UNIVERSITY OF CALIFORNIA, MERCED

Multi-frame Video Prediction with Learnable Temporal Motion Encodings

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Electrical Engineering and Computer Science

by

Rakesh Jasti

Committee in charge:

Professor Ming-Hsuan Yang, Chair
Professor Shawn Newsam
Professor Sungjin Im

2020

The thesis of Rakesh Jasti is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Professor Shawn Newsam

Professor Sungjin Im

Professor Ming-Hsuan Yang Chair

University of California, Merced

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I want to express my sincere gratitude to my supervisor, Prof. Ming-Hsuan Yang[1], for giving me the opportunity to work with him and being a great supervisor in general. I would also like to thank Varun Jampani[2] and Deqing Sun[3] for guiding and enabling me to work on this insanely cool project. I strive to be as competent at research as they are one day. Lastly, I would like to thank my friends and family for their support.

[1] Ming-Hsuan Yang's Department Homepage – https://faculty.ucmerced.edu/mhyang/
[2] Varun Jampani's Homepage – https://varunjampani.github.io/
[3] Deqing Sun's Homepage – https://deqings.github.io/

VITA

2015	B. Tech. in Pulp and Paper Technology, Indian Institute of Technology, Roorkee, India
2020	M. S. in Electrical Engineering and Computer Science, University of California, Merced

ABSTRACT OF THE THESIS

Multi-frame Video Prediction with Learnable Temporal Motion Encodings

by

Rakesh Jasti
Master of Science in Electrical Engineering and Computer Science

University of California Merced, 2020

Professor Ming-Hsuan Yang, Chair

While recent deep learning methods have made significant progress on the video prediction problem, most methods predict the immediate or a fixed number of future frames. To obtain longer-term frame predictions, existing techniques usually process the predicted frames iteratively, resulting in blurry or inconsistent predictions. In this thesis, we present a new approach that can predict an *arbitrary* number of future video frames with a single forward pass through the network. Instead of directly predicting a fixed number of future optical flows or frames, we learn temporal motion encodings, *i.e.*, temporal motion basis vectors and a network to predict the coefficients. The learned motion basis can be easily extended to arbitrary length at inference time, enabling us to predict an arbitrary number of future frames. Experiments on benchmark datasets indicate that our approach performs favorably against state-of-the-art techniques even for the next frame prediction setting. When evaluated under 5-frame or 10-frame prediction settings, the proposed method obtains bigger performance gains over the existing state-of-the-art techniques that iteratively process the predictions.

Chapter 1

Introduction

Video prediction refers to predicting future video frames by observing a sequence of video frames. Many real-world applications need the prediction of future frames conditioned on a given sequence of input video frames, such as online streaming, predicting system behavior [62, 26, 18], autonomous driving and intelligent agents [20], data augmentation [63] and video coding. For example, self-driving cars have to predict the motion of the passing vehicles. This prediction capability is vital for the autonomous systems in path planning and interacting with humans.

In most use cases, it is desirable to have video prediction models that can generate high-resolution future frames for multiple future time-steps. This multiple-step video prediction problem is highly challenging because the future states of a scene are uncertain and a scene may have complex spatio-temporal dynamics due to camera motion, lighting conditions, clutter, object deformations, occlusion, moving objects, etc.

Given the recent success of deep neural networks for understanding pixel motion (optical flow) [43, 49] and also in generating realistic image content [14], there has been a plethora of recent works [39] that leverage deep neural networks for video prediction. Deep neural networks are well suited for this problem because of their ability to learn adequate representations from high-dimensional data and increase their capacity with the size of data available with large-scale optimization algorithms. Deep learning based models fit perfectly into the learning by prediction paradigm, enabling the extraction of meaningful spatio-temporal correlations from video data in a self-supervised fashion.

However, existing methods mainly focus on predictions in a short time horizon,

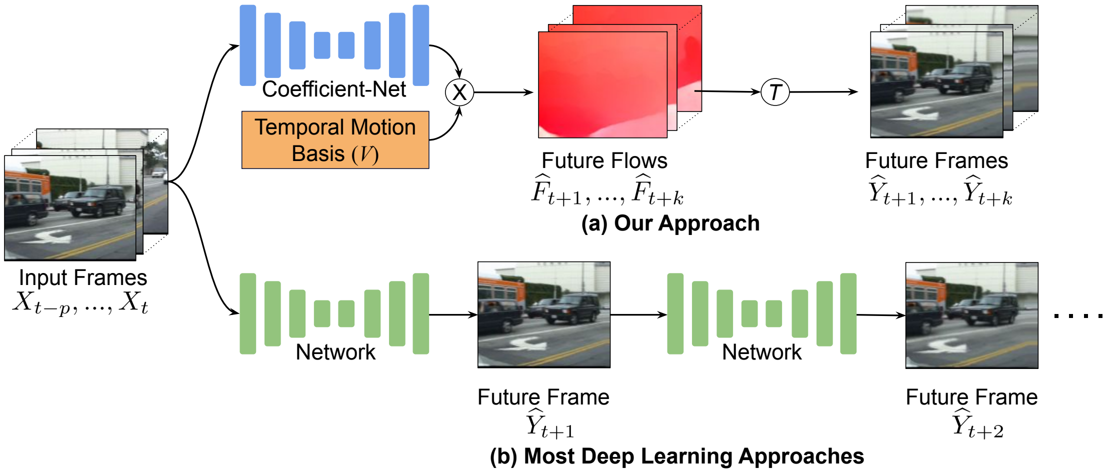

Figure 1.1: **Arbitrary-length multi-frame prediction.** Our approach (top) can predict an arbitrary number of future optical flows and synthesize corresponding future frames with a single forward pass through the network. This is in contrast to most existing deep learning works (bottom) that iteratively input the predicted frames into the network for multi-frame video prediction.

usually a single time step. As illustrated in Fig. 1.1, most existing deep learning approaches [34, 13, 42, 29] for video prediction achieve multi-frame video prediction by iteratively passing the previously predicted frames as observations into the network. Recursive future prediction strategies usually produce high quality predictions for the first few steps. But the prediction would then dramatically degrade, and could even lead to a complete miss of the video context or generating a stationary frame. Such recursive future prediction strategies usually also have more memory and runtime footprints making them less suitable for real-world applications. Besides, iteratively predicting one frame at a time and reusing those predictions as input to the network can result in error accumulation over time.

Standard convolutional neural networks (CNN) are the basic building blocks of deep learning architectures designed for visual reasoning since the Convolutional Neural Networks (CNNs) efficiently model the spatial structure of images.

Convolutional operations account for short-range intra-frame dependencies due to their limited receptive fields, determined by the kernel size. This is a well-addressed issue, that many authors circumvented by (1) stacking more convolutional layers [23], (2) increasing the kernel size (prohibitively expensive), (3) by linearly combining multiple scales [34] as in the reconstruction process of a Laplacian pyramid [8], (4) using dilated convolutions to capture long-range spatial dependencies [60], (5) enlarging the receptive fields [5] or subsampling, i.e. using pooling operations in exchange for losing resolution. The latter could be mitigated by using residual connections [16, 52], to preserve resolution while increasing the number of stacking convolutions. Vanilla CNNs lack of explicit inter-frame modeling capabilities. To properly model inter-frame variability in a video sequence, 3D convolutions come into play as a promising alternative to recurrent modeling. Several video prediction approaches leveraged 3D convolutions to capture temporal consistency [1, 55, 54, 51]. However even after addressing these limitations, CNNs still generate fixed-size results and cannot be readily used to predict an arbitrary number of multiple future video frames.

Recurrent models were conceptually designed to model the spatio-temporal representation of sequential data such as video frames. Among other sequence learning tasks, such as machine translation, speech recognition and video captioning, to name a few, Recurrent Neural Networks (RNNs) [45] demonstrated success to some degree in the video prediction scenario [50, 59, 31]. However, RNNs have some important limitations when dealing with long-term representations due to the vanishing and exploding gradient issues, making the training hard and requiring higher memory requirements.

In this thesis, we propose a deep network architecture for simultaneously predicting *multiple* and an *arbitrary* number of future video frames in a single forward pass through the network. As illustrated in Fig. 1.1, instead of directly predicting future frames or optical flows with a CNN, in this work, we propose to predict basis coefficients for a learnable temporal motion basis. These basis coefficients together with learned temporal motion basis can be used to generate an arbitrary number of future optical flows, which in turn are used to produce multiple future video frames.

Specifically, we use a 3D convolutional network (Coefficient-Net) that can capture the spatio-temporal features of the given video frames, to predict basis coefficients of the learned temporal basis vectors at each pixel. These basis vectors model long term per-pixel motion across video frames and are trained together with Coefficient-Net via standard gradient descent techniques. We parameterize our temporal motion basis vectors following linear time-invariant dynamical systems theory following the recent work of DYAN [29] and then learn these basis parameters.

We evaluate our video prediction approach on two standard datasets of the CALTECH PEDESTRIAN dataset [10] and consumer videos from the UCF-101 dataset [47]. Our method achieves state-of-the-art results on both datasets while being able to predict an arbitrary number of multiple future frames. Experiments demonstrate that the proposed algorithm can boost the visual quality of generated videos and lead to more precise results in long-term prediction compared to the prior works.

In brief, our video prediction technique has the following favorable properties:

- **Multiple and an arbitrary number of future frame predictions.** With a single forward pass through our coefficient network, we can predict multiple and arbitrary numbers of future video frames from a given video.

- **Learned temporal motion basis.** Along with the main network, we jointly learn temporal motion basis vectors that model the per-pixel motion across video frames.

- **End-to-end trainable.** Both the coefficient network and temporal motion basis are learned in an end-to-end fashion enabling the use of large-scale training data that is readily available for the video prediction task.

- **State-of-the-art performance.** Experiments on two benchmarks datasets [10, 47] indicate the state-of-the-art prediction results using our technique.

The thesis is organized as follows. First, Chapter 2 discusses related previous work. In Chapter 3, we present our approach and discuss the details of our implementation. In Chapter 4, we report experiments comparing our performance in multi-step frame prediction against the state-of-art approaches. Additionally, we perform multiple abltation studies to investigate the significance of various details and modules used in our approach. Finally, Chapter 5 provides concluding remarks and directions for future applications of our work.

Chapter 2

Related Work

Multiple deep learning based methods for future video prediction have emerged recently. Initial models focused on directly predicting pixel values by modeling the scene dynamics. However, since the pixel space is high dimensional, extracting meaningful and robust features from raw videos is challenging. As a natural next step, the subsequent works focused on reducing the dimensionality of the feature space and the supervision effort. To disentangle the factors of variation from the visual content and factorize the prediction space, authors proposed: (1) modelling the source of variability as transformations between frames (2) using a two stream computation to model the visual and motion content separately. Other works have narrowed the prediction space by conditioning the predictions on extra variables such as vehicle odometry or robot state, or reformulating the problem in a higher-level space such as semantic and instance segmentation, and human pose. In the following sections, we will delve into the recent progress on the application of deep learning techniques for future video prediction given the contextual data of a sequence of input frames. We review the recent methods for direct pixel and latent representation prediction. We provide a comprehensive description as well as an analysis of their strengths and weaknesses. Additionally, we also discuss the recent methods that specifically address the challenges in video prediction for a long time horizons.

