MULTIGRID BLOCK MATCHING MOTION ESTIMATION FOR GENERIC VIDEO CODING

THÈSE N° 1221 (1994)

PRÉSENTÉE AU DÉPARTEMENT D'ÉLECTRICITÉ

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES TECHNIQUES

PAR

FRÉDÉRIC DUFAUX
Ingénieur physicien diplômé EPFL
originaire de Montreux (VD)

acceptée sur proposition du jury:

Prof. M. Kunt, rapporteur
Prof. H. Maître, corapporteur
Prof. F. Rocca, corapporteur
Prof. D. Thalmann, corapporteur

Lausanne, EPFL
1994

A mes parents

La sagesse n'est pas science,
La science n'est pas sagesse.

The wisdom is not science,
The science is not wisdom.

Lao-tzeu

Remerciements

Même si une thèse est premièrement le fruit d'un travail individuel, son bon déroulement implique l'engagement de plusieurs personnes. Je me dois donc dans les quelques lignes qui suivent d'exprimer mes remerciements à ces personnes sans lesquelles cette thèse n'aurait pas pu prendre la présente forme, et qui ont ainsi de près ou de loin contribué à ce travail.

En premier lieu, je tiens à exprimer ma gratitude au Professeur Murat Kunt pour m'avoir donner l'opportunité d'entreprendre ce travail au sein du Laboratoire de Traitement des Signaux (LTS). Ces années passées au LTS m'ont apporté une expérience extrêmement enrichissante, aussi bien sur le plan scientifique que culturel et humain. Je le remercie également pour le suivi et la direction de cette thèse, ainsi que pour son soutien continu dans le cadre, mais aussi au-delà, de ce travail.

J'aimerais également remercier Messieurs les Professeurs M. Hasler, H. Maitre, F. Rocca et D. Thalmann pour avoir accepté d'être les membres du jury de cette thèse et pour avoir évalué et commenté ce travail.

Je tiens ensuite à remercier tous les membres du LTS que j'ai eu l'occasion de cotoyer durant ces quelques années. J'aimerais tout particulièrement exprimer ma reconnaissance à Iole Moccagatta, Touradj Ebrahimi, Wei Li et Fabrice Moscheni. Par l'intermédiaire d'une étroite et fructueuse collaboration, ils ont contribué à cet ouvrage. J'aimerais également les remercier de l'amitié et de la gentillesse qu'ils m'ont témoignées durant ces années. Finalement, je souhaite remercier Fabrice Moscheni pour la lecture de la première version de ce manuscript et pour ses commentaires à la fois pertinents, critiques et constructifs. Mes remerciements vont également à Gilles Auric pour avoir toujours mis à ma disposition l'équipement informatique indispensable à ce travail, et à Fabienne Vionnet, Isabelle Bezzi et Corinne Degott pour leur disponibilité et leur gentillesse.

Je tiens tout particulièrement à exprimer ma gratitude à mes parents et à mes deux soeurs pour leur encouragement pendant toute la période de mes études.

Finalement, au cours d'un travail de recherche tel que celui-ci, les difficultés ne manquent pas de surgir. Il est alors important de pouvoir compter sur les encouragements et le soutien de ses amis. Je souhaite donc remercier ceux qui m'ont entouré durant cette période, ma gratitude allant tout particulièrement à Jeannie, Mathieu et Célia pour leur présence et leur soutien tout au long de ce travail.

Résumé

Les techniques d'estimation et de compensation de mouvement ont montré leur efficacité afin de réduire la redondance temporelle dans le cadre du codage de séquences d'images. L'estimation de mouvement consiste à analyser le déplacement des objets composants la scène. L'information de mouvement résultante permet d'améliorer le codage prédictif inter-image. Ce travail porte sur l'étude d'algorithmes d'estimation de mouvement pour les applications de codage de séquences d'images.

Les propriétés désirées d'un algorithme d'estimation de mouvement afin d'obtenir de bonnes performances se résument comme suit. Premièrement, l'algorithme produit une prédiction compensée en mouvement précise. De plus, il ne nécessite que peu d'information complémentaire pour représenter le mouvement. Finalement, il engendre un champs de mouvement lisse, représentatif du mouvement dans la scène. Cependant, il est clair que des vecteurs de mouvement plus précis demandent plus d'information complémentaire, et vice versa. Par conséquent, un compromis sur la complexité de l'estimation de mouvement doit être trouvé afin de balancer de manière optimale ces deux propriétés en contradiction. L'algorithme d'estimation de mouvement développé dans cette dissertation prend en compte ces remarques.

Les techniques d'appariement de blocs constituent une approche prometteuse pour l'estimation de mouvement dans le cadre du codage de séquences d'images. Dans ce contexte, l'algorithme classique de recherche exhaustive est largement utilisé, grâce à sa simplicité et sa facilité d'implantation en hardware. Néanmoins, il est caractérisé par certaines limitations. En particulier, son efficacité est faible sur le contour des objets en mouvement. D'autre part, il introduit des artefacts de blocs dans l'image compensée en mouvement. De plus, il tend à produire des champs de mouvement bruités. Finalement, il nécessite une grande complexité de calculs. Cette étude a pour but de résoudre ces limitations afin de satisfaire les propriétés désirées d'un algorithme d'estimation de mouvement telles que décrites ci-dessus.

Dans cette dissertation, une technique d'estimation de mouvement multi-grille et localement adaptative par appariement de blocs est proposée. Les vecteurs de mouvement sont itérativement affinés sur une structure de grilles de résolutions différentes. Grâce à cette structure multi-grille, l'algorithme engendre simultanément une faible énergie de l'erreur de prédiction et un champs de mouvement robuste et proche du mouvement effectif dans la scène. De plus, la complexité de calculs est grandement réduite. En introduisant une taille de grille variant localement, d'une part une précision accrue des vecteurs de mouvement est obtenue sur le contour des objets en déplacement, et d'autre part la quantité d'information complémentaire est réduite dans les régions uniformes. Pour ces raisons, l'estimation de mouvement multi-grille et localement adaptative par appariement de blocs

mène à des performances plus élevées lorsqu'elle est comparée à la technique de recherche exhaustive. Cette amélioration apparaît en termes de précision et robustesse des vecteurs de mouvement, quantité d'information complémentaire nécessaire, performances de codage, qualité visuelle de la séquence reconstruite, et complexité de calculs.

Dans le but d'éviter les artefacts de blocs liés aux techniques d'appariement de blocs, une segmentation du champs de mouvement basée sur la quantification vectorielle est proposée. Les blocs contenant plusieurs objets se déplaçant dans des directions différentes sont segmentés par quantification vectorielle. A chacune des régions résultantes est assignée un vecteur de mouvement différent. La méthode améliore la compensation de mouvement le long des contours des objets en mouvement. Il en résulte des performances de codage plus élevées ainsi qu'une qualité visuelle augmentée.

Dans une étape suivante, un critère entropique est introduit afin de contrôler la procédure d'estimation de mouvement. En évaluant les coûts de transmission relatifs à l'erreur de prédiction et à l'information complémentaire de mouvement, ce critère optimise la procédure d'estimation et de compensation de mouvement. Plus précisement, il permet d'obtenir le compromis optimal sur la complexité de l'estimation de mouvement, pour une largeur de bande donnée. Cette méthode est appliquée avec succès à la technique multigrille localement adaptative par appariement de blocs et à la segmentation du champs de mouvement basée sur la quantification vectorielle.

Finalement, un système de codage générique des signaux vidéo est présenté. Ce système supporte une large variété d'applications et de ce fait est approprié pour les services multimédia. Les résultats de simulations montrent de bonnes performances en termes de codage ainsi qu'une haute qualité visuelle.

Abstract

Motion estimation and compensation techniques have shown their efficiency to reduce temporal redundancies in video coding applications. Motion estimation analyzes the movement of objects in the scene. The resulting motion information allows to improve interframe predictive coding. This work deals with the study of motion estimation algorithms in the framework of image sequence coding.

The desired features of a motion estimation algorithm in order to achieve high performances are the following. First, the algorithm provides an accurate motion compensated prediction. Second, it requires a low overhead information. Third, it leads to a smooth motion field close to the true motion in the scene. However, it is straightforward that more precise motion vectors need a higher overhead information, and vice versa. Consequently, a trade-off on the motion estimation complexity has to be found in order to optimally balance these two conflicting features. The motion estimation algorithm developed in this dissertation takes into account the above remarks.

Block matching techniques are a promising approach for motion estimation in image sequence coding. In this framework, the classical full-search algorithm is widely used due to its simplicity and ease of hardware implementation. Nevertheless, it suffers serious drawbacks. In particular, it performs poorly on moving edges and introduces block artifacts in the motion compensated frame. Furthermore, it tends to produce noisy motion fields. Finally, it requires a high computational complexity. In this study, we aim at overcoming the above drawbacks in order to fulfill all the above desired features of a motion estimation algorithm.

In this dissertation, we propose a locally adaptive multigrid block matching motion estimation technique. The motion vectors are iteratively refined on a set of grids with different resolution. Due to this multigrid structure, the algorithm produces a low energy prediction error and a robust motion field close to the true motion in the scene. Furthermore, the computation complexity is greatly reduced. Introducing a locally varying grid size allows to improve the motion vectors accuracy on moving edges and to reduce the overhead information in uniform areas. Therefore, the locally adaptive multigrid block matching motion estimation outperforms the full-search technique in terms of motion vectors accuracy and smoothness, amount of overhead information, coding performances, visual quality of the reconstructed sequences, and computational complexity.

In order to avoid the block artifacts related to the block matching techniques, a VQ-based segmentation of the motion field is proposed. Blocks which contain several objects moving in different directions are segmented by means of VQ and a different motion vector is assigned to each of the resulting regions. The method improves motion compensated

prediction along moving edges, resulting in higher coding performances and enhanced visual quality.

In a further stage, an entropy criterion is introduced to control the motion estimation procedure. By evaluating the transmission cost of both the prediction error and the overhead motion information, it achieves an optimization of the motion estimation and compensation. More precisely, it leads to the optimal trade-off on the motion estimation complexity given an allotted bandwidth. This method is applied in both the locally adaptive multigrid block matching technique and the VQ-based segmentation of the motion field.

Finally, a generic video coding system is presented. It supports a wide range of applications and is suitable for multimedia services. Simulation results show good coding performances and high visual quality.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Statement of the problem

The recent advances in technology have led to new communication media in which visual information plays a key role. Digital TV, high definition TV (HDTV), video-conference, video-phone, medical imaging, archiving, virtual reality and multimedia are some examples of emerging applications. These applications involve not only technical considerations, but also economical and political interests. Furthermore, they are tightly linked to an evolution of the society. The current effort for the development of HDTV systems in Europe, USA and Japan is a representative example.

When compared to audio or texts, video signals demand an huge amount of information. Despite the increasing disks storage capacity and the development of broadband networks, their storage and transmission requires the use of compression techniques. Image compression techniques are based on two principles: the reduction of the statistical redundancies in the data and the consideration of the human visual system imperfections. Furthermore, some applications allow for visible distortions in the reconstructed images. Image compression is a domain of intensive research and review articles on the subject can be found in the literature [1, 2, 3]. The need for image compression techniques motivates also recent standardization efforts: JPEG [4] (Joint Picture Expert Group) from the International Standardization Organization (ISO) for still-images, MPEG-I [5] and -II [6] (Motion Picture Expert Group) from ISO and the recommendation H.261 [7] from the International Consultative Committee for Telephone and Telegraph (CCITT) for image sequences.

