Quad-Tree Motion Modeling with Leaf Merging

Reji Mathew and David S. Taubman

***Abstract*—In this paper, we are concerned with the modeling of motion between frames of a video sequence. Typically, it is not possible to represent the motion between frames by a single model and therefore a quad-tree structure is often employed where smaller, variable size regions or blocks are allowed to take on separate motion models. Previous work into quad-tree representations has demonstrated the sub-optimal performance of quad-trees where the dependency between neighboring leaf nodes with different parents is not exploited. Leaf merging has been proposed to rectify this performance loss as it allows joint coding and optimization of related nodes. In this paper, we describe how the merging step can be incorporated into quad-tree motion representations for a range of motion modeling contexts. In particular, we study the impact of rate-distortion optimized merging for two motion coding schemes, these being spatially predictive coding, as used in H.264, and hierarchical coding. We present experimental results which demonstrate that node merging can provide significant gains for both the hierarchical and spatial prediction schemes. Interestingly, experimental results also show that in the presence of merging, the rate-distortion performance of hierarchical coding is comparable to that of spatial prediction. We pursue the case of hierarchical coding further in this paper, introducing polynomial motion models to the quad-tree representation and exploring resolution scalability of the merged quad-tree structure. We also present a theoretical study of the impact of leaf merging in modeling motion, identifying the inherent advantages of merging which give rise to a more efficient description of frame motion.**

***Index Terms*—Motion compensation, video coding.**

I. Introduction

FOR MODELING image features, the quad-tree structure is often used as it allows an image to be recursively divided into smaller regions with each region represented by a suitable model. For image compression, which can be considered a form of image modeling, the quad-tree representation is attractive as it lends itself to direct rate-distortion optimization, allowing a Lagrangian cost function to be globally minimized using tree pruning strategies. In this paper, we are concerned with the modeling of motion between frames of a video sequence. Typically, it is not possible to represent the motion between frames by a single model and therefore a quad-tree structure is employed where smaller, variable size regions or blocks are allowed to take on separate motion models.

Previous work into quad-tree representations [1] has demonstrated the sub-optimal performance of quad-trees where the dependency between neighboring leaf nodes with different parents is not exploited. Leaf merging has been proposed to rectify this performance loss, as it allows joint coding and optimization of related nodes. The benefits of leaf merging in the context of quad-tree motion models was recently demonstrated in [2] and [3]. In these earlier investigations of leaf merging for motion modeling [2], [3], an optimally pruned H.264 motion model was taken as the starting point for subsequent merging steps. To study the potential of leaf merging in a more general context, our initial work [4] focused on removing some of the restrictions imposed by H.264 in representing motion. In this paper, we explore the performance of leaf merging more deeply, over a wider range of conditions.

Experimental results presented in this paper demonstrate the benefits of merging. Although in this paper we are not primarily concerned with H.264 compression, we often refer to the H.264 case for comparison purposes. In our paper, we often use the term *H.264 motion model*; by this we mean the tree structure and the predictive motion coding that the H.264 standard specifies [5]. This corresponds to a tree structure of 3 levels with motion modeled by blocks of size 16×16, 8×8, and 4×4 as well as rectangular regions of size 16×8, 8×16, 8×4, and 4×8. The H.264 syntax allows forward-only, backward-only, and bi-predictive coding options. For each block, the motion vector components are differentially coded using either median or directional prediction from neighboring blocks [6].

The key contributions of this paper are as follows.

1) We describe how the intuitively appealing merging step can be incorporated into quad-tree motion representations for a range of motion modeling contexts.
2) We demonstrate with experimental results the improvement in R-D performance that can be gained by introducing merging for the two cases of hierarchical and spatially predictive motion coding (such as that employed by H.264).
3) We report on the benefits of polynomial modeling and hierarchical coding, once merging has been incorporated into the conventional quad-tree approach.
4) We present a theoretical analysis exploring the potential of leaf merging in a setting that is free from estimation issues and limitations of motion observability.

We now summarize the organization of the paper and the progression of ideas presented in this paper. In Section II, we

Manuscript received August 25, 2009; revised January 30, 2010; accepted April 14, 2010. Date of publication September 20, 2010; date of current version October 8, 2010. This paper was recommended by Associate Editor E. Steinbach.

R. Mathew is with the University of New South Wales, Sydney, NSW 2052, and also with the National ICT Australia, Sydney 1466, Australia (e-mail: reji@unsw.edu.au).

D. S. Taubman is with the University of New South Wales, Sydney, NSW 2052, Australia (e-mail: d.taubman@unsw.edu.au).

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/TCSVT.2010.2077480

discuss the main focus of our research in relation to other related work. In Section III, we introduce a general quad-tree structure for modeling motion and then describe the merging process that is applied to the nodes of the quad-tree. We then provide experimental results demonstrating the benefits of merging for both hierarchical and spatially predictive motion coding methods. These experimental results show that in the presence of merging the performance of both hierarchical coding and spatial prediction are similar. This outcome, while not altogether surprising, is an important benefit of the merging principle; it allows us to focus further on simpler and more scalable hierarchical coding schemes, as opposed to those involving spatial prediction, which are more complex and inherently non-scalable. In Section IV polynomial motion models are introduced and the resulting R-D performance of the merged quad-tree representation is compared with conventional motion modeling approaches. In Section V, we explore resolution scalability of the hierarchically coded merged quad-tree and provide accompanying experimental results illustrating the performance of the scalability feature. In Section VI, we present a theoretical analysis of the benefits of leaf merging.

II. Relation to Previous Work

The main focus of our research is on rate-distortion optimized motion modeling; specifically we address block based motion modeling which has received much attention over the years. There are multiple applications which require motion modeling of which video coding represents a prime example. Our motivation is not to create a motion model that is optimized for a particular video encoder but to provide a general framework for block based motion modeling that addresses some of the sub-optimal properties of typical block based schemes.

Optimizing our proposed motion model for a particular video encoder, such as H.264, would require the distortion measure to incorporate the characteristics of the corresponding image transform and quantizer. We note that in practical implementations of video encoders this is rarely done due to reasons of complexity. A more popular strategy is to measure the sum of absolute difference (SAD) in the Hadamard transform domain which provides a low cost alternative to assessing distortion in the transform domain; this distortion is then used within a Lagrangian cost function to make final decisions on the coding mode of each block [7], [8]. In our paper, we also use Hadamard SAD as the distortion measure and employ the Lagrangian cost function to make rate-distortion optimized decisions. The motion modeling strategy that we follow therefore closely resembles that of typical video encoder implementations and the improvements we report should also be valid to other platforms that share the same distortion metric and cost measure.

A highly scalable video encoder based on the wavelet transforms has been described in previous publications [9], [10]. To provide complete results within a video encoding framework, we integrate our proposed motion model with this wavelet based video encoder; results which take into account both motion and residue coding are presented in later sections of this paper.

In recent times, a number of different approaches have been proposed for scalable motion coding. Initially, scalable motion was proposed in the context of wavelet-based video coding, using a mesh-based motion model [9], [11]. We compare scalability performance of our proposed merged quad-tree with this mesh based approach in later sections of this paper. Many recent approaches to scalable motion coding have involved block-based motion models with scalability in resolution, fidelity or both [12]–[16]. Many of these previous approaches deal with scalable descriptions of tree based structures comprised of variable size blocks with translational motion vectors; these tree based structures however do not employ merging. The scalable motion coding proposed in this paper is intended to provide an indication of the benefits brought by merging and polynomial models, features which are not found in the existing literature.

III. Quad-Tree Motion Modeling

We begin with a hierarchical description of motion, based on disjoint blocks. At each level k in the hierarchy, the frame is partitioned into blocks $\mathcal{B}_{\mathbf{x}}^{(k)}$ each of size $S2^{-k} \times S2^{-k}$. We write $M_{\mathbf{x}}^{(k)}$ to refer to the motion model for the block region $\mathcal{B}_{\mathbf{x}}^{(k)}$. Initially we consider the case of modeling motion as pure translation with bi-directional prediction modes, which means $M_{\mathbf{x}}^{(k)}$ can describe forward-only, backward-only or bi-directional motion with respect to two reference frames. Later, we consider higher order motion models, allowing $M_{\mathbf{x}}^{(k)}$ to describe linear and affine motion flows in addition to just pure translation.

To create a quad-tree, we start with this hierarchical description of motion over K levels. We use the notation $\mathbf{b}$ to refer to a node with index $\mathbf{x_b}$ and level $k_\mathbf{b}$, that is $\mathbf{b} = (\mathbf{x_b}, k_\mathbf{b})$. A quad-tree is formed by identifying each node $\mathbf{b}$, as the parent of the four nodes at level $k_\mathbf{b} + 1$ whose blocks partition $\mathcal{B}_{\mathbf{x_b}}^{(k_\mathbf{b})}$. If a given node $\mathbf{b}$ in the tree is identified as a leaf node then this means that the descendants of $\mathbf{b}$ have been discarded (or pruned) so that $M_{\mathbf{x_b}}^{(k_\mathbf{b})}$ is the most detailed model available for regions within block $\mathcal{B}_{\mathbf{x_b}}^{(k_\mathbf{b})}$.

At the top most level in the tree hierarchy, all nodes are labeled as either leaf or branch nodes. While leaf nodes represent motion models, branch nodes indicate that motion is represented by descendant nodes at lower levels. The immediate descendants of a branch node, located a the next lower level, may in turn be leaves or branches. This tree structure can continue until reaching the lowest level, $k = K$.

We use a quad-tree of 4 levels, with block sizes ranging from $S \times S = 32 \times 32$ at level $k = 0$, the top most level in the hierarchy, to 4×4 at level $k = K = 3$, the lowest level in the tree.

In our paper, we incorporate two motion vector prediction strategies for coding motion. First is the H.264 spatial motion vector prediction strategy, which is used to code motion vectors of all leaf nodes. In this paper we also explore hierarchical motion coding which requires a different motion vector prediction strategy; various approaches to hierarchical

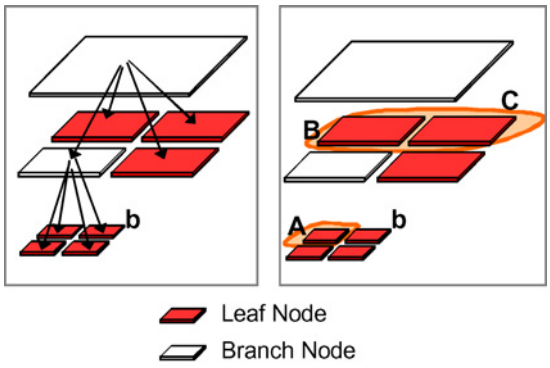

Fig. 1. Examples of motion vector prediction strategies. (Left) hierarchical prediction. (Right) spatial prediction.

