

# Fast Depth Intra Coding Based on Depth Edge Classification Network in 3D-HEVC

Chang Liu<sup>1b</sup>, Kebin Jia<sup>1b</sup>, *Member, IEEE*, and Pengyu Liu<sup>1b</sup>, *Member, IEEE*

**Abstract**—As the extension of high efficiency video coding (HEVC) standard, three dimensional-HEVC (3D-HEVC) is the latest 3D video coding standard. 3D-HEVC adopts many complicated coding algorithms to generate additional intermediate views for 3D video representation, which result in extremely high coding complexity. Therefore, this paper proposed a fast depth intra coding approach to reduce the 3D-HEVC complexity, which is based on convolutional neural network (CNN). First, we established a database based on the independent view of the depth map, which includes coding unit (CU) partition data of the depth map. Second, we constructed a depth edge classification CNN (DEC-CNN) framework to classify the edges for the depth map and embedded the network into a 3D-HEVC test platform. Finally, we utilized the pixel value of the binarized depth image to correct the above classification results. The experimental results demonstrated that our approach can reduce the intra coding time by 72.5% on average under negligible degradation of coding performance. This result outperforms the other state-of-the-art methods to reduce the coding complexity of 3D-HEVC.

**Index Terms**—3D-HEVC, CNN, depth map, edge classification.

## I. INTRODUCTION

WITH the rapid development of multimedia information technology, video television (TV) is constantly being updated. On the one hand, video TV can support higher resolutions, from standard definition (SD) to high definition (HD), full HD (FHD), and even ultra HD (UHD) [1]. On the other hand, video TV can support more views, from two dimensional (2D) TV to three dimensional (3D) TV [2], and even the free viewpoint television (FTV). Artificial intelligence (AI) and the fifth generation mobile networks (5G) has not only enhanced data computing and analysis, but also provided data interconnection for new products and services. Under this background, the traditional 2D video cannot bring the enjoyment of vision, hearing, and touch. Therefore, immersive video [3], which can reflect the 3D information of real scenes,

provides wide viewing freedom, and gives an immersive feeling, came into being.

In immersive video, the multi-view video (MVV) [4], which can provide an immersive visual experience for users, has gradually become a research hotspot in the multimedia information industry. More views create greater immersion, but pose a great challenge to data storage and transmission. The multi-view video plus depth (MVD) [5] video format only consists of a limited number of texture videos and their corresponding depth maps. These texture videos and depth maps can be used to synthesize multiple virtual views by using depth-image-based rendering (DIBR) [6]. Therefore, MVD video format can greatly reduce the MVV data. Based on this, MVD video format is considered to be the most effective 3D video format. To efficiently compress the MVD data, the Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) [7], which is formed by Video Coding Expert Group (VCEG) from the International Standard Organization (ISO) and Motion Picture Expert Group (MPEG) from the International Telecommunication Union (ITU), launched a new standard called 3D extension of high efficiency video coding (3D-HEVC) [8].

As the extension of high efficiency video coding (HEVC) [9] standard, 3D-HEVC introduces depth map coding [10]. Theoretically, the HEVC video coding framework can also be used to encode the depth map but the depth map represents the distance between the object and the camera, and the video coding framework based on HEVC is mainly suitable for the original natural texture. Therefore, the HEVC video coding framework cannot be perfectly applied to the depth map. To this end, 3D-HEVC employs many complicated coding algorithms for depth map coding, including depth modeling mode (DMM) [11], segment-wise DC coding (SDC) [12], motion parameter inheritance (MPI) [13], and view synthesis optimization (VSO) [14]. Although these coding algorithms improve the compression efficiency, they also introduce increasing greater computational complexity.

To reduce the complexity of depth map coding, many fast algorithms have been proposed [15]–[21]. Compared with texture video, a depth map contains large smooth areas, and it has distinct edge. In terms of spectrums, it is different from the uniform spectrum distribution of texture video. The spectrum of a depth map is mainly distributed in the low and high frequencies. Among many new depth map coding algorithms, DMM, as a new intra prediction mode, can better encode the edge of a depth map. However, the introduction of DMM mode also brings huge computational

Manuscript received April 21, 2021; revised August 9, 2021; accepted August 12, 2021. Date of publication August 27, 2021; date of current version March 4, 2022. This work was supported in part by the Beijing Natural Science Foundation under Grant 4212001, and in part by the Basic Research Program of Qinghai Province under Grant 2020-ZJ-709 and Grant 2021-ZJ-704. (Corresponding author: Kebin Jia.)

The authors are with the Beijing Key Laboratory of Computational Intelligence and Intelligent System, Faculty of Information Technology, and the Beijing Laboratory of Advanced Information Networks, Beijing University of Technology, Beijing 100124, China (e-mail: changliu55@126.com; kebinj@bjut.edu.cn; liupengyu@bjut.edu.cn).

Digital Object Identifier 10.1109/TBC.2021.3106143

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.

For more information, see <https://creativecommons.org/licenses/by-nc-nd/4.0/>

complexity. Therefore, in order to reduce the complexity of depth map coding, some scholars attempt to simplify the process of intra prediction mode decision in a depth map. The early works for selecting the intra prediction mode are traditional methods such as edge detection, maximum between-class variance (Otsu), and gray histogram [15]–[17]. In addition, some scholars have found that the quad-tree coding unit (CU) partition in HEVC accounts for the largest proportion of coding time. This result can also be applied to the CU partition in a depth map. Therefore, some scholars attempt to simplify the process of CU partition in depth maps. The early works mainly use some heuristic methods such as skipping or terminating the CU partition process in advance based on the features extracted manually [18]–[21]. With the rapid development of deep learning in the past few years, it has gradually moved to the traditional field of video coding. The cross field of video coding and deep learning has gradually become a hot research direction. Video coding researchers are using deep learning to solve problems, which are difficult to overcome with traditional algorithms [22]–[33]. However, some existing deep learning methods do not use the edge features of the depth map, which leads to the quality degradation of a depth map.

In this paper, we constructed a depth edge classification (DEC) network based on convolutional neural network (CNN), named DEC-CNN, to reduce the complexity of depth map intra coding. In addition, we used traditional methods to correct the classification results. The combination of deep learning and traditional methods can reduce the coding complexity of a depth map while ensuring the quality of a depth map and synthetic views. Specifically, the main contributions of this paper are summarized as follows.

(1) We established a database containing the intra CU of depth map. Since 3D-HEVC adopts MVD video format, there is a slight difference between different views (one independent view and two non-independent views). To ensure the validity of the data, we established a database based on the depth map from the independent view.

(2) We constructed a depth edge classification network named DEC-CNN for predicting intra CU partition of a depth map. Different from texture video, a depth map has obvious edge features. In order to realize edge complexity classification of coding units (CUs), we divided the CUs into two categories.

(3) We used the pixel value of the binarized depth image to correct the classification results. Different edge features have different pixel values. We set the threshold of edge complexity by counting the distribution of pixel values. Based on this, the edge complexity of CUs can be classified, and the classification results can be used as the secondary verification of the results obtained from the DEC-CNN network.

The remainder of this paper is organized as follows. Section II reviews the related works on 3D-HEVC complexity reduction. Section III presents the motivation and 3D-HEVC coding structure for the proposed approach. The details of the proposed coding approach are described in Section IV. Section V reports the experimental results. Finally, Section VI concludes this paper.

## II. RELATED WORK

In this section, two main categories of the current 3D-HEVC complexity reduction works are reviewed. They are fast intra prediction mode decision and fast intra CU size decision.

The first category focuses on fast intra prediction mode decision. In traditional methods, the complex intra prediction mode traversal can be simplified according to edge detection of the depth map. The representative methods include [15]–[17]. Specifically, in [15], a fast intra mode decision method to reduce the computational complexity of 3D-HEVC was proposed based on the characteristics of the depth map. The proposed algorithm uses the isotropic Sobel operator to detect the texture complexity and edge direction of the PU, thereby reducing the number of modes that need to be calculated during the intra coding process. Li *et al.* [16] proposed a complexity reduction scheme based on spatial correlation and the rate-distortion (RD) cost, including the maximum depth layer decision (MDLD) and the depth intra mode decision (FDIMD). The MDLD is used to predict the maximum depth layer of each coding tree unit (CTU). The FIMD is used to skip unnecessary prediction modes. Zhang *et al.* [17] proposed a low complexity intra-mode selection algorithm, which can reduce the number of intra-mode by detecting the flat area and texture. When the flat region condition is satisfied, the corresponding intra-prediction modes are skipped. If the flat region condition is not satisfied, the direction of the edge is detected.

The second category focuses on fast intra CU size decision. Various heuristic methods [18]–[21] were proposed to skip or terminate the CU partition process in advance according to the features extracted manually. To be more specific, Wang *et al.* [18] proposed an algorithm to terminate the CU splitting process early based on the gray histogram of the current CU. Peng *et al.* [19] proposed two techniques to speed up the coding of the depth map, which includes fast intra mode decision and fast CU size decision. Zhang *et al.* [20] proposed a fast depth intra coding algorithm to speed up the quad-tree decision based on the good feature-corner point (CP). The proposed algorithm can adaptively extract CPs and pre-allocate the depth level of a coding quad-tree. In [21], an early termination method for intra block partitioning was proposed. The proposed method uses a simplified form associated with the current block and its first sub-block to terminate the partitioning.