The key for high performances in image sequence coding compared to still-image coding lies in an efficient representation of the motion information. For this purpose, motion estimation and compensation techniques have been proposed and have shown their efficiency [8, 9, 3]. Their principle is the following. The displacement of objects between successive frames is estimated (motion estimation). Hence, the resulting motion information is exploited for an efficient interframe predictive coding (motion compensation). Consequently, the prediction error as well as the motion representation are transmitted instead of the frame itself.

This dissertation deals with the development of an algorithm to estimate the motion field between successive frames for video coding applications. First, the definition of the term motion field should be clarified. In image sequence processing, a distinction is made between the projection of the $3D$ motion on the $2D$ image plan and the field associated to the spatio-temporal variations of intensity. The former is referred to as the $2D$ motion field and the latter as the optical flow. In coding applications as well as in this work, the term motion field should be understood as the optical flow. In that respect, the purpose of a motion estimation algorithm can be defined as follows. It is to generate *a motion field*

close to the motion in the scene, to provide *an accurate motion compensated prediction* and to require *a low amount of overhead information.* These desired features constitute the guideline for the developments studied in this dissertation.

1.2 Investigated approach

Numerous techniques have been proposed in the literature to estimate the motion in an image sequence [10, 11, 12, 8, 9, 3, 13]. Among those techniques, block matching seems a promising approach in the framework of video coding applications [9, 3]. For this purpose, the full-search block matching algorithm is widely used. This method has several advantages: simplicity, ease of hardware implementation and low overhead information. However, it has also the following disadvantages: poor performances on moving edges, block artifacts, noisy motion fields and high computational complexity.

In this work, we investigate a block matching motion estimation technique which aims at overcoming the above drawbacks while keeping the positive points. The three main achievements of this study are introduced in the remaining of this section.

The observation that natural scenes frequently contain motion at different scales motivates the introduction of multi-level algorithms. Large range displacements are robustly estimated on large-scale structures and short range displacements are accurately estimated on small-scale structures. Besides, finer structures are important in detailed areas, whereas coarser structures are sufficient in uniform regions. Taking into account these remarks, a locally adaptive multigrid block matching motion estimation technique is introduced. It is based on the multigrid theory developed in the field of mathematics. It shares some similarities with the hierarchical block matching algorithms proposed in [14, 15] and the variable block size block matching technique proposed in [16]. Due to the multigrid structure, smooth, robust and accurate motion fields are obtained. Furthermore, the computational complexity is greatly reduced. Due to the adaptive structure more precise motion vectors are estimated on moving edges and the side information is decreased in uniform areas.

One of the main limitation of the block matching technique is its block-based nature. A promising approach in order to overcome this limitation is to relax the block-based constraint in regions where this motion model fails. In other words, by segmenting blocks which contain several objects moving in different directions, the motion compensated prediction can be greatly improved. However, the amount of side information to represent the segmentation should remain low. For this purpose, a motion field segmentation algorithm based on vector quantization (VQ) is proposed. Blocks corresponding to moving edges are segmented and different motion vectors are assigned to each of the resulting

regions. The segmentation is approximated by a finite set of different patterns. It results in an improved motion compensated prediction in visually important areas, whereas the side information is kept low.

Finally, a trade-off has to be found on the motion estimation complexity. Clearly, more precise motion vectors lead to an improved prediction but require a higher coding cost, and conversely a simpler motion estimation needs a lower amount of side information but provides a poorer prediction. By evaluating the transmission cost relative to both the prediction error and the motion information, the optimal trade-off can be reached. This procedure defines the so-called entropy criterion. The criterion allows to find the adequate motion complexity in order to optimally balance the amount of information corresponding to the prediction error and the representation of motion.

1.3 Organization of the dissertation

The dissertation is organized as follows.

Chapter 2 begins with a review of the most well-known temporal redundancies reduction techniques for image sequence coding applications. It includes motion compensated hybrid coding, $3D$ transform techniques and camera motion estimation. A motion compensated hybrid transform scheme is described in more details. In a next step, motion estimation algorithms developed in various fields as computer vision, image sequence analysis, image sequence restoration and image sequence coding are discussed. These algorithms can be divided into four groups: transform-based, gradient-based, pel-recursive and block matching techniques. After an introduction on the notion of motion, including the perception of motion and the distinction between $2D$ motion field and optical flow, basic assumptions common to most motion estimation algorithms are discussed as well as possible methods to relax them. It is followed by the review of the four above groups of motion estimation algorithms, the emphasize being on block matching techniques. Methods belonging to this last group seem indeed the most suitable for coding purposes. Nevertheless, classical algorithms, e.g. full-search, suffer serious drawbacks which are analyzed. Finally, numerous ways to solve these drawbacks (including the algorithms developed in this dissertation) are presented.

In order to evaluate the performances of motion estimation techniques, criteria have to be defined. This is the purpose of Chap. 3. Two key features of a motion estimation algorithm are to provide an accurate motion compensated prediction and to require a low cost overhead information. Therefore, the displaced frame difference (DFD) energy and the motion vectors entropy constitute two pertinent insights on the performances. In a further stage, simulations within a coding scheme have to be carried out in order to com-

pare the bit rate versus the reconstructed quality when using different motion estimation. For this reason, three motion compensated hybrid coding schemes are defined which differ only in the DFD coding. The first one is based on a wavelet transform, the second one on an interframe Differential Pulse Code Modulation (DPCM) and the third one on a segmentation of the DFD. Three test sequences difficult in terms of motion are selected for simulations. A procedure to convert these sequences from interlaced to progressive scan is also described.

In Chap. 4, a multigrid block matching motion estimation technique is introduced and analyzed. In a first stage the mathematical theory of multigrid techniques is briefly summarized. Next, the multigrid block matching algorithm is described in details. In particular, the multigrid structure, the strategy to control the data flow within the structure, the up-/down-conversion operators to map the motion field from one grid to the consecutive one and the computational complexity are studied. Simulations are carried out in order to compare: the Mean Absolute Error (MAE) and Mean Square Error (MSE) matching criteria, the coarse-to-fine and fine-to-coarse-to-fine control strategies, different up-/down-conversion operators, motion vectors sub-pixel accuracies, and finally the performances of the proposed multigrid algorithm with the full-search and classical fast search block matching techniques. These results are commented and conclusions are drawn.

In Chap. 5, the above multigrid algorithm is improved by introducing a local adaptation, leading to the so-called locally adaptive multigrid block matching technique. The split procedure to generate the adaptive multigrid structure is discussed. Two configurations are proposed: the first one aims at decreasing the side information, whereas the second one can improve both the DFD coding and the amount of side information. The problem of the criterion to decide whether to split a block is also addressed. Simulations compare the performances of the algorithm with the full-search technique. The results are analyzed and conclusions are drawn.

The VQ-based segmentation of the motion field is introduced in Chap. 6. After motivating the utility of the motion field segmentation, the general VQ-based segmentation algorithm is described. The amount of side information resulting from this operation is also evaluated. Next, realistic hypotheses are introduced in order to simplify the general algorithm and to obtain an efficient implementation. It includes a discussion on the adequate number of segmented regions in each block, the connectivity and the codebook generation. Afterwards, the problem of the segmentation decision rule is addressed. Finally, simulations are carried out in order to show the improvement of the VQ-based segmentation when compared to full-search block matching. The results are discussed and conclusions are deduced.

Chapter 7 deals with the optimization of motion compensation by means of the entropy

criterion. First, the need to control the motion estimation procedure in order to achieve the optimal allocation between motion fields accuracy and amount of overhead information is discussed. The entropy criterion is defined for this purpose. Some examples of applications are given to motivate this study. The entropy criterion demands a statistical model of the DFD. For this purpose, the correlation in the DFD is studied. Next, a memoryless Laplacian model is proposed and experimentally validated. It is followed by the analytical computation of the energy and the entropy of a Laplacian probability density function (PDF) in both the continuous and uniform quantization cases. Afterwards, the entropy criterion is applied to the locally adaptive multigrid block matching motion estimation and to the VQ-based motion field segmentation. Finally, simulations compare the efficiency of the entropy criterion with a criterion based on a threshold. Conclusions are drawn from these results.

A new generic coding scheme using the locally adaptive multigrid block matching motion estimation and the entropy criterion is presented in Chap. 8. The importance of features such as generic coding and scalability, as well as high visual quality on a wide range of bit rates are pointed out. The general description of the codec is followed by the discussion of each of its main components: Gabor-like wavelet transform, motion estimation, motion compensation, quantization and entropy coding. In order to show the generic coding capability of the system, experiments are carried out on HDTV, TV and video-phone sequences and conclusions are drawn.

Finally, a summary of the main developments, results and conclusions of the dissertation is given in Chap. 9. Possible extensions and new directions of research are also indicated.

1.4 Main contributions

To the best of our knowledge, the main contributions of this work are the following:

- Definition of a multigrid block matching algorithm which provides smooth and accurate motion fields with a greatly reduced computational complexity when compared to the full-search method.
- Proposition of a fine-to-coarse-to-fine control strategy and comparison with the classical coarse-to-fine approach.
- Presentation and comparison of different up- and down-conversion operators to map the motion vectors from one grid to the consecutive one.
- Comparison between MAE and MSE for the matching criterion.
- Comparison on the motion vectors sub-pixel accuracies.

- Definition of a locally adaptive multigrid block matching algorithm to generate more precise motion vectors in detailed areas and a decreased amount of side information in uniform ones.
- Introduction of a VQ-based motion field segmentation algorithm in order to avoid the block artifacts characteristic of block matching motion estimation techniques.
- Definition of an entropy criterion which efficiently controls the motion estimation procedure and leads to the optimal trade-off between motion fields accuracy and amount of overhead information. Application to the locally adaptive multigrid block matching technique and the VQ-based motion field segmentation algorithm.
- Study of the DFD statistic, validation of a memoryless Laplacian model and derivation of the energy and entropy of a Laplacian PDF in the uniform quantization case.
- Presentation of a generic video coding system suitable for visual communication applications.

Chapter 2

Motion estimation and compensation techniques - State of the art

2.1 Introduction

Video coding techniques exploit the redundancies of the data in order to reduce the bandwidth to represent the visual information. In natural image sequences, redundancies arise from both spatial correlation within an image and temporal correlation between successive images. Due to the different nature of the video signal in the spatial and temporal dimensions, the latter are usually processed separately. Coding techniques which aim at reducing the spatial correlation are named *intraframe coding*, whereas those which reduce the temporal correlation are called *interframe coding*. A review of intraframe and interframe coding is given by Netravali and Limb in [1], Jain in [2] and Netravali and Haskell in [17].

Among the major intraframe techniques, we can mention *predictive coding*, *transform coding*, *subband/wavelet coding* and *second generation coding*. The three first ones rely on the concepts of information theory and reduce the statistical redundancies in the data. In contrast, the forth one relies on the human visual system and describes images with a symbolic representation which is more compact. More precisely, in predictive coding pixels are predicted (with a linear or nonlinear function) from the previously transmitted pixels of the frame, and only the prediction error is transmitted. The Differential Pulse Code Modulation (DPCM) [18] technique belongs to this category. In transform coding, pixels are transformed from the space domain into another domain (the transform domain) where they have a more efficient representation. In particular, the transform concentrates the energy of the signal in few coefficients. Furthermore the coefficients resulting from the transform are decorrelated. Several transforms have been extensively studied, for instance the Discrete Cosine Transform (DCT) [19] and the Karhunen-Loeve Transform (KLT) [20]. In subband/wavelet coding, either a subband decomposition [21] or a wavelet transform [22, 23] is applied on the image. The latter operation creates a set of subbands, each of them containing a limited range of spatial frequencies. The motivation of these approaches is that the resulting subbands can be encoded more efficiently than the original image. It should be pointed out that a correspondence exists between subband, wavelet and transform representations of a signal [24]. Finally, second generation techniques, in an attempt to imitate the functions of the human visual system, describe images in terms of physical entities such as contours or regions [25, 26]. As the resulting contours and textured areas can be efficiently coded, these techniques lead to a more compact representation.

