# Deep Motion Vector Prediction for Versatile Video Coding

Lanlan Li, Kexin Wu, Hongan Wei*, Jiaqi Liu, Ying Fang, Haifeng Zheng
*Fujian Key Lab for Intelligent Processing and Wireless Transmission of Media Information*
Fuzhou University
Fuzhou, China
lilanlan@fzu.edu.cn, n191120032@fzu.edu.cn, *weihongan@fzu.edu.cn, 201127045@fzu.edu.cn, fangying@fzu.edu.cn, zhenghf@fzu.edu.cn

2021 7th International Conference on Computer and Communications (ICCC) | 978-1-6654-0950-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCC54389.2021.9674239

***Abstract*—Versatile video coding (VVC) has made significant progress in the compression efficiency of video coding. It achieves better performance and halves bitrate compared to the high efficiency video coding (HEVC) under the same visual quality. However, the complexity of VVC has also been significantly increased, especially in inter prediction. This paper proposes an improved motion vector prediction method based on neural networks for motion estimation (ME). Firstly, dynamic weights are proposed in the process of selecting the best MVP for advanced motion vector prediction (AMVP); secondly, we build the motion vector prediction model based on the deep neural network; finally, the model is embedded in VVC to acquire a more accurate MVP and reduce the encoding complexity of ME. Experimental results show that the proposed algorithm can reduce the encoding time of motion estimation under the premise of guaranteeing video quality.**

***Keywords—motion vector prediction, motion estimation (ME), deep neural networks, encoding complexity***

## I. Introduction

With the rapid development of high-definition and ultra-high-definition video, video compression technology has faced new challenges. Versatile video coding (VVC) [1], as a new standard for video coding, inherits the basic framework of the high efficiency video coding (HEVC) [2], and introduces several improvements and new coding tools to gain more compression efficiency. Inter prediction uses motion estimation (ME) and motion compensation (MC) technology to remove the temporal redundancies to achieve compression. Motion estimation technology is to find the best matching block in the previously encoded image for each pixel block of the current image, and the retrieved block is recorded by the motion vector (MV). Motion estimation involves many calculations, including motion vector prediction, motion search, rate-distortion (RD) cost function, motion vector difference (MVD), and so on. The flowchart of motion estimation is shown in Fig. 1. ME accounts for the largest proportion of the total encoding time, which is a module with the highest complexity.

So far, there have been many algorithms about VVC complexity [3-5]. Moreover, a growing number of optimization algorithms of ME have been proposed and achieved apparent results. Park *et al.* [6] improved affine motion estimation (AME) technology newly added in VVC and introduced useful features reflecting multi-type tree (MTT) and AME statistical characteristics. An algorithm for skipping redundant AME procedures using these features is proposed. Li *et al.* [7] proposed a k-nearest neighbor (KNN) algorithm that can be divided into learning stage and prediction stage to estimate the best search range of motion search, to reduce search points and complexity. In [8], an adaptive search range selection algorithm based on depth level was presented according to the statistical results of the prediction distribution of motion vector difference. The algorithm in [9] includes searching multiple motion vector predictors (MVPs) within a more accurate range to obtain additional accurate MVPs in bottom-up order. These bottom-up MVPs are obtained from prediction units (PUs) in lower-level coding units (CUs) and are applied to integer motion estimation for HEVC. In [10], an alternative motion vector referencing scheme is proposed to fully accommodate the dynamic nature of the motion field and achieves better compression performance.

Deep learning has developed quickly in recent years, the deep neural network has been widely used in various realistic scenarios and achieved remarkable results. The combination of the deep neural network and video coding has been very extensive [11-13]. In terms of inter prediction, Huo *et al.* [14] proposed the convolutional neural network-based motion compensation refinement by combining the temporal motion compensation and spatial correlation of video coding. For the motion estimation module, in [15], the CNN module based on image super-resolution can be used to generate higher quality sub-pixels to improve the performance of motion estimation.

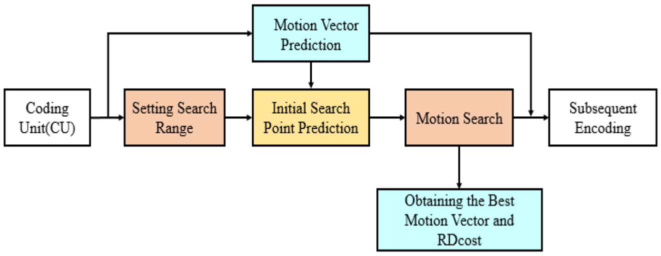


Fig. 1. The flowchart of motion estimation.

