Analog Integrated Circuits and Signal Processing (2021) 108:555–568
https://doi.org/10.1007/s10470-021-01911-1

Hardware and coding efficiency assessment of 3D-HEVC DIS tool using alternative similarity criteria

Vinicius A. Borges[1] · Murilo R. Perleberg[1,2] · Vladimir Afonso[1,3] · Marcelo S. Porto[1,2] · Luciano V. Agostini[1,2]

Received: 21 January 2021 / Revised: 15 June 2021 / Accepted: 26 June 2021 / Published online: 3 July 2021

Abstract
3D-HEVC is the state-of-the-art standard to compress three-dimensional videos. One of the 3D-HEVC novel tools is the DIS tool, which is used to efficiently compress smooth and homogeneous areas of depth maps by using four different prediction modes. The decision of which DIS mode will be used is done through the SVDC similarity criterion in the DIS original definition. This article proposes the substitution of the complex SVDC criterion for simpler and more hardware friendly criteria as SATD, SSE, and SAD. These alternative criteria were evaluated in terms of encoding efficiency and hardware impacts in comparison with the SVDC. Dedicated DIS hardware were designed using each one of each criterion and these designs were described in VHDL and synthesized for TSMC 40 nm. The best results were found with SAD criteria, with losses of only 0.2% in coding efficiency and with expressive gains of more than 50 times in power and more than 35 times in area, when compared with SVDC. The reached results showed that the use of a simpler similarity criterion is an important alternative to be used in DIS tool, mainly if an efficient hardware design is required.

Keywords 3D-HEVC · Depth intra skip · Hardware design · Similarity criterion

1 Introduction

Currently, there has plenty of technologies available for the population. Mainly in the world's actual situation, where the people need to stay at home because of the pandemic of Coronavirus, there is an expressive increase in the internet traffic [1] caused by entertainment and communication, since digital videos are commonly used in these scenarios [2]. In the near future, 3D contents, like immersive multimedia, augmented and mixed realities, tend to be massively used.

Digital videos require a high amount of data to be represented and, therefore, efficient compression techniques are necessary to allow the spread use of this type of media as we observe nowadays. This problem is still higher when 3D videos are required, since much more data are necessary to represent 3D videos. The state-of-the-art for 3D video encoding is the 3D-High Efficiency Video Coding (3D-HEVC). The 3D-HEVC was published in 2015 [3], as an extension of the High Efficiency Video Coding (HEVC) [4]. The Versatile Video Coding (VVC) is the current state-of-the-art standard for 2D video coding [5], however, VVC does not support 3D video compression yet.

The 3D-HEVC inherits several HEVC encoding tools [4], while it introduces a new format called Multi-View plus Depth (MVD) [4, 6], which allows the use of several views to encode the scene, thus highly increasing the amount of data of 3D videos. Each view is captured from a different point of view and composed of two channels, which can be seen in Original Views of Fig. 1. While the

✉ Vinicius A. Borges
vdaborges@inf.ufpel.edu.br

Murilo R. Perleberg
mrperleberg@inf.ufpel.edu.br

Vladimir Afonso
vafonso@inf.ufpel.edu.br

Marcelo S. Porto
porto@inf.ufpel.edu.br

Luciano V. Agostini
agostini@inf.ufpel.edu.br

[1] Video Technology Research Group (ViTech), Federal University of Pelotas (UFPel), Pelotas, Brazil

[2] Postgraduate Program in Computing (PPGC), Pelotas, Brazil

[3] Sul-Rio-Grandense Federal Institute (IFSul), Pelotas, Brazil

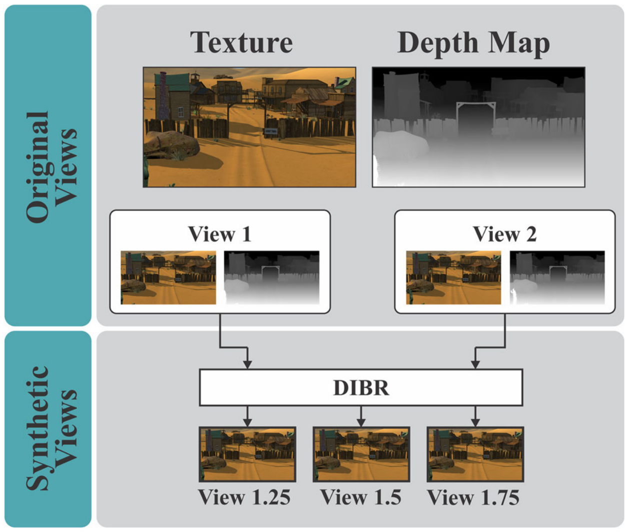


Fig. 1 Generation of synthetic views at the decoder side through the DIBR process

texture channel represents the color and the brightness of the frame, the depth-map channel represents the distance between the objects and the camera [7].

The 3D-HEVC uses the Depth-Image-Based Rendering (DIBR) process to work with different views [4, 7]. The DIBR process uses the information of the two channels at the decoder side to perform the generation of new virtual intermediate texture views (views not captured by cameras) [7]. The DIBR process is depicted in Fig. 1, where three synthetic views are generated based on the two channels (texture and depth map) of two original views. This process allows the capture, processing, and transmission of a reduced number of views on the encoder side, aiming at obtaining a higher number of views at the decoder side [4, 7].

The 3D-HEVC divides the texture and depth maps video frames in blocks called Coding Tree Units (CTU) [4, 8], which can be subdivided into a quadtree structure where each leaf is called Coding Unit (CU). The maximum CTU size is 64 $\times$ 64 samples by default of the 3D-HEVC reference software (3D-HTM) [9]. The CU is squared, and its sizes can vary from 64 $\times$ 64 samples to 8 $\times$ 8 samples [8].

The Intra prediction inherits the HEVC prediction modes (Planar, DC, and Angular) and introduces three new modes to encode depth maps: Depth Modeling Modes 1 and 4 (DMM-1 and DMM-4), and Depth Intra Skip (DIS) [4]. The DIS is used to encode smooth and homogeneous areas specifically, where these areas are filled with neighboring sample values. DIS has four different modes as will be detailed in the next section. Among the new modes for depth-map prediction, the DIS is the most relevant mode, since according to the analysis presented in section II it is responsible for encoding more than 80% of the intra-predicted depth-map samples. Also, the DIS have a high complexity to be processed as also presented in section II, since by default the 3D-HTM [9] adopts the Synthesized View Distortion Change (SVDC) as the similarity criterion to evaluate the best DIS mode to predict a CU and SVDC requires complex rendering features to be computed [10, 11].

This article presents an investigation about the use of simpler distortion criteria to be used in DIS tool in substitution of SVDC. Sum of Absolute Transformed Differences (SATD) [12, 13], Sum of Squared Errors (SSE) [12], and Sum of Absolute Differences (SAD) [12, 14] were evaluated from coding efficiency through hardware impacts. As presented in the rest of this article, the encoding efficiency and computational complexity of DIS processing are highly dependent of the similarity criterion adopted to evaluate and to define the best DIS mode to encode each CU. With the presented investigation was possible to conclude that the use of a simpler distortion criteria, as SATD, SSE or SAD, instead of SVDC, caused a small loss in coding efficiency (0.2% in BD-rate with impressive gains in terms of hardware results, mainly in power (till 50 times lower power) and area (till 35 times smaller area).

Two other works focused on hardware designs for the DIS modes are presented in the literature. The work in [14] proposed the use of SAD as the similarity criterion for the DIS mode. However, it does not compare its results with an architecture adopting the SVDC. In our previous work in [15], we presented an architecture for the DIS using the SVDC criterion.

Considering the relevance of DIS to encode videos in 3D-HEVC, probably this tool will be used in future standards to encode 3D videos. Then, the presented investigation of more efficient and hardware friendly solutions to implement this tool is the main contribution of this work, since the use of simpler similarity criteria presented a small impact in coding efficiency with an expressive impact in throughput, area, and power. The evaluated criteria are not novel, but the novelty is the application of these criteria inside the DIS tool since this is the first work in the literature with this type of investigation. The reached results were promising, as will presented in the rest of this article.

