrell. Disentangling propagation and generation for video prediction. *ArXiv*, abs/1812.00452, 2018.
- [82] Carl Edward Rasmussen. Gaussian processes for machine learning. MIT Press, 2006.
- [83] Carl Vondrick and Antonio Torralba. Generating the future with adversarial transformers. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2992–3000, 2017.
- [84] Chaochao Lu, Michael Hirsch, and Bernhard Schölkopf. Flexible spatio-temporal networks for video prediction. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2137–2145, 2017.
- [85] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. *ArXiv*, abs/1609.02612, 2016.
- [86] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. doi: 10.1109/iccv.2017.308. URL <http://dx.doi.org/10.1109/ICCV.2017.308>.
- [87] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. doi: 10.1109/cvpr.2018.00165. URL <http://dx.doi.org/10.1109/CVPR.2018.00165>.
- [88] Zhihang Hu and Jason Wang. A novel adversarial inference framework for video prediction with action control. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.
- [89] M. P. Sampat, Z. Wang, S. Gupta, A. C. Bovik, and M. K. Markey. Complex wavelet structural similarity: A new image similarity index. *IEEE Transactions on Image Processing*, 18(11):2385–2401, Nov 2009. doi: 10.1109/TIP.2009.2025923.
- [90] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*,

- Jun 2018. doi: 10.1109/cvpr.2018.00068. URL <http://dx.doi.org/10.1109/CVPR.2018.00068>.
- [91] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks, 2016.
 - [92] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *Lecture Notes in Computer Science*, page 694–711, 2016. ISSN 1611-3349.
 - [93] Tianfan Xue, Jiajun Wu, Katherine L Bouman, and William T Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances In Neural Information Processing Systems*, 2016.
 - [94] Apratim Bhattacharyya, Bernt Schiele, and Mario Fritz. Accurate and diverse sampling of sequences based on a “best of many” sample objective. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. doi: 10.1109/cvpr.2018.00885. URL <http://dx.doi.org/10.1109/CVPR.2018.00885>.
 - [95] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.
 - [96] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2016.
 - [97] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A. Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation, 2017.
 - [98] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error, 2015.
 - [99] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008. doi: 10.1109/TPAMI.2007.1167.
 - [100] J. Hardy, F. Havlak, and M. Campbell. Multiple-step prediction using a two stage gaussian process model. In *2014 American Control Conference*, pages 3443–3449, 2014. doi: 10.1109/ACC.2014.6859020.
 - [101] K. Moon and V. Pavlovic. 3d human motion tracking using dynamic probabilistic latent semantic analysis. In *2008 Canadian Conference on Computer and Robot Vision*, pages 155–162, 2008. doi: 10.1109/CRV.2008.45.

- [102] Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 567–574, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR.
- [103] James Hensman, Nicolo Fusi, and Neil D. Lawrence. Gaussian processes for big data, 2013.
- [104] Zhenwen Dai, Andreas Damianou, James Hensman, and Neil Lawrence. Gaussian process models with parallelization and gpu acceleration, 2014.
- [105] Yarin Gal, Mark van der Wilk, and Carl E. Rasmussen. Distributed variational inference in sparse gaussian process regression and latent variable models, 2014.
- [106] Andrew Gordon Wilson and Hannes Nickisch. Kernel interpolation for scalable structured gaussian processes (kiss-gp), 2015.
- [107] Andrew Gordon Wilson, Christoph Dann, and Hannes Nickisch. Thoughts on massively scalable gaussian processes, 2015.
- [108] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Stochastic variational deep kernel learning, 2016.
- [109] Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, 2018.
- [110] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.
- [111] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.
- [112] Manoj Kumar, Mohammad Babaeizadeh, Dumitru Erhan, Chelsea Finn, Sergey Levine, Laurent Dinh, and Durk Kingma. Videoflow: A conditional flow-based model for stochastic video generation, 2019.
- [113] Maruan Al-Shedivat, Andrew Gordon Wilson, Yunus Saatchi, Zhiting Hu, and Eric P. Xing. Learning scalable deep kernels with recurrent structure, 2016.
- [114] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision (IJCV)*, 127(8):1106–1125, 2019.

