A Fast H.264/AVC-Based Stereo Video Encoding Algorithm Based on Hierarchical Two-Stage Neural Classification

Jui-Chiu Chiang, Wei-Chih Chen, Lien-Ming Liu, Kuo-Feng Hsu, and Wen-Nung Lie

***Abstract*—Stereo video, targeting at matching what the humans see in the real world, offers depth perception on observed scenes. The problem of compressing such a huge amount of video data has received considerable attention in the past few years. Its challenge comes from the much more complicated parameter selection (e.g., mode decision) process than single-channel video coding. To cope with this problem, the currently developed H.264/AVC-based JMVM platform is adopted here for implementation, where the encoding of the left-view channel is purely based on predictions from the temporal domain, while for the right-view channel, combined predictions from the temporal and the inter-view domains are exploited and a hierarchical two-stage neural classifier is designed for fast mode decision. The first-stage neural classifier determines candidates of block partition for each macroblock, while the second-stage classifier aims to choose the most probable prediction sources among the temporally forward/backward and inter-view directions. All input features for both stages of neural classifiers are calculated from simple inter-frame and inter-view analyses. In our scheme, the popular fast motion estimation schemes can be also cooperated for further speedup. Experiment results reveal that our proposed algorithm is capable of achieving up to 97% of time savings with nearly ignorable quality degradation and acceptable bit-rate increase (up to 5.67%).**

***Index Terms*—H264/AVC, Joint Multi-View Video Model (JMVM), mode decision, neural classifier, stereo video coding.**

I. Introduction

THE rapid advance in 3-D display technology, coupled with the achievable high efficiency in video compression algorithms, moves our home audio-visual entertainment towards a greater perceptual realism. Many LCD-panel manufacturers are delivering their 3-D display products to the markets. The product costs are expected to be decreased rapidly once the applications (e.g., 3-DTV) are widely developed. Digital 3-D TV program broadcast via broadcasting satellite digital (BSD) satellite has been activated in Japan since December 2007. Advanced Three-dimensional Television System Technologies (ATTEST) [1] and 3-D TV projects in Europe, 3-D digital multimedia broadcast (DMB) system in Korea, and others, are currently under development. These all lead us to a promising 3-D stereo era.

One kind of input to 3-D displays is composed of two-channel (right-eye and left-eye) color videos, which are usually captured by two cameras arranged in parallelism. Since the data amount of two-channel videos is two times of that of traditional 2-D videos, the intuitive way to encode stereo videos is by the simulcast method, implying independent coding for each individual channel. The resulting compression efficiency is not satisfactory accordingly, due to the ignorance of redundancies between neighboring views at each time instant. Thus, how to maximize the coding efficiency for stereo sequences is an important topic gaining more attention.

Stereo video coding was first standardized in MPEG-2 multi-view profile [2], where the left-view sequence is encoded based on the conventional motion compensation architecture, while the right-view sequence is predicted from the previous reconstruction frame (by motion estimation, ME), as well as the corresponding decoded left-view (by disparity estimation, DE, [3], [4]). Many works attempt to explore the statistical dependency between neighboring views [5]–[7], especially for videos of high motion which present weak temporal correlation.

There are some demands in capturing images/videos via multi-view (more than two) cameras arranged in certain geometry (e.g., parallel, converged, inwards, or outwards). In such a scenario, the compression issue presents more difficulty and complexity (increasing data amounts or less correlation between neighboring views). Moving Picture Experts Group (MPEG) thus issued a "Call for Proposals" to standardize techniques for multi-view coding (MVC) [8], [9]. At this moment, the MVC schemes under development are mostly extended from the well-known H.264/AVC standard which offers good coding efficiency for single-channel videos. The reference software for MVC was Joint Multi-view Video Model (JMVM) [10] since 2006, and the updated software is called Joint Multi-view Video Coding (JMVC) [11] where several tools in JMVM are removed, such as illumination compensation, motion skip mode and some options related to scalable video coding. However, the coding architectures of JMVM and JMVC are the same.

As is well known, H.264/AVC provides several new features, including ME with variable block sizes, multiple reference frames, quarter-pixel interpolation, and rate-distortion optimization (RDO) [12]. Especially, the so-called "mode decision" concerns about the process of deciding the best partition, prediction sources, and corresponding motion vectors for a macroblock (MB). Undoubtedly, the associated computational complexity is substantially high, due to the increasing number

Manuscript received December 04, 2009; revised April 05, 2010; accepted July 29, 2010. Date of publication August 16, 2006; date of current version March 16, 2011. Part of this work was presented at the 3DTV Conference 2008. This work was supported in part by the NSC project under Grant 96-2221-E-194-040-MY2. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Dan Schonfeld.

J.-C. Chiang, W.-C. Chen, K.-F. Hsu, and W.-N. Lie are with the Department of Electrical Engineering, National Chung Cheng University, Chia-Yi 621, Taiwan (e-mail: rachel@ccu.edu.tw; ieewnl@ccu.edu.tw).

L.-M. Liu is with ASMedia Technology, Inc., Taipei 231, Taiwan

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTSP.2010.2066956

of rate-distortion cost calculations on various partition candidates [12], [13]. Consequently, many efforts have been made to develop fast algorithms for mode decision [14]–[17].

Many works for fast mode decision in H264/AVC perform possible early termination after first evaluating the SKIP mode, based on a statistical observation that the SKIP mode is most time-efficient and usually dominant at larger quantization parameter (QP) values [15]–[17]). Generally, most strategies for fast inter mode decision rely on rules and thresholds according to some useful observations/properties, implying that some easily-computed quantities are compared to pre-defined thresholds in an order that the more probable modes are always checked before others. This kind of strategy however requires considerable heuristics and will be much more complicated when considering multi-view coding where inter-view redundancies are to be explored. This motivates us to seek for a new method for the mode decision of H.264/AVC-based stereo or multi-view video coding.

To reduce the overall complexity for ME/DE and mode decision, a number of fast encoding algorithms were proposed [18]–[23]. One common strategy is to reduce the search range of ME/DE based on a loop relationship between the motion vectors and the disparity vectors, and/or to estimate the probable modes of MBs in the secondary view based on the best mode of the corresponding MBs in the primary view. In [18] and [19], unnecessary computations for ME and DE are diminished by reducing the search ranges according to some reasonable prediction mechanisms. In [20], a disparity to motion (DTM) and a motion to disparity (MTD) algorithm were separately proposed to reduce the required computation in motion and disparity search. In DTM algorithm, the motion vector is first predicted based on a known disparity vector which is estimated via full search, followed by a refinement via local search. For MTD algorithm, the strategy is similar. In [21], a fast mode decision algorithm was proposed, where the coding mode of each MB in the right view is predicted from that of a corresponding MB in the left view. It was shown that a large amount of search points can be reduced, with a slight PSNR degradation, but the actual time speed-up was not reported. Based on an observation that MBs of high motion tend to be best predicted in disparity direction, an object-based mode decision algorithm was proposed in [22], where the foreground MBs are initially processed by disparity estimation and the background MBs by motion estimation. In [23], a fast mode decision algorithm with dynamic multi-thresholds was proposed, where all the possible modes relating to block size partition are divided into four groups and sequentially evaluated group by group. Adaptive thresholding on optimal rate-distortion (RD) cost is designated for evaluating each group of modes and early termination might be activated to speed up the process. Their method, however, did not consider the speed-up in the issue of determining the prediction sources from forward-temporal, backward-temporal, or disparity views.

This paper proposes a new technique for mode decision in H.264/AVC-based stereo video coding by using a hierarchical two-stage neural classifier followed by fast motion estimation, targeting at a substantial speed-up without notable increase on bit rate. Section II introduces the H.264/AVC-based stereo video coding architecture and Section III gives some observations on mode distributions. The proposed neural classifier and related experiments are detailed in Sections IV and V, respectively. Finally, Section VI draws some conclusions.

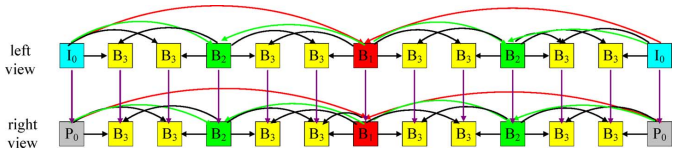

Fig. 1. Basic structure for encoding two-channel video with a Group of Pictures (GOP) size of 12 frames [10].

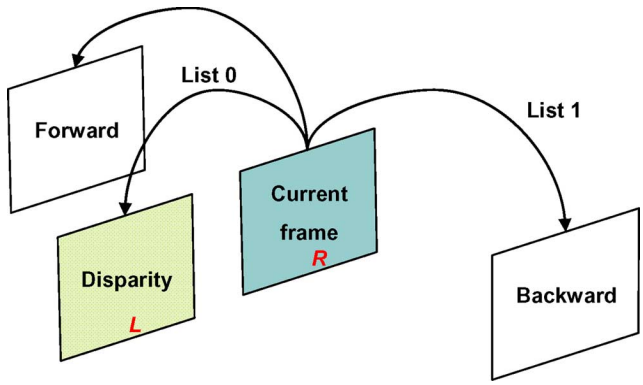

Fig. 2. Reference (prediction) sources for encoding right-view images.

II. 264/AVC-Based Stereo Video Coding Architecture

Joint Multi-view Video Model (JMVM) [10] is the reference software platform implementing H.264/AVC-based multi-view video coding and now evolved into JMVC [11]. The main coding architecture of JMVC is actually the same as JMVM. Fig. 1 illustrates the coding structure and reference relations adopted by JMVM for two-channel videos. The left-view channel is coded with hierarchical B pictures to realize temporal scalability, while the right-view channel is coded with a similar but enriched structure (in addition to predictions from the temporally forward and backward directions, extra prediction from the left-view image is allowed).