In this previous work, the weight issue about advanced motion vector prediction (AMVP) selects the optimal MVP was not considered. Deep learning was hardly introduced into motion vector prediction. So we consider combining the learning ability of the deep neural network and dynamic weights to improve motion vector prediction and reduce motion search range. We sum up the major contributions of this work as follows:

1) Dynamic weights are proposed in the process of selecting the best MVP for AMVP.
2) A motion vector prediction model is proposed based on deep neural networks that can make the MVP more accurate, and reduce the motion search range.
3) The prediction model is embedded in the VVC test model (VTM) for encoding optimization. Experimental results demonstrate that the model can reduce the encoding complexity of motion estimation while ensuring video quality.

The rest of this paper is organized as follows. Section II comprehensively analyzes and summarizes the proposed algorithm that establishing the MV prediction model based on the historical coding data and deep neural networks. Section III shows the experimental results of the proposed algorithm compared with the VTM. Finally, Section IV summarizes the whole text.

## II. MV Prediction Based on Deep Neural Networks

Motion vector prediction is an important part of ME, and VVC follows two techniques of HEVC in motion vector prediction, Merge and AMVP. The process of AMVP is as follows. Firstly, establishing the MVP candidate list of the current coding unit (CU) based on the MV of spatial and temporal neighboring CUs; then the best MVP is selected from the candidate list according to the RD cost; finally the best MVP is used to point to the starting point of the motion search. When the best MVP is close to the actual MV, the MVD between MV and MVP will become smaller, which can reduce the motion search range of ME. Therefore, we can optimize the ME by improving the motion vector prediction technology.

### *A. The Weight Selection Analysis*

In AMVP, the best MVP of current CU and two MV candidates in the final candidate list, which are set as MV$_1$ and MV$_2$ have a relationship. The relationship can be expressed as:

$$MVP = \alpha MV_1 + \left(1-\alpha\right) MV_2 , \alpha \in \left\{0{,}1\right\} , \quad (1)$$

where the weight $\alpha$ is fixed and equal to 0 or 1. After selecting the candidate with a low rate-distortion cost, the MVP value is equal to the value of MV$_1$ or MV$_2$ exactly.

To make the MVP more accurate, the algorithm in this paper sets a dynamic weight $\omega$ when AMVP selects the best MVP in the final candidate list. We set the relationship among the original MV, MV$_1$, and MV$_2$ as shown in:

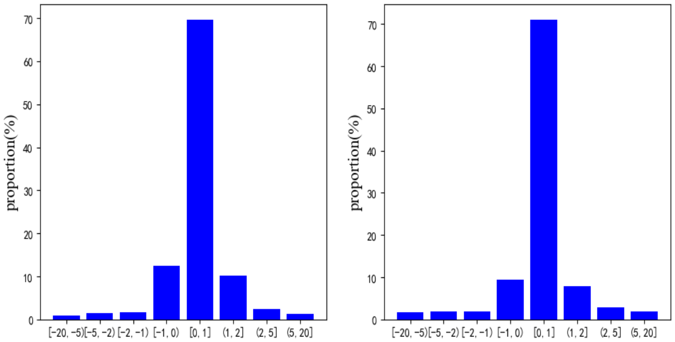
$$MV \Leftrightarrow \omega MV_1 + \left(1-\omega\right) MV_2 , \quad (2)$$

instead of being equal to 0 or 1, the value of $\omega$ is determined by both the original MV and the specific values of MV$_1$ and MV$_2$. MV has two components in the horizontal direction and the vertical direction. So (2) can be written in the following form:

$$\begin{cases} MV_x = \omega_x MV_{1x} + \left(1-\omega_x\right) MV_{2x} \\ MV_y = \omega_y MV_{1y} + \left(1-\omega_y\right) MV_{2y} \end{cases} , \quad (3)$$

where $x$, $y$ denote the x and y components of the motion vector, respectively. And $\omega_x$ denotes the weight in the horizontal direction of MV, $\omega_y$ is the weight in the vertical direction of MV.

