Low-Complexity Heterogeneous Video Transcoding Using Data Mining

Gerardo Fernández-Escribano, *Student Member, IEEE*, Jens Bialkowski, Jose A. Gámez, Hari Kalva, *Senior Member, IEEE*, Pedro Cuenca, *Member, IEEE*, Luis Orozco-Barbosa, *Member, IEEE*, and André Kaup, *Senior Member, IEEE*

***Abstract*—Recent developments have given birth to H.264/AVC: a video coding standard offering better bandwidth to video quality ratios than previous standards (such as H.263, MPEG-2, MPEG-4, etc.), due to its improved inter- and intraprediction modes at the expense of higher computation complexity. It is expected that H.264/AVC will take over the digital video market, replacing the use of previous standards in most digital video applications. This creates an important need for heterogeneous video transcoding technologies from older standards to H.264. In this paper, we focus our attention on the interframe prediction, the most computationally intensive task involved in the heterogeneous video transcoding process. This paper presents a novel macroblock (MB) mode decision algorithm for interframe prediction based on data mining techniques to be used as part of a very low complexity heterogeneous video transcoder. The proposed approach is based on the hypothesis that MB coding mode decisions in H.264 video have a correlation with the distribution of the motion compensated residual in the decoded video. We use data mining tools to exploit the correlation and derive decision trees to classify the incoming decoded MBs into one of the several coding modes in H.264. The proposed approach reduces the H.264 MB mode computation process into a decision tree lookup with very low complexity. For general validation purposes, we apply our algorithm to two of the most important heterogeneous video transcoders: MPEG-2 to H.264 and H.263 to H.264. Our results show that the our data-mining based transcoding algorithm is able to maintain a good video quality while considerably reducing the computational complexity by 72% on average when applied in MPEG-2 to H.264 transcoders, and by 62% on average when applied in H.263 to H.264 transcoders. Finally, we conduct a comparative study with some of the most prominent fast interprediction methods for H.264 presented in the literature. Our results show that the proposed data mining-based approach achieves the best results for video transcoding applications.**

***Index Terms*—Data mining, H.263, H.264, interprediction, MPEG-2, supervised classification, video transcoding.**

I. Introduction

THE H.264/AVC standard [1] achieves much higher coding efficiency than the H.263 [2], MPEG-2 [3] and MPEG-4 [4] standards, due to its improved inter- and intraprediction modes at the expense of higher computation complexity. Therefore, H.264/AVC is a strong candidate for a wide range of applications in the near future.

Heterogeneous video transcoding, transcoding different video formats such as MPEG-2, H.263, and MPEG-4 to H.264, is essential to enable gradual migration to H.264. For example, the transcoding of H.263 into H.264/AVC bitstreams for UMTS or DVB-H networks will become necessary in the near future. On the other hand, given the wide deployment of MPEG-2 infrastructure, MPEG-2 and H.264 are likely to coexist even as H.264 systems are deployed. The coexistence of these technologies enables systems that can leverage H.263, MPEG-2 and H.264 to provide innovative digital video services making use of H.263 or MPEG-2 for current devices and H.264 for new generation receivers. The key to making this seamless is by heterogeneous video transcoding from previous video coding standards to H.264 at the appropriate points in the video distribution infrastructure. However, given the significant differences between video encoding algorithms, the transcoding process of such systems is a much more complex task than the other heterogeneous video transcoding processes and new approaches to transcoding are necessary [5]–[8].

The main elements that have to be addressed in the design of an efficient heterogeneous video transcoder to H.264 are [9]: the interframe prediction, the transform coding and the intraframe prediction. Each one of these elements requires to be examined and various research efforts are underway [10]–[13]. In this paper, we focus our attention on the interframe prediction, one of the most computationally expensive tasks involved in the heterogeneous video transcoding process to H.264.

A heterogeneous transcoder to H.264 is composed of a video decoder (H.263, MPEG-2, MPEG-4, etc.) and an H.264 encoder interconnected in tandem (i.e., a decoding stage followed by an encoding stage). The idea behind the design of an efficient heterogeneous transcoder to H.264 can be simply stated as follows: the decoding stage should provide the H.264 encoding stage with all the pieces of information that may be used by the latter to speed up the encoding stage. The decoding stage of a transcoder can perform full decoding at the pixel level or partial decoding at the coefficient level. Partial decoding is used in compressed domain transcoding where the transform coefficients in the input format are directly transcoded to the output format. This transformation is straightforward when the input and output formats of the transcoder use the same transform (e.g., MPEG-2 to MPEG-4 transcoding) [7]. When these transforms differ substantially, the compressed domain transcoding becomes computationally expensive. The utility of this compressed domain transcoding is limited to intra-MB transcoding.

Manuscript received February 10, 2007; revised September 11, 2007. This work was supported by the Ministry of Science and Technology of Spain under CONSOLIDER Project CSD2006-46, CICYT Project TIN2006-15516-C04-02, by the Council of Science and Technology of Castilla-La Mancha under Project PAI06-0106, and by FEDER. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Florent Masseglia.

G. Fernández-Escribano, J. A. Gámez, P. Cuenca, and L. Orozco-Barbosa are with the Instituto de Investigación en Informática de Albacete, Universidad de Castilla-La Mancha, 02071 Albacete, Spain (e-mail: gerardo@dsi.uclm.es; jgamez@dsi.uclm.es; pcuenca@dsi.uclm.es; lorozco@dsi.uclm.es).

J. Bialkowski and A. Kaup are with the University of Erlangen-Nuremberg, 91058 Erlangen, Germany (e-mail: bial@lnt.de; kaup@lnt.de).

H. Kalva is with the Department of Computer Science and Engineering, Florida Altantic University, Boca Raton, FL 33431 USA (e-mail: hari@cse.fau.edu).

Digital Object Identifier 10.1109/TMM.2007.911838

For predicted MBs, the transcoding in compressed domain becomes prohibitively expensive. The substantial differences between previous standards and H.264 make even intratranscoding in the compressed domain relatively expensive [10]; pixel domain transcoding is shown to produce better results [11]. Pixel domain transcoders have a full decoding stage followed by a reduced complexity encoding stage. The complexity reduction is achieved by reusing the information gathered from the decoding stage.

The complexity reduction techniques for heterogeneous video transcoding to H.264 reported in the literature fall into two categories: 1) MB mode mapping in H.264 based on the MB modes of the incoming video [12] and 2) selective evaluation of MB modes in H.264 based on heuristics [13]. Because of the large number of inter- and intra-MB coding modes supported by H.264, there is no one-to-one mapping between previous standards and H.264 MB modes. A direct mapping leads to either a sub-optimal decision if the mapped mode is the final MB mode or an increase on complexity if additional evaluations have to be made to improve the mode decision. Selective evaluation is based on the observation that certain MB modes are less likely to occur for a class of videos and bitrates. If the selective evaluation is aggressive in limiting the number of allowed modes, the performance is sub-optimal. On the contrary, increasing the number of allowed modes increases the complexity.

In this paper, we present a novel macroblock (MB) mode decision algorithm for interframe prediction based on data mining techniques to be used as part of a very low complexity heterogeneous video transcoder. Multimedia Data Mining techniques have been developed over the last few years and have been mainly used to analyze or understand multimedia data. This work presents an innovative approach to complexity reduction in video transcoding based on Multimedia Data Mining techniques. The proposed approach is based on the hypothesis that MB coding mode decisions in H.264 video have a correlation with the distribution of the motion compensated residual in the decoded video. Exploiting this correlation together with the MB coding modes of the decoded video could lead to a very low complexity heterogeneous transcoder. Thus, the H.264 MB mode computation problem is posed as a data classification problem where the incoming MB coding mode and residual have to be classified into one of the several H.264 coding modes. The proposed heterogeneous transcoder is developed based on data mining principles and reduces the H.264 MB mode computation process into a decision tree lookup with very low complexity.

The rest of the paper is organized as follows. Section II reviews the principles of operation of the interframe prediction in the H.264 video coding standard. In Section III, we review some of the most relevant proposals aiming to speed-up the interframe prediction process in H.264. Section IV introduces our macroblock mode decision algorithm for interframe prediction based on data mining techniques, specifically designed for heterogeneous video transcoders to H.264. We apply our algorithm to two of the most useful heterogeneous video transcoders: MPEG-2 to H.264 and H.263 to H.264. In Section V, we carry out a performance evaluation of the proposed algorithm in terms of its computational complexity and rate-distortion results. Then, we also analyze a comparative study with some of the most prominent fast interprediction methods for H.264 presented in the literature. Finally, Section VI draws our conclusions.

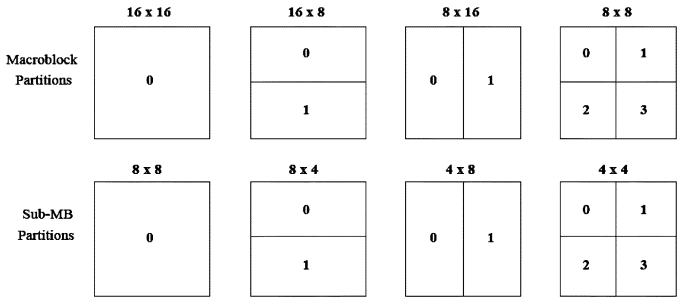

Fig. 1. Macroblock partitions, sub-macroblock partitions and partition scans for intermodes.

II. InterFrame Prediction in H.264 Video Coding Standard

A. Overview

In the H.264 standard, the macroblock mode decision in interframes is the most computationally expensive processes due to the use of tools such as the variable block-size motion estimation, quarter-pixel motion compensation, and intraprediction.

H.264 uses block-based motion compensation, the same principle adopted by every major coding standard since H.261. Important differences from earlier standards include the support for a range of block sizes (down to 4×4) and fine sub-pixel motion vectors (1/4 pixel in the luma component). H.264 supports motion compensation block sizes ranging from 16×16 to 4×4 luminance samples with many options between the two. The luminance component of each macroblock (16×16 samples) may be split up in 4 ways: 16×16, 16×8, 8×16, or 8×8. Each of the sub-divided regions is a macroblock partition. If the 8×8 mode is chosen, each of the four 8×8 macroblock partitions within the macroblock may be further split in 4 ways: 8×8, 8×4, 4×8, or 4×4 (known as sub-macroblock partitions). These partitions and sub-partitions give rise to a large number of possible combinations within each macroblock (see Fig. 1). This method of partitioning macroblocks into motion compensated sub-blocks of varying size is known as *tree structured motion compensation*.