Recently, several machine learning methods have been proposed to reduce the computational complexity in 3D-HEVC depth map coding [22]–[24]. The representative methods include [22] and [23]. Specifically, Saldanha *et al.* [22] presented a fast depth map coding for 3D-HEVC based on static decision trees. The proposed method uses data mining and machine learning to correlate the encoder context attributes and build the static decision trees. Each decision tree decides the partition of the depth map CU. In [23], an early determination of depth intra coding based on several Decision Trees was proposed. The determination includes the coding stage of Intra  $2N \times 2N$ , Intra  $N \times N$  or CU Splitting for the CUs. However, the above methods are based on probability or judgement of manual features so lack of robustness. As a new

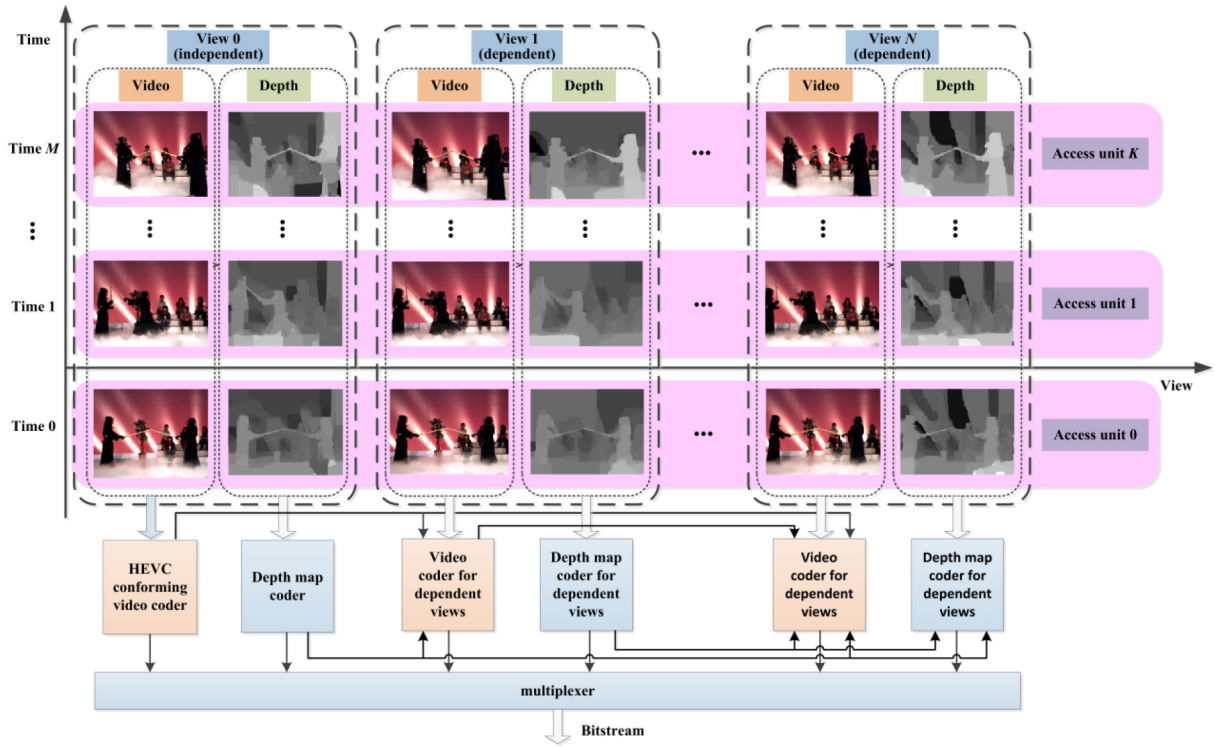


Fig. 1. 3D access unit structure and coding order.

research method in recent years, many scholars have applied CNN to the fast coding [25]–[33]. Such a solution was applied in different video coding domains, like the previous generation video coding standard HEVC [25], [26], the latest generation video coding standard VVC [27], [28], and the extended video coding standard 3D-HEVC [29]–[33]. For example, to reduce the HEVC complexity at both intra- and inter-modes, Xu *et al.* [25] proposed a deep learning approach to predict the CU partition based on CNN and long- and short-term memory (LSTM) network. To reduce the VVC complexity at intra mode, Tang *et al.* [27] presented an adaptive CU split decision based on the pooling-variable CNN. The proposed method targeting the various CU shapes utilizes the variable pooling layer size to retain the original information. To reduce the 3D-HEVC complexity at depth intra coding, Li and Yang [29] proposed a fast algorithm based on edge detection of a deep learning network. The proposed method uses a holistically nested edge detection (HED) [34] network to detect the edge of the depth map.

To reduce the 3D-HEVC complexity, our paper proposes a fast depth intra coding approach, which is based on deep learning, edge classification, and statistics. Our method differs from the above methods in four aspects.

- 1) Compared with traditional methods [15]–[21], CNN structure has a breakthrough in video feature extraction. Our method utilizes CNN to extract features of a depth map. This avoids the problem of obtaining prior information in advance and solves the problem of judgment accuracy decreasing under complex situations.
- 2) Compared with machine learning methods [22]–[24], our method can automatically extract video features

without manual extraction or complex probability calculation. This avoids learning from a large amount of complex data.

- 3) Compared with neural network methods in 3D-HEVC [29]–[33], our proposed DEC-CNN structure combines traditional methods with deep learning networks. This can reduce the coding complexity while ensuring the quality of synthetic views. To the best of our knowledge, this is the first attempt to utilize a CNN structure, edge classification, and statistics for predicting the CU partition.
- 4) To train the DEC-CNN and classify the edge complexity of depth maps, for the first time, we construct a database containing CUs of depth map. In contrast, other work has never built a database for a depth map. Techniques such as [29] rely on a trained network and do not need the database for secondary training, while the database in [30] is obtained by pre-coding standard sequences in 3D-HEVC.

### III. 3D-HEVC CODING STRUCTURE AND MOTIVATION

#### A. 3D-HEVC Coding Structure

Compared with 2D video, 3D video has a different access unit structure and coding order. Fig. 1 shows the 3D video access unit structure and coding order based on MVD video format. In Fig. 1, each texture video appears in a pair with a depth map. The texture video and the depth map are encoded according to the access unit. As shown in Fig. 1, the access unit 0-k contains all texture videos and depth maps at the same time point. In addition, the coding order does not need

to be consistent with the timing of acquiring the video image. Generally, the coding of texture video and depth map in the current access unit can refer to the encoded texture video and depth map in the previous access unit.

In 3D-HEVC coding structure, texture video and depth map coding of the independent view and other views is based on the HEVC coding structure. However, compared with HEVC, 3D-HEVC has made technical expansion. The 3D video, which needs to be encoded, includes texture videos of three views and depth maps of the corresponding three views. Fig. 1 is the 3D access unit structure and coding order under 3D-HEVC.

As shown in Fig. 1, the texture videos and depth maps of the 3D-HEVC coding structure originate from the same time, but their positions are different. The texture video and depth map exist in the same access unit. In the same access unit, firstly, the texture video and depth map corresponding to the independent view are encoded by the unmodified HEVC encoder. Secondly, the modified HEVC encoder is used to encode texture video and the depth map corresponding to other views [7]. Specifically, in the 3D-HEVC standard, 3D video is encoded in turns by the access unit. The texture videos and depth maps taken at the same view position have the same view sequence number. Because each access unit contains texture videos and depth maps of all views at the same time, all video frames in the same access unit have the same picture order count (POC). The independent view is also called the main view. This is usually called view 0. The texture video in view 0 is encoded by the original HEVC encoder. The improved HEVC encoder is used to encode all texture videos and depth maps in other auxiliary views (or dependent views). Generally speaking, the texture videos in the main view and auxiliary views are encoded before the depth map.

### B. Motivation

The characteristics of a depth map mean it contains large smooth areas and the edge is obvious, so it needs more accurate coding. Although 3D-HEVC has added prediction modes, coding tools and complex algorithms to preserve the edge information in depth map, the coding time of depth map is more time-consuming than texture video. To study the depth map coding complexity, in the Common Test Conditions (CTC), we test six sequences under four pairs of Quantization Parameters (QPs). Each pair contains QP values corresponding to the texture video and depth map. The four QP-pairs are (25, 34), (30, 39), (35, 42) and (40, 45). The test sequences in the experiment contain two different resolutions:  $1024 \times 768$  and  $1920 \times 1088$ . For  $1024 \times 768$ , it includes *Balloons*, *Kendo*, and *Kendo*. For  $1920 \times 1088$ , it includes *Poznan\_Hall2*, *Poznan\_Street*, and *Undo\_Dancer*.

