A Novel Depth Motion Vector Coding Exploiting Spatial and Inter-component Clustering Tendency

Shampa Shahriyar*, Manzur Murshed[†], Mortuza Ali[†] and Manoranjan Paul[‡]

*Faculty of Information Technology, Monash University, Australia.

[†]Faculty of Science and Technology, Federation University Australia, Australia.

[‡]School of Computing and Mathematics, Charles Sturt University, Australia.

Abstract—Motion vectors of depth-maps in multiview and free-viewpoint videos exhibit strong spatial as well as inter-component clustering tendency. This paper presents a novel coding technique that first compresses the multidimensional bitmaps of macroblock mode and then encodes only the non-zero components of motion vectors. The bitmaps are partitioned into disjoint cuboids using binary tree based decomposition so that the 0's and 1's are either highly polarized or further sub-partitioning is unlikely to achieve any compression. Each cuboid is entropy-coded as a unit using binary arithmetic coding. This technique is capable of exploiting the spatial and inter-component correlations efficiently without the restriction of scanning the bitmap in any specific linear order as needed by run-length coding. As encoding of non-zero component values no longer requires denoting the zero value, further compression efficiency is achieved. Experimental results on standard multiview test video sequences have comprehensively demonstrated the superiority of the proposed technique, achieving overall coding gain against the state-of-the-art in the range [22%, 54%] and on average 38%.

I. INTRODUCTION

Depth information is becoming increasingly important for video coding applications, such as, auto-stereoscopic display, where only few views are captured and virtual views are synthesized from them using a depth image based rendering (DIBR) technique. Spatial and temporal characteristics of depth-maps, representing the distance information of 3D points in the scene from the camera plane, are significantly different from the texture images. Depth-maps are characterized by strong spatial correlation where discontinuities mostly occur at object boundaries. While the state-of-the-art depth coding technique introduces specialized intra-coding mode to exploit the high spatial correlation in depth maps, it relies on traditional block based motion compensated predictive coding to exploits the temporal correlations [1].

In this paper, we propose a motion vector coding scheme that efficiently exploits both spatial and inter-component correlations of the motion vectors. More specifically, the proposed scheme focuses on exploiting the followings. Firstly, due to high temporal correlation, motion vectors are mostly zeros

for depth-maps. Secondly, the non-zero motion vectors do not arise in isolation, instead they exhibit strong clustering tendency. Finally, there exists a strong correlation between the components of the motion vectors, i.e., given that one of the components of a motion vector is non-zero, with high probability, the other component will be non-zero.

The current video coding standards, such as H.264 [2], HEVC [3], and 3D-HEVC [1], encode the components of a motion vector separately and thus unable to exploit the high correlation between the components of the vectors. They encode the components of the motion vectors using *differential pulse-code modulation* (DPCM) to exploit correlations among the motion vectors of the neighbouring macroblocks. However, the components of the motion vectors are encoded on a symbol-by-symbol basis, which requires allocating at least one bit per component. In contrast, the scheme proposed in this paper, encodes the motion vector information at frame level to efficiently exploit the spatial and inter-component correlations. It arranges the motion vectors of all the macroblocks of a frame into a 3D volume by considering that the horizontal (x -axis) and vertical (y -axis) displacement components belong to adjacent planes (p -axis). Information about the positions of macroblocks of identical coding mode are then extracted into different bitmaps. These bitmaps are successively encoded followed by the encoding of the non-zero motion components only.

The proposed scheme adaptively partitions the bitmaps into cuboids of various dimensions. The partitioning induced by a binary tree aims at isolating large homogenous cuboids that can be encoded as a whole using optimal bits. At each stage of coding, information from the previous stages are taken into account. Our experimental results demonstrate that this multi-stage coding technique is more efficient than symbol-by-symbol coding of the motion vectors.

The organization of the rest of the paper is as follows. In Section II, the state-of-the-art depth motion vector coding technique is reviewed and a simpler depth macroblock classification is introduced without any loss of generality. The proposed cuboid depth motion vector coding technique is presented in Section III. Experimental results demonstrating the efficacy of the proposed scheme are presented in Section IV. Finally, we conclude the paper in Section V.

