Generative Adversarial Network-Based Frame Extrapolation for Video Coding

Jianping Lin, Dong Liu, Houqiang Li, Feng Wu
CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System,
University of Science and Technology of China, Hefei 230027, China
ljp105@mail.ustc.edu.cn, {dongeliu,lihq,fengwu}@ustc.edu.cn

***Abstract*—Motion estimation and motion compensation are fundamental in video coding to remove the temporal redundancy between video frames. The current video coding schemes usually adopt block-based motion estimation and compensation using simple translational or affine motion models, which cannot efficiently characterize complex motions in natural video signal. In this paper, we propose a frame extrapolation method for motion estimation and compensation. Specifically, based on the several previous frames, our method directly extrapolates the current frame using a trained deep network model. The deep network we adopted is a redesigned Video Coding oriented LAplacian Pyramid of Generative Adversarial Networks (VC-LAPGAN). The extrapolated frame is then used as an additional reference frame. Experimental results show that the VC-LAPGAN is capable in estimating and compensating for complex motions, and extrapolating frames with high visual quality. Using the VC-LAPGAN, our method achieves on average 2.0% BD-rate reduction than High Efficiency Video Coding (HEVC) under low-delay P configuration.**

***Index Terms*—Frame extrapolation, Laplacian pyramid of generative adversarial networks (LAPGAN), motion compensation, motion estimation, video coding.**

I. Introduction

In natural video signal, successive frames are highly correlated. By exploiting this correlation, the temporal redundancy existing in the video signal can be substantially removed. Inter (frame) prediction refers to a large class of techniques to exploit the correlation between frames. In video coding, inter prediction is usually performed by motion estimation and motion compensation. In the state-of-the-art video coding standard–High Efficiency Video Coding (HEVC) [1], block-based motion estimation and compensation is adopted with simple translational motion model. That is, one block is assumed to have a translation between two frames. This motion model has the advantage of computational efficiency, but is too simple to characterize complex motions in natural videos.

High-order motion models have been investigated for video coding for a long time. In [2], a general approach to block matching-based motion estimation is proposed. Recently, affine motion models attracted the attention of researchers. A four-parameter affine motion model is presented in [3] and reportedly achieves significant bits saving for video sequences that have rotation or zooming. Nonetheless, high-order motion models like affine have a dilemma about the model complexity: if the model is too simple, it cannot characterize complex motions; but if the model is too complex, it may incur too much overhead bits to signal the model parameters, and may cause very high computational complexity for motion estimation.

Recently, deep learning has achieved a series of successes in computer vision and image processing. One noticeable work is generative adversarial network (GAN), firstly proposed in [4], which has the capability to generate plausible images with high visual quality. Later on, Denton *et al.* proposed a Laplacian pyramid of GANs to generate images in a coarse-to-fine fashion [5]. In [6], a similar multi-scale network was proposed to predict the future frames of a video sequence. These works demonstrate the feasibility of using GANs to extrapolate video frames, which provides a new approach to motion estimation and compensation. However, to the best of our knowledge, no work has been done to investigate the performance of GAN-based frame extrapolation in video coding.

In this paper, we study a frame extrapolation method for motion estimation and compensation. Take the low-delay configuration as example, before encoding the n-th frame, we can use several previous reconstructed frames, say the $(n-1)$-th, $(n-2)$-th, ..., to extrapolate a new frame using a trained deep network model. In this paper, the network we adopted is a redesigned **V**ideo **C**oding oriented **LA**placian **P**yramid of **G**enerative **A**dversarial **N**etworks (VC-LAPGAN), and we use four previous frames for extrapolation. We expect the VC-LAPGAN can identify the motion patterns in the four previous frames, so as to predict the current frame. Due to the high flexibility of deep network, we expect the VC-LAPGAN may be able to handle complex motions. We observe that, indeed, the VC-LAPGAN is capable in estimating and compensating for complex motions in natural videos, and in extrapolating frames with high visual quality. Therefore, we integrate VC-LAPGAN into the HEVC scheme, use the extrapolated frame as an additional reference. Experimental results show that our method leads to on average 2.0% BD-rate reduction under low-delay P configuration.