2 3D-HEVC DIS background

DIS was created to efficiently compress depth maps areas with homogeneous regions, where the residual coding can be potentially avoided by using the samples of neighboring blocks to represent the current block. DIS tool uses four prediction modes: Horizontal Single Depth (SD_H);

Horizontal Intra Prediction (IP_H); Vertical Single Depth (SD_V); and Vertical Intra Prediction (IP_V) [4, 11].

A representation of these modes can be seen in Fig. 2, considering the processing of an 8 × 8 CU size. The SD_H and SD_V consider only one previously processed depth neighboring sample to represent the current block, filling this entire block with the same value of this neighboring sample [11, 14]. As one can observe in Fig. 2, the SD_H uses one sample from the left neighbor block located in the position $A_{N/2}$, while the SD_V uses the sample from the above row located in the position $B_{N/2}$, to compress the entire current block, being N the representation of the block size [11, 14].

The IP_H and IP_V modes represent the current block by coping the horizontal or vertical neighboring samples to all current block rows or columns, as presented in Fig. 2. The IP_H copy the values from the left neighboring samples for the respective rows, where the entire line adopts the same value. The IP_V has a similar idea, however it copies the values from the above neighboring samples for the respective columns [11, 14].

Due to the homogeneous regions of the scene, where the neighboring blocks can have samples with very similar values, these four modes can generate very similar blocks. Although, small changes on the depth block can have a huge impact on synthetic views considering the occlusion/disocclusion of objects [16, 17], and, therefore, these DIS modes must be precisely compared.

As previously mentioned, the 3D-HTM adopts by default the SVDC metric to compare the four DIS modes and to guarantee the required precision when deciding the best mode. The SVDC evaluates the distortion produced by each mode in the texture of a synthetic view and, therefore, it requires rendering functions [10, 11]. The next subsection presents an analysis of the DIS relevance and complexity when considering the SVDC metric. In the sequence, the SVDC metric and the low complexity metrics SATD, SSE, and SAD are explained.

2.1 Analysis of DIS relevance and complexity

As cited before, among the new modes for depth-map prediction included in 3D-HEVC, the DIS is the most relevant mode since it is responsible for encoding the vast majority of the intra-predicted depth-map samples.

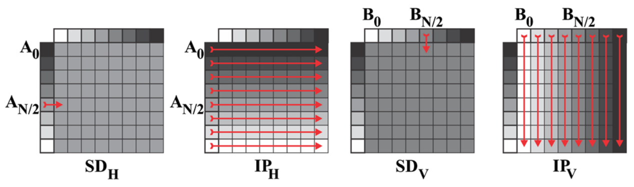


Fig. 2 DIS modes. The highlighted samples represent the neighboring samples. Samples to the left are from A_0 to A_N, and samples above are from B_0 to B_N

A set of experiments were done to support this conclusion. These experiments respected the Common Test Conditions (CTC) document [18] recommended by the Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3 V). The CTC specifies eight 3D video sequences, including five 1920 × 1088 sequences (*GT_Fly*, *Poznan_Hall2*, *Poznan_Street*, *Shark*, and *Undo_Dancer*) and three 1024 × 768 sequences (*Balloons*, *Kendo*, and *Newspaper*). It also specifies three views (texture + depth map) of these video sequences for the experiments. Each video sequence used between 200 and 300 Access Units (AU). Each AU encapsulates all texture pictures and their respective depth maps that belong to the same time instant [4]. Finally, the pairs of Quantization Parameters (QPs) defined in the CTC document are: 25/34, 30/39, 35/42, and 40/45 (texture/depth). The experiments were done using the 3D-HTM version 16.2 [9].

According to this first set of experiments when All-Intra Configuration [18] is considered and only Intra-frame is allowed, 81.07% of the depth maps samples are encoded with DIS [14]. On the other hand, when Random Access configuration [18] is considered, when Inter-frame and Intra-frame tools are available together, 81.13% of the Intra-frame predicted samples are encoded with DIS modes. But in the Random-Access configuration, 92.76% of the depth map samples are predicted with inter-frame tools, while 7.24% of depth map samples are predicted with intra-frame prediction modes.

As the DIS is the most relevant mode on 3D-HEVC intra prediction, it also has an important computational cost to be processed. DIS tool support 64 × 64, 32 × 32, 16 × 16, and 8 × 8 CU sizes. Also, by default, the 3D-HTM [9] adopts the SVDC as the similarity criterion to evaluate the best DIS mode to predict a CU. The SVDC is a complex similarity criterion since it requires rendering features to be computed [10, 11]. Since the rendering features are responsible from 18% up to 26% of the 3D-HTM total encoding time [19], the DIS computational effort is relevant. Considering All-Intra configuration, the processing of depth maps has a high complexity, being responsible for up to 86.3% of total encoding time. From the depth maps processing, DIS tool is responsible for 5.6% of the computational complexity, while DMM-1, DMM-4 and other Intra modes inherited from the HEVC are responsible for 27.2%, 3.2% and 12.1%, respectively. The other 51.9% are due to residual encoding. When Random-Access configuration is used, the Intra-frame tools require 12.52% of total encoding time [20] and DIS is responsible for 27.5% of the Intra-frame computational effort.

2.2 Synthesized view distortion change

The definition of the SVDC is given by (1) [10, 11], where the $\widetilde{S'_T}$ represents a texture synthesized with a depth map that contains the distorted depth data for the current block B, S'_T means a texture rendered using encoded depth data for already encoded blocks and original depth data for other blocks, $S'_{T,Ref}$ denotes a reference texture rendered using the original texture and the original depth map, I represents the set of all samples in the synthesized view, and (x, y) means a position in a given set I.

$$SVDC = \sum_{(x,y)\in I} \left[\widetilde{S'_T}(x,y) - S'_{T,Ref}(x,y)\right]^2 - \sum_{(x,y)\in I} \left[S'_T(x,y) - S'_{T,Ref}(x,y)\right]^2 \tag{1}$$

The information required as input for the SVDC calculation is represented in Fig. 3, where three different renders are required for generating S'_T, $S'_{T,Ref}$, and $\widetilde{S'_T}$ [11]. The $S'_{T,Ref}$ is rendered using the original data for both the current view and for a neighboring view. The S'_T is rendered considering the already encoded data for the depth map channel of the current view, except the current block being processed, while the other channel is composed of the original data. Lastly, the $\widetilde{S'_T}$ is rendered considering the already encoded data for the depth map channel of the current view, and considering the current block being encoded with the DIS mode being evaluated, while the other channel is composed of the original data.

Therefore, to render each CU, which is mandatory to use the SVDC metric, it is necessary a render model. This model is divided into initialization and the partial re-rendering [10, 11]. The render model starts with the process of synthesizing views of the original views, where the complete synthesized view is generated using the original input depth maps and the input textures. These input depth maps are also stored as the renderer model depth state of the left view ($S_{D,l}$) and of the right view ($S_{D,r}$), and the rendered view as the synthesized view state $\widetilde{S'_T}$ [10, 11]. This part is performed before the encoding of the depth map is started.

The partial re-rendering is responsible for updating the renderer model when the DIS mode to a block B is finished and the depth data of the predicted block is known [11]. This depth data will be a crucial part of the process, changing one of the depth states ($S_{D,l}$ or $S_{D,r}$), and re-rendering only local parts of the synthesized view state S'_T.

