Decomposing camera and object motion for an improved video sequence prediction

Meenakshi Sarkar
Department of Aerospace Engineering
Indian Institute of Science
Bangalore, India 560012
meenakshisar@iisc.ac.in

Debasish Ghose
Department of Aerospace Engineering
Indian Institute of Science
Bangalore, India 560012
dghose@iisc.ac.in

Abstract

We propose a novel deep learning framework that focuses on decomposing the motion or the flow of the pixels from the background for an improved and longer prediction of video sequences. We propose to generate multi-time step pixel level prediction using a framework that is trained to learn the temporal and spatial dependencies encoded in video data separately. The proposed framework called VANet would be able to extend the static video predictions where the camera is stationary, to the dynamic case where the camera is mounted on a moving platform. This framework decomposes the flow of the image sequences into velocity and acceleration maps and learns the temporal transformations using a convolutional LSTM network. The content of the static background is filtered through a convolutional neural network and then combined with the masks generated from the temporal transformation network to generate the final prediction.

1 Introduction and Related Works

Prediction is an integral part of our day to day planning and decision making process and it often requires us to understand the complex interactions between the dynamics of various objects in the environment. This is why it is often considered as a fundamental component of intelligence Bubic, Cramon, and Schubotz (2010). Video prediction often decodes much useful information about the surroundings in a format which is rich in information and can be exploited by learning algorithms. However, the nature of the complex interactions between the dynamics of the different objects in a scene, makes long term video prediction a daunting learning problem Finn, Goodfellow, and Levine (2016), Finn and Levine (2017), Mathieu, Couprie, and LeCun (2015), Villegas et al. (2017), Gao et al. (2019), Villegas et al. (2019). Based on the state of the art literature, multi-time step video prediction can be broadly divided into two categories: (i) Video prediction in a fully observable static background where the camera remains still during the course of the recording Finn and Levine (2017), Mathieu, Couprie, and LeCun (2015), Villegas et al. (2017); and (ii) Video prediction in dynamic background where the camera is mounted on a moving platform (such as car or a mobile robot). The latter case is often referred to as prediction in partially observable scenario in the literature Gao et al. (2019), Villegas et al. (2019). The notion of partial observability comes from the continuous occlusion of the background from the motion of the camera.
In the context of automation, planning of different manipulation tasks is often associated with video prediction in a fully observable environment where the camera is fixed and stationary. However, in the case of motion planning problems of autonomous cars and mobile robots, we mostly deal with a partially observable environment as the camera keeps moving forward. Combining video prediction with model based policy gradient algorithms Kaiser et al. (2020) or planning algorithms Hafner et al. (2019), improves sample efficiency of reinforcement learning algorithms by reducing the required number of episodic interactions with the environment compared to other model free methods without

Pre-registration workshop NeurIPS (2020), Vancouver, Canada.

compromising performance. Moreover, Ebert et al. (2018), Dasari et al. (2020) have recently shown us how visual predictions can aid the robot control problems, especially in unstructured environments. In the last decade the major focus of understanding spatio-temporal dynamics of video prediction was mostly confined to the case of fully observable environments with static cameras Srivastava, Mansimov, and Salakhudinov (2015), Oh et al. (2015), Vondrick, Pirsiavash, and Torralba (2015), Finn, Goodfellow, and Levine (2016), Mathieu, Couprie, and LeCun (2015), Villegas et al. (2017), Xu, Ni, and Yang (2018), Wichers et al. (2018). Most of these frameworks exploit some form of optical flow and content decomposition paradigm to generate pixel level predictions. Many times these predictions were coupled with an adversarial training in order to generate realistic images. However, with the availability of high compute power, there is a recent trend in generating high fidelity video predictions with various generative architectures such as Generative Adversarial Networks (GAN) and Variational Autoencoders (VAE) Liang et al. (2017), Denton and Fergus (2018), Babaeizadeh et al. (2018), Lee et al. (2018), Castrejon, Ballas, and Courville (2019), Gao et al. (2019), Villegas et al. (2019).

Few of these recent works, Gao et al. (2019), Villegas et al. (2019) tried to address the partial observability problem in dynamic scene prediction with the 'hallucination' powers of the generative (GAN and VAE) models. While these frameworks seem to generate realistic predictions for the moving camera problem, their accuracy comes at the cost of high on-board compute capabilities, a luxury most robotics engineer cannot afford for a small or medium scale robot.