This work was supported by the National Program on Key Basic Research Projects (973 Program) under Grant 2015CB351803, by the Natural Science Foundation of China under Grants 61772483, 61331017, 61390512, and 61425026. *(Corresponding author: Dong Liu.)*

978-1-5386-4458-4/18/$31.00 ©2018 IEEE

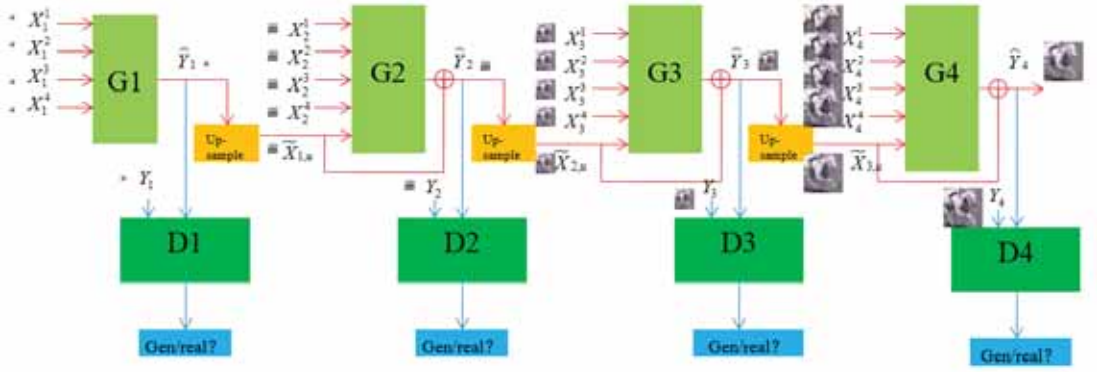


Fig. 1. Illustration of our proposed VC-LAPGAN.

II. Proposed Method

A. Network Structure

In this work, we redesign and retrain a LAPGAN based on the existing multi-scale network in [6] to make it suitable for video coding. Fig. 1 depicts the Laplacian pyramid structure used in our VC-LAPGAN. The multi-scale network is defined as follows. There are in total 4 scales. Let $s_1, \ldots, s_4$ be the sizes of the inputs of our network. Typically, in our experiments, we set $s_1 = 16 \times 16$, $s_2 = 32 \times 32$, $s_3 = 64 \times 64$, $s_4 = 128 \times 128$ during training. As mentioned before, we use four previous frames to extrapolate the current frame. During training, we sample collocated patches from the frames. These patches sampled from previous frames are indexed by $i \in \{1, 2, 3, 4\}$, and are down-sampled three times to achieve the four scales. Let $k \in \{1, 2, 3, 4\}$ denotes the k-th scale. Then let $\mathcal{U}$ be the upscaling operator. Let X_k^i be the i-th previous patch at the k-th scale, and Y_k be the expected output (i.e. the ground-truth of the current patch) at the k-th scale, and G_k be a network that operates at the k-th scale. We recursively define the network G_k, that makes a prediction of Y_k, by

$$\hat{Y}_k = \mathcal{U}(\hat{Y}_{k-1}) + G_k\left(X_k^1, X_k^2, X_k^3, X_k^4, \mathcal{U}(\hat{Y}_{k-1})\right) \quad (1)$$

where $\mathcal{U}(\hat{Y}_0)$ is assumed to be 0 and omitted. Therefore, the network makes a series of predictions, starting from the lowest resolution, and uses the prediction of size s_k as a starting point to make the prediction of size s_{k+1}.

In addition, there are 4 discriminative nets at different scales. Each net D_k takes the generated patch $\hat{Y}_k$ and the real patch Y_k as inputs. D_k is trained to indicate whether the patch is real or generated, which is a binary classification problem.

In our VC-LAPGAN, each of the generative nets $\{G_1, \ldots, G_4\}$ is fully convolutional, i.e. composed of several convolutional layers exclusively. Each convolutional layer is followed by a Rectified Linear Unit (ReLU) [7]. Each of the discriminative nets $\{D_1, \ldots, D_4\}$ is composed of several convolutional layers followed by several fully-connected layers followed by a logistic regression layer. The specification of the nets are summarized in Table I. Note that the VC-LAPGAN is to be integrated into video coding scheme, so we need to control the network complexity. For that purpose, our designed VC-LAPGAN is greatly simplified than the multi-scale network in [6]. However, we empirically find that the simplification has little impact on the compression efficiency.