Inside the initialization and the partial re-rendering, there are four major steps to render each block [11], being warping, interpolation, hole filling, and blending, as presented in Fig. 4. The warping is the first step, being responsible for shifting the block samples for generating the new synthesized view [11], as can be seen in Fig. 4. The shift operation is computed by (2) according to the depth sample value and the camera parameters [11]. On (2), the d_v and v represent the position of the block samples in the new synthesized view and the depth sample value, respectively, while s, o, and n represent the scale factor, offset vector and shifting parameters that depend on the camera.

$$d_v = (v * s + o) \gg n \tag{2}$$

Although, several gaps can appear in the warped vector due to occlusion and disocclusion of objects considering the synthesized view [11]. Therefore, the second and third steps, called interpolation and hole filling, presented in Fig. 4, are responsible for filling these holes in the warped vector [11]. For this, it is necessary a sequence of rules aiming for the correct filling of the gaps [11] mainly based on the distance between two neighboring samples in the warped vector. Interpolation is carried out in non-occluded ranges that are not disoccluded. Considering each x'_{FP} position to be filled, which is located between the boundaries x'_s and x'_e, $\hat{x}$ will be the position of the up-sampled view derived from the interval boundaries [11], which is given by (3). The boundaries x'_s and x'_e represent fractional positions created by warping process from the first neighboring sample on each side of x'_{FP}. Also, x_s represents the first position of the next neighboring input view samples.

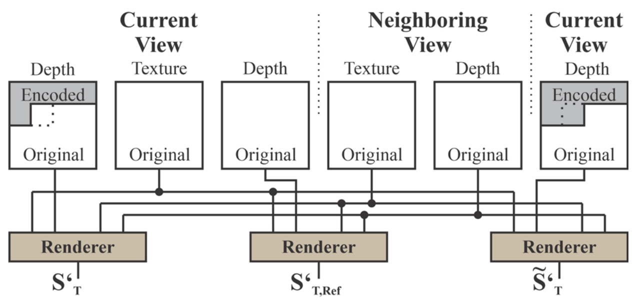


Fig. 3 Information required for rendering each SVDC input

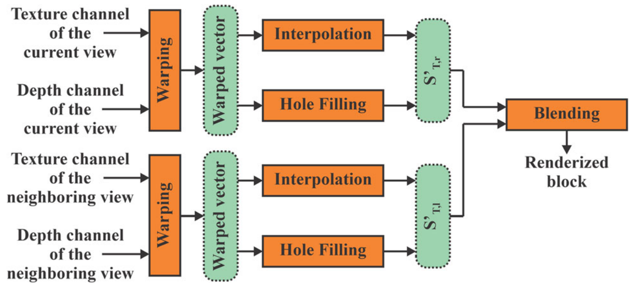

Texture channel of the current view
Depth channel of the current view
Texture channel of the neighboring view
Depth channel of the neighboring view
Warping
Warped vector
Interpolation
Hole Filling
S'T,r
S'T,l
Blending
Renderized block

Fig. 4 Block diagram of renderer steps

The up-sampled view is interpolated using the FIR-filters from HEVC motion-compensation [11].

$$\hat{x} = 4 \cdot \left(\frac{x'_{FP} - x'_s}{x'_e - x'_s} + x_s \right) \tag{3}$$

If the width of the warped interval $x'_e - x'_s$ is greater than two times the width of the sampling distance, it means that this point has a disocclusion, and then, hole filling is performed instead of interpolation. In this case, the gap will be filled with the first sample after the gap $(s_{T,l}(x_e))$, which belongs to the right interval boundary, or with the last sample before the gap $(s_{T,l}(x_s))$, which belongs to the left interval boundary, in the case that the sample to be filled is close to this boundary [11].

If the interval boundaries are reversed $(x'_e < x'_s)$, it means that this interval has an occlusion. In this case, a search is carried out to find the biggest values (i.e., most distant samples) of the depth map between the interval boundaries. The found values are discarded, and a new interval is carried out with the remaining values [11].

The last step is the blending, and it is responsible for blending the left synthesized view $s'_{T,l}$ and the right synthesized view $s'_{T,r}$ aiming to generate a final rendered view s'_T that will be used in SVDC, as shown in Fig. 4. To blend the two views, it is necessary to apply some rules to decide the use of each view according to information of occlusion and disocclusions of the two captured views [11].

2.3 Sum of absolute transformed differences

The SATD was used by default in some internal decisions of the 3D-HEVC, as in the comparison between the 35 Intra modes from the 3D-HEVC Intra prediction tool [12], and also for comparing the fractional candidates in the Fractional Motion Estimation of Inter prediction tool [9].

The SATD definition is presented in (4), where HT is the 2D Hadamard Transform of W [13] and (x, y) indicates the sample position of the block B being evaluated. Equation (5) shows the matrix product to perform the Hadamard Transform over the input W, where H is the Hadamard Matrix and H^T is the transposed of H on the (x, y) position. Equation (6) shows an example of the 4×4 Hadamard Matrix (H). In Eq. (7), W represents the difference between the sample of the original block (S_D) and the predicted block $(\tilde{S_D})$ on the (x, y) position.

$$SATD = \sum_{(x,y) \in B} |HT(x, y)| \tag{4}$$

$$HT = H \cdot W \cdot H^T \tag{5}$$

$$H = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{6}$$

$$W = S_D(x, y) - \tilde{S_D}(x, y). \tag{7}$$

2.4 Sum of squared errors

The SSE was adopted only in a few steps of the 3D-HEVC standard, mainly in the Inter and Intra prediction RD-cost calculations [9, 12]. The RD-cost locally estimates the rate-distortion cost of each option to encode each block, avoiding the full Rate-Distortion Optimization (RDO) which has a prohibitive processing cost. The RD-cost is used to define which tools tend to better predict each block. This calculation aims to locally find the tools that can reach the best tradeoff between the number of bits used to represent a block and the image quality. Then, only a few encoding options are applied to the full RDO. In the context of this work, the SSE is used to measure this impact in the image quality [11].

The SSE definition is presented in (8), where S_D and $\tilde{S_D}$ are the original and reconstructed (encoded and decoded) depth map sample, respectively [12], while (x, y) indicates the sample position of the block B being evaluated. So, each difference between S_D and $\tilde{S_D}$ were raised to the square power, and further accumulated with the difference of the other samples from the block. As can be seen, by (8), the SSE does not require rendering operations, while the most complex operation performed is the multiplication for the squared power.

$$SSE = \sum_{(x,y) \in B} \left[S_D(x, y) - \tilde{S_D}(x, y) \right]^2. \tag{8}$$

2.5 Sum of absolute differences

The SAD was adopted as the similarity criterion in several tools of the 3D-HEVC encoder, as the Integer Motion Estimation [9], and also commonly adopted in hardware implementations of other encoding tools due to its low-complexity [14, 21].

The SAD value can be obtained by the definition presented in (9). It is very similar to the SSE, where S_D and $\tilde{S_D}$ are the original and reconstructed (encoded and decoded) depth map sample, respectively [12], while (x, y) indicates the sample position of the block B being evaluated. Although, different than in SSE where the difference between S_D and $\tilde{S_D}$ were raised to square power, here is only obtained the absolute value of this difference, to be further accumulated with the differences of other samples.

$$SAD = \sum_{(x,y)\in B} \left| S_D(x,y) - \tilde{S_D}(x,y) \right| \quad (9)$$

As one can notice, the SAD also does not require any rendering operation, neither the transform of the processed block. Furthermore, the SAD is the simplest between the presented similarity criteria. It can be performed only with sums and subtractions, since the SAD only requires obtaining the difference between every sample from the original to the reconstructed depth map sample, and later accumulating the absolute from all differences obtained.

3 Similarity criterion efficiency evaluation on DIS

Once changing the similarity criterion can affect the visual quality, experiments were conducted with the 3D-HTM reference software [9] to evaluate quality losses.

To evaluate the impact of each criterion in the quality losses, several experiments were conducted using the 3D-HTM reference software and using the same methodology presented in Section II.A. In these new experiments, the similarity criterion of the DIS tool was modified from SVDC to each one of the three evaluated similarity criteria. For each criterion, several experiments were performed, considering the Random-Access (RA) temporal configuration [18]. Based on the experimental results, the BD-rate metric was computed. This metric measures the increase in the required bit rate to obtain the same image quality in PSNR, when comparing a new method with an anchor method (SVDC in this case) [18, 22].

The Results presented in Table 1 show the BD-Rate increase of three metrics regarding the SVDC metric. For each metric, the Video Total represents the impact of only the encoded texture channels considering the total bit rate of the video (texture + depth map), while the Synthesized Total represents the impact in the synthesized views (through the DIBR process) with the encoded texture and depth map regarding synthesizing the views with the original data.