Interframe coding can be considered as a particular case of predictive coding where the prediction is based on pixel values from the previous frame. For instance, in the portion of a scene with small motion, pixels are precisely predicted from the pixels at the same location in the previous frame. The last observation is not true any longer in scenes with large motion. In this case, pixels in the previous frame spatially displaced by the appropriate displacement vector are a much more efficient prediction. This prediction is

named *motion compensated* prediction. The difficulty of this approach lies undoubtedly in estimating accurately the motion between two frames. This is the goal of *motion estimation.* In coding scheme based on the above principle, the motion compensated prediction error, more commonly called *displaced frame difference* (DFD), is transmitted instead of the frame itself. It results in a more efficient representation of the visual data. The motion information requires usually to be send along as overhead.

Motion estimation algorithms have been studied and developed for very different applications in image processing. Among the most important ones, we can mention image sequence analysis [10, 11, 12], image sequence interpolation and restoration [27, 28, 29], and image sequence coding [8, 9, 3]. The purpose of motion estimation is very different depending on the application. In image sequence analysis, the motion information is used to extract useful features of the image sequence. In image sequence interpolation and restoration, adaptive filtering exploits the motion information in order to avoid blurring of moving objects. Finally, in image sequence coding motion information allows to reduce the temporal redundancies in the image sequence.

As the above applications are very different in nature, they have led to very different motion estimation algorithms. For example, in image sequence coding the motion estimation is used for predictive coding. Thus, the fact that it represents the motion in the scene is not an intrinsic goal. Therefore, the classical denomination of motion estimation is maybe inappropriate and misleading. Furthermore, the motion information should usually be transmitted along with the image sequence as overhead information, unless the decoder is able to reconstruct the motion field on-line. This may be the case for pel-recursive motion estimation algorithms [8]). However, as this constraint restricts severely the motion estimation technique (the prediction should be causal), in the coming discussion the assumption is made that motion vectors have to be transmitted to the decoder. Consequently, the motion estimation algorithms should provide a good prediction as well as a low coding cost of side information. The fact to obtain a very precise motion field, in the sense of the motion present in the sequence, is secondary.

In the remaining of this chapter, first temporal redundancies reduction techniques are discussed in Sec. 2.2. In particular, motion compensation coding is described. Next, motion estimation algorithms are reviewed in Sec. 2.3. Finally the main issues are summarized in Sec. 2.4.

2.2 Temporal redundancies reduction - motion compensated coding

The challenge of image sequence coding compared to still image coding lies in an appropriate processing of motion information. High performances can be reached by combining intraframe and interframe coding techniques together. This is called *hybrid coding.* An efficient combination consists in coding the DFD resulting from motion compensation by an intraframe technique. One important example of such a coding scheme is the case where a $2D$-transform is applied on the temporal prediction error. It is referred to as *motion compensated hybrid transform coding.* Recent standards are based on this idea, MPEG-I [5, 30] and MPEG-II [6, 31] from ISO and H.261 [7] from CCITT, which are performing a $2D$ DCT of the DFD.

A good motion estimation technique and an efficient coding of the DFDs are imperative for high quality video coding. However, in coding the final objective of a motion estimation algorithm is unclear. For instance, a very precise motion estimation leads on the one hand to a very low DFD energy but on the other hand to a high overhead motion information. Conversely, a coarse motion estimation produces a low overhead motion information but a high DFD energy. Consequently, the optimal motion estimation algorithm should simultaneously provide an accurate motion field, i.e. a low energy DFD, while keeping the side motion information low. It should therefore aim at jointly minimizing the amount of DFD and motion information. This problem is addressed in Chap. 7. It should be noted also that the optimal motion estimation algorithm depends on the subsequent intraframe coding technique, as well as the type of applications (e.g. HDTV, TV, video-phone, ...) and the target bit rate.

Finally, the approach applying a transform coding technique to the DFD resulting from motion compensation presents two drawbacks. Natural images exhibit a high correlation. The energy of such a signal is optimally compacted by the KLT. In practice the DCT is preferred, as in this framework it approximates closely the KLT and has a faster implementation. However, observations show that DFDs have very low correlations [32, 33]. Therefore, the use of the DCT as an optimal transform is not valid anymore in this context. More generally, a transform coding technique which aims at decorrelating the data performs poorly on the DFD. Furthermore, in case of a block-based motion estimation technique (e.g. block matching algorithms [9, 3] which are widely used in video coding, see Sec. 2.3.6 for more details), block artifacts can be introduced in the DFD, reducing the efficiency of the intraframe coding technique (e.g. transform coding) if the latter is applied on the DFD. In spite of the last remarks, this type of coding schemes is the most efficient up to now, and therefore the most widely used in the field of video coding.

An alternative to the classical hybrid transform coding is to invert the order of interframe

and transform techniques. Namely, input images are first $2D$ transformed. Then, motion estimation and compensation is performed directly on the coefficients of the transform. This way, the transform is applied on the images, with high efficiency as natural images have high correlations, rather than on the DFDs which have low correlations. Therefore, these schemes avoid the above mentioned problems relative to hybrid coding. Systems based on this idea and using a wavelet transform have been proposed in [34, 35]. However, due to the subsampled nature of the subbands, accurate motion fields, both in the sense of density (e.g. small block size for a block-based motion estimation technique) and accuracy (i.e. 1/2 pixel or higher accuracy) are required in order to achieve good performances. Therefore, the overhead information is significantly increased. Furthermore, subband signals are very different from image signals. Except for the DC component, they can be modeled by a Laplacian probability distribution function with zero mean [36]. Consequently, classical motion estimation techniques developed to evaluate the displacement between two images perform less efficiently on the subbands. Despite this drawback, systems in [34, 35] are using a block matching technique. It is clear that these methods would benefit from new dedicated motion estimation algorithms.

Other approaches to reduce temporal redundancies for video coding have been investigated. We can mention $3D$-transform techniques and global camera motion estimation. The $2D$-transform techniques can be straightforwardly extended to $3D$. Systems based on a $3D$ subband decomposition [37, 38, 39] and on a $3D$ wavelet transform [40, 41] have been proposed, leading to good visual quality. Furthermore, these systems do not rely on motion estimation, which is from an hardware point of view the most complex part of an hybrid transform encoder, resulting in a decreased encoder complexity. However, these systems suffer a very serious drawback: they require several frame buffers (depending on the number of frames simultaneously transformed), increasing both encoder and decoder cost, as well as introducing a coding delay unsuitable for video-conference or video-phone applications. Motion compensation techniques use local estimates of the motion for prediction. Camera motion, such as pan and zoom, can be more efficiently handled if it is globally estimated. This is the goal of global motion estimation. In [42, 43, 44], the camera motion is modeled by two parameters: a pan vector and a zoom factor, which are evaluated by either a pel-recursive algorithm in [42] or a block matching technique in [43, 44]. In [45], a global motion compensated technique which considers zoom and pan as well as rotation motion parameters is proposed, these latter being estimated by a pel-recursive technique. Finally, an edge based camera motion estimation is proposed in [46], in which camera motion is modeled by seven parameters including pan, zoom, rotation and translation. The global motion estimation technique can be combined with classical motion compensated coding schemes. Camera motion is estimated in a first stage, then local motion is estimated on the globally compensated frames as previously. Improvements due to the two stage global/local motion compensation are reported in [43]. Research effort devoted to these two topics is regretfully too little.

2.2.1 Motion compensated hybrid intraframe/interframe coding

Motion compensated hybrid intraframe/interframe coding schemes have achieved the highest performances up to now, and therefore are the most widely used in video coding. This type of schemes is illustrated in Fig. 2.1. The way a motion compensation technique is applied in such a scheme is discussed in more details now. One recent and important example of an hybrid scheme is the test model of MPEG-II [6, 31]. As most of the principles we discuss are used in this test model, the same nomenclature as MPEG-II is adopted.

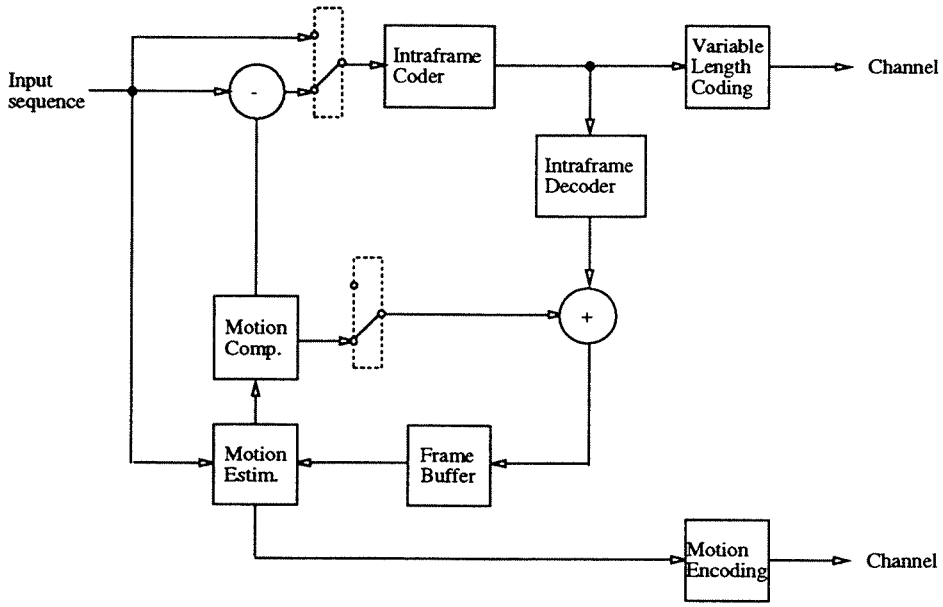

Figure 2.1: Motion compensated hybrid intraframe/interframe encoder block diagram of a coding scheme where the DFD resulting from motion compensation is coded by an intraframe technique.

In Fig. 2.1, the encoder includes not only the intraframe coder but also the decoder. Consequently, decoded frames can be used in the encoder for motion estimation and compensation. This enables the decoder to reconstruct exactly the same sequence of predicted frames. It should be noted also that this technique requires a frame buffer.

Motion compensated coding requires a temporal reference. In order to avoid error propagation and to allow random access in the sequence, this temporal reference should be updated regularly. MPEG-II defines a group of pictures (GOP) layer. The first frame of a GOP is intraframe coded (I-picture). Following frames are alternatively motion compen-

sated predicted (P-pictures) or motion compensated interpolated (B-pictures, B standing for bi-directional). The three coding modes in a GOP are illustrated in Fig. 2.2. P-pictures are predicted frames coded relative to the previous P- or I-picture (forward prediction). B-pictures are coded relative to the previous and next P- or I-pictures. They can be either interpolated from these two frames, or forward predicted from the previous one, or finally backward predicted from the next one. The choice of the number of frames in a GOP, N_{GOP}, and the number of B-pictures, N_{B-pict}, between two P-pictures (or between one I-picture and one P-picture) is left open. Typically, in the MPEG-II test model [6], $N_{GOP} = 12$ and $N_{B-pict} = 2$. It should be pointed out that the introduction of B-pictures increases significantly the hardware complexity of both the encoder and decoder. First it requires to perform twice the motion estimation in the encoder, once forward and once backward. Second, the number of frame buffers in the encoder and decoder, as well as, the coding delay, are increased. However, B-pictures solve the problem of uncovered background by allowing a temporal reference in the future, and lead therefore to higher coding performances.

