

Received November 20, 2017, accepted January 7, 2018, date of publication February 8, 2018, date of current version September 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2803733

Novel Pixel Recovery Method Based on Motion Vector Disparity and Compensation Difference

TING-LAN LIN¹, (Member, IEEE), XUTAO WEI², XUBO WEI², TZU-HAO SU²,
AND YU-LIANG CHIANG²

¹Department of Electronic Engineering, National Taipei University of Technology, Taipei 10608, Taiwan

²Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan 32023, Taiwan

Corresponding author: Ting-Lan Lin (tinglan@ntut.edu.tw)

This work was supported by the Ministry of Science and Technology, Taiwan, under Grant MOST 105-2221-E-033-020, Grant 105-2622-E-033-009-CC3, and Grant 106-2221-E-033-010.

ABSTRACT As compressed videos are transmitted in the communication networks, video packet loss inevitably occurs. This problem can be solved by error concealment method. We used the motion vector of the available neighboring blocks to estimate the lost motion vector for the lost block. These estimates propagate to predict all other missing motion vectors. We further improved the work by using the idea of the motion vector disparities between neighboring available blocks to modify the motion vector weightings. Furthermore, the differences between the compensated pixels and the decoded pixels in the neighboring blocks are computed for another weighting for improvement. These two novelties are combined as a final indicator to prediction weightings. By comparison against the state-of-the-art method, the four proposed algorithms increase the average peak signal-to-noise ratio (PSNR) by up to 1.86, 1.93, 1.94, and 2.04 dB on average, showing the gradual improvement of our design systems. For other video quality measurements, the average gains of the proposed work against the state-of-the-art work can be up to 0.0575 in structural similarity index metric (SSIM), -0.0278 in video quality metric (VQM) (the lower the better), -0.0008 in motion-based video integrity evaluation (MOVIE) (the lower the better), and 2.77 in subjective evaluation. The proposed work performs slightly worse than a pixel-based state-of-the-art method in PSNR and SSIM but performs better in VQM and MOVIE (both correlate better with human perception) and subjective experiments, with much lower computational complexity.

INDEX TERMS H.264, motion vector, video error concealment, motion vector disparities, motion-compensated differences.

I. INTRODUCTION

Video compression algorithms, such as MPEG-1 [1], MPEG-2 [2], H.263 [3] and H.264 [4], are important in the digital era. Among which, H.264 is widely used in video communication due to its good compression rate [5]. H.264 increases coding-efficiency with new modules, such as multiple reference frame. It also has flexible block sizes from 16×16 , to 8×4 , to 4×4 and so on in variety of ways, giving more flexibilities for application. Video packet loss may occur during communication. Therefore it is important to develop a video packet recovery algorithm for the end user to watch the video with decent experience during the worse condition of the communication channel. Redundant slices and flexible macroblock ordering are error-resilient mechanisms for H.264 encoder. For the decoder part, error concealment technique is popular to recover the

lost packets. Error concealment methods had been discussed in many literatures. Boundary matching algorithm to estimate the lost motion vectors is developed in [6]. In [7], edges in the missing blocks are estimated. Optical flow method is utilized in [8] and [9] to recover the lost pixels. The work in [10] minimized a joint spatial-temporal cost function for the lost motion vector. The work in [11] predicted the missing motion vectors from the available motion vectors. Statistical methods such as B-spline modeling, autoregressive models and Bayesian methods are developed for the error concealment algorithms in [12]–[14], respectively. Recursive algorithm is designed for the motion vector estimation in [15]. Multiple error concealment methods are combined in a single method by switching and blending which are proposed in [16]–[18]. Motion vector extrapolation was first proposed in [19], which is extended in [20] for

better performances. In [21], a hybrid method is proposed by re-weighting the extrapolated motion vectors with available neighboring information. The work in [22] used coding residuals in surrounding blocks as the reliability index to assist the recovery of the lost motion vectors. In [23], the coding partition information in the previous frame were used to group the motion vectors of possible objects in the lost blocks. Liu *et al.* [24] presented a sequential pixel recovery method with adaptive linear predictor by using Bayesian information and spatially neighboring available pixels. In [25], an error concealment method based on adaptive dual dictionary learning and regularization was proposed, with the uses of the sparsity of the observed space and the latent space. Akbari *et al.* [26] used wavelet decomposition and sparse optimization framework to recover the lost pixels. However in the proposed work, to save the computational complexity, we develop a *low-complexity motion vector recovery* method that is based on neither other coding parameters as in [22] and [23], nor pixel domain recovery with complicated optimization algorithm as in [24]–[26]; our efficient structure only uses the neighboring motion vectors to estimate the lost motion vector.

To be specific, the proposed work is categorized as the motion-compensation-based error concealment method, since the work aims to estimate the motion vectors for the lost 4×4 blocks, and the estimated motion vectors are used to take the pixels in the reference frame. Therefore we analyze recent works falling into this category of motion-compensation-based error concealment method, such as [21]–[23]. The works in [21]–[23] require the additional information from previous frame, additional computation for processing, and additional memory for storage. To elaborate, for additional information, the works [21]–[23] need motion vectors of previous frame to perform motion vector extrapolation to assist the estimation of the current lost motion vectors; the works in [22] and [23] further require additional residual information and partition information respectively. For required additional computation for information from previous frame, the works [21]–[23] need to perform additional motion vector extrapolation, and evaluate the overlapped area to estimate the extrapolated motion vectors. For required additional memory for information from previous frame, the works [21]–[23] need to store additional $M \times N$ (number of blocks in a frame) extrapolated motion vectors or extrapolated partition information. Therefore, the works [21]–[23] consume too much resources.

In the proposed work, we aim to develop a *low-cost* motion-compensation-based error concealment method. The proposed work only requires motion vectors in surrounding blocks of the lost block in current frame (which are also used in [21]–[23]). What is more, for the proposed work, there are no requirements for *additional computation for information from previous frame* and *additional memory for information from previous frame*. **Therefore the proposed work is much more efficient and low-cost than the state-of-the-art works [21]–[23].**

As for the compared method in the experimental section, for fairness, *we mainly compare with the work in [21] (and not [22] and [23] since they are too much “expensive” as discussed) since it uses the closest level of computational resources (information, complexity and storage) as our design.* (In fact, the requirements for computational resources (information, complexity and storage) for [21] are still much higher than the proposed work.)

