MOTION ESTIMATION BY QUADTREE PRUNING AND MERGING

Marco Tagliasacchi, Mauro Sarchi, Stefano Tubaro

Dipartimento di Elettronica e Informazione
Politecnico di Milano, Italy

ABSTRACT

In this paper we propose a rate-distortion optimized motion estimation algorithm that is built upon a quadtree structure. Each node of the quadtree represents a block in the current frame together with its motion vector, and the block size decreases from the root to the leaves. In the first step, the quadtree is pruned according to a rate-distortion criterion in order to obtain blocks of variable sizes. A further rate rebate can be achieved by merging those leaf nodes of the quadtree that can be efficiently represented by the same motion vector. The proposed merging scheme provides a reduction of up to 50% of the rate spent for the motion model with respect to the case that performs pruning only.

1. INTRODUCTION

Accurate motion modeling plays an essential role in any video coding architecture, especially to efficiently represent video contents at low and very low bit-rates. In fact, most of the coding gain achieved by the latest state-of-the-art standard, H.264/AVC [1], comes from improved motion estimation and signalling tools, including blocks of variable sizes, quarter-pel motion accuracy, and a more efficient entropy coding of motion information.

Motion estimation algorithms used in video coding application cannot neglect the amount of information needed to represent the motion model. In fact, at low bit-rates, most of the bit budget is usually allocated to describe the motion, while little remains for encoding prediction residuals. Therefore, the accuracy of the motion representation needs to be tuned according to the target bit-rate: at high bit-rates an accurate (thus costly) motion representation is usually desirable. When the bit-rate decreases the amount of bits allocated to motion is usually reduces, therefore achieving a coarser motion representation. Motion estimation algorithms used in state-of-the-art video coding schemes always employ rate-distortion optimization criteria in order to obtain the best motion model representation, measured in terms of the energy of the prediction error, satisfying a given rate constraint [2][3].

In this paper we propose a motion estimation algorithm, specifically designed for video coding applications, that produces a region based motion representation. Here the goal is not to identify the motion models of independently moving regions but to provide a more compact representation of the motion model, without sacrificing its accuracy. For the sake of clarity, before illustrating the details of the proposed algorithm, Figure 1 shows the result of the motion estimation for a frame of the Table Tennis sequence. We notice that blocks characterized by the same motion vector are grouped together in order to reduce the amount of bits allocated to motion.

The work presented was developed within VISNET, a Network of Excellence (http://www.visnet-noe.org), funded by the European Commission.

Fig. 1. Table Tennis, SIF@30fps. Each region has a single motion vector assigned to it.

The work presented in this paper has been inspired from [4], where a quadtree-based coding scheme is used to efficiently encode images that can be modeled as piece-wise polynomials. Each node in the quadtree corresponds to an image block. A prune-merge scheme is used to segment the input image into regions. The novelty of the work in [4] lies in the fact that merging might involve leaf nodes in the quadtree that are not necessarily children of the same node. One or two polynomial models are used to approximate the image within each region. Both the pruning and the merging phase are rate-distortion optimized in order to achieve the best image approximation for a target rate. The idea of this work is to apply and adapt the prune-merging scheme to the problem of motion estimation.

A similar work has recently appeared in [5], where a merging scheme is applied to the variable block size representation provided by H.264/AVC. The goal of the present work is to study the benefits of merging blocks without reference to a specific coding architecture.

2. QUADTREE MOTION MODEL

The proposed motion estimation algorithm is based on a quadtree data structure, where each node represents a block of the current frame. At the top level of the quadtree the block size is $16 \times 16$. At the third (lowest) level, the block size is $4 \times 4$.

First, motion estimation is performed for blocks of size $4 \times 4$ using an exhaustive search approach with a search window of $\pm W$ pixels which is chosen based on the spatial resolution of the sequence (QCIF: $W = 16$ pixels, CIF/SIF $W = 32$ pixels, 4CIF $W = 48$ pixels). In each node we store the distortion (SAD - Sum of Absolute Differences) associated with each of the $(2W+1)^2$ candidate motion vectors.
vectors (MVs). Without the need of performing further block comparisons, we infer the distortion associated to each candidate motion vector for blocks of size 8 × 8. In fact, for a given candidate motion vector for a block of size 8 × 8, the SAD is simply obtained as the sum of the previously stored SADs of the same candidate motion vector applied to its child 4 × 4 nodes. The same procedure allows to compute the distortions for 16 blocks.

3. PRUNING ALGORITHM

We assume that motion vectors are encoded in a similar way as in H.264/AVC, by encoding the prediction error between the motion vector and its predictor using Exp-Golomb codes. The motion vector predictor is obtained as the median of the motion vectors of its causal neighbors. Therefore, the rate needed to encode a motion vector is:

\[ R_{MV} = 2|m_{x} - mvp_{x}| + 1 + 2|m_{y} - mvp_{y}| + 1 \]  

(1)

where \((m_{x}, m_{y})\) are the two components of the motion vector to be encoded and \((mvp_{x}, mvp_{y})\) is the motion vector predictor.

The goal of pruning is to find the optimal partitioning of each 16 × 16 block into sub-blocks. Using the quadtree representation, this is equivalent to pruning the quadtree in such a way that leaf nodes are not necessarily represented by 4 × 4 blocks. The pruning algorithm processes the quadtree according to a depth-first order. This means that each 16 × 16 block is considered one after the other and the pruning decisions for a 16 × 16 block are taken before considering the next 16 × 16 block.