Instead of using stochastic frameworks, We focus on addressing this problem by understanding the physics of the motion in two different inertial frames: one associated with the moving camera and the other one associated with the dynamic objects in the scene. Our framework is designed with the simple idea of understanding the relative velocity of the object as it appears in the inertial frame associated with the camera. For a motion planning problem, the realistic approximation of the scene in the background does not play any significant role. However, the performance of the motion planner or the policy generator would largely depend upon how accurately we can approximate the relative motion of the objects in the scene with respect to the camera. The objects would appear to move faster or slower than their original velocities in the image frames as the velocity of the camera influences the relative velocities of the objects. This observation led to the idea of decomposing the flow of the pixels into two different components of velocity and acceleration. Previous works Villegas et al. (2017) on decomposition of video sequences into motion and content, stopped at understanding the velocity maps or first order pixel difference maps of two consecutive frames. Those frameworks work well for fully observable scenarios where the camera is stationary. However, when the recording agent itself is dynamic, we need to decompose the motion further into the second order pixel difference maps that we refer as acceleration maps along with the velocity maps. Deterministic models often suffer from the problem of collective averaging of predicted pixels values which often result in blurry image frames compared to their stochastic counterparts. However, unlike VAEs deterministic models do not require large computational resources which makes them suitable for small scale robotic applications and this is why we propose to study the comparative performance of the proposed second order deterministic visual prediction model with stochastic frameworks such as SVAP Lee et al. (2018) and SVG Denton and Fergus (2018)

In this paper we propose to conduct a fairly extensive empirical study on the performance of our generalised physics based deterministic prediction framework and compare it to the state of the art generative architectures in the context of a generalised problem of video prediction in a partially observable environment. We also propose a new cost function that we believe will help the deep frameworks learn to reconstruct the velocity and acceleration maps associated with each video frame.

2 Our Approach

For a generalised set up where the camera is mounted on a dynamic platform moving on a smooth trajectory, the relative velocity vector of any object appearing on the camera frame keeps getting modified. We intend to capture the dynamics of this changing relative velocity vector with a first order pixel difference or velocity map and a second order pixel difference map that we call acceleration map. This is why we need 3 consecutive image frames $(\mathbf{x}_t, \mathbf{x}_{t-1}, \mathbf{x}_{t-2})$ at timestep $t, t-1$ and $t-2$ to make prediction of the future frame $\mathbf{x}_{t+1}$ at timestep $t+1$, where $\mathbf{x}_t \in \mathbb{R}^{w \times h \times c}$ represents the image frame at time t with dimension $w \times h \ \times c$. Due to the physics based design of our framework our network is highly interpretable. Our framework can be thought of as the next generation and more improved version of the Motion and Content network (MCNet) proposed by Villegas et al.