In our prior work [27], we improved the work in [21]. In the proposed work, we aim to further improve our prior work in [27] to have even better performance than the work in [21].

The novelties of our work are as follows:

- 1. We modify the scheme in the state-of-the-art method [21] that uses the farther available neighboring motion vectors without reusing the estimates, into the scheme that uses the closer available neighboring motion vectors with estimation propagation. This idea and initial results had been presented in our prior work in a conference paper in [27]. The improvement of the proposed method against the state-of-the-art method [21] is up to 1.86 dB in PSNR (Peak Signal to Noise Ratio in dB).**
- 2. The estimation process in previous proposed work uses only simple average and direct weighting on the closest available neighboring motion vectors. We further proposed to use the motion vector disparities between neighboring motion vectors in vertical and horizontal directions for differentiated importance of weighting on different neighboring motion vectors. This method further improved the performance gain against the state-of-the-art method [21] by up to 1.93 dB.**
- 3. In addition, we consider the sum of the absolute difference between the motion-compensated pixels and the fully decoded pixels as the indicator of modifying different weightings. This modification outperformed the state-of-the-art method [21] by up to 1.94 dB.**
- 4. Finally, by combining the two previous novel weighting methods and the estimation propagation scheme, the largest gain against the state-of-the-art method [21] can be up to 2.04 dB in PSNR on average. Furthermore, compared with the state-of-the-art method [21], the our improvement gains on average can be by up to 0.0575 in SSIM (Structural Similarity Index Metric), -0.0278 in VQM (Video Quality Metric) (the lower the better), -0.0008 in MOVIE (MOTION-based Video Integrity Evaluation) (the lower the better), and 2.77 in subjective scores (averaged over 50 subjects).**
- 5. The proposed work requires only motion vectors in the surrounding blocks of the lost block in the current frame, which is the most efficient algorithm in computational resources (information, complexity and storage) compared to relevant state-of-the-art works [21]–[23].**

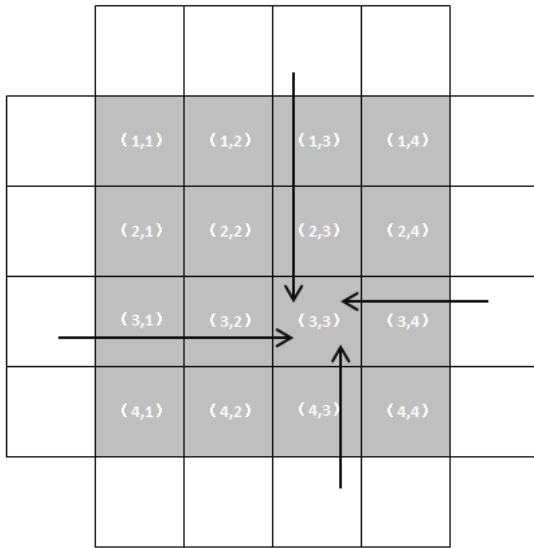


FIGURE 1. Illustration of the method in [21]. Each block has size 4×4, and gray area is the lost 16×16 MB. The white area is available blocks.

6. Compared with a state-of-the-art pixel-based method [24], the proposed method performs better in terms of VQM, MOVIE (both relate better with human perception) and subjective experiments, with much lower computational complexity, even though our PSNR and SSIM are slightly worse.

The paper is organized as follows: section 2 discusses the state-of-the-art method in [21], and our proposed extension using motion vector estimation propagation (which is proposed in our prior work in [27]). Section 3 develops an improved weighting method by using motion vector disparities in the neighboring available blocks. Section 4 uses the motion-compensated differences as new weightings to our algorithm. Section 5 combines the previous two novelties into a final weighting for the motion vector estimation with propagation. Section 6 demonstrates the experimental results. Section 7 is the conclusion of this paper.

II. PROPOSED MOTION VECTOR ESTIMATION PROPAGATION METHOD

In this section, we introduced the method in the state-of-the-art work [21] as the basis of the proposed work. We then develop an algorithm of the proposed motion vector estimation propagation. Parts of the descriptions in this section are taken from our prior work in [27].

In [21], for each lost 4×4 block, the method uses 4 nearest EMV (extrapolated motive vector) and 4 nearest MVs in top, bottom, left, and right available 16×16 MB. As shown in fig. 1 for example, the (3,3) block is a lost block to be estimated with the motion information from top, bottom, left, and right closest available blocks. The distance between the lost block and the available block is important. However, as the distance becomes larger (for this example the distance can be up to 2 blocks away, and 3 for the maximum), the

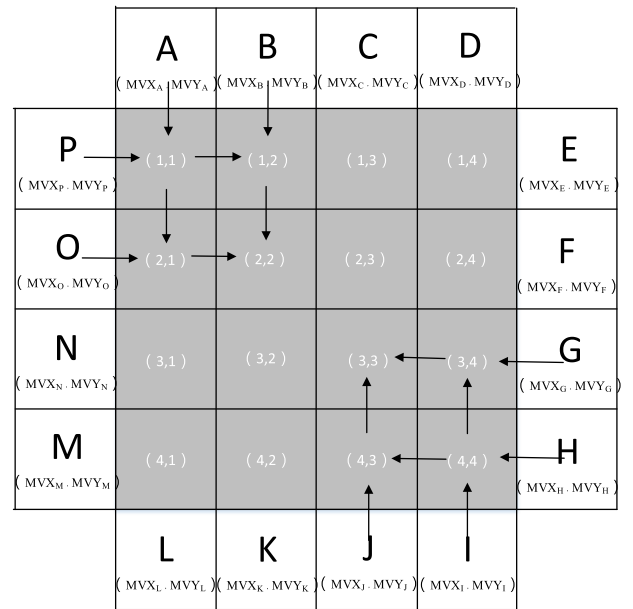


FIGURE 2. The gray area is the lost 16×16 MB. The white area is available blocks. The estimation of the lost motion vector is in the direction of arrow.

correlation becomes smaller, which affects the performance of error concealment. We aim to change the procedure of error concealment to avoid this problem.