To extend the original H.264/AVC coding architecture for the right-view channel, the reconstructed forward motion frame and the decoded left-view image are placed in the same List 0 reference buffer (but not simultaneously), while the reconstructed backward motion frame is placed in List 1 buffer, as depicted in Fig. 2. In this manner, stereo video coding allows five combinations of prediction sources for B frames in the right-view sequence, namely "forward motion," "backward motion," "left-view disparity," "forward motion plus backward motion," and "disparity plus backward motion."

Mode decision rule in MVC is similar to that in H.264/AVC. For a given QP, the best mode for a MB can be determined by minimizing the Lagrange cost as follows:

$$\begin{aligned} &J(s,c,\mathrm{MODE}|\mathrm{QP},\lambda_{\mathrm{MODE}}) \\ &\quad = SSD(s,c,\mathrm{MODE}|\mathrm{QP}) \\ &\qquad + \lambda_{\mathrm{MODE}} R(s,c,\mathrm{MODE}|\mathrm{QP}), \end{aligned} \tag{1}$$

$$\lambda_{\mathrm{MODE}} = 0.85 \times 2^{(\mathrm{QP}-12)/3} \tag{2}$$

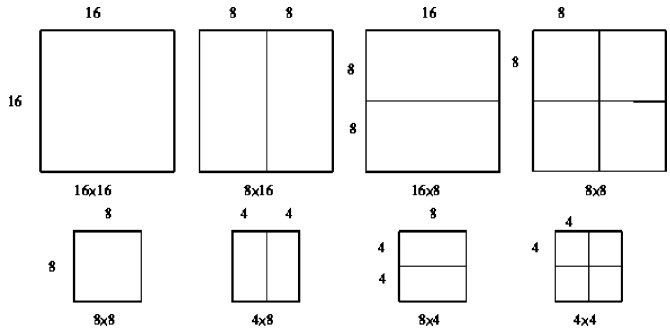

Fig. 3. Variable block sizes supported in H.264/AVC.

where MODE represents the mode under evaluation; s and c denote the original and the reconstructed MB data (based on the considered MODE and QP), respectively; λ_{MODE} is a Lagrangian multiplier; SSD represents the sum of squared difference between s and c; and R stands for the number of bits to encode c, e.g., headers, motion vectors and quantized residual coefficients. However, considering an additional prediction source from the inter view will make the complexity of mode decision much higher than that for traditional H.264/AVC. Consequently, lots of computing time is required accordingly.

For H.264/AVC baseline profile with one reference frame only, there are 259 $(= 3 + 256)$ possible modes for an inter-coded MB. The first three modes correspond to 16×16, 16×8 and 8×16 partitions. The 4 kinds of sub-block partition for each 8×8 block, as illustrated in the second row of Fig. 3, lead to $4^4 = 256$ sub-block combinations for each MB. For stereo video coding based on JMVM/JMVC, the 5 combinations of prediction sources make the possible inter modes amount to 160 055 ($160\,055 = 5 + 5^2 + 5^2 + 20^4$), where 5 is for 16×16 pixels, 5^2 for 16×8 pixels, 5^2 for 8×16 pixels, and $(5+5+5+5)^4$ for 8×8 pixels and its sub-block partitions (i.e., 8×4, 4×8 and 4×4 pixels). Note that each of the four 8×8 blocks in a MB can have its own respective partition. Obviously, the number of possible modes has been substantially increased from 259 to 160 055, seemingly a heavy computational load for stereo video coding.

III. Observations on Mode Distribution

Mode distribution is actually closely related to video contents (e.g., motion activity) and coding parameters such as QP, temporal hierarchy, and reference structure. With a prior knowledge of mode distribution, an efficient algorithm can be designed accordingly.

A. Mode Distribution Versus Motion Activity

Different from single-channel video coding, the inter-view redundancy should be carefully considered and removed for stereo video coding. Usually, the background and low motion areas are best predicted from the temporal domain, while high motion areas (foreground) tend to be predicted from the disparity domain. As shown in Fig. 4(b), most of the MBs predicted from disparity domain (the brighter boxes) come from the dancing persons and the changing shadows on stage.

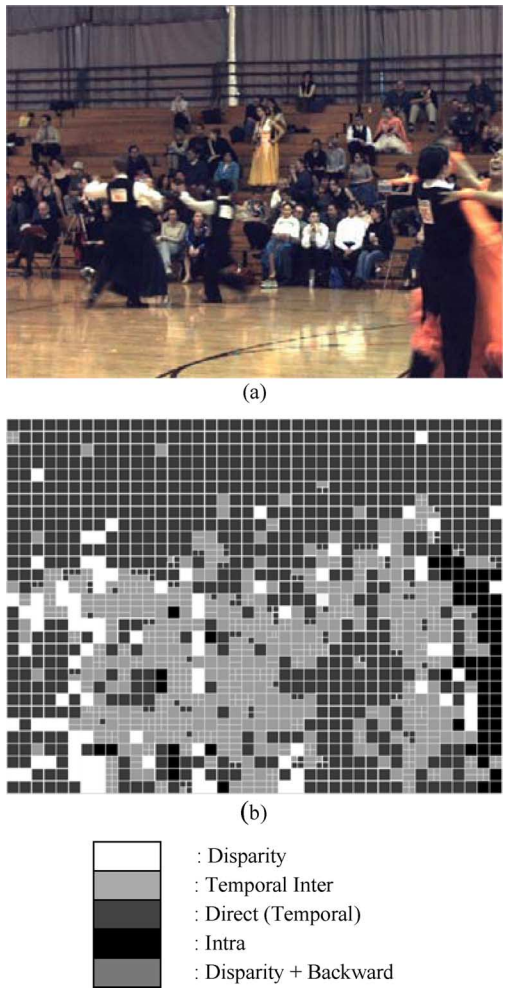

Fig. 4. Mode distribution for frame #7 of "Ballroom" sequence. (a) Original video. (b) Mode distribution.

TABLE I
Mode Distribution at Different QPs.

	QP=16	QP=28	QP=40
Direct	26.60%	64.59%	79.81%
16×16	19.06%	21.37%	14.71%
16×8	10.17%	4.86%	2.14%
8×16	9.31%	4.30%	1.97%
8×8	20.76%	2.98%	0.42%
Intra	14.10%	1.90%	0.95%

B. Mode Distribution Versus QP Value

It is clear from (1) and (2) that block partition, or the mode decision, depends on QP setting. Table I summarizes statistical distributions of 6 kinds of block partitions (Direct, 16×16, 16×8, 8×16, 8×8 and its sub-blocks, and Intra) under different QP values by operating JMVM with full mode search (i.e., all kinds of block partitions and prediction sources are considered) for several test videos. It reveals that the most probable modes are Direct and 16×16 for larger QPs. This can be easily derived from (2) that when QP is large, λ_{MODE} is accordingly large and the bit rate R dominates the RD cost. Thus, the Direct and 16×16 modes are most likely chosen due to fewer bit rate requirement. On the

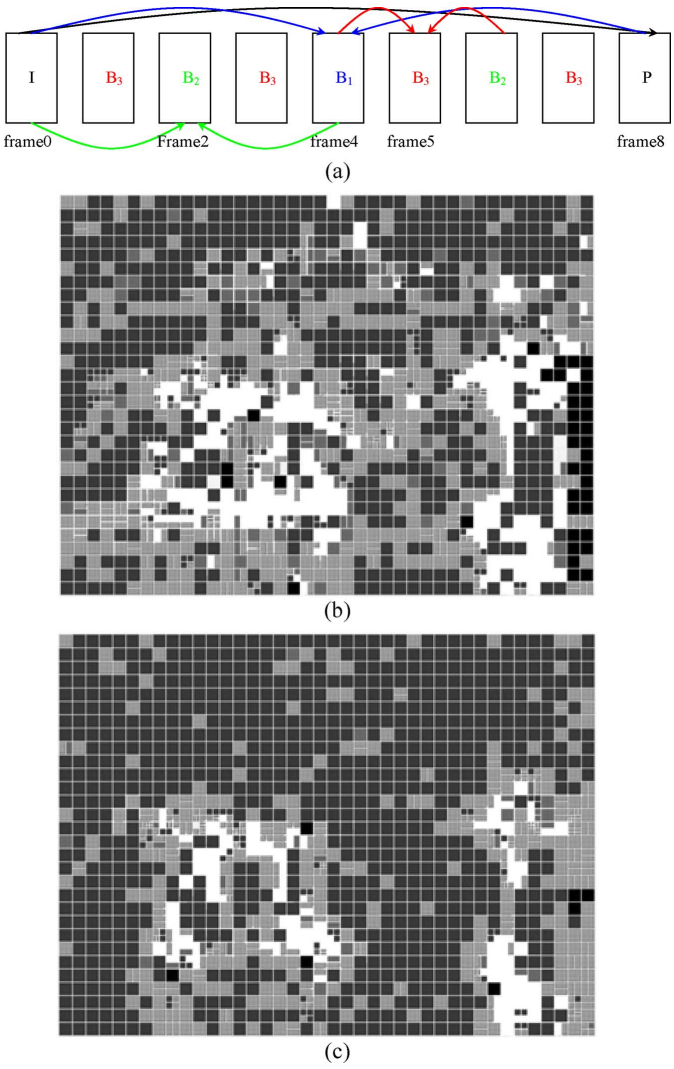

Fig. 5. Temporal hierarchy and mode distributions for frames with different positions in the hierarchy, for the "Ballroom" sequence. (a) Illustration of temporal hierarchy in a GOP of size 8. (b) Mode distribution of frame #4. (c) Mode distribution of frame #5.

other hand, these 6 kinds of modes are more equally distributed when QP is small (accordingly, λ_{MODE} is relatively small and the distortion will be dominant). Hence, the fast mode decision algorithm should be adaptive with different QP values.