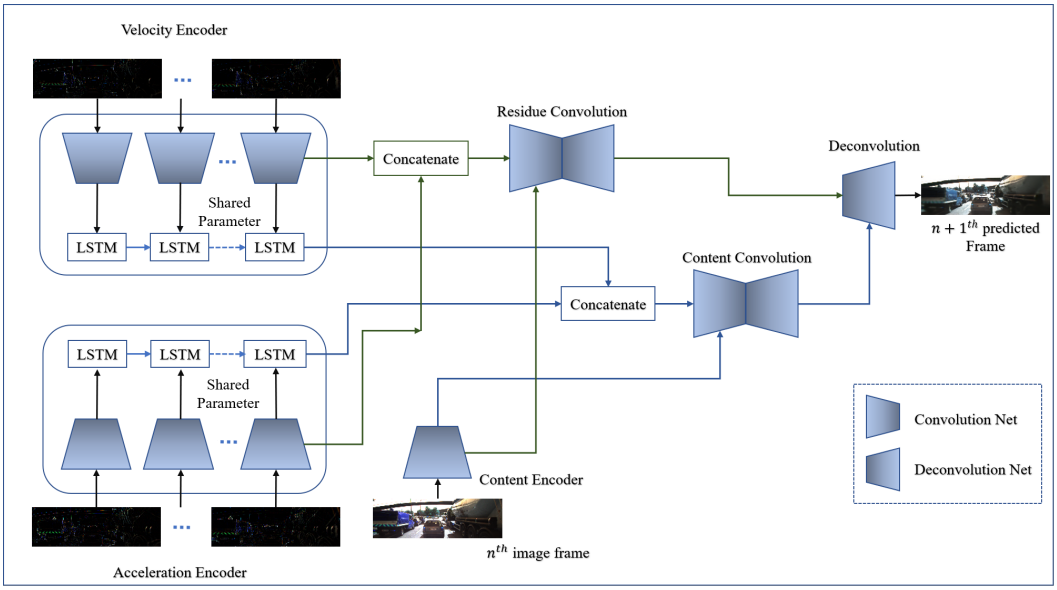

Figure 1: Architecture of VANet while being trained on the KITTI raw dataset. The network learns the temporal dependencies from the velocity and acceleration encoders which takes the first and second order pixel difference maps as inputs, respectively. The content encoder takes the last or n^{th} frame as input to encode the spatial information. Content convolution network combines the spatial encoding with the motion features. Similarly, the residues from the content, velocity and acceleration encoders are fused together in the residue convolution network. Finally, the decoder generates the predicted future frame.

(2017). While MCNet pioneered the idea of disentangling first order pixel difference map from images sequences with a motion encoder for unsupervised video prediction, we further generalised it to incorporate the complex interactions between the dynamics of the camera and object inertial frames. The entire algorithm which we refer to as the Velocity Acceleration Network (VANet) as shown in figure 1, can be decomposed into the following components:

- **Velocity Encoder:** The velocity encoder (f^{vel}), parameterised with θ^{vel}, is designed to capture the temporal dependencies embedded in the velocity map of two consecutive image frames, $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ at time t and $t-1$, respectively. This network takes the velocity map $\mathbf{v}_t = (\mathbf{x}_t - \mathbf{x}_{t-1}) \in \mathbb{R}^{w \times h \times c}$ at time t as input and maps them into two tensors: velocity feature encoding $\mathbf{v}_t^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ and the memory cell state $\mathbf{c}_t^{vel} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ at time t as:
$$(\mathbf{v}_t^{en}, \mathbf{c}_t^{vel}) = f^{vel}(\mathbf{v}_t, \mathbf{v}_{t-1}^{en}, \mathbf{c}_{t-1}^{vel}; \theta^{vel}) \tag{1}$$
The memory cell state $\mathbf{c}_t^{vel}$ captures the temporal structure embedded in the velocity maps of $\mathbf{v}_{1:t}$. f^{vel} is designed with convolutional LSTM Shi et al. (2015) networks and, in essence, embeds the velocity component of the pixel space into a low dimensional spatio-temporal feature space.
- **Acceleration Encoder:** The acceleration encoder (f^{acc}) parameterised with θ^{acc} is designed in the same image as of the velocity encoder with only difference of capturing the temporal dependencies embedded in the acceleration map of two consecutive velocity maps, $\mathbf{v}_t$ and $\mathbf{v}_{t-1}$ at time t and $t-1$ respectively. This network takes acceleration map $\mathbf{a}_t = \mathbf{v}_t - \mathbf{v}_{t-1} \in \mathbb{R}^{w \times h \times c}$ as input and generates two tensors: acceleration feature encoding $\mathbf{a}_t^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ and the memory cell state $\mathbf{c}_t^{acc} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ at time t as follows:
$$(\mathbf{a}_t^{en}, \mathbf{c}_t^{acc}) = f^{acc}(\mathbf{a}_t, \mathbf{a}_{t-1}^{en}, \mathbf{c}_{t-1}^{acc}; \theta^{acc}) \tag{2}$$
The memory cell state $\mathbf{c}_t^{acc}$ captures the temporal structure embedded in the acceleration maps of $\mathbf{a}_{1:t}$. This encoder is also designed with convolutional LSTM networks and maps the acceleration component of the pixel space into a low dimensional spatio-temporal feature space.