In this section, we proposed the novel motion vector estimation propagation method. The illustration is shown in fig. 2. For a lost 16×16 MB (Macroblock) in a frame, it has available 16×16 MB neighbors in top, bottom, left and right locations. The available motion vectors in the neighboring MBs are used for the recovery of the motion vectors in the lost MB. The number of 4×4 blocks in a lost 16×16 MB is 16, whose indexes are shown in fig. 2. To recover the lost motion vectors for each 4×4 block, we start from the most top-left 4×4 block (1,1). It has top and left available neighbor block A and block P, and their motion vectors $MV_A = (MVX_A, MVY_A)$ and $MV_P = (MVX_P, MVY_P)$, thus their average is computed as the estimated motion vectors of block (1,1):

$$MV_{(1,1)} = \left(\frac{1}{2} \times MV_A + \frac{1}{2} \times MV_P\right) \quad (1)$$

For (1,2) block, it has a top neighboring motion vector of block B, and the left estimated motion vectors of block (1,1). Their average is computed as the estimated motion vectors of block (1,2):

$$MV_{(1,2)} = \left(\frac{1}{2} \times MV_B + \frac{1}{2} \times MV_{(1,1)}\right) \quad (2)$$

Note that this is where the “propagation” comes in when later estimation uses previous estimation. Similarly the procedure is performed for block (2,1), which has a left neighboring motion vector of block O, and the top estimated motion vectors of block (1,1). Their average is computed as the estimated

motion vectors of block (2,1):

$$MV_{(2,1)} = \left(\frac{1}{2} \times MV_{(1,1)} + \frac{1}{2} \times MV_O\right) \quad (3)$$

For the lost block (2,2), it has the top estimated motion vectors of block (1,2), and the left estimated motion vectors of block (2,1). The average of them is the estimated motion vectors of block (2,2):

$$MV_{(2,2)} = \left(\frac{1}{2} \times MV_{(1,2)} + \frac{1}{2} \times MV_{(2,1)}\right) \quad (4)$$

As can be seen, these procedures propagate the estimated from the corner block which has the most number of available neighboring motion vector (2 available neighbors for (1,1) in this case), to the more center block which has the least number of available neighboring motion vector (0 available neighbors for (2,2) in this case); this processing order is due to the fact that the estimate with more available neighbors is more reliable to start with. As the estimate motion vectors propagate, the more center block has estimated neighboring motion vectors to use for its recovery of motion vectors. Above procedure starts from the top-left corner 4×4 block (1,1), and the estimate propagation is not suggested to go farther than (2,2) since the estimation error may accumulate with more propagation. Therefore we start over the similar procedure with the top-right 4×4 block (1,4), propagating the estimates (with the similar computations) to the more center block (2,3). Similarly, we start the estimate propagation from the bottom-left corner 4×4 block (4,1) to the more center block (3,2), and from the bottom-right corner 4×4 block (4,4) to the more center block (3,3). The method designed in this section is denoted as **Proposed**.

III. WEIGHTING USING DISPARITY OF THE MOTION VECTORS IN THE NEIGHBORING AVAILABLE BLOCKS

As discussed in the previous section, the prior work in [27] designed a motion vector estimation propagation algorithm that directly uses the neighboring available motion vectors as predictors with simple average. In this section, we aim to use the disparities of neighboring motion vectors in different directions for different weights.

The design idea is as follows. For the horizontal neighboring motion vectors, if their “directions” (to be defined later) vary a lot, it is likely that they are less reliable to prediction, thus their weights should be low in the estimation. Similarly, if the “directions” for the vertical neighboring motion vectors have higher variation, less weights should be put on them for estimation. This is the basic idea for the design. To put into practice, we first define “direction” of a motion vector (MVX, MVY) of a neighboring block by their *arctangent*: $\tan^{-1}\left(\frac{MVY}{MVX}\right)$; this is usually a measurement for the angle of a set of two dimensional vector (MVX, MVY) .

To start the algorithm, we take fig. 2 as illustration. As demonstrated in the previous section, we start from the left-top of the missing macroblock. Specifically, we start from the missing block (1,1) whose immediate vertical neighbor is block A on top and the immediate horizontal neighbor is

block P on the left. Note that in the estimation in the prior work in [27], the estimated motion vector of (1,1) is the direct and simple average of the motion vectors MV_A and MV_P of the neighbors:

$$MV_{(1,1)} = \left(\frac{1}{2} \times MV_A + \frac{1}{2} \times MV_P\right) \quad (5)$$

For the extended method, we used the directional disparity to improve the simple weighting. The horizontal motion vector disparity can be computed by the neighboring block A and block B (they form a horizontal relation), whose motion vectors are (MVX_A, MVY_A) and (MVX_B, MVY_B) , respectively, and their directions are $\tan^{-1}\left(\frac{MVY_A}{MVX_A}\right)$ and $\tan^{-1}\left(\frac{MVY_B}{MVX_B}\right)$, respectively. The motion vector disparity of the block A and block B is computed as their absolute difference in directions, denoted as DD_{AB} , the horizontal directional disparity of block A and B:

$$DD_{AB} = \left| \tan^{-1}\left(\frac{MVY_A}{MVX_A}\right) - \tan^{-1}\left(\frac{MVY_B}{MVX_B}\right) \right| \quad (6)$$

Similarly, the vertical motion vector disparity can be computed by the neighboring block P and block O ((they form a vertical relation)), with the directions being $\tan^{-1}\left(\frac{MVY_P}{MVX_P}\right)$ and $\tan^{-1}\left(\frac{MVY_O}{MVX_O}\right)$. The vertical directional disparity in motion vectors are DD_{PO}

$$DD_{PO} = \left| \tan^{-1}\left(\frac{MVY_P}{MVX_P}\right) - \tan^{-1}\left(\frac{MVY_O}{MVX_O}\right) \right| \quad (7)$$

As mentioned, **large disparity in horizontal direction (DD_{AB} in this case)** means the motion vectors in horizontal direction is inconsistent and thus unreliable, therefore the motion vectors $MV_P = (MVX_P, MVY_P)$ from block P (the horizontal (left) to the (1,1)) should receive less weights in prediction, and the motion vectors $MV_A = (MVX_A, MVY_A)$ **from block A (the vertical (top) to the (1,1)) should receive higher weights**. To sum up, larger DD_{AB} means higher weights for MV_A , so the MV_A is weighted by DD_{AB} as $DD_{AB} \times MV_A$. Similarly, if the disparity in **vertical direction (DD_{PO} in this case) is larger**, it means the motion vectors in vertical direction in this neighborhood are inconsistent and should have lower weights. Therefore the motion vectors from block A (the vertical neighbor of (1,1)) should weight lower and the **block P (the horizontal neighbor of (1,1)) should weight higher** in prediction. Thus larger DD_{PO} means higher weights for MV_P , and the MV_P is weighted by DD_{PO} as $DD_{PO} \times MV_P$.