To analyze the $\omega$ distribution of CU, we have selected one video from each class in the VVC standard test sequence for encoding and saved MV, MV$_1$, and MV$_2$ to calculate $\omega$. The horizontal $\omega_x$ and vertical $\omega_y$ of MV corresponding to each CU are calculated by (3). The ranges of all $\omega_x$ and $\omega_y$ are statistically analyzed and the statistical results are shown in Fig. 2. The x coordinate represents the range of $\omega_x$ or $\omega_y$ and the y coordinate represents the percentage of CU in a given range as a percentage of the total.



(a)

(b)

Fig. 2. Range statistics for $\omega$. (a) $\omega_x$; (b) $\omega_y$.

As can be seen from Fig. 2(a) and Fig. 2(b), the $\omega_x$ and $\omega_y$ have the same distribution that is mainly between [-1, 1].

### *B. Construction of MV Prediction Model*

1) *Feature Extraction:* We need to analyze image features that are related to $\omega$. The position, size, division depth, and texture features of the current CU to be encoded are considered. The texture features include the maximum, minimum, average, and variance of the luminance gradient for CU.

Since a moving object in the natural image may have continuity in the spatial and the temporal, covering multiple coding blocks, which may have similar motion information. The motion information of the current CU to be encoded will also be affected by the MV of the reference CU in the spatial and the temporal. In this paper, two reference MV candidates of the final list of AMVP are also selected as input features.

2) *Dataset:* A dataset is built for the MV prediction model in this paper. Video sequences are selected according to the VVC standard common test sequence. Details of video sequences used for creating the dataset are shown in Table I.

TABLE I. DETAILS OF VIDEO SEQUENCES USED FOR CREATING A DATASET

| Video Sequences | Resolutions | Frame Rates (fps) |
|---|---|---|
| Tango2 | 3840×2160 | 60 |
| CatRobot | 3840×2160 | 60 |
| BasketballDrive | 1920×1080 | 50 |
| RaceHorses | 832×480 | 30 |
| BlowingBubbles | 416×240 | 50 |

P frames after all the keyframes in the video are extracted used for subsequent encoding. Keyframes are judged by the method of inter-frame difference. Firstly, the video is read and the inter-frame difference between every two frames is calculated successively; then the average difference intensity is obtained; finally, the frame with the local maximum value of the average difference intensity is selected as the keyframe of the video. Frames extracted from the video were encoded in VTM, and features of each CU were extracted and saved in the encoding process.

After analysis of the extracted feature data, the data of corner blocks were removed. And the singular values that differ greatly from most of the data also have been removed. The dataset consists of 400000 pairs of data, of which 80% are used as the training set and 20% are used as the verification set.

3) *MV prediction model:* The network structure for the $\omega$ prediction proposed is shown in Fig. 3. Because fully connected neural networks have a strong nonlinear fitting ability, and this algorithm needs to merge all the features of the input, this paper adopts the full connection layer neural network for prediction. Except for the input layer and the output layer, the Relu activation function is used after each hidden layer. Two networks have been trained for the prediction of the horizontal and vertical components of MV, respectively.

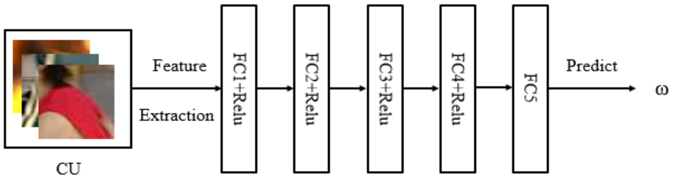


Fig. 3. The network structure of $\omega$ prediction.

The network adopts the Adam optimization algorithm as the optimization algorithm, and the mean-square error (MSE) is selected as the loss function, the calculation of the loss function is shown in (4).

$$\mathrm{Loss}\left(\mathrm{y},\mathrm{y}'\right)=\frac{1}{n}\sum_{i=1}^{n}(y_i-y_i')^2 . \qquad (4)$$

The network inputs are the position, size, and division depth of CU, MV$_1$, MV$_2$, maximum, minimum, average, and variance of the luminance gradient for CU. The sample label is the $\omega$ calculated by (3), the horizontal component label is $\omega_x$ and the vertical component label is $\omega_y$. The network output is the predicted $\omega$. The predicted MV is calculated by the predicted $\omega$, and it replaces the best MVP selected by the original VVC encoder as the starting point of motion search.

### C. Motion Estimation with MV Prediction Model

1) *The best $\omega$ range:* The proposed model needs to combine with the original encoder. We should set a $\omega$ range threshold that determines whether to use the network for the current CU to be encoded. Several $\omega$ ranges are set for testing, and the results are shown in Fig. 4.