A separate Motion Vector (MV) (previously calculated in the motion estimation module) is required for each partition or sub-partition. Each motion vector must be coded and transmitted; in addition, the choice of partition(s) must be encoded in the compressed bitstream. Choosing a large partition size (e.g., 16×16, 16×8, 8×16) means that a small number of bits are required to signal the choice of motion vector(s) and the type of partition; however, the motion compensated residual may contain a significant amount of energy in areas with high detail. Choosing a small partition size (e.g., 8×4, 4×4, etc.) may give a lower-energy residual after motion compensation but requires a larger

number of bits to signal the motion vectors and choice of partition(s). The partition size used therefore has a significant impact on the compression performance. In general, a large partition size is appropriate for homogeneous areas of the frame and a small partition size may be beneficial for areas with high detail.

The resolution of each chroma component in a macroblock (Cr and Cb) is half that of the luminance (luma) component. Each chroma block is partitioned in the same way as the luma component, except that the partition sizes have exactly half the horizontal and vertical resolution (an 8×16 partition in luma corresponds to a 4×8 partition in chroma; an 8×4 partition in luma corresponds to 4×2 in chroma; and so on). The horizontal and vertical components of each motion vector (one per partition) are halved when applied to the chroma blocks.

Each partition in an intercoded macroblock is predicted from an area of the same size in a reference picture. The offset between the two areas (the motion vector) has 1/4-pixel resolution (for the luma component). If the video source sampling is $4:2:0$, 1/8 pixel samples are required in the chroma components (corresponding to 1/4-pixel samples in the luma). The luma and chroma samples at sub-pixel positions do not exist in the reference picture and so it is necessary to create them using interpolation from nearby image samples. Sub-pixel motion compensation can provide significantly better compression performance than integer-pixel compensation, at the expense of increased complexity. Quarter-pixel accuracy outperforms half-pixel accuracy.

Encoding a motion vector for each partition can take a significant number of bits, especially if small partition sizes are chosen. Motion vectors for neighboring partitions are often highly correlated and so each motion vector is predicted from vectors of nearby, previously coded partitions. The method of forming the prediction MVp depends on the motion compensation partition size and on the availability of nearby vectors.

In addition, H.264 also allows intramodes in interframes. Therefore, although the current macroblock belongs to an interslice, H.264 must examine all intraprediction modes: an MB may make use of 4×4 and 16×16 block prediction modes, referred to as *Intra_4* $\times$ 4 and *Intra_16* $\times$ 16, respectively. Recently, the *Intra_8* $\times$ 8 block prediction mode has been added as part of the Fidelity Range Extension (FREXt) of the standard. There are nine 4×4 and 8×8 possible block prediction directions and four 16×16 block prediction directions. These intraprediction modes include a directional prediction greatly improving the prediction in the presence of directional structures. Finally, H.264 also allows a SKIP mode in interframes referring to the 16×16 mode where no motion and residual information is encoded.

The basic process for interframe prediction in H.264 can be briefly summarized as follows: an encoder determines the coding mode of a MB by evaluating all possible modes (inter, intra, and skip), such as different block sizes motion estimation, different block sizes intraprediction and multiple reference frames. The MB coding mode which produces the least cost will be used in the final coding.

In the H.264 JM reference software (version 10.2) [14], RD-cost has been defined to evaluate the cost for MB mode decision in order to achieve the best *Rate-Distortion* (RD) performance results. The *Rate-Distortion* optimization method is based on a *Lagrange* multiplier [15], [16]. The H.264 reference software implementation has an option to make use of this optimization method to choose the best macroblock mode decision. In this way, the H.264 reference software implementation selects the macroblock mode exhibiting the minimum *Lagrange* cost. This implies that for each existing macroblock partition (sub-partition) within the MB, *bit-rate* and *distortion* are calculated by actually encoding and decoding the video. Therefore, the encoder can achieve the best Rate-Distortion performance results, at the expense of extra complexity.

For evaluating the RD-cost, the H.264 reference software implementation has to obtain the *encoding rate*, $\boldsymbol{R}$, and the *distortion*, $\boldsymbol{D}$, of each macroblock partition (sub-macroblock partition). The former is obtained by first computing the difference between the original macroblock and its predictor. Thereafter, a 4×4 *integer transform* (IT) has to be applied followed by a quantization process. The distortion, $\boldsymbol{D}$, is obtained by performing an inverse quantization process followed by its inverse IT and then comparing the original macroblock to the reconstructed one. The H.264 reference software implementation chooses then the decision mode having the minimum cost, $\boldsymbol{J}$. The cost is evaluated using the *Lagrange* function $J = D + \lambda \times R$, where λ is the *Lagrange* multiplier. One of the main drawbacks of this method is its excessive computational cost. On the contrary, the encoder can achieve the best Rate-Distortion performance results. However, for many applications, the use of the Lagrange multiplier may be prohibitive. This is the case when developing a transcoding architecture aimed to work in real-time.

B. Observations and Motivations

As mentioned in the previous sections, in the current H.264 JM reference software, to encode a given MB in an interframe, the encoder tries all possible prediction modes in the following order; SKIP, Inter16 $\times$ 16, Inter16 $\times$ 8, Inter8 $\times$ 16, Inter8 $\times$ 8, (Inter8 $\times$ 4, Inter4 $\times$ 8, Inter4 $\times$ 4), Intra4 $\times$ 4, Intra8 $\times$ 8, and Intra16 $\times$ 16. It is worth mentioning that each intermode decision requires the full motion estimation process. This implies that for each macroblock partition (sub-macroblock partition) within the MB, motion estimation is done first for all block types and the resulting cost is used for the mode decision. Moreover, although the probability of coding MBs in intramodes is much less than that of intermodes in interframes, the encoder computes the cost of intramode for every MB. This "try all and select the best" philosophy is optimal in deciding the coding mode of MB, but it is achieved at the expense of high computational complexity. The complexity analysis described in [17] shows that examining all possible modes takes the most time out of the total encoding time.

Based on these observations, we present an innovative approach to jointly optimize mode decision and motion estimation in heterogeneous video transcoding applications. With our proposal, the H.264 MB mode computation problem is posed as a data classification problem where the decoded MB coding mode and residual have to be classified into ***one*** of the several H.264 coding modes. Unlike the reference software where MVs are estimated for all intermode blocks types, in our approach, no

motion estimation is required for a particular mode if that mode is not selected by the mode decision algorithm. In the same way, no intraprediction is required for a particular inter-MB if that mode is not selected by the mode decision algorithm. The proposed approach will perform the motion estimation only for the final MB mode determined by the decision tree.

III. Relevant Proposals to Speed-Up the InterFrame Prediction in H.264

Due to the fact that the interframe prediction in H.264 is the most computationally expensive processes, several fast intermode decision algorithms have been proposed in the literature. These algorithms achieve significant time saving with negligible loss of coding efficiency. In the following, we introduce some of the most relevant ones.

Lim *et al.* proposed a fast intermode selection algorithm using the information supplied by the intraprediction mode step and the edge map [18]. Consequently, the process of intraprediction is performed first. However, the small probability of intramodes in interframes in real video sequences (maximum 9% and average of 3%, [19]) suggests that the current practice of deciding the best intramode first and subsequent decision of a intermode may have a certain limit in reducing the computational complexity.

Based on this observation, Jeon *et al.* proposed a new mode decision method based on "selective intramode decision" that investigates intramodes after deciding the best intermode [20]. Furthermore, this algorithm investigates various intramodes only when it is believed to be certainly worthwhile. The proposed algorithm provides considerable reduction in computational complexity only with a small coding loss.

Kim *et al.* proposed an adaptive mode decision algorithm using the property of an all-zero coefficients block that is produced by quantification and coefficient thresholding to effectively eliminate unnecessary intermodes [21]. As a result, the proposed algorithm is two times faster than the H.264 reference software and shows better performance that the others fast algorithms mentioned above [18], [20]. However in this method, there are several threshold values that should be predefined. Furthermore, the transform coefficients must be available to make a fast decision.

Recently, a fast intermode decision algorithm for H.264 video coding has been proposed by Wu *et al.* in [22]. This algorithm makes use of the spatial homogeneity of a video object's textures and the temporal stationary characteristics inherent in video sequences. However, the method suffers from a drawback to obtain an edge image for the texture information and a difference image for the temporal stationarity characteristics.

In [13], a fast mode decision and motion estimation algorithm for H.264 with a focus on MPEG-2/H.264 transcoding has been proposed by Lu *et al.* Their fast motion search is composed of a fast intermode decision for the B- and P-frames, and fast intraprediction for intracoding. In addition, fast motion estimation is developed by reusing the motion information from MPEG-2. In Lu's algorithm, the suggested intermode prediction for SKIP, 16×16, and DIRECT modes used neighboring MBs of the target MB in the current frame.

In [23], Bialkowski *et al.* proposed a fast H.263 to H264 transcoder for the interframes re-using the motion vectors. The H.263 sequence is fully decoded and the H.263 motion vectors are stored. These motion vectors are used as an approximation. The authors use motion vector refinement strategies to improve the transcoding process, but without changing the intermode that was selected for the macroblock in the H.263 encoder. In [24], the authors extend this principle by an approach which additionally includes checking for the intramode as well as skip mode. This is possible because, the modes used in the H.263 are a subset of the modes in the H.264 standard.

Another approach for a H.263 to H.264 transcoder is proposed by Fung and Siu in [25]. They proposed a motion vector decomposition algorithm to improve the transcoder process, instead of using full H.264 motion estimation. The main idea is to process the incoming H.263 motion vectors for a macroblock and obtain new motion vectors if the amount of the residual is bigger than the threshold that they defined in their approach. The definition of the threshold is based on the H.263 decoded DCT coefficients for the macroblock.

Nguyen and Tag in [26] proposed a motion vector re-estimation scheme using vector median filtering and a fast intraprediction selection scheme based on coarse edge information obtained from integer transform coefficients. Besides, a rate control method based on quadratic model is proposed for selecting quantification parameters at the sequence and frame levels.