As can be seen, the SATD metric has resulted in the highest impact in the encoding efficiency, being a 0.203% increase of BD-Rate for video total views and a 0.240% increase of the BD-Rate for synthesized total views. This occur because the four DIS modes were developed to better encode smooth and homogeneous areas of depth maps and these areas do not present relevant information at the frequency domain. Besides, the use of a distortion metric at the frequency domain can generate some wrong decisions since the DIS modes produce predictions with different behaviors at the frequency domain: IP_H and IP_V allow differences between lines and columns, respectively, where the SD_H and SD_V modes use only one sample to represent the complete predicted block.

The SSE metric has resulted in the smaller BD-Rate increase in Video Total, with only a 0.182% increase in the BD-Rate considering the average of all videos. Although, when considering the impact in the synthesized views, the SSE metric has presented better results only than the SATD. However, if considering the lower resolution videos, the SSE have resulted in a smaller impact in the encoding efficiency than SATD and SAD in *Balloons* and *Kendo* sequences, while the *Newspaper* have resulted in a smaller impact than SATD, with a negligible decrease in the encoding efficiency than the SAD criterion. Therefore, in the average of the lower resolution videos, the SSE has presented better results than both SATD and SAD metrics, since that the SSE has an average of video total and

Table 1 BD-rate evaluation of similarity criteria in the 3D-HEVC DIS tool

Sequence	SATD		SSE		SAD	
	Video total (%)	Synthesized total (%)	Video total (%)	Synthesized total (%)	Video total (%)	Synthesized total (%)
Balloons	0.183	0.268	0.161	0.227	0.163	0.231
Kendo	0.234	0.350	0.207	0.306	0.221	0.339
Newspaper	0.235	0.303	0.215	0.265	0.218	0.263
GT_Fly	0.156	0.223	0.152	0.242	0.143	0.196
Poznan_Hall2	0.444	0.359	0.378	0.300	0.399	0.293
Poznan_Street	0.111	0.150	0.104	0.149	0.102	0.142
Undo_Dancer	0.085	0.032	0.071	0.005	0.071	0.024
Shark	0.176	0.233	0.170	0.228	0.169	0.208
1024 × 768	0.217	0.307	0.194	0.266	0.201	0.278
1920 × 1088	0.194	0.199	0.175	0.185	0.177	0.173
Average	0.203	0.240	0.182	0.215	0.186	0.212

synthesized total of 0.25% smaller than SAD, and 10.4% smaller than SATD.

Finally, the SAD has resulted in a BD-Rate higher than the SSE metric for the Video Total, while when considering the Synthesized Total, the SAD has presented the smaller BD-Rate increase, with only 0.212% increase of the BD-Rate, second only to the SSE in the lower resolution sequences.

4 Designed architecture

Four independent architectures were designed to evaluate the different similarity criteria in the DIS tool. These four architectures have the same structure to obtain the predicted block of the four DIS modes, defining the best mode and generating the residue by only changing the similarity criterion unit used to evaluate each DIS mode.

Therefore, the next section presents the high-level view of the DIS Architecture, adopted by the four developed architectures, while in the sequence the small units that compute the similarity value from each criterion were presented and explained.

4.1 DIS architecture

The DIS architecture is capable of processing all CU sizes supported by the 3D-HEVC DIS (64 × 64, 32 × 32, 16 × 16, and 8 × 8), processing all these sizes sequentially. When processing the smaller CU sizes, only a part of the designed hardware is used, while other parts are disabled with clock-gating logic to reduce power dissipation.

Figure 5 shows the generic architecture for the processing of an 8 × 8 CU, as an example. The DIS architecture receives as input the values of neighboring samples of the block to be predicted (A_0 to A_{N-1}, B_0 to B_{N-1}, $A_{N/2}$, and $B_{N/2}$) to generate the four different predictions. These samples are sent to the DIS Modes Unit, which will predict the current block, line by line, according to the current DIS mode, resulting in the predicted vector. All four DIS modes are processed sequentially. Each mode will generate a predicted block that goes to Similarity Criterion Unit in Fig. 5.

The DIS Modes Unit was designed to generate one line of the predicted CU at each clock cycle, independently of the CU size. This means that each prediction mode of an 8 × 8 CU, for example, will require eight cycles to be completed. Since there are four DIS prediction modes, then the 8 × 8 CU is completely predicted in 32 cycles (eight cycles per prediction mode). At the other end, a 64 × 64 CU is predicted in 256 cycles, since 64 cycles are required to generate each one of the four available predictions.

Then, the Similarity Criterion Unit is responsible for evaluating the predicted vector, generating a distortion value based on the original block, which indicates how similar the predicted block is when compared with the original block. Each Similarity Criterion was developed using several pipeline stages, aiming that only one operation was performed in each clock cycle. Specific details from each Similarity Criterion Unit will be better explained in the next section.

The obtained distortion values are then stored at the registers presented in Fig. 5. When the four distortions of the four DIS predictions are ready, they are compared through the Comparator in Fig. 5. This comparison defines the best DIS mode that will be used to encode this CU.

To avoid the use of internal memories, the designed architecture does not store the four prediction modes for all supported CU sizes. If the predictions were stored, the architecture would require internal storage of four predictions for each CU size. Considering that each sample in a

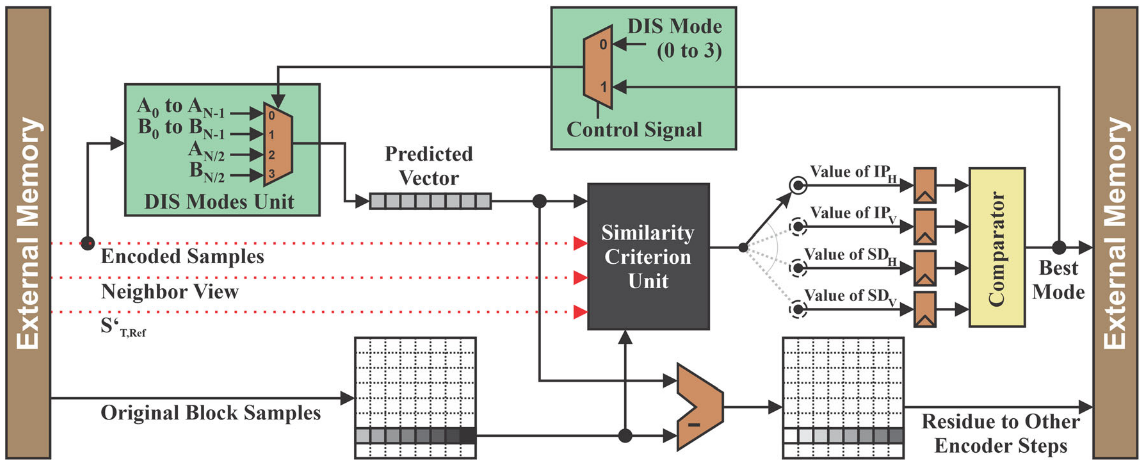


Fig. 5 DIS architecture considering an 8 × 8 block size

CU has eight bits, then a total of 131,072 bits would be required to store the intermediate results.

Since the predictions were not stored, then, when the best DIS mode is defined, the predicted block of the best mode must be generated again. This is done through the DIS Modes Unit when the best mode generated by the Comparator is uses as input, as presented in Fig. 5. Then a control signal defines when the DIS Modes Unit is generating the predictions to define which is the best one, and when this unit is generating the best prediction again. This process requires a variable number of cycles, depending on the CU size, which can vary from eight cycles for 8 × 8 CUs to 64 cycles for 64 × 64 CUs.

As soon as each line of the predicted block was generated, it is subtracted from the original CU to generate the residue, as presented in Fig. 5. This residue is used in the other encoder steps to define if the DIS mode will be used or not to encode this CU (since many other encoder tools are available as previously discussed).