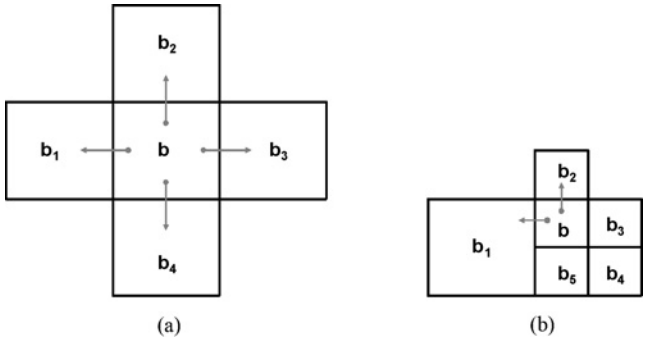

Fig. 2. Potential merge examples. (a) All nodes are leaf nodes at the top most level in the quad-tree; for node **b**, the set of merge candidates include $\mathcal{T}_{\mathbf{b}} = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4\}$ such that $k_b = k_{b_i} = 0, 1 \leq i \leq 4$. (b) Nodes from different levels in the quad-tree; for node **b**, the set of merge candidates include $\mathcal{T}_{\mathbf{b}} = \{\mathbf{b}_1, \mathbf{b}_2\}$; nodes $\mathbf{b}_3$ and $\mathbf{b}_5$ are excluded from the set as they belong to the same parent as the current node **b**.

coding have been explored by earlier efforts, examples of schemes that are similar to the one considered in this paper include [14], [15], [17]. For the hierarchical case, motion vectors of leaf nodes at a particular level are coded with respect to a vector from the immediate parent (branch) node; with the predictors at the highest level taken as (0, 0). Note that for hierarchical coding, branch nodes are also required to convey motion information, as the motion vector of a branch node is used as a reference for coding motion vectors of immediate descendants. Although this introduces some redundancy, it enables scalable access to the motion information and as we will show in the later sections, this redundancy can be reduced by incorporating branch nodes into the merging process.

Examples of the two motion vector prediction strategies are shown in Fig. 1. The diagram on the left illustrates the case of hierarchical coding; the motion vector of the node labeled as **b** is coded differentially with respect to the parent branch node located at the next higher level. This hierarchical coding strategy continues upward to the top most level in the tree. The diagram on the right provides an example of spatial prediction where the motion vector of node **b** is coded differentially with respect to a median predictor obtained from spatially neighboring leaf nodes [i.e., MEDIAN(A, B, C)].

A. Pruning Algorithm

R-D optimization seeks to minimize distortion D_f with a constraint that rate R_f should not exceed a specified maximum value $R_{f,\max}$; that is

$$\begin{aligned} &\min\{D_f\} \\ \text{subject to } R_f \quad &\leq R_{f,\max}. \end{aligned} \tag{1}$$

This constrained problem is typically solved by converting it to an unconstrained problem by employing the Lagrangian multiplier method. This is illustrated in (2) where the goal of (1) has been converted to minimizing the Lagrangian cost function where λ is referred to as the Lagrangian multiplier and is equal to the negative slope of the R-D curve, that is $\lambda = -\partial D_f / \partial R_f$

$$\min\{D_f + \lambda R_f\}. \tag{2}$$

Given an operational R-D slope of $-\lambda$, the quad-tree algorithm performs tree pruning to produce a quad-tree structure that minimizes the Lagrangian cost objective $D_f + \lambda R_f$, where D_f and R_f represent the total distortion and total number of bits associated with coding the whole motion field for a particular frame. To ensure global optimality,[1] tree pruning is conducted in a bottom-up manner, starting at the bottom of the tree and sequentially progressing upward to the root. Pruning is performed by comparing the Lagrangian cost of a parent node with its four children (including unpruned descendants). For a given node **b** we write $D_{f,\mathbf{b}}$ and $R_{\mathbf{b}}$ for the distortion and rate associated with representing block $\mathcal{B}_{\mathbf{x_b}}^{(k_\mathbf{b})}$ with motion model $M_{\mathbf{x_b}}^{(k_\mathbf{b})}$; and we use the terms $\bar{D}_{f,\mathbf{b}}$ and $\bar{R}_{\mathbf{b}}$ to refer to the distortion and rate associated with representing block $\mathcal{B}_{\mathbf{x_b}}^{(k_\mathbf{b})}$ using an optimally pruned set of motion models from node **b** and its descendants. Using this notation, we can describe the pruning process as follows. Given a parent node **p**, at any level in the hierarchical tree structure, the four children $\mathbf{c}_i$, $1 \leq i \leq 4$, are pruned away if [1]

$$\sum_{i=1}^{4} \left[\bar{D}_{f,\mathbf{c}_i} + \lambda \bar{R}_{\mathbf{c}_i}\right] \geq D_{f,\mathbf{p}} + \lambda R_{\mathbf{p}}. \tag{3}$$

When pruning occurs, $\bar{D}_{f,\mathbf{p}} = D_{f,\mathbf{p}}$ and $\bar{R}_{\mathbf{p}} = R_{\mathbf{p}}$. Otherwise $\bar{D}_{f,\mathbf{p}} = \sum_{i=1}^{4} \bar{D}_{f,\mathbf{c}_i}$ and $\bar{R}_{\mathbf{p}} = R_{\mathbf{p}}^{Branch} + \sum_{i=1}^{4} \bar{R}_{\mathbf{c}_i}$; the term $R_{\mathbf{p}}^{Branch}$ represents the rate of the parent node for the case of hierarchical coding, for the alternative case of spatial prediction $R_{\mathbf{p}}^{Branch} = 0$ at all times.[2] When the recursive application of this pruning step has been completed, the total distortion D_f and rate R_f is equal to the respective sum of all $\bar{D}_{f,\mathbf{b}}$ and $\bar{R}_{\mathbf{b}}$ from top level nodes **b** such that $k_{\mathbf{b}} = 0$. Tree pruning yields a globally minimal value for $D_f + \lambda R_f$ for hierarchical coding; while it is somewhat greedy for spatially predictive coding.[3]

[1] A globally optimal solution amongst all possible pruned trees.

[2] For hierarchical coding, the motion parameters of the children are coded differentially with respect to the parent branch node; therefore total rate for a branch node includes the rate of the branch itself and all descendants.

[3] Spatial prediction creates dependencies between the current block being coded and spatially neighboring blocks. The impact that the motion model of the current node has on neighboring nodes that are yet to be coded cannot be determined.

B. Merging Principle

To exploit the redundancy that exists between neighboring nodes of a quad-tree, leaf merging considers the possibility of jointly coding and optimizing neighboring nodes that belong to different parents.

Given a R-D optimally pruned quad-tree, merging is performed by visiting each node in a certain order and applying a set of rules to identify possible merge targets. For a given leaf node $\mathbf{b}$, we identify a set $\mathcal{T}_{\mathbf{b}}$ of possible merge targets $\{\mathbf{b}_i\}$ such that $\mathbf{b}_i$ is a spatially neighboring leaf node located to the top, bottom, left or right of $\mathbf{b}$. As in [2], for a leaf node $\mathbf{b}$, a neighboring leaf node $\mathbf{b}_i$ is only considered a valid merge target if either the neighboring node is located at a higher level in the tree (i.e., $k_{\mathbf{b}_i} < k_{\mathbf{b}}$), or if located at the same level (i.e., $k_{\mathbf{b}_i} = k_{\mathbf{b}}$), the neighboring node belongs to a different immediate parent. This restriction that nodes with the same parent cannot merge together ensures that the merging phase cannot undo the R-D optimizing decisions made during the initial tree pruning phase—otherwise needless inefficiency would be introduced into the overall signalling scheme. Furthermore, the rule that a node may only merge with a neighboring node at the same or higher level ensures that hierarchical decoding is possible. Examples of valid merge candidates, for two different cases, are illustrated in Fig. 2(a) and (b). A leaf node $\mathbf{b}$ could potentially merge with at most one of the merge targets $\mathbf{b}_i$ in the set $\mathcal{T}_{\mathbf{b}}$ to produce a new merged region. To decide on merging, we first calculate the impact of merge on the overall Lagrangian cost $D_f + \lambda R_f$, taking into account distortion of the merged region, motion model coding cost and the cost of signalling merge information. Merging is allowed to take place only if it reduces the overall Lagrangian cost. This ensures that each modification to the original pruned quad-tree will only serve to reduce the overall Lagrangian functional $D_f + \lambda R_f$.

In our investigations, for each potential merge between $\mathbf{b}$ and $\mathbf{b}_i \in \mathcal{T}_{\mathbf{b}}$, three avenues are always explored for calculating the merged motion model, these being: 1) using the motion model of node $\mathbf{b}$ for the potential merged region; 2) using the motion model of node $\mathbf{b}_i$ for the potential merged region; and 3) calculating new motion vectors for the merged region, using a weighted average of originally estimated block motion vectors from all descendant nodes of the region at a given level. Finally, from all the explored options, the motion model that is most beneficial in terms of Lagrangian cost is chosen for the potential merge region.

To calculate the cost of a merge, the number of bits required to signal a merge decision and merge direction needs to be determined. As followed in [2] for a leaf node $\mathbf{b}$, we first determine the set of possible merge targets $\mathcal{T}_{\mathbf{b}}$. If $\mathcal{T}_{\mathbf{b}}$ is empty then no further action is required. If $\mathcal{T}_{\mathbf{b}}$ contains at least one target then a binary merge flag is required to signal a "merge" decision or a "no merge" decision. If the merge flag signals a "merge" decision then the merge direction (from leaf $\mathbf{b}$ to either top, bottom, left or right merge target $\mathbf{b}_i$) needs to be communicated. The number of bits required is either 0, 1, or 2 corresponding to the number of targets in $\mathcal{T}_{\mathbf{b}}$ being 1, 2 or greater. Due to the rules used for selecting merge targets, the number of targets in $\mathcal{T}_{\mathbf{b}}$ cannot exceed 4. These merge coding costs are incorporated into the Lagrangian cost function to determine the benefit of merging.

The order in which each node in the quad-tree is visited and tested for potential merging is an optimization consideration; in our work, merging is performed in a single raster-scan pass within each level of the tree, starting with the lowest level whose leaves are the smallest.

The procedure discussed so far for merging can be used for both spatial motion vector prediction (which relates to the H.264 coding scenario) and hierarchical coding of motion. Note that for the case of hierarchical coding, branch nodes also need to convey motion parameters since we employ differential motion vector coding between child and parent. For hierarchical coding therefore, branch nodes are also included in the merging process with the only change being in the rules used to identify possible merge targets. For a branch node $\hat{\mathbf{b}}$, possible merge targets that can be included in the set $\mathcal{T}_{\hat{\mathbf{b}}}$ are neighboring leaf blocks $\mathbf{b}_i$ that are located at a higher level than $\hat{\mathbf{b}}$ (i.e., $k_{\mathbf{b}_i} < k_{\hat{\mathbf{b}}}$) and neighboring branch nodes $\hat{\mathbf{b}}_i$ or leaf nodes $\mathbf{b}_i$ that are at the same level as $\hat{\mathbf{b}}$ but belonging to a different immediate parent. For a leaf node $\mathbf{b}$, in addition to the possible leaf targets $\mathbf{b}_i$ that can be included in $\mathcal{T}_{\mathbf{b}}$, branch nodes $\hat{\mathbf{b}}_i$ at the same level as $\mathbf{b}$ but belonging to a different immediate parent are also included. All other aspects of the merging process remains the same, thereby allowing hierarchical coding to be incorporated with minimal changes to the merging algorithm.

C. Motion Signalling

We now consider motion signalling for the merged quad-tree for both spatial predictive and hierarchical coding schemes. We define a node as being a member of a merged region if either: 1) the current node signals a merge to a neighboring node, or 2) a neighboring node signals a merge into the current node.