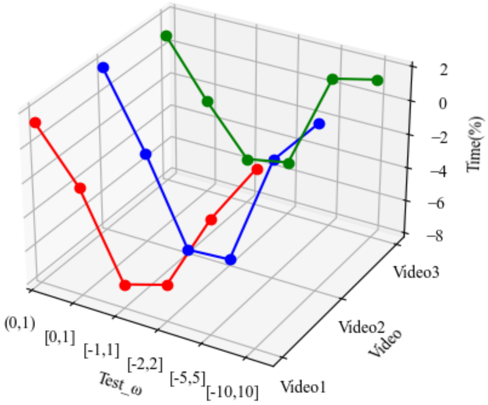


Fig. 4. Results of the best $\omega$ range.

The z-axis represents the percentage change of motion search time compared to the original algorithm. According to Fig. 4, the best $\omega$ range is [-1, 1], so the $\omega$ range that will be chosen in the actual encoding process is [-1, 1]. If the $\omega$ is within this range, the network will be used; if not, we will use the original algorithm in VVC.

2) *Application of the model:* The purpose of the algorithm in this paper is to make MVP as close to MV as possible and reduce the time of motion search. Using the model to predict and calculate MV also can save the time of selecting the best MVP for AMVP. Therefore, the MV prediction network is embedded into the motion vector prediction module, and the $\omega$ of coCU is used as the $\omega$ of the current CU. coCU refers to the CU at the same position in the previous P frame, which has a strong temporal correlation with the current CU.

The overall framework of the algorithm is shown in Fig. 5. The first part is to establish a dataset according to the historical encoding data. In the second part, the MV prediction model is obtained by building the fully connected deep neural network and using the network to train the dataset. In the third part, the model built is embedded into the VVC encoder to predict MV, and then the network predicts $\omega$ according to the input features. Based on the predicted $\omega$ of the network, the predicted MV is calculated and the block pointed by the predicted MV is used as the starting point of motion search.

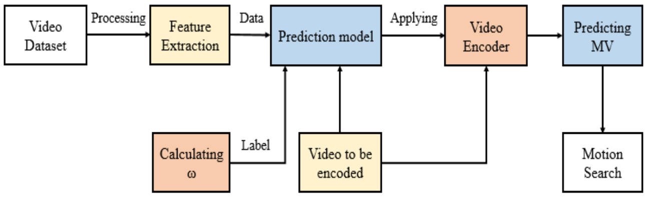


Fig. 5. The framework of the proposed algorithm.

## III. Experimental Results

In order to verify the optimization effect of the proposed MV prediction model for VVC. The model is embedded into VTM 6.0, and the low delay P (LDP) configuration is used for testing. The resolutions of the video sequences include 3840 × 2160, 1920 × 1080, 832 × 480, and 416 × 240. The quantitative parameters (QPs) used for encoding are 22, 27, 32, and 37.

Table II describes the comparison of MV prediction time in AMVP, motion search time, BDBR, and BDPSNR between the VVC coding framework embedded in the proposed model and the VTM 6.0. △Time1 in Table II is the decreased time of AMVP compared to VTM 6.0 and its calculation is shown in:

$$\triangle Time1=\frac{Time1_{ref}-Time1_{pro}}{Time1_{ref}}\times 100\%, \qquad (5)$$

where $\text{Time1}_{\text{ref}}$ is the encoding time of AMVP in VTM 6.0, $\text{Time1}_{\text{pro}}$ is the encoding time of AMVP with the proposed model.

△Time2 is the reduced time of motion search compared to VTM 6.0 and its calculation is shown in:

$$\triangle Time2=\frac{Time2_{ref}-Time2_{pro}}{Time2_{ref}}\times 100\%, \qquad (6)$$

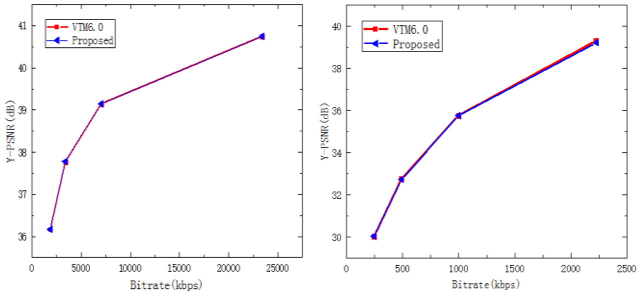
where $\text{Time2}_{\text{ref}}$ is the encoding time of motion search in VTM6.0 and $\text{Time2}_{\text{pro}}$ is the encoding time of motion search with the proposed model.