The two weighting contributions are combined as follows to be the estimate of $\widetilde{MV}_{(1,1)}$:

$$\widetilde{MV}_{(1,1)} = (DD_{AB} \times MV_A + DD_{PO} \times MV_P) \quad (8)$$

After normalization, the estimate $MV_{(1,1)}$ is formulated as

$$MV_{(1,1)} = (w_v_DD_{(1,1)} \times MV_A + w_h_DD_{(1,1)} \times MV_P) \quad (9)$$

where $w_{v_DD(1,1)} = \frac{DD_{AB}}{DD_{AB}+DD_{PO}}$ and $w_{h_DD(1,1)} = \frac{DD_{PO}}{DD_{AB}+DD_{PO}}$. For $MV_{(1,2)}$, the immediate top neighbor MV_B is weighted by DD_{BC} by the same logic. And as in the prior work in [27], the estimate of the immediate left neighbor $MV_{(1,1)}$ is used for the estimation propagation, which is now to be weighted by DD_{PO} by the same idea described previously. Thus the normalized weighted estimate of $MV_{(1,2)}$ is

$$MV_{(1,2)} = (w_{v_DD(1,2)} \times MV_B + w_{h_DD(1,2)} \times MV_{(1,1)}) \quad (10)$$

where $w_{v_DD(1,2)} = \frac{DD_{BC}}{DD_{BC}+DD_{PO}}$ and $w_{h_DD(1,2)} = \frac{DD_{PO}}{DD_{BC}+DD_{PO}}$. For $MV_{(2,1)}$, the immediate top and left neighbors are $MV_{(1,1)}$ and MV_O , so the estimation of $MV_{(2,1)}$ is modified by DD_{AB} and DD_{ON} as:

$$MV_{(2,1)} = (w_{v_DD(2,1)} \times MV_{(1,1)} + w_{h_DD(2,1)} \times MV_O) \quad (11)$$

where $w_{v_DD(2,1)} = \frac{DD_{AB}}{DD_{AB}+DD_{ON}}$ and $w_{h_DD(2,1)} = \frac{DD_{ON}}{DD_{AB}+DD_{ON}}$. Finally, the immediate neighbors of $MV_{(2,2)}$ are $MV_{(1,2)}$ and $MV_{(2,1)}$, therefore the estimation is weighted by DD_{BC} and DD_{ON} :

$$MV_{(2,2)} = \left(\frac{DD_{BC}}{DD_{BC} + DD_{ON}} \times MV_{(1,2)} + \frac{DD_{ON}}{DD_{BC} + DD_{ON}} \times MV_{(2,1)} \right) \quad (12)$$

where $w_{v_DD(2,2)} = \frac{DD_{BC}}{DD_{BC}+DD_{ON}}$ and $w_{h_DD(2,2)} = \frac{DD_{ON}}{DD_{BC}+DD_{ON}}$. Above procedure is performed for the top-left 4 missing blocks, and for the rest of top-right, left-bottom and right bottom blocks, the procedure is similar and can be easily extended. This algorithm is denoted as **Proposed_MVD**, proposed motion-vector disparity method.

IV. WEIGHTING USING DIFFERENCES BETWEEN PIXELS IN THE NEIGHBORING BLOCKS AND THEIR MOTION-COMPENSATED BLOCKS

In the previous section, the motion vector disparity in neighboring blocks is used to modify the prediction weighting in section 2. In this section, we further proposed to improve the weighting by using the idea of pixel differences between the decoded pixels and the motion-compensated pixels of the neighboring available blocks.

As we know, the motion vectors mainly are used in the decoder to recover the decoded pixel. Take block A (of size 4×4) in fig. 2 for example, its motion vector $MV_A = (MVX_A, MVY_A)$ is first used to produce the motion-compensated pixels by copying the 4×4 pixels with the displacement of $MV_A = (MVX_A, MVY_A)$ from its location in the reference frame. The motion-compensated 4×4 pixels are denoted by $MC(A(x, y), MV_A)$, where $A(x, y)$ is the 4×4 location of the block A. This motion-compensated 4×4 pixels are added with some coding information in the bitstream to become the true decoded pixel of 4×4 pixels in block A, denoted as $DP(A(x, y))$. We can infer that if the contents

of 4×4 $MC(A(x, y), MV_A)$ and 4×4 $DP(A(x, y))$ differ a lot, it means the motion vector is not a good predictor. This is the design idea for the algorithm in this section.

To measure the “difference” between 4×4 $MC(A(x, y), MV_A)$ and 4×4 $DP(A(x, y))$ of block A, we use the sum of the absolute difference between them, defined as D_A :

$$D_A = \sum_{x=1}^4 \sum_{y=1}^4 |MC(A(x, y), MV_A) - DP(A(x, y))| \quad (13)$$

Therefore, if the value of D_A is large, the MV_A is not a good predictor, and if D_A is small, the MV_A is a good predictor. This is to be used for the algorithm design. In fig. 2, to start again with the top-left 4 blocks, the block (1,1) is first processed. The two closest available neighbors of it is block A and block P. We compute the differences D_A and D_P using the above procedure, respectively. As discussed, higher D means lower predictability, therefore if D_A is larger, the predictability of MV_A is lower, which means **the predictability of MV_P is larger**; this means the MV_P is weighted by D_A as $D_A \times MV_P$. And similarly the MV_A is weighted by D_P as $D_P \times MV_A$. The contribution of the two neighbors are combined as the initial motion vector estimation $\widetilde{MV}_{(1,1)}$:

$$\widetilde{MV}_{(1,1)} = (D_P \times MV_A + D_A \times MV_P) \quad (14)$$

With the normalization process, the estimation is modified as follows

$$MV_{(1,1)} = (w_{v_D(1,1)} \times MV_A + w_{h_D(1,1)} \times MV_P) \quad (15)$$

where $w_{v_D(1,1)} = \frac{D_P}{D_A+D_P}$, $w_{h_D(1,1)} = \frac{D_A}{D_A+D_P}$. With the same procedure, the estimation of motion vector in the block (1,2) is:

$$MV_{(1,2)} = (w_{v_D(1,2)} \times MV_B + w_{h_D(1,2)} \times MV_{(1,1)}) \quad (16)$$