C. Mode Distribution Versus Temporal Hierarchy

For "hierarchical B" structure in MVC, temporal distance between a current frame and its reference frames becomes varying. For instance, when GOP size is 8, as depicted in Fig. 5(a), frames #4, #2, and #5 (assigned as B_1, B_2, and B_3 in the temporal hierarchy, respectively) have a temporal distance of 4, 2, and 1, respectively, with respect to their reference frames. Experimentally, for frames on top of the hierarchy (e.g., B_1), the preferred prediction source would come from the disparity domain, due to reduced correlation between two frames of larger temporal distance. Fig. 5(b) and (c) verifies this point, where the number of disparity prediction (the brighter boxes) for frame #4 is larger than that for frame #5.

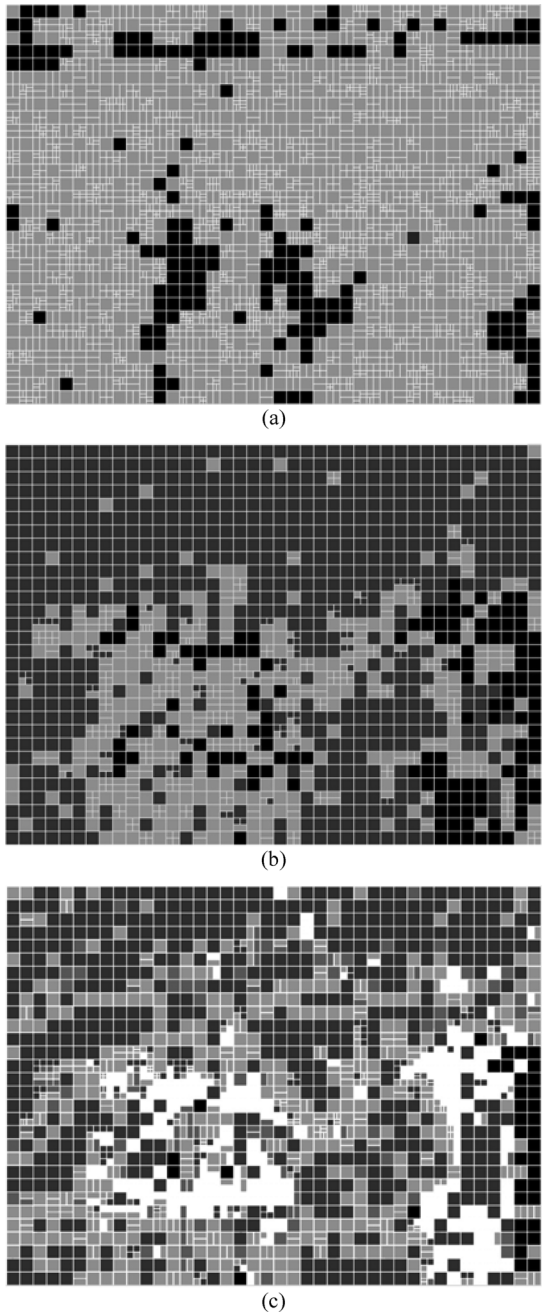

Fig. 6. Mode distributions for frame #4 of "Ballroom" sequence coded at $\mathrm{QP} = 16$ (a) H.264/AVC baseline profile, (b) H.264/AVC main profile, and (c) MVC.

D. Block Partition Versus Reference Structure

To evaluate the impact of reference structure on the selection of block sizes, Fig. 6 demonstrates the results of three experiments: (a) H.264/AVC baseline profile with IPPP coding structure, (b) H.264/AVC main profile with IBBB coding structure, and (c) MVC (stereo video coding based on JMVM) with IBBB coding structure (all with a GOP size of 12). The test video is "Ballroom" and the QP is set to 16. For the baseline profile, only forward motion prediction is estimated, whereas both forward and backward motion predictions are allowed for the main profile. From Fig. 6(a) and (b), it is obvious that small block partition (e.g., 4×8, 8×4, and 4×4) in baseline profile is more than

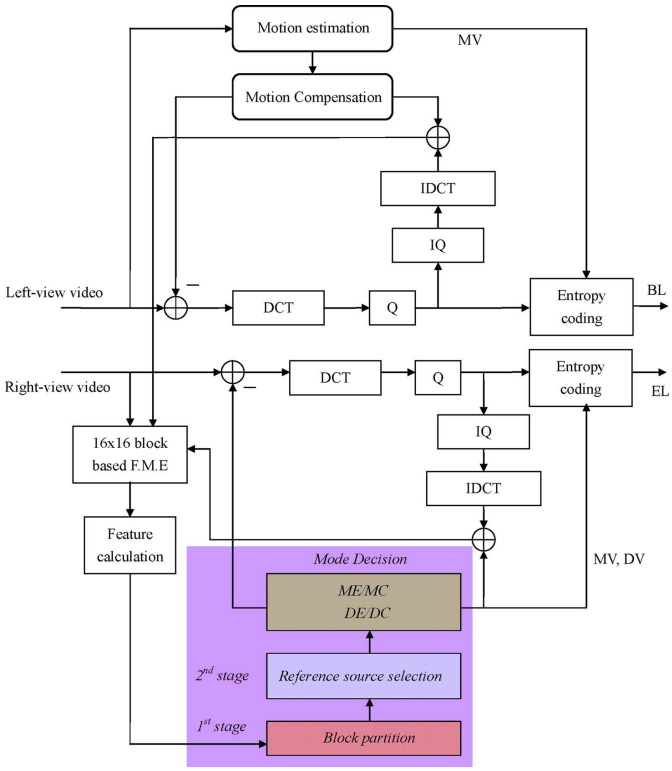

Fig. 7. Block diagram of the proposed inter-coding architecture for stereo videos.

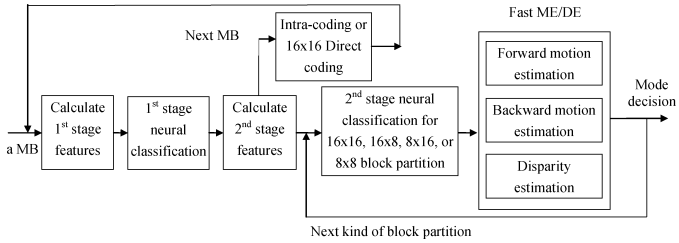

Fig. 8. Illustration of hierarchical two-stage neural classification for fast mode decision.

that in main profile. Comparing Fig. 6(b) and (c), it is found that sub-blocks predicted from temporal domain might be merged into larger blocks which can be predicted from the disparity domain. All these behaviors reveal a possibility that coding efficiency may not be notably degraded if sub-block partitions (i.e., 8×8 and below) are ignored.

IV. Proposed Neural Classification Scheme

We propose a two-stage neural classifier for fast inter-coding of stereo videos (the intra-coding part is beyond our discussion), based on the observations of mode distributions described in the previous section. The first-stage neural classifier is responsible for determining the probable block partitions, while the second-stage neural classifier is for determining the prediction source candidates, both are based on features calculated from temporally-neighboring or neighboring-view frames. In general, any fast motion estimation algorithm under development (e.g., the one adopted in JMVM) can be integrated with our two-stage neural classifier to further speed up the encoding process. The proposed inter-coding architecture is shown in Fig. 7.

According to the discussion in Section II, there are a total of 160 050 possible modes for each right-view MB based on JMVM implementation. These modes can be completely arranged into a two-stage hierarchy:

1) first-stage: block partition into 16×16 Direct, 16×16, 16×8, 8×16, 8×8 (and below), and Intra, and
2) second-stage: prediction sources including forward motion, backward motion, and disparity frames.

In case of 8×8 block partition, a further partitioning into 4×8, 8×4, or 4×4 sub-block size will be performed after the second-stage neural classification (i.e., during the process of fast ME/DE). Fig. 8 illustrates the processing flow diagram of our algorithm.

A. First-Stage Neural Classifier

In general, block partition is closely related to the characteristics of image content, especially the texture. For example, smooth areas would be likely encoded by larger blocks, such as Direct/SKIP or 16×16 inter modes. On the contrary, smaller sub-blocks, such as 8×8, 8×4, 4×8, or 4×4, are probably used to encode areas of complex texture. Another factor connects to block partition is the motion. According to experiments, smaller sub-blocks are best chosen to encode areas of complex motion activity (e.g., areas near boundaries of motion objects), in order to produce precise predictions.

Consequently, we first compute forward-motion-compensated and disparity-compensated residuals based on the 16×16 size, denoted as $D_{t-h,t}$ and $D_{d,t}$, respectively, for the current MB, where d stands for disparity reference, t and $t-h$ represent the time indices, and temporal distance h depends on the position in the temporal hierarchy for the current frame. For example, for a B_1 frame, h is half of the GOP size [please see Fig. 5(a)]. To remove the influence of prediction source on block partition, $D_{t-h,t}$ or $D_{d,t}$, which has the least L_1-norm, is chosen and its associated features (see below) are calculated as the inputs to the neural network. Note that the result of $D_{t-h,t}$ and $D_{d,t}$ (referring to the "FME for 16×16 macroblock" box in Fig. 7) can be used again once the 16×16 inter or the disparity mode is chosen as one of the candidates for evaluation afterwards. There are in total nine image features calculated from $D_{t-h,t}$ or $D_{d,t}$ for the first-stage neural classifier, as listed as follows.