2.1 Prediction in the Raw Pixel Space

Early neural network approaches focus on predicting raw pixels conditioned on the observed frames [48, 6, 4, 34, 24]. Building on the recent success of the Laplacian Generative Adversarial Network (LAPGAN) [8], Mathieu *et al.* [34] propose using GANs [14] with a multi-scale approach and image gradient loss for video prediction that was trained in a adversarial setting. They employ novel GDL regularization with ℓ_1-based reconstruction and adversarial training. Significant improvements over the previous state-of-the-art models [48] are reported by the authors in terms of prediction sharpness. However, the results suffer from blurriness artifacts due to the difficulty in directly regressing raw pixel values from the input frames. Lotter *et al.* [32] outperformed the previous works [48, 34] with the Predictive Coding Network (PredNet). PredNet learns feature representations by stacking convolutional LSTMs vertically. Each ConvLSTM layer produces a layer-specific prediction at every time step to transmit the local $\ell_1$error term to the next layer.

Under the assumption that video sequences are symmetric in time, Kwon and Park [27] explore a retrospective prediction scheme by training a generator that predicts both the forward and backward frames (reversing the input sequence to predict the past). Their cycle GAN-based approach ensure the consistency of bi-directional prediction through retrospective prediction scheme. They employ distinct discriminators to enforce the supervision on the frame content generation and on the temporal sequence consistency. In the same spirit, other works focused on both, forward and backward predictions [19, 36].

For modeling short-term features, Wang *et al.* [55] integrated 3D convolutions into a recurrent network demonstrating favorable results in both video prediction and early activity recognition. While 3D convolutions efficiently preserves local dynamics, RNNs enables long range video reasoning. The Eidetic 3d LSTM (E3d-LSTM) network features a gated-controlled self-attention module, i.e. eidetic 3D memory, that effectively manages historical memory records across multiple time steps. Outperforming previous works, Yu *et al.* proposed the Conditionally Reversible Network (CrevNet) [61] consisting of two modules, an invertible Auto-Encoder and a Reversible Predictive Model (RPM). While the bijective two-way Auto-Encoder ensures no information loss and reduces the memory consumption, the RPM extends the reversibility from spatial to temporal domain. Some other works used 3D convolutional operations to model the time dimension.

Extracting a robust representation from raw pixel values is an overly complicated task due to the high-dimensionality of the pixel space and several challending factors like camera motion, dynamically moving objects, occlusions, etc. Besides, iteratively applying direct pixel-prediction methods for multiple frame predictions exacerbates the artifacts and causes an exponential growth in the prediction error on the long-term horizon (Fig. 1.1).

2.2 Predictions through Latent Factors

More recently, several approaches focus on learning the underlying factors of variations like optical flow [29, 42, 17], transformation kernels [12], human pose vectors [53], etc. These methods assume that the visual information is already available in the input sequence. These methods are highly efficient in dealing with the redundancy of pixel information in the input sequence of frames. Let $X = (X_{t-n}, ..., X_{t-1}, X_t)$ be a video sequence of n frames, where t denotes time. Formally, these methods can be defined as:

$$\hat{Y}_{t+1} = \mathcal{T}(\mathcal{G}(X_{t-n:t}), X_{t-n:t}) \tag{2.1}$$

where $\mathcal{T}$ is the transformation function operating on the parameters generated by the learned function $\mathcal{G}$. The transformation $\mathcal{T}$ is applied to the last observed frame X_t to generate the future frame predictions $\hat{Y}_{t+1}$. The function $\mathcal{T}$ can be a kernel based resampling, or a vector based resampling, or a combination of both.

Kernel-based Resampling. Kernel based resampling methods synthesize pixels by convolving input patches with a predicted kernel. Such methods can be represented as follows:

$$\hat{Y}_{t+1} = \mathcal{K}(x, y) * P_t(x, y) \tag{2.2}$$

where $\mathcal{K}(x, y) \in R^{N \times N}$ is the kernel predicted by $\mathcal{G}$ at (x, y) and $P_t(x, y)$ is an $N \times N$ patch centered at (x, y) in X_t. Finn *et al.* [12] propose two kernel based motion prediction modules outperforming previous approaches [34], (1) the Dynamic Neural Advection (DNA) module predicting different distributions for each pixel and (2) the

CDNA module that instead of predicting different distributions for each pixel, it predicts multiple discrete distributions applied convolutionally to the input image via convolution. While the DNA module generates per-pixel motion, CDNA masks out the objects moving in consistent directions. Inspired by these modules, several works have been since introduced. Using adversarial training, Vondrick *et al*. proposed a cGAN-based model [54] consisting of a discriminator and a CNN generator featuring a transformer module inspired on the CDNA model. Different from the CDNA model, transformations are not applied recurrently on a per-frame basis. To deal with in-the-wild videos and make predictions invariant to camera motion, authors stabilized the input videos.

Vector-based Resampling. As an alternative, vector based resampling methods synthesize future frames in the affine transformation space. Jaderberg *et al*. [22] propose the ST module to regress the different affine transformation parameters for each input, to be applied as a single transformation to the whole feature map(s) or image(s). The ST module can be incorporated at any part of the CNNs and it is fully differentiable. Inspired by the ST module, Liu *et al*. [30] propose the Deep Voxel Flow (DVF) architecture. It consists of a multi-scale flow-based Encoder-Decoder model originally designed for the video frame interpolation task, but was also evaluated on a video prediction tasks. Liang *et al*. [28] use a flow-warping layer based on a bilinear interpolation. In addition to the DNA and CDNA modules, Finn *et al*. [12] propose the Spatial Transformer Predictor (STP) motion-based model producing 2D affine transformations for bilinear sampling.

Hybrid Resampling. Since vector based methods consider few pixels in synthesis, their results often appear degraded by speckled noise patterns. Vector based methods can, however, model large displacements without a significant increase in parameter count. On the other hand, kernel based methods produces visually pleasing results for small displacements, but require large kernels to be predicted at each location to capture large motions. As such, the kernel-based approach can easily become not only costly at inference, but also difficult to train. Combining kernel and vector-based resampling into a hybrid solution, Reda *et al*. [42] proposed the Spatially Displaced Convolution (SDC) module that synthesizes high resolution images applying a learned per-pixel motion vector and kernel at a displaced location in the source image. Their 3D CNN model trained on synthetic data and featuring the SDC modules, reported promising predictions of a high-fidelity.

Two-stream Video Prediction. One popular hypothesis is that a video sequence can be factorized into content and motion, and these can be processed on separate pathways. The Motion-content Network (MCnet) [52] predicts the next frame by combining the predicted motion feature and the extracted content feature. However, MCnet cannot capture the motion information in the long-term prediction. In a similar fashion, DR-NET [9] designed an adversarial training strategy to disentangle the motion and content representations. Recently, Gao *et al*. [13] improve the sharpness of predictions by reducing the occlusion effects of using optical flow. They compute occlusion maps using the flow information and inpaint the occluded regions in the predictions.

Stochastic Variational Video Prediction. Another popular hypothesis is that the outcome of an event is stochastic. As the vast majority of video prediction models are deterministic, they are unable to manage probabilistic environments. To address this issue, several authors proposed modeling future uncertainty with probabilistic models. Babaeizadeh *et al*. [2] proposed the Stochastic Variational Video Prediction (SV2P) by incorporating latent variables into the deterministic CDNA architecture. The time-invariant posterior distribution is the encoding of the entire video sequence via a feed forward convolutional network. Denton *et al*. propose the SVG network [7] by combining a deterministic architecture with stochastic latent variables. Different from SV2P, it outputs a different posterior distribution for every time step whose parameters were estimated during training. The SVG model is easier to train and reported sharper predictions in contrast to [2].

2.3 Long-Term Prediction

While most of the recent works focus on predicting the immediate next frame, our work addresses generating frame predictions for an arbitrary number of time-steps simultaneously in a single forward pass of a neural network. Recent works [38, 53, 57] exhibit long-term frame predictions.

Oh *et al*. [38] first propose long-term video predictions conditioned by control inputs from Atari games. Although the proposed Encoder-Decoder based models reported very long-term predictions (+100), performance drops when dealing with small objects and results in blurry predictions due to the squared error when generating the predictions for the synthetic videos. Villegas *et al*. [53] regress future frames through supervised prediction of human poses and analogy formulation. The authors compare the model against [59, 34] and report long-term results. To make the model unsupervised on the human pose, Wichers *et al*. [57] adopt different training strategies: end-to-end prediction minimizing the ℓ_2 loss, and through analogy formulation, constraining the predicted features to be close to the outputs of the future encoder. However, for each new time-step, the discussed models update the recurrent decoder states to obtain the next frame prediction.

Recently, the DYAN framework by Liu *et al*. [29] models a sequence of optical flows using a temporal motion basis and the corresponding set of basis coefficients. For a given input video, the basis coefficients are computed using FISTA [3], an iterative optimization algorithm, with the objective of minimizing the frame reconstruction loss. Instead of using an optimization scheme, we learn a deep network that can predict basis coefficients while also learning the temporal motion basis using standard back-propagation techniques. This enables us to leverage large amounts of training data resulting in better performance.

Chapter 3

Approach

Video prediction aims to synthesize the future frames in a video sequence conditioned on the input sequence of frames. Given a sequence of input RGB frames $X_{1:t} = (X_1, X_2, ..., X_t)$; $X_i \in \mathcal{R}^{n\times3}$, each with n pixels, our model generates the future frames $\widehat{Y}_{t+1:t+k} = (\widehat{Y}_{t+1}, \widehat{Y}_{t+2}, ..., \widehat{Y}_{t+k})$; $Y_i \in \mathcal{R}^{n\times3}$ for k future time-steps. In addition to $X_{1:t}$, we also use optical flows from FlowNet2 [21] as input to encapsulate the motion between the frames. Let $F_i \in \mathcal{R}^{n\times2}$ denotes the backward optical flow from the i^{th} frame to the $i-1^{th}$ frame.

We formulate frame prediction as future optical flow predictions and then synthesize future frames using these predicted optical flows. As illustrated in Fig. 3.1, our network takes $X_{1:t}$ and $F_{2:t}$ as input and produces arbitrary (k) number of future optical flows $\widehat{F}_{t+1:t+k} = (\widehat{F}_{t+1}, \widehat{F}_{t+2}, ..., \widehat{F}_{t+k})$. Given these predicted flows, we recursively warp a given or estimated previous frame to generate the current frame:

$$\hat{Y}_i = \mathcal{T}(\widehat{Y}_{i-1}, \widehat{F}_i), \tag{3.1}$$

where $\widehat{Y}_t = X_t$ and $\mathcal{T}$ denotes warping the previous frame to the current frame with the optical flow. While we apply warping recursively, our network only performs the forward pass once, which we will explain in the following paragraphs.