TABLE II. Experimental Results Compared with VTM6.0

| **Video Sequences** | | **△Time1 (%)** | **△Time2 (%)** | **BD BR (%)** | **BD PSNR (dB)** |
|---|---|---|---|---|---|
| Class A | CatRobot | 64.28 | 5.87 | 0.07 | -0.005 |
| | DaylightRoad2 | 59.36 | 4.95 | -0.04 | 0.003 |
| | Tango2 | 48.50 | 6.15 | 0.13 | -0.009 |
| Class B | BasketballDrive | 54.65 | 7.87 | 0.05 | -0.003 |
| | Cactus | 71.29 | 3.74 | 0.02 | -0.004 |
| | MarketPlace | 55.95 | 4.54 | -0.07 | 0.001 |
| | BQTerrace | 87.02 | 6.45 | 0.03 | -0.006 |
| | RitualDance | 47.05 | 5.03 | -0.10 | 0.005 |
| Class C | BasketballDrill | 69.40 | 5.02 | -0.21 | 0.009 |
| | RaceHorses | 65.13 | 6.38 | 0.18 | -0.010 |
| | BQMall | 73.26 | 5.67 | -0.06 | 0.002 |
| | PartyScene | 73.05 | 6.11 | 0.06 | -0.004 |
| Class D | BasketballPass | 71.42 | 6.62 | 0.14 | -0.011 |
| | BQSquare | 86.91 | 3.52 | -0.15 | 0.007 |
| | BlowingBubbles | 74.63 | 4.61 | 0.12 | -0.005 |
| **Average** | | **66.79** | **5.50** | **0.01** | **-0.002** |

As can be seen from Table II, compared with the VTM 6.0, the proposed model has saved the time of 66.79% for AMVP, 5.5% time of motion search averagely. The BDBR has increased of 0.01% and BDPSNR has decreased of 0.002 dB on average. The results prove that the proposed algorithm of this paper has reduced encoding time with a slight decrease in encoding performance.

To further examine the performance of this proposed algorithm, Y-PSNR and bitrate data of two sequences are selected in this paper, and RD curves corresponding to different QPs of the sequence are drawn as shown in Fig. 6. The blue line represents the RD curve of the proposed algorithm, and the red line represents the RD curve of the original algorithm. As can be seen from Fig. 6, the red line and the blue line are almost identical, indicating that there is almost no difference in encoding performance between the proposed algorithm and the original VVC encoder.



(a) BasketballDrive (b) BlowingBubbles

Fig. 6. RD Curves of VTM 6.0 and our proposed method.

PSNR is the most commonly used objective evaluation index for images, however, it does not consider the visual characteristics of the human eyes. Structural similarity (SSIM) is a method based on structural information, which is used to measure the similarity between the original signal and the processed signal. Therefore, in order to more comprehensively evaluate the coding performance of the proposed model, we provide SSIM comparison between the original encoder and the proposed model. The results of the experiments are presented in Table III.

It can be observed from experimental results that SSIM decreased by 0.000060 on average, which is insignificant. Therefore, it is proved that the proposed method saves encoding time while maintaining almost the same image quality as the original algorithm.

TABLE III. SSIM COMPARISON BETWEEN THE PROPOSED ALGORITHM AND THE ORIGINAL ALGORITHM

| Video Sequences | | VTM6.0 | Proposed | △SSIM |
|---|---|---|---|---|
| | | *SSIM* | *SSIM* | |
| Class A | CatRobot | 0.986372 | 0.986296 | -0.000076 |
| | DaylightRoad2 | 0.991271 | 0.991312 | 0.000041 |
| | Tango2 | 0.987532 | 0.987518 | -0.000014 |
| Class B | BasketballDrive | 0.993241 | 0.993191 | -0.000050 |
| | Cactus | 0.987607 | 0.987544 | -0.000063 |
| | MarketPlace | 0.984639 | 0.984625 | -0.000014 |
| | BQTerrace | 0.998081 | 0.997988 | -0.000093 |
| | RitualDance | 0.993189 | 0.993274 | 0.000085 |
| Class C | BasketballDrill | 0.979017 | 0.979137 | 0.000120 |
| | RaceHorses | 0.991300 | 0.991010 | -0.000290 |
| | BQMall | 0.996197 | 0.996219 | 0.000022 |
| | PartyScene | 0.988269 | 0.988191 | -0.000078 |
| Class D | BasketballPass | 0.984379 | 0.983970 | -0.000409 |
| | BQSquare | 0.984811 | 0.984893 | 0.000082 |
| | BlowingBubbles | 0.990538 | 0.990370 | -0.000168 |
| **Average** | | **0.989096** | **0.989036** | **-0.000060** |