Fig. 2 shows the proportion of depth map coding in the total coding time. We can see that compared with the coding time of texture video, the coding time of the depth map is about 3-6 times, accounting for more than 80% of the total coding time on average. Fig. 3 shows the coding process of the depth map. We can see that in the 3D-HEVC depth intra coding, the corresponding PU size and the optimal prediction mode

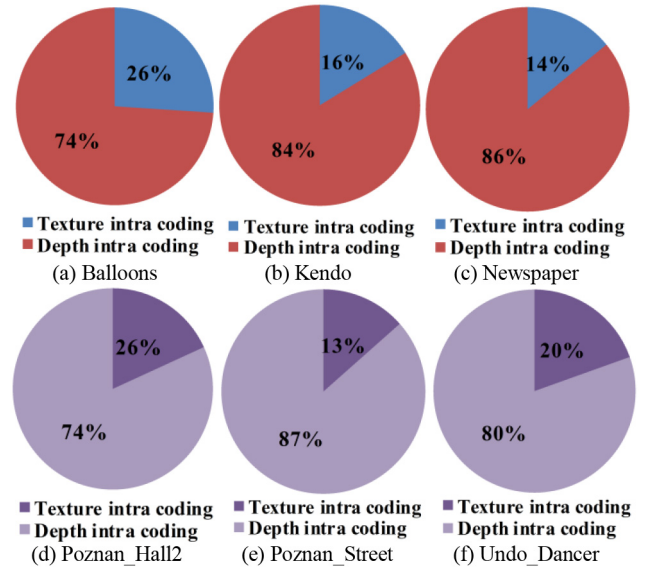


Fig. 2. Proportion of depth intra coding time in total coding time, (a)(b)(c) resolution:  $1024 \times 768$ , (d)(e)(f): resolution:  $1920 \times 1088$ .

TABLE I  
RELATIONSHIP BETWEEN  $N$  AND PU SIZE

| PU size | $64 \times 64$ | $32 \times 32$ | $16 \times 16$ | $8 \times 8$ | $4 \times 4$ |
|---------|----------------|----------------|----------------|--------------|--------------|
| $N$     | 3              | 3              | 3              | 8            | 8            |

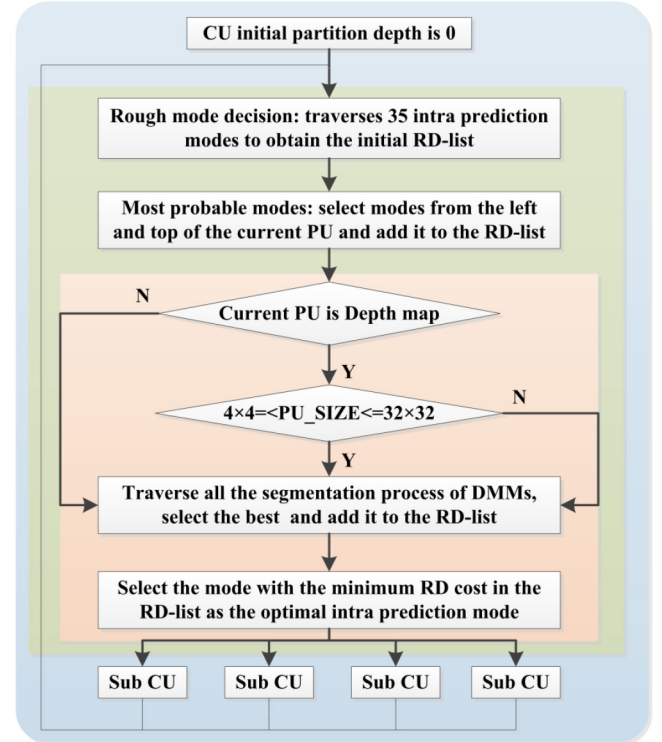


Fig. 3. Flowchart of 3D-HEVC depth intra coding.

of each PU need to be obtained. The specific process is as follows:

Step 1: In rough mode decision (RMD), formula (1) to (4) are used to calculate the SATD (Sum of Absolute Transformed



Differences) of 35 candidate intra prediction modes, and  $N$  modes corresponding to the minimum  $SATD$  are selected as the initial RD-list. The corresponding relationship between  $N$  and PU size is shown in Table I.

$$J_{\text{predSATD}} = SATD + \lambda_{\text{pred}} \times B_{\text{pred}} \quad (1)$$

$$\lambda_{\text{pred}} = \sqrt{\lambda_{\text{mode}}} \quad (2)$$

$$\lambda_{\text{mode}} = \sqrt{\lambda} \quad (3)$$

$$\lambda = \alpha \times W_k \times 2^{((QP-12)/3)} \quad (4)$$

where  $J_{\text{predSATD}}$  denotes the  $SATD$  value of prediction mode in current PU,  $SATD$  is the sum of the absolute values of the coefficients obtained after the Hadamard transformation of the predicted residuals,  $\lambda$  is Lagrange multiplier,  $B_{\text{pred}}$  is rate of current prediction mode,  $\alpha$  and  $W_k$  refers to the weight coefficient related to coding parameters and coding structure configuration.

Step 2: Construct RD-list by cooperating with most probable modes (MPM) and the  $N$  prediction modes selected by using the RMD process with the least cost of  $SATD$ .

Step 3: For the current PU, traverse all the segmentation process of DMMs, select the best and add it to the RD-list.

Step 4: Formula (5) and (6) are used to calculate RD cost of each mode in the RD-list, and the mode with the minimum RD cost is selected as the optimal intra prediction mode.

$$J_{\text{mode}} = (SSE_{\text{luma}} + w_{\text{chroma}} \times SSE_{\text{chroma}}) + \lambda_{\text{mode}} \times B_{\text{mode}} \quad (5)$$

$$w_{\text{chroma}} = 2^{(QP-QP_{\text{chroma}})/3} \quad (6)$$

where  $J_{\text{mode}}$  denotes RD cost value,  $SSE_{\text{luma}}$  is the sum of squares for error under luma component,  $SSE_{\text{chroma}}$  is the sum of squares for error under chroma component,  $\lambda_{\text{mode}}$  is Lagrange multiplier,  $B_{\text{mode}}$  is rate of current prediction mode,  $w_{\text{chroma}}$  refers to the weight coefficient related to coding parameters and coding structure configuration.

Step 5: If the maximum depth of quad-tree is not reached 3, divide each CU into four equal size sub CU (CU size range:  $64 \times 64$  to  $8 \times 8$ ) recursively.

Based on the above analysis, the final CTU partition structure needs to traverse the depth of 0-3. Each CU needs to traverse 35 prediction modes through RMD and RDO process. Taking a  $64 \times 64$  CTU as an example,  $11935 (1 \times 35 + 4 \times 35 + 16 \times 35 + 64 \times 35 + 256 \times 35 = 11935)$   $SATD$  cost calculations and  $2623 (1 \times 3 + 4 \times 3 + 16 \times 3 + 64 \times 8 + 256 \times 8 = 2623)$  RD cost calculations are required to obtain the final partition result. When the video is encoded at a higher frame rate and higher resolution, the number of CTUs to be encoded increases exponentially, which leads to a significant increase in coding complexity. Therefore, in the stage of CU partition, if the CU under some unnecessary depths can be selectively skipped, the prediction process can be skipped. In other words, we can avoid traversing all the depth to realize the rapid selection of CU depth and reduce the coding complexity.

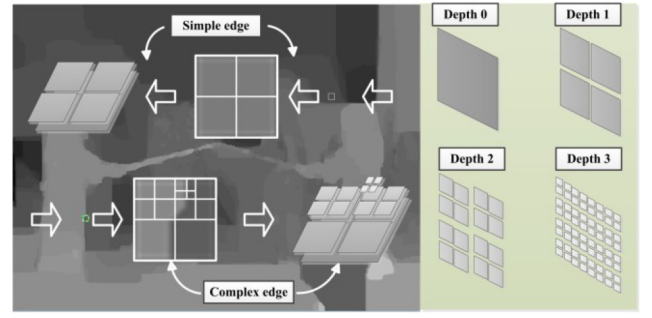


Fig. 4. Correlation between edge complexity and CU depth.

#### IV. FAST DEPTH INTRA CODING BASED ON DEPTH EDGE CLASSIFICATION NETWORK

According to the above analysis, 3D-HEVC adopts the partition method based on full depth search for all CTUs in the depth map. However, studies have shown that the depth of CTU is highly correlated with the edge of a depth map. For complex edge regions, large depth and small size CTU are usually used for coding. On the contrary, for simple edge areas, small depth and large size CTU are usually used for coding. As shown in Fig. 4, the maximum depth of the CTU in a complex edge area can reach 3, whereas the CTU in a simple edge area is only 1.

Based on this, CTU can be divided into two categories according to the edge complexity. One is simple edge CTU, the other is complex edge CTU. If the type of CTU can be predicted by the edge of the depth map before coding, and the depth range of CTU can be determined, then the coding process of CU beyond the depth range can be skipped, and the intra coding time of depth map can be reduced.