where $w_{v_D(1,2)} = \frac{D_P}{D_B+D_P}$, $w_{h_D(1,2)} = \frac{D_B}{D_B+D_P}$. For the block (2,1), the estimation is

$$MV_{(2,1)} = (w_{v_D(2,1)} \times MV_{(1,1)} + w_{h_D(2,1)} \times MV_O) \quad (17)$$

where $w_{v_D(1,1)} = \frac{D_O}{D_A+D_O}$, $w_{h_D(1,1)} = \frac{D_A}{D_A+D_O}$. And finally, the motion vector for block (2,2) is

$$MV_{(2,2)} = (w_{v_D(2,2)} \times MV_{(1,2)} + w_{h_D(2,2)} \times MV_{(2,1)}) \quad (18)$$

where $w_{v_D(2,2)} = \frac{D_O}{D_B+D_O}$, $w_{h_D(2,2)} = \frac{D_B}{D_B+D_O}$. This idea can be easily extended to the rest of top-right, bottom-left and bottom right corner of the missing macroblocks. This method is denoted as **Proposed_MCD**, the proposed motion-compensation difference.

TABLE 4. The MOVIE comparisons of the error-concealed videos for different methods. It shows the Proposed method, Proposed_MVD, Proposed_MCD, and Proposed_MVD_MCD, and their performance gains against the state-of-the-art [21], denoted by Gain1, Gain2, Gain3 and Gain4. Gain5 is the gain of Proposed_MVD_MCD over [24].

Table with columns: Video, [21], [24], Proposed, Gain1, Proposed_MVD, Gain2, Proposed_MCD, Gain3, Proposed_MVD_MCD, Gain4, Gain5. Rows are grouped by video ID (0, 2, 4, 6, 8, 10, 12, 14) and video type (Coastguard, Foreman, City, Crew, Flower, Football, Stefan, AVERAGE).

TABLE 5. The average subjective score (each is average over 50 subjects) comparisons of the error-concealed videos for different methods. It shows the Proposed_MVD_MCD and its performance gains against the state-of-the-art [21], denoted by Gain1. Gain2 is the gain of Proposed_MVD_MCD over [24].

QP	Video	[21]	[24]	Proposed_MVD_MCD	Gain1	Gain2
20	Coastguard	76.72	73.1	78.64	1.92	5.54
	Foreman	70.08	75.54	73.96	3.88	-1.58
	City	76.28	74.18	79.62	3.34	5.44
	Crew	77.02	71.47	75.8	-1.22	4.33
	Flower	76.1	76.96	77.38	0.92	0.42
	Football	69.36	74.35	70.7	1.34	-3.65
	Stefan	70.78	74.24	74.48	3.7	0.24
	AVERAGE	73.76	74.26	75.79	1.98	1.53
22	Coastguard	75.48	69.97	77.2	1.72	7.23
	Foreman	70.16	72.37	75.22	5.06	2.85
	City	75.9	71.01	78.76	2.86	7.75
	Crew	74.82	70.83	76.36	1.54	5.53
	Flower	74.3	69.25	76.16	1.86	6.91
	Football	68.22	68.85	70.94	2.72	2.09
	Stefan	71.36	71.32	75	3.64	3.68
	AVERAGE	72.89	70.51	75.66	2.77	5.15
24	Coastguard	74.5	73.22	75.56	1.06	2.34
	Foreman	68.34	68.97	73.92	5.58	4.95
	City	75.38	69.28	77.02	1.68	7.74
	Crew	74.38	69.43	75.92	1.54	6.49
	Flower	76.24	65.36	74.48	-1.76	9.12
	Football	67.88	68.02	70.7	2.82	2.68
	Stefan	70.64	67.44	74.78	4.41	7.34
	AVERAGE	72.48	68.81	74.62	2.19	5.81
26	Coastguard	77.08	69.72	76	-1.08	6.28
	Foreman	67.42	65.12	72.7	5.28	7.58
	City	75.66	69.38	77.9	2.24	8.52
	Crew	76.92	73.22	78.26	1.84	5.04
	Flower	75.74	70.29	76.22	0.48	5.93
	Football	70.7	64.96	67.94	-2.76	2.98
	Stefan	71.84	68.27	74.06	2.22	5.79
	AVERAGE	73.62	68.7	74.72	1.17	6.02
28	Coastguard	76.7	69.78	76.04	-0.66	6.26
	Foreman	67.92	68.34	73.76	5.84	5.42
	City	74.46	69.94	76.12	1.66	6.18
	Crew	76.38	73.62	75.2	-1.18	1.58
	Flower	74.9	72.48	76.58	1.68	4.1
	Football	67.92	65.46	70.58	2.66	5.12
	Stefan	69.84	67.4	73.96	4.12	6.56
	AVERAGE	72.58	69.57	74.6	2.01	5.03
30	Coastguard	72.5	68.41	74.7	2.2	6.29
	Foreman	67.98	68.99	73.06	5.08	4.07
	City	74.48	67.98	75.96	1.48	7.98
	Crew	76.12	65.86	74.94	-1.18	9.08
	Flower	73.7	67.02	75.86	2.14	8.84
	Football	68.06	69.94	70.24	2.18	0.3
	Stefan	69.46	65.17	72.7	3.24	7.53
	AVERAGE	71.75	67.62	73.92	2.16	6.3
32	Coastguard	77.2	68.9	75.88	-1.32	6.98
	Foreman	67.52	69.23	72.98	5.46	3.75
	City	75.32	69.46	76.38	1.06	6.92
	Crew	76.16	69.62	76.14	-0.02	6.54
	Flower	73.8	67.56	75.02	1.22	7.46
	Football	66.1	66.63	69.2	3.1	2.57
	Stefan	70.48	68.52	73.3	2.82	4.78
	AVERAGE	72.36	68.56	74.12	1.76	5.56

our prior work in [27], **Proposed_MVD** in section 3, **Proposed_MCD** in section 4, and **Proposed_MVD_MCD** in section 5. The **Proposed_MVD_MCD** is also used to

compare a recent pixel-based error concealment method in [24]; the codes are provided by the authors and available in [28]. We used video encoder H.264 (JM 18.0).