*Corresponding author email: ssha204@student.monash.edu. This research is partly supported by the Australian Research Council Discovery Project DP130103670.

II. STATE-OF-THE-ART IN DEPTH MOTION VECTOR CODING

3D-HEVC standard incorporates block-based depth-map coding using a mix of intra-frame and inter-frame modes, similar to the prevailing predictive coding technique used for texture frames. Traditional motion-compensated inter-frame mode encodes a macroblock using a motion vector to identify the best-predicted block in an already-encoded reference frame and the entropy-coded quantized block-residual. Motion compensated depth macroblocks, however, are also able to reuse the motion vectors of the corresponding texture macroblocks to achieve further coding efficiency. There are several well-known methods [4], [5], [6] that take advantage of shared motion between texture and depth data for coding efficiency. Due to change of lighting condition, illumination compensation are not similar for texture and depth. Recently, we have reported that motion vectors in texture and depth domains are not sufficiently correlated across domain. Consequently, at constant bitrate coding, reuse of texture motion vectors introduces significant distortions in the decoded depth-maps, leading to perceptually inferior view synthesis in free-viewpoint videos [7]. Therefore, we consider a simpler classification of depth macroblocks with only four modes, representing as many as 44 sub-modes used in the HEVC standard. The classification hierarchy with corresponding Huffman-coded mode code words are shown in Figure 1.

Fig. 1. Typical coding modes of depth macroblocks

Macroblocks are encoded in the raster-scan order, addressed from the leftmost macroblock of the top row to the right-most macroblock of the bottom row in sequence. Intra macroblocks are encoded raw or using residual coding with spatial prediction. As Skip macroblocks have no motion vector or residual to encode, these are encoded using run-length coding without any explicit code word for these blocks. After encoding a non-Skip macroblock, the *macroblock address increment* to the next non-Skip macroblock in the current row or the end of the row (denoting the run-length of the Skip macroblocks) is encoded using Huffman codes for runs of length 1-33 and an Escape code [8]. Any run longer than 33 is encoded by using the Escape code for each sub-run of 33.

For both Inter-Zero and Inter-MC macroblocks, quantized transform coefficients of the residual are encoded using an entropy coder such as *context-adaptive variable-length coding* (CAVLC) or *context-adaptive binary arithmetic coding* (CABAC). Motion vectors of Inter-MC macroblocks are encoded independently at component-level using predictive coding to exploit spatial correlation of neighbouring motion

vectors. The median of motion vectors used in the “already-encoded” adjacent macroblocks is considered as the prediction. The difference value of each of two components is then encoded using signed Exp-Golomb codes of order 0 [9].

III. CUBOID DEPTH MOTION VECTOR CODING TECHNIQUE

Motion vectors in depth domain exhibit stronger spatial and inter-component clustering tendency (see Figure 2). Hence jointly exploiting the spatial and inter-component correlations is crucial to achieve high compression efficiency for depth motion vectors. The run-length based entropy coders such as LZW [10] have been reported inefficient for encoding multidimensional data as their coding performance is very sensitive to the scanning order, which is difficult to optimize in higher dimensions. We are, therefore, motivated to develop a novel *cuboid coding* framework that compresses first the 3D bitmap of the zero and non-zero components jointly and then the values of the non-zero components individually in a predefined scanning order. The clustering correlations are efficiently exploited during bitmap compression without any explicit run-length. As encoding of non-zero component values no longer requires denoting the zero value, further compression efficiency is achieved. The proposed coder/decoder (codec) is now elaborated in the following subsections:

A. The Encoder

Not all macroblocks of a P (predicted) or B (bi-predicted) frame are inter-coded when the rate-distortion optimization prefers other available coding modes. A frame of motion vectors is then split into two planes representing the x - and y -components. The encoder compresses both the planes together in the following two stages.

Stage 1: Joint-coding of 3D Zero/Non-zero Bitmap Let W and H be the width and height of the frame in macroblocks. Let $B_{Z/NZ}$ be the 3D bitmap of size $2 \times W \times H$ where existence of non-zero components of the motion vectors is denoted by 1’s in their corresponding positions. The spatial and inter-component correlations in $B_{Z/NZ}$ are jointly exploited by partitioning the bitmap using a binary tree such that the portion of the bitmap covered by a leaf cuboid is either highly correlated or not correlated at all.