All these transcoding mechanisms (MPEG-2 to H.264 and H.263 to H.264) are based on the analysis of the motion vectors information and the macroblock information of the incoming sequence to enhance the transcoding process. However, our approach is based on the analysis of residual information using data mining techniques. The residual information can be used to determine whether the motion vectors are useful in the H.264 encoding stage. If the amount of residual is high, it means that the motion estimation and the motion compensation for the macroblock can be improved. A fast MB mode decision algorithm for H.264 encoding based on the homogeneity of a MB is reported in [27] but homogeneity of motion compensated residual has more information about motion and is expected to lead to better MB mode decisions in a transcoder. As shown later, data mining algorithm can be used to exploit this relationship.

IV. Applying Machine Learning/Data Mining Techniques for the Mode Decision Tree

Machine learning algorithms are used in the Data Mining (DM) step of a Knowledge Discovery from Data (KDD) process, whose main goal is the extraction of knowledge from structured or raw data [28]. While DM has become the most popular term in the field of knowledge discovery, it is in fact only a step in the KDD process, where important steps are carried out before (data cleaning, data pre-processing, etc.) or after (model evaluation, knowledge dissemination, etc.) the application of DM algorithms. Thus, the DM step and hence the machine learning algorithms can be viewed as the core of KDD. The way in which DM algorithms obtain the knowledge can vary depending on the machine learning paradigm they are based on. In the case of the inductive approach, used in this work, the goal is to look for regularities into the data and

transform them into generalizations that will be expressed by using a knowledge representation model. DM has been used in an extensive range of applications including search engines, medical diagnosis, stock market analysis, classifying DNA sequences, speech and handwriting recognition, object recognition in computer vision, game playing, and robot motion.

The two main goals in DM can be broadly classified as: prediction and description. In predictive DM, the goal is to determine the value of a *target* (dependent) variable by using the values taken by some *predictive* variables (or attributes). In descriptive DM, the main goal is the discovering of association or relations between the variables and/or instances in the dataset (e.g., clusters, associations, graphs, etc.). In this paper we focus on predictive DM and more concretely in *classification* as our target (i.e., the *class*) variable (the MB decision mode) can take a finite number of (nominal) outcomes. From the different classification models available in the literature we use decision trees [29].

Multimedia Data Mining techniques have been developed over the last few years and have been mainly used to analyze or understand multimedia data [30]. Our approach in this paper is different; we aim to analyze intrinsic properties of multimedia data, but, instead of producing the discovered model as the output, our goal is to integrate it into a multimedia data treatment which in this case is a transcoder. In our opinion, the complexity of transcoding to H.264 creates an opportunity for applying machine learning algorithms in order to reduce the complexity of the transcoder. DM can be used to develop decision trees that will classify MB mode decisions without having to evaluate all the possible combination of partitions and sub-partitions. We propose a DM based approach to select the H.264 MB mode in a video transcoder. Two different video standards, H.263 and MPEG-2, will be used as input to the transcoder to demonstrate that the proposed techniques work well and can reduce the complexity of transcoding heterogeneous input video to the H.264 format.

In concrete, in this paper, we describe the process of using DM to build a classifier for very low complexity transcoding. The decision tree(s) will be used to determine the coding mode of an MB in P frames of the output H.264 video, based on the information gathered during the decoding stage of the input sequence. Fig. 2 depicts the process for building the decision trees to be used in the MPEG-2 or H.263 to H.264 transcoding process. In the training stage, the incoming video sequence is decoded and during the decoding stage, the MB coding mode, the coded block pattern (CBPC), and the mean and variance of the residual information for this MB (calculated for its 4×4 sub-blocks—resulting in 16 means and 16 variances for each MB) are saved. The decoded sequence is then encoded using the reference H.264 software implementation. The coding mode of the corresponding MBs in H.264 is also saved. Based on the data from the decoded sequence and the corresponding H.264 coding mode decision for each MB, a learning algorithm is used to grow the decision trees that classify a MB into one of the 11 H.264 MB coding modes.

A. Decision Trees

One of the most commonly used approaches in inductive knowledge discovery is the use of *decision trees*. A decision

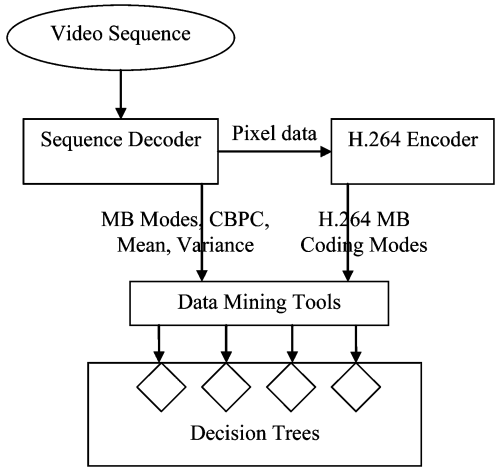

Fig. 2. Process for building decision trees to be used in a transcoder.

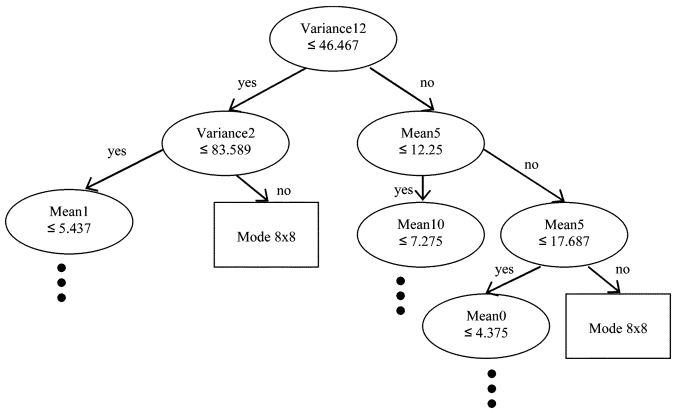

Fig. 3. Fraction of one of the decision trees learnt during our experiments. Inner (test) nodes are represented by ellipses while leaf (decision) nodes are represented by using boxes.

tree is made by mapping the observations about a set of data on to a tree made of arcs and nodes. The nodes are the variables and the arcs the possible values for that variable. Graphically, in a decision or classification tree, the inner nodes represent tests about the predictive attributes, the leaves are labels of the class variable and each branch descending from an inner node, asking about attribute X_i corresponds to one of the (possibly discretized) values for this attribute (see Fig. 3). In order to classify a given input (instance), the algorithm starts at the root node, tests the attribute specified by this node and descends by the appropriate branch to a new node. If this new node is a leaf then its class label is returned as output, otherwise the test process is repeated.

Decision trees constitute a standard choice in classification tasks because they possess very good properties, among others: 1) they usually achieve high accuracy rates in their predictions, but at the same time they are descriptive in the sense that the rules obtained from them (by following the conditions from the root to a leave) are fully interpretable by human beings; 2) there exists efficient algorithms to learn them from data, with the additional advantage that feature or attribute selection is simultaneously carried out, that is, only those attributes that are necessary to take the decision are included in the tree, while irrele-

```
@RELATION mpeg2-trainingset-node_1

@ATTRIBUTE mean0 Numeric
@ATTRIBUTE variance0 Numeric
@ATTRIBUTE mean1 Numeric
@ATTRIBUTE variance1 Numeric
.....
@ATTRIBUTE mean15 Numeric
@ATTRIBUTE variance15 Numeric
@ATTRIBUTE mode_mpeg2 {0,1,2,4,8}
@ATTRIBUTE CBPC0 {0,1}
.....
@ATTRIBUTE CBPC6 {0,1}
@ATTRIBUTE class {0,1,8,9}
```

Fig. 4. ARFF file format example (MPEG-2 to H.264).

vant or redundant attributes are discarded; 3) they can deal with both nominal and numerical predictive attributes; and 4) once the tree is learnt, prediction is carried out really fast. From these good properties of decision trees, 2) and 3) are inherent to the learning process, but 1) and 4) are related to the obtained model and are of crucial importance in our work. Thus, by obtaining an interpretable model, a tree or its corresponding set of rules in this case, we can almost directly translate it into C (or other programming language) code, while if other type or model is learnt, i.e., a neural network, we have to undertake a complex programming task and/or the use of external specific libraries. On the other hand, because the induced model is repetitively used during the transcoding process it is indispensable to have a short response time.

The decision tree that we propose to solve the interprediction problem is a model of the data that encodes the distribution of the class label (MB mode in H.264) in terms of the attributes (MB mode, CBPC, Mean and Variance from the incoming sequence). The final goal of this decision tree is to help find a simple structure to show the possible dependences between the attributes and the class.

With respect to the DM algorithm used to learn the model(s)/tree(s), we choose C4.5 which is a greedy, recursive, top-down algorithm for the induction of decision trees from data [29]. The algorithm starts at the root node with all the available data, and selects the best test, i.e., the most informative one with respect to the class, among the available. This test is placed as the root node and the data set is partitioned following the possible outcomes of the test. Then, the process is recursively repeated for each partition until a stopping criterion is met. In C4.5 the best attribute is selected by using information gain (based on Shannon's entropy) and numerical attributes are discretized on-line by searching for the threshold that yields the maximum entropy reduction (see [29] for details).

B. Creating the Training Files

In this section, we describe the data preparation process that leads us from video sequences to an appropriate input for the DM process. As commented before, we use J48 learning algorithm included in WEKA to learn the decision tree for MB mode classification. WEKA is a data mining suite that includes algorithms and tools for data pre-processing, classification, regression, clustering, association rules, and visualization [31]. It is also well-suited for developing new machine learning schemes. It is open source software issued under the GNU General Public License.

The input of WEKA algorithms are datasets representing flat files, that is files in which columns represent variables and rows instances. These files are known as ARFF (Attribute-Relation File Format) files [31]. An ARFF file is written in ASCII text and has two differentiated sections: the first section is a header while the second one contains the raw data. In the header section, we have the attribute declaration, that is, the name and possible values for each variable. For each macroblock, in the case of the MPEG-2 to H.264 transcoder, the proposed algorithm uses the following predictive attributes.