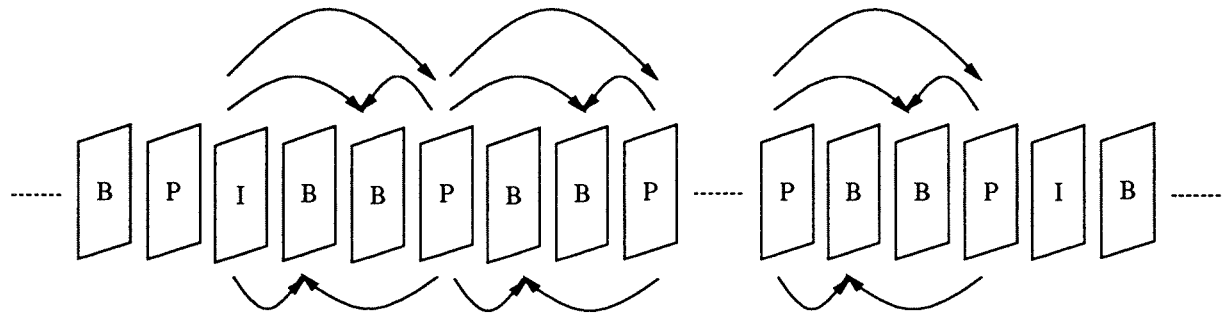

Figure 2.2: The three coding modes, I-, P- and B-pictures, in a GOP ($N_{B-pict} = 2$).

If $\hat{I}(\vec{r}, t)$ is the decoded image intensity at location $\vec{r}$ and time t, and $\vec{d}_f$ is the forward displacement during the time interval Δt_f (respectively $\vec{d}_b$ the backward displacement during Δt_b), the predicted image $I_p(\vec{r}, t)$ is given by the following formula:

Motion compensated forward prediction:

$$I_p(\vec{r}, t) = \hat{I}(\vec{r} - \vec{d}_f, t - \Delta t_f) \ , \tag{2.1}$$

Motion compensated backward prediction:

$$I_p(\vec{r}, t) = \hat{I}(\vec{r} + \vec{d}_b, t + \Delta t_b) \ , \tag{2.2}$$

Motion compensated interpolation:

$$I_p(\vec{r}, t) = \frac{1}{2}\left(\hat{I}(\vec{r} - \vec{d}_f, t - \Delta t_f) + \hat{I}(\vec{r} + \vec{d}_b, t + \Delta t_b)\right) \ . \tag{2.3}$$

The displacement $\vec{d}_f$ (respectively $\vec{d}_b$) is evaluated by motion estimation between the images $I(\vec{r},t)$ and $\hat{I}(\vec{r},t-\Delta t_f)$ (respectively between $I(\vec{r},t)$ and $\hat{I}(\vec{r},t+\Delta t_b)$). Whereas in this description prediction and interpolation are independent, an alternative technique is proposed in [47] in which a joint motion compensated prediction and interpolation is performed.

In MPEG-II, the displacements are estimated on a block-based. Even though the motion estimation algorithm is not specified by the standard, block matching techniques [9, 3] (see Sec. 2.3.6 for more details) are commonly used. In case of 1/2 pixel accuracy motion estimation (or possibly higher), bilinear interpolation is commonly used in Eq. (2.1), (2.2) and (2.3).

In motion compensated coding, the motion information has usually to be coded and send to the decoder as overhead. This operation is commonly performed by a lossless coding technique. For instance, in MPEG-II a single translational displacement vector is assigned to a block of pixels. Its two components, horizontal and vertical, are differentially coded independently. A Variable Length Code (VLC) is used to encode the difference relative to the previous vector. A technique to code motion fields based on entropy coding and composite source model is proposed in [48], outperforming the above method. The previous techniques deal with block-based motion fields, in [49] the coding of dense motion vectors is addressed. Nevertheless, little research effort has been devoted to the subject of motion information coding.

Interlaced formats are common in TV and HDTV. They introduce useless difficulties in the processing of motion information. Two classical ways to deal with interlaced sequences, which have been adopted in MPEG-II, are *frame coding* and *field coding* [6, 50]. They are illustrated in Fig. 2.3. In the former case, the odd lines of a picture corresponds to the odd field, whereas the even ones corresponds to the even field. In the latter case, the top lines corresponds to the odd field and the bottom ones to the even field.

MPEG-II, as well as the method proposed in [50], supports both frame and field coding, and motion estimation and compensation is performed in each case as follows. In frame coding, the motion estimation and compensation is carried out between two pictures. In field coding, for the odd field, motion estimation and compensation is carried out with both the odd and even field in the previous picture, and the best prediction among the two is selected. Similarly, the even field is predicted from either the even field in the previous picture, or the odd field in the same picture.

Nevertheless, none of these two techniques, namely frame coding and field coding, gives optimal performances. Actually, on the one hand in case of frame coding and fast motion, high spatial frequency components are artificially introduced in the pictures. On the other

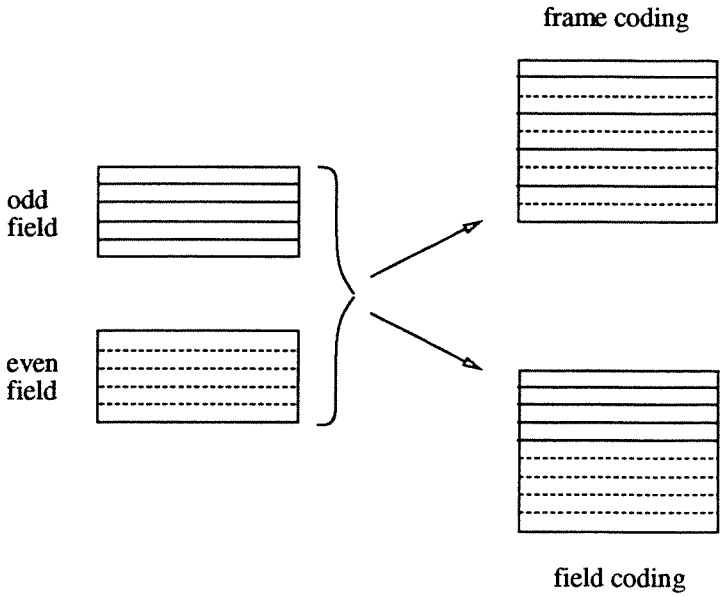

Figure 2.3: Frame and field coding.

hand, in case of field coding, the high redundancies between fields in still area cannot be fully exploited. Alternatives have been proposed in [51, 52, 53]. In [51, 52], the input signal is separated into odd and even fields. The even fields are directly coded, whereas the odd fields are encoded by motion compensated interfield interpolation. Similarly, in [53] a technique handling efficiently both progressive and interlaced formats is proposed. In case of interlaced source materials, odd fields are directly coded as a progressive sequence, and even fields are predicted by spatial and temporal interpolation of the corresponding decoded odd fields.

Color video signals are commonly represented by a luminance component Y, and two chrominance components U and V [54]. In most of the video coding schemes, the three components Y, U and V are coded independently. Due to their lower perceptual relevance, the chrominance components are commonly spatially subsampled. Furthermore and for the same reason, higher compression ratios are usually achieved for these components. Consequently, the U and V components correspond to a relatively small part of the overall bit rate. Besides, due to the higher resolution of the luminance signal, the motion vectors are estimated only on this component, and their amplitudes and spatial resolutions are adapted accordingly for the chrominance signals.

In Chap. 3, a simulation environment based on hybrid intraframe/interframe coding is defined which is used throughout this dissertation to evaluate and compare motion estimation algorithms performances.

2.3 Review of motion estimation algorithms

Very different motion estimation algorithms have been proposed in the literature. These algorithms have been developed for very different applications such as computer vision, image sequence restoration or image sequence coding. They can be divided into four groups:

- transform-domain techniques,
- gradient techniques,
- pel-recursive techniques,
- block matching techniques.

In this section, first the notion of motion is introduced. Next, the basic assumptions common to most motion estimation algorithms are discussed. Then the four groups of motion estimation techniques are reviewed. Finally, block matching methods are emphasized, as the motion estimation algorithm proposed in this dissertation belongs to this last group. In particular, the drawbacks of these techniques as well as the possible solutions to overcome them are discussed.

2.3.1 The notion of motion

Let us first clarify the meaning of motion or motion estimation. The notion of motion is easily understood intuitively. Everybody knows to perceive and estimate the displacement of objects in the surrounding world. Physics provides a precise mathematical definition of the motion. The velocity or more commonly the motion $\vec{v}$ at a spatial location $\vec{r}$ and a time t is defined as the ratio of the displacement $\vec{d} = \vec{r}(t) - \vec{r}(t_0)$ by the infinitesimal time interval $\Delta t = t - t_0$

$$\vec{v}(\vec{r}, t) = \lim_{\Delta t \to 0} \frac{\vec{d}}{\Delta t} = \frac{d}{dt}\, \vec{r}\,. \tag{2.4}$$

However, due to the complexity of the human visual system, the perception of motion is not such a simple notion. It involves not only the physical stimulus, but also the perception and the judgment. Therefore, the *perceived motion* may differ from the *physical motion.* In this case we speak about *apparent motion.* For instance, it may happen that a motion is sensed when the scene does not actually contain motion. Extensive experiments have emphasized some properties of the human visual system related to the perception of motion. Several models have been proposed [55, 56, 57, 58], and this domain is an active research area. As images are intended to be viewed by human observers, it is clear that the field of image sequence processing can benefit from these results.

In applications of visual information communication, such as television or motion pictures, natural scene information is sampled along the temporal axis. The impression of continuous motion is preserved, thanks to the high temporal sampling frequency and to the integration capability of the eyes. For the current television systems, the latter is 50 Hz for PAL [59] (in Western Europe) and SECAM [60] (in Eastern Europe and Middle East), and 60 Hz for NTSC [61] (in North America and Japan), whereas it is only 24 Hz for motion pictures. The last number may seem surprising as the image quality is superior in motion pictures than in TV, this is due to the higher spatial resolution.

Due to the discrete nature of image sequence, the motion $\vec{v}$ and displacement $\vec{d}$ are related by

$$\vec{v} = \frac{\vec{d}}{\Delta t} , \tag{2.5}$$

where Δt is the temporal sampling interval, in other words the time interval between two consecutive images. These two quantities differ only by a factor Δt, and are therefore interchangeable. For instance we will speak about the motion vectors resulting of a motion estimation technique, even though the latter estimates displacements.

In a digital image sequence, the $4D$ space-time continuum is projected on a $3D$ discrete samples grid. A distinction is made between $2D$ *motion field* and *optical flow* [62]. The former is the projection on the $2D$ image plan of the $3D$ motion in the scene. The latter is the field associated to the spatio-temporal variations of intensity. In the ideal case, the optical flow corresponds to the $2D$ motion field, however it is not obligatory. For instance, it may happen that the optical flow is zero when a motion exists in the scene. Conversely, the optical flow may be nonzero due to illumination changes when the scene is stationary. When a motion estimation technique aims at finding the $3D$ motion of moving objects, it is important to estimate the $2D$ motion field. For this purpose, approaches based on parametric models have been proposed [63, 12]. As mentioned previously, in image sequence coding the motivation is fundamentally different. The resulting motion estimation techniques are based on spatio-temporal intensity variations, and are therefore estimating the optical flow. Nevertheless, we will speak without distinction about motion field and optical flow throughout this dissertation.