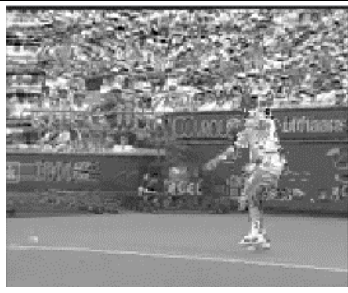
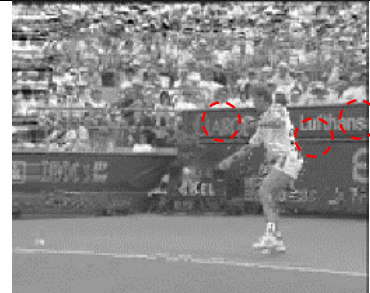
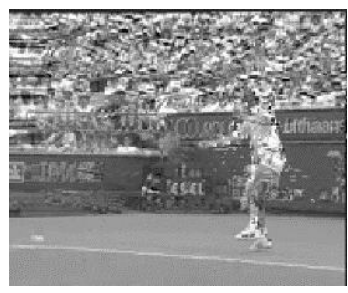
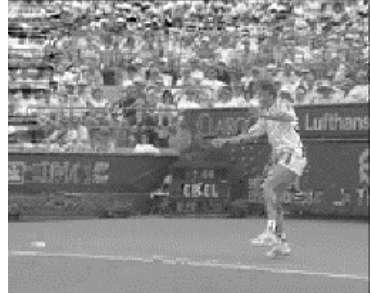
Method Frame #	[21]	[24]	Proposed_MVD_MCD
104			
	Frame PSNR=18.04, Frame SSIM=0.6300	Frame PSNR=24.2, Frame SSIM=0.8634	Frame PSNR=22.17, Frame SSIM=0.8373
105			
	Frame PSNR=18.11, Frame SSIM=0.6374	Frame PSNR=24.33, Frame SSIM=0.8693	Frame PSNR=22.19, Frame SSIM=0.8341

FIGURE 3. Visual comparisons among the work in [21], [24], and Proposed_MVD_MCD for Stefan with QP=20.

The considered videos are Coastguard, Foreman, City, Crew, Flower, Football and Stefan. We consider the Quantization Parameters (QP) for 20, 22, 24, 26, 28, 30 and 32 for different experimental settings. The packet is lost in every GOP (group of pictures) for error concealment for every video by above-mentioned methods.

The PSNR (Peak Signal to Noise Ratio in dB) is computed for the recovered video for each method (higher PSNR, better recovered quality). The PSNR comparison is shown in table 1. As can be seen in table 1 for QP=20, the **Proposed** is constantly better than the work in the state-of-the-art [21]; the Gain1 are all positive for all 7 videos and had an average gain of 1.86 dB. To extend this work by **Proposed_MVD**, the gain compared with [21] improved to 1.93 dB on average. For the work **Proposed_MCD**, the average gain against [21] can be up to 1.94 dB. Finally the combined method **Proposed_MVD_MCD** is better than [21] by 2.04 dB on average. As can be seen, the improvement grows from the **Proposed** to **Proposed_MVD_MCD**, indicating the effective and positive construction of the algorithms. For the QP=22, the trends are similar. All the four proposed method are better than [21] in every video, and the average gain are 1.79 dB, 1.86 dB, 1.83 dB and 1.93 dB respectively. It can be seen that both **Proposed_MVD** and **Proposed_MCD** improved from the **Proposed**, and the final combination **Proposed_MVD_MCD** improved the most. The rest of the comparisons showed the same performance trends. The comparisons of QP=24 are shown, which show similar results that the 4 proposed

algorithms (**Proposed**, **Proposed_MVD**, **Proposed_MCD** and **Proposed_MVD_MCD**) outperforms [21] in all cases by 1.76 dB, 1.85 dB, 1.79 dB and 1.91 dB on average. For QP=26, the gains on average are 1.76 dB, 1.83 dB, 1.78 dB and 1.86 dB. For QP=28, the average gains are 1.58 dB, 1.66 dB, 1.61 dB and 1.70 dB. For QP=30, the average gains are 1.47 dB, 1.53 dB, 1.49 dB and 1.56 dB. Finally for QP=32, the gains are 1.3 dB, 1.35 dB, 1.30 dB and 1.37 dB. These results indicate that the **Proposed** (as our prior work in [27]) can perform better than the state-of-the-art [21], and the **Proposed_MVD** and **Proposed_MCD** proposed in this paper can further improve the performance of **Proposed**. Finally the proposed combination of **Proposed_MVD_MCD** reaches the largest performance gain against the state-of-the-art [21]. To compared with the pixel-based error concealment method in [24], on average, the proposed work is slightly worse by 0.54 dB to 0.76 dB.

Table 2 shows that the comparisons in SSIM (Structural Similarity Index Metric) [29], which is typically between 0 and 1, and higher SSIM means better quality. As shown, all the average gains of the proposed methods against the state-of-the-art [21] in different QPs are positive, meaning better quality in SSIM. Again, the **Proposed_MVD_MCD** has the best gain among all the proposed works for almost all the QPs; the gain is 0.0499 for QP=20, 0.0459 for QP=22, 0.0466 for QP=24, 0.0439 for QP=28, 0.0405 for QP=30, and 0.0368 for QP=32. For QP=26, **Proposed_MCD** is the best with gain 0.0575. The **Proposed_MVD_MCD** is slightly worse than the pixel-based error concealment

Method Frame #	[21]	[24]	Proposed_MVD_MCD
117			
	Frame PSNR=27.63, Frame SSIM=0.7710	Frame PSNR=29.41, Frame SSIM=0.8418	Frame PSNR=28.75, Frame SSIM=0.8184
118			
	Frame PSNR=27.66, Frame SSIM=0.7721	Frame PSNR=29.39, Frame SSIM=0.8403	Frame PSNR=28.91, Frame SSIM=0.8230

FIGURE 4. Visual comparisons among the work in [21], [24], and Proposed_MVD_MCD for crew with QP=26.

method in [24] in SSIM by 0.0021 to 0.0057 on average.

In table 3, VQM (Video Quality Metric) [30] score is compared. Lower VQM means better spatial and temporal quality of the tested video (VQM=0 is the best quality), therefore if the gain is negative, the proposed work is visually better than the state-of-the-art [21] in VQM. Again, the average VQM gains are all negative for all QPs, meaning that the proposed results are visually better in VQM compared with the state-of-the-art [21]. For VQM, the **Proposed_MVD_MCD** is always the best among the proposed methods with average gains -0.0278 for QP=20, -0.0223 for QP=22, -0.0247 for QP=24, -0.0271 for QP=26, -0.0249 for QP=28, -0.0221 for QP=30, and -0.0198 for QP=32. The **Proposed_MVD_MCD** is on average better than the pixel-based error concealment method in [24] by 0.1056 to 0.1578 in VQM.

