Chapter 3
APPROACH

Rather than struggle in high-dimension pixel space, our idea is to model in transformation space for video imagination. In our framework, firstly, we send latent variable and condition code encoded from image into **transformation generator**, which outputs a group of transformation sequences. Secondly, we apply those transformation sequences to the original image and recon- struct frames through a **volumetric merge network**. Finally, we combine frames as an imaginary video then use **video critic network** to achieve adversarial training.

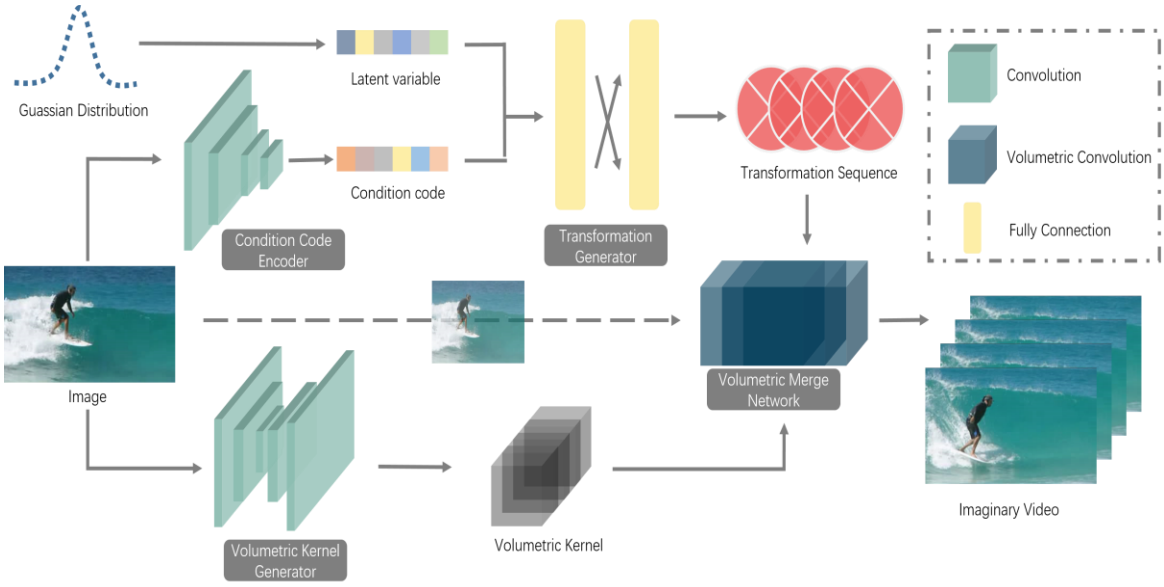

Figure 2: Pipeline of video imagination from single image. To produce one imaginary video, the input image is first encoded into a condition code and sent to transformation generator together with a latent variable. The generated transformation sequence is applied to input image later in volumetric merge network where frames are reconstructed with transformed images and volumetric kernels. Those four frames form one imaginary video.

Problem definition

Firstly we use formulations to describe this task: Given an image X, outputs m imaginary videos V' corresponding to different reasonable motions. Each imaginary video contains T consecutive frames f_T.

Ideally, we would like to model the distribution $P(V \mid X)$ of all possible imaginary V given X . Practically, we aim to train a neural network T_θ with parameters θ, which implicitly models a distribution $P_T(V \mid X)$. Through training, we expect $P_T(V \mid X)$ to converge to a good estimate of $P(V \mid X)$. $T_\theta(X)$ yields a sample V' drawn from $P_T(V \mid X)$, so we have

$$V' = T_\theta(X) \sim P_T(V \mid X) \tag{1}$$

Instead of directly modeling in pixel space, we choose to model distribution in transformation space. We build this model based on a key assumption that the major motions between frames can be modeled by transformations. That means letting M_T denote motion between X and f_T, M_T can be represented by a transformation sequence Φ_T containing p transformations. Letting $\odot$ denote the operation of apply transformation sequence to image, we have $f_T = \Phi_T \odot X$. Letting Φ represent the group of transformation sequences of all videos, we have $V = \Phi \odot X$. By introducing G implicitly modeling $P_G(\Phi \mid X)$ in transformation space, we have a new description of target:

$$V' = G_\theta(X) \odot X \sim P_T(\Phi \odot X \mid X) = P_G(\Phi \mid X) \odot X \tag{2}$$