2.3.2 Basic assumptions

Most of the motion estimation techniques rely on the two following assumptions:

- the illumination is uniform along motion trajectories,
- the problems due to uncovered areas are neglected.

Even though these assumptions are not absolutely indispensable, they are very common in most proposed motion estimation algorithms and in particular in the ones discussed in the following. Nevertheless, algorithms relaxing these two assumptions have been studied.

In order to take into account the variations of illumination, different motion estimation techniques have been proposed in the field of computer vision and image analysis [64, 65, 66, 67] and in image coding [68, 69].

In order to overcome the problem of uncovered background, techniques have been proposed in [70, 71, 72, 73, 74] in the framework of video coding. These techniques store the background information in a long term memory. Whenever an area is uncovered, the information unavailable with classical prediction techniques can be retrieved from this background frame store. These algorithms are effective for video-conference and videophone sequences, which are characterized by a still background and moving foreground objects. Consequently, they are of interest for low bit rate image coding applications.

A further assumption often done in the field of video coding is that objects undergo translational motion. This subject will be discussed in more details in the review of motion estimation techniques.

The different motion estimation techniques which are discussed in the next sections are defined on a block-by-block or pel-by-pel basis. Techniques performing object-based motion estimation with an a priori knowledge of the scene content have also been developed [75, 76, 77]. In a first stage, the scene is represented in terms of objects. Then the motion estimation is carried out on these objects, rather than blockwise or pelwise. These techniques are particularly suitable for object-based coding schemes [78, 79].

Finally, in coding motion is commonly estimated between an original image and the previous decoded one. However, for simplicity the case where the motion is estimated between two original images is considered in the remaining of this section. The following notation is used. The image intensity at pixel location $\vec{r} = (x, y)^T$ and at time t is denoted by $I(\vec{r}, t)$, and $\vec{d} = (d_x, d_y)^T$ is the displacement during the interval Δt.

2.3.3 Transform-domain techniques

Transform-domain methods are applied on the coefficients resulting from a transform (for instance, the Fourier transform [80, 81, 82] or the Gabor transform [83, 84]). They are based on the relationship between transformed coefficients of shifted images.

In the case of the Fourier transform and translational motion, this relationship can be derived as follows. Defining $\mathcal{F}(\omega_x, \omega_y, t)$ as the $2D$ Fourier transform of the image $I(x, y, t)$,

and considering a translation by a displacement vector $(d_x, d_y)^T$ during the interval Δt,

$$I(x, y, t) = I(x - d_x, y - d_y, t - \Delta t) \ , \tag{2.6}$$

the respective Fourier transforms are related by

$$\mathcal{F}(\omega_x, \omega_y, t) = \mathcal{F}(\omega_x, \omega_y, t - \Delta t) \cdot e^{i(2\pi f_x d_x + 2\pi f_y d_y)} \ , \tag{2.7}$$

where $f_x = \omega_x / 2\pi$ and $f_y = \omega_y / 2\pi$. The translation results only in a phase variation of the Fourier coefficients.

The algorithm proposed in [80] consists in computing the phase difference between two images at a number of frequencies, thus generating an overdetermined system of linear equations. Solving this system gives an estimate of the displacement vector. However, the method requires that all objects move with the same displacement. Moreover, the method is much more complex in case of rotational motion or zoom.

The transform-domain motion estimation techniques do not have a widely spread use, especially in the field of image sequence coding.

2.3.4 Gradient techniques

Gradient techniques rely on the hypothesis that the image luminance is invariant along motion trajectories. Therefore a change in the image intensity $I(\vec{r}, t)$ is due only to a displacement $\vec{d}$. It is expressed by

$$I(\vec{r}, t) = I(\vec{r} - \vec{d}, t - \Delta t) \ . \tag{2.8}$$

The Taylor series development of the latter term gives

$$I(\vec{r} - \vec{d}, t - \Delta t) = I(\vec{r}, t) - \vec{d} \cdot \vec{\nabla} I(\vec{r}, t) - \Delta t \, \frac{\partial I(\vec{r}, t)}{\partial t} + \text{higher order terms} \ , \tag{2.9}$$

where $\vec{\nabla} = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})^T$ is the gradient operator. Neglecting the higher order terms (first order approximation), Eq. (2.8) becomes

$$\frac{\vec{d}}{\Delta t} \cdot \vec{\nabla} I(\vec{r}, t) + \frac{\partial I(\vec{r}, t)}{\partial t} = 0 \ . \tag{2.10}$$

Assuming the limit $\Delta t \to 0$ and defining the motion vector $\vec{v} = (v_x, v_y)^T = \vec{d} / \Delta t$, we obtain

$$\vec{v} \cdot \vec{\nabla} I(\vec{r}, t) + \frac{\partial I(\vec{r}, t)}{\partial t} = 0 \ . \tag{2.11}$$

The latter equation is known as the *spatio-temporal constraint equation* or the *optical flow constraint equation* [13]. Eq. (2.11) expresses nothing else that the initial hypothesis of invariant luminance along motion trajectories:

$$\frac{dI}{dt} = 0 \ . \tag{2.12}$$

As the image intensity change at a point due to motion gives only one constraint (Eq. (2.11)), while the motion vector at the same point has two components, the motion field, actually the optical flow, cannot be computed without additional constraints. In fact, only the projection of $\vec{v}$ on $\vec{\nabla} I$, in other words the component of $\vec{v}$ parallel to the intensity gradient, can be determined from Eq. (2.11). Therefore additional constraints must be introduced together with the spatio-temporal constraint in order to solve the optical flow [13, 85, 86, 87].

In [13], Horn and Schunck introduce a smoothness constraint, that is to minimize the square of the optical flow gradient magnitude

$$\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 \text{ and } \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2 \ . \tag{2.13}$$

Consequently, the optical flow is obtained by minimizing the following error term

$$\iint \left(\left(\vec{v} \cdot \vec{\nabla} I + \frac{\partial I}{\partial t} \right)^2 + \alpha^2 \left(\left(\frac{\partial v_x}{\partial x} \right)^2 + \left(\frac{\partial v_x}{\partial y} \right)^2 + \left(\frac{\partial v_y}{\partial x} \right)^2 + \left(\frac{\partial v_y}{\partial y} \right)^2 \right) \right) dxdy \ , \tag{2.14}$$

where α^2 is a weighting factor. This minimization problem is solved by the variational calculus and an iterative procedure. In [85, 86, 87], variations of this algorithm have been proposed. The above approach deals with images at a single resolution scale, hierarchical schemes based on the spatio-temporal constraint equation have been developed in [88, 89].

All these techniques allow to obtain a dense motion field, qualitatively interesting for motion analysis applications. However, from an image sequence coding point of view, they suffer of two serious drawbacks. First the smoothness constraint leads to an increased energy of the prediction error. Second, the dense motion field requires a high overhead information.

2.3.5 Pel-recursive techniques

Pel-recursive techniques can be considered as a subset of the gradient techniques. The spatio-temporal constraint is minimized recursively. The recursion is usually carried out on a pel-by-pel basis, leading to a dense motion vectors field. More generally, it can

be performed on a block-by-block basis. These methods have been developed for image sequence coding applications.

The first pel-recursive algorithm has been proposed by Netravali and Robbins in [8]. This algorithm is based on the minimization of the prediction error. The displaced frame difference DFD is defined as

$$\mathrm{DFD}(\vec{r}, t, \vec{d}) = I(\vec{r}, t) - I(\vec{r} - \vec{d}, t - \Delta t) \ , \tag{2.15}$$

Ideally, we would have

$$\mathrm{DFD}(\vec{r}, t, \vec{d}) = 0 \ . \tag{2.16}$$

In practice, the pel-recursive algorithm minimizes the DFD^2 (or possibly $|\mathrm{DFD}|$). In [8], the DFD^2 is iteratively minimized by the steepest descent technique,

$$\vec{d}^{\,(k+1)} = \vec{d}^{\,(k)} - \frac{\epsilon}{2} \nabla_{\vec{d}} \, \mathrm{DFD}^2(\vec{r}, t, \vec{d}^{\,(k)}) \ , \tag{2.17}$$

with a gain $\epsilon > 0$, and k denotes the iteration index. From the definition of the DFD, Eq. (2.15), we have

$$\nabla_{\vec{d}} \, \mathrm{DFD}^2(\vec{r}, t, \vec{d}) = 2 \ \mathrm{DFD}(\vec{r}, t, \vec{d}) \cdot \nabla_{\vec{r}} \ I(\vec{r} - \vec{d}, t - \Delta t) \ . \tag{2.18}$$

Substituting Eq. (2.18) in Eq. (2.17), the displacement vector update becomes

$$\vec{d}^{\,(k+1)} = \vec{d}^{\,(k)} - \epsilon \ \mathrm{DFD}(\vec{r}, t, \vec{d}^{\,(k)}) \cdot \nabla_{\vec{r}} \ I(\vec{r} - \vec{d}^{\,(k)}, t - \Delta t) \ . \tag{2.19}$$

The iteration from k to $k + 1$ is carried out either on one pel location, or from one pel to its consecutive neighbor. Improved algorithms based on the same principle have been proposed in [90, 91, 92, 93], a comparison between some of these algorithms is given in [3].

When the update of the displacement vector is based only on previously transmitted data (causality), the decoder is able to estimate the same motion field than the encoder. In this case, no overhead motion information is required, which is of course an advantage of these methods. However, it is obtained at a cost of an increased complexity at the decoder, as the latter should also estimate the motion field. Furthermore, the causality constraints these algorithms and reduces their prediction capability compared to non-causal methods. For instance, if the pel-recursive motion estimation technique (with recursion on pels) is combined with a DCT coding of the DFD, the decoder is unable to reconstruct the motion vectors. In this example, the recursion should be carried out on a block-by-block basis to allow the decoder to estimate correctly the motion field, resulting in a decreased accuracy. An alternative is to design non-causal pel-recursive algorithms and to transmit motion vectors, as in other motion estimation techniques.

Pel-recursive algorithms suffer from two more drawbacks. First, as the error function to minimize contains generally many local minima, the iterative procedure may converge to a local minimum rather than the global one. In particular, these algorithms are very sensitive to noise. Second, large displacements and discontinuities in the motion field cannot be efficiently handled.

2.3.6 Block matching techniques

Block matching motion estimation techniques are the most widely used in image sequence coding [9, 94, 3]. Recent standards such as MPEG-I [5, 30], MPEG-II [6, 31] and H.261 [7] are based on them, even though the algorithm to estimate the motion vectors is not specified explicitly.

Block matching algorithms are based on the matching of blocks (possibly features) between two images, minimizing a disparity measure. As these methods directly minimize the DFD, they are very suitable for image sequence coding. Furthermore, due to the block-based nature of these techniques, they require only a small overhead motion information.