3.1 Temporal motion basis

A main goal of this work is to predict arbitrary number of future frames with a single forward pass through the network. To achieve this, instead of directly predict-

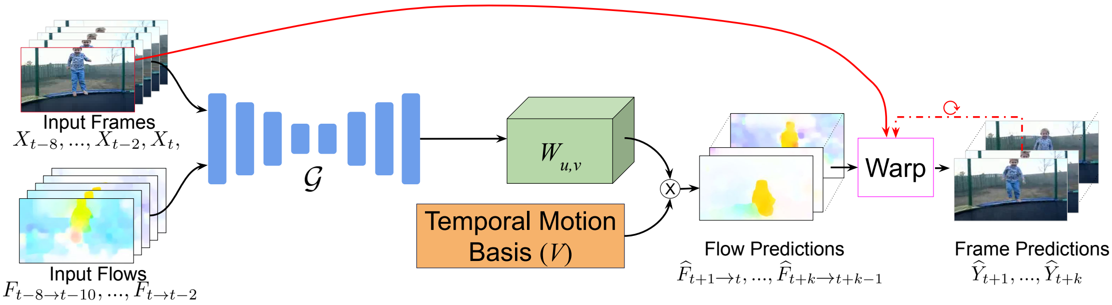
Input Frames
$X_{t-8}, ..., X_{t-2}, X_t,$
Input Flows
$F_{t-8\to t-10}, ..., F_{t\to t-2}$
$\mathcal{G}$
$W_{u,v}$
Temporal Motion Basis (V)
X
Flow Predictions
$\widehat{F}_{t+1\to t}, ..., \widehat{F}_{t+k\to t+k-1}$
Warp
Frame Predictions
$\widehat{Y}_{t+1}, ..., \widehat{Y}_{t+k}$

Figure 3.1: **Approach overview.** Given input frames and the corresponding estimated optical flows, the network $\mathcal{G}$ predicts basis coefficients $W_{u,v}$. We linearly combine these coefficients with learned temporal motion basis (V) to predict arbitrary-length future optical flows. The predicted optical flows are in turn used to synthesize future frames via warping image pixels.

ing future optical flows $\widehat{F}_{t+1:t+k}$, we represent optical flows as a linear combination of temporal motion basis vectors and predict the basis coefficients using our network. In DYAN [29], Liu *et al.* show that an input sequence of optical flows can be modeled with Linear Time Invariant (LTI) systems and this representation is successfully used for future optical flow prediction. During the training, DYAN learns a structured motion basis D in Eq. (3.2) of size $m \times N$ using a set of N dynamics-based atoms (columns of D) to encode the input sequence of optical flows $\widehat{F}_{t+1:t+m}$. These atoms are the impulse responses of a low order LTI system. The atoms are represented as complex numbers $p_i = \rho_i e^{j\psi_i}$ and are parameterized by magnitude ρ_i and phase ψ_i:

$$D = \begin{bmatrix} 1 & 1 & \dots & 1 \\ p_1 & p_2 & \dots & p_N \\ p_1^2 & p_2^2 & \dots & p_N^2 \\ \vdots & \vdots & \vdots & \vdots \\ p_1^{m-1} & p_2^{m-1} & \dots & p_N^{m-1} \end{bmatrix} \tag{3.2}$$

Each row in the dictionary D represents the temporal encoding for the particu-

$$V_{1,..} = \begin{bmatrix} 1 & 1 & 0 & -1 & 0 & 1 & 0 & \dots \\ 1 & \rho_1 \cos\psi_1 & \rho_1 \sin\psi_1 & -\rho_1 \cos\psi_1 & -\rho_1 \sin\psi_1 & \rho_2 \cos\psi_2 & \rho_2 \sin\psi_2 & \dots \\ 1 & \rho_1^2 \cos 2\psi_1 & \rho_1^2 \sin 2\psi_1 & (-\rho_1)^2 \cos 2\psi_1 & (-\rho_1)^2 \sin 2\psi_1 & \rho_2^2 \cos 2\psi_2 & \rho_2^2 \sin 2\psi_2 & \dots \\ 1 & \rho_1^3 \cos 2\psi_1 & \rho_1^3 \sin 2\psi_1 & (-\rho_1)^3 \cos 2\psi_1 & (-\rho_1)^3 \sin 2\psi_1 & \rho_2^3 \cos 2\psi_2 & \rho_2^3 \sin 2\psi_2 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \rho_1^{m-2} \cos(m-2)\psi_1 & \rho_1^{m-2} \sin(m-2)\psi_1 & (-\rho_1)^{m-2} \cos(m-2)\psi_1 & (-\rho_1)^{m-2} \sin(m-2)\psi_1 & \rho_2^{m-2} \cos(m-2)\psi_2 & \rho_2^{m-2} \sin(m-2)\psi_2 & \dots \\ 1 & \rho_1^{m-1} \cos(m-1)\psi_1 & \rho_1^{m-1} \sin(m-1)\psi_1 & (-\rho_1)^{m-1} \cos(m-1)\psi_1 & (-\rho_1)^{m-1} \sin(m-1)\psi_1 & \rho_2^{m-1} \cos(m-1)\psi_2 & \rho_2^{m-1} \sin(m-1)\psi_2 & \dots \end{bmatrix}_{m \times b}$$

Figure 3.2: **Motion Basis $V_{1,m}$.** The temporal motion basis $V_{1,m}$ is used to encode the motion dynamics in a given dataset. The goal of motion basis is to capture the motion encoding in the least possible combination of poles.

$$\begin{bmatrix} 1 & \rho_1^m \cos(m)\psi_1 & \rho_1^m \sin(m)\psi_1 & (-\rho_1)^m \cos(m)\psi_1 & (-\rho_1)^m \sin(m)\psi_1 & \rho_2^m \cos(m)\psi_2 & \rho_2^m \sin(m)\psi_2 & \dots \\ 1 & \rho_1^{m+1} \cos(m+1)\psi_1 & \rho_1^{m+1} \sin(m+1)\psi_1 & (-\rho_1)^{m+1} \cos(m+1)\psi_1 & (-\rho_1)^{m+1} \sin(m+1)\psi_1 & \rho_2^{m+1} \cos(m+1)\psi_2 & \rho_2^{m+1} \sin(m+1)\psi_2 & \dots \\ 1 & \rho_1^{m+2} \cos(m+2)\psi_1 & \rho_1^{m+2} \sin(m+2)\psi_1 & (-\rho_1)^{m+2} \cos(m+2)\psi_1 & (-\rho_1)^{m+2} \sin(m+2)\psi_1 & \rho_2^{m+2} \cos(m+2)\psi_2 & \rho_2^{m+2} \sin(m+2)\psi_2 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Figure 3.3: **Extended Motion Basis.** The rows of the motion basis are extended for future time-steps by using the parameter ρ_j and ψ_j. Thus, the motion basis can be used for predicting arbitrary future frames.

lar time-step. To avoid computing with complex numbers, Liu *et al.* [29] replace the columns of D with the real and imaginary parts of the complex numbers $p_i = \rho_i e^{j\psi_i}$ to obtain the motion basis $V_{1,m}$ in Fig. 3.2. The parameters ρ_j and phase ψ_j for each basis vectors are learned using back propagation. For further details, please refer to DYAN [29].

We reconstruct the input sequence $F_{1:m}$ by using the temporal motion basis $V_{(1,m)}$ and a corresponding sparse vector of coefficients $W_{u,v} \in \mathcal{R}^{n\times 2\times b}$ that selects and weighs the individual basis such that:

$$\min |F_{1:m} - W_{u,v} V_{(1,m)}^{\top}| \tag{3.3}$$

A key observation we make use of in this work is that we can easily extend the rows of the basis matrix $V_{(1,m)}$ to Fig. 3.3 for the additional future time-steps without adding any new parameters, as each basis vector is only represented by two parameters ρ_j and ψ_j. We explicitly denote the arbitrary dimensionality in V with $V_{(a,b)}$, denoting the basis matrix with rows from time-step a to time-step b.

3.2 Coefficient-Net

As illustrated in Fig. 3.1, we train a network which we call 'Coefficient-Net' ($\mathcal{G}$) that takes given frames $X_{1:t}$ and computed flows $F_{2:t}$ as input and predicts basis coefficients $W_{u,v} \in \mathcal{R}^{n\times 2\times b}$ at each pixel and for each of the horizontal u and vertical v motion directions:

$$W_{u,v} = \mathcal{G}\big(X_{1:t}, F_{2:t}\big). \tag{3.4}$$

Note that this network output ($W_{u,v}$) dimension is independent of the number of future frames we want to predict. We then obtain k future optical flows $\widehat{F}_{t+1:t+k}$ as a pixel-wise multiplication of $W_{u,v}$ and basis matrix $V_{(t+1,t+k)}$ with k rows:

$$\widehat{F}_{t+1:t+k} = W_{u,v} V_{(t+1,t+k)}^{\top}. \tag{3.5}$$

We use these predicted flows to iteratively estimate future video frames as explained earlier in Eq. (3.1). To predict q frames instead of k, we simply extend the basis matrix to q rows $V_{(t+1,t+q)}$ and then multiply with the predicted basis coefficients $W_{u,v}$. This enables us to make arbitrary number of future frame predictions with a single forward pass through the network. During network training, we use Coefficient-Net to re-generate both input frames $\widehat{Y}_{t-l:t}$ and future frames $\widehat{Y}_{t:t+k}$ by using basis matrix $V_{(t-l,t+k)}$ with $l+k$ rows. This is to make use of both input frames and future frames as ground-truth supervision during training. We discuss the details of the parameters in the later parts of this section.

3.3 Network Architecture

In order to capture spatio-temporal features from the input frames and flows, we use an encoder-decoder based 3D CNN with skip connections for coefficient-net $\mathcal{G}$. The 3 channels from the input RGB images and the 2 channels from the corresponding optical flows are concatenated to form 5 channel input at each time-step. We design our $\mathcal{G}$ inspired by various U-net type encoder-decoder architectures [44, 35, 11]. The feature maps are downsampled by applying 3D convolutions with a stride of $(1,2,2)$ followed by a Leaky Rectified Unit (LeakyRELU) [15] to capture the long-range spatial dependencies. Following [42], we use 3x7x7 and 3x5x5 convolution kernels in the first and second layers for capturing large displacements. In the subsequent layers, we use 3x3x3 convolution kernels. Each decoding layer concatenates features from the corresponding encoder layer and applies transposed convolution [46] followed by LeakyRELU. To minimize the checkerboard artifacts [37], the last two decoding layers are replaced with nearest neighbor upsampling followed by convolutions. The outputs from the decoder are convolved repeatedly to reduce the time-dimension to 1. The architecture details are provided in Table 3.1. Additionally, we enforce ℓ_0 regularization [33] on the final convolution layer based on stochastic gates to learn sparse basis coefficients. Lastly, we constrain the range of basis coefficients to $(-r, r)$ to improve the training stability. We choose the values for r empirically to account for the range of pixel motions in the given dataset. Greater range of motion directly relates to greater magnitudes of optical flow which require larger values of basis coefficients. We achieve this by applying modified Sigmoid (σ) activation function:

$$Z(s) = 2r\big(\sigma(s) - 0.5)\big) \tag{3.6}$$

3.4 Loss Functions

Our primary training loss for the Coefficient-Net is the ℓ_1 pixel reconstruction loss over the generated images:

$$L_1 = \sum_{i=t-l}^{t+k} \|X_i - \hat{Y}_i\|_1 \tag{3.7}$$

We note that, during training, the prediction $\widehat{Y}_i$ is synthesized by warping the ground truth frame X_{i-1} with the optical flow prediction $\widehat{F}_i$. We observe that this results in stable training and convergence to a better solution. However during inference stage, we warp the previous frame prediction $\widehat{Y}_{i-1}$ to generate the next frame $\widehat{Y}_i$. Additionally, we enforce ℓ_0 regularization [33] on the basis coefficients to generate sparse representations. The total loss is expressed as:

$$L_P = L_1 + \lambda \|W_{u,v}\|_0 \tag{3.8}$$

3.5 Training

We implement our work using PyTorch [40] neural network framework. We trained our Coefficient-Net using frames extracted from the MINI-KINETICS-200 dataset [58] after applying pre-processing steps suitable to the validation datasets as discussed in Chapter 4. We divide every 30 frames into a training sample. We use a batch size of 16 and train over 4 Nvidia V100 GPUs. We optimize with Adam [25] with the standard hyper-parameters and a learning rate of $1e^{-3}$ with no weight decay. Each training clip consists of **11** randomly cropped consecutive input frames. The input for the network $\mathcal{G}$ (Fig. 3.1) is every second frame sampled from the 11 consecutive images. The 5 evenly-spaced images capture greater spatio-temporal dynamics compared to using the 5 consecutive input frames for the same compute power requirements. Thus for a given training example of frames $X_{1:11}$, the input for module $\mathcal{G}$ are frames $X_{1:11:2}$ and the corresponding backward flows. During training, we condition V and $W_{u,v}$ to generate consecutive l input and k future frames. Using a coarse grid search, we observe that restricting the temporal motion basis V for $l = 5$ and $k = 5$ frames during training is a suitable right trade-off between capturing the motion dynamics and the computational power requirements.