In view of the excellent performance of CNN in feature extraction and the edge characteristics of the depth map, this paper proposes a fast depth intra coding approach based on CNN. First, we established a database based on the independent view of depth map. Second, we constructed a DEC-CNN network to classify the two types of edges for the depth map. Finally, we utilized the pixel value of the binarized depth image to correct the above classification results.

##### A. Database Establishment

To establish the database, the CU partition data of the depth map from the independent view were collected.

In our database, 100 frames were selected from the standard test sequences (*Newspaper*:  $1024 \times 768$  and *Undo\_Dancer*:  $1920 \times 1088$ ), 50 frames from *Newspaper* and 50 frames from *Undo\_Dancer*. Considering the temporal correlation between video frames, frames 0-29 ( $2 \times 30$  frames, 11,520 CTUs) were selected as a training set, frames 260-269 ( $2 \times 10$  frames, 3,840 CTUs) as a validation set, and frames 280-289 ( $2 \times 10$  frames, 3,840 CTUs) as a testing set. This ensures the reliability and effectiveness of the dataset. Specifically, all video frames were coded by the 3D-HEVC test platform HTM-16.0 [35]. And four QP-pairs  $\{(25, 34), (30, 39), (35, 42) \text{ and } (40, 45)\}$  were applied to encode at All-Intra with the configuration file

TABLE II  
CONFIGURATION OF ESTABLISHED DATABASE

| Type       | Frame Range | Edge Complexity | No. of CTUs | QP       | No. of Samples |
|------------|-------------|-----------------|-------------|----------|----------------|
| Training   | 0-29        | Simple edge     | 5760        | (25, 34) | 46,080         |
|            |             |                 |             | (30, 39) |                |
|            |             |                 |             | (35, 42) |                |
|            |             |                 |             | (40, 45) |                |
|            |             | Complex edge    | 5760        | (25, 34) |                |
|            |             |                 |             | (30, 39) |                |
|            |             |                 |             | (35, 42) |                |
|            |             |                 |             | (40, 45) |                |
| Validation | 260-269     | Simple edge     | 1920        | (25, 34) | 15,360         |
|            |             |                 |             | (30, 39) |                |
|            |             |                 |             | (35, 42) |                |
|            |             |                 |             | (40, 45) |                |
|            |             | Complex edge    | 1920        | (25, 34) |                |
|            |             |                 |             | (30, 39) |                |
|            |             |                 |             | (35, 42) |                |
|            |             |                 |             | (40, 45) |                |
| Testing    | 280-289     | Simple edge     | 1920        | (25, 34) | 15,360         |
|            |             |                 |             | (30, 39) |                |
|            |             |                 |             | (35, 42) |                |
|            |             |                 |             | (40, 45) |                |
|            |             | Complex edge    | 1920        | (25, 34) |                |
|            |             |                 |             | (30, 39) |                |
|            |             |                 |             | (35, 42) |                |
|            |             |                 |             | (40, 45) |                |

*baseCfg\_3view+depth\_AllIntra* [36]. After encoding the depth map, the CTU structure for all the frames can be obtained. According to the high correlation between the depth of CTU and its edge complexity, the CTU with the maximum depth less than 1 is defined as the simple edge CTU. Conversely, the CTU that can reach depth 3 is defined as the complex edge CTU. Each CTU with its corresponding binary label, which indicates simple edge CTU or complex edge CTU, is a sample. Based on this, as reported in Table II, there are total 76,800 samples in our database. Furthermore, each set was equally divided into two subsets. One subset is with simple edge, and the other set is with complex edge. As such, our database contains video frames at different edge complexity, which ensures sufficient training data for learning to classify the edge and predict the CTU partition.

### B. Proposed DEC-CNN Approach

Fig. 5 illustrates the framework of our DEC-CNN approach. Specifically, our DEC-CNN approach contains pre-processing, DEC-CNN, and post-processing. In the following, we briefly present the details of these modules.

1) *Module 1*: Module 1 is the pre-processing unit, which can process the original CTU into a CTU that can be accepted by Module 2. In this module, first, the texture video and depth map corresponding to the independent view are found from the original 3D video sequence to be encoded (compared with the dependent view, the independent view are intermediate views in the process of view synthesis, which can provide more reference information). Second, the luma component is extracted from the original CTU of the depth map corresponding to the texture video in the independent view (compared with the chroma component, the luma component contains more structure information, and can extract more texture information).

We can see from Fig. 5 that DEC-CNN is fed with an entire CTU with the size of  $64 \times 64$ . We use the block-wise colorful feature map at the right represents CTU with different edge complexity has different feature distribution. The CTUs with various features contribute to the edge classification with different significance. This makes the information with different features reasonably used.

2) *Module 2*: Module 2 is the core module of the whole framework, named DEC-CNN, which is mainly used for edge feature extraction and classification. Compared with high-level features such as image semantics, an edge feature is considered as a shallow feature. Based on this, the shallow feature can be obtained by CNN with a simple structure. Meanwhile, the simple network structure means shorter running time, and it is easier to meet the demand of CTU for rapid depth selection. The details of the DEC-CNN are as follows.

The proposed DEC-CNN network consists of five convolutional layers, two dense blocks, a dropout layer, and a Softmax layer. As Fig. 5 shows, we adopt convolutional layers with the activation function of Rectified Linear Unit (ReLU), to extract edge features from the processed CTUs. Specifically, let  $W_{cm}$  and  $B_{cm}$  denote the weight and bias matrices of the  $m$ -th convolutional layer, the expression of our convolutional layer for the  $n$ -th processed CTU is defined as

$$C_0(\text{CTU}_n) = \text{CTU}_n, \quad (7)$$

$$C_m(\text{CTU}_n) = \text{ReLU}(W_{cm} * C_{m-1}(\text{CTU}_n) + B_{cm}), \quad 1 \leq m \leq M \quad (8)$$

where  $C$  represents the convolution layer and the total number of CNN layers is denoted as  $M$ . Between each convolution layer, in order to reduce the dimensional of features and the influence of irrelevant information, we use the max pooling layer to reduce the dimension of features. After the convolution

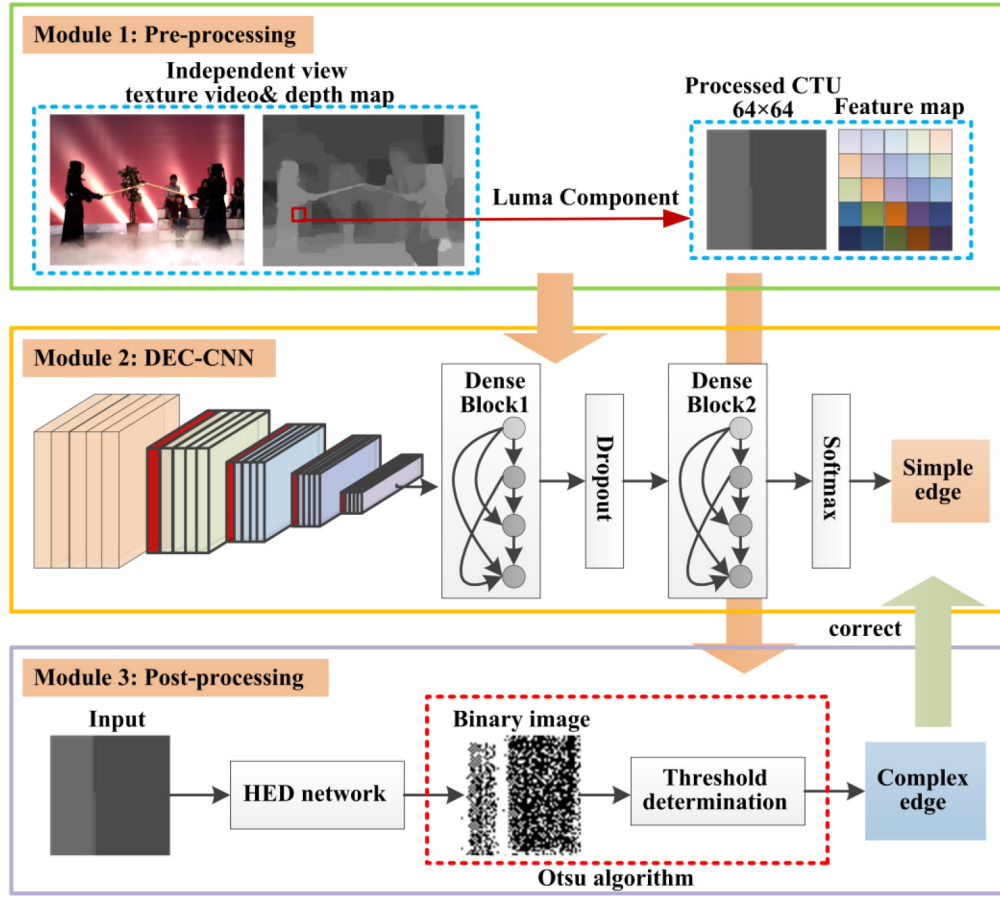


Fig. 5. Framework of our DEC-CNN approach.

layer, in order to realize the multiplexing and propagation of features, two dense blocks [37] with different dimensions and a dropout layer with probabilities of 50% are used. As shown in Fig. 5, in the dense block, the input of each layer comes from the output of all previous layers. Specifically, let  $l$  represents layer,  $x_l$  is the output of  $l$  layer, and  $H_l$  is a nonlinear transformation. The expression of the dense block is defined as

$$x_l = H_l([x_0, x_1, \dots, x_{l-1}]) \quad (9)$$

where  $[x_0, x_1, \dots, x_{l-1}]$  indicates that the output features of layer 0 to layer  $l-1$  is concatenated. The last layer is Softmax. The network structure using Softmax can get good performance in single label classification task. And it is a single label classification task to judge the complexity of CTU edge. Finally, an edge complexity class is output to prepare for further edge complexity class correction in Module 3.