1) MB mean and MB variance.
2) Differences of local means and local variances between the left and the right 8×16 areas, denoted as $mean_{8\times16}\_Diff$ and $var_{8\times16}\_Diff$, respectively, in Fig. 9(a).
3) Differences of local means and local variances between the top and the bottom 16×8 areas, denoted as $mean_{16\times8}\_Diff$ and $var_{16\times8}\_Diff$, respectively, in Fig. 9(b).
4) Percentage of the number of high-residue pixels (with respect to a pre-determined threshold).
5) Differences of residue means and residue variance between the current MB and the area of eight surrounding MBs, as shown in Fig. 9(c).

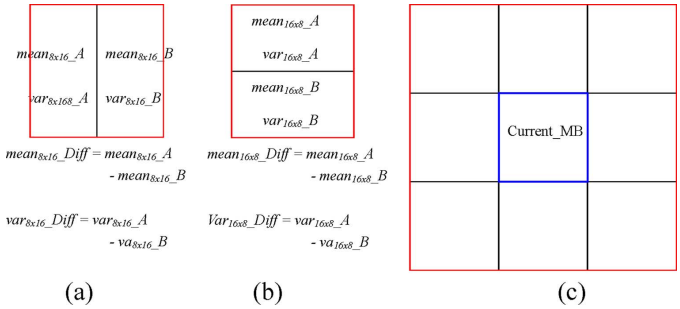

Fig. 9. Illustration of image features calculated for the first-stage neural classifier.

Note that the masks in Fig. 9(a) and (b) can be considered as two templates used to identify the existence of 8×16 and 16×8 block partitions, respectively.

A three-layer back-propagation neural network (BPNN) is carried out for the first-stage classifier. There are 9, 9, 6 neurons for the input, hidden, and output layer, respectively. All the input feature values are normalized to 0.0–1.0. After sufficient training, the neural network is capable of outputting six scores of 0.0–1.0 for any input feature vector. Our strategy is to prune out output neurons of lower scores, e.g., choose K_1 out of 6. K_1 can be user-defined according to quality or speed-up request. Clearly, a smaller K_1 will speed up the following mode decision process substantially, but also increase the bit rate (i.e., inaccurate prediction due to the skipping of some modes). On the other hand, a larger K_1 results in a fewer time reduction, though the coding performance can be maintained ($K_1 = 6$ is equivalent to the original JMVM encoder). To compromise the coding performance and speed-up, an adaptive-K_1 algorithm, which selects output neurons by comparing their scores to an empirically determined threshold, is adopted. Once an output neuron's score is above the threshold, the corresponding block partition is accepted as one candidate for further evaluation. Note that these six kinds of partitions do not have equal weighting in the processing time needed. For examples, "16×16 Direct" and "Intra" take their advantages of not further going into the second-stage classifier and motion/disparity estimation process, while "8×8" requires further splitting into 8×4, 4×8, and 4×4 sub-blocks and is most time-consuming.

B. Second-Stage Neural Classifier

According to the discussion in Section II, prediction source selection for MVC is much more complex than that for traditional H.264/AVC. Similarly, image features calculated from the motion- or disparity-compensated residual MBs, including $D_{t-h,t}$, $D_{t,t+k}$, and $D_{d,t}$, are figured out for the second-stage neural classifier (note the additional $D_{t,t+k}$ from backward motion reference).

Similar to the first-stage, local means and local variance of the residual MBs are calculated. Here, "local" means the sub-MB or sub-block area chosen by the first-stage neural classifier (the two stages form a hierarchy, not cascade). For example, if the 16×8 partition is one of the K_1-out-of-6 candidates at the first stage, local means and local variance for the second stage will be calculated per 16×8 area. The six image features adopted at the second stage are listed as follows.

1) Local mean and local variance of $D_{t-h,t}$.
2) Local mean and local variance of $D_{t,t+k}$.
3) Local mean and local variance of $D_{d,t}$.

There are 6, 5, and 3 neurons for the input, hidden, and output layer, respectively, in the second stage neural classifier. As mentioned in Section II, our stereo video coding based on JMVM implementation allows five combinations of predictions from three different sources. However, we do not make classification into five classes, but three, i.e., forward motion, backward motion, and disparity. Similarly, we perform a K_2-out-of-3 selection process and combine the survived prediction sources, resulting in 1–5 prediction candidates. For example, if output neurons of "disparity" and "backward motion" are selected, then predictions from "disparity," "backward," and "disparity + backward" directions will be examined.

Similarly, an adaptive-K_2 algorithm is devised for the second-stage classifier. First, the three output neuron scores are clustered (with a measure of nearest neighbor distance [24]) based on a distance threshold, leading to a result of 1–3 clusters. Only output neurons belonging to the cluster of the highest average score are selected.

C. Fast Motion Estimation

Following the decision of probable modes by our two-stage neural classifier, any popular fast algorithms (e.g., three-step search, diamond search, etc.) can be used for motion/disparity estimation to further speed up the processing. In this paper, the FME (fast motion estimation) algorithm supported by JMVM encoder is adopted directly.

V. Experiment Results

To evaluate the performance of the proposed algorithm, six stereo video sequences, each composed of 100 frames of 640×480 pixels, are used for experiments. Among them, "Ballroom" (high object motion), "Race1" (with both object and camera motions), "Exit" and "Vassar" (low object motion) are multi-view sequences captured by 1-D linear camera array (hence, only two neighboring channels are used for test) and "Soccer2" (with both object and camera motions) and "Puppy" (low object motion) are stereo videos.

In the following, comparisons are made on the bit rate, peak signal-to-noise ratio (PSNR) and total encoding time (not just the mode decision time) of the right-view channel. The computing platform is JMVM 8.0, which is operated at full-mode decision for comparison.

Table II lists the encoding parameters in our experiments. Among them, vertical disparity is searched over a smaller range ($[-8, 8]$ pixels) by considering the near parallelism between two cameras. The "Direct" mode is now defined to refer to "forward + backward motion" after simulations indicating that up to 14% of bit-rate increase is incurred if the Direct mode is implemented using disparity and backward motion frames. According to the observations in Sections III-B and III-C, mode distribution changes with varying QPs and positions in temporal hierarchy. Hence, training of the neural classifiers is performed for QP =16, 28, and 40, for B_1, B_2, and B_3 hierarchy positions,

TABLE II
ENCODING PARAMETERS FOR STEREO VIDEO CODING

Frame format	4:2:0
GOP size	12 frames/16 frames
Number of prediction frames for B frames	3 frames (forward, backward and disparity direction)
Disparity search range	X∈ [-16, 16] pixels, Y∈ [-8, 8] pixels
Direct mode	Forward motion + backward motion
Motion search range	[-16, 16] pixels
RDO	On
Deblocking filter	On
1/4 Sub-pixel search	On
Quantization Parameters	16, 28, 40

and for GOP =12 and 16. Among the six videos, only three GOPs of "Ballroom," "Exit," and "Vassar" are used for training and the others (including those remaining GOPs of the training videos) are used for test. Note that when training the first-stage and second-stage neural classifiers (denoted as NN1 and NN2, respectively), input features and ground truth of each training sample are calculated according to the result of full-mode search by using JMVM. For examples, the optimal mode found by JMVM is accepted as the ground truth; input features (i.e., three sets of local means and local variances) for NN2 are computed based on the block partition information (e.g., 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, or 4×4) of the optimal mode.

In each entry of Tables III–VI, the top number represents the result of GOP $= 12$, while the bottom number with parenthesis shows that for GOP $= 16$.

A. Coding Performance of Single-Stage Processing

We apply several schemes individually to see their own performances and then combine them for further speed-up. In addition to the proposed NN1 and NN2 classifiers, "8×8 off" and "probability-based" schemes operating at the first stage (i.e., block partition selection) are also introduced. "8×8 off" means disabling of 8×8 block and its sub-blocks (according to the observation in Section III-D), while "probability-based" always selects, from the 6 candidates, the top K_1 ones based on their prior statistics.

Table III(a)–(e) shows the performances of "8×8 off," "probability-based ($K_1 = 3$)," NN1 (fixed $K_1 = 3$), NN1 (adaptive-K_1) and NN2 schemes, respectively. All figures therein, including ΔPSNR, Δbitrate, and Time saving (TS), are calculated with respect to the JMVM full-mode decision. Table III(a) shows that the speed-up by disabling 8×8 block and its sub-block partitions is quite efficient (TS $= 63\%$ on average, less than 3% of bit rate increase and ignorable PSNR loss), which verifies our observation in Section III-D).

On the other hand, the "probability-based" scheme selects fixed block partition candidates (e.g., Direct, 16×16 and 16×8 when QP =28, 40, according to Table I) for any input MB. Table III(b) shows that though an average of 74% of time savings can be achieved at moderate-to-low bit rates, the speed-up performance degrades substantially (TS $\cong 23\%$) at high bit rates, due to a uniform mode distribution. By the way, it produces a noticeable bit rate increase, especially for the "Ballroom" sequence.