The tree is constructed by a simple divide-and-conquer algorithm with “greedy” optimization heuristic. Let the current cuboid encompass N elements of $B_{Z/NZ}$ of that z are 0’s. The “divide” step first checks whether it contains only one kind of elements, i.e., $z = 0$ or $z = N$. In that case, the cuboid is classified as a Type I (all 0’s) or Type II (all 1’s) leaf node that requires no further information to encode. Otherwise, the cuboid is split recursively into two sub-cuboids along p (plane), x -, or y -axis such that 0-1 elements are “maximally polarized” by the split. This heuristic is expected to divide $B_{Z/NZ}$ into sub-cuboids that are either highly polarized (i.e., almost all 0’s or almost all 1’s with near minimum entropy) or further sub-partitioning is unlikely to achieve any compression (i.e., almost random with near maximum entropy).

A cuboid of size $P \times X \times Y$ can be split in $P - 1$, $X - 1$, and $Y - 1$ ways along p -, x -, and y -axis, respectively. Let split s form two sub-cuboids with $N_{s,1}$ and $N_{s,2}$ elements $N_{s,1} + N_{s,2} = N$ of that $z_{s,1}$ and $z_{s,2}$ are 0's, respectively $z_{s,1} + z_{s,2} = z$. Polarization of 0-1 elements by this split can be measured as

$$\rho_s = \underbrace{\left| \frac{z_{s,1}}{N_{s,1}} - \frac{z_{s,2}}{N_{s,2}} \right|}_{\text{Polarization of 0's}} = \underbrace{\left| \frac{N_{s,1} - z_{s,1}}{N_{s,1}} - \frac{N_{s,2} - z_{s,2}}{N_{s,2}} \right|}_{\text{Polarization of 1's}}$$

and we are interested to find the maximally-polarized split

$$s^* = \underset{\forall s}{\operatorname{argmax}} \rho_s$$

The “conquer” step of the algorithm ensures that sub-partitioning of the leaf nodes are infeasible to achieve any further compression by discarding the splitting decision made in the “divide” step, if the zero-order entropy of the subtrees is greater than the zero-order entropy of the current cuboid. In that case, the cuboid is classified as a Type III leaf node, which is compressed by denoting the number of 0's or 1's in $\lceil \log_2 N \rceil$ bits and then the *binary arithmetic coded* bit-stream of the cuboid with $p_0 = z/N$ in $\lceil N(-p_0 \log_2 p_0 - (1 - p_0) \log_2 (1 - p_0)) \rceil + 1$ bits.

A split (non-leaf) node is classified into Type P, Type X, and Type Y depending on whether the optimal split is found along p -, x -, or y -axis, respectively. The split position is encoded using fixed-length codes. If the optimal split is found at position $l \in \{1 \dots L\}$, where L is the range of the cuboid in the optimal dimension, l is encoded in $\lceil \log_2 L \rceil$ bits.

TABLE I
HUFFMAN CODES OF BITMAP TREE NODES

Node	Type	Code word	Node	Type	Code word
Leaf	I	001	Split	X	11
	II	0001		Y	01
	III	10		P	0000

The type of each node of the tree is identified using Huffman codes shown in Table I that were generated by analysing the probability of each type in a large set of test depth-map sequences. The entire tree is encoded in the *pre-order depth-first traversal* sequence where the root of the tree is encoded first, then recursively the left subtree, and finally, recursively the right subtree.

Figure 2 presents the partitioning of $B_{Z/NZ}$ corresponding to Frame 2 (using Frame 1 as reference) of View 2 of the *Newspaper* free-viewpoint test sequence by the proposed divide-and-conquer algorithm. The clearly demarked 39 2D and 61 3D partitions in the joint 3D-view, every one of them encompassing both the planes, demonstrate not only the high spatial and inter-component correlations of depth motion vectors; but also the efficiency of the proposed algorithm in achieving partitioning of $2 \times 128 \times 96 = 24576$ elements with only 100 cuboids.