Similar to [29], the motion basis $V_{l,k}$ is initialized with the individual bases uniformly distributed in the first quadrant between the two rings around the unit circle defined by $0.75 \leq \rho \leq 1.15$, their 3 mirror images in the 2nd, 3rd and 4th quadrants, and a fixed basis at $(\rho, \psi) = (1, 0)$. Following [29], the temporal motion basis $V_{l,k}$ is initialized with $b = 161$ basis vectors. However, unlike [29], we do not normalize each column of $V_{l,k}$ to norm 1 and instead observe superior results when replaced with a learned normalization factor.

Table 3.1: Coefficient-Net Architecture. Given input sequence of images and the corresponding optical flows, our architecture $\mathcal{G}$ predicts the basis coefficients of dimensions $2 \times 161 \times h \times w$. In the table, we use the input resolution size of UCF-101 dataset to demonstrate the feature map dimensions at each layer.

Layer Name	**Kernel Size**	**Stride**	**Channels out**	**Input Size**	**Output Size**	**Input Features**
Conv1	$3 \times 7 \times 7$	2	64	256×320	128×160	Images and Flows
Conv2	$3 \times 5 \times 5$	2	128	128×160	64×80	Conv1
Conv3a	$3 \times 5 \times 5$	2	256	64×80	32×40	Conv2b
Conv3b	$3 \times 3 \times 3$	1	256	32×40	32×40	Conv3a
Conv4a	$3 \times 3 \times 3$	2	512	32×40	16×20	Conv3b
Conv4b	$3 \times 3 \times 3$	1	512	16×20	16×20	Conv4a
Conv5a	$3 \times 3 \times 3$	2	512	16×20	8×10	Conv3b
Conv5b	$3 \times 3 \times 3$	1	512	8×10	8×10	Conv4a
Conv6a	$3 \times 3 \times 3$	2	1024	8×10	4×5	Conv4b
Conv6b	$3 \times 3 \times 3$	1	1024	4×5	4×5	Conv5a
Deconv5	$1 \times 4 \times 4$	2	512	4×5	8×10	Conv6b
Deconv4	$1 \times 4 \times 4$	2	256	8×10	16×20	Deconv5+Conv5b
Deconv3	$1 \times 4 \times 4$	2	128	16×20	32×40	Deconv4+Conv4b
Deconv2	$1 \times 4 \times 4$	2	64	32×40	64×80	Deconv3+Conv3b
Deconv1	$1 \times 4 \times 4$	2	32	64×80	128×160	Deconv2+Conv2
Deconv0	$1 \times 4 \times 4$	2	16	128×160	256×320	Deconv1+Conv1
Coefficients	$5 \times 3 \times 3$	1	322	256×320	256×320	Deconv0+Images and Flows

Chapter 4

Experiments

We evaluate the proposed method against the state-of-the-art approaches on benchmark datasets. We show the model performance on both the immediate-frame prediction and multi-frame prediction settings. We also perform multiple ablation studies to analyze the importance of various components in our architecture. All the source code and trained models will be made available to the public.

4.1 Dataset and Metrics.

We evaluate our model on the CALTECH PEDESTRIAN (CALTECHPED) [10] and UCF-101 [47] datasets, using Mean-Squared-Error (MSE/L2), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM) [56].

4.2 Results on the Caltech Pedestrian dataset.

Following the settings in Lotter *et al.* [32], we adjusted the frame rate of CALTECHPED dataset and MINI-KINETICS-200 to 10 and downsize and center-crop the videos to 128×160. We train our model on the MINI-KINETICS-200 dataset and directly evaluate on the CALTECHPED dataset. Following [32], we extract frames from sets $6 - 10$ of the CALTECHPED dataset resulting in 3885 testing samples. During training on the MINI-KINETICS-200 dataset, the frame sequences are subsampled to 10 FPS and the frames are downsized and random cropped to match the same dimensions of 128×160.

Table 4.1: Frame prediction results on the Caltech Pedestrian dataset. MSE results are in $1e^{-3}$.

Method	**MSE↓** **1 frame**	**SSIM↑** **1 frame**	**MSE↓** **5 frames**	**SSIM↑** **5 frames**	**MSE↓** **10 frames**	**SSIM↑** **10 frames**
BeyondMSE [34]	3.73	0.8432	19.09	0.6019	47.69	0.4532
MCNet [52]	3.13	0.8743	10.79	0.7354	19.95	0.6423
DYAN [29]	3.88	0.8671	9.08	0.7568	13.87	0.6858
CopyLast	7.95	0.7779	16.72	0.6561	23.39	0.5915
Ours	**2.45**	**0.8965**	**6.83**	**0.7936**	**11.52**	**0.7185**

The range r of the basis coefficients $W_{u,v}$ is restricted to $(-2, 2)$ using the modified Sigmoid activation function in Eq. (3.6). Following [29], we normalize the image pixel values between 0 and 1 before computing the image metrics. We summarize the quantitative results in Table 4.1. Results show that our approach performs favorably against the current state of the art models. For fairness, we use the codebases of [34, 52, 29] and evaluate their models on our validation set. For DYAN, we use the Coarse2Fine Optical Flows [41] as the inputs and obtain multi-frame predictions recursively as suggested in [29]. The sample qualitative results are shown in Fig. 4.1, 4.2, 4.3, 4.4, 4.5. As shown in the figures, our model generates sharper predictions. A significant difference in the visual results between DYAN and Ours can be observed in the later time-steps of the prediction.

4.3 Results on the UCF-101 dataset.

On the UCF-101 dataset, we train our model to generate higher resolution predictions of size 240×320. The training set of videos from MINI-KINETICS-200 is resized and random-cropped to match the dimensions of the UCF-101 dataset [47]. Following [34], we test on 10% of the dataset. Every 10th video was extracted from the test list provided by [34]. The frames from the entire video are used in the validation set

Table 4.2: SSIM and PSNR metrics are measured for the frame predictions on the UCF-101 dataset utilizing the complete image.

Method	**PSNR↑** **1 frame**	**SSIM↑** **1 frame**	**PSNR↑** **5 frames**	**SSIM↑** **5 frames**	**PSNR↑** **10 frames**	**SSIM↑** **10 frames**
BeyondMSE [34]	27.00	0.8231	20.57	0.6290	16.86	0.5109
MCNet [52]	29.99	0.9003	24.78	0.8073	22.07	0.7454
DYAN [29]	30.70	0.8872	25.51	0.8119	23.65	0.7717
CopyLast	31.32	0.8883	25.48	0.8041	23.31	0.7568
Ours with different norms						
No Norm enforced	29.51	0.8891	25.03	0.8084	21.72	0.6970
$L1$ Norm	29.37	0.9002	25.46	0.8254	23.51	0.7773
$L0$ Norm - Our main model	**30.02**	**0.9063**	**25.83**	**0.8313**	**23.76**	**0.7823**

resulting in 6440 validation examples. We set the range r of the basis coefficients $W_{u,v}$ to $(-2, 2)$ in Eq. (3.6). Table 4.2 shows the quantitative results. Following the prior works, we also perform experiments to measure the metrics only in the areas of motion. Similar to [34, 52], we compute EpicFlow [43] optical flows between the previous and the current ground truth frames, compute the magnitude, and normalize it to $[0, 1]$. The pixels where no motion was observed are masked out by using the computed optical flow magnitude. If the flow magnitude is less than 0.2, the pixel value is replaced and set to 0 in the prediction and the corresponding ground truth frame. The image metrics are computed on the resulting masked images and are summarized in Table 4.3. Results demonstrate that our method shows consistent performance gains on all the metrics across the different number of future frame prediction settings. Fig. 4.6, 4.7, 4.8, 4.9, 4.10 shows sample qualitative results on the UCF-101 dataset.

Table 4.3: SSIM and PSNR metrics are measured for the frame predictions on the UCF-101 dataset for the areas of motion in the image using motion based pixel mask.

Method	PSNR↑ 1 frame	SSIM↑ 1 frame	PSNR↑ 5 frames	SSIM↑ 5 frames	PSNR↑ 10 frames	SSIM↑ 10 frames
BeyondMSE [34]	32.98	0.9236	27.11	0.8640	23.80	0.8298
MCNet [52]	34.40	0.9363	29.84	0.8962	27.53	0.8738
DYAN [29]	34.52	0.9362	30.80	0.8999	28.95	0.8794
Ours	**34.84**	**0.9407**	**31.01**	**0.9055**	**29.18**	**0.8852**

Table 4.4: Results on the CALTECHPED dataset with models trained on the KITTI and MINI-KINETICS-200 datasets using Coarse2Fine [41] and FlowNet2 [21] optical flows.

Method	Optical Flows	Training dataset	MSE↓ 1 frame	SSIM↑ 1 frame	MSE↓ 5 frames	SSIM↑ 5 frames	MSE↓ 10 frames	SSIM↑ 10 frames
DYAN [29]	Coarse2Fine [41]	KITTI	3.88	0.8671	9.08	0.7568	13.87	0.6858
DYAN	FlowNet2 [21]	KITTI	3.72	0.8665	10.39	0.7380	16.50	0.6631
DYAN:Extended	FlowNet2 [21]	KITTI	3.37	0.8812	9.12	0.7581	14.85	0.6774
DYAN:Extended	FlowNet2 [21]	MINI-KINETICS-200	3.19	0.8854	8.46	0.7733	13.48	0.6992
Ours	FlowNet2 [21]	KITTI	2.96	0.8869	7.738	0.7800	12.58	0.7046
Ours	FlowNet2 [21]	MINI-KINETICS-200	**2.45**	**0.8965**	**6.83**	**0.7936**	**11.52**	**0.7185**

4.4 Ablation studies

4.4.1 DYAN trained with FlowNet2 optical flows.

The DYAN results in Table 4.1 and Table 4.2 are evaluated using the Coarse2Fine Optical Flows as originally suggested in [29]. To test the significance of FlowNet2 optical flows, we train DYAN on the KITTI dataset with FlowNet2 optical flows and evaluate the results on the CALTECHPED dataset. In Table 4.4, we observe that DYAN trained with FlowNet2 optical flows performs better than the original DYAN model. However, Coefficient-Net trained on KITTI dataset using the FlowNet2 optical flows consistently outperforms the DYAN models. This suggests that our high capacity Coefficient-Net captures is better at capturing diverse motion ranges compared to the DYAN.