MOVIE (MOTION-based Video Integrity Evaluation) [31] measurement comparison is performed in table 4. Lower MOVIE score indicates better motion integrity of the tested video. Therefore negative gains mean better video quality in MOVIE measurements. Similarly, all the proposed works are better than the state-of-the-art [21] in MOVIE since their average gains are all negative for all QPs; the gains range from -0.0006 to -0.0008 . The **Proposed_MVD_MCD** outperforms the pixel-based error concealment method in [24] by 0.0014 to 0.0020 on average in MOVIE.

Table 5 shows the subjective experiment results. Here we only use the final version of the proposed work

Proposed_MVD_MCD to compare with the state-of-the-art work [21] and the recent pixel-based error concealment method in [24]. For each video under each QP, the resulting 3 videos from the 3 methods are shown to the subject. The rating system is for the subject to rate the videos with scores from 0 to 100, where 0 means the worst quality and 100 means the best quality. The subject can watch the videos for more than once and can refine the rating. The subject gave the final score to each of the 3 videos. One subject is responsible to evaluate all comparison videos for all the videos under all QPs. The subjective experiment is performed for 50 subjects, and each subject went through the above rating procedure. Each score in the table 5 is the average of the 50 scores from the 50 subjects for a specific video under a specific QP. As can be seen, even though there are negative gains of **Proposed_MVD_MCD** from [21] in some videos in some QPs, the average gains for all videos in specific QPs are positive for all QPs, ranging from 1.17 to 2.77. Also, compared with [24], the average gains of **Proposed_MVD_MCD** range from 1.53 to 6.3. This is also consistent with the VQM and MOVIE results which indicate that the **Proposed_MVD_MCD** provides better video quality. This shows that the proposed work produces better video quality evaluated by human observers.

For visual demonstration of [21], [24], and the proposed **Proposed_MVD_MCD**, example error-concealed frames by the methods are shown in fig. 3 and fig. 4. In fig. 3, the quality of the proposed **Proposed_MVD_MCD** is obviously better than [21]. And even though the frame PSNR and SSIM

TABLE 6. The time complexity comparison among different compared methods on average for 1 frame recovery.

	[21]	[24]	Proposed	Proposed MVD	Proposed MCD	Proposed MVD_MCD
Time (sec)	0.2068	28.6037	0.1556	0.1564	0.1725	0.2068

of [24] are better, the error-concealed areas by [24] are mostly blurred (due to the fact that the missing area is difficult to be estimated by the neighboring pixels when the missing area contains high frequency components), whereas those by the proposed **Proposed_MVD_MCD** are not, as shown in the red-circle areas. This situation persists in the next frame 105. The blurriness artifacts draws attentions of human observers and therefore the subjective scores of [24] is lower. Similar situation can be observed in fig. 4. The proposed **Proposed_MVD_MCD** is better than [21]. The proposed **Proposed_MVD_MCD** is visually better than [24] as shown in the red-circle areas. The visual results are consistent with the previous measurements of VQM, MOVIE (both correlate with human perceptions) and subjective experiments.

For complexity, with the PC using Intel Core i5-3230M 2.60GHz, RAM 8.00GB, the average execution time of frame recovery for each compared methods are in table 6. As can be seen, the complexity of the proposed work **Proposed_MVD_MCD** is similar to the one of [21], but is only about 0.7% of that of the pixel-based work [24]; the high complexity of the pixel-based error concealment methods is discussed in the introduction and also in [24]. This shows the efficiency of the proposed work.

VII. CONCLUSION

In this paper, we developed a new algorithm to solve packet loss problem in H.264. We used motion vector estimation method. From each corner of the lost MB, the motion vectors are estimated by the immediately available neighbors. And the estimates are further used to predict the motion vectors of the next missing block. Compared with the state-of-the-art method, we outperform by up to 1.86 dB on average. This work is improved by our design of estimation weightings with motion vector disparity and the motion-compensated differences; the improvement gains on average are up to 1.93 dB and 1.94 dB, respectively compared with the state-of-the-art method. These two mechanisms are combined and show the further improvement on the state-of-the-art method by up to 2.04 dB. The comparisons are also made in other video quality metrics; on average, the proposed work is better than the state-of-the-art method by up to 0.0575 in SSIM, -0.0278 in VQM (the lower the better), -0.0008 in MOVIE (the lower the better), and 2.77 in subjective evaluation. Compared to a state-of-the-art pixel-based method, the proposed method performs slightly worse in terms of PSNR and SSIM, but better in terms of VQM, MOVIE (both relate better with human perception) and subjective experiments, with much lower computational complexity.