- [115] Matias Tassano, Julie Delon, and Thomas Veit. Dvdnet: A fast network for deep video denoising. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1805–1809. IEEE, 2019.
- [116] Pablo Arias and Jean-Michel Morel. Video denoising via empirical bayesian estimation of space-time patches. *Journal of Mathematical Imaging and Vision*, 60(1):70–93, 2018.
- [117] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Recurrent back-projection network for video super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3897–3906, 2019.
- [118] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3224–3232, 2018.
- [119] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018.
- [120] Yonatan Wexler, Eli Shechtman, and Michal Irani. Space-time completion of video. *IEEE Transactions on pattern analysis and machine intelligence*, 29(3): 463–476, 2007.
- [121] Muhammad Haris, Greg Shakhnarovich, and Norimichi Ukita. Space-time-aware multi-resolution video enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [122] Gaurav Shrivastava. *Diverse Video Generation*. PhD thesis, University of Maryland, College Park, 2021.
- [123] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [124] Chenyang Lei, Yazhou Xing, and Qifeng Chen. Blind video temporal consistency via deep video prior. In *Advances in Neural Information Processing Systems*, 2020.
- [125] Oded Shahrar, Alon Faktor, and Michal Irani. *Space-time super-resolution from a single video*. IEEE, 2011.
- [126] Richard R Schultz and Robert L Stevenson. Extraction of high-resolution frames from video sequences. *IEEE transactions on image processing*, 5(6): 996–1011, 1996.

- [127] Michal Irani and Shmuel Peleg. *Image sequence enhancement using multiple motions analysis*. Hebrew University of Jerusalem. Leibniz Center for Research in Computer ..., 1991.
- [128] T. Wiegand, E. Steinbach, and B. Girod. Affine multipicture motion-compensated prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(2):197–209, 2005. doi: 10.1109/TCSVT.2004.841690.
- [129] Assaf Shocher, Nadav Cohen, and Michal Irani. “zero-shot” super-resolution using deep internal learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3118–3126, 2018.
- [130] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.
- [131] Sefi Bell-Kligler, Assaf Shocher, and Michal Irani. Blind super-resolution kernel estimation using an internal-gan. *Advances in Neural Information Processing Systems*, 32, 2019.
- [132] Dongwei Ren, Kai Zhang, Qilong Wang, Qinghua Hu, and Wangmeng Zuo. Neural blind deconvolution using deep priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3341–3350, 2020.
- [133] Assaf Shocher, Shai Bagon, Phillip Isola, and Michal Irani. Ingan: Capturing and retargeting the "dna" of a natural image. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019.
- [134] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Computer Vision (ICCV), IEEE International Conference on*, 2019.
- [135] Yael Vinker, Eliahu Horwitz, Nir Zabari, and Yedid Hoshen. Deep single image manipulation. *arXiv preprint arXiv:2007.01289*, 2020.
- [136] Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising. *Advances in Neural Information Processing Systems*, 32:6970–6980, 2019.
- [137] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.
- [138] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Deep video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5792–5801, 2019.

- [139] Liad Pollak Zuckerman, Eyal Naor, George Pisha, Shai Bagon, and Michal Irani. Across scales and across dimensions: Temporal super-resolution using deep internal learning. In *European Conference on Computer Vision*, pages 52–68. Springer, 2020.
- [140] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.
- [141] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4161–4170, 2017.
- [142] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.
- [143] Tae Hyun Kim, Mehdi S. M. Sajjadi, Michael Hirsch, and Bernhard Scholkopf. Spatio-temporal transformer network for video restoration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [144] Mengyu Chu, You Xie, Laura Leal-Taixé, and Nils Thuerey. Temporally coherent gans for video super-resolution (tecogan). *arXiv preprint arXiv:1811.09393*, 1(2):3, 2018.
- [145] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [146] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slo-mo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, 2018.
- [147] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.
- [148] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5437–5446, 2020.
- [149] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020.

- [150] Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [151] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.
- [152] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4): 259–268, 1992.
- [153] MTCAJ Thomas and A Thomas Joy. *Elements of information theory*. Wiley-Interscience, 2006.
- [154] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2019.
- [155] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
- [156] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [157] Gaurav Shrivastava, Ser-Nam Lim, and Abhinav Shrivastava. Video dynamics prior: An internal learning approach for robust video enhancements. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [158] Erika Lu, Forrester Cole, Tali Dekel, Weidi Xie, Andrew Zisserman, David Salesin, William T Freeman, and Michael Rubinstein. Layered neural rendering for retiming people in video. *arXiv preprint arXiv:2009.07833*, 2020.
- [159] Xiankai Lu, Wenguan Wang, Jianbing Shen, David Crandall, and Jiebo Luo. Zero-shot video object segmentation with co-attention siamese networks. *IEEE transactions on pattern analysis and machine intelligence*, 44(4):2228–2242, 2020.
- [160] Erika Lu, Forrester Cole, Tali Dekel, Andrew Zisserman, William T Freeman, and Michael Rubinstein. Omnimate: Associating objects and their effects in video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4507–4515, 2021.