4.4.2 Extending the motion basis for DYAN.

The DYAN results in Table 4.1 and Table 4.2 are obtained recursively as originally recommended in [29]. We perform experiments to compare the DYAN performance on multi-frame prediction using a single forward pass by extending the DYAN motion basis. This is performed by modifying the DYAN model and removing the basis normalization step. This enables us to extend the DYAN basis for arbitrary time-steps. The modified DYAN model is trained on the KITTI dataset with FlowNet2 opticals flows and evaluated on the CALTECHPED dataset. In Table 4.4, we observe that the newly modified single pass DYAN model performs better than the original DYAN model trained on the KITTI dataset with FlowNet2 optical flows. Nevertheless, our higher capacity network Coefficient-Net outperforms the DYAN models.

4.4.3 Training on the KITTI or MINI-KINETICS-200 dataset.

For validating on CALTECHPED dataset, prior works mainly train on the KITTI dataset. We study the relevance of using the MINI-KINETICS-200 dataset for Coefficient-Net compared with the KITTI dataset. We compare DYAN performance with our network performance when trained with the KITTI or MINI-KINETICS-200 dataset. Table 4.4 shows the quantitative results. Our model trained on the KITTI dataset performs better than DYAN and worse than our model trained on the MINI-KINETICS-200 dataset. We also observe that our high capacity Coefficient-Net can learn from large datasets to better capture diverse motion ranges compared to the DYAN model.

4.4.4 Sparsity of basis coefficients.

To validate the importance of ℓ_0 sparsity on the basis coefficients, we train models using ℓ_0 norm, ℓ_1 norm, and finally without any norm enforced on the basis coefficients (Eq. (3.8)). We train all models using the MINI-KINETICS-200 dataset and evaluate them on the UCF-101 dataset. The results in Table 4.2 show that when no sparsity norm is used, our network trained on predicting 5 frames does not generalize well to predicting more future frames (10 frames). This is attributed to the implicit assumption for a sparse representation of the dynamics of the optical flow in the motion basis [29].

4.4.5 Limitations.

We observe that our model does not perform well when there are multiple independently moving objects in a given video. This is partly due to the limitation of the motion basis space we choose to represent temporal pixel motion dynamics. An interesting future direction would be to improve the expressiveness of the temporal motion basis while retaining the desired property of being able to predict arbitrary-length future video frames.

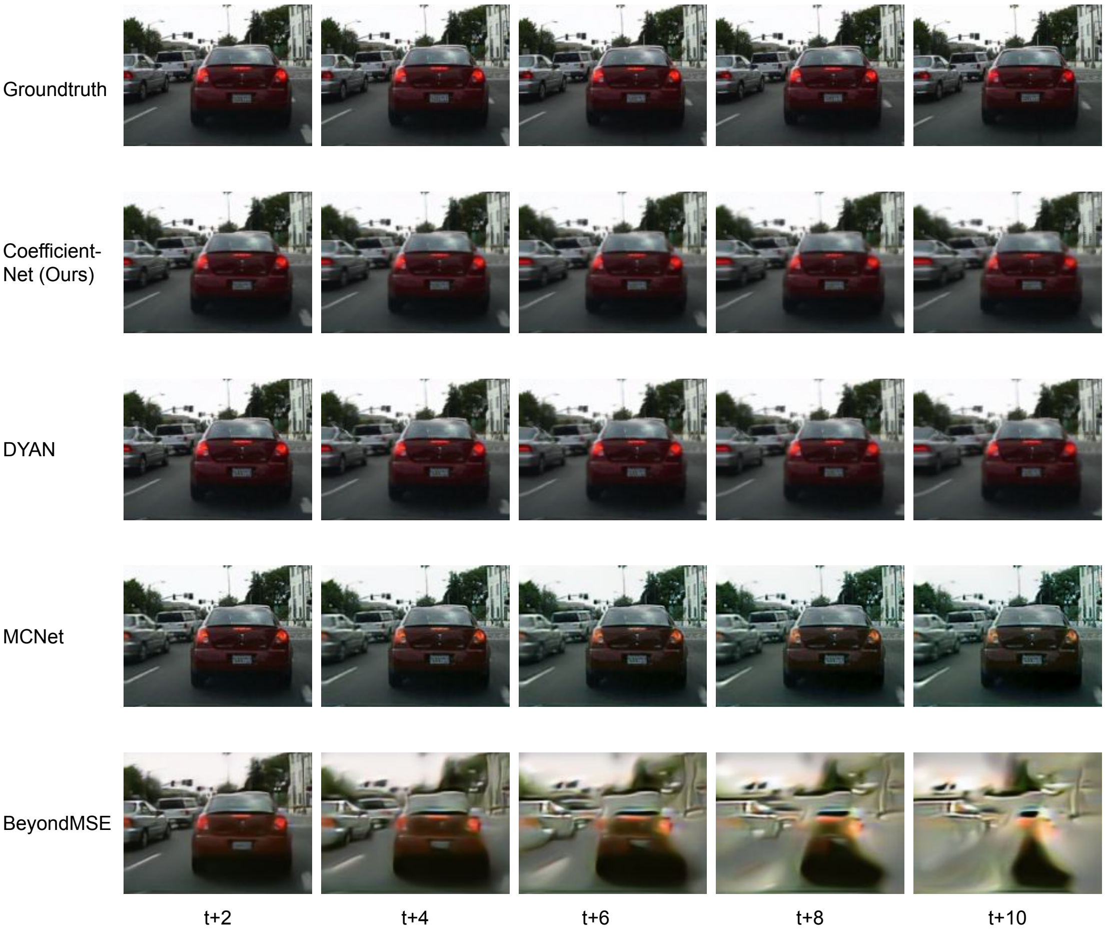

Figure 4.1: Qualitative results on a sample video from the Caltech Pedestrian dataset. Results show that our predicted frames matches more closely with ground-truth frames compared to predictions with other state-of-the-art techniques. The silver car on the left is the least distorted in the Coefficient-Net results when compared to the other methods.

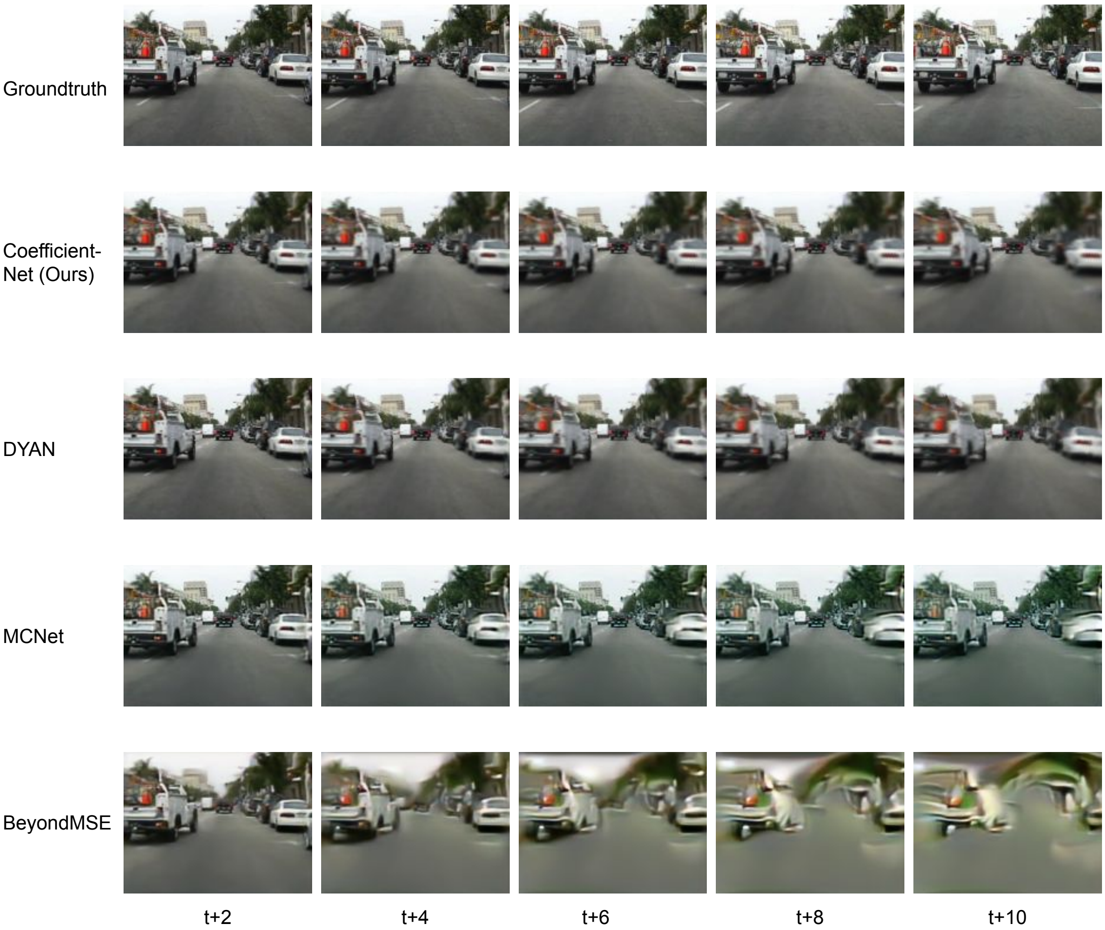

Figure 4.2: Qualitative results on a sample video from the Caltech Pedestrian dataset. As shown in the image, the white car on the right is modeled more accurately by our Coefficient-Net.