The motion model for block matching algorithms assume an image composed of rigid objects in translational motion. Although this model is clearly restrictive, it is justified by the fact that complex motion can be decomposed as a sum of translational components. Considering the problem of predictive coding, motion estimation aims at finding the displacement vector $\vec{d}$ which allows to predict $I(\vec{r},t)$ from $I(\vec{r},t-\Delta t)$, that is,

$$I(\vec{r},t) = I(\vec{r}-\vec{d},t-\Delta t)\ . \tag{2.20}$$

In block-based motion estimation, the image of size $M \times N$ pixels is partitioned into blocks $B(i,j)$ of size $b \times b$ pixels (the blocks are supposed square for simplicity), with $i = 1,\ldots,M/b$, and $j = 1,\ldots,N/b$. The same displacement vector is assigned to all pixels within a block

$$\forall\ \vec{r} \in B(i,j)\ ,\quad \vec{d}(\vec{r}) = \vec{d}(i,j)\ . \tag{2.21}$$

Using block matching, the displacement vector is evaluated by matching the information content of a measurement window $\mathcal{W}$, of size $w \times w$ pixels (the window is supposed square for simplicity), with that of a corresponding measurement window within a search area $\mathcal{S}$, placed in the previous frame. The matching criterion is defined by

$$f(\vec{d}) = \frac{1}{w^2} \sum_{\vec{r}\ \in\ \mathcal{W}} \parallel I(\vec{r},t) - I(\vec{r}-\vec{d},t-\Delta t) \parallel\ , \tag{2.22}$$

where the most widely used distance measures are the quadratic norm $\parallel x \parallel = x^2$ (*mean square error* (MSE) criterion) and the absolute value $\parallel x \parallel = |x|$ (*mean absolute error*

(MAE) criterion). The displacement vector $d(i,j)$ assigned to the block $B(i,j)$ is estimated by searching the spatial position minimizing the matching criterion,

$$\vec{d}(i,j) = \arg \min_{\vec{d} \in S} f(\vec{d}) \ . \tag{2.23}$$

Another possible matching criterion is to maximize the cross-correlation between two images [95].

The absolute minimum of the matching criterion is reached only by an exhaustive search of a series of discrete candidate displacements within a maximum displacement range, this technique being called full-search block matching. Despite the heavy computations it requires, it is widely used in video coding, due to its simplicity and ease of hardware implementation. This algorithm has initially been designed to estimate displacements with a precision of one pixel. However, a higher precision can be obtained (e.g. $1/2, 1/4, \ldots$ pixel accuracy). For this purpose, the image intensity has to be interpolated at fractional pixel locations, increasing significantly the computational load. In practice, this stage is implemented as a post-processing, where the one pixel accuracy displacement vectors are refined to a fractional pixel precision.

In order to decrease the computational load of the full-search algorithm, fast search techniques have been proposed [9, 94, 96, 97, 98, 99, 100]. In [9], Jain and Jain propose a $2D$-logarithmic search. It tracks the direction of minimum disparity, checking five points (actually some have already been checked) at each step, and decreasing the distance between search points when the minimum is the center of the search location (see Fig. 2.4). Koga *et al.* [94] introduce a 3-step search as illustrated in Fig. 2.5. Height search points around the previously obtained minimum are checked at each step, except for the first step which has nine search points. Srinivasan and Rao [96] present a procedure named conjugate direction search (see Fig. 2.6). First, the minimum in the x-direction is estimated. Second, the minimum in the y-direction is determined, starting from the minimum obtained at the first stage. In the algorithm, a minimum is detected when the disparity at a location is smaller than the one obtained at the consecutive up and down (respectively left and right) locations. The technique proposed by Kappagantula and Rao in [97] is a combination of the $2D$-logarithmic search and the 3-step search, and the one by Ghanbari in [99] is a variation of the $2D$-logarithmic search. An orthogonal search procedure is proposed by Puri *et al.* in [98] where at each iteration four new locations are searched. The above fast search algorithms reduce the number of locations searched, Liu and Zaccarin introduce in [100] two fast search techniques based on motion field and pixel subsampling. Some comparative results between these different algorithms are given in the cited publications.

These fast search techniques allow to decrease significantly the computation time compa-

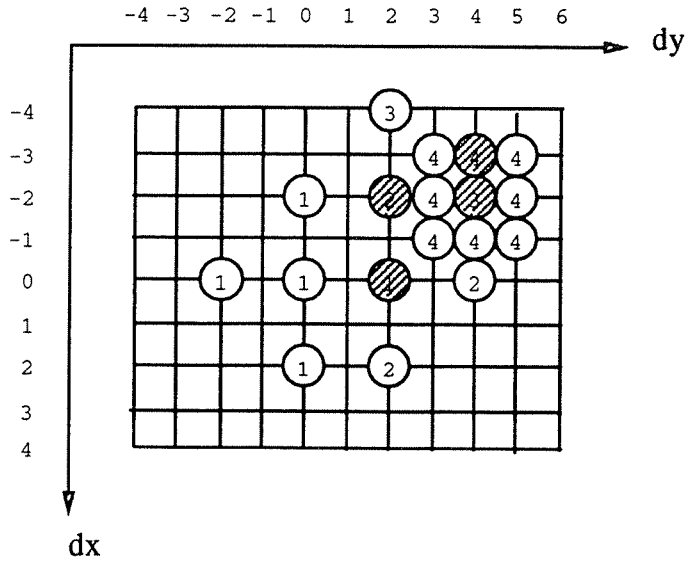

Figure 2.4: 2D-logarithmic search [9]: example for a displacement dx=-3 and dy=4, and 18 search positions (the numbers $i = 1, \ldots, 4$ in circle indicate the search points at step i, the shaded ones indicate the displacement after step i).

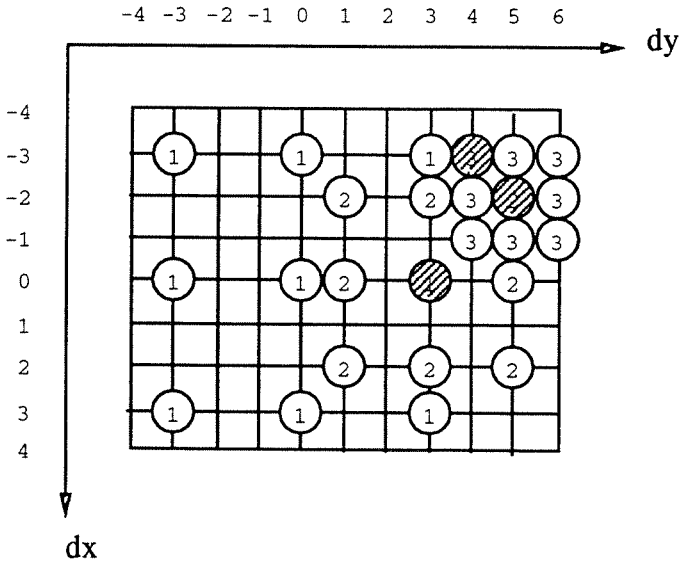

Figure 2.5: 3-step search [94]: example for a displacement dx=-3 and dy=4, and 25 search positions (the numbers $i = 1, \ldots, 3$ in circle indicate the search points at step i, the shaded ones indicate the displacement after step i).

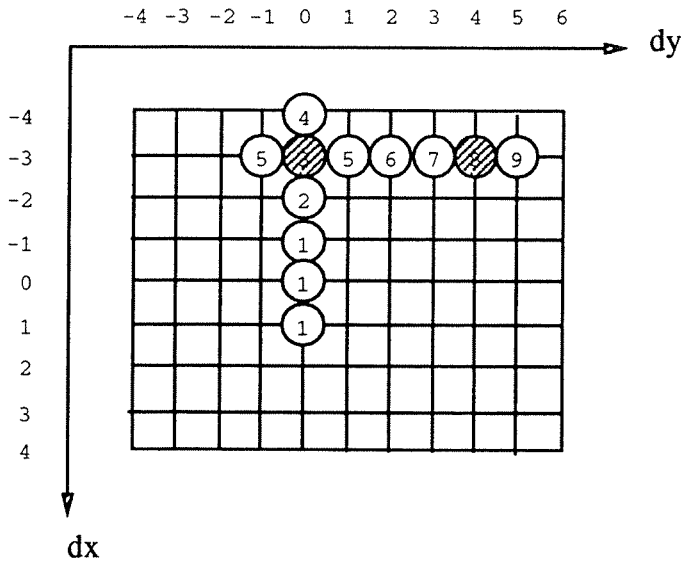

Figure 2.6: Conjugate direction search [96]: example for a displacement dx=-3 and dy=4, and 12 search positions (the numbers $i = 1, \ldots, 9$ in circle indicate the search points at step i, the shaded ones indicate the minimum obtained in each direction).

red to the full-search algorithm. However, the convergence towards the global minimum is guaranteed only when the matching criterion $f(\vec{d})$ is a monotonic function of $\vec{d}$. Figure 2.7 shows examples of the matching criterion function obtained for the sequences "Table Tennis" and "Mobile Calendar". In the first example, the function is smooth and monotonic and the fast search techniques perform very well, whereas in the second example the function has many local minima and the fast search algorithms are probably going to converge to one of them, rather than to the global one.

2.3.7 Drawbacks of block matching techniques and possible solutions

Despite their successful applications in video coding, classical block matching techniques, such as the full-search or the fast search algorithms described above, suffer several drawbacks. Among the major ones, we can mention: the difficulty to get simultaneously accurate and reliable motion fields while keeping a low overhead information, the block artifacts, the high computational complexity and the limitation of the translational motion model. In this section, these drawbacks are discussed in more details, as well as different solutions. In particular, the motion estimation technique proposed in this dissertation, which is based on block matching, provides a promising approach.

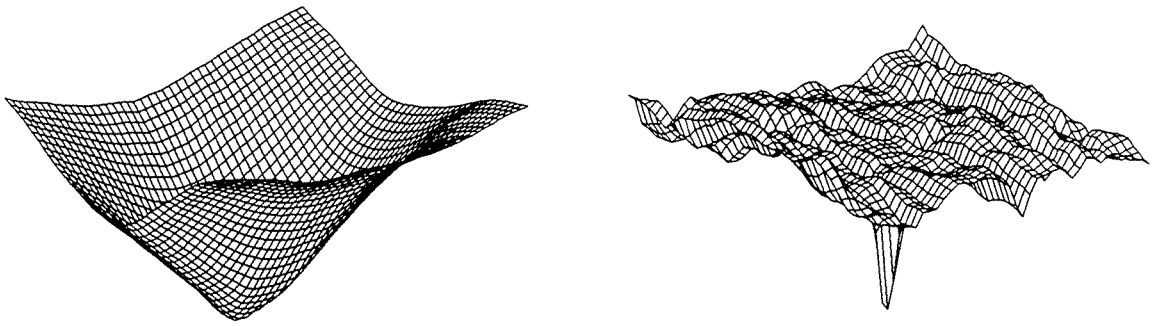

Figure 2.7: Typical matching criterion function $f(\vec{d})$, left) "Table Tennis", right) "Mobile Calendar" (maximum displacement: ± 21 pixels in both directions, matching window size: $\pm 32 \times 32$ pixels).

As already mentioned, the goal of a motion estimation algorithm in image coding is to provide an accurate prediction, while keeping a low coding cost overhead information. The fact that the obtained displacement vectors represent the motion in the scene is not an intrinsic goal. Nevertheless, in order to avoid artificial discontinuities in the motion compensated prediction and to reduce the transmission cost of the motion vectors, it is desired to obtain a smooth motion field. In other words, the motion vectors should be representative of the true motion in the scene. Therefore, a good motion estimation algorithm should simultaneously provide an accurate prediction minimizing the DFD energy, a low cost side information, and a reliable motion field close to the true motion in the scene.

According to the matching procedure, Eq. (2.22) and (2.23), it is obvious that the best prediction is achieved when the size w of the matching window $\mathcal{W}$ is equal to the size b of the block of pixels $B(i,j)$ to which the displacement is assigned. This size is the most critical parameter of the block matching motion estimation techniques. It determines the accuracy and the reliability of the motion field, as well as, the amount of side information. Let us now analyze in more details the conflicting requirements on both the matching window size and the block size.