3) *Module 3*: Module 3 is the post-processing unit, which can correct the classification results from Module 2. Table III gives several examples of CTUs with different edge features. We can see that different edge features have different pixel distribution. Table IV shows the classification results of edge complexity according to Table III. The above experiments show that we can use the pixel value of the image to determine the edge complexity of the image.

In Module 3, in view of the excellent performance of HED [34] network in edge detection, we first use HED network to obtain the edge detection graph of

64×64 CTU. The expression of HED process is defined as

$$H = E(X; \alpha) \quad (10)$$

where  $X$  represents the input 64×64 CTU,  $E$  is the edge detection mapping function,  $\alpha$  is the parameter (such as scale factor), and  $H$  is the image after detection. Second, in view of the image obtained by the above edge detection is a probability graph, considering that Otsu [38] algorithm is a common method to determine the adaptive threshold, we use this algorithm to binarize the image after edge detection into a more easily processed edge graph. The expression of Otsu process is defined as

$$\sigma_n^2 = W_0(U_0 - U)^2 + W_1(U_1 - U)^2 \quad (11)$$

where  $W_0$  represents the proportion of non-edge pixels in the whole image, and the average gray is  $U_0$ .  $W_1$  represents the proportion of edge pixels in the whole image, and the average gray is  $U_1$ . The average gray of the whole image is  $U$ , and  $\sigma_n^2$  is the between-class variance. Let the pixel  $k$  in the whole range of pixels of the image in turn to get the corresponding between-class variance  $\sigma_n^2$ . When  $\sigma_n^2$  is the largest, the corresponding  $k$  is the best threshold  $T$ . When the pixel value of a pixel is greater than or equal to the  $T$ , it will be determined as an edge region, otherwise, it will be determined as a non-edge region. Then, we use  $\hat{W}_0$  represents the proportion of pixels in the non-edge region, and  $\hat{W}_1$  represents the proportion of pixels in the edge region. If  $\hat{W}_1 \geq \hat{W}_0$ , the

TABLE III  
PIXEL VALUE OF CTUS WITH DIFFERENT EDGE FEATURES




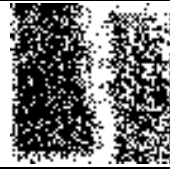


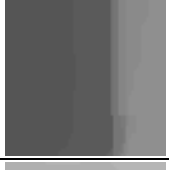
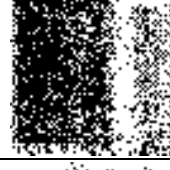




| Processed CTU   | Binary image  | Average pixel value | Processed CTU  | Binary image  | Average pixel value |
|---|---|---------------------|--|---|---------------------|
|  |  | 58.458              |  |  | 109.442             |
|  |  | 67.859              |  |  | 111.002             |
|  |  | 71.407              |  |  | 149.539             |

TABLE IV  
PIXEL VALUE OF CTUS WITH DIFFERENT EDGE FEATURES

| Average pixel value range | Edge complexity |
|---------------------------|-----------------|
| 0-70                      | simple          |
| 70-150                    | complex         |

current input CTU is judged as a complex CTU. Otherwise, the current input CTU is judged as a simple CTU. Finally, the classification results from Module 3 can be used to correct the result from Module 2.

### C. Loss Function for Training DEC-CNN Model

Since all the components of Module 2 are deep networks, they can be jointly trained in an end-to-end manner. Here, let  $l_n \in \{0, 1\}$  indicate whether the  $n$ -CTU is simple ( $l_n = 1$ ) or not ( $l_n = 0$ ). As such, given the processed CTUs, the labels  $\{l_n\}_{n=1}^N$  can be obtained. Then, considering that the edge complexity classification is essentially the problem of probability distribution of edge pixels, we consider applying categorical\_crossentropy to train our DEC-CNN network.

$$L = -\frac{1}{N} \sum_{n=1}^N (l_n \times \ln Y_n + (1 - l_n) \times \ln(1 - Y_n)) \quad (12)$$

where  $Y_n$  is the value predicted by DEC-CNN. However, there is a problem of over fitting in the training process. Considering that regularization can solve the problem of over fitting, we add regularization term to the above loss function. In addition, considering that edge features are low dimensional and dense, we choose the L2 regularization term, defined as

$$L = -\frac{1}{N} \sum_{n=1}^N (w l_n \times \ln Y_n + (1 - l_n) \times \ln(1 - Y_n)) + \frac{\lambda}{2N} \sum_w w^2 \quad (13)$$

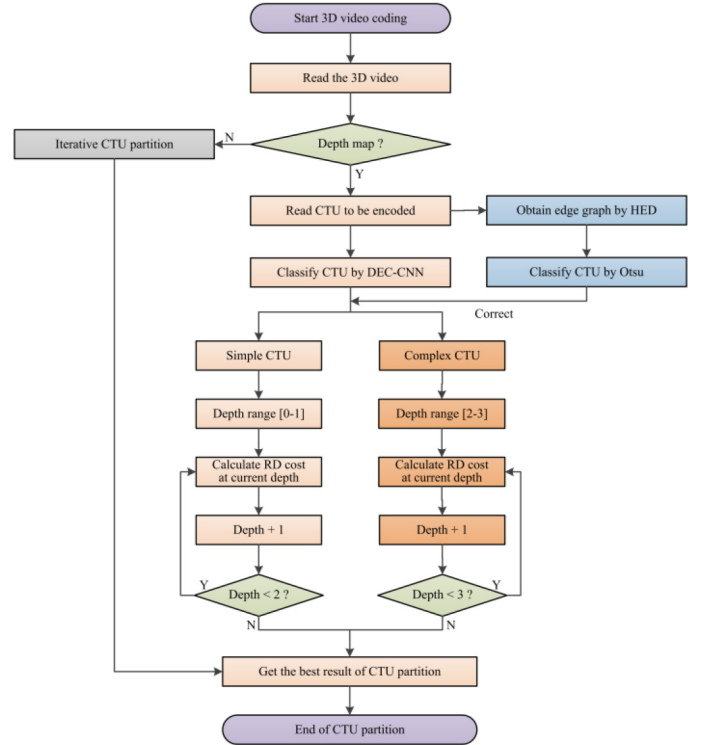


Fig. 6. Flowchart of fast depth intra coding.

where  $Y_n$  is the value predicted by DEC-CNN,  $\lambda$  is regularization parameter. And  $w$  represents the parameters in the network, which is from the regularization term. In addition,  $\frac{\lambda}{2N} \sum_w w^2$  is a penalty term.

### D. Flowchart of Proposed Approach

Fig. 6 shows the flowchart of the proposed fast depth intra coding.



TABLE V  
COMPARISON OF CODING TIME BETWEEN HTM-16.0 AND PROPOSED APPROACH

| Sequences                       | QP       | Encoding Time (s) |           | $\Delta T_1$ (%) |
|---------------------------------|----------|-------------------|-----------|------------------|
|                                 |          | HTM-16.0          | Proposed  |                  |
| Balloons                        | (25, 34) | 5096.3630         | 1563.2480 | 69.3             |
|                                 | (30, 39) | 5386.8340         | 1597.0500 | 70.3             |
|                                 | (35, 42) | 5321.2400         | 1439.6310 | 72.9             |
|                                 | (40, 45) | 5404.1850         | 1475.9590 | 72.6             |
| Kendo                           | (25, 34) | 4655.7810         | 1455.4880 | 68.7             |
|                                 | (30, 39) | 4951.6000         | 1370.0250 | 72.3             |
|                                 | (35, 42) | 4937.5680         | 1378.3520 | 72.1             |
|                                 | (40, 45) | 5140.6310         | 1363.2930 | 73.4             |
| Poznan_Hall2                    | (25, 34) | 10769.7350        | 3404.0010 | 68.4             |
|                                 | (30, 39) | 11527.4210        | 3336.4060 | 71.1             |
|                                 | (35, 42) | 11678.1040        | 3305.6690 | 71.7             |
|                                 | (40, 45) | 12464.9300        | 3328.9310 | 73.3             |
| Poznan_Street                   | (25, 34) | 18026.0640        | 4775.0170 | 73.5             |
|                                 | (30, 39) | 17528.8700        | 4323.0590 | 75.3             |
|                                 | (35, 42) | 15166.9820        | 4086.9930 | 73.1             |
|                                 | (40, 45) | 14730.8540        | 3838.4130 | 73.9             |
| Average1 (QP=(25, 34))          |          | 9636.9858         | 2799.4385 | <b>70.9</b>      |
| Average2 (QP=(30, 39))          |          | 9848.6813         | 2656.6350 | <b>73.0</b>      |
| Average3 (QP=(35, 42))          |          | 9275.9735         | 2552.6613 | <b>72.5</b>      |
| Average4 (QP=(40, 45))          |          | 9435.1500         | 2501.6490 | <b>73.5</b>      |
| Average5 (Resolution:1024×768)  |          | 5111.7753         | 1455.3808 | 71.5             |
| Average5 (Resolution:1920×1088) |          | 13986.6200        | 3799.8111 | 72.8             |
| Average                         |          | 9549.1977         | 2627.5960 | <b>72.5</b>      |