For both coding schemes, nodes that are not members of a merged region need to communicate their own motion model. If however, a node is a member of a merged region then motion information is transmitted for only a single node in that region; we refer to this node as the anchor node. The way in which we identify the anchor node for spatial predictive and hierarchical coding schemes differ; however the rules for both cases ensure that the anchor node can be implicitly identified.

For the hierarchical case, the anchor node is the only member node of the region that is not signaled as being merged. An example of an anchor node for the case of hierarchical coding is given in Fig. 3(a). This anchor node inevitably has the maximum block size of member nodes in the region thereby ensuring that decoding can be performed hierarchically.

For spatially predictive coding, a different methodology is required to identify the anchor node, to make certain that the motion vector prediction strategy remains valid. For spatial prediction, the motion of a given block is coded with respect to its neighboring blocks located above and to the left. For a merged region, the anchor node needs to be chosen so that the spatial neighboring blocks used for predicting its motion, are all located outside the region. To meet this criterion, the anchor

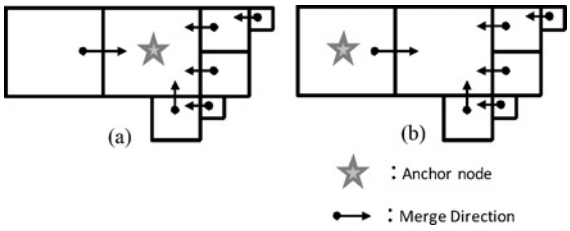

Fig. 3. Example of a merged region and its implicitly defined anchor node for the cases of (a) hierarchical and (b) spatial prediction.

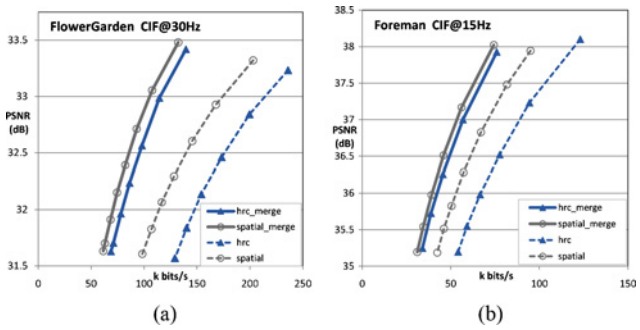

Fig. 4. Rate-distortion improvements achieved by performing merging, for the two cases of spatial and hierarchical motion prediction for (a) *Flower Garden* and (b) *Foreman* sequences.

node for spatially predictive coding is defined to be the first node in the merged region that is encountered during decoding; this is the node which appears first in the raster coding order, regardless of its size. In the example of Fig. 3(b), the anchor node, identified by the *star* sign, is the top-left block of the merged region.

For both cases of spatial and hierarchical prediction, a VLC scheme based on Exponential Golomb codes (as employed in H.264) is used to communicate motion vector difference values.

D. Contrasting Merging with Spatial Prediction

Spatial prediction alone (as used by H.264 and other ITU and ISO/MPEG coders), lacks the capability to explicitly identify the neighboring node whose motion description should be shared by a current one; merging allows this explicit signalling to be performed. Merging also explicitly identifies merged regions, which enables joint optimization of the motion model to be performed over the whole region. Spatial prediction provides only a loose association of neighboring nodes.

E. R-D Performance Results

We now present experimental results illustrating the benefits of applying merging to quad-tree structures. Results presented in Fig. 4(a) relate to 50 frames of the *Flower Garden* sequence at CIF resolution and 30 frames/s. In Fig. 4(b) results for 50 frames of the *Foreman* sequence at CIF resolution and 15 frames/s are shown. In all the graphs, the rate corresponds to the rate required to code the quad-tree structure, block motion and any merge information. The distortion or PSNR refers to the Y component of the motion compensated image. In Fig. 4(a) and (b), the graph labeled *"hrc"* corresponds to the R-D performance of the hierarchically coded quad-tree without any node merging and the graph labeled *"hrc_merge"* shows the improved performance attained by introducing merging to the quad-tree motion representation. For the *Flower Garden* and *Foreman* sequences, respective bit rate savings of approximately 45% and 35% are observed when merging is applied to the hierarchically coded quad-tree. The graphs in the figures labeled *"spatial"* and *"spatial_merge"* correspond to the respective R-D performance before and after merging, for quad-tree structures employing spatially predictive coding. For this case, results show bit rate savings of approximately 35% and 25% for the *Flower Garden* and *Foreman* sequences respectively. Note that once merging is included the performance of hierarchical motion representation can be brought close to that achieved by spatial prediction with merging. Node merging therefore opens up possibilities for efficient hierarchical motion modeling which is attractive for scalable video coding.

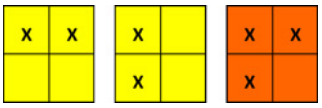

Fig. 5. Nominal locations within a block used in representing (Left) horizontal linear model (center) vertical linear model and (right) affine model.

IV. Hierarchical and Polynomial Motion Modeling

In an attempt to further improve performance of the hierarchical motion representation, we introduce the option of representing motion by polynomial motion models. We note that polynomial motion models [18], [19] and more recently elastic motion models [20] have been the subject of previous research where the main focus has been on applying these higher order motion models to improve rate-distortion performance. Our primary motivation in introducing polynomial motion models is to encourage the formation of larger regions during the merging process so as to create smoother motion representations with reduced artificial block discontinuities. Since polynomial models are able to represent more complex motion flows, they are more suited to describing the associated motion of larger regions.

A. Motion Models

We use the notation $\mathbf{v}_{\mathbf{b}}^{(k)}$ to refer to a motion vector belonging to node $\mathbf{b}$ at level k. At the lowest level of the hierarchy $k = K$, motion is always modeled as pure translation, with motion vectors $\mathbf{v}_{\mathbf{b}}^{(K)}$. At all other levels, $M_{\mathbf{b}}^{(k)}$ can represent motion with 2, 4 or 6 parameters, corresponding to translation, linear and affine flows. In practice, $M_{\mathbf{b}}^{(k)}$ is represented through 1, 2 or 3 motion vectors, $\mathbf{v}_{\mathbf{b},1}^{(k)}$, $\mathbf{v}_{\mathbf{b},2}^{(k)}$, $\mathbf{v}_{\mathbf{b},3}^{(k)}$, with *nominal* locations $\eta_{\mathbf{b},1}^{(k)}$, $\eta_{\mathbf{b},2}^{(k)}$, $\eta_{\mathbf{b},3}^{(k)}$ as depicted in Fig. 5.

During the pruning phase, the parameters $\mathbf{v}_{\mathbf{b},1}^{(k)}$, $\mathbf{v}_{\mathbf{b},2}^{(k)}$, $\mathbf{v}_{\mathbf{b},3}^{(k)}$ of the motion model $M_{\mathbf{b}}^{(k)}$ for block $\mathcal{B}_{\mathbf{b}}^{(k)}$ are obtained by a weighted least squares fitting procedure, based upon a collection of initial translational motion estimates on blocks

of size $(S)2^{-j} \times (S)2^{-j}$ for various j in the range $k+1$ to K. The weights used in this procedure are selected so as to give relatively greater importance to locations that are more sensitive to motion compensation errors, such as edges and highly textured regions. Lower importance is assigned to locations that can tolerate motion compensation error, such as flat or smooth regions.

During the merging phase new motion models (single vector, linear and affine) are determined for each merged region based on the weighted least squares fitting procedure using block motion vectors from all descendant nodes of the region at a given level.

B. Motion Compensation

Motion compensation based on model $M_{\mathbf{b}}^{(k)}$ is performed by first generating a set of corresponding motion vectors for each descendants at level K (i.e., each 4×4 block that constitutes the region $\mathcal{B}_{\mathbf{b}}^{(k)}$), according to

$$\mathbf{v}_{\mathbf{b}'}^{(3)} = \sum_{i=1}^{3} \alpha_{\mathbf{b}',i} \mathbf{v}_{\mathbf{b},i}^{(k)}, \quad \forall \mathcal{B}_{\mathbf{b}'}^{(3)} \subset \mathcal{B}_{\mathbf{b}}^{(k)}$$

where the $\alpha_{\mathbf{b}',i}$ depend on the motion model and the central location of block $\mathbf{b}'$. In this way, conventional motion compensation operations can be employed on 4×4 blocks when performing motion compensation for larger blocks with polynomial motion models.

C. R-D Performance with Motion Models

In Fig. 6(a) and (b), the R-D performance of hierarchically coded merged quad-tree representations is shown with and without polynomial motion models. The introduction of polynomial models means that for forward-only and backward-only prediction modes, the corresponding motion model $M_{\mathbf{b}}^{(k)}$ can be a single vector, linear or affine. For the bi-directional case, the forward and backward motion models can each be drawn from any one of the three model types. We employ adaptive arithmetic coding to code the motion model type of a block.

For the *Flower Garden* sequence, a maximum bit rate saving of approximately 17% is observed when comparing the merged quad-tree with polynomial motion models *"hrc_merge_models"* with that of the merged quad-tree employing only translational motion vectors *"hrc_merge"*. A similar trend can be seen for the *Foreman* sequence although the gains due to introducing polynomial motion models are generally lower; a maximum bit rate saving of approximately 8% is observed.

D. Comparative Performance Results

In Fig. 6(a) and (b), we compare the R-D performance of conventional tree based motion modeling schemes (*"h264_motion"* and *"hrc"*) with our hierarchically coded general quad-tree structure which incorporates both polynomial motion models and node merging (*"hrc_merge_models"*). The graphs labeled *"h264_motion"* relate to coding frame motion as specified by the H264 standard [5].[4] The graphs labeled *"hrc"* were presented earlier in Section III; these graphs relate to the performance of a typical quad-tree, without any merging or polynomial motion models and is reproduced here for reference.

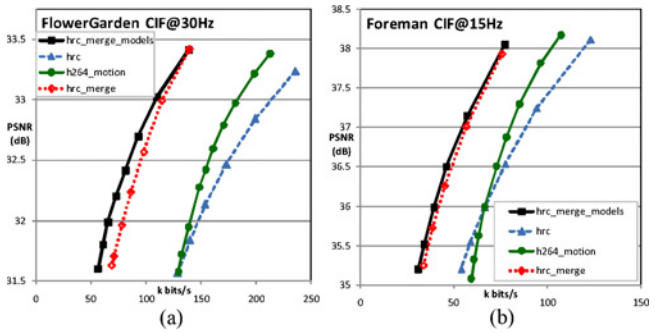

Fig. 6. Influence of merging and polynomial motion models on R-D performance for (a) *Flower Garden* and (b) *Foreman* sequences.

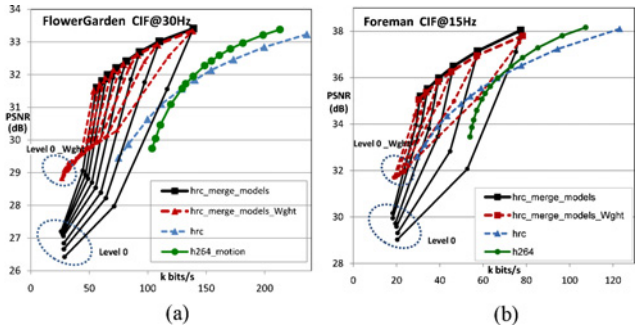

Fig. 7. Scalability performance for (a) *Flower Garden* and (b) *Foreman* sequences.