The accuracy and reliability of the motion estimate depends on the size of the matching window, as well as the amount of motion. In order to cope with large displacements, a large search area is required. In this case, a large matching window is needed to get reliable estimates. The last remark is particularly justified when a fast search technique is applied. However, the estimate becomes inaccurate if the motion inside the matching window is not constant. Besides, small windows are required for local and accurate estimates. However, in the case of large displacements, there may be no correspondence between two small windows located in two consecutive images, and the estimate tends

to be unreliable. Therefore, the necessity to have a small window in order to obtain an accurate motion field, and the need of a large matching window to get reliable estimates define conflicting requirements.

The accuracy of the motion estimation and the amount of overhead information depends on the block size. The basic assumption is that the pixels within a block have the same motion vector. The smaller the block size, the more realistic is this hypothesis, and therefore the more precise is the obtained motion field. However, the price to pay is a larger overhead information. Conversely, a large block size requires less overhead motion information, but leads to a motion field with a decreased accuracy. In particular, due to the block-based nature of the method, the latter may introduce block artifacts in the motion compensated image, especially along the edge of moving objects. Consequently, the need on the one hand of a small block size to obtain an accurate motion estimation, and on the other hand of a large block size to keep a small overhead information are in opposition.

As a result of the above remarks, the matching window size w and the block size b, which should be equal in order to achieve the optimal prediction, are subject to conflicting requirements. Consequently, the block matching motion estimation techniques lead potentially to noisy motion fields which do not correspond to the true motion in the scene. The discontinuities in the motion field decrease the efficiency of the subsequent coding technique, introducing noticeable and disturbing artifacts. Furthermore, a noisy motion field is more expensive to code, when using an entropy coder. Finally, block artifacts are present in the motion compensated image, and the motion estimation and compensation techniques performs poorly on moving edges. In this dissertation, we aim at solving these problems.

The above described algorithms deal with images at a single resolution scale. In order to overcome the conflicting requirements on the matching window size, and thus to obtain simultaneously accurate and reliable motion fields, hierarchical algorithms have been proposed [14, 15]. The multigrid block matching technique discussed in Chap. 4 is based on the same idea of a procedure at different resolution scales [101, 102, 53]. In this technique, the motion vectors are iteratively refined on a set of grids. Coarse grids provide reliable estimates, which are accurately refined on finer grids. On the one hand, the algorithm is quasi-optimal in terms of minimizing the matching criterion, when compared to the full-search technique. On the other hand, resulting motion fields are smooth and close to the true motion in the scene. Finally, the multigrid algorithm has a much lower computational complexity. This subject is addressed in more details in Chap. 4.

The hypothesis that each pixel within a block move with the same displacement is fundamental in block matching techniques. In order to solve the contradiction on the block size, and therefore to obtain accurate motion fields while keeping a low overhead motion infor-

mation, a locally variable block size block matching algorithm has been introduced [16]. In Chap. 5, we propose similarly a locally adaptive multigrid algorithm [102, 53]. It is based on a quad-tree segmentation, which generates large blocks in uniform area, and small blocks in highly detailed ones. This way, the accuracy of the motion field is improved on moving edges. Simultaneously the overhead information is reduced in uniform regions. The advantage of the quad-tree decomposition is that it requires a low overhead information to code the segmentation information. This algorithm is discussed in details in Chap. 5, simulation results showing the improvement of this method when compared to full-search block matching are also given in [103, 104, 105] .

In order to overcome the problem of block artifacts in the motion compensated frame, due to the hypothesis that each pixel within a block as the same motion, different techniques have been investigated. A simple method to reduce the above mentioned drawback is to use overlapped windows [106, 107, 108, 109]. In [107, 108, 109], overlapped windows are used for both motion estimation and compensation, whereas in [106] only the latter case is considered. A very different technique, based on control gird interpolation, has been proposed in [110]. The basic idea of control grid interpolation is the following. First, spatial displacements are specified for a small number of points in an image, named control points and normally chosen as vertices of a rectangular gird. Next, the displacement of the other points is determined by interpolating between the control points. Block matching algorithms can be considered as a trivial special case of control grid interpolation in which interpolation is performed by nearest-neighbor, block artifacts being a result of the latter operation. This drawback is solved by the control grid interpolation using a higher order interpolation, for instance bilinear interpolation as in [110]. In [111], a very similar algorithm is proposed. The grid is composed of triangular patches and an affine transform is used to represent the transformation of these patches.

Another approach to solve the problem of block artifacts is to segment the block-based motion field. In [112], blocks corresponding to moving edges are segmented, taking into account the information of the previous frame. Similarly, in Chap. 6, we investigate another approach which segments the block-based motion field by means of *vector quantization* (VQ) [113, 114]. In this method, blocks which contain several objects moving in different directions are segmented. The segmentation is approximated by a finite set of different patterns. The capability of VQ to provide an efficient way to represent these patterns is exploited, leading to small overhead segmentation information and greatly improved prediction along edge of moving objects.

Finally, in the standard block matching technique, the motion is restricted to translation. Nevertheless, block matching based motion estimation algorithms relaxing the latter constraint have been investigated. In [115], an affine model for image matching is proposed, where each block undergoes an affine transform, instead of a translation in the

standard technique. Similarly, a generalized block matching algorithm is proposed in [116] which includes complex motion, as rotation or nonlinear deformation.

As a conclusion, despite the above mentioned drawbacks promising solutions exist. Therefore, block matching motion estimation techniques seem very appropriate in image sequence coding.

2.4 Summary

In this chapter, in a first stage the most well-known techniques to reduce temporal redundancies in image sequence coding have been discussed. The most widely used approach is based on the principle of hybrid coding, namely on the combination of motion estimation and compensation with an intraframe technique to code the resulting prediction error. Some specific problems relative to the application of motion compensation in an hybrid video coding scheme have been addressed.

Finally, motion estimation algorithms have been reviewed. Block matching techniques appear to be very suitable for video coding applications. Full-search and fast search techniques have been discussed. Both suffer several drawbacks, which have been pointed out. In particular, accuracy and reliability on the one side, and accuracy and low overhead information on the other side, define both conflicting requirements.

The locally adaptive multigrid block matching motion estimation technique proposed in this dissertation aims at overcoming these contradictions. Due to its multigrid structure, it provides simultaneously an accurate and reliable motion field. In addition, it requires a much lower computational complexity when compared to the full-search technique. Furthermore, the local adaptation allows a more accurate motion field in detailed area, and a decreased number of motion vectors in uniform area. Finally, a VQ-based segmentation of the motion fields overcomes the block artifacts problem and provides more precise motion vectors on moving edges with a low overhead segmentation information.

Chapter 3

Simulation environment for motion estimation algorithms evaluation

3.1 Introduction

In image sequence coding, the two key features of a motion estimation algorithm are a good motion compensated prediction and a low cost side information. The former can be evaluated by the DFD energy, whereas the latter can be estimated by the entropy of the motion vectors. Therefore, these two measures give an insight on the algorithm performances. However, they are not sufficient to enable the evaluation of the overall performances of a motion estimation algorithm. In particular, in order to compare the bit rate corresponding to DFD and motion vectors information, simulations within a video coding scheme have to be carried out. In addition, different motion estimation techniques may lead to different artifacts in the reconstructed sequence. Furthermore, as the motion is estimated between an original and a reconstructed frame, the motion estimation sensitivity to coding artifacts introduced in the reconstructed frame is an important feature, in particular robustness against noise. Consequently, it should be pointed out that the performances of a motion estimation algorithm depends on the subsequent coding techniques, as well as, the target bit rate.

Taking into account the above remarks, in this chapter a simulation environment is defined which is used throughout this dissertation to evaluate and compare motion estimation algorithms performances in practical applications. Three different hybrid interframe/intraframe coding schemes are considered (corresponding to Fig. 2.1) to provide more reliable results. The first one is based on a wavelet transform, the second one on an interframe DPCM, and the third one on a segmentation of the DFD.

The goal of this simulation environment is to compare as accurately as possible motion estimation algorithms and not to provide overall video coding performances. The following choices have been made. We consider a GOP of 12 pictures and predictive coding (i.e. $N_{GOP} = 12$ and $N_{B-pict} = 0$ in Fig. 2.2, see Sec. 2.2.1). The input sequences are progressive, actually obtained by de-interlacing of interlaced source materials. Only the luminance signal is coded.

In order to compare the performances of video coding schemes, the two following comparisons can be made: the reconstructed images quality for a given bit rate, or conversely the bit rate for a given reconstructed images quality.

Ideally, the quality of the reconstructed sequences should be estimated by subjective tests [117]. In practice, it is evaluated by the *peak-signal-noise-ratio* (PSNR) due to the difficulty of subjective tests. The PSNR is defined as

$$\mathrm{PSNR} = 10 \log_{10} \left(\frac{255^2}{\frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2} \right) , \tag{3.1}$$

where x_i (respectively $\hat{x}_i$), $i = 1, \ldots, N$, are the samples of the original (respectively reconstructed) signal. It should be pointed out that the PSNR is sometimes a poor measure of the visual quality and it is widely used in coding due to the lack of perceptually reliable visual quality measures

To measure the DFD energy and motion vectors entropy, the *energy* E and *entropy* H (0th-order entropy) of a signal x_i are defined as

$$E = \frac{1}{N} \sum_{i=1}^{N} x_i^2, \tag{3.2}$$

and

$$H = -\sum_{i=1}^{N} p(x_i) log_2(p(x_i)) \ , \tag{3.3}$$

where $p(x_i)$ is the probability of x_i.

Software simulations are carried out on the Cray-2 and Cray-YMP supercomputers from the Swiss Federal Institute of Technology at Lausanne.

In the remaining of this chapter, in Sec. 3.2 the test sequences used in simulations are presented. The video schemes are described in Sec. 3.3. Finally, Sec. 3.4 draws a summary of the chapter.

3.2 Test sequences

3.2.1 Video signals formats

The recommendation 601 from the Comité Consultatif International des Radiocommunications (CCIR) defines a digital source format for video applications, referred to as CCIR-601 [54]. The recommendation H.261 from CCITT defines another source format, named common intermediate format or CIF [7]. Table 3.1 summarizes the characteristic of these two formats.

3.2.2 De-interlacing

The CCIR-601 is an interlaced scan format. The interlacing introduces difficulties in the processing of motion information. As in this chapter the goal is to define a simulation environment to evaluate motion estimation algorithms, it is preferable to avoid these difficulties. For this purpose, the interlaced scan is converted to a progressive scan.

	scan	rate [Hz]	subsampling	lines	pixels per line
CCIR-601	interlaced	50	4:2:2		
luma Y				288	704
chroma U,V				288	352
CIF	progressive	29.97	4:2:0		
luma Y				288	352
chroma U,V				144	176

Table 3.1: CCIR-601 and CIF source formats.

Several algorithms have been studied for interlaced-to-progressive conversion. Techniques based on linear spatial interpolation, nonlinear spatial interpolation [118, 119, 120], median filtering [121] or spatio-temporal interpolation [122] have been proposed. In our case, the interlaced-to-progressive conversion is performed by spatial interpolation as follows [123, 124]. Only one field parity is considered, and the number of lines of these fields is doubled. The latter operation is carried out by spatial interpolation using a gradient based compensation, similar to the method proposed in [119]. The direction of maximum correlation between successive rows in the processed field is determined. For this purpose a displacement between two consecutive rows of the field is locally estimated by solving a 1-D constraint equation on blocks of the field. Then the number of lines is doubled by interpolating between consecutive lines using the previously evaluated interline displacement, as shown in Fig. 3.1.