TABLE III
PERFORMANCE ON PSNR, BIT RATE, AND TIME SAVING (TS) FOR SEVERAL CODING SCHEMES AT QP =16, 28, AND 40 AND WITH GOP = 12 AND 16 (NUMBERS WITH PARENTHESIS). (a) "8×8 OFF". (B) PROBABILITY-BASED. (c) NN1 WITH A FIXED $K_1 = 3$. (d) NN1 WITH ADAPTIVE K_1. (e) NN2

(a)

sequence	QP=16 ΔPSNR (dB)	QP=16 Δbitrate (%)	QP=16 TS (%)	QP=28 ΔPSNR (dB)	QP=28 Δbitrate (%)	QP=28 TS (%)	QP=40 ΔPSNR (dB)	QP=40 Δbitrate (%)	QP=40 TS (%)
Ballroom	-0.05	-0.02	63.75	-0.02	1.29	63.72	-0.01	0.50	63.44
	(-0.04)	(-0.07)	(63.58)	(-0.02)	(1.33)	(64.31)	(-0.01)	(0.37)	(62.55)
Exit	-0.04	0.27	63.93	-0.01	0.60	63.64	-0.01	0.11	63.81
	(-0.03)	(0.39)	(64.67)	(-0.01)	(1.56)	(62.77)	(-0.01)	(-0.51)	(62.31)
Vassar	-0.06	-0.28	64.02	-0.01	0.43	63.69	0.00	-0.30	63.47
	(-0.06)	(-0.26)	(64.42)	(-0.01)	(0.23)	(64.95)	(0.00)	(-0.04)	(63.76)
Soccer2	-0.04	-0.06	63.99	-0.01	0.92	63.23	0.00	-0.20	63.40
	(-0.04)	(-0.20)	(63.61)	(-0.02)	(0.5)	(63.39)	(0.00)	(-0.11)	(63.48)
Race1	-0.02	1.13	63.71	-0.02	1.88	63.18	0.00	0.70	62.97
	(-0.03)	(1.41)	(63.13)	(-0.02)	(1.13)	(62.58)	(0.00)	(1.34)	(62.05)
Puppy	-0.04	0.16	63.37	-0.04	2.79	63.64	0.00	2.13	62.66
	(-0.03)	(0.05)	(62.20)	(-0.05)	(2.97)	(63.06)	(0.00)	(1.41)	(62.93)

(b)

sequence	QP=16 ΔPSNR (dB)	QP=16 Δbitrate (%)	QP=16 TS (%)	QP=28 ΔPSNR (dB)	QP=28 Δbitrate (%)	QP=28 TS (%)	QP=40 ΔPSNR (dB)	QP=40 Δbitrate (%)	QP=40 TS (%)
Ballroom	-0.11	1.33	23.03	-0.08	6.79	74.40	-0.08	4.66	73.65
	(-0.11)	(1.51)	(23.51)	(-0.08)	(7.43)	(74.35)	(-0.08)	(4.96)	(74.48)
Exit	-0.14	-0.21	23.08	-0.04	4.93	74.03	-0.05	1.96	73.89
	(-0.14)	(-0.05)	(24.93)	(-0.05)	(5.48)	(75.11)	(-0.05)	(2.13)	(73.59)
Vassar	-0.04	-0.29	22.77	-0.01	0.50	74.04	-0.01	0.61	73.70
	(-0.05)	(-0.28)	(23.79)	(-0.01)	(0.81)	(75.45)	(-0.01)	(0.78)	(73.76)
Soccer2	-0.10	0.08	23.47	-0.05	4.25	73.99	-0.05	1.04	73.42
	(-0.11)	(0.38)	(23.84)	(-0.06)	(4.34)	(74.66)	(-0.06)	(1.37)	(74.91)
Race1	-0.06	1.73	23.28	-0.04	1.68	73.92	-0.03	-1.53	73.72
	(-0.07)	(1.59)	(23.77)	(-0.04)	(0.64)	(74.69)	(-0.05)	(0.75)	(74.99)
Puppy	-0.05	-0.06	22.67	-0.01	0.11	74.20	-0.01	0.02	73.50
	(-0.05)	(0.01)	(23.45)	(-0.01)	(0.23)	(74.92)	(-0.01)	(-0.28)	(74.95)

(c)

sequence	QP=16 ΔPSNR (dB)	QP=16 Δbitrate (%)	QP=16 TS (%)	QP=28 ΔPSNR (dB)	QP=28 Δbitrate (%)	QP=28 TS (%)	QP=40 ΔPSNR (dB)	QP=40 Δbitrate (%)	QP=40 TS (%)
Ballroom	-0.03	0.71	40.47	-0.06	3.94	70.04	-0.07	4.12	75.08
	(-0.02)	(1.03)	(27.17)	(-0.06)	(5.07)	(71.48)	(-0.07)	(3.87)	(73.85)
Exit	-0.01	1.24	35.02	-0.02	1.06	71.67	-0.04	1.22	77.21
	(-0.05)	(0.34)	(28.88)	(-0.03)	(4.34)	(72.75)	(-0.03)	(1.89)	(73.42)
Vassar	-0.00	1.46	21.75	-0.01	-0.01	70.99	-0.01	-0.23	77.13
	(-0.03)	(0.22)	(24.74)	(-0.01)	(1.71)	(73.41)	(-0.01)	(0.19)	(73.72)
Soccer2	-0.01	1.92	24.46	-0.04	0.90	68.52	-0.05	0.17	75.32
	(-0.01)	(1.94)	(23.44)	(-0.05)	(1.04)	(69.23)	(-0.04)	(-0.14)	(74.17)
Race1	-0.01	3.72	28.56	-0.04	2.37	68.71	-0.04	0.21	73.51
	(-0.01)	(3.27)	(30.77)	(-0.06)	(3.72)	(69.94)	(-0.07)	(1.33)	(73.79)
Puppy	-0.01	2.43	23.25	-0.01	0.39	69.83	-0.02	0.56	76.76
	(-0.05)	(0.11)	(21.72)	(-0.01)	(5.87)	(71.96)	(-0.01)	(-0.27)	(73.94)

(d)

sequence	QP=16 ΔPSNR (dB)	QP=16 Δbitrate (%)	QP=16 TS (%)	QP=16 Avg. K_1	QP=28 ΔPSNR (dB)	QP=28 Δbitrate (%)	QP=28 TS(%)	QP=28 Avg. K_1	QP=40 ΔPSNR (dB)	QP=40 Δbitrate (%)	QP=40 TS (%)	QP=40 Avg. K_1
Ballroom	-0.03	0.71	40.28	2.93	-0.07	4.47	78.55	2.24	-0.08	5.10	80.17	2.27
	(-0.02)	(1.03)	(27.26)	(2.99)	(-0.06)	(5.08)	(71.55)	(2.99)	(-0.09)	(4.34)	(82.09)	(2.22)
Exit	-0.01	1.25	34.13	2.92	-0.04	0.76	81.02	2.11	-0.05	1.46	82.85	2.11
	(-0.05)	(0.34)	(28.91)	(2.99)	(-0.03)	(4.34)	(72.80)	(2.98)	(-0.04)	(1.68)	(83.01)	(2.09)
Vassar	0.00	1.50	23.40	2.95	-0.02	-0.10	81.63	2.07	-0.02	-0.88	82.05	2.21
	(-0.03)	(0.22)	(24.91)	(2.99)	(-0.01)	(1.71)	(73.60)	(2.99)	(-0.02)	(-0.09)	(83.69)	(2.09)
Soccer2	-0.01	1.93	25.18	2.95	-0.05	0.87	74.20	2.36	-0.05	0.51	80.47	2.31
	(-0.01)	(1.94)	(23.58)	(2.98)	(-0.05)	(1.04)	(69.42)	(2.99)	(-0.05)	(-0.25)	(81.80)	(2.24)
Race1	-0.01	3.74	28.75	2.90	-0.02	2.58	73.51	2.44	-0.05	0.79	78.20	2.48
	(-0.01)	(3.27)	(30.82)	(2.99)	(-0.06)	(3.99)	(69.95)	(2.99)	(-0.08)	(2.38)	(80.79)	(2.33)
Puppy	-0.01	2.86	27.54	2.93	-0.02	0.61	79.34	2.05	-0.03	0.64	80.69	2.35
	(-0.05)	(0.10)	(24.49)	(2.96)	(-0.01)	(5.87)	(72.22)	(2.99)	(-0.01)	(-0.21)	(82.77)	(2.19)

(e)

sequence	QP=16 ΔPSNR (dB)	QP=16 Δbitrate (%)	QP=16 TS (%)	QP=28 ΔPSNR (dB)	QP=28 Δbitrate (%)	QP=28 TS (%)	QP=40 ΔPSNR (dB)	QP=40 Δbitrate (%)	QP=40 TS (%)
Ballroom	0.00	0.02	32.31	-0.01	0.48	43.36	-0.02	0.31	61.11
	(-0.01)	(0.13)	(39.24)	(-0.03)	(0.49)	(57.03)	(-0.03)	(0.12)	(55.88)
Exit	0.00	0.12	31.48	-0.01	0.48	47.55	-0.01	0.16	71.85
	(0.00)	(0.35)	(39.84)	(-0.04)	(0.40)	(66.58)	(-0.03)	(-0.27)	(67.49)
Vassar	0.00	0.01	32.54	0.00	0.00	48.82	0.00	-0.63	76.81
	(0.00)	(0.03)	(36.38)	(-0.03)	(0.16)	(67.86)	(-0.02)	(-2.61)	(71.49)
Soccer2	0.00	0.73	28.18	-0.01	1.36	32.18	-0.02	1.48	58.30
	(-0.01)	(1.33)	(37.31)	(-0.03)	(1.02)	(41.99)	(-0.02)	(0.66)	(53.23)
Race1	-0.01	0.73	23.02	-0.02	0.98	29.46	-0.02	-1.11	38.75
	(-0.02)	(1.87)	(37.33)	(-0.02)	(1.66)	(31.77)	(-0.03)	(1.14)	(35.87)
Puppy	0.00	0.12	33.86	0.00	0.09	54.15	-0.01	-0.38	75.66
	(-0.01)	(0.22)	(40.06)	(-0.08)	(0.11)	(71.32)	(-0.03)	(-1.48)	(69.74)