- Thirty two numerical variables: sixteen means $(\mathrm{mean}0, \ldots, \mathrm{mean}15)$ and sixteen variances $(\mathrm{variance}0, \ldots, \mathrm{variance}15)$ of 4×4 residual sub-blocks.
- MB mode in MPEG-2 ($\mathrm{skip}\{0\}$, $\mathrm{intra}\{1\}$, and three non-intramodes { 2-MC, coded; 4—MC, no coded; 8—No MC, coded }).
- Coded block pattern (CBPC) in MPEG-2 $\{0, 1\}$.
- Corresponding H.264 MB coding mode decision for that MB ($\mathrm{skip}\{0\}$, 16×16 $\{1\}$, 8×8 $\{8\}$ and intra $\{9\}$) as determined by the standard reference software.

If the goal is to train a decision tree for a H.263 to H.264 transcoder, the MB modes available in the H.263 decoding state are intra or inter, only.

Finally, the *class* variable, i.e, the variable that we are trying to understand, classify, or generalize is the H.264 MB coding mode, and can take four values: skip, 16×16, 8×8, or intra (labelled as 0, 1, 8 and 9 in the file). The Fig. 4 shows our declaration for the ARFF files in the MPEG-2 to H.264 scenario. After the header, the ARFF file contains the data section where each row represents an instance. In our case each training instance represents a macroblock sample and is obtained by encoding the MPEG-2 or H.263 decoded sequence with a quantization parameter of 25 and RD optimization enabled depending on the type of the decision tree created—MPEG-2 to H.264 transcoder or a H.263 to H.264 transcoder.

After extensive experimentation, we found that sequences that contain regions varying from homogenous to high-detail serve as good training sets. Good sample sequences could be Flower and Football. A large training set can thus be easily created using a large video sequence. We used the P frames to create the training dataset. The goal is to develop a single, generalized, decision tree that can be used for transcoding any MPEG-2 or H.263 video. In order to not obtain biased conclusions, the sequences used to train the model are not going to be used to test it.

C. Creating the Classifier

Until this moment, we have referred to the classifier to be learnt as a decision tree, however, our classifier is a *hierarchical classifier* composed of four decision trees, three of them learnt from data plus a simple one consisting of a single decision. Fig. 5 shows the structure of the proposed classifier whose building process is described in the following paragraphs.

The classifier for the proposed transcoder is a hierarchical decision tree that aggregates three learnt decision trees and

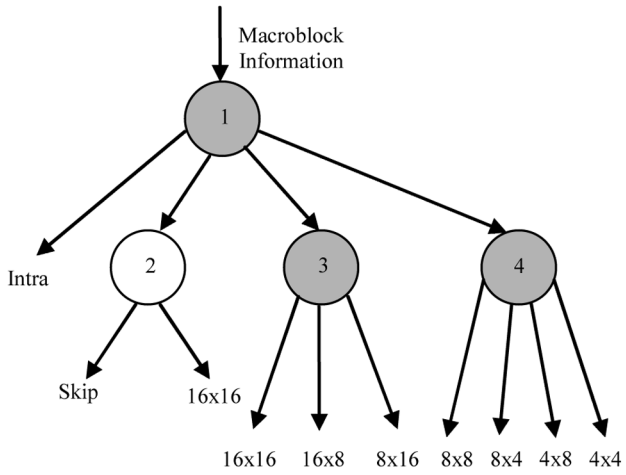

Fig. 5. Decision tree.

works in two different levels. It is obvious that we have two different Decision Trees, one for the MPEG-2 input video and the other for H.263 input. Both Decision Trees have the same node structure and were trained following the same steps described below. However, the training and test files were made using MPEG-2/H.264 or H.263/H.264 data depending of the scenario. The nodes of the Decision Tree shown in Fig. 5 are as follows.

- **Node 1** (Fig. 5) represents the decision tree learnt by using the training set described in Section IV-B, that is, it uses as input the mean and variance of each one of the sixteen 4×4 residual sub-blocks, the MB mode in MPEG-2 or H.263, the coded block pattern (CBPC), and the corresponding H.264 MB coding mode decision for that MB as determined by the standard reference software. The inputs for this node are all the coded MBs and a tree decision generated with WEKA is used to decide how the MB should be coded in H.264. This tree examines whether the MB has a very high residual or a medium residual. This tree makes an initial decision of the input into the following four possible labels: intra, skip, inter 16×16, and inter 8×8. In the case of an intradecision, no further processing is done because, in this paper, we focus on interframe mode decisions. In other cases, we continue with the decision tree because additional classification has to be carried out to determine possible sub-partitions.
- **Node 2** (Fig. 5) represents a single decision in order to decide between skip and inter 16×16. The inputs for this node are skipped MBs in the bitstream classified by the node 1. This node evaluates only the H.264 16×16 mode (without the sub-modes 16×8 or 8×16) and the best option, skipped or inter 16×16, is selected.
- **Node 3** (Fig. 5) is used for refining the initial decision mode of inter 16×16 into one of the possible (class labels) modes: 16×16, 16×8, and 8×16 MBs. The inputs to this node are the 16×16 MBs classified by the node 1. In order to take this decision, we train a new decision tree by using as input only the samples (MBs) that were encoded as 16×16 MBs in the H.264 reference encoder, and using as predictive attributes the mean and variances of each one of the sixteen 4×4 residual sub-blocks, the MB mode used in the format of the incoming sequence for the macroblock, the coded block pattern (CBPC), and as class variable the corresponding H.264 MB coding sub-mode decision in the 16×16 mode, as determined by the standard reference software: 16×16, 16×8, or 8×16.
- Analogously to Node 3, **Node 4** (Fig. 5) is used for refining a decision from node 1, inter 8×8 mode, into one of the possible modes: 8×8, 8×4, 4×8, or 4×4. The inputs for this node are the MBs classified by the node 1 as 8×8. This tree is run four times, once for each of the four sub-macroblocks in the MB. This tree is different from the others in that it only uses four means and four variances to make the decision. Furthermore, the training set used in this case was made by using only the samples (MBs) that were encoded as inter 8×8 MBs by the H.264 reference encoder. It contains four means and four variances of 4×4 residual sub-blocks, the MB mode, the coded block pattern (CBPC), and the corresponding H.264 MB sub-partition decision in the 8×8 mode, as determined by the standard reference software: 8×8, 8×4, 4×8, or 4×4.

At this point the reader can think about the possible alternative of building a flat classifier, that is, a decision tree that uses as class labels all the possible decision modes: intra, skip, 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, and 4×4. Of course, this is an alternative, likely the first one considered when facing this problem. A flat tree has the disadvantage of having to deal with all the classes at once that (at least in our experiments) usually lead to really poor accuracy rates and more complex trees. As two different transcoders are studied in this paper, we trained two different hierarchical classifiers, one to work with MPEG-2 and another to work with H.263. Due to space constraints we cannot show all the rules obtained by the DM algorithms and which constitute the hierarchical classifiers integrated in the transcoders, though the process described in herein should be sufficient for those interested in developing the decision trees and repeating these experiments.

D. About the Learnt Classifiers

In this subsection, we provide some data about the learnt classifiers. This analysis is based on the use of the data set obtained using a single frame (first P frame) of the Flower garden sequence as the input. The MPEG-2 to H.264 classifier was trained with a single P frame at 704×576 resolution and H.263 to H.264 classifier was trained using a single P frame at 352×288 resolution. Even though a single frame was used, the actual training samples were obtained from the MBs from that frame resulting a sufficiently large training set. The input bitstream produced datasets with 1204/320, 906/225, and 256/380 instances (MPEG2/H.263) and 38, 38, and 14 attributes used to train the decision trees placed in node 1, node 3, and node 4, respectively. Table I shows the key details of the obtained classifiers for both transcoders. Where #leaves, #nodes, and #vars stand for the number of leaves, inner nodes and variables included in the tree. These statistics give a measure about the complexity of the learnt trees, because the number of leaves coincides with the number of generated rules and the number

TABLE I
MAIN DATA FOR THE TREES

	MPEG-2 to H.264				H.263 to H.264			
	#leaves	#nodes	#vars	e(10CV)	#leaves	#nodes	#vars	e(10CV)
Tree at Node1	27	23	17	15.11	15	14	13	35.93
Tree at Node3	63	59	32	28.80	27	26	18	46.20
Tree at Node4	13	9	7	28.20	38	37	12	51.05

TABLE II
RESULTS OF EVALUATING THE LEARNT CLASSIFIERS OVER DIFFERENT DATASETS. N/A MEANS THAT THE RESULT IS NOT AVAILABLE BECAUSE NO RECORDS WHERE LABELLED AS 8×8 IN THE AKIYO SEQUENCE

	MPEG-2 to H.264			H.263 to H.264		
	Akiyo	Paris	Tempete	Akiyo	Paris	Tempete
Tree at Node1	13.40	12.28	14.06	0.31	9.06	25.00
Tree at Node3	0.00	23.30	42.14	0.63	16.88	37.90
Tree at Node4	n/a	18.75	8.33	n/a	35.93	46.25

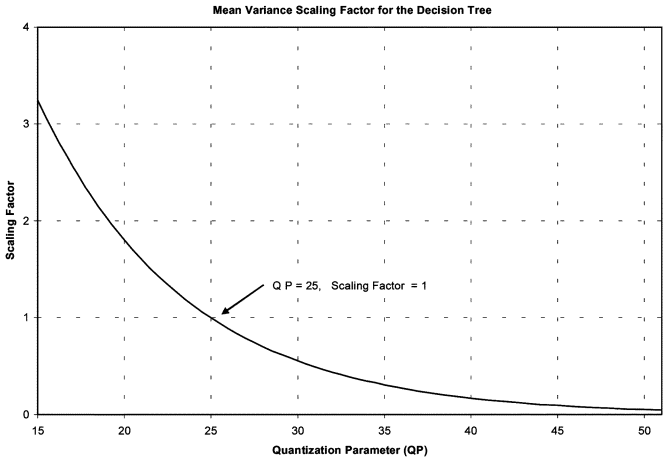

Fig. 6. Mean and variance scaling factor for the decision trees.

of nodes corresponds to the number of conditions to be tested per rule.