4.2 SVDC unit

The SVDC Unit has three different modules, which are the Render Model, the SVDC Tree, and the Accumulator, as can be seen in Fig. 6. Two instances of these three modules are required. The SVDC input information includes up to five frames stored in the external memory, as presented by the red dotted lines in Fig. 5, which inputs are exclusive needed for the SVDC unit. The frames stored in this external memory includes two frames containing the original data of the two channels from the neighboring view, and one frame containing the original texture data of the current view.

Also, it includes one frame containing the view $S'_{T,Ref}$ rendered using only the original data, and one last frame containing both previously encoded depth blocks and original depth blocks to be encoded. The external memory size is a function of the target resolution and, for an HD 1080p resolution, it is necessary 10.4 Mbytes.

The first Render Model module generates the $\widetilde{S'_T}$, thus it considers the already encoded data for the depth map of the current view, including the current CU encoded with the DIS mode being evaluated, as discussed in Section II.

The second Render Model module generates the S'_T, thus it considers the already encoded depth samples of the current view, but with the original data for the current CU, also as discussed in Sect. 2. Since $S'_{T,Ref}$ represents the synthesized view rendered only with the original data, which does not vary along with the frame processing, the developed SVDC Unit assumes that this input was already rendered before the frame processing.

The two instances of the Render Model are identical and the difference is only the input data. The two instances were necessary to increase the throughput and avoid the use of internal memory that would be needed to store each output for further evaluation. The main modules of the Render Model architecture are presented in Fig. 7.

The Render Model processes the data from the current and neighboring views in parallel. On the Render Model, each input pass by the warping process to generate the warped vector, as presented in Fig. 7. This warping process all input values in parallel in one clock cycle. It applies (2) to each input value in order to obtain the position of the samples in the new view being synthesized. This module used 64 multipliers, 64 adders and 64 controlled shifts to implement (2).

Therefore, the warped vector from the two views was generated in the first pipeline stage of the Render Model to be processed in the next clock cycle. After the warping process, the warped vectors of the two views will be processed in the second pipeline stage of the Renderer Model, in order to compose the synthesized view.

For that, firstly, the gaps in the warped vector will be filled using interpolation or hole filling, depending on the distance between neighboring samples in the Warped Vector. In the case of the interpolation, (3) is applied to

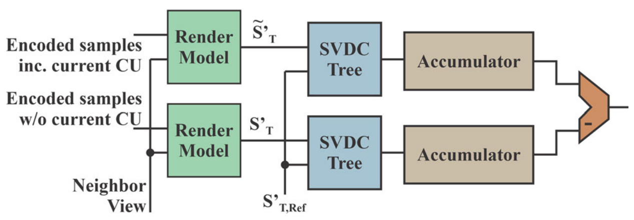


Fig. 6 SVDC similarity unit, composed of render model, SVDC tree, and accumulators

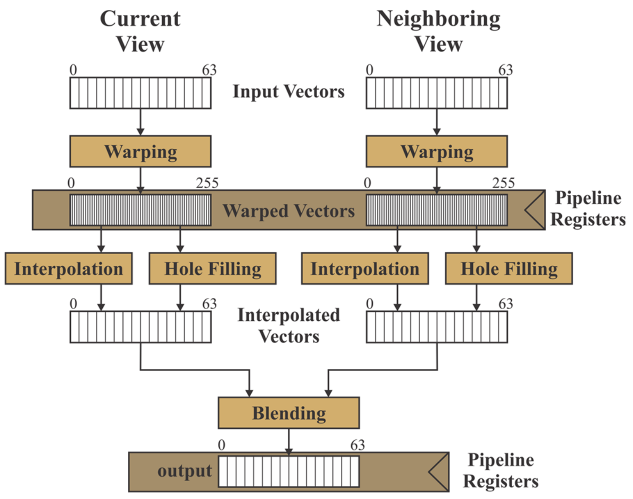


Fig. 7 Processing flow of the render model

each position of the interpolated vector to define which fractional samples of the warped vector should be used to fill the interpolated vector to compose the new synthesized view.

The interpolation module includes 64 dividers, 128 subtractors, 64 adders and 64 shift lefts of two bits (multiplication by four). The fractional samples are composed of a set of filters that interpolates three new samples at every two neighboring samples [21]. In the case of hole filling, the hole is filled with the neighboring sample of the hole according to the location of the synthesized view to the current view.

Finally, the result of the two processed views will be blended in a single vector, based on the hole filling rules, generating a virtual intermediated view wished. This virtual intermediated view is then processed by the Metric Units, Tree and Accumulator Units presented in Fig. 8. Since the intermediary view was generated line by line, it was also processed by these Units line by line. Firstly, the intermediary view passes by the Metric Unit. The SVDC metric adopts the Squared Difference for the Metric Unit, which was represented in Fig. 9(a). In Fig. 9(a), the A and B inputs are the render output and $S'_{T,Ref}$, respectively. The inputs are subtracted and the result is multiplied itself to obtain the value raised in the squared power, generating the difference of the position being processed raised to the squared power.

After that, the output of all Metric Units was accumulated in only one value. For that, the Tree unit presented in Fig. 8 has several pipeline stages for sum the values two by two. In the first pipeline stage, this module has 32 adders. In the rest of the Tree, 31 adders are spread among the other five pipeline stages. Since the processing is performed line by line, an accumulator is necessary to group the results of all lines.

The Accumulator Unit presented in Fig. 8 is used to store the value from the line being processed, to be further accumulated with the result of the next line. For that, the Accumulator has one adder to accumulate the similarity value, a multiplexer to reset the accumulated value and a register to store the value. Therefore, after the Tree computes the metric value from all lines, the Accumulator

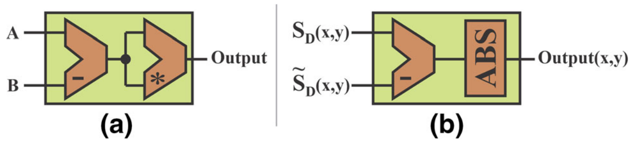


Fig. 9 Details of the metric units: **a** squared difference; **b** absolute difference

obtains the final value of the whole block. Finally, as can be seen in Fig. 6, the values of the Accumulators output were, subtracted to generate the final distortion value according to the SVDC metric.

Figure 10 shows a clock cycles diagram of the SVDC architecture, considering the processing of the distortion value for a DIS mode to an N × N CU. One can observe that the Interpolator starts its processing two cycles before the predicted block being computed by the DIS unit. This was necessary for obtaining the interpolated samples in time for the Blending process since the interpolation filters have three pipeline stages. Moreover, the interpolation and Tree require more cycles than other parts since only these modules have pipeline stages to be filled. Furthermore, one clock cycle was necessary for the Control and for Store the obtained SVDC result. So, as can be seen, the SVDC unit requires N + 13 clock cycles to obtain the SVDC value to each DIS mode to a block with N × N size. Therefore, considering that it has to process four DIS modes per block and also the residue block from the best mode, it will require a total of 9,560 clock cycles to process all blocks from one CTU.