In addition to the data shown in Table III(c) and (d), Fig. 10 plots time saving versus bit rate increase at various K_1 ($K_1 \in \{2, 3, 4, 5\}$) for the proposed NN1 and for the "probability-based" method. The test sequence is "Exit" coded at QP $= 28$ with GOP $= 12$. It can be derived from the figure that the optimal K_1 setting (i.e., the K that results

TABLE IV
HIT RATE PERFORMANCE OF THE PROPOSED NN1 FOR VIDEOS CODED WITH GOP =12 AND 16 (NUMBERS WITH PARENTHESIS)

sequence	QP=16		QP=28		QP=40	
	Top 1	Top 3	Top 1	Top 3	Top 1	Top 3
Ballroom	31% (29%)	66% (65%)	58% (57%)	85% (85%)	76% (78%)	93% (93%)
Exit	25% (27%)	61% (65%)	72% (61%)	92% (91%)	88% (91%)	99% (99%)
Vassar	33% (34%)	73% (74%)	79% (68%)	96% (94%)	89% (92%)	98% (99%)
Soccer2	23% (23%)	60% (60%)	53% (50%)	87% (76%)	77% (80%)	95% (97%)
Race1	21% (20%)	52% (53%)	54% (64%)	86% (98%)	71% (69%)	94% (93%)
Puppy	30% (31%)	64% (65%)	88% (77%)	99% (92%)	88% (83%)	99% (99%)

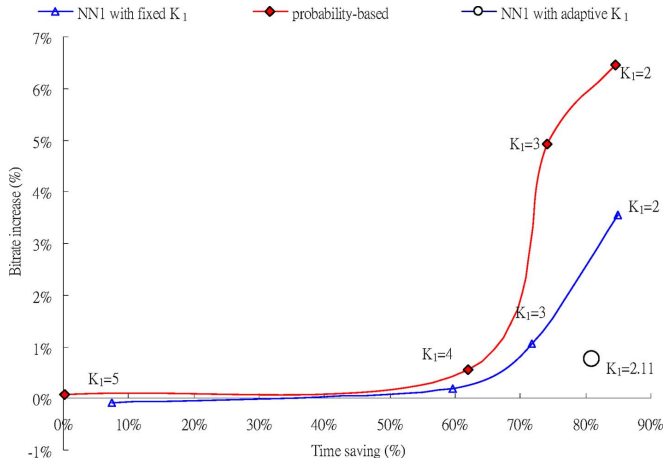

Fig. 10. Time saving versus bitrate increase at varying K_1 values for the "Exit" sequence coded at $\mathrm{QP} = 28$ and $\mathrm{GOP} = 12$.

in high time saving and few bit rate increase) for NN1 and "probability-based" is 3 and 4, respectively. On the other hand, NN1 with adaptive K_1 (average $K_1 = 2.11$) has a better performance than NN1 with fixed $K_1 = 3$.

To evaluate the reliability of the proposed NN1, Table IV shows the hit rate statistics, where Top 1/Top 3 denotes the percentage that the true block partition (obtained via full-mode search) occurs in the list of top 1/top 3 output neurons. It is found that the average top-3 hit rate is more than 90% at QP =28, 40. Although the average hit rate is lower at QP =16, it is still above 63%. Note that the hit rate is in principle subject to mode error propagation between MBs.

The details of bit rate and speed-up performances of the proposed NN1 are presented in Table III(c) and (d) for fixed $K_1 = 3$ and adaptive K_1, respectively. It is found that the adaptive K_1 method gains a higher time saving than that of fixed $K_1 = 3$ method since the resulting K_1 on average is less than 3. Both kinds of NN1 schemes outperform "8×8 off" in time reduction, especially at large QP values. In particular, the adaptive-K_1 method has 81.5% of time saving on average at $\mathrm{QP} = 40$, at a cost of 1.29% of bit rate increase on average. However, NN1 does not guarantee the same performance at small QPs. It achieves only 28.3% of time saving when QP is 16 and the bit rate increase is no more ignorable (1.57% on average), especially for "Race1" sequence which contains high camera motion.

From Table III(e), up to 76% of time savings can be achieved by NN2, which is less than that (up to 83.7%) of NN1.

B. Coding Performance of Hierarchical Two-Stage Processing

The experimental result of NN1 (adaptive-K_1) + NN2 is shown in Table V(a). It shows that up to 93% of time reduction can be achieved and the bit rate increase is below 6.1% for large QPs. Integrating NN2 with NN1 is capable of further gaining 5%–12% of time saving, at an expense of extra bit-rate increase (0.71% on average), for moderate and high bit rate scenarios. Note that up to 59% of time reduction can be observed when QP is 16.

Table V(b) shows the performance of integrating "8×8 off" and NN2. It achieves 69%–88% of time saving over three different QP values, with a bit rate increase of less than 3%. Comparing Table V(a) and (b), it is illustrated that the proposed NN1+NN2 scheme yields better results in terms of larger time reduction and acceptable bit rate increase at large QPs. However, the performance of "8×8 off" +NN2 seems to be better for high bit rate scenario (i.e., at small QPs).

C. Coding Performance of the Proposed Two-Stages Scheme Combining With Fast Motion Estimation

Table VI(a) shows the performance of the overall scheme NN1 (adaptive-K_1)+NN2+FME. It yields up to 80% of time saving even at $\mathrm{QP} = 16$ and up to 97% of time saving when QP is 40, at a cost of bit rate increase up to 6.04%. Table VI(b) shows the performance of the combined scheme: "8×8 off"+NN2+FME. It comes to the same conclusion that the performance of "8×8 off"+NN2+FME is dominant at small QPs, while our proposed NN1(adaptive-K_1)+NN2+FME is better at moderate-to-high QPs. By observing that the Direct mode is frequently chosen, the NN1 scheme is modified to always select it as one of the candidates. Table VI(c) shows that this modification effectively suppresses bit rate increase, while maintaining similar time savings as those in Table VI(a).

TABLE V
PERFORMANCE ON PSNR, BIT RATE, AND TIME SAVING (TS) FOR TWO-STAGE CODING SCHEMES AT QP =16, 28, AND 40 AND GOP = 12 AND 16 (NUMBERS WITH PARENTHESIS). (a) NN1 (ADAPTIVE-K_1) + NN2. (b) "8 × 8 OFF" + NN2

(a)

sequence	QP=16			QP=28			QP=40		
	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)
Ballroom	-0.03 (-0.02)	0.79 (1.38)	59.24 (57.11)	-0.08 (-0.09)	5.53 (5.96)	83.58 (84.04)	-0.11 (-0.13)	5.75 (6.09)	89.50 (89.01)
Exit	-0.01 (-0.06)	1.38 (0.61)	55.95 (56.61)	-0.05 (-0.02)	2.18 (4.87)	86.28 (87.82)	-0.06 (-0.07)	1.57 (1.53)	92.28 (91.44)
Vassar	0.00 (-0.04)	1.53 (0.29)	48.93 (52.77)	-0.02 (-0.08)	0.14 (1.77)	86.87 (88.46)	-0.02 (-0.04)	-1.30 (-1.92)	93.15 (92.47)
Soccer2	-0.01 (-0.02)	2.77 (3.27)	45.75 (52.92)	-0.08 (-0.09)	4.49 (3.25)	81.00 (78.18)	-0.08 (-0.08)	2.47 (1.72)	88.39 (88.45)
Race1	-0.01 (-0.02)	4.41 (5.17)	44.11 (56.24)	-0.07 (-0.08)	4.54 (4.53)	78.06 (77.56)	-0.10 (-0.12)	1.83 (3.23)	83.35 (85.38)
Puppy	-0.01 (-0.05)	2.98 (0.35)	54.02 (56.50)	-0.02 (-0.11)	0.63 (6.11)	86.55 (88.61)	-0.04 (-0.04)	0.17 (-1.34)	92.66 (91.60)

(b)

sequence	QP=16			QP=28			QP=40		
	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)
Ballroom	-0.04 (-0.05)	0.02 (0.25)	72.41 (74.80)	-0.03 (-0.05)	1.67 (1.75)	75.35 (80.95)	-0.03 (-0.04)	0.60 (0.22)	83.04 (80.40)
Exit	-0.03 (-0.03)	0.41 (0.62)	72.62 (73.24)	-0.02 (-0.06)	1.33 (0.97)	76.53 (84.83)	-0.02 (-0.03)	0.04 (-0.11)	86.83 (84.32)
Vassar	-0.06 (-0.06)	-0.28 (-0.22)	72.79 (74.65)	-0.01 (-0.04)	0.03 (0.17)	76.39 (85.04)	-0.01 (-0.02)	-0.81 (-2.62)	88.64 (86.24)
Soccer2	-0.04 (-0.04)	0.66 (1.20)	70.66 (73.28)	-0.04 (-0.05)	2.39 (1.92)	73.67 (75.63)	-0.02 (-0.03)	1.40 (0.58)	81.19 (80.21)
Race1	-0.03 (-0.04)	1.60 (2.51)	68.63 (73.46)	-0.04 (-0.04)	1.91 (2.48)	71.59 (72.60)	-0.03 (-0.04)	0.08 (1.53)	73.71 (73.94)
Puppy	-0.03 (-0.03)	-0.15 (-0.05)	73.37 (74.36)	0.00 (-0.08)	0.10 (0.05)	77.77 (86.41)	-0.01 (-0.03)	-0.41 (-1.53)	88.08 (84.47)

D. Performance Comparison With Two Prior Works [21], [23]

Two prior works [21], [23] (their brief descriptions can be seen in Section I) are implemented and compared with our proposed algorithm (i.e., "8 × 8 off"+ NN2+ FME at QP = 16 and NN1 (adaptive-K_1)+NN2+FM at QP = 28,40). To make a fair comparison, fast motion estimation as adopted in JMVM 8.0 is also carried out for [23]. Table VII presents the experiment results with GOP = 12; all performance figures are calculated with respect to the JMVM full-mode encoding. It shows that [21] and [23]+FME achieve, respectively, up to 67.05% and 95.51% of time saving, while the bit rate increase is up to 5.5% and 2.56%, respectively.