In comparison to the H.264 model, results show substantial bit rate savings for the merged quad-tree representation; for the *Flower Garden* sequence, bit rate savings in the range of 35–55% are evident, while for the *Foreman* sequence savings of approximately 25–45% are observed. It is interesting to note that in both Fig. 6(a) and (b) the H.264 model, in general, displays better performance than the hierarchically coded conventional quad-tree motion model (*"hrc"*). As previously discussed, this behavior is to be expected, since spatial prediction of leaf nodes is more efficient than hierarchical coding in the absence of merging. However the performance gain of *"h264_motion"* in comparison to *"hrc"* diminishes at lower bit rates. This is because at low rates, many leaf nodes are modeled at the top most level in the tree hierarchy and for *"hrc"* this corresponds to blocks of size 32×32 while *"h264_motion"* is limited to a maximum block size of 16×16. Due to the lower number of branch nodes and the larger block size available, at low bit rates the relative performance of *"hrc"* improves.

The introduction of merging also produces a smoother motion field with less artificial discontinuities; results illustrating this fact have been presented in our recent work [21].

[4]This corresponds to communicating the structure and content of the H.264 motion tree employing the motion vector prediction and coding strategies as specified by the standard.

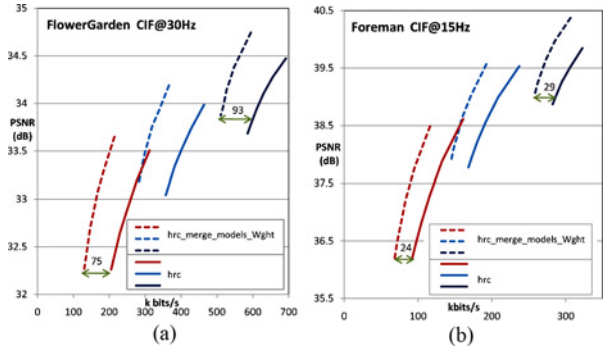

Fig. 8. Performance of coding motion and corresponding residue frames for (a) *Flower Garden* and (b) *Foreman* sequences.

V. Scalable Motion Modeling

Having established the efficiency of the merged quad-tree model, we now explore its scalability performance. Hierarchical coding of the motion vectors and the particular merging rules we follow ensure that top-down hierarchal decoding can be performed. Results of scalable decoding indicate a rapid degradation in the quality of decoded images when the decoding process is terminated at an intermediate level. This is because during both the pruning and merging phase, the goal so far has been to achieve the most efficient representation at full resolution by minimizing the Lagrangian cost $J_f = D_f + \lambda R_f$. Since only leaf nodes contribute to the total distortion D_f, scalability performance is not taken into account.

A. Scalability Objective

To improve scalability performance, we modify the Lagrangian cost function. When terminating decoding at an intermediate resolution level, motion compensation is performed using leaf nodes that may already be available; in those cases where leaf nodes are not available, information contained in branch nodes is utilized. If we knew ahead of time that decoding would be terminated at level k, the relevant Lagrangian cost should be $J_f^{(k)} = D_f^{(k)} + \lambda R_f^{(k)}$, where $D_f^{(k)}$ represents the total distortion of all nodes for which motion compensation is performed. This corresponds to: 1) leaf nodes in the tree, located at level k or above, and 2) branch nodes in the tree, located at level k. The term $R_f^{(k)}$ represents the bits required to code the motion parameters, as well as the quad-tree structure and any merge information, corresponding to nodes at level k or above. We now formulate a modified cost objective $\hat{J}_f$, composed of a weighted summation of the individual costs $J_f^{(k)}$ for each level k of the quad-tree; specifically

$$\hat{J}_f = \sum_k \alpha_k J_f^{(k)} = \sum_k \alpha_k D_f^{(k)} + \lambda \sum_k \alpha_k R_f^{(k)}. \quad (4)$$

Here, α_k represents the weights or priority values assigned to each level, such that $\sum_k \alpha_k = 1$. In the following, we use $\hat{D}_f$ and $\hat{R}_f$ to denote the total distortion and rate terms of the modified cost objective $\hat{J}_f$, such that $\hat{D}_f = \sum_k \alpha_k D_f^{(k)}$ and $\hat{R}_f = \sum_k \alpha_k R_f^{(k)}$.

We now detail the contribution of a node $\mathbf{b}$ in the quad-tree, to both $\hat{D}_f$ and $\hat{R}_f$. For a leaf node $\mathbf{b}$ at level $k_\mathbf{b}$, the contribution to $\hat{D}_f$ is given by $(\sum_{k_\mathbf{b} \le i} \alpha_i) D_{f,\mathbf{b}}$, where the distortion term $D_{f,\mathbf{b}}$ represents the motion compensated residual energy for the block region $\mathcal{B}_\mathbf{b}$ calculated using the motion parameters of node $\mathbf{b}$. If $\mathbf{b}$ is a branch node in the combined tree then the contribution to $\hat{D}_f$ is given by $\alpha_k D_{f,\mathbf{b}}$. The contribution to $\hat{R}_f$ for a given leaf or branch node $\mathbf{b}$ is equal to $(\sum_{k_\mathbf{b} \le i} \alpha_i) R_\mathbf{b}$, where $R_\mathbf{b}$ represents the number of bits required to signal information associated with node $\mathbf{b}$.

B. Scalability Performance

We now present scalability performance results obtained by terminating the decoding process at each level of the quad-tree model. In Fig. 7(a) and (b), the black solid lines identify the performance of the merged quad-tree at various scales, when the optimization objective ignores all but the full resolution performance. By contrast, the dashed lines identify the performance obtained with the proposed weighted cost objective. Scalable decoding starts at the coarsest level, consisting of 32×32 blocks, which is labeled as "*Level 0*" or "*Level 0_Wght*," as appropriate. As levels 1, 2, and 3 are progressively incorporated into the content which is decoded and used for motion compensation, the performance achieved with the two different optimization objectives eventually arrives at the curves identified as "*hrc_merge_models*" and "*hrc_merge_models_Wght*," respectively. The weights used in our paper correspond to 0.7 for level 3, the finest resolution, and 0.1 for the levels 0, 1 and 2 ($\alpha_0 = \alpha_1 = \alpha_2 = 0.1, \alpha_3 = 0.7$).

When comparing the hierarchical decoding paths of the two merged quad-tree representations, it is clear that the rate-distortion performance achieved at the lower resolution levels is dramatically improved by using the weighted Lagrangian cost objective. For both sequences, gains of approximately 1 to 2 dB can be observed for the lowest two resolution scales (i.e., levels 0 and 1). The small difference observed between the full resolution curves ("*hrc_merge_models*" and "*hrc_merge_models_Wght*") reveals that these scalability improvements are achieved without significant loss in overall efficiency.

For reference, we also show in Fig. 7(a) and (b), results of conventional tree based motion modeling schemes. For most cases, the merged quad-tree with the weighted cost objective outperforms both the *"hrc"* reference and the inherently non-scalable *"h264_motion"* reference, even when decoded at reduced resolutions.

C. Residue Coding

To understand the impact on compression, we code using JPEG2000, the full resolution motion compensated residue frames corresponding to motion modeling cases characterized by "*hrc_merge_models_Wght*" and "*hrc*." For the *Flower Garden* sequence the JPEG2000 bit stream is decoded at rates of 76, 228, and 456 kbits/s; for the *Foreman* sequence decoding occurs at rates of 38, 114, and 228 kbits/s. Results are illustrated in Fig. 8(a) and (b), with the graphs showing the total rate required for coding both motion and residue

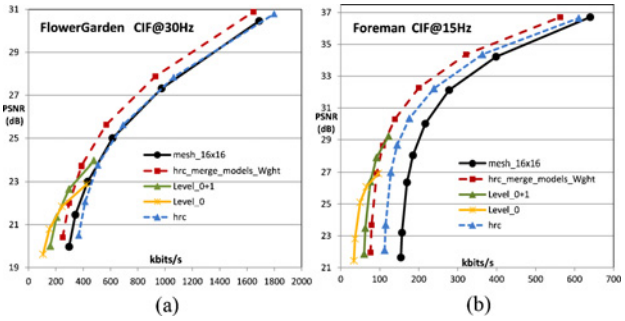

Fig. 9. Performance of the SIV encoder employing different motion modeling schemes; results relate to (a) *Flower Garden* and (b) *Foreman* sequences.

frames and the PSNR referring to the Y component of decoded frames. The superior results of the proposed merged quad-tree motion model validates its applicability to video coding applications; the advantages observed earlier in the motion compensated domain carry across to residue coding. The bit-rate saving achieved for lowest and highest coding rate scenarios are indicated on the graphs.

D. Wavelet-based Video Encoding Results

To evaluate the performance of our proposed merged quad-tree model within a full video encoding framework, we integrate the quad-tree motion model with the wavelet-based scalable interactive video (SIV) codec developed by Secker and Taubman [9]; further details of an implementation of the SIV encoder can be found in [10]. The SIV encoder offers a highly scalable operating environment; due to its open-loop structure the SIV codec is able to avoid some of the problems of drift encountered by typical in-loop DPCM coders [15]. The SIV encoder first conducts motion-compensated lifting steps to implement the temporal wavelet transform; in our work the up-date step of the temporal lifting implementation is skipped. The resulting temporal sub-bands are transformed by a 2-D discrete wavelet transform. To perform temporal filtering, the SIV encoder uses 1/3 wavelet filters while the spatial transform is achieved with 9/7 wavelet filters. In our work the SIV encoder performs 4 levels of temporal decomposition and 4 levels of spatial decomposition. After the transform stage, the resulting spatio-temporal subbands are embedded within the JPEG2000 codestream syntax.

R-D performance of the SIV encoder employing quad-tree motion models are presented in Fig. 9(a) and (b) for the *Flower Garden* and *Foreman* sequences respectively. The bit-rate for these graphs relate to the rate required to decode both texture and motion information; the PSNR value refers to the Y component of decoded frames. The results presented in these graphs relate to two scalable decoding scenarios. In the first case, the motion model is decoded at its full resolution (i.e., with no loss) and the accompanying residue information is decoded at various rates ranging from low to high bit-rate values. The graphs labeled "*hrc_merge_models_Wght*" (red dashed line) relate to the performance of the merged quad-tree. For both sequences the merged quad-tree structures have been optimized for a particular λ value of 15 (i.e., $\lambda = 15$). The graphs labeled "*hrc*" (blue dashed line) refer to corresponding quad-tree representations without merging. Clearly the merged quad-tree representation displays superior performance at all bit rates. The graphs labeled "*mesh_16x16*" refer to the performance of a mesh model as described in [9] where the control points for the mesh lie on a grid with a regular interval spacing of 16×16. The motion information of this mesh model is coded by the SIV encoder as a two component image employing JPEG2000 schemes. The merged quad-tree model performs better at all bit rates compared to the mesh model; bit rate savings of up to 10% is evident for the *Flower Garden* sequence while for the *Foreman* sequence the bit rate savings are even more impressive, typically in the range of 10–50%. For the *Foreman* sequence, the merged quad-tree requires considerably less bits than the more expensive mesh model therefore when decoding at low bit rates the advantage of the merged quad-tree representation becomes more prevalent.