Stage 2: Independent Coding of Non-zero Components

Once the bitmap is encoded, individual non-zero motion components are encoded in a predefined order of the bitmap

Fig. 2. Binary tree decomposition of the 3D zero/non-zero bitmap of View 2 of the *Newspaper* free-viewpoint test sequence (Frame 2 using Frame 1 as reference): 2D-views of x -plane (left-top) and y -plane (left-bottom); and joint 3D-view of both planes (right). Here color dots indicate nonzero motion components.

traversal. This stage is similar to the state-of-the-art as presented in Section II; but significantly more efficient (using on average half a bit less per motion component) as it does not require encoding the component of Inter-MC blocks having zero difference from the median predictor. Absence of zero values is exploited to achieve this additional compression efficiency by allowing all positive or negative differences, whichever is majority in a given plane, using the codes of its preceding difference value and then encoding an extra bit per plane to identify the sign of majority. For example, if the difference of non-zero components of a plane, from their predictions, are mostly positive (negative), the +ve (-ve) sign is denoted with 0 (1) and each positive (negative) value d is encoded with codes to $d - 1$ ($d + 1$).

The 3D bitmap $B_{Z/NZ}$ inherently encodes the positions of the Inter-MC macroblocks. Similar bitmaps ($B_{S/NS}$ and $B_{I/NI}$) can also be used to efficiently encode the positions of other types of macroblocks (Skip and Intra), obviating the necessity of encoding mode code words for individual macroblocks.

As the tree is encoded in the pre-order depth-first traversal sequence, the reconstruction of the 3D bitmap $B_{Z/NZ}$ from the encoded tree is straightforward. Then all the non-zero motion components are decoded in the same traversal order of the bitmap used at the encoder. Complexity of our divide-and-conquer partitioning is directly proportional to the number of leaf nodes, which can be constrained by setting a lower limit on the cuboid volume and/or considering a sub-partitioning space.

IV. EXPERIMENTAL RESULTS

We evaluated the performance of the proposed cuboid coding scheme against the state-of-the-art motion vector coding scheme in Section II on the motion vectors of depth-maps of the main view of eight standard multiview test video sequences *Balloons* (I), *Newspaper* (II), *Lovebird1* (III), *Kendo* (IV), *GT_Fly* (V), *Undo_Dancer* (VI), *Poznanhall* (VII), and *Pozanstreet* (VIII). Frame resolution of the first and last four sequences is 1024×768 and 1920×1088 pixels, respectively.

The first 49 depth-maps of each sequence were used with GOP (group of pictures) 8, Intra-period 24, and macroblocks of 8×8 pixels without any sub-blocking. Motion compensation was carried out on the decoded reference frames using the diamond motion search with search-length 30 and Lagrange multiplier based rate-distortion cost optimization. For simplicity, the macroblocks with residual above a threshold were considered intra-coded without using an explicit intra coder. Threshold values were adjusted for each test sequence to achieve typical fraction of the Intra macroblocks between 5-15%. Macroblocks with (0,0) motion vectors were classified with Skip or Inter-Zero modes depending on whether the residual was zero or not. The remaining macroblocks were deemed Inter-MC. Residual coding performance is not considered here as it is done using state-of-the-art technique and proposed technique only performs in motion vector and prediction mode coding.

Compression performances of both the existing and proposed depth motion vector and prediction mode coding schemes, are reported in Table II. The proposed scheme has reduced the expected number of bits for macroblock mode information by in the range [0.31,1.48] and on average 0.96 bits per macroblock due to higher dimensional coding of three mode bitmaps. By taking advantage of encoding only the non-zero components, it has also reduced the expected number of bits for motion vectors by in the range [0.49,1.02] and on average 0.85 bits per vector. Coding gain in this two areas, against the state-of-the-art, are in the range of [16%,75%] and [19%,39%], respectively. Proposed coding technique performs better in high-resolution video due to maximal exploitation of clustering tendency.