- **Content Encoder:** The content encoder (f^{con}) parameterised with θ^{con} is designed to encapsulate the spatial information embedded in the latest image frame $\mathbf{x}_t$ with a convolutional neural network. The idea here is to map the high dimensional image frames $\mathbf{x}_t \in \mathbb{R}^{w \times h \times c}$ into a low dimensional spatial feature embedding $\mathbf{x}_t^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$. Mathematically, it can be represented as:

$$\mathbf{x}_t^{en} = f^{con}(\mathbf{x}_t; \theta^{con}) \tag{3}$$

- **Content Convolution Network:** This is the part where we start combining the spatial encoding $\mathbf{x}_t^{en}$ coming from the content encoder network with the motion encoding of $\mathbf{v}_t^{en}$ and $\mathbf{a}_t^{en}$. We first combine velocity and acceleration encoding, $[\mathbf{v}_t^{en}, \mathbf{a}_t^{en}] \in \mathbb{R}^{w^{'} \times h^{'} \times 2c^{'}}$, through convolution operations to create the final relative velocity encoding $\mathbf{v}_{rel_t}^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ as:

$$\mathbf{v}_{rel_t}^{en} = f^{motion}([\mathbf{v}_t^{en}, \mathbf{a}_t^{en}]; \theta^{motion}) \tag{4}$$

  We then combine the relative velocity encoding $\mathbf{v}_{re_tl}^{en}$ with the spatial feature encoding tensor $\mathbf{x}_t^{en}$ with layers of convolution operation and generate the spatio-temporal feature embedding for the next time-step, $\hat{\mathbf{x}}_{t+1}^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ given as

$$\hat{\mathbf{x}}_{t+1}^{en} = f^{conv}([\mathbf{x}_t^{en}, \mathbf{v}_{rel_t}^{en}]; \theta^{conv}) \tag{5}$$

  Here, f^{motion} and f^{conv} are both designed using CNN having bottle-neck architecture (Hinton and Salakhutdinov (2006)), that first projects tensor pairs $[\mathbf{v}_t^{en}, \mathbf{a}_t^{en}]$ and $[\mathbf{x}_t^{en}, \mathbf{v}_{rel_t}^{en}]$ into a low dimensional feature space and then pull back to the original feature space of $w^{'} \times h^{'} \times c^{'}$.

- **Residue Convolution Network:** The idea of temporal transformation of the residues generated from the f^{con} in order to compensate for the loss of information from mapping the high dimensional image frames $\mathbf{x}_t \in \mathbb{R}^{w \times h \times c}$ to a low dimensional feature space $\mathbf{x}_t^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ was first introduced in Villegas et al. (2017). We carry forward the same idea of multi-scale motion-content residue network but with our modified relative residue velocity encoding $[\check{\mathbf{v}}_{rel_t}^{en}]^i \in \mathbb{R}^{w^i \times h^i \times c^i}$ at layer i given as:

$$[\check{\mathbf{v}}_{rel_t}^{en}]^i = f_{res}^{motion}([\check{\mathbf{v}}_t^{en}, \check{\mathbf{a}}_t^{en}]; \theta_{res}^{motion})^i \tag{6}$$

  where, $[\check{\mathbf{v}}_t^{en}]^i$ and $[\check{\mathbf{a}}_t^{en}]^i$ are the residue velocity and acceleration encoding from the i^{th} layer of f^{vel} and f^{acc}, respectively. The relative residue velocity encoding $[\check{\mathbf{v}}_{rel_t}^{en}]^i$ is then combined with the content residue $[\check{\mathbf{x}}_t^{en}]^i$ generated from the i^{th} layer of f^{con} as:

$$[\mathbf{r}_{t+1}^{en}]^i = f_{res}^{conv}([\check{\mathbf{x}}_t^{en}, \check{\mathbf{v}}_{rel_t}^{en}]; \theta_{res}^{conv}) \tag{7}$$

  Like the content convolution network, f_{res}^{motion} and f_{res}^{conv} also uses the CNN bottle-neck architecture to combine the tensor pair of $[\check{\mathbf{v}}_t^{en}, \check{\mathbf{a}}_t^{en}]$ and $[\check{\mathbf{x}}_t^{en}, \check{\mathbf{v}}_{rel_t}^{en}]$.