4.3 SATD unit

The developed hardware for the SATD only requires the block predicted with the DIS mode being processed, and also the original block being encoded, processing these blocks line by line, and the SATD were processed over subblocks of 8 × 8 size. The main processing of the SATD was presented in Fig. 11(a). As can be seen, firstly, a line from the original and from the predicted block was subtracted. Then, this line of differences is divided into slices with the size of eight values, to be processed the Hadamard

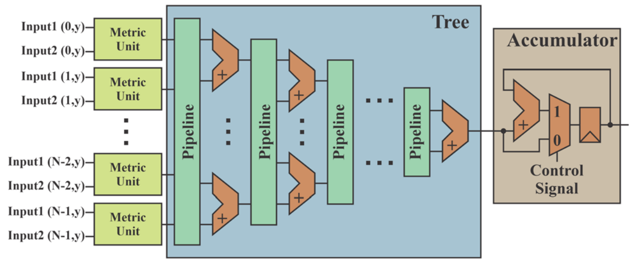


Fig. 8 Tree and accumulator units

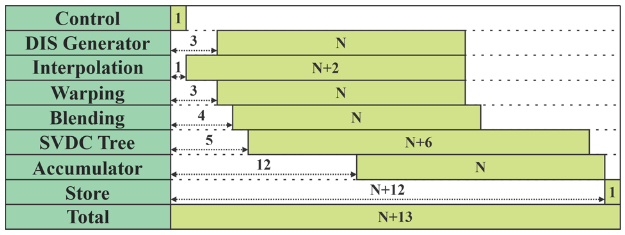


Fig. 10 Clock cycles diagram of the SVDC unit to each DIS mode

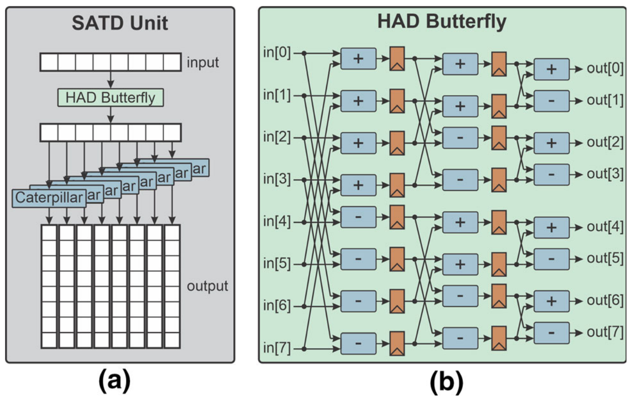


Fig. 11 SATD architectures: **a** SATD unit; **b** HAD butterfly

transform. For that, the line of differences is passed first to the HAD Butterfly, presented in Fig. 11(b).

After, each position of the transformed line processed goes to a Caterpillar Unit, presented in Fig. 12. This Caterpillar is a modification of the HAD Butterfly, necessary since the SATD Unit processes the input blocks line by line, so in each clock cycle, it computes the transform of one input value into a vector with eight positions. Since the SATD processing was performed for 8 × 8 blocks, it requires eight Caterpillar Units working in parallel, resulting in an 8 × 8 block that contains the Hadamard transform of the differences from the input blocks. Finally, all samples from the Caterpillar output block were accumulated in only one value in order to implement (4). For that, the absolute value from each value of the output block is obtained, and all these absolute values are then summed using a Tree and Accumulator Units similar to the one used in the SVDC architecture and presented in Fig. 8.

Figure 13 shows a clock cycles diagram focusing only at the SATD architecture, considering the processing a DIS mode to an N × N CU. As can be seen, after obtaining the difference between the two input blocks, each pipeline from HAD Butterfly requires eight clock cycles since the

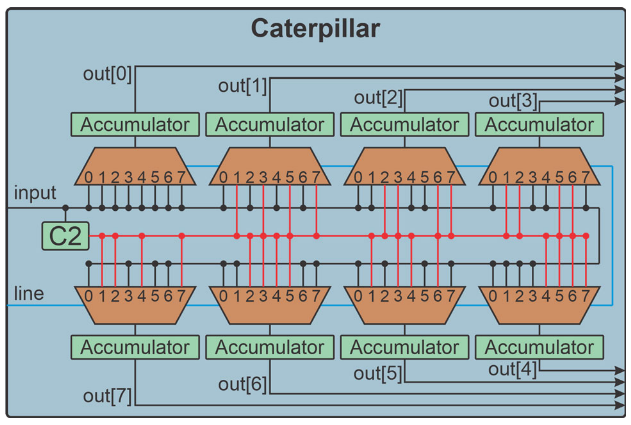


Fig. 12 Caterpillar unit

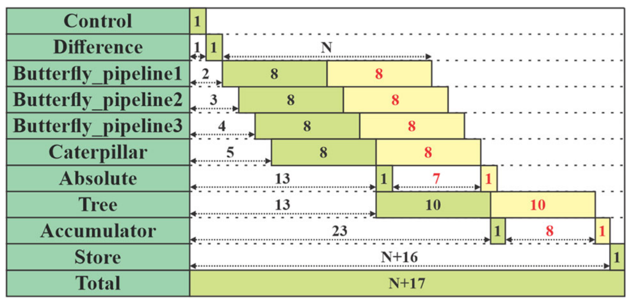


Fig. 13 Clock cycles diagram of the SATD unit to one DIS mode

Hadamard transform was applied over subblocks of 8 × 8 size. After, the Caterpillar processes the eight transformed lines of the subblock in eight cycles. Finally, the absolute values from the transformed block were accumulated by the Tree and the Accumulator in 11 clock cycles, since the SATD Tree have several pipeline stages.

Since that the transformed was performed over 8 × 8 subblocks, the processing of some units was repeated at every eight cycles, as represented in red in Fig. 13. So, for blocks with a height of 8, 16, 32, and 64 sizes, the processing from some units was repeated one, two, four and eight consecutive times, respectively. Finally, after the repeating process, the obtained value for the DIS mode being processed was stored for further comparison with other DIS modes, so totaling N + 17 clock cycles to obtain the SATD value of each DIS mode. Considering all the four DIS modes, the residue block, and all blocks from one CTU, the SATD Unit requires a total of 12,110 clock cycles to process each CTU.

4.4 SSE unit

The SSE Unit was very simple compared with the SVDC and with the SATD since its processing is only performed by the Metric Unit, Tree, and Accumulator Unit, being all these presented in Fig. 8. Firstly, for the Metric Unit, the SSE adopts the Squared Difference, as presented in Fig. 9(a). The inputs A and B of this Squared Difference were the sample from the predicted block and from the original block, respectively. As in the SVDC, the output of the SSE unit goes to the tree structure, which accumulates all values in several cycles to generate the SSE value of the line being processed. Finally, the Accumulator was used to sum the SSE values from all lines, obtaining the final SSE value from the mode being processed.

Figure 14 shows a clock cycles diagram focusing on the SSE architecture, considering the processing of a DIS mode to an N × N CU. As can be seen, after obtaining the difference between the samples of each line from the predicted and original blocks, all samples were accumulated by the Tree in N + 6 clock cycles since the Tree has six

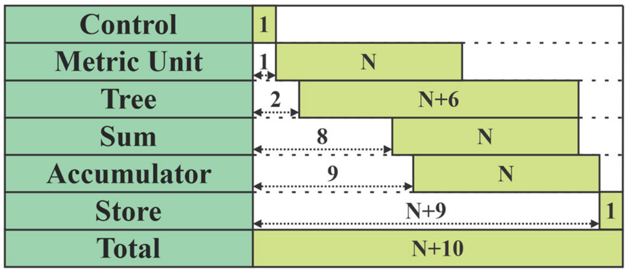


Fig. 14 Clock cycles diagram of the SSE and SAD units to each DIS mode

pipeline stages. Finally, as can be seen in Fig. 14, the SSE architecture requires N + 10 cycles to obtain the SSE value to each DIS mode to an N × N block size. So, when considering the processing of all four DIS modes and the residue for all blocks from a CTU, the SSE architecture requires 9,135 clock cycles to process the DIS mode to each CTU.

4.5 SAD unit

The SAD Unit is very similar to the SSE. It was also performed only by the Metric Unit, Tree, and Accumulator Units. The only difference for the SSE Unit is that the Metric Unit from the SAD adopts the Absolute Difference unit, presented in Fig. 9(b). In the Absolute Difference unit, it was first obtained the difference between the sample of the original block (S_D) and the reconstructed block ($\tilde{S}_D$). Then, the absolute value is obtained from this difference. After, in the same way as the SSE Unit, the SAD Unit accumulate all differences obtained by using the Tree and the Accumulator in several clock cycles since these units have several pipeline stages, so obtaining the SAD value of the DIS mode being processed.

The SAD Unit has the same pipeline distribution as the SSE Unit. So, the clock cycles diagram from Fig. 14 is also used to represent the clock cycles distribution from the SAD Unit, where the SAD Unit requires N + 10 clock cycles to obtain the SAD value of the DIS mode being evaluated, and also totalizing 9,135 clock cycles to process the DIS mode to each CTU when considering the four DIS modes, the residue and all blocks from a CTU.