As we can see, the number of rules is not excessive, 103 for the MPEG hierarchical classifier and 80 for the H.263 case, and from the number of inner nodes we observe that the obtained rules have few conditions to be tested in their antecedent. With respect to the number of variables used in the tree, it is clear that in the first level more variables are considered redundant because only 17 (13) are used from the 38 available. To summarize the structural complexity of the learnt classifiers, it is clear that the classifier obtained for H.263 is the simplest one. With respect to the last column, it shows the expected error for our classifier computed by using a standard technique of 10 fold cross-validation (10cv). From the expected error shown in Table I, MPEG-2 to H.264 classifier seems more acceptable compared to the H.263 to H.264 classifier which has higher expected error. Even though we show the error column for the sake of completeness, expected error is not the correct way of evaluating the constructed hierarchical classifier. The reason is that the error shown in Table I is based on assuming a uniform cost of errors, while it is clear that in the case of a transcoder the cost of misclassifying a 16×16 MB sample into Skip is different from misclassifying it into 8×8. In this type of scenarios it is better to measure the error by using a nonuniform cost of errors, that is, by using a cost matrix that for every pair of class labels (Ci, Cj) indicates the cost of classifying a given sample as Ci when its actual value is Cj. Nevertheless, this is not our case because 1) such a cost matrix is complex to develop and is content dependent and 2) the better way of evaluating our proposal is to introduce the obtained classifiers in the transcoder and to evaluate their performance directly over the transcoding process, by using the parameters usually considered to measure the quality of a transcoder (BitRate, PSNR, and time reduction). This evaluation is presented in Section V and the results show that the two learnt classifiers are quite good based on the set evaluation criteria.

To finish with this subsection and also for the sake of completeness we present in Table II the error obtained when applying the classifier learnt with the training set obtained from the flowers sequence to the test set obtained from three different sequences: Akiyo, Paris, and Tempete. The goal of this experiment is to show the generalization power of the learnt classifiers, i.e., its capability when used to test unseen sequences. As we can see, the average error in each level is much lower than the cross validated error computed over the flower sequence (the one used for training), so we can conclude that our classifiers exhibit a good behavior with respect to generalization.

E. Adapting the Mean and the Variance Thresholds

Since the MB mode decision, and hence the thresholds, depend on the quantization parameter (QP) used in the H.264 encoding stage, the mean and variance threshold will have to be different at each QP. The two solutions here are 1) develop the decision trees for each QP and use the appropriate decision tree depending on the QP selected and 2) develop a single decision tree and adjust the mean and variance threshold used by the trees based on the QP. The first option is complex as we have to develop and switch between 52 different decision trees resulting in 156 WEKA trees in a transcoder. Since the QP used by H.264 is designed to change the quantization step size and the relationship between the QPs is well defined, this relationship can be used to adjust the mean and variance thresholds. The proposed transcoder uses a single decision tree developed for a mid-QP of 25 and then adjusted for other QPs. Since the quantization step size in H.264 doubles when QP increases by 6, the thresholds are adjusted by 12.5% for a change in QP of 1. Fig. 6 shows the change in scale factor applied to the decision tree as a function of the QP. The scale factor is applied to the thresholds in the decision trees and a single classifier with thresholds that are adjusted with the QP are used.

Figs. 7 and 8 show examples of the results obtained by applying our proposed algorithm to the sequence Paris in the CIF format, working with a MPEG-2 to H.264 trancoder, in the Fig. 7; or with a H.263 to H.264 transcoder in the Fig. 8. Subfigures (a) and (b) illustrate general information. Subfigures (c) and (d) show the mean and variance of the residual. The mean is shown adding $+128$ to the value of each pixel. This operation is done only for displaying purposes. subfigures (e) and (f) show the differences between the intermode selection made by the H.264 reference implementation (with the RD-optimized option enabled), and the proposed algorithm, with a value of 28 for QP. From these figures, it is clear that our algorithm obtains very similar results to those obtained using

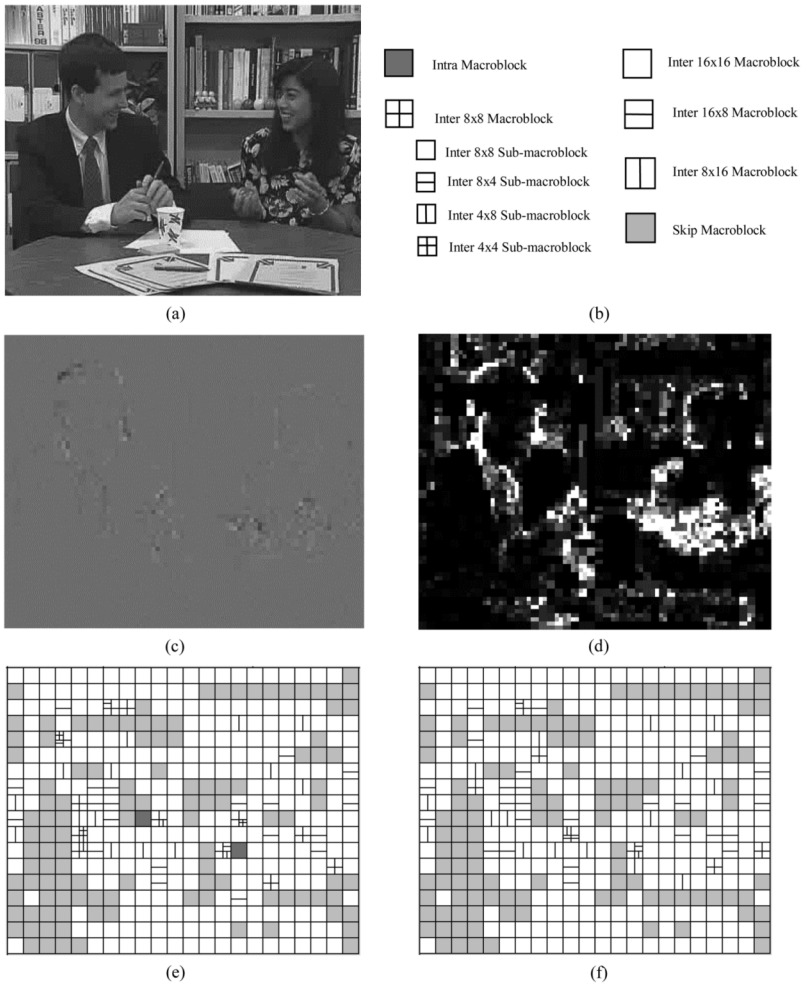

Fig. 7. Macroblock partitions generated by the proposed algorithm and the H.264 reference implementation (RD-optimized enabled) for the first P-frame in the Paris sequence. MPEG-2 to H.264 Transcoder. (a) Second Y frame (first P frame), sequence Paris (CIF). (b) Different kinds of intermacroblocks in the grid pictures. (c) Mean of the first P frame, sequence Paris (CIF). (d) Variance of the first P frame, sequence Paris (CIF). (e) H.264 $\mathrm{QP} = 28$ First P frame, sequence Paris (CIF). (f) Proposed $\mathrm{QP} = 28$ First P frame, sequence Paris (CIF).

the full estimation of the H.264 reference implementation. It is easy to see, comparing these grid images with the mean and the variance images in the Figs. 7 and 8, there is some relationships between the processed residual information and the H.264 Mode Partition selection in the both standards, MPEG-2 and H.263. High energy is displayed in the variance images with light colors. On the other hand, the dark regions in the variance means there is no movement information. The Macroblock Partition modes generated by the H.264, and the proposed algorithm we proposed, are also compared to measure the accuracy of the MB mode classification using grid images. A grid-image showing the MB modes overlaid on a corresponding frame is used to visually compare the MB mode classification.

V. Performance Evaluation

In order to evaluate the proposed macroblock partition mode decision algorithm for heterogeneous MPEG-2 or H.263, to H.264 video transcoders proposed in this work, we have implemented the proposed approach based on the H.264 reference software [14] (version JM 10.2). The platform used is a Pentium IV, 3.4 GHz and 2 Gbyte of RAM. Fig. 9 shows the overall operation of the proposed transcoder in a general transcoding scenario. The incoming video sequence is decoded and the information required by the decision trees is gathered in this stage. The additional computation here is the computation of the mean and variance of the 4×4 sub-blocks of the residual MBs. The MB coding mode decision determined by the decision trees is used in the low complexity H.264 encoding stage. This is an H.264 reference encoder with the MB mode decision replaced by simple mode assignment from the decision tree. The H.264 video encoder takes as input the decoder MPEG-2 or H.263 video (pixel data), the residual information, the mode decision and the coded block pattern (CBPC) for each macroblock of the P frames.

We have conducted an extensive set of experiments with videos representing a wide range of motion, texture, and color. Experiments were conducted to evaluate the performance of the proposed algorithm when transcoding videos at commonly used resolutions: CIF and QCIF. The input to the transcoder is a high quality MPEG-2/H.263 video, depending of the scenario. Since the proposed transcoders address transcoding P frames in MPEG-2/H.263 to H.264 P frames, MPEG-2 and H.263 bitstreams were created without B frames. Since the B frames, which are much smaller than P frames, are not used in the