We also investigate the possibility of integrating [23] with FME and our NN2 for prediction source selection (the last column of Table VII). The encoding process can be further speeded-up. Comparing this integrated method with our proposed one, it is interesting to find that they have a tradeoff between time saving and bit rate increase.

Additionally, a qualitative comparison between the proposed, [21], and [23] is summarized as follows.

1) Strategy of block partition:
   Proposed: candidate partitions are determined according to the output of neural network 1.
   [21]: search in disparity direction is fully examined, while search in temporal directions is accomplished by finding a twin MB in the disparity frame and copying the associated MB mode and MVs.
   [23]: sequential evaluation of most probable modes and early termination via thresholding on RD costs are realized.
2) Strategy of prediction source selection:
   Proposed: candidate sources are determined according to the output of neural network 2.
   [21]–[23]: All prediction sources are examined.
3) Pros:
   Proposed: time-saving performance is good (from 84%–97%).
   [21]: complexity in motion estimation is substantially reduced.
   [23]: time saving is moderate and bit rate increase is less.

TABLE VI
PERFORMANCE ON PSNR, BIT RATE, AND TIME SAVING (TS) FOR TWO-STAGE CODING SCHEMES COMBINING FME, AT QP = 16, 28, AND 40 AND GOP = 12 AND 16 (NUMBERS WITH PARENTHESIS. (a) NN1(ADAPTIVE-K_1)+NN2+FME. (b) "8 × 8 OFF"+NN2+ FME. (c) NN1(ADAPTIVE-K_1)+NN2+FME (WITH DIRECT MODE A MUST CANDIDATE)

(a)

sequence	QP=16			QP=28			QP=40		
	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)
Ballroom	-0.03 (-0.02)	0.84 (1.41)	79.06 (78.42)	-0.08 (-0.09)	4.75 (5.63)	92.26 (92.60)	-0.10 (-0.12)	4.48 (4.28)	95.00 (95.11)
Exit	-0.01 (-0.06)	1.67 (0.89)	77.25 (77.28)	-0.05 (-0.07)	2.29 (5.64)	93.79 (94.64)	-0.09 (-0.07)	1.85 (0.59)	96.49 (96.41)
Vassar	-0.01 (-0.04)	1.58 (0.34)	74.73 (76.56)	-0.02 (-0.05)	-0.35 (1.49)	94.17 (95.24)	-0.03 (-0.03)	-2.99 (-2.92)	97.26 (97.09)
Soccer2	-0.01 (-0.02)	2.80 (3.46)	72.34 (76.39)	-0.08 (-0.10)	3.77 (3.07)	91.14 (90.48)	-0.08 (-0.09)	2.50 (1.77)	94.81 (95.12)
Race1	-0.02 (-0.03)	4.72 (5.77)	71.15 (78.69)	-0.07 (-0.07)	4.09 (4.41)	89.33 (88.87)	-0.08 (-0.13)	0.15 (4.03)	92.18 (93.26)
Puppy	-0.02 (-0.05)	2.97 (0.35)	78.92 (80.03)	-0.02 (-0.11)	0.87 (6.04)	94.14 (95.57)	-0.05 (-0.04)	1.40 (-0.19)	97.04 (96.10)

(b)

sequence	QP=16			QP=28			QP=40		
	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)
Ballroom	-0.04 (-0.05)	0.05 (0.32)	85.19 (87.02)	-0.03 (-0.05)	1.76 (1.70)	88.57 (90.91)	-0.02 (-0.04)	0.00 (-0.38)	92.35 (91.09)
Exit	-0.03 (-0.03)	0.52 (0.69)	85.03 (86.09)	-0.02 (-0.05)	1.51 (2.21)	89.56 (93.34)	-0.05 (-0.03)	-0.59 (-0.30)	94.57 (93.42)
Vassar	-0.06 (-0.06)	-2.06 (-0.20)	86.07 (87.25)	-0.01 (-0.04)	-0.22 (-0.15)	89.81 (93.91)	-0.01 (-0.02)	-2.75 (-3.55)	95.56 (94.31)
Soccer2	-0.04 (-0.04)	0.74 (1.27)	84.83 (85.59)	-0.05 (-0.05)	2.38 (1.63)	88.17 (88.94)	-0.03 (-0.03)	0.99 (0.13)	91.55 (91.33)
Race1	-0.03 (-0.04)	2.05 (2.58)	83.92 (86.23)	-0.04 (-0.05)	3.14 (2.68)	85.98 (86.44)	-0.05 (-0.05)	1.37 (3.50)	87.79 (87.53)
Puppy	-0.03 (-0.03)	-0.18 (-0.11)	87.00 (88.13)	-0.01 (-0.09)	0.09 (0.24)	90.66 (94.51)	-0.03 (-0.03)	0.58 (-0.17)	95.27 (93.93)

(c)

sequence	QP=16			QP=28			QP=40		
	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)	ΔPSNR (dB)	Δbitrate (%)	TS (%)
Ballroom	-0.04 (-0.03)	0.30 (1.03)	78.66 (77.97)	-0.08 (-0.09)	4.71 (5.67)	92.29 (93.14)	-0.10 (-0.13)	4.39 (4.26)	94.97 (95.01)
Exit	-0.02 (-0.07)	0.92 (0.60)	76.86 (77.25)	-0.05 (-0.08)	2.08 (3.86)	93.58 (95.25)	-0.09 (-0.07)	1.67 (0.45)	96.54 (96.52)
Vassar	-0.03 (-0.04)	0.12 (0.03)	74.09 (76.53)	-0.02 (-0.05)	-0.46 (0.23)	94.10 (95.95)	-0.03 (-0.03)	-3.21 (-3.35)	97.19 (96.99)
Soccer2	-0.01 (-0.03)	1.77 (2.35)	72.55 (76.24)	-0.08 (-0.10)	3.41 (3.86)	91.12 (91.33)	-0.08 (-0.09)	2.49 (1.75)	94.77 (94.81)
Race1	-0.02 (-0.03)	3.08 (4.39)	70.58 (78.20)	-0.07 (-0.08)	3.63 (4.30)	88.87 (89.77)	-0.08 (-0.13)	0.05 (3.88)	92.11 (93.29)
Puppy	-0.04 (-0.05)	0.66 (0.32)	78.04 (80.32)	-0.02 (-0.10)	0.44 (0.78)	94.32 (96.21)	-0.05 (-0.04)	0.89 (-0.33)	97.09 (96.99)

4) Cons:

Proposed: bit rate increase is slightly higher than [23]-based methods and neural network trainings for different coding settings are needed.

[21]: time saving is limited due to full search in disparity direction.

[23]: determination of multi-thresholds is experimentally heuristic and time-consuming. No fast algorithm is provided for prediction source selection.

TABLE VII
PERFORMANCE COMPARISON BETWEEN THE PROPOSED, [21], [23]+FME, AND [23]+NN2+FME. THE GOP SIZE DURING ENCODING IS 12