The second scenario presented in the graphs relate to utilizing the scalability feature of the merged quad-tree motion model. The graphs labeled "*Level_0*" (solid orange line) correspond to decoding only layer 0 of the quad-tree. This means only the coarsest motion description is decoded; the accompanying residue information which is coded assuming full resolution motion, is progressively decoded in the same manner as before. When operating at low bit rates, it is some times beneficial to sacrifice the bits dedicated to motion so as to allocate more bits for texture coding. This is evident when comparing graphs "*Level_0*" with "*hrc_merge_models_Wght*"; note that the graphs labeled "*Level_0*" are plotted for only the low bit rate range. The graphs labeled "*Level_0+1*" (solid green line) relate to decoding just the first two layers of quad-tree hierarchy. Similar observations can be made for this case; at low bit rates the performance of "*Level_0+1*" is better than the full resolution model "*hrc_merge_models_Wght.*" These results indicate the benefits of operating in a highly scalable environment where texture and accompanying motion information can be independently decoded at different rates.

VI. Theoretical Study

We now present a theoretical analysis of the R-D performance of quad-tree motion models. The goals of this analysis are: 1) to derive the asymptotic R-D behavior of both the pruned quad-tree and the pruned quad-tree with merging, and 2) to approximate the comparative advantage of leaf merging.

In this theoretical study, we make the assumption that the motion field can be precisely described in a piecewise manner, with the motion of each piece given by a 2-D polynomial model and the contour of each piece given by a smooth boundary. This piecewise description, while allowing for an analytical examination of the performance of the quad-tree, is also a reasonable model for motion occurring in real video sequences. For example, the boundary of each moving object in a video frame could possibly be approximated by a smooth curve and the corresponding motion of the object explained by a 2-D polynomial.

We make some further assumptions in our theoretical paper and we briefly summarize these in the following discussion. We assume that the number of distinct motion regions com-

prising the motion field is finite (typically small) and that these motion regions are spatially slow varying in nature. We assume we are operating in a high rate (i.e., small quantization error) coding range with no predictive coding of the motion models. We also assume that the coefficients of the polynomial motion models have equal statistical distribution. While these assumptions simplify the situation, they allow us to achieve mathematically tractable expressions which provide insights into the behavior of quad-tree motion models.

In the following discussion, we use T to represent the dimension of a square video frame. This means that for a node $\mathbf{b}$ in the quad-tree, the size of the corresponding spatial block $\mathcal{B}_{\mathbf{b}}$ can be specified as $T2^{-k_{\mathbf{b}}} \times T2^{-k_{\mathbf{b}}}$ where $k_{\mathbf{b}}$ refers to the level at which node $\mathbf{b}$ is located in the tree.

We first focus on the R-D characteristics of a single node (or block) and then develop an approximation of the R-D performance of the pruned quad-tree structure. Later, we extend this R-D model to include leaf merging, in order to understand its benefits.

A. Motion Compensation

Let $m_{r\to t,b}(\mathbf{s})$ signify a polynomial model that describes the motion for a block region $\mathcal{B}_b$ of a target frame $F_t(\mathbf{s})$ with respect to a reference frame $F_r(\mathbf{s})$. We use the notation $F_{r\to t}(\mathbf{s})$ to refer to the prediction that is attained by performing motion compensation on the reference frame $F_r(\mathbf{s})$ in accordance with motion model $m_{r\to t,b}(\mathbf{s})$, that is $F_{r\to t}(\mathbf{s}) = F_r(\mathbf{s} - m_{r\to t,b}(\mathbf{s}))$, $\forall \mathbf{s}\epsilon\mathcal{B}_b$. If the difference between $F_r(\mathbf{s})$ and $F_t(\mathbf{s})$ can be described purely in terms of the motion that has occurred between the two frames then $F_{r\to t}(\mathbf{s}) = F_t(\mathbf{s})$. If, however, new information appears in $F_t(\mathbf{s})$ that cannot be explained by motion compensation of $F_r(\mathbf{s})$, (e.g., random noise), then $F_{r\to t}(\mathbf{s})$ becomes only an approximation to $F_t(\mathbf{s})$, that is $F_{r\to t}(\mathbf{s}) \approx F_t(\mathbf{s})$. We define $d(\mathbf{s})$ to be the residue signal, such that $d(\mathbf{s}) = F_{r\to t}(\mathbf{s}) - F_t(\mathbf{s})$.

B. Quantized Polynomial Motion Model

In this section, we derive the R-D characteristics of a quantized, 2-D polynomial motion model that represents the underlying motion of a single block region. This relates to the R-D behavior of a leaf node of the quad-tree where the node is located completely within a single piece of the piecewise description of the motion field. In our derivation, we first focus on errors resulting in the motion description due to quantization of the motion model. The effect of these errors on the motion compensated prediction is then analyzed to obtain a measure of total distortion in the image domain.

Let $m^q_{r\to t,b}(\mathbf{s})$ represent a quantized version of the motion model. We use $\delta_{m,b}(\mathbf{s})$ to refer to the difference between $m_{r\to t,b}(\mathbf{s})$ and $m^q_{r\to t,b}(\mathbf{s})$; this is shown in (5). The distortion of the motion description due to quantization, as measured in terms of total squared error, is denoted as $D^q_{m,b}$ and is given by (6)

$$\delta^q_{m,b}(\mathbf{s}) = m_{r\to t,b}(\mathbf{s}) - m^q_{r\to t,b}(\mathbf{s}) \ \forall \mathbf{s}\epsilon\mathcal{B}_b \tag{5}$$

$$D^q_{m,b} = \int\int_{\mathcal{B}_{\mathbf{b}}} \left\|\delta^q_{m,b}(\mathbf{s})\right\|^2 d\mathbf{s}. \tag{6}$$

A Pth order 2-D polynomial motion model $m_{r\to t,b}(\mathbf{s})$, for a block region $\mathcal{B}_b$, can be expressed in terms of a set of orthonormal basis functions, as follows:

$$m_{r\to t,b}(\mathbf{s}) = \sum_{i=0}^{P}\sum_{j=0}^{P-i} a_{ij}s_1^i s_2^j = \sum_{i=0}^{P}\sum_{j=0}^{P-i} l_{ij}\varphi_{ij}(\mathbf{s}) \tag{7}$$

where $\varphi_{ij}(\mathbf{s})$ refers to the 2-D polynomial basis functions over the block region; and l_{ij} represents the corresponding coefficients of the expansion. The basis functions $\varphi_{ij}(\mathbf{s})$ represent an orthonormalization of the standard polynomial basis set $\{s_1^i s_2^j, 0 \le i, j, i+j \le P\}$. As illustrated in [22], Legendre polynomials provide one possible realization of the expansion shown in (7). Using l^q_{ij} to refer to the quantized coefficients, the total distortion $D^q_{m,b}$ resulting from quantization of the polynomial model, can now be expressed as

$$\begin{aligned} D^q_{m,b} &= \sum_{i=0}^{P}\sum_{j=0}^{P-i} \left(l_{ij} - l^q_{ij}\right)^2 \int\int_{\mathcal{B}_{\mathbf{b}}} \varphi^2_{ij}(\mathbf{s})d\mathbf{s} \\ &= \sum_{i=0}^{P}\sum_{j=0}^{P-i} \left(l_{ij} - l^q_{ij}\right)^2. \end{aligned}$$

Defining A_m as a bound on the magnitude of the polynomial $m_{r\to t,b}(\mathbf{s})$, we can obtain a bound on the magnitude of all coefficients l_{ij} as illustrated below

$$\begin{aligned} A_m^2 T^2 2^{-2k_{\mathbf{b}}} &\ge \int\int_{\mathcal{B}_{\mathbf{b}}} m^2_{r\to t,b}(\mathbf{s})d\mathbf{s} \\ &= \sum_{i=0}^{P}\sum_{j=0}^{P-i} \left|l_{ij}\right|^2 \ge \left|l_{ij}\right|^2 \ \forall i, j \\ &\Longrightarrow \left|l_{ij}\right| \le A_m T 2^{-k_{\mathbf{b}}}. \end{aligned} \tag{8}$$

Assuming uniform quantization of l_{ij} over the range $-A_m T2^{-k_{\mathbf{b}}}$ to $A_m T2^{-k_{\mathbf{b}}}$, with rate r_{ij} bits, the expected quantization error can be estimated from the usual high rate model [23]

$$E\left[\left(l_{ij} - l^q_{ij}\right)^2\right] \le \frac{(2A_m T2^{-k_{\mathbf{b}}}2^{-r_{ij}})^2}{12} = \frac{A_m^2 T^2 2^{-2k_{\mathbf{b}}}2^{-2r_{ij}}}{3}.$$

Since the polynomial basis vectors are orthonormal and all coefficients have the same range bounds,[5] optimal allocation of a given global bit budget R_b is achieved by sharing the available bit budget equally among all coefficients. As there are $\frac{(P+1)(P+2)}{2}$ coefficients in total, the rate r_{ij} for each coefficient is given by

$$r_{ij} = \frac{2R_b}{(P+1)(P+2)}.$$

The expected total squared error can now be expressed as

$$\begin{aligned} E[D^q_{m,b}] &\le \frac{1}{3}A_m^2 T^2 2^{-2k_{\mathbf{b}}} \sum_{i=0}^{P}\sum_{j=0}^{P-i} 2^{-2r_{ij}} \\ &= \frac{1}{3}A_m^2 T^2 2^{-2k_{\mathbf{b}}} \left[\frac{(P+1)(P+2)}{2}\right] 2^{\frac{-4R_b}{(P+1)(P+2)}}. \end{aligned}$$

[5]Of course, in practice we would expect the coefficients to have different statistical distributions, but the current conservative analysis uses only the worst case bounds given by (8).