- **Decoder:** Finally we up-pool the spatio-temporal feature embedding $\hat{\mathbf{x}}_{t+1}^{en}$ and combine it with the residual encoding of $[\mathbf{r}_{t+1}^{en}]^i$ in a layer wise manner to generate the final prediction of $\tilde{\mathbf{x}}_{t+1}$. The decoder network g^{dec} is responsible to map the reduced dimensional $\hat{\mathbf{x}}_{t+1}^{en} \in \mathbb{R}^{w^{'} \times h^{'} \times c^{'}}$ back into the high dimensional pixel level representation of $\tilde{\mathbf{x}}_{t+1} \in \mathbb{R}^{w \times h \times c}$ which is same as the original image frames.

$$\tilde{\mathbf{x}}_{t+1} = g^{dec}([\hat{\mathbf{x}}_{t+1}^{en}, \mathbf{r}_{t+1}^{en}; \theta^{dec}) \tag{8}$$

  Where, $\mathbf{r}_{t+1}^{en}$ is a list of all residual encoding from f_{res}^{conv} from all its layers. the decoder, g^{dec} uses deconvolutional neural networks Zeiler, Taylor, and Fergus (2011) which basically consists of multiple successive operations of deconvolution, rectification and unpooling. The residual embeddings from f_{res}^{conv} is combined via the connection between the Residual Convolution Network and the decoder in a layer-wise manner. The final output layer is passed through a $tanh$ non-linearity.

3 Inference and Training

3.1 Inference of multi-time step prediction:

In section 2 we discussed how to make prediction for immediate future frame $\mathbf{x}_{t+1}$ at time-step t with image frames $\mathbf{x}_t$, $\mathbf{x}_{t-1}$ and $\mathbf{x}_{t-2}$ at timesteps $\{t, t-1$ and $t-2\}$ and velocity maps $\mathbf{v}_t$ and $\mathbf{v}_{t-1}$. However, for multi-time step prediction, our network observes the velocity and acceleration maps for last n frames as the difference between image frames $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ and velocity maps $\mathbf{v}_t$ and $\mathbf{v}_{t-1}$ where $t \in \{2, n\}$ and we assume $\mathbf{x}_1$ is the first observed frame. From this history of past n frames the velocity and acceleration encoders learns the relative pixel dynamics of the scene and then the final frame $\mathbf{x}_n$ is given as input to the Content Encoder. The network then transforms $\mathbf{x}_n$ into $\tilde{\mathbf{x}}_{t+1}$ with the learned dynamics features.For $t \in [n+1, n+T]$ where T is the desired number of prediction steps, VANet starts using its own prediction as input to generate the velocity and acceleration maps.

3.2 Training and Loss function:

Since, VANet shares many structural similarities with the architecture of MCNet in Mathieu, Couprie, and LeCun (2015), we also divided our loss function $\mathcal{L}$ into 2 major sub-loss functions as:

$$\mathcal{L} = \alpha\mathcal{L}_{image} + \beta\mathcal{L}_{adv} \tag{9}$$

where, $\mathcal{L}_{image}$ and $\mathcal{L}_{adv}$ constitutes the image loss and loss from adversarial training respectively, and $\alpha, \beta \in \mathbb{R}^+$. We further subdivide $\mathcal{L}_{image}$ into two components: (i) reconstruction loss $\mathcal{L}_{recon}$ and (ii) the total gradient difference loss $\mathcal{L}_{TGDL}$ given as:

$$\mathcal{L}_{image} = \mathcal{L}_{recon} + \mathcal{L}_{TGDL} \tag{10}$$

The reconstruction loss: $\mathcal{L}_{recon}$ is the total L_p norm distance between the ground truth image frame $\mathbf{x}_{n+i}$ and predicted future frames $\tilde{\mathbf{x}}_{t+i}$ for $i \in \{1, T\}$ averaged over the entire dataset of $D = \{x^i_{1,..,n,n+1,...,n+T}\}_{i=1}^N$. p can be 1 or 2, and is given by:

$$\mathcal{L}_{recon}(\mathbf{x}_{n+1:n+T}, \tilde{\mathbf{x}}_{n+1:n+T}) = \sum_{i=n+1}^{n+T} \|\mathbf{x}_{n+i} - \tilde{\mathbf{x}}_{n+i}\|^p \tag{11}$$