REFERENCES

- [1] *Coding of Moving Pictures and Associated Audio for Digital Storage Media at Up to About 1.5 Mbit/s—Part 2:Video*, Standard ISO/IEC 11172-2(MPEG-1), International Standards Organization/International Electro Technical Commission(ISO/IEC) JTC 1, Mar. 1993.
- [2] *Information Technology—Generic Coding of Moving Pictures and Associated Audio Information—Part 2: Video*, Standard ISO/IEC 13818-2:2013, ITU-T International Standards Organization/International Electro Technical Commission (ISO/IEC) JTC 1, Nov. 1994.
- [3] *Video Coding for Low Bit Rate Communication*, document H.263, International Telecommunication Union-Telecommunication, 1995.
- [4] *High Efficiency Video Coding*, document H.265, Series H: Audiovisual and Multimedia Systems, Apr. 2013.
- [5] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, 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, Jul. 2003.
- [6] Y.-K. Wang, M. M. Hannuksela, V. Varsa, A. Hourunranta, and M. Gabbouj, "The error concealment feature in the H.26L test model," in *Proc. IEEE Int. Conf. Image Process.*, vol. 2, Sep. 2002, pp. 729–732.
- [7] S.-C. Hsia, "An edge-oriented spatial interpolation for consecutive block error concealment," *IEEE Signal Process. Lett.*, vol. 11, no. 6, pp. 577–580, Jun. 2004.
- [8] S. Belfiore, M. Grangetto, E. Magli, and G. Olmo, "Concealment of whole-frame losses for wireless low bit-rate video based on multi-frame optical flow estimation," *IEEE Trans. Multimedia*, vol. 7, no. 2, pp. 316–329, Apr. 2005.
- [9] J.-W. Suh and Y.-S. Ho, "Error concealment technique based on optical flow," *Electron. Lett.*, vol. 38, no. 18, pp. 1020–1021, Aug. 2002.
- [10] Y. Chen, Y. Hu, O. C. Au, H. Li, and C. W. Chen, "Video error concealment using spatio-temporal boundary matching and partial differential equation," *IEEE Trans. Multimedia*, vol. 10, no. 1, pp. 2–15, Jan. 2008.
- [11] J. Wu, X. Liu, and K. Y. Yoo, "A temporal error concealment method for H.264/AVC using motion vector recovery," *IEEE Trans. Consum. Electron.*, vol. 54, no. 4, pp. 1880–1885, Nov. 2008.
- [12] K. Seth, V. Kamakoti, and S. Srinivasan, "Efficient motion vector recovery algorithm for H. 264 using B-spline approximation," *IEEE Trans. Broadcast.*, vol. 56, no. 4, pp. 467–480, Dec. 2010.
- [13] Y. Zhang, X. Xiang, D. Zhao, S. Ma, and W. Gao, "Packet video error concealment with auto regressive model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 1, pp. 12–27, Jan. 2012.
- [14] G. Zhai, X. Yang, W. Lin, and W. Zhang, "Bayesian error concealment with DCT pyramid for images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 9, pp. 1224–1232, Sep. 2010.
- [15] J.-T. Chien, G.-L. Li, and M.-J. Chen, "Effective error concealment algorithm of whole frame loss for H.264 video coding standard by recursive motion vector refinement," *IEEE Trans. Consum. Electron.*, vol. 56, no. 3, pp. 1689–1695, Aug. 2010.
- [16] D. Agrafiotis, D. R. Bull, and C. N. Canagarajah, "Enhanced error concealment with mode selection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 8, pp. 960–973, Aug. 2006.
- [17] Y. Xu and Y. Zhou, "H.264 video communication based refined error concealment schemes," *IEEE Trans. Consum. Electron.*, vol. 50, no. 4, pp. 1135–1141, Nov. 2004.
- [18] M. C. Hwang, J. H. Kim, D. T. Duong, and S. J. Ko, "Hybrid temporal error concealment methods for block-based compressed video transmission," *IEEE Trans. Broadcast.*, vol. 54, no. 2, pp. 198–207, Jun. 2008.
- [19] Q. Peng, T. Yang, and C. Zhu, "Block-based temporal error concealment for video packet using motion vector extrapolation," in *Proc. IEEE Int. Conf. Commun., Circuits Syst. West Sino Expo.*, vol. 1, Jul. 2002, pp. 10–14.
- [20] B. Yan and H. Gharavi, "A hybrid frame concealment algorithm for H.264/AVC," *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 98–107, Jan. 2010.
- [21] J. Zhou, B. Yan, and H. Gharavi, "Efficient motion vector interpolation for error concealment of H.264/AVC," *IEEE Trans. Broadcast.*, vol. 57, no. 1, pp. 75–80, Mar. 2011.
- [22] T.-L. Lin, W.-C. Chen, and C.-K. Lai, "Recovery of lost motion vectors using encoded residual signals," *IEEE Trans. Broadcast.*, vol. 59, no. 4, pp. 705–716, Dec. 2013.
- [23] T.-L. Lin et al., "Video motion vector recovery method using decoding partition information," *IEEE J. Display Technol.*, vol. 12, no. 11, pp. 1451–1463, Nov. 2016.

[24] J. Liu, G.-T. Zhai, X.-K. Yang, B. Yang, and L. Chen, "Spatial error concealment with an adaptive linear predictor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 353–366, Mar. 2015.

[25] X. Liu, D. Zhai, J. Zhou, S. Wang, D. Zhao, and H. Gao, "Sparsity-based image error concealment via adaptive dual dictionary learning and regularization," *IEEE Trans. Image Process.*, vol. 26, no. 2, pp. 782–796, Feb. 2017.

[26] A. Akbari, M. Trocan, and B. Granado, "Sparse recovery-based error concealment," *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1339–1350, Jun. 2017.

[27] X. Wei et al., "Video error concealment method using motion vector estimation propagation," in *Proc. IEEE Int. Conf. Appl. Syst. Innov. (ICASI)*, May 2017, pp. 1335–1338.

[28] *Jing Liu's Homepage*. Accessed: Oct. 2017. [Online]. Available: <https://sites.google.com/site/jingliu198810/publication>

[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[30] *VQM Software*. Accessed: Jul. 10, 2017. [Online]. Available: <https://www.its.bldrdoc.gov/resources/video-quality-research/request-software.aspx>

[31] K. Seshadrinathan and A. C. Bovik, "Motion tuned spatio-temporal quality assessment of natural videos," *IEEE Trans. Image Process.*, vol. 19, no. 2, pp. 335–350, Feb. 2010.



XUBO WEI is currently pursuing the B.S. degree with the Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan, Taiwan. His research interests are image processing and video compression.



TZU-HAO SU is currently pursuing the B.S. degree with the Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan, Taiwan. His research interests are image processing and video compression.



TING-LAN LIN (S'08–M'11) received the B.S. and M.S. degrees in electronic engineering from Chung Yuan Christian University, Taoyuan, Taiwan, in 2001 and 2003, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California at San Diego, La Jolla, CA, USA, in 2010. In 2008, he interned with the Display System Group, Qualcomm, San Diego, CA, USA. Since 2011, he has been an Assistant Professor with the Department of Electronic Engineering, Chung Yuan Christian University, and since 2015, he has been an Associate Professor with the Department of Electronic Engineering, Chung Yuan Christian University. Since 2018, he has been an Associate Professor with the Department of Electronic Engineering, National Taipei University of Technology, Taipei, Taiwan. His current research interests include (multiview) video compression and pixel estimation with the techniques of mathematical optimization and deep learning algorithms.



XUTAO WEI is currently pursuing the B.S. degree with the Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan, Taiwan. His research interests are image processing and video compression.



YU-LIANG CHIANG is currently pursuing the B.S. degree with the Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan, Taiwan. His research interests are image processing and video compression.

...