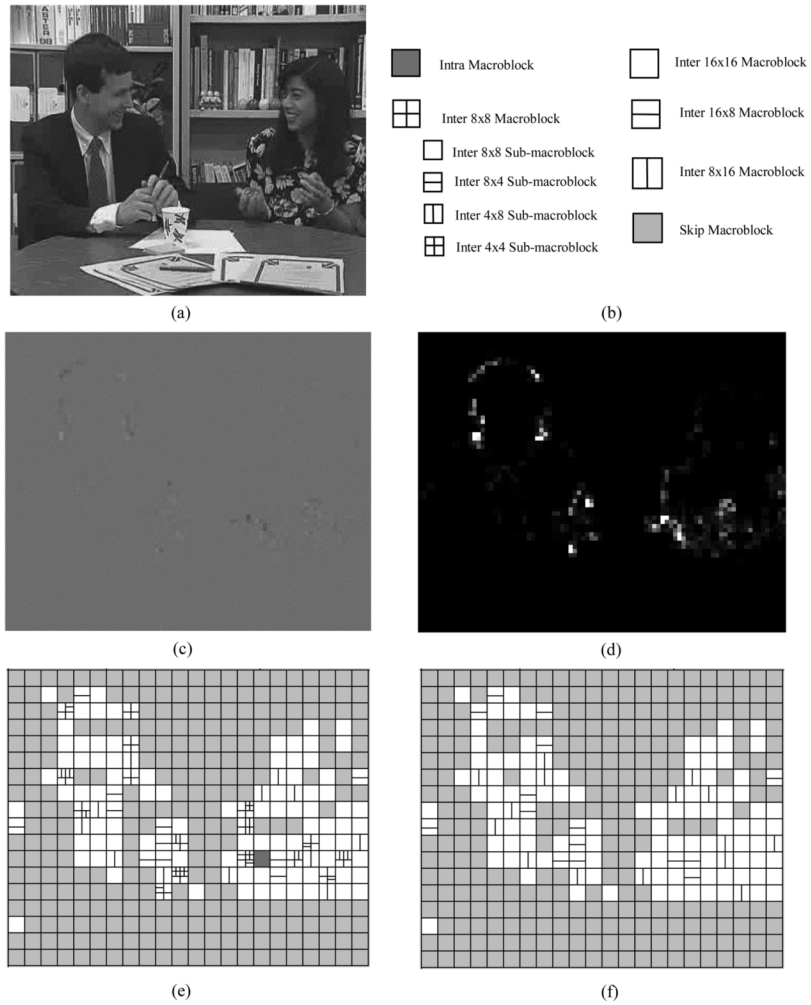

Fig. 8. Macroblock partitions generated by the proposed algorithm and the H.264 reference implementation (RD-optimized enabled) for the first P-frame in the Paris sequence. H.263 to H.264 Transcoder. (a) Second Y frame (first P frame), sequence Paris (CIF). (b) Different kinds of intermacroblocks in the grid pictures. (c) Mean of the first P frame, sequence Paris (CIF). (d) Variance of the first P frame, sequence Paris (CIF). (e) H.264 $\mathrm{QP} = 28$ First P frame, sequence Paris (CIF). (f) Proposed $\mathrm{QP} = 28$ First P frame, sequence Paris (CIF).

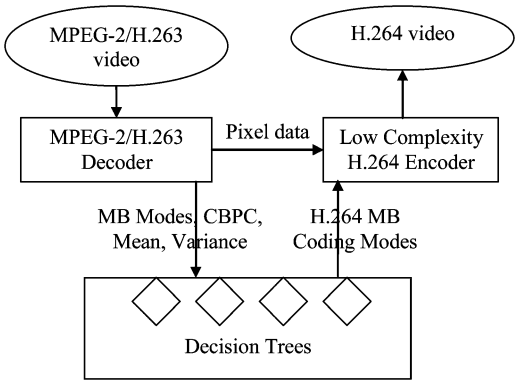

Fig. 9. Proposed transcoder.

TABLE III
CHARACTERISTICS OF MPEG-2 VIDEO BIT STREAMS

Sequence, Format		Bit Rate (Mbit/sec)	GOP	Motion search range	Format
Akiyo	(CIF)	1.15	I11(P)	15	Progressive
Paris	(CIF)	1.15	I11(P)	15	Progressive
Tempete	(CIF)	1.15	I11(P)	15	Progressive
Akiyo	(QCIF)	0.768	I11(P)	15	Progressive
Paris	(QCIF)	0.768	I11(P)	15	Progressive
Tempete	(QCIF)	0.768	I11(P)	15	Progressive

TABLE IV
CHARACTERISTICS OF H.263 VIDEO BIT STREAMS

Sequence, Format		Quantification Parameter (QP)	GOP	Motion search range	Format
Akiyo	(CIF)	12	I11(P)	16	Progressive
Paris	(CIF)	12	I11(P)	16	Progressive
Tempete	(CIF)	12	I11(P)	16	Progressive
Akiyo	(QCIF)	9	I11(P)	16	Progressive
Paris	(QCIF)	9	I11(P)	16	Progressive
Tempete	(QCIF)	9	I11(P)	16	Progressive

input video, the video has to be encoded at higher than the typical encoding rates for equivalent broadcast quality. The characteristics for the MPEG-2 video sequences, like the input bit-rate for the sequences, are shown in Table III. Table IV shows the same information, but in this case, for the H.263 input sequences. The H.263 encoder implementation does not have a Rate Control option, so we fixed the QP for the frames to get, approximately the same output bit-rate, like in the MPEG-2 encoder.

For re-encoding the input sequences, we use quantization parameters (QP) from $\mathrm{QP} = 28$ to $\mathrm{QP} = 40$. The size of the GOP

TABLE V
MOST RELEVANT PARAMETERS IN THE H.264 ENCODER CONFIGURATION FILE

	Type of sequence	
	CIF	QCIF
FramesToBeEncoded	200	200
FrameRate	25	25
SourceWidth	352	176
SourceHeight	288	144
ProfileIDC	100	100
IntraPeriod	12	12
UseHadamard	0	0
DisableSubpelME	0	0
SearchRange	16	16
NumberReferenceFrames	1	1
EnableIPCM	0	0
SymbolMode	0 (UVLC)	0 (UVLC)

is 12 frames; where the first frame of every GOP was encoded as I-frame, and the rest of the frames of the GOP were encoded as a P-frames (same GOP format in the input sequences). The rate control and CABAC algorithms were disabled for all the simulations. The number of reference in P frames was set to 1 and the motion search range was set to ± 16 pels with a MV resolution of 1/4 pel. The ProfileIDC was set to High for all the simulations, with the FREXt options enabled. The Table V shows the most relevant information we used in the encoder configuration file to run the simulations. The other parameters, not shown in the table, were set to the default option. The time results were normalized using the solution given by the reference software, as shown later, so they are independent from the machine used to run the simulations. The results are reported for ten different sequences, in the MPEG-2/H.264 and in the H.263/H.264 scenarios: five for each of the two resolutions shown in Tables III and IV.

The performance of the proposed very low complexity transcoder is compared with a reference transcoder comprised of a full MPEG-2 decoder followed by a full H.264 encoder, or a full H.263 decoder followed by a full H.264 encoder. We compare the performance of our proposal to the full H.264 encoder when the RD-optimized option is enabled. The metrics used to evaluate the comparative performance are: the rate-distortion function, the difference of coding time ($\Delta\mathrm{Time}$), the PSNR difference ($\Delta\mathrm{PSNR}$) and the bit-rate difference ($\Delta\mathrm{Bitrate}$). The averaged PSNR values of luma (Y) and chroma (U, V) is used in the rate-distortion function graphs. The averaged PSNR is based on the following equation:

$$\overline{\mathrm{PSNR}} = \frac{4 \times \mathrm{PSNR}_Y + \mathrm{PSNR}_U + \mathrm{PSNR}_V}{6}.$$

In order to evaluate the timesaving of the fast MB mode decision algorithm, the following calculation is defined to find the time differences. Let T_{JM} denote the coding time used by the H.264/AVC reference software encoder (version JM 10.2) and T_{FI} be the time taken by the fast MB mode decision algorithm proposed, and $\Delta\mathrm{Time}$ is defined as

$$\Delta\mathrm{Time}(\%) = \frac{T_{FI} - T_{JM}}{T_{JM}} \times 100.$$

The PSNR and bit-rate differences are calculated according to the numerical averages between the RD-curves derived from JM 10.2 encoder and the fast MB mode decision methods, respectively. The detail procedures in calculating these differences can be found from a JVT document authored by Bjontegaard [32], which is recommended by JVT Test Model Ad Hoc Group [33]. Note that PSNR and bit-rate differences should be regarded as equivalent, i.e., there is either the decrease in PSNR or the increase in bit-rate, but not both at the same time.

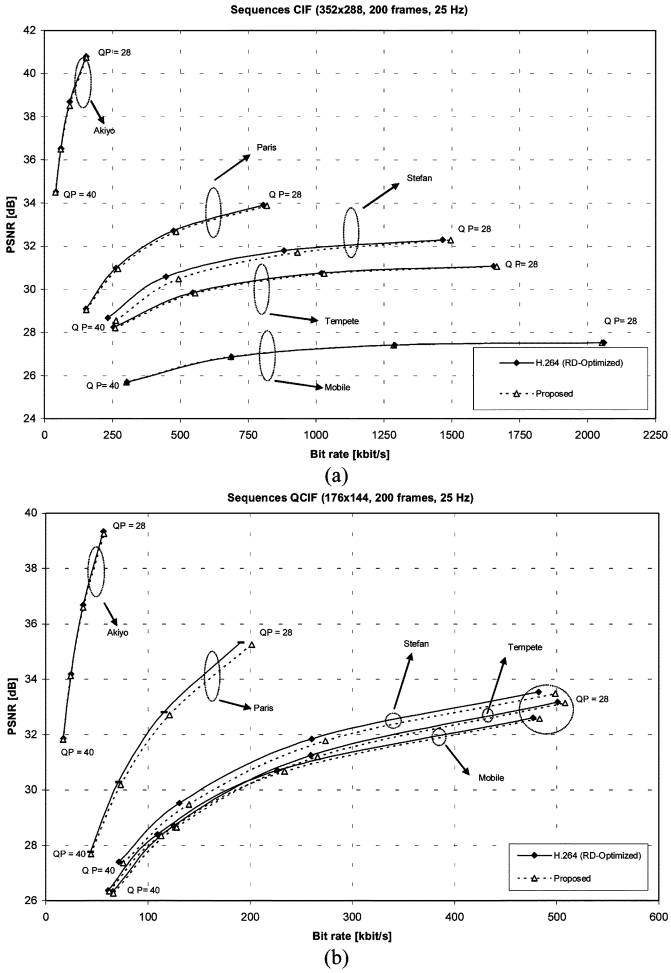

Fig. 10. Rate distortion results comparing the performance of the proposed and reference MPEG-2/H.264 transcoder. (a) CIF and (b) QCIF.

Fig. 10 shows the *Rate Distortion* results of applying the following options: Full RD-optimized H.264 encoding with the reference transcoder and the proposed low complexity H.264 encoding stage, in the MPEG-2/H.264 transcoder scenario. Fig. 11 shows the *Rate Distortion* results of applying the following options: Full RD-optimized H.264 encoding with the reference transcoder and the proposed low complexity H.264 encoding stage, in the H.263/H.264 transcoder scenario. The RD-optimized encoding stage represents the upper bound on the RD performance. The goal of this work is to design decision trees that result in RD performance as close as possible to the RD-optimized H.264 encoding (upper bound).