First, read the 3D video and then determine whether the current coded video is a depth map. Second, read the CTU to be encoded. For the CTU not from the depth map, utilize original iterative CTU partition method to get the best result of CTU partition. Otherwise, for the CTU from the depth map, use DEC-CNN to classify the current CTU. Meanwhile, utilize HED network to obtain the edge graph for the current CTU and use Otsu algorithm to classify the current CTU. Then, the classification result from Otsu is used to correct the result from DEC-CNN. If the two results are not consistent, the result from “HED + Otsu” is selected as the final result. Finally, the RD cost is calculated recursively for all CU in the depth range. And then the optimal CTU partition structure is obtained to realize the fast depth intra coding.

## V. EXPERIMENTAL RESULTS

In this section, experimental results are presented to validate the effectiveness of our approach in reducing 3D-HEVC complexity. In the first Section, the configuration of the experiments and the settings of the approach are discussed. In Section V-B, we compare the complexity reduction among different methods. Then, we compare the RD performance between the proposed method and the original method. Finally,

a series of ablation experiments are conducted to analyze the impact of major components in the proposed approach.

### A. Configuration and Settings

1) *Configuration of Experiments*: In the experiment, various types, different resolution, and frame rate of HTM standard video test sequences, including *Balloons*, *Kendo*, *Poznan\_Hall2*, and *Poznan\_Street*, were selected to evaluate our proposed approach. We experimented on the above four test sequences other than those used to establish the database. We set QP as (25, 34), (30, 39), (35, 42), and (40, 45). The final number of experimental frames is 20. In HTM-16.0, the configuration was applied with the default configuration file *baseCfg\_3view+depth\_AllIntra*. The remaining parameters follow the prescribed conventional test conditions. We incorporated the DEC-CNN structure into 3D-HEVC codec to accelerate the process of 3D video depth map coding. All the experiments were conducted on Intel Xeon CPU E31230 @ 3.20GHz, 8.00GB RAM, and the Windows 10 Enterprise 64-bit operating system. Note that a GPU was used to accelerate the training speed, but it was disabled when testing the 3D-HEVC complexity reduction.

2) *Training Settings*: As introduced in Section IV-A, the standard test sequences (*Newspaper* and *Undo\_Dancer*) were

TABLE VI  
COMPARISON OF CODING TIME BETWEEN PROPOSED APPROACH AND OTHER APPROACHES

| Sequences                       | [21]             | [17]             | [23]             | [22]             | [29]             | Proposed         |
|---------------------------------|------------------|------------------|------------------|------------------|------------------|------------------|
|                                 | $\Delta T_2$ (%) | $\Delta T_3$ (%) | $\Delta T_4$ (%) | $\Delta T_5$ (%) | $\Delta T_6$ (%) | $\Delta T_1$ (%) |
| Balloons                        | 25.9             | 37.9             | 45.6             | 54.3             | 31.9             | 71.3             |
| Kendo                           | 26.3             | 36.5             | 51.3             | -                | 32.7             | 71.6             |
| Poznan_Hall2                    | 25.9             | 35.7             | 79.1             | 64.7             | 36.0             | 71.1             |
| Poznan_Street                   | 24.0             | 35.7             | 61.4             | -                | 36.7             | 73.9             |
| Average5 (Resolution:1024×768)  | 26.1             | 37.2             | 48.5             | 54.3             | 32.3             | 71.5             |
| Average5 (Resolution:1920×1088) | 25.0             | 35.7             | 70.3             | 64.7             | 36.4             | 72.8             |
| Average                         | <b>25.5</b>      | <b>36.5</b>      | <b>59.4</b>      | <b>59.5</b>      | <b>34.3</b>      | <b>72.5</b>      |

TABLE VII  
COMPARISON OF RD PERFORMANCE BETWEEN HTM-16.0 AND PROPOSED APPROACH

| Sequences     | video 0 | video 1 | video 2 | video PSNR / video bitrate | video PSNR / total bitrate | synth PSNR / total bitrate |
|---------------|---------|---------|---------|----------------------------|----------------------------|----------------------------|
| Balloons      | 0.0%    | 0.0%    | 0.0%    | 0.0%                       | 1.1%                       | 2.9%                       |
| Kendo         | 0.0%    | 0.0%    | 0.0%    | 0.0%                       | 0.8%                       | 12.5%                      |
| Poznan_Hall2  | 0.0%    | 0.0%    | 0.0%    | 0.0%                       | -0.9%                      | 10.6%                      |
| Poznan_Street | 0.0%    | 0.0%    | 0.0%    | 0.0%                       | -1.2%                      | 9.1%                       |
| 1024×768      | 0.0%    | 0.0%    | 0.0%    | 0.0%                       | 1.0%                       | 7.7%                       |
| 1920×1088     | 0.0%    | 0.0%    | 0.0%    | -0.4%                      | -1.1%                      | 9.8%                       |
| Average       | 0.0%    | 0.0%    | 0.0%    | 0.0%                       | -0.1%                      | 8.7%                       |

utilized to establish the database. We used 60 frames to train our DEC-CNN and validate the DEC-CNN on the other 20 frames. In training the model, the hyper-parameters were tuned on the validation sets. We used the RMSprop optimizer. The batch size for training was 1024, initial learning rate was 0.001, and the training epoch was 2000.

### B. Comparison of Complexity Reduction

First, we compare our approach with the original coding method HTM-16.0, heuristic methods [17], [21], machine learning methods [22], [23] and CNN-based methods [29]. Table V and Table VI reports the results of complexity reduction in  $\Delta T$ , defined as

$$\Delta T = \frac{T_{\text{reference}} - T_{\text{proposed}}}{T_{\text{reference}}} \times 100\% \quad (14)$$

where  $T_{\text{reference}}$  and  $T_{\text{proposed}}$  represent the coding time of the reference approach and the proposed approach, respectively. As observed in Table V, at QP = (25, 34), (30, 39), (35, 42), and (40, 45), our approach reduces the encoding complexity by 70.9%, 73.0%, 72.5% and 73.5% on average. Compared with HTM-16.0, the proposed approach yields an average 72.5% encoding time. Among them, the coding time of the *Poznan\_Street* sequence has the largest savings. Because the background of the *Poznan\_Street* sequence does not change and the object only moves in the foreground. To further assess the complexity reduction of the proposed approach, other methods are tested under the same test platform and coding performance evaluation method as this paper. From Table VI, the average

$\Delta T$  is 72.5%, significantly higher than the latest CNN-based 3D-HEVC complexity reduction approach [29] (34.3%), and the representative machine learning approaches [23] (59.4%) and [22] (59.5%). More complexity reduction can be obtained when comparing with the other two heuristic approaches.

### C. Comparison of RD Performance

Table VII shows the performance comparison of the proposed approach with HTM-16.0. In Table VII, video PSNR/video bitrate indicates the BD-rate of coded texture views over video bitrate. Video PSNR/total bitrate indicates the BD-rate of coded texture views over total bitrate. And synth PSNR/total bitrate indicates the BD-rate of synthesized views over total bitrate. It can be seen from Table VII, compared with the original coding algorithm, an average -0.1% and 8.7% BD-rate loss for coded views and synthesized views, respectively.

Fig. 7 shows the subjective results in synthesized view on videos *Balloons* at QP = (25, 34) and *Poznan\_Street* at QP = (35, 42). We can see that our DEC-CNN approach is able to reduce encoding time while maintaining acceptable coding performance. The reason for BD-rate loss is the datasets. Since when training the network, the datasets are made of limited video sequences. The data source is not rich, resulting in the limitations of the DEC-CNN network. In future experiments, some depth maps from other video sequences will be collected to expand the dataset, improve the network performance, and then improve the coding performance.