Quantization errors in the motion representation result in displacement errors in the motion compensated domain; therefore, at any given location, performing motion compensation with $m^q_{r\to t,b}(\mathbf{s})$ is equivalent to performing motion compensation with $m_{r\to t,b}(\mathbf{s})$ on a reference frame $F_r(\mathbf{s})$ displaced by $\delta^q_{m,b}(\mathbf{s})$; specifically

$$F^{\delta(s)}_{r\to t}(\mathbf{s}) = F_r(\mathbf{s} - m_{r\to t,b}(\mathbf{s}) - \delta^q_{m,b}(\mathbf{s}))\ \forall \mathbf{s}\epsilon\mathcal{B}_b.$$

Assuming that the motion field is spatially slow varying, we can approximate $m_{r\to t,b}(\mathbf{s})$ by $m_{r\to t,b}(\mathbf{s}-\delta^q_{m,b}(\mathbf{s}))$, so that

$$\begin{aligned} F^{\delta(s)}_{r\to t}(\mathbf{s}) &\approx F_r(\mathbf{s} - m_{r\to t,b}(\mathbf{s}-\delta^q_{m,b}(\mathbf{s})) - \delta^q_{m,b}(\mathbf{s})] \\ &= F_{r\to t}(\mathbf{s}-\delta^q_{m,b}(\mathbf{s})). \end{aligned}$$

At a particular location, the error present in the motion compensated frame can now be approximated as

$$d^{\delta(s)}(\mathbf{s}) \approx \left(F_{r\to t}(\mathbf{s}) - F_{r\to t}(\mathbf{s}-\delta^q_{m,b}(\mathbf{s}))\right).$$

Using $S_{r\to t}(\omega)$ to denote the power spectral density function of $F_{r\to t}(\mathbf{s})$, and noting that a shift of δ in the spatial domain is equivalent a linear phase shift in the Fourier domain, we can write the expected squared error in the image domain as

$$\begin{aligned} E\left[\left(d^\delta(\mathbf{s})\right)^2 \mid \delta\right] &= \frac{1}{(2\pi)^2}\int\int S_{r\to t}(\omega)\left|1-e^{-j\omega^t\delta}\right|^2 d\omega \quad (9) \\ &\approx \frac{1}{(2\pi)^2}\int\int S_{r\to t}(\omega)\cdot(\omega^t\delta)^2 d\omega \quad (10) \end{aligned}$$

where ω denotes the 2-D frequency. The approximation term in (10) is derived using a linear approximation of $\left|1-e^{-j\omega^t\delta}\right|^2$ for small values of $\omega^t\delta$ [24]. Equation (10) is further simplified in [24] to the linear relationship shown below for small and uniform errors in the motion field

$$E\left[\left(d^\delta(\mathbf{s})\right)^2 \mid \delta\right] \approx \Psi_{r\to t}\,\|\delta\|^2$$

where

$$\Psi_{r\to t} = \frac{1}{2(2\pi)^2}\left[\int\int S_{r\to t}(\omega)\omega_1^2 d\omega + \int\int S_{r\to t}(\omega)\omega_2^2 d\omega\right]. \quad (11)$$

The term $\Psi_{r\to t}$, can be regarded as a motion sensitivity factor. We can now derive, the expected total squared error in the motion compensated prediction of a block $\mathcal{B}_b$ as

$$\begin{aligned} D^{\delta(s)}_{f,b}(R_b) &= E\left[\int\int_{\mathcal{B}_\mathbf{b}}\left(d^{\delta(s)}(\mathbf{s})\right)^2 d\mathbf{s}\right] \\ &\approx \int\int_{\mathcal{B}_\mathbf{b}}\Psi_{r\to t}\cdot E\left[\left\|\delta^q_{m,b}(\mathbf{s})\right\|^2\right]\cdot d\mathbf{s} \\ &= \Psi_{r\to t}T^2 2^{-2k_\mathbf{b}} E\left[\left\|\delta^q_{m,b}(\mathbf{s})\right\|^2\right] \\ &= \Psi_{r\to t}E[D^q_{m,b}] \\ &\le \Psi_{r\to t}\frac{1}{3}A_m^2 T^2 2^{-2k_\mathbf{b}}\left[\frac{(P+1)(P+2)}{2}\right]2^{\frac{-4R_b}{(P+1)(P+2)}}. \quad (12) \end{aligned}$$

For notational convenience, we often write the above expression as

$$D^{\delta(s)}_{f,b}(R_b) = \sigma_M^2 T^2 2^{-2k_\mathbf{b}} 2^{\frac{-4R_b}{(P+1)(P+2)}} \quad (13)$$

where $\sigma_M^2 \le \Psi_{r\to t}\frac{1}{3}A_m^2\left[\frac{(P+1)(P+2)}{2}\right]$. While (13) refers to the expected squared error for a block region, we note that the rate-distortion model for individual blocks would in practice vary according to the frequency content of the underlying block as suggested by the expression for the motion sensitivity factor $\Psi_{r\to t}$.

C. R-D Performance of the Pruned Quad-Tree

In this theoretical analysis, each node of the quad-tree is allowed to represent the underlying motion with a single 2-D polynomial model. In cases where the spatial block $\mathcal{B}_\mathbf{b}$ straddles a motion discontinuity boundary, it is not possible to model the underlying motion with a single smooth model. In such cases, the quad-tree algorithm partitions the current block into four equal size blocks, corresponding to the children of the parent node **b**. Among the four children, those nodes that fall completely within one side of the motion discontinuity boundary and are located within a smooth motion region, can be represented by a single 2-D polynomial model; these blocks become leaf nodes. However those blocks that contain motion discontinuity will need to be partitioned again to the next finer level and hence these blocks become branch nodes. This recursive partitioning of the motion field can continue level by level, progressively increasing the accuracy of the quad-tree structure in representing the piecewise smooth motion field. It is evident that any increase in the accuracy of the quad-tree representation is achieved at the expense of coding an ever growing number of leaf nodes in the vicinity of motion discontinuity boundaries. As each leaf node represents an independent smooth motion model, the coding cost of the quad-tree rapidly increases. In this theoretical analysis, we assume that the motion parameters of each leaf node are coded independently, without employing any predictive coding techniques.

We now derive an approximation of the R-D performance of the pruned quad-tree. We first analyze the optimal R-D operating points of leaf nodes in the quad-tree and then determine the depth to which the tree will grow. Finally we derive a total R-D description of the pruned quad-tree in an example setting.

The R-D relationship of leaf nodes which contain no motion discontinuity is represented by the exponentially decaying function shown in (12). R-D optimality dictates that all leaf nodes must operate at the same slope $-\lambda$ on their R-D curves. It is well known that for exponentially decaying R-D functions, this constraint of operating at equal slope $-\lambda$ means all leaf nodes have the same distortion value regardless of the level in the tree to which they belong [23]. That is

$$\begin{aligned} &\frac{\partial D^{\delta(s)}_{f,b}}{\partial R_b} = -\lambda \quad \forall k_\mathbf{b} \ge 0 \\ &\Rightarrow D^{\delta(s)}_{f,b} = \frac{\lambda(P+1)(P+2)}{4\ln 2}. \quad (14) \end{aligned}$$

This means that the rates allocated to nodes **a** and **b** at different levels in the tree, $k_{\mathbf{a}}$and $k_{\mathbf{b}}$, are related according to

$$\Rightarrow R_b = R_a + \frac{(P+1)(P+2)}{2}(k_{\mathbf{a}} - k_{\mathbf{b}}). \tag{15}$$

It follows that the Lagrangian cost of leaf nodes **a** and **b** are related by

$$D_{f,b}^{\delta(s)} + \lambda R_b = D_{f,a}^{\delta(s)} + \lambda R_a + \lambda \left[\frac{(P+1)(P+2)}{2}(k_{\mathbf{a}} - k_{\mathbf{b}})\right]. \tag{16}$$

During the pruning phase, we assume that nodes which contain motion discontinuities are allocated zero rate, since a single polynomial model is unable to describe the underlying motion of the node. For such a node **p**, we use $\sigma_{F_t}^2(T^2 2^{-2k_{\mathbf{p}}})$ as an upper bound for the expected distortion, where $\sigma_{F_t}^2$ represents the variance of the target frame and $T^2 2^{-2k_{\mathbf{p}}}$ is the area of the block $\mathcal{B}_{\mathbf{p}}$. The quad-tree pruning algorithm compares the Lagrangian cost of node **p** with that of its four children, in accordance with (3); node **p** is partitioned if descendant nodes provide a lower Lagrangian cost representation. We use the notation K to refer to the level beyond which it is no longer beneficial to grow the tree; that is K represents the finest resolution level of the pruned quad-tree. We now consider the pruning test of (3) comparing a node **p** at level $k_{\mathbf{p}} = K - 1$ with its four children at level K. Let N_c represent the number of children that are located completely within smooth motion regions and hence can be described by a single polynomial model. The remaining $(4 - N_c)$ children straddle motion boundaries and consequently are allocated zero rate; the distortion of these children is given by $\sigma_{F_t}^2(T^2 2^{-2K})$. Substituting into (3) the R-D relationships for the parent and children, we obtain the inequality shown below to test pruning

$$\begin{aligned}(4 - N_c)\sigma_{F_t}^2(T^2 2^{-2K}) + N_c(D_{f,c}^{\delta(s)}(R_c) + \lambda R_c) \\ \geq 4\ \sigma_{F_t}^2(T^2 2^{-2K})\end{aligned} \tag{17}$$

$$\Rightarrow D_{f,c}^{\delta(s)}(R_c) + \lambda R_c \geq \sigma_{F_t}^2(T^2 2^{-2K}). \tag{18}$$

Equation (18) above implies that children are pruned if the Lagrangian cost of a child node **c** modeling a smooth motion region, is greater than or equal to the corresponding cost of operating with no motion compensation (i.e., at zero rate for node **c**). Conversely, the children are retained at level K if $\sigma_{F_t}^2(T^2 2^{-2K}) > D_{f,c}^{\delta(s)}(R_c) + \lambda R_c$.

To determine the depth to which the quad-tree can grow, we find the terminating condition for tree decomposition. Since $D_{f,c}^{\delta(s)}(R_c)$ depends only on λ in accordance with (14), the terminating condition is the same for all nodes on the boundary. For a node **c** operating at a given λ, the condition for terminating quad-tree decomposition at level $k_c = K$, in terms of maximum and minimum limits on Lagrangian cost, is given by

$$\begin{aligned}\sigma_{F_t}^2(T^2 2^{-2K}) &> D_{f,c}^{\delta(s)}(R_c) + \lambda R_c \\ &> \frac{1}{4}\sigma_{F_t}^2(T^2 2^{-2K}) + \lambda \left[\frac{(P+1)(P+2)}{2}\right].\end{aligned} \tag{19}$$

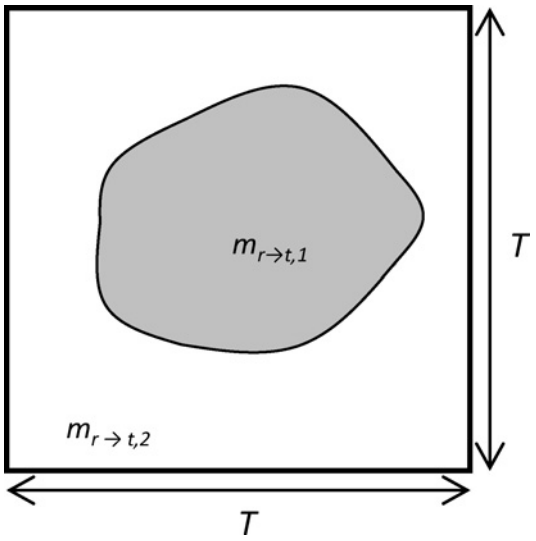

Fig. 10. Example of a motion field composed of two ploynomial motion flows $m_{r\to t,1}$ and $m_{r\to t,2}$, seperated by smooth boundary geometry.

The maximum limit in (19) is derived from the condition that must be satisfied for leaf nodes to be retained at level K; this was discussed above. Similarly, the minimum limit is derived from the inequality on Lagrangian cost that must be satisfied to ensure that tree growing does not proceed to level $K+1$. In the expression for the minimum limit in (19), the term $\frac{1}{4}\sigma_{F_t}^2(T^2 2^{-2K})$ represents the cost of operating with no motion compensation at level $K+1$ while the term $\lambda\left[\frac{(P+1)(P+2)}{2}\right]$ refers to the difference in Lagrangian costs between consecutive levels as expressed by (16). In the following discussion, we use R_K to refer to the rate allocated to leaf nodes without motion discontinuity at level K; that is $R_K = R_c$ when $k_c = K$. Also we use D_l to refer to the distortion of leaf nodes without motion discontinuity at all levels in the tree.

To enable us to derive R-D characteristics for a pruned quad-tree, we consider an example of a piecewise smooth motion field, such as that shown in Fig. 10, where a smooth boundary separates regions of smooth motion flow. We assume that the motion of each region can be precisely described by a 2-D polynomial model.