A. Analyzing the Results

As seen from Figs. 10 and 11, the PSNR obtained when applying our algorithm deviates slightly from the results obtained when applying the considerably more complex full

TABLE VI
$\Delta\mathrm{BitRate}$, $\Delta\mathrm{PSNR}$ AND TIME REDUCTION RESULTS

Sequence, Format, QP´s			ΔBitRate, ΔPSNR and Time Reduction (mean) from H.264 (RD-Optimized)					
			Proposed (MPEG-2/H.264)			Proposed (H.263/H.264)		
			Time (%)	ΔPSNR (dB)	ΔBitRate (%)	Time (%)	ΔPSNR (dB)	ΔBitRate (%)
Akiyo	CIF	(28,32,36,40)	- 72.49	- 0.052	1.78	- 62.12	- 0.003	0.18
Paris	CIF	(28,32,36,40)	- 73.63	- 0.083	3.75	- 61.00	- 0.021	0.95
Tempete	CIF	(28,32,36,40)	- 71.01	- 0.045	2.95	- 61.18	- 0.040	2.75
Stefan	CIF	(28,32,36,40)	- 71.10	- 0.245	13.76	- 61.11	- 0.255	13.91
Mobile	CIF	(28,32,36,40)	- 70.04	- 0.012	0.96	- 61.90	- 0.037	2.49
Akiyo	QCIF	(28,32,36,40)	- 72.31	- 0.037	0.92	- 66.67	- 0.005	0.22
Paris	QCIF	(28,32,36,40)	- 73.36	- 0.216	5.94	- 61.39	- 0.050	2.70
Tempete	QCIF	(28,32,36,40)	- 71.37	- 0.120	3.91	- 61.19	- 0.051	2.71
Stefan	QCIF	(28,32,36,40)	- 70.84	- 0.205	7.79	- 60.89	- 0.215	7.91
Mobile	QCIF	(28,32,36,40)	- 71.26	- 0.114	4.13	- 59.92	- 0.070	3.83

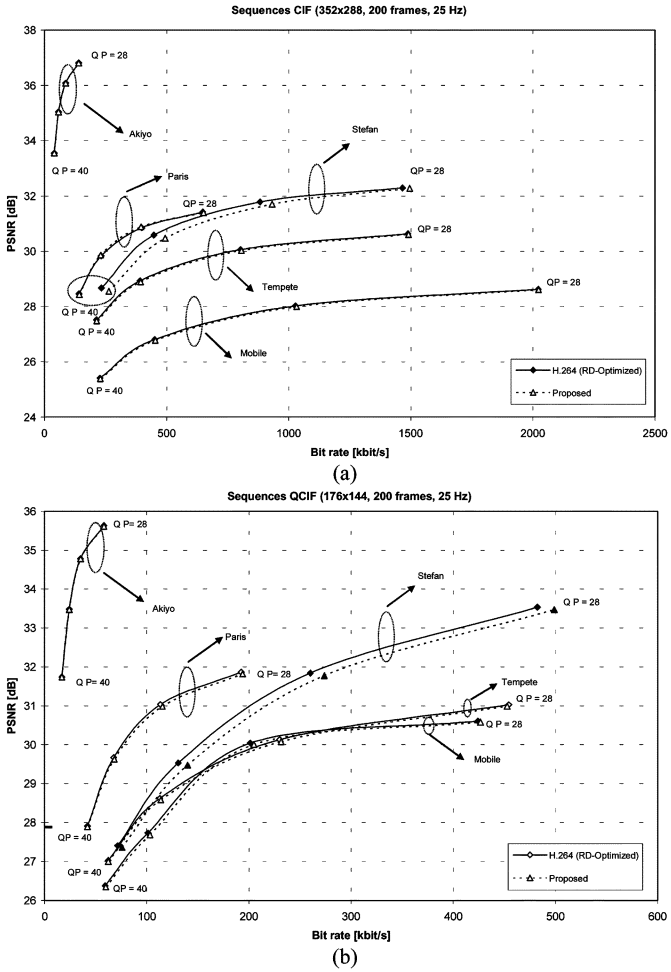

Fig. 11. Rate Distortion Results comparing the performance of the proposed and reference H.263/H.264 transcoder. (a) CIF and (b) QCIF.

RD-optimized H.264 encoding with the MPEG-2/H.264 or the H.263/H.264 reference transcoder. Table VI shows the results in terms of $\Delta\mathrm{Time}$, Δ PSNR, and Δ Bitrate for the proposed DM based encoding stage compared with the RD optimized encoding stage for encoding 200 frames of each sequence. PSNR and bit-rate differences are calculated according to the numerical averages between the RD-curves derived from JM encoder (RD-optimized option enabled), the algorithm under study. The negligible drop in RD performance is more than offset by the reduction in computational complexity. However, in terms of time saving, which is a critical issue in video transcoding applications, the proposed method significantly reduces the time for re-encoding the sequences. As shown in the Table VI, the proposed transcoder reduces the interframe prediction time by over 72% with RD optimization enabled, in a MPEG-2/H.264 transcoder scenario, and by over 62% in an H.263/H.264 trancoder scenario.

B. Comparison of Different Fast MB Mode Decision Algorithms

As explained in Section III, recently fast MB mode decision algorithms for interframe prediction in H.264/AVC video coding have been proposed in the literature. In this final study, we undertake a comparative analysis with some of the most prominent ones presented in [18] and [20]–[22]. In this experiment, although test conditions are not perfectly the same, objective comparisons is still possible because all three algorithms follow Bjontegaard and Sullivan's common test rule [34]. The comparison metrics were produced and tabulated based on the difference of coding time $(\Delta\mathrm{Time})$, the PSNR difference $(\Delta\mathrm{PSNR})$ and the bit-rate difference $(\Delta\mathrm{Bitrate})$. The common encoding parameters are as follows: the size of the GOP is 12 frames; where the first frame of every GOP was encoded as I-frame, and the rest of the frames of the GOP were encoded as a P-frames. The RD optimization was enabled. The video sequences used were Paris (CIF) and Mobile (CIF). It should be noted that we have selected the common sequences used in [18] and [20]–[22] in order to make a fair comparison. A group of experiments were carried out on the test sequence with the 4 quantization parameters, i.e., $\mathrm{QP} = 28$, 32, 36, and 40 as specified in [34]. The results are tabulated in Table VII.

As shown in Table VII, the performance of our fast MB mode decision algorithm for interframe prediction in terms of time saving, which is a critical issue in video transcoding applications, achieves the best results, with a negligible loss of video quality $(< 0.08\,\mathrm{dB})$, and with a slight increment in bit rate with respect to the JM reference software. This is due to proposed approach reduces the H.264 MB mode computation process into a decision tree lookup with very low complexity and the transcoder performs the fast motion estimation just for the final MB mode determined by the decision tree. Furthermore, the proposed transcoder can be implemented easily due to it only

TABLE VII
COMPARISON OF DIFFERENT INTER-MB MODE DECISION ALGORITHMS

Sequence	Method	ΔTime (%)	ΔPSNR (dB)	ΔBitrate (%)
Paris (CIF)	**Our Proposal (MPEG-2/H.264)**	**-73.63**	**-0.08**	**3.75**
	Our Proposal (H.263/H.264)	**-61.00**	**-0.02**	**0.95**
	Lim´s algorithm	-34.37	-0.04	0.73
	Jeon´s algorithm	-34.69	-0.05	0.92
	Kim´s algorithm	−62.05	-0.02	0.36
	Wu´s algorithm	-31.90	-0.04	0.87
Mobile (CIF)	**Our Proposal (MPEG-2/H.264)**	**-70.04**	**-0.01**	**0.96**
	Our Proposal (H.263/H.264)	**-61.90**	**-0.03**	**2.49**
	Lim´s algorithm	- 10.96	-0.01	0.01
	Jeon´s algorithm	- 22.83	-0.02	0.07
	Kim´s algorithm	- 46.93	-0.01	0.10
	Wu´s algorithm	- 9.97	-0.01	0.13

requires the mean and variance of the residual (MPEG-2 or H.263), and other information available in the decoding stage, for creating a set of rules to compare the mean and variance against a threshold.

VI. CONCLUSION

In this paper, we proposed Data Mining techniques as a part of macroblock partition mode decision algorithm for inter-frame transcoding in an MPEG-2/H.263 to H.264 transcoder. The proposed algorithms exploit the correlation between the MPEG-2/.H.263 MC residual and the H.264 MB coding modes. The WEKA tool was used to develop decision trees for H.264 coding mode decision. The proposed algorithm has very low complexity as it only requires the mean and variance of the MPEG-2/H.263 residual, and the MB information coming from the input MPEG-2/H.263 video. Our results show that the proposed algorithm is able to maintain a good picture quality while considerably reducing the computational complexity by 72% on average with MPEG-2, and by 62% on average with H.263.

REFERENCES

[1] Advanced video coding for generic audiovisual services ITU-T Recommend. H.264, 2003.

[2] Video coding for low bit rate communication TU-T Recommend. H.263I: Transmission of Non-Telephone Signals, 1998.

[3] *Generic Coding of Moving Picture and Associated Audio*, ISO/IEC Std. 13818-2, 1994.

[4] *Inf. Technology Generic Coding of Audio-Visual Objects- Part 2: Visual*, ISO/IEC Std. 14 496-2, Mar. 1999.

[5] T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," *IEEE Trans. Multimedia*, vol. 2, no. 2, pp. 101–110, Jun. 2000.

[6] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: An overview," *IEEE Signal. Process. Mag.*, vol. 20, no. 2, pp. 18–29, Mar. 2003.

[7] H. Kalva, A. Vetro, and H. Sun, "Performance optimization of the MPEG-2 to MPEG-4 video transcoder," in *Proc. SPIE Conf. Microtechnologies For New Millennium, VLSI Circuits Syst.*, May 2003.

[8] S. Dogan, A. H. Sadka, and A. M. Kondoz, "Efficient MPEG-4/H/263 video transcoder for interoperability of heterogeneous multimedia networks," *Electron. Lett.*, vol. 35, no. 11, pp. 863–864, May 1999.