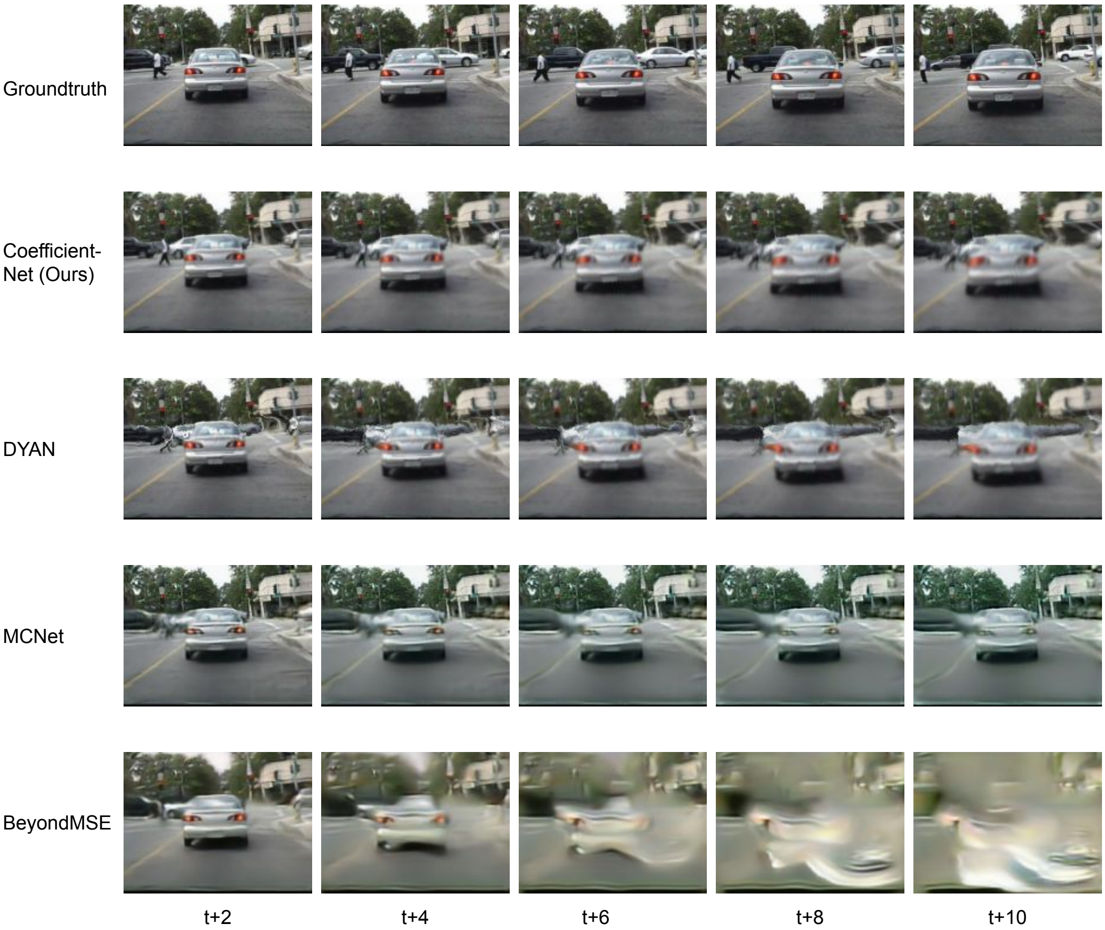

Figure 4.3: Qualitative results on a sample video from the Caltech Pedestrian dataset. The image illustrates that results generated through Coefficient-Net contain less noise compared to the other methods.

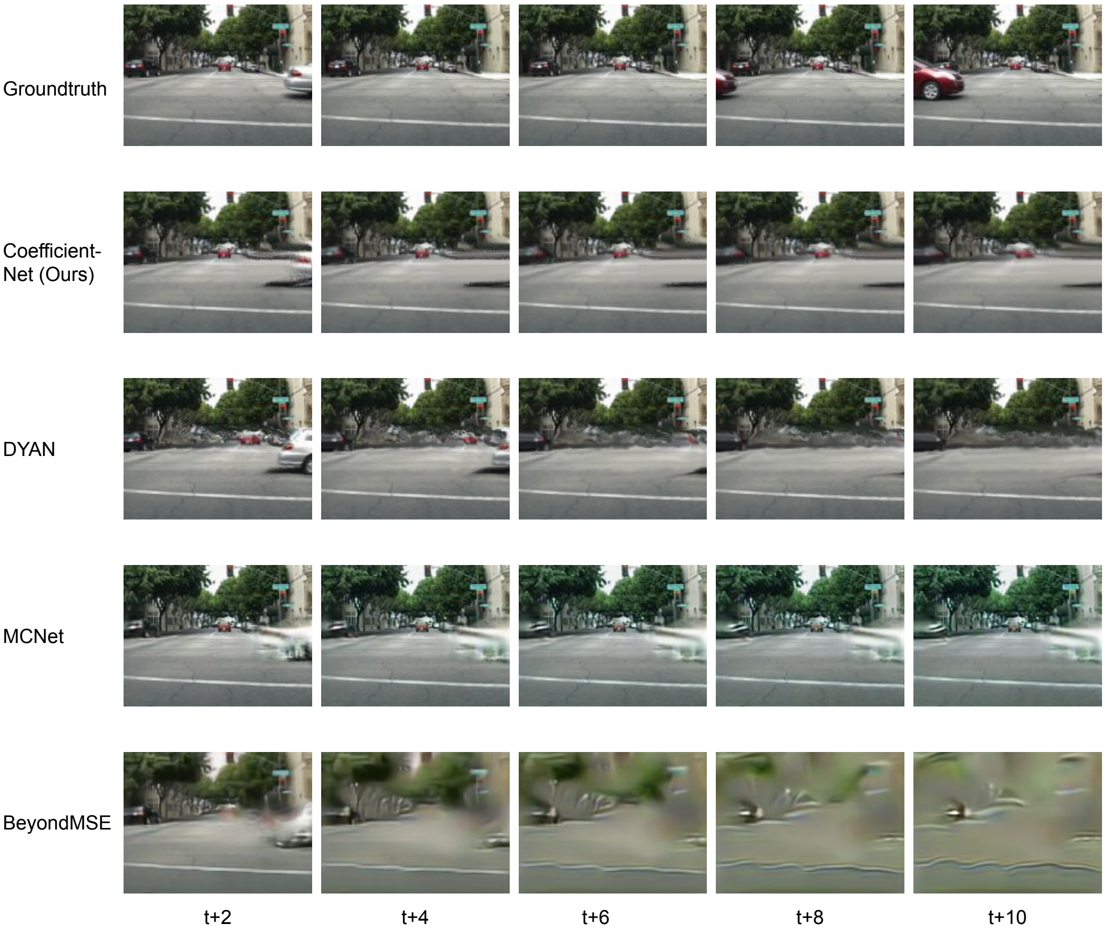

Figure 4.4: Qualitative results on a sample video from the Caltech Pedestrian dataset. The image illustrates that results generated through Coefficient-Net contain less noise compared to the other methods.

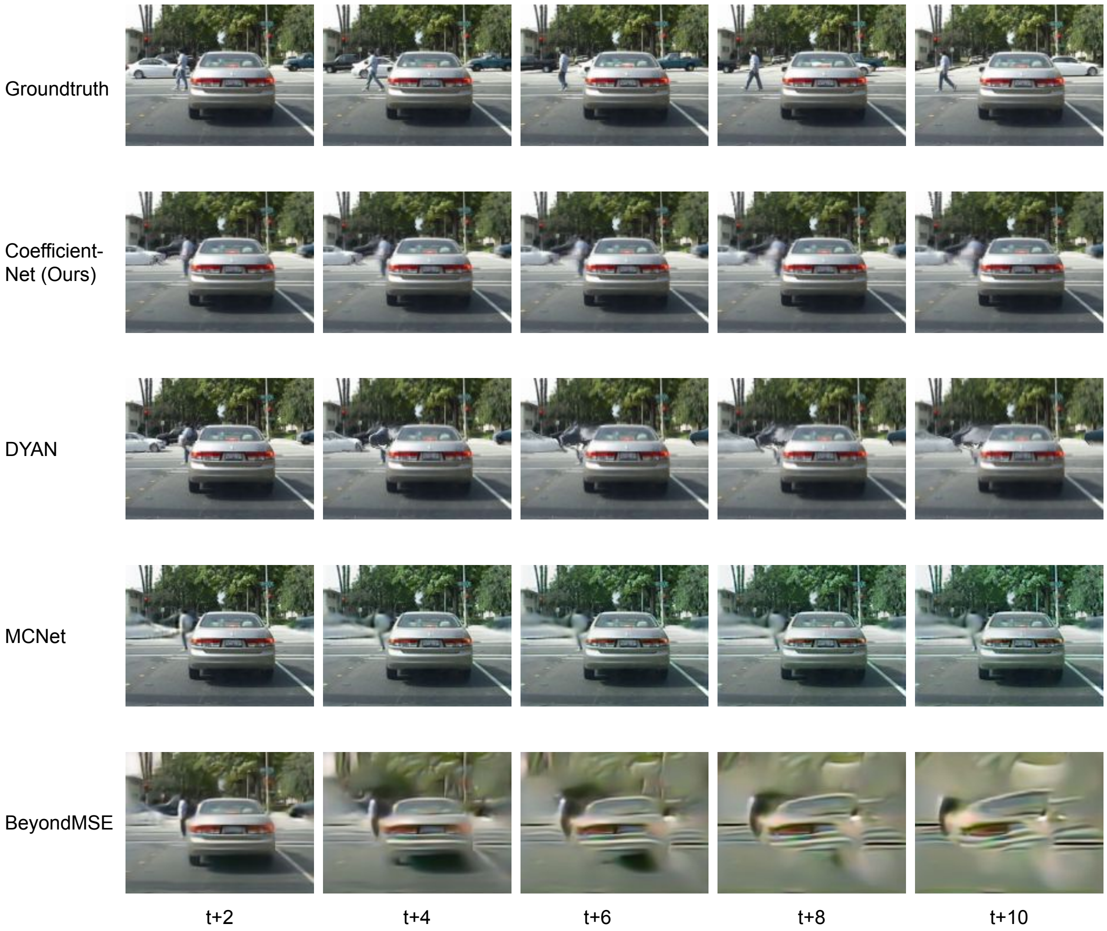

Figure 4.5: Qualitative results on a sample video from the Caltech Pedestrian dataset. We notice less distortion in the results generated by Coefficient-Net compared to the other methods.

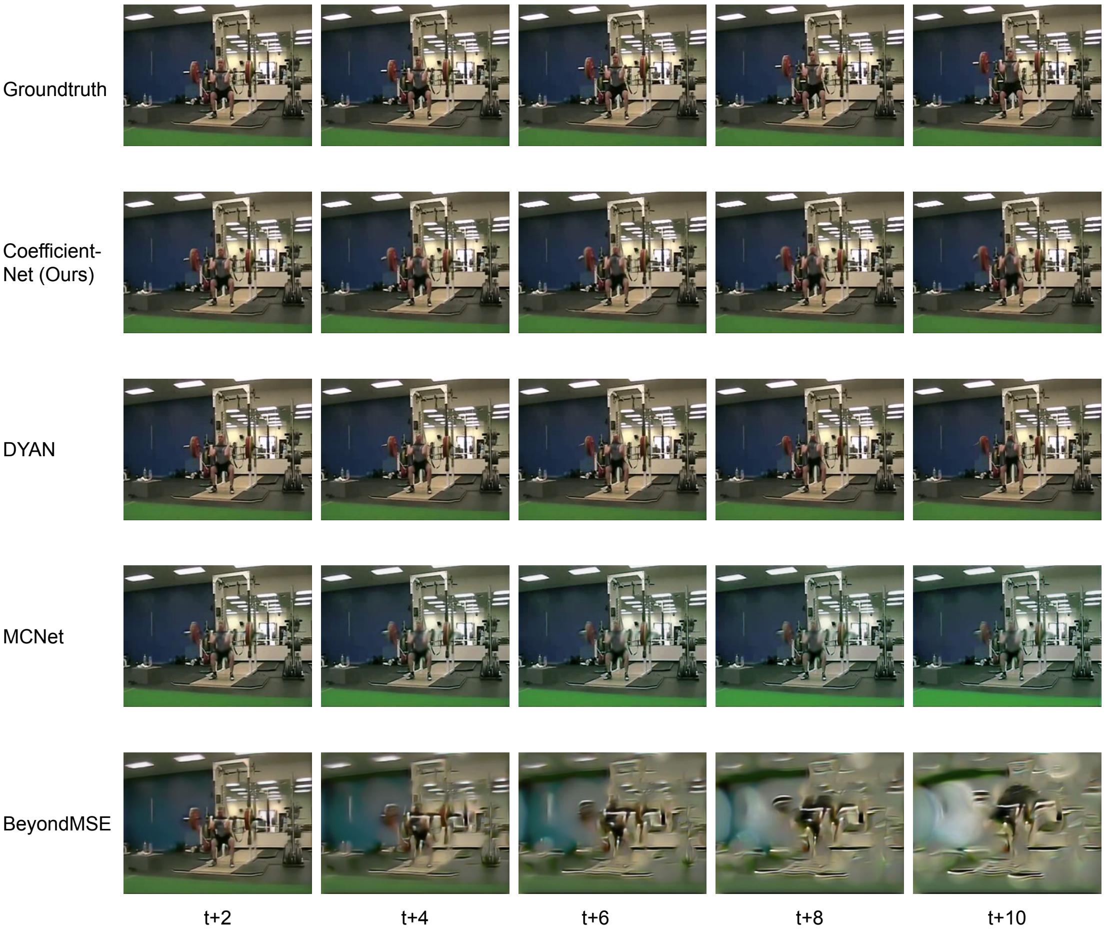

Figure 4.6: Qualitative results on a sample UCF-101 dataset video. We observe that our network predicted frames have motion that more closely follows the ground-truth motion when compared to the prediction by the state-of-the-art DYAN [29] network. The arms and torso of the person are highly distorted in the DYAN compared to out Coefficient-Net.