Let L represent the length of the motion discontinuity boundary. Given that the motion discontinuity boundary intersects a block at level k of the quad-tree, the average length of the chord intersecting the block can be expressed as $\ell T 2^{-k}$, where $T2^{-k}$ represents the dimensions of a block at level k with T referring to the dimensions of the square video frame and $\ell = 0.7351$, as determined in [25].[6] At level k, the average number of blocks M_k that intersect the motion discontinuity boundary is equal to

$$M_k = L/(\ell T 2^{-k}). \tag{20}$$

Note that for levels $k < K$, all M_k blocks will be partitioned to finer blocks at the next level, $k+1$. Therefore M_k represents the number of branch nodes for a level $k < K$. The number of

[6]Calculations for average chord length are correct only for sufficiently large values of k, depending on the curvature of the boundary.

leaf nodes that correspond to smooth motion flows can now be expressed as

$$N_k = 4M_{k-1} - M_k = L/(\ell T 2^{-k}). \tag{21}$$

This is because each of the M_{k-1} boundary nodes at level $k-1$ is partitioned into 4 children, M_k of which are partitioned further.

At level K, blocks containing motion discontinuity become leaf nodes as no further partitioning is possible. The total number of leaf nodes for the whole quad-tree, from level 1 to K, can now be found as

$$\begin{aligned} N_{1\to K} &= \sum_{k=1}^{K} N_k + M_K \\ &= \frac{L}{\ell T}(2^{K+1} - 2) + \frac{L}{\ell T}2^K \\ &= \frac{L}{\ell T}\left[3 \cdot 2^K - 2\right]. \end{aligned} \tag{22}$$

With this estimate of the total number of leaf nodes in the tree, we can formulate expressions for distortion and rate for the whole motion field. Total distortion is given by (23), where an expected distortion of $\sigma_{F_t}^2(T^2 2^{-2K})$ is assumed for each of the M_K leaf nodes with motion discontinuity, while all other leaf nodes have equal distortion D_l

$$\begin{aligned} D_f &= \left[\sum_{k=1}^{K} N_k \cdot D_l\right] + M_K \cdot \sigma_{F_t}^2(T^2 2^{-2K}) \\ &= \left[\frac{L}{\ell T}(2^{K+1} - 2) \cdot D_l\right] + \frac{L}{\ell T}2^K \cdot \sigma_{F_t}^2(T^2 2^{-2K}). \end{aligned} \tag{23}$$

We show in [21] that the tree pruning algorithm selects the finest level K and associated rate R_K such that $D_l \sim 2^{-2K}$; therefore we can conclude that $D_f \sim 2^{-K}$.

The total number of bits for the pruned quad-tree R_f is composed of two components, as shown in (24); firstly R_{Tree} the rate required to signal the pruned tree structure, and secondly R_{Leaf_Motion} the rate required for communicating the quantized motion models. To signal the tree structure, each node is assigned one bit to convey its identity as either a branch or leaf. To communicate motion parameters, we make use of the linear relationship in (15) between rates of leaf nodes at different levels

$$R_f = R_{Tree} + R_{Leaf\_Motion} \tag{24}$$

$$R_{Tree} = \sum_{k=0}^{K}(N_k + M_k) = 2\frac{L}{\ell T}(2^{K+1} - 1) \tag{25}$$

$$R_{Leaf\_Motion} = \sum_{k=1}^{K} N_k \cdot \left(R_K + \frac{(P+1)(P+2)}{2}(K-k)\right) \tag{26}$$

$$= \frac{L}{\ell T}\left[R_K(2^{K+1} - 2) + \frac{(P+1)(P+2)}{2}(2^{K+1} - 2K - 2)\right].$$

Since $D_f \sim 2^{-K}$ and $R_f \sim 2^K$, the asymptotic R-D behavior of the pruned quad-tree can be described by $D_f \sim \frac{1}{R_f}$.

D. Extension to Leaf Merging

Let G be the number of distinct motion regions in a given motion field. Consider a pruned quad-tree that models this motion field, successfully capturing all G motion regions in its representation. As discussed earlier, the number of leaf nodes that will need to be encoded will be far greater than G. Applying leaf merging to this quad-tree will ideally create G sets of joint nodes, making the number of leaf nodes that need to be encoded with motion parameters, equal to the number of distinct motion regions in the original motion field.

Consider two spatially neighboring leaf nodes **a** and **b**. Let A_a and A_b represent the area of the corresponding regions $\mathcal{B}_a$ and $\mathcal{B}_b$ respectively. Merging **a** and **b** would create a potentially new merged region $\mathcal{R}_{\mathbf{a+b}} = \mathcal{B}_a \cup \mathcal{B}_b$, with a total area $A_{a+b} = A_a + A_b$. R-D optimality implies allocating a rate R_{a+b} such that the distortion of the new merged region would equal D_l; that is $D_{a+b}(R_{a+b}) = D_l$. The expression for R_{a+b}, is shown below in (27), this is derived in the same way as (15) with the rate of a given region now written in terms of its area with respect to the area of a block at the lowest level K

$$R_{a+b} = R_K + \frac{(P+1)(P+2)}{4}\log_2\left(\frac{A_{a+b}}{T^2 2^{-2K}}\right). \tag{27}$$

In addition to the rate for describing motion, the merged region requires side information relating to the presence and direction of merge. We assume all leaf nodes require one bit to signal the merge decision and merged nodes require at most two bits to describe the direction of merge. For each leaf node, merging is performed only if it brings about a reduction in the overall Lagrangian cost. As we shall see, as long as $R_K \geq 3$, merging of nodes that belong to the same motion region always results in a reduction in Lagrangian cost.

E. R-D Performance of Quad-Tree with Leaf Merging

We now derive an approximation of the R-D performance of a quad-tree with leaf merging. We assume that all leaf nodes that belong to a particular smooth motion region, are merged together under R-D optimality conditions. This means that total distortion D_f^m is the summation of the distortions of G smooth motion regions and M_K leaf nodes that contain motion discontinuities. That is

$$\begin{aligned} D_f^m &= G D_l + M_K \cdot \sigma_{F_t}^2(T^2 2^{-2K}) \\ &= G D_l + \frac{L}{\ell T}2^K \cdot \sigma_{F_t}^2(T^2 2^{-2K}). \end{aligned} \tag{28}$$

Since $D_l \sim 2^{-2K}$ (as explained in [21]), we can conclude from (28) that $D_f^m \sim 2^{-K}$.

The total bit rate for the merged quad-tree, R_f^m, is given by

$$R_f^m = R_{Tree} + R_{Merge} + R_{\mathrm{Re}\,gion\_Motion}.$$

The rate required to signal the pruned tree structure, R_{Tree}, is unaltered by merging and is specified by the expression given earlier in (25). An upper bound on the rate needed to signal merge information R_{Merge}, is obtained by allocating 3 bits to each leaf node that models a smooth motion region; that is $R_{Merge} < 3\frac{L}{\ell T}(2^{K+1} - 2)$. From (27), we can see that the rate

required to code the motion of a single merged region can be upper bounded by equating the area of the merged region to T^2; this gives $R_{\mathrm{Re}\,gion\_Motion} \leq G\left[R_K + \frac{(P+1)(P+2)}{2}K\right]$. The total bit rate R_f^m can now be bounded as follows:

$$R_f^m < 2\frac{L}{\ell T}(2^{K+1}-1) + 3\frac{L}{\ell T}(2^{K+1}-2) + G\left[R_K + \frac{(P+1)(P+2)}{2}K\right]. \tag{29}$$

We note that in the presence of merging $D_f^m \sim 2^{-K}$ and $R_f^m \sim 2^K$, giving a R-D behavior that can be described by $D_f^m \sim \frac{1}{R_f^m}$. The asymptotic R-D behavior of the pruned quad-tree with merging is therefore of the same form as that observed earlier for the pruned quad-tree. However the introduction of merging leads to a more efficient representation of frame motion.

F. Comparative Advantage of Leaf Merging

We can see from (21) and the derivations for (22) that at least half of the leaf nodes containing motion are located at level K, the lowest level of the quad-tree; therefore the overall advantage of merging is very much determined by the gains attained at this level. If all leaf nodes modeling motion at level K merge, then for each node, the rate R_K is replaced with merge signalling bits; in our analysis we assume at most 3 bits to signal merge—1 bit to indicate the presence of merge and 2 bits to signal the direction of merge. From earlier discussions, we know that the remaining half of the leaf nodes, located at higher levels, have bit rates lower bounded by R_K; if we assume that all these leaf nodes also merge then we are able to make an initial conservative approximation of the advantage of leaf merging. Ignoring the fixed cost of signalling the tree structure, the ratio of bit rates before and after merging can be approximated by $\frac{R_K}{3}$. This simple analysis suggests that if, for example, $R_K = 4$, the gain through merging is limited to a reduction in bit rate by a factor of 1.3. However this first impression of the comparative advantage of merging does not take into account the reduction in distortion achieved by joint coding of neighboring nodes. This accompanying reduction in distortion is due to the fact that for exponentially decaying R-D functions, R-D optimality dictates that all leaf nodes or merged regions have the same distortion value regardless of their size [refer to (14) and related discussions]. In the following, we show with an aid of a general example, that even for low values of R_K, merging can achieve a reduction in total bit rate by a factor greater than 2.

At high bit rates (small λ) we can expect that $2^{K+1} \gg G$. Using the expressions for D_f and D_f^m given in (23) and (28) respectively, the ratio at high bit rates of total distortion of a pruned quad-tree, to that of a pruned quad-tree with merging, can be approximated by

$$\frac{D_f}{D_f^m} = \frac{\sum_{k=1}^{K} N_k \cdot D_l \;+\; M_K \cdot \sigma_{F_t}^2(T^2 2^{-2K})}{G D_l + M_K \cdot \sigma_{F_t}^2(T^2 2^{-2K})} \approx \frac{2D_l + \sigma_{F_t}^2(T^2 2^{-2K})}{\sigma_{F_t}^2(T^2 2^{-2K})}.$$

Similarly, by employing the relationships for R_f given in (24) to (26) and R_f^m in (29), the ratio of bit rates for the two tree representations can be approximated using the same assumption of $2^{K+1} \gg G$; this is shown below

$$\frac{R_f}{R_f^m} \approx \frac{2 + R_K + \frac{(P+1)(P+2)}{2}}{5}.$$

Clearly, the greater the value of R_K, the greater the bit rate saving achieved by merging. Even for the simplest case of translational motion, we note that a coarsely quantized motion model, should in most cases, spend at least four bits to communicate some useful motion; this corresponds to just two bits for each of the vertical and horizontal displacements . We know from experience, when using VLC tables as specified by H.264 [5] to code motion, on average a rate greater than four bits per vector is required. Therefore in general we can assume $R_K \geq 4$; this means for translational motion, merging reduces total bit rate by a factor of at least 1.4. As merging reduces both total distortion and rate, a lower Lagrangian cost is achieved.