5 Results and comparisons

The hardware from each metric was developed in Verilog HDL and synthesized for ASIC, considering the TSMC 40 nm standard cells library, and using the Cadence Encounter RTL Compiler tool [23]. The RTL Compiler synthesis effort was defined as *high*. The RTL Compiler performs the ASIC logic synthesis and generates the estimated results of circuit cell area, power consumption, and length of the circuit critical path, which is used to define the maximum operation frequency. With this operation frequency it is possible to define the throughput according to the number of cycles required by each unit. The obtained results were presented in Table 2, which presents the cell area, the maximum frequency reached, and the frequency needed to process one view of 1080p@30fps, considering each of the four architectures. Table 2 also presents the power consumption when processing a different number of views of 1080p@30fps, and the average BD-rate increase of each similarity criterion. The power was also obtained with Cadence RTL Compiler tool, adopting the tool default transition probability (20% of probability from each input pin changes its value).

Based on the results presented in Table 2, Fig. 15 presents the comparison of the obtained results in the form of radar chart. For each variable compared, the values were normalized according to the percentage of the higher value from each variable. It should be noted that the power presented are the values from when processing two views of 1080p@30fps, while the throughput was presented considering the inverse of the maximum resolution supported by each metric. Therefore, from Fig. 15, the best metric is the one that results in the diamond with a smaller area.

As can be seen in Table 2 and Fig. 15, the SVDC implementation requires at least 13 times more hardware resources then the other criteria since it requires performing rendering operations, where the warping has several multiplications, and the interpolation unit has several dividers and interpolation filters to generate some fractional samples to be used by the blending operation, which highly increase the complexity of the SVDC metric. In Table 2 and Fig. 15 it can also be seen that the SVDC resulted in a power consumption between 938% and 4,946% higher than the other criteria when processing two views of 1080p@30fps. Still, due to the higher complexity of the SVDC, which has several multiplicators and dividers, it reaches the throughput needed to processes only two views of 1080p@30fps, a resolution between 77.78% and 81.82% smaller than the other criteria.

However, the SVDC reaches higher complexity efficiency, being used by default in the 3D-HTM and so resulting in no increase considering the BD-rate metric. The other criteria are less accurate, generating BD-Rate losses. Therefore, the SVDC is the best criterion only for applications that requires the best compression efficiency of the 3D-HEVC standard since the synthesis results are worse than the other evaluated criteria considering any other evaluated variable. Although, it should be noted that the SVDC results in a maximum throughput of at least 77% smaller than the other similarity criteria.

Table 2 Synthesis results

	SVDC	SATD	SSE	SAD
Area (k gates)	11,133	862.7	337.8	311.5
Maximum frequency (MHz)	292.53	3,115.3	3,067.5	3,236.2
Frequency (MHz) 1080p@30fps 1 view	146.26	341.96	292.54	292.54
Maximum throughput 1080p@30fps	2 views	9 views	10 views	11 views
Power (mW) 2 views	4174.00	402.12	97.92	82.72
Power (mW) 3 views	N.R	487.62	133.34	121.87
Power (mW) 9 views	N.R	976.57	387.24	365.65
BD-rate increase (%)	–[a]	0.240	0.215	0.212

N.R.—Not Reached due to the required frequency

[a]The SVDC is the default criterion on DIS of 3D-HTM

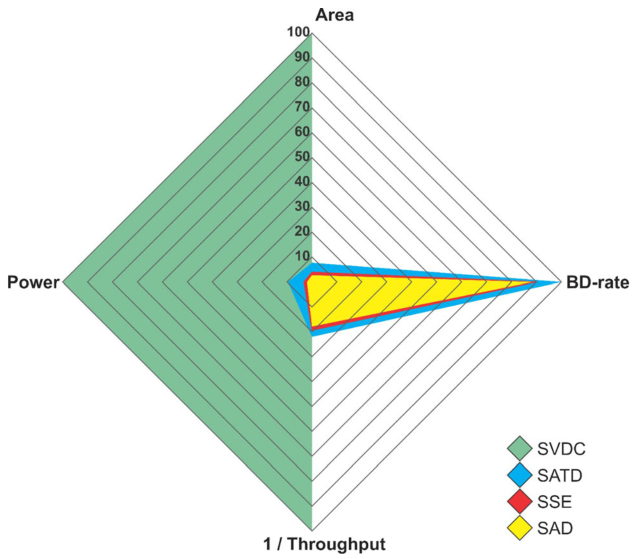


Fig. 15 Multivariable comparison results of different similarity criteria

When disregarding the results of the SVDC, the other similarity criteria have resulted in the same template for any variable, where the SATD has resulted in the worst results in all variables. The SATD has an area between 2.55 and 2.77 times bigger, a maximum throughput between 10% and 18.2% smaller, and a power consumption for processing two views of HD 1080p videos between 75.65% and 79.43% bigger than other similarity criteria. Lastly, the SATD still has a BD-Rate increase between 1.116 and 1.132 times bigger than the other criteria.

As presented in Table 1, the SSE have presented a smaller impact in the encoding efficiency then the SATD and SAD for the lower resolution sequences, while considering all sequences, the SSE have results very similar to the SAD results. Although, when compared to SAD, it has presented the worst hardware results among all variables, since the area requirements and power consumption was 8.4% and 10.3% higher than the results of SAD, while the maximum throughput supported by SSE was also smaller than the SAD, since the SSE requires a multiplier operator to obtain the difference of two samples raised to the squared power. Finally, the SAD has resulted in the best results among all variables, although it was the similarity criterion with the smaller complexity. In fact, it was expected that SAD presents the worst compression efficiency when compared with the SATD [24] and SSE due to the capacity of these two modes to intensify the errors of each predicted block. Although it can be seen in Table 2 and Fig. 15 that the SATD and SSE are worse than the SAD in all evaluated variables, being that the SAD has proved to be the best similarity criterion for the DIS tool in applications that does not require the complete compression efficiency of the 3D-HEVC standard. When compared with the SVDC, the SAD showed a few better results. As 97.2% smaller area than SVDC, a maximum throughput 5.5 times bigger, and a 97.9% lesser power consumption for the processing of two views of HD 1080p videos. However, SAD has a 0.212% BD-Rate increase comparing with SVDC.

6 Conclusion

This article presented some alternatives to reduce the 3D-HEVC DIS tool complexity, allowing hardware friendly implementations. The main idea explored here was the substitution of the complex SVDC similarity criterion used in DIS tool for simpler and more hardware friendly criteria. Three similarity criteria were investigated: SATD, SSE, and SAD. These criteria were evaluated in terms of encoding efficiency and hardware impacts in comparison with the SVDC.

The use of the alternative criterion showed small coding efficiency losses between 0.240% and 0.212% in BD-rate. By the other hand, the hardware related gains were expressive and with important differences among the evaluated criteria. The frequency gains ranged from 10 to 11 times, where area gains ranged from 13 to 35 times. The power gains are even more expressive, ranging from 10 to 50 times when one view is processed.

Among the alternative criteria, the best results were found with SAD in both, coding efficiency and hardware results. The coding efficiency loss was of only 0.212% when compares with SVDC. The hardware results showed that SAD can use 35 times less area than SVDC, reaching a maximum operation frequency 11 times higher than SVDC and with a power 50 times lower than SVDC when processing one video view.

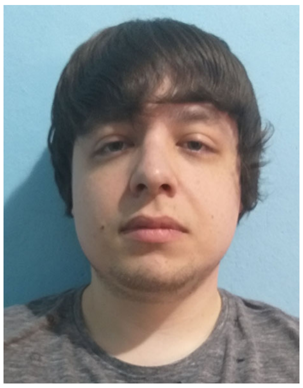
With the results presented in this article one can conclude that the use of a simpler similarity criterion is an important alternative to be used in DIS tool, mainly if an efficient hardware design is required.

Acknowledgements This investigation was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. It was also financed in part by the Fundação de Amparo à pesquisa do Estado do Rio Grande do Sul – Brasil (FAPERGS), and by the Conselho Nacional de Desenvolvimento Científico e Tecnológico – Brasil (CNPq).