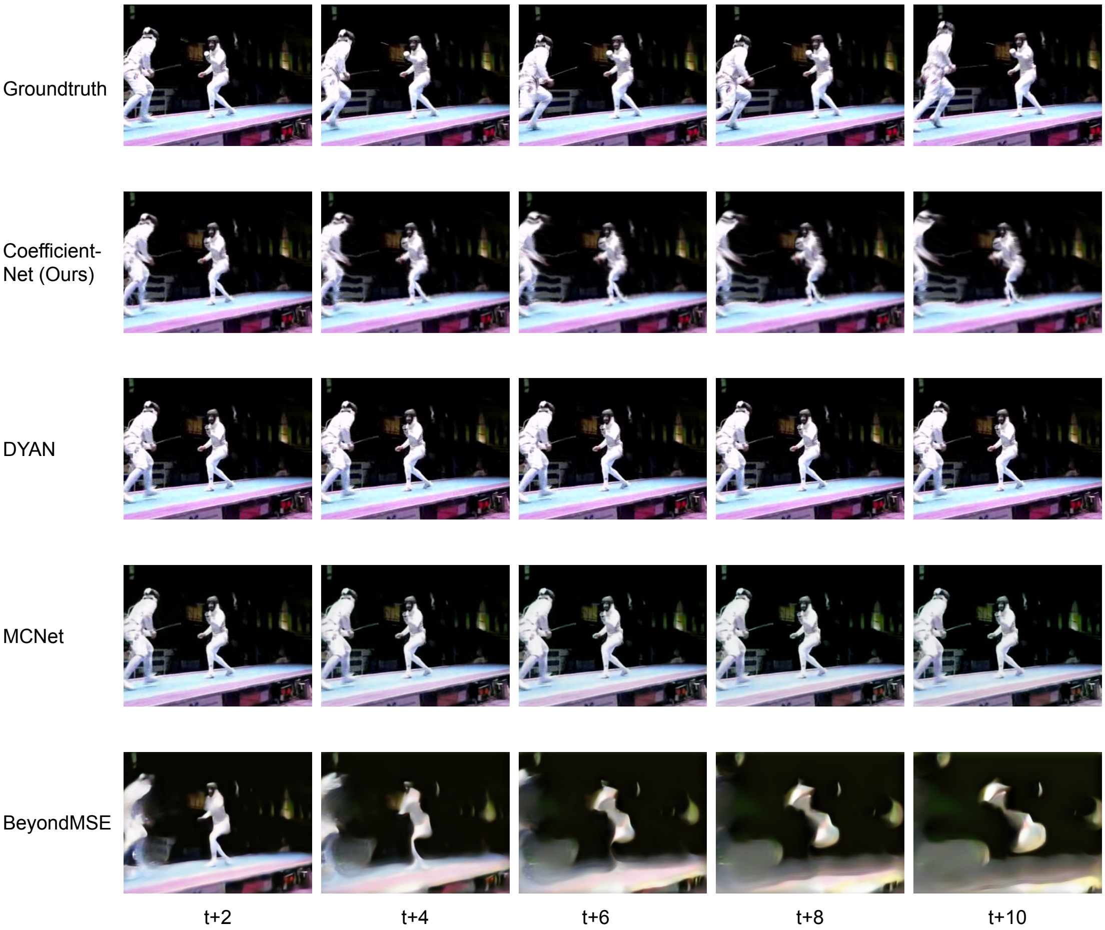

Figure 4.7: Qualitative results on a sample UCF-101 dataset video. While DYAN generates static images copied from the previous time-step, results from our Coefficient-Net demostrate motion similar to the ground-truth.

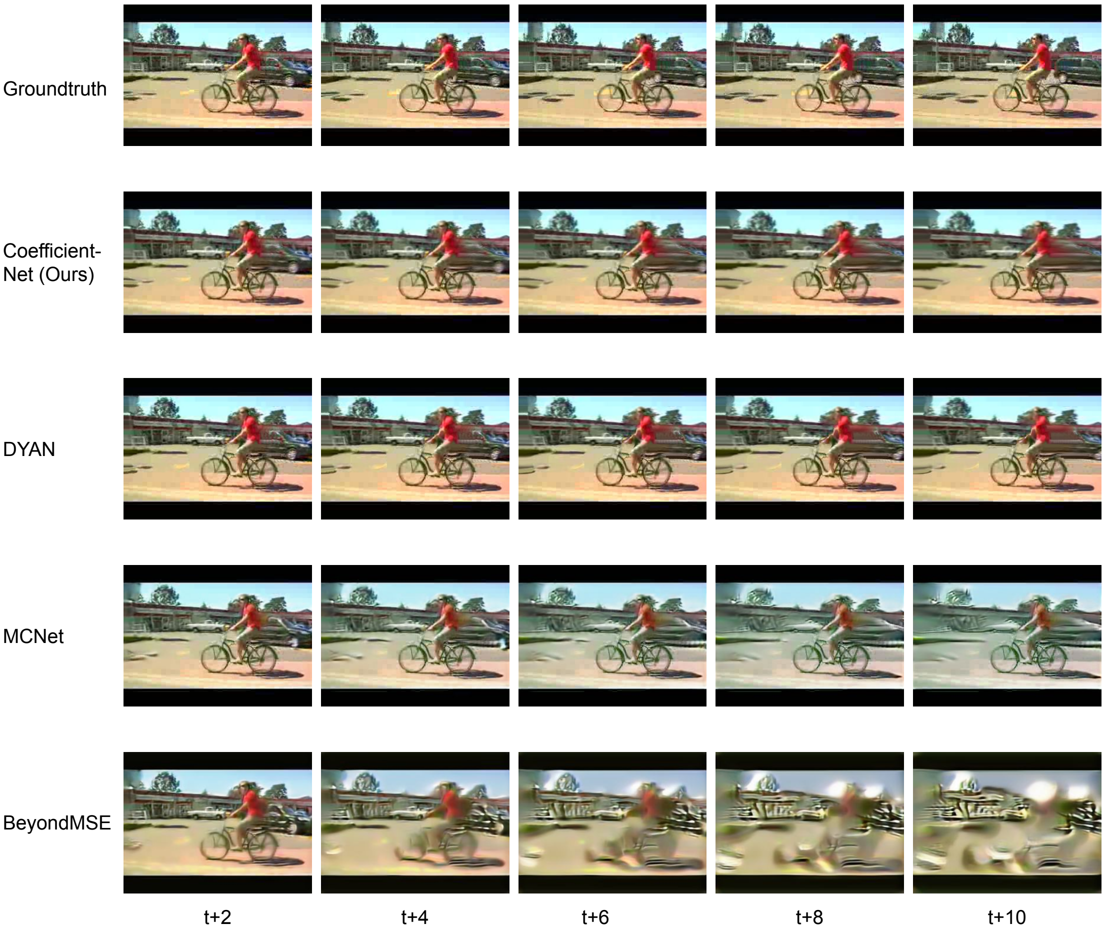

Figure 4.8: Qualitative results on a sample UCF-101 dataset video. The person on the bicycle is less distorted in the Coefficient-Net results and closely follows the ground-truth motion when compared to the other results.

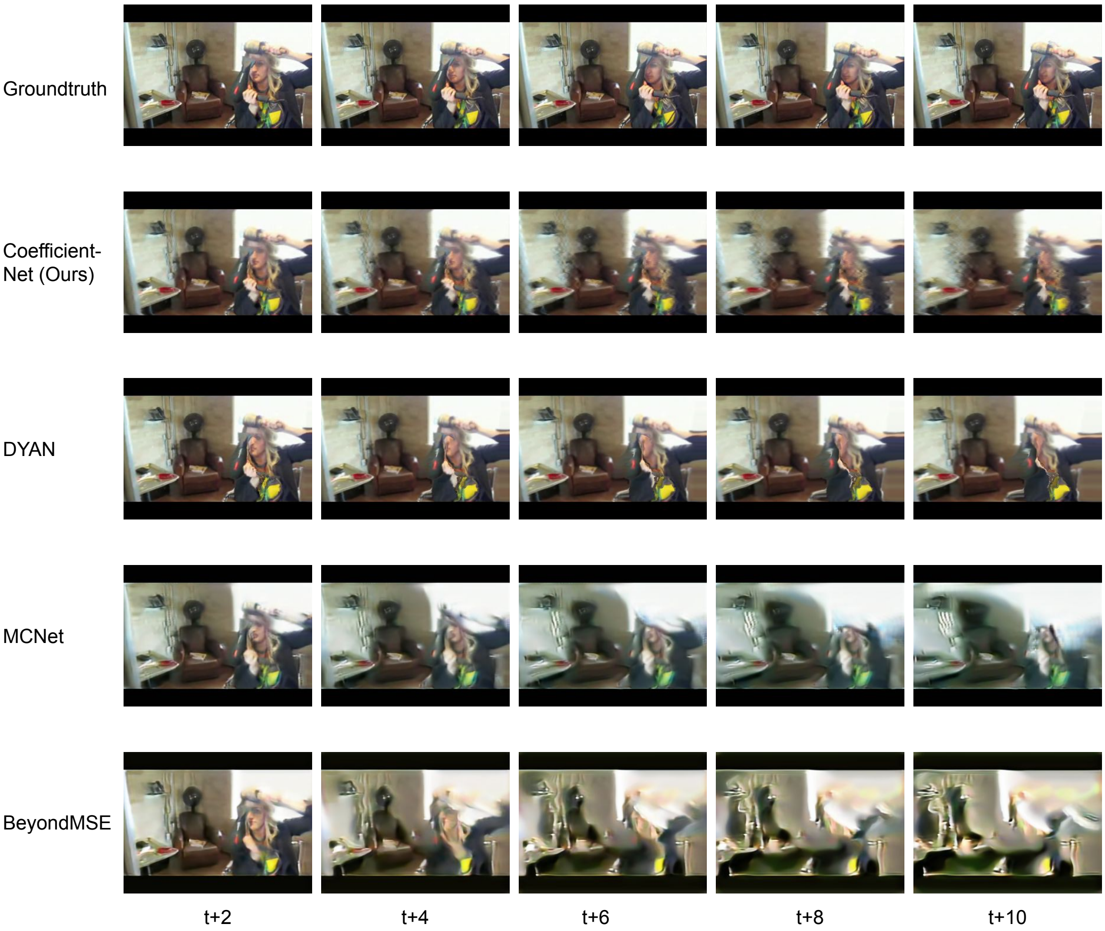

Figure 4.9: Qualitative results on a sample UCF-101 dataset video. While DYAN generates a static image over the future time-steps, our Coefficient-Net closely follows the ground-truth motion.