We introduce the total gradient difference loss $\mathcal{L}_{TGDL}$ which is further divided into gradient difference loss from the predicted image frames: $\mathcal{L}_{GDL}$ and velocity gradient difference loss from the velocity maps generated from the predicted image frames: $\mathcal{L}_{VGDL}$. This is expressed as:

$$\mathcal{L}_{TGDL} = \mathcal{L}_{GDL} + \mathcal{L}_{VGDL} \tag{12}$$

$$\begin{aligned}\mathcal{L}_{GDL}(\mathbf{x}, \tilde{\mathbf{x}}) = \sum_{t=n+1}^{n+T} \sum_{i,j}^{w,h} ||\mathbf{x}_{t,i,j} - \ \mathbf{x}_{t-1,i,j}| - |\tilde{\mathbf{x}}_{t,i,j} - \tilde{\mathbf{x}}_{t-1,i,j}||^\lambda + \\ + ||\mathbf{x}_{t,i,j} - \mathbf{x}_{t,i-1,j}| - |\tilde{\mathbf{x}}_{t,i,j} - \tilde{\mathbf{x}}_{t,i-1,j}||^\lambda + ||\mathbf{x}_{t,i,j} - \mathbf{x}_{t,i,j-1}| - |\tilde{\mathbf{x}}_{t,i,j} - \tilde{\mathbf{x}}_{t,i,j-1}||^\lambda\end{aligned} \tag{13}$$

$\mathcal{L}_{GDL}$ in Eq. (12) is similar to the $\mathcal{L}_{gdl}$ loss in Villegas et al. (2017) in that it gives an average of the gradient difference loss between the predicted frames and ground truth. However, unlike Villegas et al. (2017), we also add the component of temporal difference loss: $||\mathbf{x}_{t,i,j} - \mathbf{x}_{t-1,i,j}| - |\tilde{\mathbf{x}}_{t,i,j} - \tilde{\mathbf{x}}_{t-1,i,j}||^\lambda$ to the expression of $\mathcal{L}_{GDL}$ in Eq. (13), so that the velocity encoder can learn the pixel dynamics more efficiently. Here, λ can be chosen to be 1 or 2.

$$\begin{aligned}\mathcal{L}_{VGDL}(\mathbf{x}, \tilde{\mathbf{x}}) = \sum_{t=n+1}^{n+T} \sum_{i,j}^{w,h} ||\mathbf{v}_{t,i,j} - \ \mathbf{v}_{t-1,i,j}| - |\tilde{\mathbf{v}}_{t,i,j} - \tilde{\mathbf{v}}_{t-1,i,j}||^\lambda + \\ + ||\mathbf{v}_{t,i,j} - \mathbf{v}_{t,i-1,j}| - |\tilde{\mathbf{v}}_{t,i,j} - \tilde{\mathbf{v}}_{t,i-1,j}||^\lambda + ||\mathbf{v}_{t,i,j} - \mathbf{v}_{t,i,j-1}| - |\tilde{\mathbf{v}}_{t,i,j} - \tilde{\mathbf{v}}_{t,i,j-1}||^\lambda\end{aligned} \tag{14}$$

The expression for velocity gradient difference loss $\mathcal{L}_{VGDL}$ given in Eq. (14) is similar to that of $\mathcal{L}_{GDL}$ in Eq. (13) with the replacement of $\mathbf{x}$ and $\tilde{\mathbf{x}}$ with the ground truth velocity maps $\mathbf{v}$ and predicted velocity maps $\tilde{\mathbf{v}}$, respectively. The $\mathcal{L}_{VGDL}$ loss is designed so that the acceleration encoder can disentangle and approximate the motion of the pixel dynamics to the second order.

Due to the averaging effect to the reconstruction loss $\mathcal{L}_{image}$ and the blurring effects introduced with the convolution operations, we add an adversarial loss $\mathcal{L}_{adv}$ to our total loss $\mathcal{L}$ in Eq. (9). Similar to Mathieu, Couprie, and LeCun (2015), $\mathcal{L}_{adv}$ is defined as:

$$\mathcal{L}_{adv} = -\log D([\mathbf{x}_{1:n}, G(\mathbf{x}_{1:n})]) \tag{15}$$

where, $[\mathbf{x}_{1:n}]$ is the concatenation of all input images and $G(\mathbf{x}_{1:n}) = [\tilde{\mathbf{x}}_{n+1:n+T}]$ generates the concatenation of all the predicted future images and $[\mathbf{x}_{n+1:n+T}]$ is the concatenation of all ground truth images. $D(.)$ is the output from the discriminator network which is trained with the loss function as:

$$\mathcal{L}_{dec} = \log D([\mathbf{x}_{1:n}, [\mathbf{x}_{n+1:n+T}]) - \log(1 - D([\mathbf{x}_{1:n}, G(\mathbf{x}_{1:n})])) \tag{16}$$