- [161] Yoni Kasten, Dolev Ofri, Oliver Wang, and Tali Dekel. Layered neural atlases for consistent video editing. *ACM Transactions on Graphics (TOG)*, 40(6): 1–12, 2021.
- [162] John YA Wang and Edward H Adelson. Representing moving images with layers. *IEEE transactions on image processing*, 3(5):625–638, 1994.
- [163] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 231–242, 1998.
- [164] Eli Shechtman, Yaron Caspi, and Michal Irani. Increasing space-time resolution in video. In *Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part I* 7, pages 753–768. Springer, 2002.
- [165] Eli Shechtman, Yaron Caspi, and Michal Irani. Space-time super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):531–545, 2005.
- [166] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2341–2353, 2010.
- [167] Michal Irani Assaf Shocher, Nadav Cohen. "zero-shot" super-resolution using deep internal learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [168] Liad Pollak Zuckerman, Eyal Naor, George Pisha, Shai Bagon, and Michal Irani. Across scales and across dimensions: Temporal super-resolution using deep internal learning. In *European Conference on Computer Vision (ECCV)*. Springer, 2020.
- [169] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *European Conference on Computer Vision*, pages 707–723. Springer, 2022.
- [170] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- [171] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

- [172] Pia Bideau and Erik Learned-Miller. It’s moving! a probabilistic model for causal motion segmentation in moving camera videos. In *European Conference on Computer Vision*, pages 433–449. Springer, 2016.
- [173] Thomas Brox and Jitendra Malik. Object segmentation by long term analysis of point trajectories. In *European conference on computer vision*, pages 282–295. Springer, 2010.
- [174] Ioana Croitoru, Simion-Vlad Bogolin, and Marius Leordeanu. Unsupervised learning from video to detect foreground objects in single images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4335–4343, 2017.
- [175] Alon Faktor and Michal Irani. Video segmentation by non-local consensus voting. In *BMVC*, number 7, page 8, 2014.
- [176] Margret Keuper, Bjoern Andres, and Thomas Brox. Motion trajectory segmentation via minimum cost multicut. In *Proceedings of the IEEE international conference on computer vision*, pages 3271–3279, 2015.
- [177] Peter Ochs and Thomas Brox. Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *2011 international conference on computer vision*, pages 1583–1590. IEEE, 2011.
- [178] Anestis Papazoglou and Vittorio Ferrari. Fast object segmentation in unconstrained video. In *Proceedings of the IEEE international conference on computer vision*, pages 1777–1784, 2013.
- [179] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12240–12249, 2019.
- [180] Josef Sivic, Frederik Schaffalitzky, and Andrew Zisserman. Object level grouping for video shots. *International Journal of Computer Vision*, 67(2):189–210, 2006.
- [181] T.Brox and J.Malik. Object segmentation by long term analysis of point trajectories. In *European Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science. Springer, Sept. 2010. URL <http://lmb.informatik.uni-freiburg.de/Publications/2010/Bro10c>.
- [182] Yeong Jun Koh and Chang-Su Kim. Primary object segmentation in videos based on region augmentation and reduction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7417–7425. IEEE, 2017.

- [183] Yinlin Hu, Yunsong Li, and Rui Song. Robust interpolation of correspondences for large displacement optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 481–489, 2017.
- [184] Yanchao Yang, Antonio Loquercio, Davide Scaramuzza, and Stefano Soatto. Unsupervised moving object detection via contextual information separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 879–888, 2019.
- [185] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018.
- [186] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. *Advances in Neural Information Processing Systems*, 33:7198–7211, 2020.
- [187] Raanan Fattal. Single image dehazing. *ACM transactions on graphics (TOG)*, 27(3):1–9, 2008.
- [188] S.G. Narasimhan and S.K. Nayar. Chromatic framework for vision in bad weather. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 598–605 vol.1, 2000. doi: 10.1109/CVPR.2000.855874.
- [189] Yuval Bahat and Michal Irani. Blind dehazing using internal patch recurrence. In *2016 IEEE International Conference on Computational Photography (ICCP)*, pages 1–9. IEEE, 2016.
- [190] Zhao Yang, Qiang Wang, Luca Bertinetto, Weiming Hu, Song Bai, and Philip HS Torr. Anchor diffusion for unsupervised video object segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 931–940, 2019.
- [191] Muhammad Faisal, Ijaz Akhter, Mohsen Ali, and Richard Hartley. Exploiting geometric constraints on dense trajectories for motion saliency. *arXiv preprint arXiv:1909.13258*, 3(4), 2019.
- [192] Hongmei Song, Wenguan Wang, Sanyuan Zhao, Jianbing Shen, and Kin-Man Lam. Pyramid dilated deeper convlstm for video salient object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 715–731, 2018.
- [193] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning motion patterns in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3386–3394, 2017.

- [194] Tianfei Zhou, Shunzhou Wang, Yi Zhou, Yazhou Yao, Jianwu Li, and Ling Shao. Motion-attentive transition for zero-shot video object segmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13066–13073, 2020.
- [195] Gaurav Shrivastava, Ser-Nam Lim, and Abhinav Shrivastava. Video decomposition prior: Editing videos layer by layer. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=nfMyERXNru>.