To make diversity samplings of V feasible, we introduce latent variable z that follows a specific distribution (e.g. Guassian Distribution). Hence, we can modify target of Gθ from modelling the distribution PG(Φ|X) to modelling PG(Φ|X,z). This implicit distribution allow us to sample different imaginary videos V' through sampling different z. Therefore, everything reduces to the following target:

$$V' = G_\theta(X,z)X \sim P_G(\Phi \mid X,z) \odot X \tag{3}$$

Transformation Generator

The job of transformation generator is implicitly modeling $PG\ (\Phi|X\ ,\ z)$ so that it can generate transformation conditioned on image. Given the condition code of a static image X, together with a latent variable z, the goal of transformation generator is learning to generate a transformation group Φ.

To be specific, transformation generator outputs T transformation sequences $\{\Phi_1, \Phi_2, \ldots, \Phi_T\}$ corresponding to transformations between X and $\{f_1, f_2, \ldots, f_T\}$. Each transformation sequence Φ_T contains P transformations formed by K parameters. Transformations are generated in a sequential fashion in hope of a be1er description of warp motion, because motion can often be decomposed in a layer-wise manner.

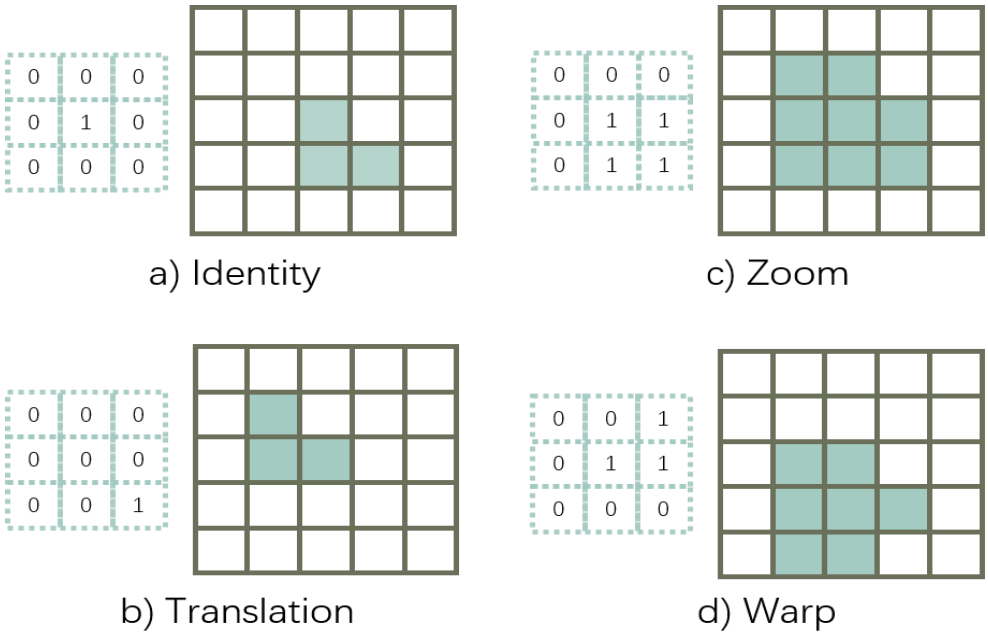

Figure 3: Different convolution kernels result in different motions. The doted square denotes convolution kernel and the right side image shows the result of applying the kernel. One simple kernel can model motion like Identity, Translation ,Zoom, Warp.

We use two kinds of transformations to model motion. Since in adversarial training, the gradient back-propagation starts from critic network then flows to the frames, the transformation type we choose needs to allow gradient propagating from transformed images to transformation generator. Fortunately, prior works in [5, 12] revealed that there is a group of transformations having this adorable a1ribution. We build two distinct models to form Φ based on prior works.

Affine Transformation: Simply formed by 6 parameters, affine transformation can model motions including translation, rotation, zoom, and shear. Works in [3, 36] have shown that affine trans- formation provides a good approximation of 3-D moving objects motion. Affine transformation works on coordinates, which would raise a problem of undefined pixel locations. In practice, we use differentiable bilinear interpolation to complete those pixels.

Convolutional Transformation: A convolution kernel can naturally model simple motions like translation, zoom and warp as shown in Figure 3. The kernel size can vary with application scene. For example, a 5*5 kernel allows pixels translating over a distance of 2 pixels.