To reveal the combined impact of merging on both D_f^m and R_f^m, we graph the R-D relationships derived earlier in Sections VI-C and VI-E for a range of λ values. We first establish certain relationships that allow us to determine K and R_K for a given λ. We use $R_{\min}$ to refer to the minimum rate required for describing the motion of a block, such that $D_{f,b}^{\delta(s)}(R_{\min}) = \sigma_{F_t}^2(T^2 2^{-2k})$; a general expression for $R_{\min}$is given in [21]. In video coding, $R_{\min}$ is akin to the minimum rate that must be exceeded to ensure that motion compensated inter coding produces a lower energy residue signal than the case of intra coding. We define k_d as the level at which the minimum rate $R_{\min}$ is sufficient to provide the desired distortion value when operating at R-D slope $-\lambda$; that is

$$D_{f,b}^{\delta(s)}(R_{\min})\,|_{k=k_d} = \sigma_{F_t}^2(T2^{-k_d})^2 = \frac{\lambda(P+1)(P+2)}{4\ln 2}.$$

We show in [21] that K, the finest level in the quad-tree, is located one or two levels above k_d, that is $k_d > K \geq k_d - 2$ for most reasonable values of $R_{\min}$. To maintain constant distortion across all levels, the rate allocated to a block must change linearly from a minimum value of $R_{\min}$ at level k_d such that at level K, the corresponding rate R_K is given by

$$R_K = R_{\min} + \left[\frac{(P+1)(P+2)}{2}\right](k_d - K).$$

Given a λ value, we are now able to determine D_l, K and R_K. These relationships are used to plot in Fig. 11 a set of R-D graphs for quad-tree models representing a motion field similar to that shown in Fig. 10 with $G = 2$ and affine models perfectly describing the motion of both regions; that is $P = 1$. In this example, we let $L = 2T$ and consider a range of values for $R_{\min}$; specifically $R_{\min}$ is equated to 2, 4 and 6 bits.

We graph the R-D relationship by varying λ for each value of $R_{\min}$ being considered; in our example λ is varied within the limits $\frac{\sigma_{F_t}^2 T^2}{2^{26}} < \lambda < \frac{\sigma_{F_t}^2 T^2}{2^5}$ enabling us to explore R-D behavior

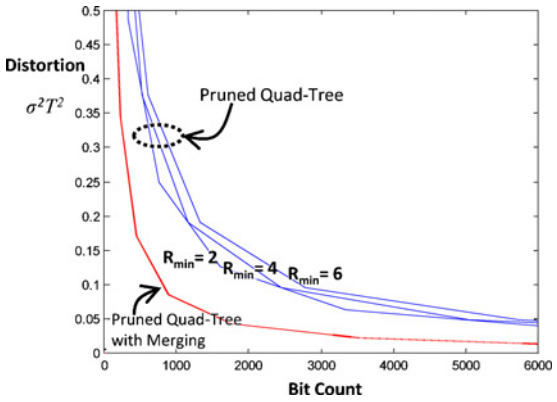

Fig. 11. R-D approximations, derived from theoretical analysis, illustrating the comparative advantage of leaf merging for bit rate values of $R_{\min}$ equal to 2, 4 and 6.

for values of K within the range $1 \leq K \leq 10$. R-D curves of pruned quad-trees, corresponding to (23), (25) and (26), are shown as dashed lines in Fig. 11. The vertical axis in this case refers to distortion of the motion compensated frame with the distortion being proportional to $\sigma_{F_t}^2$ and the area of the frame T^2. The horizontal axis of the graph refers to the total bit count required to communicate the quad-tree motion models. Total bit count depends on the quad-tree structure and the bit rate associated with each leaf node in the tree. Since we have assumed that the length of the motion boundary is proportional to the image dimension (i.e., $L = 2T$), the quad-tree structure and hence the average number of leaf nodes in the quad-tree, depend only on the level of the tree decomposition K

The R-D curves of the merged quad-tree, corresponding to (28) and (29), are shown as solid lines in Fig. 11. The performance of the merged quad-tree is not as sensitive to the value of $R_{\min}$, giving virtually overlapping curves for the three different values of $R_{\min}$. This lack of sensitivity to $R_{\min}$ is because for the merged quad-tree, the proportion of bits allocated to describing motion model parameters is greatly reduced when compared to a pruned quad-tree without merging. The graphs show considerable bit rate savings for leaf merging, especially at higher bit rates.

VII. Conclusion

R-D optimized leaf merging can be applied to quad-tree motion models improving the efficiency of representation of the motion field. Experimental results demonstrate that significant improvement in R-D performance can be gained by introducing merging for the two cases of hierarchical and spatially predictive motion coding. In the presence of merging, the R-D performance of hierarchical coding is similar to that of spatial prediction; hierarchical coding therefore becomes a viable option. Scalable decoding of the hierarchically coded motion tree can be accomplished by merely terminating decoding at an intermediate level. The Lagrangian cost objective, which the pruning and merging stages seek to minimize, can be appropriately modified to take into account scalability performance. Experimental investigations show that the modified cost objective can dramatically improve the performance of scalable decoding without significantly impacting full resolution coding efficiency. Theoretical analysis confirm the inherent advantages of leaf merging and support the experimental evidence showing improved R-D performance.

References

[1] R. Shukla, P. Dragotti, M. Do, and M. Vetterli, "Rate-distortion optimized tree-structure compression algorithms for piecewise polynomial images," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 343–359, Mar. 2005.

[2] R. D. Forni and D. Taubman, "On the benefits of leaf merging in quad-tree motion models," in *Proc. IEEE Int. Conf. Image Process.*, vol. 2. Sep. 2005, pp. 858–861.

[3] M. Tagliasacchi, M. Sarchi, and S. Tubaro, "Motion estimation by quadtree pruning and merging," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2006, pp. 1861–1864.

[4] R. Mathew and D. S. Taubman, "Hierarchical and polynomial motion modeling with quad-tree leaf merging," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2006, pp. 1881–1884.

[5] *Advanced Video Coding for Generic Audiovisual Services*, ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), ITU-T and ISO/IEC JTC 1, May 2003.

[6] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[7] S.-Y. Choi and S. lk Chae, "Hierarchical motion estimation in Hadamard transform domain," *IEEE Electron. Lett.*, vol. 35, no. 25, pp. 2187–2188, Dec. 1999.

[8] C.-M. Mak, C.-K. Fong, and W.-K. Cham, "Fast motion estimation for H.264/AVC in Walsh–Hadamard domain," *IEEE Trans. Circuits Systems Video Tech.*, vol. 18, no. 6, pp. 735–745, Jun. 2008.

[9] A. Secker and D. S. Taubman, "Lifting-based invertible motion adaptive transform framework for highly scalable video compression," *IEEE Trans. Image Process.*, vol. 12, no. 12, pp. 1530–1542, Dec. 2003.

[10] *SVC Core Experiment 1, Description of UNSW Contribution*, document m11441, ISO/IEC JTC1/SC29/WG11 (MPEG), Oct. 2004.

[11] D. S. Taubman and A. Secker, "Highly scalable video compression with scalable motion coding," in *Proc. IEEE Int. Conf. Image Process*, 2003, pp. 273–276.

[12] D. Taubamn, R. Mathew, and N. Mehrseresht, "Fully scalable video compression with sample-adaptive lifting and overlapped block motion," in *Proc. SPIE Visual Commun. Image Process.*, vol. 5685. 2005, pp. 366–377.

[13] M. Mrak, N. Sprljan, and E. Izquierdo, "Evaluation of techniques for modeling of layered motion structure," in *Proc. IEEE Int. Conf. Image Process.*, 2006, pp. 1905–1908.

[14] R. Xiong, J. Xu, S. Li, and Y.-Q. Zhang, "Layered motion estimation and coding for fully scalable 3D wavelet video coding," in *Proc. IEEE Int. Conf. Image Process.*, 2004, pp. 2271–2274.

[15] D. Maestroni, A. Sarti, M. Tagliasacchi, and S. Tubaro, "Scalable coding of variable size blocks motion vectors," in *Proc. IEEE Int. Conf. Image Process.*, 2004, pp. 1333–1336.

[16] S. S. Tsai and H.-M. Hang, "Motion information scalability for mc-ezbc," *Signal Process.: Image Commun.*, vol. 19, no. 7, pp. 675–684, Aug. 2004.

[17] K. Illgner and F. Muller, "Hierarchical coding of motion vector fields," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 1995, pp. 566–569.

[18] T. Wiegand, E. Steinbach, A. Stensrud, and B. Girod, "Multiple reference picture video coding using polynomial motion models," in *Proc. SPIE Visual Commun. Image Process.*, vol. 3309. 1998, pp. 134–145.

[19] D. Tzovaras, S. Vachtsevanos, and M. Strintzis, "Optimization of quadtree segmentation and hybrid two-dimensional and three-dimensional motion estimation in a rate-distortion framework," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 9, pp. 1726–1738, Dec. 1997.

[20] A. A. Muhit, M. R. Pickering, M. R. Frater, and J. F. Arnold, "Extended motion compensation using larger blocks and an elastic motion model," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, 2008, pp. 257–260.

[21] R. Mathew, "Quad-tree motion models for scalable video coding applications," Ph.D. dissertation, Faculty Eng., UNSW, Sydney, Australia, 2009 [Online]. Available: http://handle.unsw.edu.au/1959.4/44600

[22] P. Prandoni and M. Vetterli, "Approximation and compression of piecewise smooth functions," *Phil. Trans. Roy. Soc. London*, vol. 357, no. 1760, pp. 2573–2591, Sep. 1999.

[23] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.
[24] A. Secker and D. Taubman, "Highly scalable video compression with scalable motion coding," *IEEE Trans. Image Processing*, vol. 13, no. 8, pp. 1029–1041, Aug. 2004.
[25] P. W. Kuchel and R. J. Vaughan, "Average lengths of chords in a square," *Math. Mag. Math. Assoc. Am.*, vol. 54, no. 5, pp. 261–269, Nov. 1981.

Reji Mathew received the B.E. degree from the University of Western Australia, Perth, Australia, in 1990, and graduated with the M.E. and Ph.D. degrees from the University of New South Wales (UNSW), Sydney, NSW, Australia, in 1996 and 2010, respectively.

He is currently with UNSW, where he pursues his research interests in video coding, motion compensation, and scalable motion representations. His previous work experience includes employment with Motorola Labs, Motorola Australian Research Centre, Sydney, and National ICT Australia, Sydney.

David Taubman (M'95–SM'06) received the B.S. and B.E. (electrical) degrees from the University of Sydney, Sydney, Australia, in 1986 and 1988, respectively, and the M.S. and Ph.D. degrees from the University of California, Berkeley, in 1992 and 1994, respectively.

From 1994 to 1998, he was with Hewlett-Packard's Research Laboratories, Palo Alto, CA. He joined the University of New South Wales, Sydney, NSW, Australia, in 1998, where he is currently a Professor with the School of Electrical Engineering and Telecommunications. He is a co-author with M. Marcellin of the book *JPEG2000: Image Compression Fundamentals, Standards and Practice*. His current research interests include highly scalable image and video compression, inverse problems in imaging, perceptual modeling, joint source/channel coding, and multimedia distribution systems.

Dr. Taubman was awarded the University Medal from the University of Sydney, the Institute of Engineers Australia Prize, and the Texas Instruments Prize for Digital Signal Processing, all in 1998. He has received two Best Paper Awards from the IEEE Circuits and Systems Society for the 1996 paper "A common framework for rate and distortion based scaling of highly scalable compressed video," and from the IEEE Signal Processing Society for the 2000 paper "High performance scalable image compression with EBCOT."