sequence		Proposed			[21]			[23] + FME			[23]+ NN2+ FME		
		QP=16	QP=28	QP=40	QP=16	QP=28	QP=40	QP=16	QP=28	QP=40	QP=16	QP=28	QP=40
Ballroom	ΔPSNR (dB)	-0.04	-0.08	-0.10	-0.12	-0.09	-0.21	-0.12	-0.08	-0.01	-0.12	-0.09	-0.04
	Δbitrate (%)	0.05	4.75	4.48	0.58	5.50	1.92	-0.35	-0.04	0.56	-0.13	0.70	0.48
	TS (%)	85.19	92.26	95.00	47.18	56.90	57.85	79.95	87.72	88.33	84.23	90.26	92.03
Exit	ΔPSNR (dB)	-0.03	-0.05	-0.09	-0.07	-0.13	-0.34	-0.14	-0.05	-0.03	-0.15	-0.05	-0.05
	Δbitrate (%)	0.52	2.29	1.85	0.91	4.29	6.62	-0.26	-1.23	0.28	-0.11	-0.14	-0.48
	TS (%)	85.03	93.79	96.49	67.05	66.17	56.02	74.08	88.93	91.41	80.39	92.34	95.41
Vassar	ΔPSNR (dB)	-0.06	-0.02	-0.03	-0.09	-0.01	0.00	-0.09	-0.04	0.00	-0.08	-0.04	-0.01
	Δbitrate (%)	-0.26	-0.35	-2.99	0.3	0.37	0.68	-0.64	-1.16	-0.81	-0.59	-1.19	-2.90
	TS (%)	86.07	94.17	97.26	46.14	45.11	45.25	79.83	90.71	89.48	84.94	93.95	96.46
Soccer2	ΔPSNR (dB)	-0.04	-0.08	-0.08	-0.04	-0.07	-0.04	-0.11	-0.07	-0.03	-0.12	-0.09	-0.05
	Δbitrate (%)	0.74	3.77	2.50	1.32	3.11	-0.15	-0.15	-1.43	-0.44	-0.25	0.92	1.33
	TS (%)	84.83	91.14	94.81	66.56	65.91	65.65	78.50	85.97	86.27	80.59	88.47	90.16
Race1	ΔPSNR (dB)	-0.03	-0.07	-0.08	-0.05	-0.08	-0.11	-0.10	-0.13	-0.06	-0.10	-0.15	-0.09
	Δbitrate (%)	2.05	4.09	0.15	3.80	4.96	0.09	2.37	2.56	-3.24	2.72	3.42	-2.13
	TS (%)	83.92	89.33	92.18	63.76	64.43	65.65	82.70	88.09	89.78	82.66	87.34	90.15
Puppy	ΔPSNR (dB)	-0.03	-0.02	-0.05	-0.05	-0.01	-0.01	-0.11	-0.07	-0.01	-0.11	-0.07	-0.03
	Δbitrate (%)	-0.18	0.87	1.40	0.00	-0.34	-0.38	-0.05	-1.73	1.73	0.09	-1.50	0.40
	TS (%)	87.00	94.34	97.04	65.47	65.25	65.41	88.22	95.51	91.49	91.26	96.64	96.76

VI. CONCLUDING REMARKS

This research is motivated by the huge amount of modes that will be encountered for H.264/AVC-based stereo video coding, due to an additional prediction source from the disparity domain for hierarchical B-frames. Unfortunately, traditional algorithms for speeding up H.264/AVC mode decision cannot work in this case. In this work, we propose a fast stereo video encoding algorithm based on hierarchical two-stage neural classification, including fast block partition selection and fast prediction source determination. By changing the threshold parameters in NN1 and NN2, scalable complexity and speed-up can be easily achieved.

Experiment results verify that our proposed NN1+NN2+FME scheme is capable of achieving up to 97% of time reduction at moderate-to-low bit rates, at an expense of only a slight bit rate increase (up to 5.67%). For high bit rate scenario, "8×8 off"+NN2+FME scheme can be adopted instead to maintain the performance.

Our algorithm is currently simulated on stereo videos, targeting on 3-D TV broadcast application in a near future. However, the proposed speeding algorithm can be also applicable to multiple-view coding once the reference structure of each frame is determined (thus, prediction sources at the second stage should be re-defined).

REFERENCES

[1] C. Fehn, "A 3D-TV system based on video plus depth information," in *Proc. 37th Asilomar Conf. Signals, Syst., Comput.*, Nov. 2003, vol. 2, pp. 1529–1533.

[2] "MPEG-2 Multi-View Profile," 1996, ISO/IEC IS 13818-2 AMD3.

[3] J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 7, pp. 787–800, Jul. 2003.

[4] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 519–528.

[5] E. Martinian, A. Behrens, J. Xin, A. Vetro, and H. Sun, "Extension of H.264/AVC for multi-view video compression," in *Proc. IEEE Int. Conf. Image Process.*, 2006, pp. 2981–2984.

[6] P. Merkle, K. Müller, A. Smolic, and T. Wiegand, "Efficient compression of multi-view video exploiting inter-view dependencies based on H.264/MPEG4-AVC," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2006, pp. 1717–1720.

[7] R. S. Wang and Y. Wang, "Multi-view video sequence analysis, compression, and virtual viewpoint synthesis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 3, pp. 397–410, Apr. 2000.

[8] "Requirements on multi-view video coding v.2," Poznan, Poland, 2005, ISO/IEC JTC1/SC29/WG11, Doc. N7282.

[9] "Update call for proposal on multi-view video coding," Nice, France, 2005, ISO/IEC JTC1/SC29/WG11, Doc. N7567.

[10] "Joint multi-view video model (JMVM) 8.0," Geneva, Switzerland, 2008, ISO/IEC JTC1/SC29/WG11/JVT-AA207.

[11] "WD 4 reference software for MVC," Geneva, Switzerland, 2009, ISO/IEC JTC1/SC29/WG11/JVT-AD207.

[12] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 688–703, Jul. 2003.

[13] T. Wiegand, G. J. 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, Jul. 2003.

[14] F. Pan, X. Lin, S. Rahardja, K. P. Lim, Z. G. Li, D. Wu, and S. Wu, "Fast mode decision algorithm for intra prediction in H.264/AVC video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 7, pp. 813–822, Jul. 2005.

[15] C. Grecos and M. Yang, "Fast mode prediction for the baseline and main profile in the H.264 video coding standard," *IEEE Trans. Multimedia*, vol. 8, no. 6, pp. 1125–1134, Dec. 2006.
[16] D. Wu, F. Pan, K. P. Lim, S. Wu, Z. G. Li, X. Lin, S. Rahaedja, and C. C. Kuo, "Fast inter-mode decision in H.264/AVC video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 7, pp. 953–958, Jul. 2005.
[17] L.-J. Pan and Y.-S. Ho, "Fast mode decision algorithm for H.264 inter-prediction," *Electron. Lett.*, vol. 43, no. 24, pp. 1351–1353, Nov. 2007.
[18] Y. Kim, J. H. Lee, C. Park, and K. Sohn, "MPEG-4 compatible stereoscopic sequence codec for stereo broadcasting," *IEEE Trans. Consumer Electron.*, vol. 51, no. 4, pp. 1227–1236, Nov. 2005.
[19] L.-F. Ding, S.-Y. Chien, Y.-W. Huang, Y.-L. Chang, and L.-G. Chen, "Stereo video coding system with hybrid coding based on joint prediction scheme," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2005, vol. 6, pp. 6082–6085.
[20] P. Lai and A. Ortega, "Predictive fast motion/disparity search for multi-view video coding," *Proc. SPIE*, vol. 6077, Vis. Commun. and Image Process., 2006.
[21] L.-F. Ding, T.-K. Tsung, S.-Y. Chien, W.-Y. Chen, and L.-G. Chen, "Content-aware prediction algorithm with inter-view mode decision for multi-view video coding," *IEEE Trans. Multimedia*, vol. 10, no. 8, pp. 1553–1564, Aug. 2008.
[22] S.-Y. Lee, K.-M. Shin, and K.-D. Chung, "An object-based mode decision algorithm for multi-view coding," in *Proc. IEEE Int. Symp. Multimedia*, 2008, pp. 74–81.
[23] Z. Peng, G. Jiang, and M. Yu, "A fast multiview video coding algorithm based on dynamic multi-threshold," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2009, pp. 113–116.
[24] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley-Interscience, 2001.

Jui-Chiu Chiang was born in Keelung, Taiwan, in 1973. She received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1997, the M.S. degree in electronic engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1999, and the Ph.D. degree in automatic and signal processing from Paris-Sud University, Orsay, France, in 2004.

Since 2005, she has been with the Department of Electrical Engineering, National Chung Cheng University, Chia-Yi, Taiwan, as an Assistant Professor. Her current research interests include image coding, video coding, distributed video coding, scalable video coding, and multi-view video coding.

Wei-Chih Chen was born in Taipei, Taiwan, in 1984. He received the M.S. degree in electrical engineering from National Chung Cheng University, Chia-Yi, Taiwan, in 2008. He is currently pursuing the Ph.D. degree in the Department of Electrical Engineering, National Chung Cheng University.

His research interests include digital image processing, video coding, and 3-D TV.

Lien-Ming Liu was born in Yunlin, Taiwan, in 1982. He received the M.S. degree in electrical engineering from National Chung Cheng University, Chia-Yi, Taiwan, in 2008.

He is currently an Engineer with ASMedia Technology, Inc., Taipei, Taiwan. His research interests include digital image processing, pattern recognition, and 3-D TV.

Kuo-Feng Hsu was born in Taoyuan, Taiwan, in 1986. He received the M.S. degree in electrical engineering from National Chung Cheng University, Chia-Yi, Taiwan, in 2010.

His research interests include digital image processing, video coding, and 3-D TV.

Wen-Nung Lie was born in Tainan, Taiwan, in 1962. He received the B.S., M.S., and Ph.D. degrees in electrical engineering, from National Tsing Hua University, Hsinchu, Taiwan, in 1984, 1986, and 1990, respectively.

From September 1990 to June 1996, he served as an Assistant Scientist at the Chung Shan Institute of Science and Technology (CSIST), Taiwan, where he was devoted to the development of infrared imaging system and target tracker for military applications. In August 1996, he joined the Department of Electrical Engineering, National Chung Cheng University, Chia-Yi, Taiwan, where he is currently a Professor and the Deputy Head of the Office of R&D. From July 2000 to January 2001, he was a Visiting Scholar at the University of Washington, Seattle. His research interests include image/video compression, networked video transmission, audio/image/video watermarking, multimedia content analysis, standard-compliant multimedia encryption, infrared image processing, 3-D TV related, 2-D-to-3-D image/video conversion, and industrial inspection by computer vision techniques.