Volumetric Merge Network

Volumetric merge network is responsible for reconstructing frames $\{f1, f2,\ldots, fT\}$ based on the generated transformation Φ and image X . The transformation group Φ is finally applied to image X, producing an intermediate image group I consisting of T intermediate image sequences $\{I1, I2,\ldots, IT\}$ that will be used to reconstruct $\{f1, f2,..,fT\}$ accordingly. Combining frames temporally, volumetric merge network outputs imaginary video V ‘.

Since the transformation is generated in a sequential fashion, it is intuitive to take the

sequence of intermediate images as an extended dimension representing transformation. That is, we consider each transformed sequence IT as one entity IT $R^{W \times H \times P}$ that has 3 dimensions as width W, height H, and transformations P. This 3-D entity, as shown in Figure 4, allows us to reconstruct frame by merging it in a volumetric way. Each pixel is reconstructed through volumetric kernels. The kernels can take both neighbor pixel values and intermediate image differences into consideration.

Parameters in volumetric kernels can be obtained either from clipping a crop of generated transformations or through a specific volumetric kernel generator as shown in Figure 2. Volumetric kernel generator (a full convolution network) concentrates more on capturing the dependency in spatial domain, while generated transformations can give volumetric kernel be1er understanding of correlation between intermediate images.

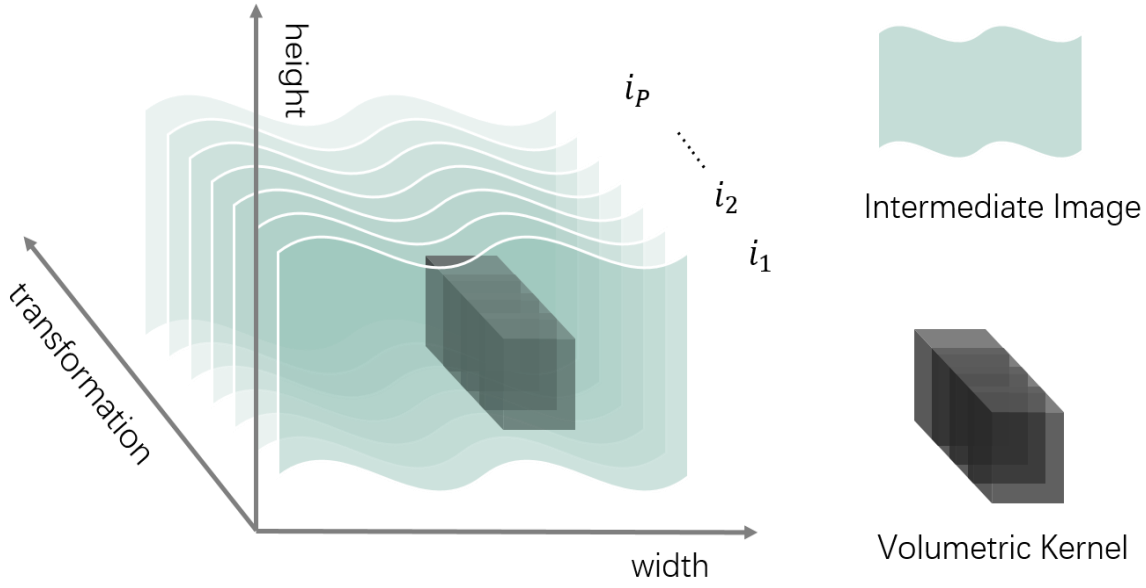

Figure 4: Intermediate image sequence IT as 3-D entity. A volumetric kernel can take both neighbour pixel values and intermediate image differences into consideration.

Video Critic Network

To meet the requirement of a better criterion, we design a video critic network *Critic* to achieve adversarial training. Video critic network *Critic* receives synthesized video $\hat{V}$ and real video V as input alternatively, and outputs criticism judging how convincing the input is.

A convincing video means that the frame looks clear and the motion between frames seems consecutive and reasonable. Video critic network *Critic* needs to give reference of whether the input is plausible and realistic, which requires understanding of both static appearance and dynamic scene. The similar requirement can be found in action recognition task, where lately researchers have made progress [30]. We draw inspiration from those works, and design *Critic* to have the structure of spatial-temporal convolution networks [13].