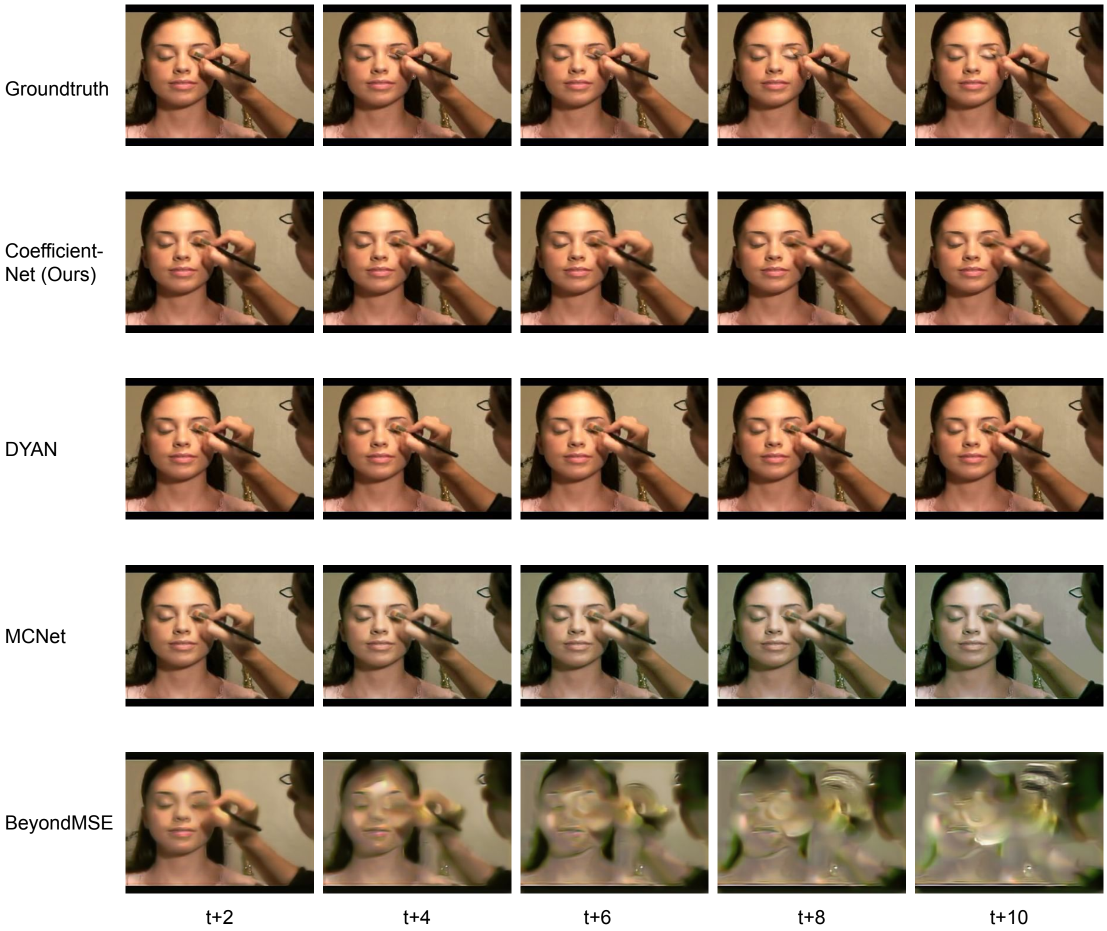

Figure 4.10: Qualitative results on a sample UCF-101 dataset video. While DYAN generates a static image over the future time-steps, our Coefficient-Net closely follows the ground-truth motion.

Chapter 5

Conclusion

In this thesis, after formulating the the task of video prediction, we have closely reviewed the various approaches employed for the task: prediction in the raw pixel space, prediction through latent factors and their applicability for long-term prediction. The current existing methods are limited to predictions in the short-term horizons. While frames in the immediate future are extrapolated with high accuracy, in the long term horizon the prediction problem becomes challenging. Initial solutions consisted of conditioning the prediction on previously predicted frames. However, these recursive models tend to accumulate prediction errors that progressively diverge the generated prediction from the expected outcome. On the other hand, due to memory issues, there is a lack of resolution in predictions.

To this end, we introduce a novel deep learning technique for multi-frame video prediction which can predict an arbitrary number of video frames with a single forward pass. To this end, we propose learning temporal motion encodings instead of directly predicting future frames or optical flows. Specifically, we predict fixed-dimensional basis coefficients and multiply them with arbitrary time-length motion basis vectors to predict an arbitrary number of future optical flows. We then use these predicted optical flows to synthesize future frames. The motion basis vectors are learned together with the main network. To our knowledge, this is the first neural network approach for video prediction that can predict an arbitrary number of future frames with a single forward pass through the network.

In the end, we have discussed the performance results on the two most popular datasets, the UCF-101 dataset and the CALTECHPED dataset using the most common metrics to demonstrate the superior performance of our method compared with several state-of-the-art video prediction techniques. We also provide useful insights for the future research directions and open problems.

In conclusion, video prediction is a promising research area for the self-supervised learning of rich spatiotemporal data to improve the prediction capabilities of the existing intelligent decision-making systems. While great strides have been made, there is still room for improvement in video prediction using the modern deep learning techniques.

Bibliography

[1] S. Aigner and M. Körner. Futuregan: Anticipating the future frames of video sequences using spatio-temporal 3d convolutions in progressively growing gans. *arXiv preprint arXiv:1810.01325*, 2018.

[2] M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine. Stochastic variational video prediction. In *ICLR*, 2018.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[4] W. Byeon, Q. Wang, R. Kumar Srivastava, and P. Koumoutsakos. Contextvp: Fully context-aware video prediction. In *ECCV*, 2018.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40:834–848, 2017.

[6] E. Denton and R. Fergus. Stochastic video generation with a learned prior. In *ICML*, 2018.

[7] E. Denton and R. Fergus. Stochastic video generation with a learned prior. In *ICML*, 2018.

[8] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NeurIPS*, 2015.

[9] E. L. Denton et al. Unsupervised learning of disentangled representations from video. In *NeurIPS*, 2017.

[10] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *TPAMI*, 34(4):743–761, 2012.

[11] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015.

[12] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *NeurIPS*, 2016.

[13] H. Gao, H. Xu, Q.-Z. Cai, R. Wang, F. Yu, and T. Darrell. Disentangling propagation and generation for video prediction. In *ICCV*, 2019.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Y.-H. Ho, C.-Y. Cho, W.-H. Peng, and G.-L. Jin. Sme-net: Sparse motion estimation for parametric video prediction through reinforcement learning. In *ICCV*, 2019.

[18] M. Hoai and F. De la Torre. Max-margin early event detectors. *IJCV*, 107(2):191–202, 2014.

[19] R. Hou, H. Chang, B. Ma, and X. Chen. Video prediction with bidirectional constraint network. In *FG*, 2019.

[20] A. Hu, F. Cotter, N. Mohan, C. Gurau, and A. Kendall. Probabilistic future prediction for video scene understanding. *arXiv preprint arXiv:2003.06409*, 2020.

[21] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017.

[22] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NeurIPS*, 2015.

[23] V. Jain, J. F. Murray, F. Roth, S. Turaga, V. Zhigulin, K. L. Briggman, M. N. Helmstaedter, W. Denk, and H. S. Seung. Supervised learning of image restoration with convolutional networks. In *ICCV*, 2007.

[24] N. Kalchbrenner, A. van den Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. Video pixel networks. In *ICML*, 2017.

[25] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[26] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *ECCV*, 2012.

[27] Y.-H. Kwon and M.-G. Park. Predicting future frames using retrospective cycle gan. In *CVPR*, 2019.

[28] X. Liang, L. Lee, W. Dai, and E. P. Xing. Dual motion gan for future-flow embedded video prediction. In *ICCV*, 2017.

[29] W. Liu, A. Sharma, O. I. Camps, and M. Sznaier. Dyan: A dynamical atoms-based network for video prediction. In *ECCV*, 2018.

[30] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, 2017.

[31] W. Lotter, G. Kreiman, and D. Cox. Unsupervised learning of visual structure using predictive generative networks. *arXiv preprint arXiv:1511.06380*, 2015.

[32] W. Lotter, G. Kreiman, and D. D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. In *ICLR*, 2017.

[33] C. Louizos, M. Welling, and D. P. Kingma. Learning sparse neural networks through l0 regularization. In *ICLR*, 2018.

[34] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016.

[35] F. Milletari, N. Navab, and S.-A. Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, 2016.

[36] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV*, 2016.

[37] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. In *Distill*, 2016.

[38] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh. Action-conditional video prediction using deep networks in atari games. In *NeurIPS*, 2015.

[39] S. Oprea, P. Martinez-Gonzalez, A. Garcia-Garcia, J. A. Castro-Vargas, S. Orts-Escolano, J. Garcia-Rodriguez, and A. Argyros. A review on deep learning techniques for video prediction. *arXiv preprint arXiv:2004.05214*, 2020.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[41] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. In *CVPR*, 2017.

[42] F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, A. Tao, and B. Catanzaro. Sdc-net: Video prediction using spatially-displaced convolution. In *ECCV*, 2018.

[43] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, 2015.

[44] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015.

[45] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[46] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *TPAMI*, 39:640–651, 2014.

[47] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[48] N. Srivastava, E. Mansimov, and R. Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015.

[49] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2017.

[50] A. Terwilliger, G. Brazil, and X. Liu. Recurrent flow-guided semantic forecasting. In *WACV*, 2019.

[51] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. In *CVPR*, 2017.

[52] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. In *ICLR*, 2017.

[53] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee. Learning to generate long-term future via hierarchical prediction. In *ICML*, 2017.

[54] C. Vondrick and A. Torralba. Generating the future with adversarial transformers. In *CVPR*, 2017.

[55] Y. Wang, L. Jiang, M.-H. Yang, L.-J. Li, M. Long, and L. Fei-Fei. Eidetic 3d lstm: A model for video prediction and beyond. In *ICLR*, 2018.

[56] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[57] N. Wichers, R. Villegas, D. Erhan, and H. Lee. Hierarchical long-term video prediction without supervision. In *ICML*, 2018.

[58] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *ECCV*, 2018.

[59] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 2015.

[60] F. Yu, V. Koltun, and T. Funkhouser. Dilated residual networks. In *CVPR*, 2017.

[61] W. Yu, Y. Lu, S. Easterbrook, and S. Fidler. Crevnet: Conditionally reversible video prediction. *arXiv preprint arXiv:1910.11577*, 2019.

[62] K.-H. Zeng, W. B. Shen, D.-A. Huang, M. Sun, and J. Carlos Niebles. Visual forecasting by imitating dynamics in natural sequences. In *ICCV*, 2017.

[63] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *CVPR*, 2019.