[9] H. Kalva, "Issues in H.264/MPEG-2 video transcoding," presented at the Consumer Commun. Networking Conf., Jan. 2004.

[10] Y. Su, J. Xin, A. Vetro, and H. Sun, "Efficient MPEG-2 to H.264/AVC intra transcoding in transform-domain," presented at the IEEE Int. Symp. Circuits Syst., May 2005.

[11] B. Petljanski and H. Kalva, "DCT domain intra MB mode decision for MPEG-2 to H.264 transcoding," presented at the ICCE, Jan. 2006.

[12] Y.-K. Lee, S.-S. Lee, and Y.-L. Lee, "MPEG-4 to H.264 transcoding using macroblock statistics," presented at the ICME, Jul. 2006.

[13] X. Lu, A. M. Tourapis, P. Yin, and J. Boyce, "Fast mode decision and motion estimation for H.264 with a focus on MPEG-2/H.264 transcoding," presented at the IEEE Int. Symp. Circuits Systems, May 2005.

[14] *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG*, Reference Software to Committee Draft. JVT-F100 JM10.2, 2006.

[15] G. Sullivan and T. Wiegand, "Rate-Distortion optimization for video compression," *IEEE Signal. Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[16] T. Wiegand *et al.*, "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.

[17] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: Tools, performance, and complexity," *IEEE Circuits Syst. Mag.*, vol. 4, no. 1, pp. 7–28, Jan. 2004.

[18] K. P. Lim *et al.*, "Fast inter mode selection," *JVT MPEG VCEG, D. JVT-I020*, Sep. 2003.

[19] J. Lee and B. Jeon, "Pruned mode decision based on variable block sizes motion compensation for H.264," *Lecture Notes Comput. Sci.*, vol. 2899, pp. 410–418, Nov. 2003.

[20] B. Jeon and J. Lee, "Fast mode decision for H.264," *JVT MPEG VCEG, Doc. JVT-J033*, Dec. 2003.

[21] Y.-H. Kim, J.-W. Yoo, S.-W. Lee, J. Shin, J. Paik, and H.-K. Jung, "Adaptive mode decision for H.264 encoder," *Electron. Lett.*, vol. 40, no. 19, pp. 1172–1173, Sep. 2004.

[22] D. Wu, F. Pan, K. P. Lim, S. Wu, Z. G. Li, X. Lin, S. Rahardja, and C. C. Ko, "Fast intermode decision in H.264/AVC video coding," *IEEE Trans. Circuits. Syst., Video Technol.*, vol. 15, no. 7, pp. 953–958, Jul. 2005.

[23] J. Bialkowski, M. Menden, M. Barkowsky, A. Kaup, and K. Illgner, "A fast H.263 to H.264 inter-frame transcoder with motion vector refinement," presented at the Picture Coding Symp., San Francisco, CA, Dec. 2004.

[24] J. Bialkowski, M. Barkowsky, and A. Kaup, "Overview of low-complexity video transcoding from H.263 to H.264," presented at the Int. Conf. Multimedia Expo, Toronto, ON, Canada, Jul. 9–12, 2006.

[25] K.-T. Fung and W.-C. Siu, "Low complexity H.263 to H.264 video transcoding using motion vector decomposition," presented at the IEEE Int. Symp. Circuits Systems, May 2005.

[26] V.-A. Nguyen and Y.-P. Tag, "Efficient video transcoding from H.263 to H.264/AVC standard with enhanced rate control," *EURASIP*, pp. 1–15, 2006.

[27] H.-M. Wang, J.-K. Lin, and J.-F. Yang, "Fast inter mode decision algorithm based on hierarchical homogeneous detection & cost analysis for H.264/AVC coders," in *Proc. IEEE ICME*, 2006, pp. 709–712.

[28] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Mag.*, vol. 17, pp. 37–54, 1996.

[29] J. R. Quinlan, *C4.5: Programs For Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[30] V. Petrushin and L. Khan, *Multimedia Data Mining and Knowledge Discovery*. New York: Springer Verlag, 2007.

[31] H. W. Ian and F. Eibe, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2005.

[32] G. Bjontegaard, "Calculation of average PSNR differences between rd-curves," in *Proc. 13th VCEG-M33 Meeting*, Austin, TX, Apr. 2001.

[33] Evaluation sheet for motion estimation JVT Test Model Ad Hoc Group, 2003, draft ver. 4.

[34] G. Sullivan and G. Bjontegaard, Recommended simulation common conditions for H.26L coding efficiency experiments on low-resolution progressive-scan source material 2001, ITU-T VCEG, Doc. VCEG-N81.

Gerardo Fernández-Escribano (S'04) received the M.Sc. degree in computer engineering in 2003 and the Ph.D. degree from the University of Castilla-La Mancha, Spain, in 2007.

In 2004, he joined the Department of Computer Engineering at the University of Castilla-La Mancha, where he is currently a Researcher of computer architecture and technology. His research interests include multimedia standards, video transcoding, video compression, video transmission, and machine learning mechanism. He has also been a Visiting Researcher at the Florida Atlantic University, Boca Raton, and at the Friedrich Alexander Universität, Erlangen-Nuremberg, Germany.

Jens Bialkowski received the Dipl.-Ing. degree in electrical engineering from the University of Ulm, Germany, in 2001.

In January 2002, he joined the Chair of Multimedia Communications and Signal Processing, University of Erlangen-Nuremberg. There, he is currently a Research Assistant in cooperation with Siemens AG (Corporate Technology), Munich, as he pursues the Dr.-Ing. degree. His research is focused on digital video processing, where he is concentrating on investigation of video transcoding.

Jose A. Gámez received the M.S. and Ph.D. degrees in computer science from the University of Granada, Spain, in 1991 and 1998, respectively.

He joined the Department of Computer Science, University of Castilla-La Mancha (UCLM) in 1991, where he is currently an Associate Professor. He served as Vice-Dean of the Escuela Politecnica Superior de Albacete (UCLM) from 1998 to 2004, and he currently serves as Chair of the Department of Computing Systems (UCLM). His research interests include probabilistic reasoning, Bayesian networks, evolutionary algorithms, machine learning, and data mining. He has edited five books and published more than 50 papers over these topics.

Hari Kalva (M'92–SM'05) joined the Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, as an Assistant Professor in August 2003. Prior to that, he was a Consultant with Mitsubishi Electric Research Labs, Cambridge, MA. He was a Co-Founder and the Vice President of Engineering of Flavor Software, a New York company founded in 1999, that developed MPEG-4 based solutions for the media and entertainment industry. He is an expert in digital audio-visual communications systems, with over 12 years of experience in multimedia research, development, and standardization. He has made key contributions to the MPEG-4 Systems standard and also contributed to the DAVIC standards development. His research interests include pervasive media delivery, content adaptation, video transcoding, video compression, and communication. He has over 50 published papers and five patents (11 pending) to his credit. He is the author of one book and coauthor of five book chapters.

Pedro Cuenca (M'99) received the M.Sc. degree in physics (electronics and computer science, award extraordinary) from the University of Valencia, Spain, in 1994. He received the Ph.D. degree in computer engineering in 1999 from the Polytechnic University of Valencia.

In 1995, he joined the Department de Computer Engineering at the University of Castilla-La Mancha, Spain. He is currently an Associate Professor of communications and computer networks and Vice-Dean of the Escuela Politécnica Superior de Albacete (School of Computer Engineering). He has also been a Visiting Researcher at The Nottingham Trent University, Nottingham, U.K., and at the Multimedia Communications Research Laboratory, University of Ottawa, Ottawa, ON, Canada. His research topics are centered in the area of high-performance networks, wireless LAN, video compression, QoS video transmission, and error-resilient protocol architectures. He has published over 70 papers in international journals and conferences on computer networks and performance evaluation. He has been a Reviewer for several journals and several international conferences.

Dr. Cuenca has served in the organization of International Conferences as Chair, Program Co-Chair, and Technical Program Comittee Member. He is a member of the IFIP 6.8 Working Group.

Luis Orozco-Barbosa (M'91) received the B.Sc. degree in electrical and computer engineering from the Universidad Autonoma Metropolitana, Mexico, in 1979, the Diplome d'études Approfondies in computer science from the École Nationale Supérieure d'Informatique et de Mathématiques Appliquées (ENSIMAG), France, in 1984, and the Doctorat de l'Université in computer science from the Université Pierre et Marie Curie, France, in 1987.

From 1991 to 2002, he was a Faculty Member at the School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, ON, Canada. In 2002, he joined the Department of Computer Engineering, Universidad de Castilla-La Mancha, Spain. He has also been appointed Director of the Albacete Research Institute of Informatics, a National Centre of Excellence. He has conducted numerous research projects with the private sector and served as Technical Advisor for the Canadian International Development Agency (CIDA). He has published over 180 papers in international journals and conferences on computer networks and performance evaluation. His current research interests include Internet protocols, network planning, wireless communications, traffic modeling, and performance evaluation.

André Kaup (M'96–SM'99) received the Dipl.-Ing. and Dr.-Ing. degrees in electrical engineering from the Aachen University of Technology (RWTH), Aachen, Germany, in 1989 and 1995, respectively.

From 1989 to 1995, he was with the Institute for Communication Engineering, Aachen University of Technology, where he was responsible for industrial, as well as academic, research projects in the area of high-resolution printed image compression, object-based image analysis and coding, and models for human perception. In 1995, he joined Siemens Corporate Technology, Munich, Germany, where he chaired European research projects in the area of very low bit-rate video coding, image quality enhancement, and mobile multimedia communications. In 1999, he was appointed Head of mobile applications and services, with research focusing on multimedia adaptation for heterogeneous communication networks. Since 2001, he has been a Full Professor and Head of the Chair for Multimedia Communications and Signal Processing at the University of Erlangen-Nuremberg. His current research interests are in image and video coding, multimedia signal processing, and visual feature processing. From 1997 to 2001, he was an Adjunct Professor at the Technical University of Munich.

Dr. Kaup is a member of the German Informationstechnische Gesellschaft. He was elected Siemens Inventor of the Year 1998 and is the recipient of the 1999 ITG Award. From 1997 to 2001, he was also Head of the German MPEG delegation.