TABLE I
Specification of Our Proposed VC-LAPGAN

Generative Net	G_1	G_2	G_3	G_4
# filters	16,32,16,1	16,32,16,1	16,32,64, 32,16,1	32,64,128, 64,32,1
filter size	3,3,3,3	5,3,3,5	5,3,3,3,3,5	7,5,5,5,5,7
# parameters	11376	15216	52080	535520
# total parameters	614192			
Discriminative Net	D_1	D_2	D_3	D_4
# filters	32	32,32,64	32,32,64	32,32,64, 128
filter size	3	3,3,3	5,5,5	7,7,5,5
# fully-connected nodes	256,128	256,128	256,128	256,128

B. Derivation of Training Data

When Mathieu *et al.* [6] trained their multi-scale network, they used the Sports 1M dataset [8], which contains low-resolution videos (around 320×240) that are severely compressed. By observing the experimental results, we found the model trained by Sports 1M leads to blurry results and cannot compensate for large motions. Thus, we rebuild a training dataset cropped from uncompressed, high-resolution video sequences downloaded from CDVL [9]. We used 100 videos whose resolutions are either 720p or 1080p. We convert their color space from RGB to YUV since HEVC operates in YUV color space. To match with the high resolution in our training dataset, we set the patch size to 128×128 (as mentioned before) which is much larger than the patch size 32×32 used in [6].

To generate the training samples, we divide all the frames in each of the source videos into quintuple-frame groups, each containing five consecutive frames. We then randomly pick a pixel in each quintuple-frame group and extract a quintuple-patch group centered at that pixel from the video frames. Since some videos consist of many shots, to avoid scene change, we

compute the color histogram between patches to detect shot boundaries and remove the groups across the shot boundaries. To avoid including a large number of samples with no or little motion, we randomly sample 700,000 quintuple-patch groups without replacement according to the flow magnitude. During sampling, a patch group with larger motion is more likely to be chosen than the one with smaller motion. Furthermore, samples with little texture are also not very useful to train our network. We therefore compute the content complexity of patches in each sample and finally select 600,000 quintuple-patch groups with larger content complexity to form the training dataset.

We perform data augmentation on the fly during training, where we randomly flip the samples horizontally as well as vertically and randomly swap their temporal order. This forces the optical flow within the samples to be distributed symmetrically so that the network is not biased towards a certain direction.

C. Training Method

The loss functions of our VC-LAPGAN are the same as those in [6]. Specifically, the loss function for generative nets is:

$$l^G = \sum_{k=1}^{4} \{\lambda_{adv} l^G_{adv}(\hat{Y}_k) + \lambda_{l_1} l^G_1(\hat{Y}_k, Y_k) + \lambda_{gdl} l^G_{gdl}(\hat{Y}_k, Y_k)\} \tag{2}$$

where λ's are weights, l^G_{adv} is the adversarial loss in the general GAN, l^G_1 is the loss–difference between the predicted $\hat{Y}_k$ and the real Y_k, and l^G_{gdl} is also the difference between the prediction and the real, but calculated in the gradient domain. As observed in [10], using the gradient-domain loss is helpful to make the generated images more sharp.

The loss function for discriminative nets is:

$$l^D = \sum_{k=1}^{4} l^D_{adv}(\hat{Y}_k, Y_k) \tag{3}$$

where l^D_{adv} is the adversarial loss in the general GAN. As a common practice, we use stochastic gradient descent with back-propagation to train the generative nets and the discriminative nets alternately.

D. Using VC-LAPGAN in HEVC

Our experimental results show that the VC-LAPGAN is capable in generating visually plausible frames. However, the generated frames may not be exactly the same to the frame to predict. For example, the motion in the previous frames is at a constant speed, but suddenly increases or decreases in the current frame; in such cases, the extrapolated frame is different from the ground-truth. For the purpose of maximizing the utility of the extrapolated frame, we put it into the reference frame list for the current frame, so as to enable normal block-based motion estimation and compensation on the extrapolated frame. It is worth noting that the extrapolated frame is regarded as a long-term reference frame, so as to avoid the ambiguity of its picture-order-count (POC). In addition, although our network is trained with fixed-size patches, we can directly use the generative nets for frame extrapolation on videos with any size, because the generative nets are fully convolutional.