The pruning algorithm can be summarized as follows. For each 16 × 16 block:

1. Compute the motion vector predictor
2. For each candidate MV \(k\), compute the Lagrangian cost as

\[ J_{16 \times 16}(k) = SAD_{16 \times 16}(k) + \lambda R_{MV}(k) + R_{nosplit}^{16 \times 16} \]  

(2)

3. Compute the lowest Lagrangian cost

\[ J_{16 \times 16}^* = \min_{k} J_{16 \times 16}(k) \]  

(3)

4. For each 8 × 8 child block \(i, i = 0, 1, 2, 3\)
   a) Compute the motion vector predictor
   b) For each candidate MV \(l\), compute the Lagrangian cost as

\[ J_{8 \times 8}(l) = SAD_{8 \times 8}(l) + \lambda R_{MV}(l) + \lambda R_{split/nosplit} \]  

(4)

(c) Compute the lowest Lagrangian cost

\[ J_{8 \times 8}^* = \min_{l} J_{8 \times 8}(l) \]  

(5)

5. Compute the total Lagrangian cost of the child blocks:

\[ J_{8 \times 8}^{tot} = \sum_{i=0}^{3} J_{8 \times 8}^* + \lambda R_{16 \times 16}^{split} \]  

(6)

6. If \(J_{16 \times 16}^* \leq J_{8 \times 8}^{tot}\) do not split the block. Assign to the 16 × 16 block the motion vector

\[ MV = \arg \min_{k} J_{16 \times 16}(k) \]  

(7)

Go to next 16 × 16 block

7. Otherwise, for each 8 × 8 child block \(i, i = 0, 1, 2, 3\)
   a) Compute the motion vector predictor
   b) For each candidate MV \(l\), compute the Lagrangian cost as

\[ J_{8 \times 8}(l) = SAD_{8 \times 8}(l) + \lambda R_{MV}(l) + R_{8 \times 8}^{split} \]  

(8)

(c) Compute the lowest Lagrangian cost

\[ J_{8 \times 8}^* = \min_{l} J_{8 \times 8}(l) \]  

(9)

(d) For each 4 × 4 child block \(j, j = 0, 1, 2, 3\)
   i. Compute the motion vector predictor
   ii. For each candidate MV \(m\), compute the Lagrangian cost as

\[ J_{4 \times 4}(m) = SAD_{4 \times 4}(m) + \lambda R_{MV}(m) \]  

(10)

iii. Compute the lowest Lagrangian cost

\[ J_{4 \times 4}^* = \min_{m} J_{4 \times 4}(m) \]  

(11)

Fig. 2. Table Tennis, SIF@30fps. a) Pruning with \(\lambda = 0.5\). b) Pruning with \(\lambda = 1.5\).
where \( \lambda \) is the Lagrangian multiplier that can be adjusted based on the target bit-rate. \( R_{\text{split}}^{(8 \times 8)} \) is the number of bits used to communicate to the decoder the binary decision of splitting the \( 16 \times 16 \) block. We use a context adaptive arithmetic coder to encode this information, in such a way that the average number of bits is less than \( \lambda \) is small, as the Lagrangian cost function depends more on the distortion than on the rate needed to encode the motion vectors.

\[
J_{\text{tot}}^{16 \times 16} = \sum_{j=0}^{3} J_{4 \times 4}^{j} + \lambda R_{8 \times 8}^{\text{split}}
\]

(f) If \( J_{8 \times 8}^{n} \leq J_{4 \times 4}^{n} \), do not split the block. Assign to the \( 8 \times 8 \) block \( i \) the motion vector

\[
MV_i = \arg \min_{l} J_{8 \times 8}^{l}(i)
\]

5. EXPERIMENTAL RESULTS

We carried out extensive experimental results on several test sequences. Figure 4 shows the rate-distortion curve for the Foreman (QCIF), Table Tennis (SIF), Coastguard and Bus sequences (CIF). We compare the performance of the merging algorithm with the case where only pruning is performed. The points of the curve are generated for different values of lambda, ranging from 0 to 2.5: at lower values of lambda, the cost of encoding the motion model increases but the distortion decreases. The rate refers to the total number of bits needed to encode the motion model (splitting decisions, merging decisions and motion vectors). As the merging algorithm is performed after pruning, the cost of encoding the splitting decisions is the same in both cases.
The proposed algorithm consistently outperforms the pruning algorithm at all bit-rates. As expected, the coding gain of the proposed algorithm is higher at low bit-rates (higher values of $\lambda$), as the Lagrangian cost function tends to weight more the cost associated with encoding the motion model. A gain of up to +0.6dB is reported for 

Bus. A smaller coding gain is observed for Foreman (up to +0.4dB), Table Tennis and Coastguard (up to +0.3dB). These figures should be interpreted carefully. In fact, the maximum coding gain is upper bounded by the lowest distortion that can be achieved when $\lambda = 0$, i.e when $4 \times 4$ blocks are used everywhere. In this case, no pruning nor merging is actually performed, therefore the two approaches converge to the same upper bound. Even with an arbitrary large bit-budget, a value of 34.4dB is obtained for Foreman, 31.6dB for Table Tennis, 29.9dB for Bus and 32.7dB for Coastguard. Therefore, it is more interesting to interpret these results from the rate perspective. Figure 4 shows that a 40%-50% rate rebate is achieved at low bit-rates, and that the margin remains significant in the whole range of bit-rates relevant for video coding applications.

6. CONCLUSIONS

This paper proposes a two-step pruning/merging motion estimation algorithm based on a quadtree structure. Our experimental results show that there are benefits coming from merging together leaf nodes of the quadtree representation produced in output of the pruning phase. Ongoing research activities are focused on integrating the proposed algorithm into a complete video coding architecture.

7. ACKNOWLEDGEMENTS

The authors would like to thank Prof. David Taubman for valuable discussions that inspired the work presented in this paper.

8. REFERENCES