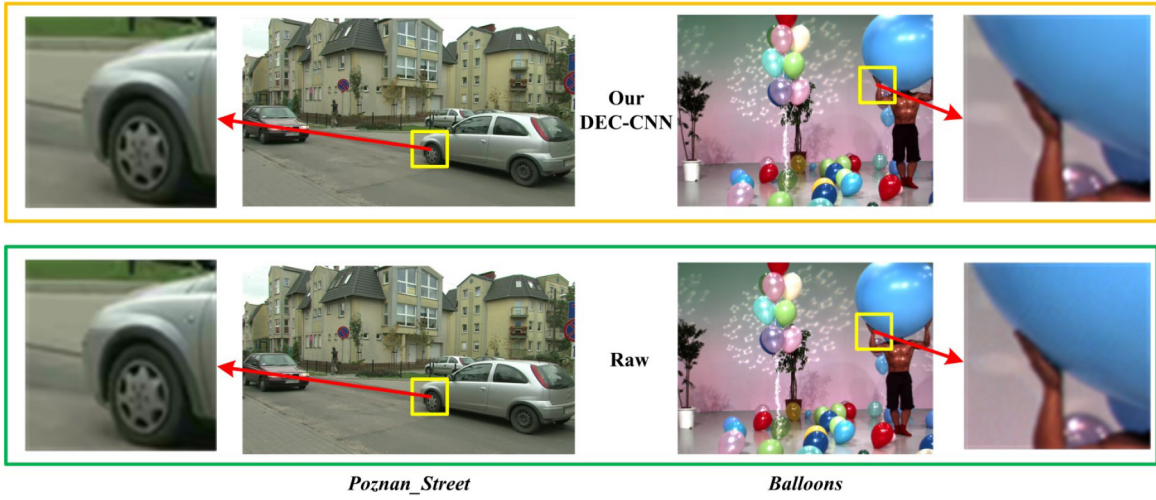


Fig. 7. Comparison of the subjective quality for the synth 1.75 (synthesized view 1.75) of the *Poznan\_Street* and synth 0.25 (synthesized view 0.25) of the *Balloons*.

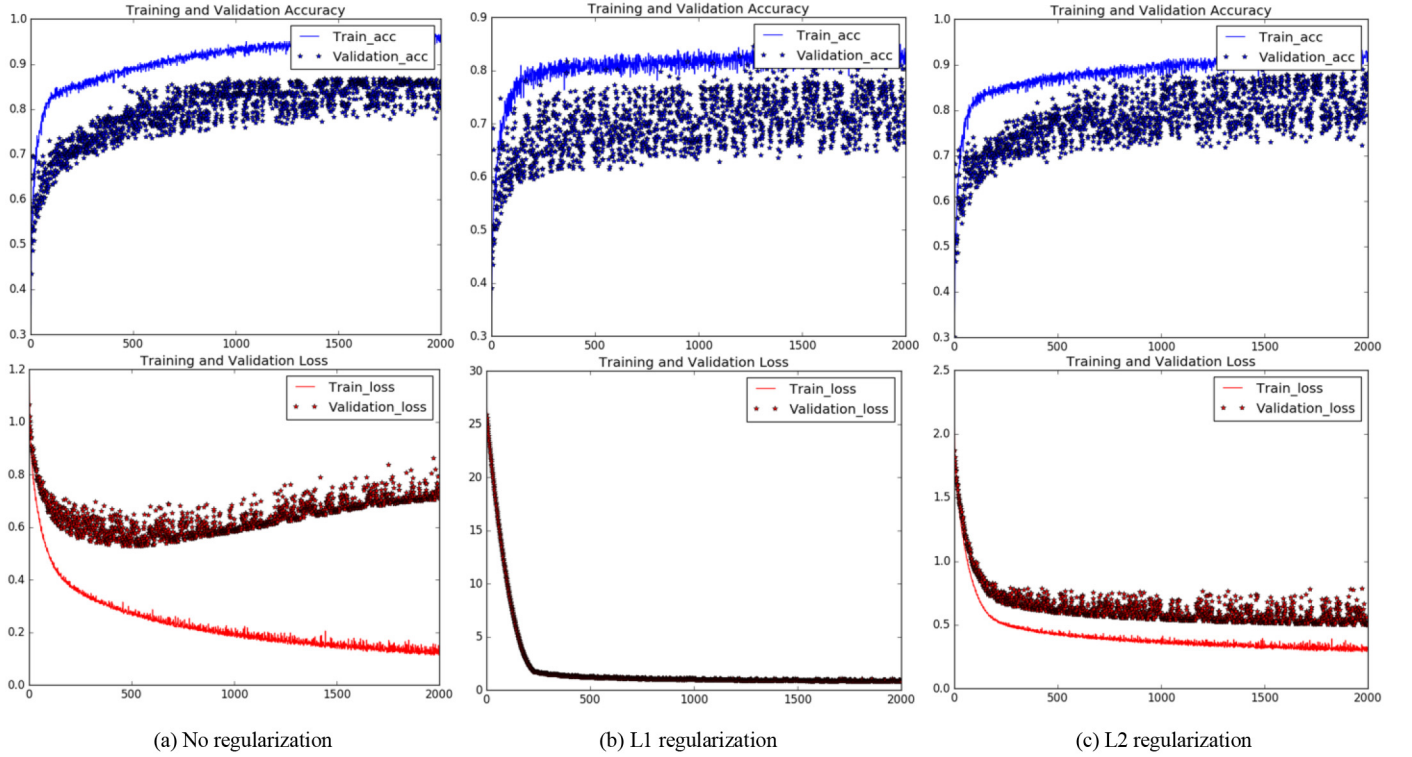


Fig. 8. Training results under different regularization.

#### D. Ablation Study

We further analyze the effectiveness of the proposed L2 regularization and the Module 3 in the framework of our DEC-CNN approach. Fig. 8 shows the training results under different regularization. From Fig. 8, we can see that in terms of accuracy and loss, regularization is better than no regularization. In addition, among the three regularization options, L2 regularization has the best performance on the validation set. Therefore, we choose to add L2 regularization to the categorical\_crossentropy loss function.

In order to further illustrate the necessity of CTU classification using Module 3, we do the following comparative experiments. The method of combining Module 1,

Module 2 and Module 3 to judge CTU category is defined as Method A. At the same time, only the method combining Module 1 and Module 2 is defined as Method B. Fig. 9 shows the quality of the synthesized views after 3D video coding using Method A and Method B, respectively. From Fig. 9, we can see that utilizing the framework without Module 3 (Method B) instead of our DEC-CNN approach (Method A) degrades the average PSNR by 0.4273dB for  $QP = (25, 34)$ . The results show that the Method A, which combines traditional method, can achieve more accurate CTU classification, so as to obtain better quality of the synthesized views. The above results prove the effectiveness of the proposed L2 regularization term

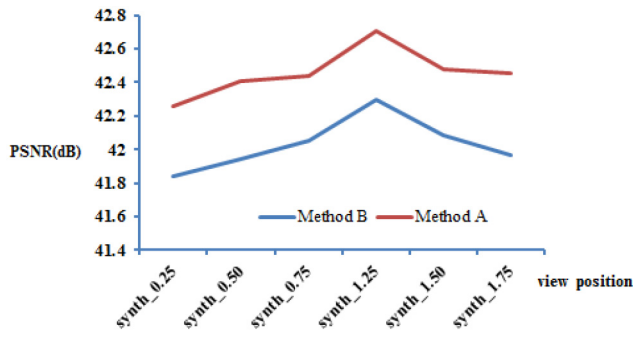


Fig. 9. Performance comparison between our DEC-CNN approach and DEC-CNN without Module 3.

and the Module 3 in the framework of our DEC-CNN approach.

## VI. CONCLUSION

In this paper, we have proposed the DEC-CNN approach for reducing the encoding complexity of 3D-HEVC, which is to determine the division of CTU by learning the edge classification of the depth map, instead of traversing the 0-3 depth of all CTU. First, we established a database based on the independent view of the depth map to deepen the DEC-CNN network. Then, we proposed a depth edge classification network, DEC-CNN, to classify the edges for the depth map for avoiding traversing all the depth to realize the rapid selection of CTU depth and reducing the 3D-HEVC complexity. More importantly, we proposed utilizing the pixel value of the binarized depth image to correct the edge classification results. As such, combining traditional methods with deep learning networks can reduce the coding complexity of a depth map while ensuring the quality of synthetic views. Finally, the experimental results validated that our DEC-CNN approach can be efficiently used to reduce 3D-HEVC complexity without any significant loss in RD performance for the synthesized views.