4 Experimental Setup and Methodology

In order to test the proposed framework, we plan to test it at 3 different task objective following similar empirical analysis done by Villegas et al. (2019). We plan to evaluate the performance of the network with respect to the structural integrity of the predicted frames with respect to the ground truth. We plan to conduct rigorous studies using five different metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), VGG Cosine Similarity, Fréchet Video Distance (FVD). We also plan to conduct a comparative analysis of the relative computational load of VANet with respect to other SOTA models such as SAVP Lee et al. (2018) and SVG Denton and Fergus (2018) and MCNet Villegas et al. (2017). While SVAP and SVG are computationally heavy generative models, MCNet is deterministic and shares a lot of similarity with the architecture if VANet.

1. **Object Interaction** We would like to evaluate our network on the BAIR robot push datset given by Ebert et al. (2017) in order to evaluate its performance in different object interaction tasks. This dataset represents the interaction between different objects and a manipulator. We anticipate that due to the stochastic nature of the interaction between the objects and the manipulator, our deterministic model might face some limitation in long term prediction tasks. However, this type of detailed comparative analysis would help us establish a baseline for the limitation of deterministic video prediction networks in comparison to their stochastic neighbors like SAVP Lee et al. (2018) or SVG Villegas et al. (2019).
2. **Structured Motion** One of the well established structured motion prediction datasets is the KTH Schuldt, Laptev, and Caputo (2004) human action dataset. Although this dataset is recorded in a fully observable setting, tracking the complex human motion for long duration is a daunting task. Thus, comparing the performance of VANet with SAVP, SVG and MCNet will establish how our proposed framework performs in comparison to the state of the art generative as well as deterministic methods. The analysis of the comparative performance between VANet and MCNet would of particular interest here as it would help us understand whether the proposed generalised approach toward decomposing motion into velocity and acceleration maps provides any improvement or not.
3. **Partial Observability** Given the primary objective of the proposed VANet is to generate predictions of object motion in a partially observable scenario, we plan to focus majority of our testing and comparative analysis on this type of tasks. Right now, the only dataset that provides the scope of analysing our network's performance in a partially observable scenario, is KITTI Geiger et al. (2013) dataset. Here, the video is recorded from a camera fixed on the dashboard of a moving car. Thus, the background of the images keeps getting updated and creates complex interactions between the moving objects on the streets and the relative velocity of the camera.

4.1 Ablation Study

We also plan to conduct a thorough ablation study on our proposed framework by removing various components from it. If we remove the acceleration encoder from the network we can analyse whether that would affect the network's overall performance in making long term predictions for both fully observable and partially observable cases. Due to modular nature of our proposed loss function we can further simplify the loss function by turning off the effect of $\mathcal{L}_{VGDL}$ and $\mathcal{L}_{GDL}$ from propagating through the network. Similarly, we also plan to study the effect of information loss from the convolution operations on the network by training VANet only on the reconstruction loss of $\mathcal{L}_{adv}$ by removing the adversarial loss $\mathcal{L}_{adv}$ from $\mathcal{L}$.

References

Babaeizadeh, M.; Finn, C.; Erhan, D.; Campbell, R. H.; and Levine, S. 2018. Stochastic variational video prediction.

Bubic, A.; Cramon, D. Y. V.; and Schubotz, R. 2010. Prediction, cognition and the brain. *Frontiers in Human Neuroscience* 4.

Castrejon, L.; Ballas, N.; and Courville, A. 2019. Improved conditional vrnns for video prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Dasari, S.; Ebert, F.; Tian, S.; Nair, S.; Bucher, B.; Schmeckpeper, K.; Singh, S.; Levine, S.; and Finn, C. 2020. Robonet: Large-scale multi-robot learning.