TABLE II
BD-RATE RESULTS OF OUR METHOD THAN DIFFERENT ANCHORS

	HEVC			HEVC with adjusted references		
	Y	U	V	Y	U	V
Class B	−2.7%	−2.2%	0.6%	−2.6%	−2.2%	0.7%
Class C	−1.3%	−0.8%	0.1%	−1.2%	−1.1%	0.1%
Class D	−1.4%	−0.6%	0.3%	−1.4%	−1.3%	0.9%
Class E	−3.4%	−3.3%	−1.7%	−3.1%	−3.0%	−1.0%
Overall	**−2.2%**	**−1.7%**	**−0.0%**	**−2.0%**	**−1.8%**	**0.3%**

III. EXPERIMENTS

A. Training

We use the deep learning software TensorFlow [11] to train the VC-LAPGAN on an NVIDIA Tesla K40C graphical processing unit. Several implementation details are as follows. When preparing the training data at different scales, we use bicubic down-sampling rather than Gaussian down-sampling, since we empirically found that bicubic down-sampling can make the trained model to produce results that appear more sharp. For the upscaling operator, we use bilinear interpolation since it is easily differentiable. As for the weights in (2), we found $\lambda_{adv} = 1$, $\lambda_{l_1} = 1$ and $\lambda_{gdl} = 1$ work well and used it for all the experiments.

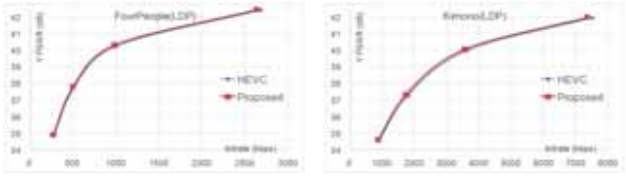
B. Comparison with HEVC Anchor

We integrate the trained network into HEVC reference software HM version 12.0, and test on the HEVC common test sequences. Since we consider frame extrapolation, we only test the low-delay P configuration. According to the HEVC common test conditions, we use four classes, 16 sequences to test. Class F is not used as it is screen content and has special motion patterns. Please note that the test sequences are not used for training, so that the results can demonstrate the generalization ability of our network. To evaluate the coding efficiency, we use the BD-rate [12] to measure on luma and chroma channels independently.

As mentioned before, we use the previous four frames to extrapolate, and add the generated frame into the reference frame list for the current frame. We compare our method to the HEVC anchor. The results are summarized in Table II. As can be observed, the proposed method achieves on average -2.2%, -1.7%, -0.0% BD-rate for Y, U, V components, respectively.

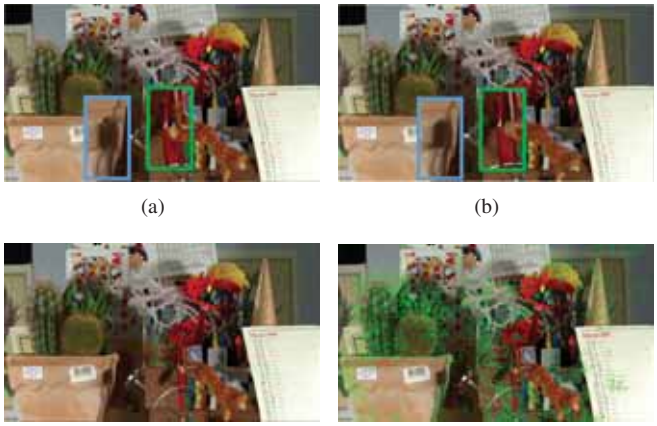
Some typical R-D curves are shown in Fig. 2. We can observe that the PSNR improvements are almost the same for the entire bit-rate range, showing that the method is not sensitive to the quantization error in video coding.