Learning and Implementations

Our framework consists of fully feed-forward networks. The trans- formation generator consists of 4 fully connected layers. The latent code sampled from a guassian distribution has 100 dimensions, and the condition code has 512 dimensions. We can encode X into condition code either through refined AlexNet or a 5 layer convolutional network. The volumetric merge network consists of 3 volumetric convolutional layers, while the last layer uses element-wise kernel. We use a five-layer spatio-temporal convolutional network as the critic network.

We employ Wasserstein GAN [2] to train our framework. The generator loss Lg is defined as:

$$Lg = -\mathrm{E}_{V \sim PT(V|X)} Critic(v) \tag{4}$$

The critic loss L_C is defined as:

$$L_C = \mathrm{E}_{V \sim PT(V|X)} C(v) - \mathrm{E}_{V \sim P(V|X)} Critic(v) \quad (5)$$

Alternatively, we minimize the loss L_g once after minimizing the loss L_d 5 times until a fixed number of iterations. Ultimately, the optimal video critic network C is hoped to produce good estimate of Earth-Mover (EM) distance between $P(V / X)$ and $PT(V/X)$. We use the RMSProp optimizer and a fixed learning rate of 0.00005. ReLU activation functions and batch normalization are also employed.

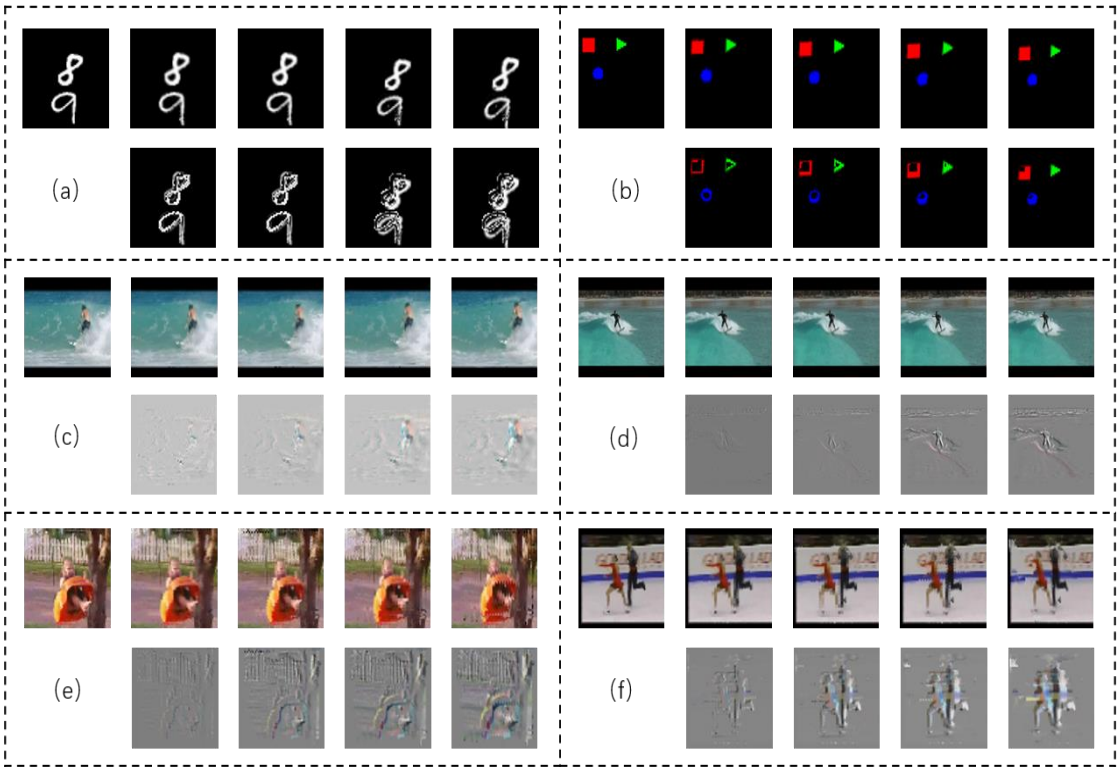

Figure 5: Quality Performance of our framework. In each doted box, the first shows the synthesized imaginary videos given the fist frame as input. The second row shows the difference images of synthesized frames and input. (a)(b) demonstrate the results experiment on moving MNIST and 2D shapes dataset. (c)(d) shows the result of surfing class on UCF101 dataset in different resolution as c) 64*64 and d)128*128. (e)(f) shows the results given image from swing and ice-dancing categories in UCF101 dataset.