Denton, E., and Fergus, R. 2018. Stochastic video generation with a learned prior.

Ebert, F.; Finn, C.; Lee, A.; and Levine, S. 2017. Self-supervised visual planning with temporal skip connections. In *Conference on Robot Learning (CoRL)*.

Ebert, F.; Finn, C.; Dasari, S.; Xie, A.; Lee, A. X.; and Levine, S. 2018. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *CoRR* abs/1812.00568.

Finn, C., and Levine, S. 2017. Deep visual foresight for planning robot motion. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2786–2793.

Finn, C.; Goodfellow, I.; and Levine, S. 2016. Unsupervised learning for physical interaction through video prediction. In *Proc.. of Thirtieth Conference on Neural Information Processing Systems*, NIPS '16, 64–72.

Gao, H.; Xu, H.; Cai, Q.-Z.; Wang, R.; Yu, F.; and Darrell, T. 2019. Disentangling propagation and generation for video prediction. In *In Proc.. of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*.

Hafner, D.; Lillicrap, T.; Fischer, I.; Villegas, R.; Ha, D.; Lee, H.; and Davidson, J. 2019. Learning latent dynamics for planning from pixels.

Hinton, G. E., and Salakhutdinov, R. R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Kaiser, L.; Babaeizadeh, M.; Milos, P.; Osinski, B.; Campbell, R. H.; Czechowski, K.; Erhan, D.; Finn, C.; Kozakowski, P.; Levine, S.; Mohiuddin, A.; Sepassi, R.; Tucker, G.; and Michalewski, H. 2020. Model-based reinforcement learning for atari.

Lee, A. X.; Zhang, R.; Ebert, F.; Abbeel, P.; Finn, C.; and Levine, S. 2018. Stochastic adversarial video prediction.

Liang, X.; Lee, L.; Dai, W.; and Xing, E. P. 2017. Dual motion gan for future-flow embedded video prediction.

Mathieu, M.; Couprie, C.; and LeCun, Y. 2015. Deep multi-scale video prediction beyond mean square error. *CoRR* abs/1511.05440.

Oh, J.; Guo, X.; Lee, H.; Lewis, R. L.; and Singh, S. 2015. Action-conditional video prediction using deep networks in atari games. In *Proc.. of the Twenty-ninth International Conference on Neural Information Processing Systems*, NIPS'15, 2863–2871.

Schuldt, C.; Laptev, I.; and Caputo, B. 2004. Recognizing human actions: a local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, volume 3, 32–36 Vol.3.

Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; kin Wong, W.; and chun WOO, W. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Proc.. of the Twenty-ninth International Conference on Neural Information Processing Systems*, NIPS'15, 802–810.

Srivastava, N.; Mansimov, E.; and Salakhudinov, R. 2015. Unsupervised learning of video representations using lstms. In *Proc.. of Thirty-second International Conference on Machine Learning*, ICML '15, 843–852.

Villegas, R.; Yang, J.; Hong, S.; Lin, X.; and Lee, H. 2017. Decomposing motion and content for natural video sequence prediction. *CoRR* abs/1706.08033.

Villegas, R.; Pathak, A.; Kannan, H.; Erhan, D.; Le, Q. V.; and Lee, H. 2019. High fidelity video prediction with large stochastic recurrent neural networks. In *In Proc.. of the* 32^{nd} *Advances in Neural Information Processing Systems*. 81–91.

Vondrick, C.; Pirsiavash, H.; and Torralba, A. 2015. Anticipating the future by watching unlabeled video. *CoRR* abs/1504.08023.

Wichers, N.; Villegas, R.; Erhan, D.; and Lee, H. 2018. Hierarchical long-term video prediction without supervision. In *Proc.. of the* 35^th *International Conference on Machine Learning, (ICML)*, PMLR, 80, 6038–6046.

Xu, J.; Ni, B.; and Yang, X. 2018. Video prediction via selective sampling. In *Proc.. of the Thirty-second Conference on Neural Information Processing Systems*, NIPS'18, 1705–1715.

Zeiler, M. D.; Taylor, G. W.; and Fergus, R. 2011. Adaptive deconvolutional networks for mid and high level feature learning. In *In Proc.. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2018–2025.