## REFERENCES

- [1] M. Cheon and J. Lee, "Subjective and objective quality assessment of compressed 4K UHD videos for immersive experience," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 7, pp. 1467–1480, Jul. 2018.
- [2] C.-H. Fu, Y.-L. Chan, H.-B. Zhang, S. H. Tsang, and M.-T. Xu, "Efficient depth intra frame coding in 3D-HEVC by corner points," *IEEE Trans. Image Process.*, vol. 30, pp. 1608–1622, Dec. 2020.
- [3] M. Wien, J. M. Boyce, T. Stockhammer, and W. Peng, "Standardization status of immersive video coding," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 5–17, Mar. 2019.
- [4] T. Li, L. Yu, H. Wang, and Z. Kuang, "A bit allocation method based on inter-view dependency and spatio-temporal correlation for multi-view texture video coding," *IEEE Trans. Broadcast.*, vol. 67, no. 1, pp. 159–173, Mar. 2021.
- [5] Y. Yang, Q. Liu, X. He, and Z. Liu, "Cross-view multi-lateral filter for compressed multi-view depth video," *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 302–315, Jan. 2019.
- [6] K. Gu, J. Qiao, S. Lee, H. Liu, W. Lin, and P. Le Callet, "Multiscale natural scene statistical analysis for no-reference quality evaluation of DIBR-synthesized views," *IEEE Trans. Broadcast.*, vol. 66, no. 1, pp. 127–139, Mar. 2020.
- [7] G. Tech, Y. Chen, K. Müller, J. Ohm, A. Vetro, and Y. Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 35–49, Jan. 2016.
- [8] K. Müller *et al.*, "3D high-efficiency video coding for multi-view video and depth data," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3366–3378, Sep. 2013.
- [9] W. Hamidouche, M. Raullet, and O. Déforges, "4K real-time and parallel software video decoder for multilayer hevc extensions," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 169–180, Jan. 2016.
- [10] Y. Zhang, L. Zhu, R. Hamzaoui, S. Kwong, and Y.-S. Ho, "Highly efficient multiview depth coding based on histogram projection and allowable depth distortion," *IEEE Trans. Image Process.*, vol. 30, pp. 402–417, Nov. 2020.
- [11] C. Park, "Efficient intra-mode decision algorithm skipping unnecessary depth-modeling modes in 3D-HEVC," *Electron. Lett.*, vol. 51, no. 10, pp. 756–758, May 2015.
- [12] K. Zhang, J. An, H. Huang, J. Lin, Y. Huang, and S. Lei, "Segmental prediction for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 11, pp. 2425–2436, Nov. 2017.
- [13] J. Lei, S. Li, C. Zhu, M. Sun, and C. Hou, "Depth coding based on depth-texture motion and structure similarities," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 2, pp. 275–286, Feb. 2015.
- [14] W. Sun, O. C. Au, L. Xu, Y. Li, and W. Hu, "Seamless view synthesis through texture optimization," *IEEE Trans. Image Process.*, vol. 23, no. 1, pp. 342–355, Jan. 2014.
- [15] R. Zhang, K. Jia, P. Liu, and Z. Sun, "Edge-detection based fast intra-mode selection for depth map coding in 3D-HEVC," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Sydney, NSW, Australia, 2019, pp. 1–4.
- [16] T. Li, H. Wang, Y. Chen, and Y. Li, "Fast depth intra coding based on spatial correlation and rate distortion cost in 3D-HEVC," *Signal Process. Image Commun.*, vol. 80, pp. 1–14, Feb. 2020.
- [17] R. Zhang, K. Jia, P. Liu, and Z. Sun, "Fast intra-mode decision for depth map coding in 3D-HEVC," *J. Real Time Image Process.*, vol. 17, no. 5, pp. 1637–1646, Oct. 2020.
- [18] C. Wang, G. Feng, C. Cai, and X. Han, "Fast CU size decision algorithm for depth map intra-coding in 3D-HEVC," *Commun. Technol.*, vol. 50, no. 4, pp. 655–661, Apr. 2017.
- [19] K. Peng, J. Chiang, and W. Lie, "Low complexity depth intra coding combining fast intra mode and fast CU size decision in 3D-HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Phoenix, AZ, USA, 2016, pp. 1126–1130.
- [20] H. Zhang, Y. Chan, C. Fu, S. Tsang, and W. Siu, "Quadtree decision for depth intra coding in 3D-HEVC by good feature," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Shanghai, China, 2016, pp. 1481–1485.
- [21] T. Li, L. Yu, S. Wang, and H. Wang, "Simplified depth intra coding based on texture feature and spatial correlation in 3D-HEVC," in *Proc. Data Compression Conf.*, Snowbird, UT, USA, 2018, p. 421.
- [22] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Fast 3D-HEVC depth map encoding using machine learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 3, pp. 850–861, Mar. 2020.
- [23] C. Fu, H. Chen, Y. Chan, S. Tsang, H. Hong, and X. Zhu, "Fast depth intra coding based on decision tree in 3D-HEVC," *IEEE Access*, vol. 7, pp. 173138–173147, 2019.
- [24] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Fast 3D-HEVC depth maps intra-frame prediction using data mining," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Calgary, AB, Canada, 2018, pp. 1738–1742.
- [25] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan, "Reducing complexity of HEVC: A deep learning approach," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5044–5059, Oct. 2018.
- [26] K. Kim and W. W. Ro, "Fast CU depth decision for HEVC using neural networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 5, pp. 1462–1473, May 2019.
- [27] G. Tang, M. Jing, X. Zeng, and Y. Fan, "Adaptive CU split decision with pooling-variable CNN for VVC intra encoding," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Sydney, NSW, Australia, 2019, pp. 1–4.
- [28] J. Zhao, Y. Wang, and Q. Zhang, "Adaptive CU split decision based on deep learning and multifeature fusion for H.266/VVC," *Sci. Program.*, vol. 2020, Aug. 2020, Art. no. 8883214.
- [29] Y. Li and J. Yang, "Fast intra coding algorithm for depth map in 3D-HEVC," *J. Optoelectron. Laser*, vol. 31, no. 2, pp. 222–228, Feb. 2020.
- [30] Y. Chen, L. Yu, T. Li, H. Wang, and S. Wang, "Fast CU size decision based on AQ-CNN for depth intra coding in 3D-HEVC," in *Proc. Data Compression Conf. (DCC)*, Snowbird, UT, USA, 2019, p. 561.
- [31] J.-R. Lin *et al.*, "Visual perception based algorithm for fast depth intra coding of 3D-HEVC," *IEEE Trans. Multimedia*, early access, Mar. 31, 2021, doi: [10.1109/TMM.2021.3070106](https://doi.org/10.1109/TMM.2021.3070106).



- [32] L. Zhu, Y. Zhang, S. Wang, H. Yuan, S. Kwong, and H. H.-S. Ip, "Convolutional neural network-based synthesized view quality enhancement for 3D video coding," *IEEE Trans. Image Process.*, vol. 27, no. 11, pp. 5365–5377, Nov. 2018.
- [33] C. Liu, K. Jia, P. Liu, and Z. Sun, "Fast depth intra coding based on layer-classification and CNN for 3D-HEVC," in *Proc. Data Compression Conf. (DCC)*, Snowbird, UT, USA, 2020, p. 381.
- [34] S. Xie and Z. Tu, "Holistically-nested edge detection," *Int. J. Comput. Vis.*, vol. 125, nos. 1–3, pp. 3–18, Dec. 2017.
- [35] *3D-HEVC Test Model*. Accessed: Jul. 2018. [Online]. Available: [https://hevc.hhi.fraunhofer.de/svn\\_3DVCSoftware/tags/HTM-16.0/](https://hevc.hhi.fraunhofer.de/svn_3DVCSoftware/tags/HTM-16.0/).
- [36] D. Rusanovsky, K. Muller, and A. Vetro, *Common Test Conditions of 3DV Core Experiments*, document ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on 3D Video Coding Extension Development, 7th Meeting, ITU and ISO/IEC, Geneva, Switzerland, 2014, pp. 11–17.
- [37] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, 2017, pp. 2261–2269.
- [38] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.



**Chang Liu** received the B.E. degree from Shandong Technology and Business University, Shandong, China, in 2016. She is currently pursuing the Ph.D. degree in electronic science and technology with Beijing University of Technology, Beijing, China.

Her research interest includes video coding for HEVC, 3D-HEVC, and VVC.



**Kebin Jia** (Member, IEEE) received the M.S. and Ph.D. degrees in information and communication engineering from the University of Science and Technology of China, Anhui, China, in 1990 and 1998, respectively.

He is currently a Professor and the Director of First-Class Disciplines Construction Office, Beijing University of Technology, Beijing, China, where he is currently a Full Professor with the Faculty of Information Technology, and the Director of Institute of Multimedia Information Processing and Imaging Technology. He has served as PI for more than 15 research projects from the National Natural Science Foundation of China (NSFC), 973 National Basic Research Program, and 863 Program. He has published more than 200 research publications and authored two books in these areas. His research interests include multimedia and database systems, content-based image/video retrieval, image/video coding and processing, data mining, and pattern recognition. His group has received the Award of Outstand and Innovator Group Award of the Committee of Education of Beijing. He is a Senior Member of the Chinese Institute of Electronics.



**Pengyu Liu** (Member, IEEE) received the B.S. degree in communication and information system from Jilin University, Jilin, China, in 2004, and the Ph.D. degree in circuit and system engineering from the Beijing University of Technology, Beijing, China.

She is currently an Associate Professor with the Faculty of Information Technology, Beijing University of Technology. Her research interest includes the development of multimedia information processing and intensive study of video coding technology.