TABLE II
COMPRESSION OF MACROBLOCK MODE INFORMATION
(BITS/MACROBLOCK) AND MOTION VECTORS (BITS/INTER-MC) BY THE
STATE-OF-THE-ART AND THE PROPOSED CODING SCHEMES

Seq	State-of-the-art		Proposed		Coding Gain (%)		
	Mode	MV	Mode	MV	Mode	MV	All
I	1.97	3.40	1.39	2.46	29	28	29
II	2.11	2.64	1.00	2.15	53	19	38
III	2.22	4.05	1.31	3.24	41	20	29
IV	1.88	3.66	1.57	2.64	16	28	22
V	2.29	3.20	0.94	2.30	59	28	43
VI	1.92	2.35	0.70	1.43	64	39	54
VII	1.98	2.82	0.50	2.06	75	27	51
VIII	1.78	3.07	1.08	2.14	39	30	35

Finally, to gain some insight on the underlying strength of the proposed technique in achieving such significant coding gains, we present the dimensional attributes of the leaf nodes generated by the proposed binary tree based decomposition in Table III. For all three bitmaps $B_{Z/NZ}$, $B_{S/NS}$, and $B_{I/NI}$, the proportion of multi-dimensional, 2D and 3D (if possible), leaf nodes is at least 60%. Leaf nodes collectively cover the whole bitmap without any overlap and each one is encoded as a unit to efficiently exploit the spatial and inter-component (if possible) correlations of 0's and 1's without relying on any scanning order as needed by run-length coding.

TABLE III
PROPORTION OF MULTI-DIMENSIONAL PARTITIONING OF MODE BITMAPS
BY THE PROPOSED BINARY TREE BASED DECOMPOSITION

Seq	Zero/Non-zero			Skip/Non-skip		Intra/Non-intra	
	1D	2D	3D	1D	2D	1D	2D
I	30%	61%	9%	27%	73%	0%	100%
II	24%	62%	14%	13%	87%	0%	100%
III	29%	60%	11%	25%	75%	16%	84%
IV	27%	58%	15%	22%	78%	11%	89%
V	29%	66%	5%	27%	73%	0%	100%
VI	39%	59%	2%	37%	63%	0%	100%
VII	40%	57%	3%	25%	75%	4%	96%
VIII	29%	64%	6%	27%	73%	8%	92%

V. CONCLUSION

This paper has presented an efficient motion vector coding technique that can exploit the high spatial and inter-component correlations of depth motion vectors. A simple but highly effective heuristic based binary tree decomposition is used to adaptively partition the macroblock position bitmaps of various modes into highly polarized or random cuboids. These coding blocks are then encoded with binary arithmetic coding in the pre-order depth-first traversal sequence of the tree followed by non-zero motion components that also take the advantage of the absence of zero values. The proposed technique has overwhelmingly outperformed the state-of-the-art motion vector and prediction mode coding technique used in the video coding standards with comparative coding gain as high as 54%. The proposed cuboid coding technique is elegant and it may have applications in other domains that will be investigated in future.

REFERENCES

- [1] K. Muller, H. Schwarz, D. Marpe, C. Bartnik, S. Bosse, H. Brust, T. Hinz, H. Lakshman, P. Merkle, F. Rhee, G. Tech, M. Winken, and T. Wiegand, "3D high-efficiency video coding for multi-view video and depth data," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3366–3378, 2013.
- [2] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, 2003.
- [3] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, 2012.
- [4] "H.264-based depth map sequence coding using motion information of corresponding texture video," in *Advances in Image and Video Technology*, ser. Lecture Notes in Computer Science, 2006, vol. 4319.
- [5] S. Grewatsch and E. Miiller, "Sharing of motion vectors in 3D video coding," in *Int. Conf. on Image Processing (ICIP)*, vol. 5, 2004, pp. 3271–3274.
- [6] W. Su, D. Rusanovskyy, M. Hannuksela, and H. Li, "Depth-based motion vector prediction in 3D video coding," in *Picture Coding Symposium (PCS)*, May 2012, pp. 37–40.
- [7] S. Shahriyar, M. Murshed, M. Ali, and M. Paul, "Efficient coding of depth map by exploiting temporal correlation," in *Int. Conf. on Digital Image Computing: Techniques and Applications (DICTA)*, Nov 2014, pp. 1–8.
- [8] J. F. Arnold, M. R. Frater, and M. R. Pickering, *Digital Television: Technology and Standards*. John Wiley and Sons, 2007.
- [9] S. W. Golomb, "Run-length encodings," *IEEE Trans. Inf. Theory*, vol. 12, pp. 399–401, 1966.
- [10] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Theory*, vol. 23, no. 3, pp. 337–343, 1977.