One issue remains in the comparison. In the HM software, the default reference frames for the current frame are *not* the immediately previous four. Therefore, our method, using the previous four frames to extrapolate, is possibly introducing more information for motion estimation and compensation. To make the comparison more fair, we revise the default setting in the HM software, i.e. using the immediately previous four



(a) (b)

Fig. 2. R-D curves of the sequences `FourPeople` (a) and `Kimono` (b).



(a) (b)

(c) (d)

Fig. 3. The 17-th frame of `Cactus`, compressed with QP = 22. (a) The predicted frame by using the HEVC anchor; (b) The extrapolated frame by VC-LAPGAN. Note the zoomed-in portions shown in (a) and (b) indicated by green and blue blocks. (c) The original frame; (d) The blocks that chose the extrapolated frame (b) as reference are shown in green.

frames as reference frames. This anchor is named "HEVC with adjusted references." Compared to this anchor, our method does not introduce more information. The comparison results are also summarized in Table II. We can observe the proposed method achieves on average -2.0%, -1.8%, 0.3% BD-rate for Y, U, V components, respectively. The BD-rate reduction of the two tests is almost the same, which partially verifies the coding gain is not coming from using more information. Thus, we believe the trained VC-LAPGAN model is indeed capable in frame extrapolation.

The generated frame usually has a high visual quality, as shown in Fig. 3. It can be observed that the predicted frame by using the HEVC anchor contains obvious blocking artifacts, while the extrapolated frame is free of such artifacts. We also analyze the blocks that chose the extrapolated frame as reference, as shown in Fig. 3 (d). Obviously, the blocks correspond to regions with complex motions, which demonstrates the benefit of our method.

C. Comparison of Frame Extrapolation with [6]

Fig. 4 compares the extrapolated frames by our trained VC-LAPGAN and the network in [6]. Our result is visually better. First, our result is more sharp, which is attributed to the new training data that we built from high-resolution uncompressed videos. Second, this is attributed to the color space conversion. Because the network in [6] was trained on RGB videos,

(a) (b)

Fig. 4. The extrapolated frames by the network in [6] (a) and our VC-LAPGAN (b).

when using the network in video coding, which operates in YUV color space, it is necessary to perform RGB and YUV conversions. On the contrary, our network is trained directly on YUV videos, avoiding the color space conversion. It can be observed in Fig. 4 that color distortion is noticeable in the result of using the network in [6]. In addition, note that our VC-LAPGAN has much fewer parameters than the network in [6], and thus incurs less computational complexity.

IV. Conclusion

This paper presents a frame extrapolation method for video coding with a Laplacian pyramid of generative adversarial networks (VC-LAPGAN). We reuse the basic structure proposed in [6], but we simplify the network to trade-off the compression efficiency and computational complexity. We rebuild a training dataset from high-resolution, uncompressed video sequences, using some data pre-processing and data augmentation techniques. Thus, our VC-LAPGAN is more suitable for video coding. Experimental results show that the proposed method achieves on average 2.0% BD-rate reduction, compared to HEVC.

References

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circ. Syst. Video Techn.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[2] V. E. Seferidis and M. Ghanbari, "General approach to block-matching motion estimation," *Optical Engineering*, vol. 32, no. 7, pp. 1464–1474, 1993.

[3] L. Li, H. Li, D. Liu, Z. Li, H. Yang, S. Lin, H. Chen, and F. Wu, "An efficient four-parameter affine motion model for video coding," *IEEE Trans. Circ. Syst. Video Techn.*, vol. 28, no. 8, pp. 1934–1948, 2018.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680.

[5] E. L. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *NIPS*, 2015, pp. 1486–1494.

[6] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," arXiv:1511.05440, 2015.

[7] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *ICML*, 2010, pp. 807–814.

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014, pp. 1725–1732.

[9] M. H. Pinson, "The consumer digital video library [best of the web]," *IEEE Signal Processing Magazine*, vol. 30, no. 4, pp. 172–174, 2013.

[10] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *ECCV*, 2012, pp. 611–625.

[11] M. Abadi, A. Agarwal, P. Barham *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," arXiv:1603.04467, 2016.

[12] G. Bjontegaard, "Calcuation of average PSNR differences between RD-curves," VCEG-M33, 2001.