## IV. CONCLUSIONS

This paper presents a new algorithm that enhanced motion vector prediction to reduce the ME encoding complexity in VVC. This new algorithm proposes dynamic weights in the process of selecting the best MVP for AMVP, and builds the MV prediction model based on the deep neural network. The model is introduced into the VVC encoder for optimization. Experimental results show that the proposed algorithm can reduce 66.79% AMVP time and 5.5% motion search time on average compared to VTM 6.0. The results demonstrate that the proposed algorithm reduces the encoding complexity with a negligible impact on encoding performance and video quality.

## ACKNOWLEDGMENT

This work is supported by the Natural Science Foundation of Fujian Province (No. 2019J01222).

## REFERENCES

[1] B. Bross, J. Chen and S. Liu, "Versatile Video Coding (Draft 6)" Jt. Video Expert. Team ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Jul. 2019.

[2] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp.1649–1668, Dec. 2012.

[3] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard and C. Bergeron, "Tunable VVC Frame Partitioning based on Lightweight Machine Learning," IEEE Transactions on Image Processing, vol. 29, pp. 1313–1328, Sept. 2019.

[4] S. H. Tsang, N. W. Kwong, and Y. L. Chan, "FastSCCNet: Fast Mode Decision in VVC Screen Content Coding via Fully Convolutional Network," Visual Communications and Image Processing (VCIP), Dec. 2020, pp. 177–180.

[5] Q. Zhang, Y. Wang, L. Huang, and B. Jiang, "Fast CU Partition and Intra Mode Decision Method for H.266/VVC," IEEE Access, vol. 8, pp. 117539–117550, Jun. 2020.

[6] S. H. Park, J. W. Kang, "Fast Affine Motion Estimation for Versatile Video Coding (VVC) Encoding," IEEE Access, vol. 7, pp. 158075–158084, Oct. 2019.

[7] Y. Li, Y. Liu, H. Yang, and D. Yang, "An adaptive search range method for HEVC with the k-nearest neighbor algorithm," Visual Communications and Image Processing (VCIP), Apr. 2016, pp. 1–4.

[8] H. Kibeya, F. Belghith, M. Ayed, and N. Masmoudi, "Adaptive motion estimation search window size for HEVC standard," 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Jun. 2017, pp. 410–415.

[9] T. S. Kim, C. E. Rhee, H. J. Lee, and S. I. Chae, "Fast Integer Motion Estimation With Bottom-Up Motion Vector Prediction for an HEVC Encoder," IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 12, pp. 3398–3411, Dec. 2018.

[10] J. Han, Y. Xu, and J. Bankoski, "A dynamic motion vector referencing scheme for video coding," 2016 IEEE International Conference on Image Processing (ICIP), Aug. 2016, pp. 2032–2036.

[11] Y, Li, D. Liu, H. Li, L. Li, and F. Wu, et al. "Convolutional neural network-based block up-sampling for intra frame coding, " IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 9, pp. 2316–2330, Sept. 2018.

[12] Z. Zhao, S. Wang, S. Wang, X. Zhang, and S. Ma, et al. "Enhanced Bi-prediction with Convolutional Neural Network for High- Efficiency Video Coding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 29, no. 11, pp. 3291–3301, Nov. 2019.

[13] C. Jia, S. Wang, X. Zhang, S. Wang, and J. Liu, et al. "Content-Aware Convolutional Neural Network for In-Loop Filtering in High Efficiency Video Coding," IEEE Transactions on Image Processing, vol. 28, no. 6, pp. 3343–3356, Jul. 2019.

[14] S. Huo, D. Liu, F. Wu, and H. Li, "Convolutional Neural Network-Based Motion Compensation Refinement for Video Coding," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), May 2018, pp. 1–4.

[15] N. Yan, D. Liu, H. Li, B. Li, and L. Li, et al. "Convolutional Neural Network-Based Fractional-Pixel Motion Compensation," IEEE Transactions on Circuits and Systems for Video Technology, vol. 29, no. 3, pp. 840–853, Mar. 2019.