References

1. COVID-19's impact on streaming video. *Bitmovin*, Aprril 2021. http://bitmovin.com
2. Jama, A. M., et al. (2015). Video transmission over wireless networks review and recent advances. *International Journal of Computer Applications Technology and Research, 4*, 444–448.
3. International Telecommunication Union. Recommendation ITU-T H.265: High Efficiency Video Coding. April 2015.
4. Tech, G., et al. (2016). Overview of the Multiview and 3D extensions of high efficiency video coding. *IEEE Transactions on Circuits and Systems for Video Technology, 26*(1), 35–49. https://doi.org/10.1109/TCSVT.2015.2477935
5. Versatile Video Coding | JVET, April 2021. https://jvet.hhi.fraunhofer.de/
6. Merkle, P., Smolic, A., Muller, K., & Wiegand, T. (2007). Multi-view video plus depth representation and coding. In *2007 IEEE international conference on image processing*, San Antonio, TX, pp. 201–204. https://doi.org/10.1109/ICIP.2007.4378926
7. Fehn, C. (2004). Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV. In *Proceedings of SPIE 5291, Stereoscopic Displays and Virtual Reality Systems XI*, May 2004.
8. Sullivan, G., et al. (2012). Overview of the High Efficiency Video Coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology, 22*(12), 1649–1668. https://doi.org/10.1109/TCSVT.2012.2221191
9. Three-Dimensional High Efficiency Video Coding (3D-HEVC) Reference Software Retrieved, April 2021. https://hevc.hhi.fraunhofer.de/3dhevc
10. Tech, G., Schwarz, H., Muller, K., & Wiegand, T. 3D video coding using the synthesized view distortion change. In *Picture Coding Symposium*, pp. 25–28. https://doi.org/10.1109/PCS.2012.6213277
11. Chen, Y., Tech, G., Wegner, K., & Yea, S. (2015). Test Model 11 of 3DHEVC and MV-HEVC. *JCT3V-K1003*. Geneva.
12. Afonso, V., et al. (2017). Low-power and high-throughput hardware design for the 3D-HEVC depth intra skip. In *International symposium on circuits and systems*, May 2017. https://doi.org/10.1109/ISCAS.2017.8050463
13. Liu, X., Zhang, Y., Hu, S., Kwong, S., Kuo, C.-J., & Peng, Q. (2015). Subjective and objective video quality assessment of 3D synthesized views with texture/depth compression distortion. *IEEE Transactions on Image Processing, 24*(12), 4847–4861. https://doi.org/10.1109/TIP.2015.2469140
14. Borges, V., et al. (2020). A hardware design for 3D-HEVC depth intra skip with synthesized view distortion change. In *Symposium on integrated circuits and systems design*, August 2020. https://doi.org/10.1109/SBCCI50935.2020.9189925
15. Dou, H., et al. (2015). View synthesis optimization based on texture smoothness for 3D-HEVC. In *International conference on acoustics, speech and signal processing*, April 2015. https://doi.org/10.1109/ICASSP.2015.7178209
16. Ahmad, W., et al. (2015). Complexity and implementation analysis of synthesized view distortion estimation architecture in 3D High Efficiency Video Coding. In *International conference on 3D imaging*, December 2015. https://doi.org/10.1109/IC3D.2015.7391818
17. Afonso, V., Saldanha, M., Conceição, R., Perleberg, M., Porto, M., Zatt, B., Suzin, A., & Agostini, L. (2019). Real-time architectures for 3D video coding. In *VLSI architectures for future video coding. England* (Ch. 6, pp. 191–226). The Institution of Engineering and Technology (IET). https://doi.org/10.1049/PBCS053E_ch6
18. Müller, K., & Vetro, A. (2014). *Common test conditions of 3DV core experiments*. San José
19. Sze, V., Budagavi, M., & Sullivan, G. (2014). *High efficiency video coding (HEVC)*. Springer. https://doi.org/10.1007/978-3-319-06895-4
20. Soares, L. B., Diniz, C. M., da Costa, E. A. C., & Bampi, S. (2016). A novel pruned-based algorithm for energy-efficient SATD operation in the HEVC coding. In *2016 29th Symposium on Integrated Circuits and Systems Design (SBCCI)*, Belo Horizonte, pp. 1–6. https://doi.org/10.1109/SBCCI.2016.7724049
21. Afonso, V., et al. (2016). Hardware implementation for the HEVC fractional motion estimation targeting real-time and low-energy. *Journal of Integrated Circuits and Systems, 11*(2), 102–120.
22. Bjontegaard, G. (2008). Improvements of the BD-PSNR model. *VCEG-AI11*. July 2008.
23. Encounter RTL Compiler, April 2021. http://www.cadence.com
24. Seidel, I., Monteiro, M., Bonotto, B., Agostini, L. V., & Güntzel, J. L. (2019). Energy-Efficient Hadamard-based SATD hardware

architectures through calculation reuse. *IEEE Transactions on Circuits and Systems I: Regular Papers, 66*(6), 2102–2115. https://doi.org/10.1109/TCSI.2019.2900004



Vinicius A. Borges received a certificate in Eletronics from the Sul-rio-grandense Federal Institute (IFSul), Pelotas, RS, Brazil, in 2016. He is currently pursuing the B.S. degree in computer science at Federal University of Pelotas (UFPel). He is a member of the Video Technology Research Group (ViTech) at the UFPel. His research interests include 2D/3D video-coding algorithms, and video VLSI design for video coding.

Murilo R. Perleberg received the B.S. degree in computer engineering from the Federal University of Pelotas (UFPel), RS, Brazil, in 2018, and the M.S. degree in computer science from the UFPel in 2020. He is currently a pursuing the Ph.D. degree in computer science at UFPel. He is a member with the Video Technology Research Group (ViTech) at the UFPel. His research interests include video VLSI design for video coding.

Vladimir Afonso received the B.S. degree in industrial automation from the Sul-rio-grandense Federal Institute (IFSul), Pelotas, RS, Brazil, in 2008, the M.S. degree in computer science from the Federal University of Pelotas (UFPel), RS, Brazil, in 2013, and the Ph.D. degree on microelectronics from the Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brazil, in 2019. He is a Professor since 2009 at the IFSul and a researcher with the Video Technology Research Group (ViTech) at the UFPel. His research interests include 2D/3D video-coding algorithms, and FPGA-based and VLSI designs for video coding.

Marcelo S. Porto received the M.S. and Ph.D. degrees in computer science from Federal University of Rio Grande do Sul (UFRGS), Brazil, in 2008 and 2012, respectively. He is currently a Professor with the Federal University of Pelotas (UFPel), Brazil, and a member of the Video Technology Research Group (ViTech) and the Group of Architectures and Integrated Circuits (GACI). He is currently the Coordinator of the Postgraduate Program in Computing (PPGC) of UFPel. He also has been holding the status of CNPq (National Council for Scientific and Technological Development) Productivity Research Fellow, since 2016. His research interests include video coding, motion estimation algorithms, point cloud compression, coding complexity reduction, and energy-efficient VLSI design for video coding.

Luciano V. Agostini is a Brazilian Distinguished Researcher through a CNPq PQ-1D grant. He received the M.S. and Ph.D. degrees in Computer Science from Federal University of Rio Grande do Sul, Porto Alegre, Brazil, in 2002 and 2007 respectively. He is a professor since 2002 at Federal University of Pelotas (UFPel), Brazil, where he leads the Video Technology Research Group (ViTech). He is advisor at the UFPel Master and Doctorate in Computer Science courses. He was the Executive Vice President for Research and Graduate Studies of UFPel from 2013 to 2017. He has more than 300 published papers in journals and conference proceedings. His research interests include 2D and 3D video coding, algorithmic optimization, arithmetic circuits, and digital systems. Currently he is an Associated Editor of IEEE TCSVT and IEEE OJCAS. Dr. Agostini is a Senior Member of IEEE and ACM. He is a member of the IEEE CAS, CS, and SPS societies. At the IEEE CAS, he is a member of the MSA and VSPC Technical Committees. He is also member of SBC and SBMicro Brazilian societies.