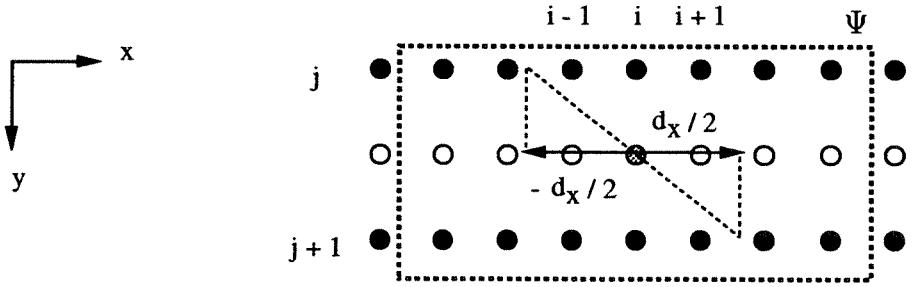

Figure 3.1: Interline spatial interpolation. The black dots indicate the rows of the field, the white ones the missing row to interpolate, and the grey dot represents the current pixel for which displacement is estimated, whereas the dashed box delimits the window Ψ.

The displacement estimation and the interpolation are performed in the following way. For each pixel of the missing row, a displacement d_x is estimated. Let $I(x, y)$ be the image intensity at the spatial location (x, y). If the interline velocity v_x is assumed uniform inside a window Ψ around the pixel, the 1-D constraint equation can be straightforwardly

derived, that is

$$v_x \frac{\partial I}{\partial x} + \frac{\partial I}{\partial y} = 0 \quad , \quad \forall (x,y) \in \Psi \ . \tag{3.4}$$

In the discrete case, velocity and displacement are equivalent, $v_x = d_x$. By integrating this equation over the window Ψ, an error term is obtained:

$$\iint_{\Psi} \left(d_x \frac{\partial I}{\partial x} + \frac{\partial I}{\partial y} \right)^2 \ dxdy. \tag{3.5}$$

Minimizing the error term according to d_x, we obtain the following expression

$$d_x = - \frac{\iint_{\Psi} \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \ dxdy}{\iint_{\Psi} \frac{\partial I}{\partial x} \frac{\partial I}{\partial x} \ dxdy} \ . \tag{3.6}$$

In Eq. (3.6), the derivatives are computed by finite difference, and the integration is simply replaced by a summation over the pixels (i,j) in the window Ψ. Since each missing pixel has an associated displacement vector, the interpolation is performed in the direction of displacement, as described in Fig. 3.1. The interpolated pixel value is given by

$$I(i, j + \frac{1}{2}) = \frac{1}{2} \left(I(i - \frac{d_x}{2}, j) + I(i + \frac{d_x}{2}, j + 1) \right) \ , \tag{3.7}$$

where the values of $I(i - \frac{d_x}{2}, j)$ and $I(i + \frac{d_x}{2}, j + 1)$ are obtained by linear interpolation between two adjacent pixels.

By applying the above interlaced-to-progressive technique to the luminance component of the CCIR-601 format, a progressive CCIR-601 format is obtained which characteristics are described in Table 3.2. This format is used in the simulations to compare motion estimation algorithms.

	scan	rate [Hz]	lines	pixels per line
CCIR-601 luma Y	progressive	25	576	704

Table 3.2: Progressive CCIR-601 source format.

3.2.3 Test sequences

Because of their different characteristics, the three sequences "Mobile Calendar", "Table Tennis" and "Flower Garden" have been selected. In particular, "Mobile Calendar" contains highly detailed moving areas, "Table Tennis" contains fast motions and a zoom, and

"Flower Garden" includes a pan and high-activity areas. Figure 3.2 shows the first frame of each sequence.

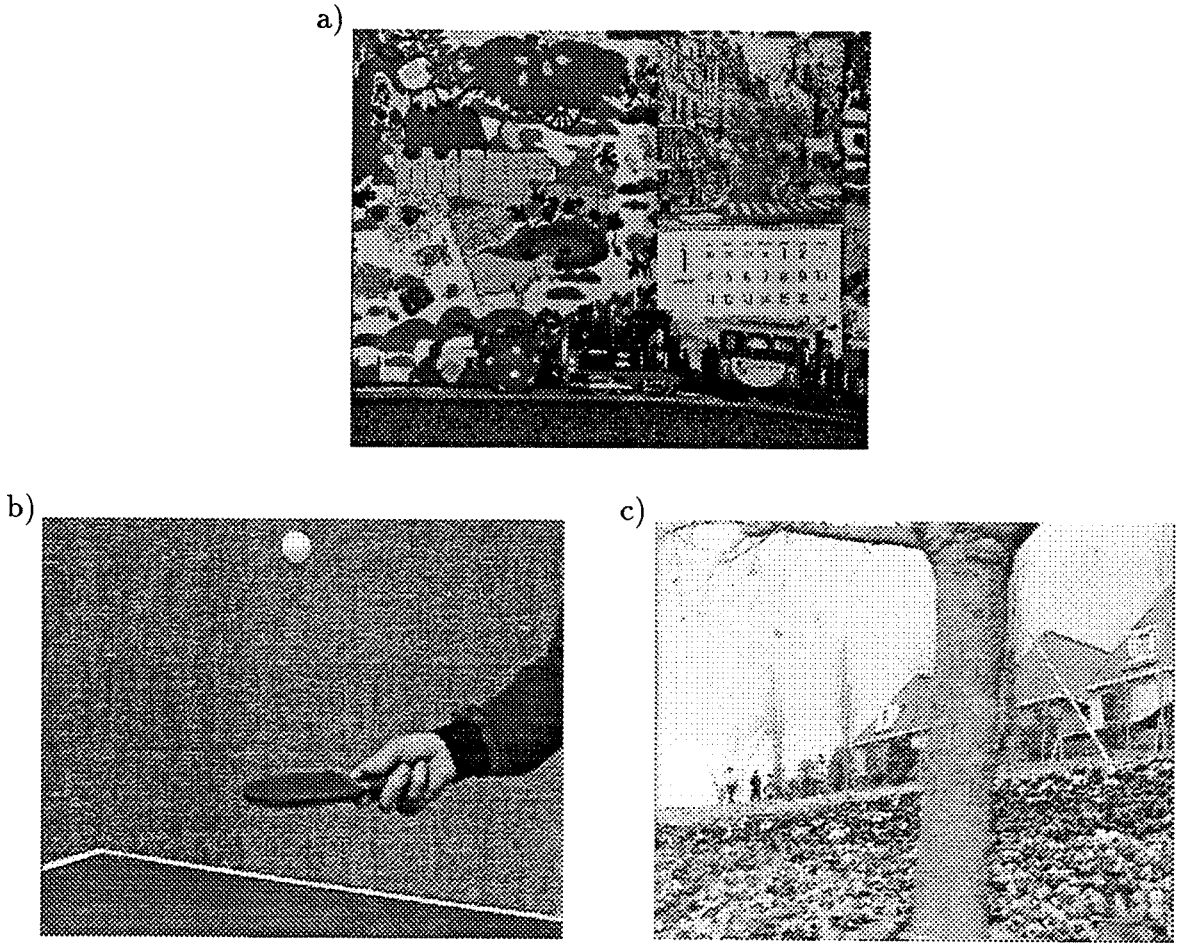

Figure 3.2: A frame of a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

3.3 Video coding schemes

In this section, the three different coding schemes considered for simulations are described. These three schemes are based on the principle of hybrid interframe/intraframe coding as depicted by Fig. 2.1 and discussed in Sec. 2.2. They differ only in the coding of the DFDs, the remaining of the systems being identical. The fact to use three different schemes is motivated by the desire of more reliable conclusions.

3.3.1 Scheme $\mathcal{A}$: motion compensated wavelet transform-based coding

The first scheme, referred to as scheme $\mathcal{A}$, belongs to the class of hybrid transform coding. Figure 3.3 shows the block diagram of the encoder. The schemes proposed by MPEG-I [5, 30], MPEG-II [6, 31] and H.261 [7] are using a DCT transform. In contrast, in this scheme a Gabor-like wavelet transform is applied [125]. The artifacts introduced by a wavelet transform are perceptually more acceptable when compared to the block artifacts characteristic of the DCT. Furthermore, the wavelet transform generates a multiresolution data representation of particular interest for generic coding [126, 53]. The coefficients resulting from the transform are uniformly quantized. These coefficients, as well as the motion vectors, are entropy coded with the adaptive arithmetic coder proposed in [127]. For more details on the wavelet transform used in this scheme, readers can refer to [128, 129, 125].

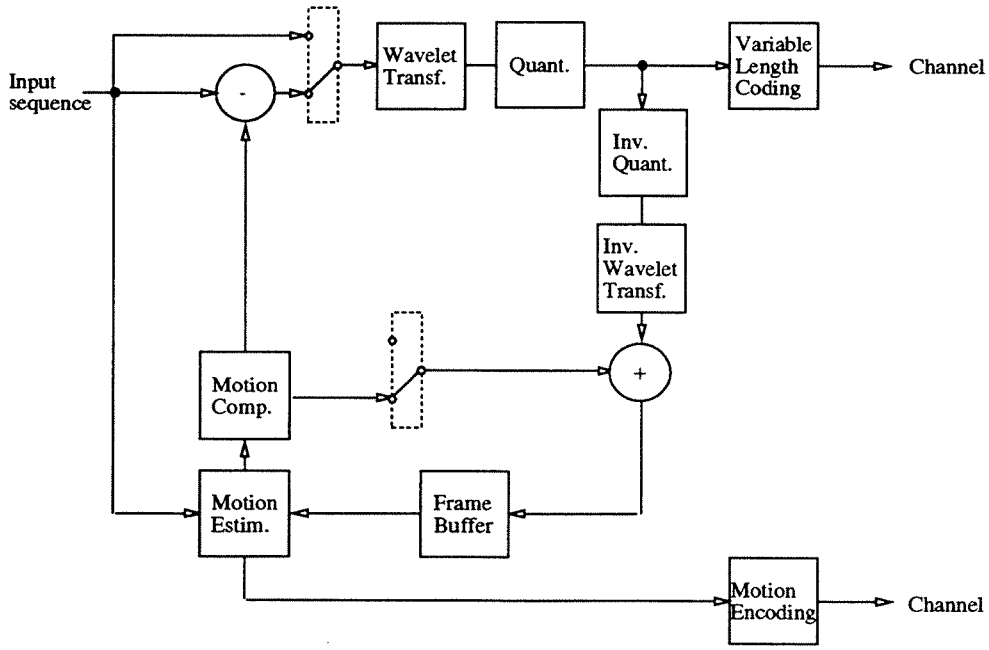

Figure 3.3: Scheme $\mathcal{A}$: motion compensated wavelet transform-based coding, encoder block diagram.

3.3.2 Scheme $\mathcal{B}$: motion compensated interframe DPCM coding

The second scheme, named $\mathcal{B}$, is based on an interframe DPCM technique, whereas in the intraframe mode (I-picture) the same wavelet transform as in the first scheme is

performed. Consequently, both schemes are similar, except that for the P-pictures no transform is applied in the second case. This simpler scheme is in accordance with the observation that the correlation in the DFD is very low, therefore the transform, whose goal is to decorrelate the DFD pixels, brings only small improvements. In particular, this scheme is consistent with a 0th-order Markov process [130] modeling of the DFD. The encoder block diagram is shown in Fig. 3.4. The quantization and entropy coder are identical to the first scheme.

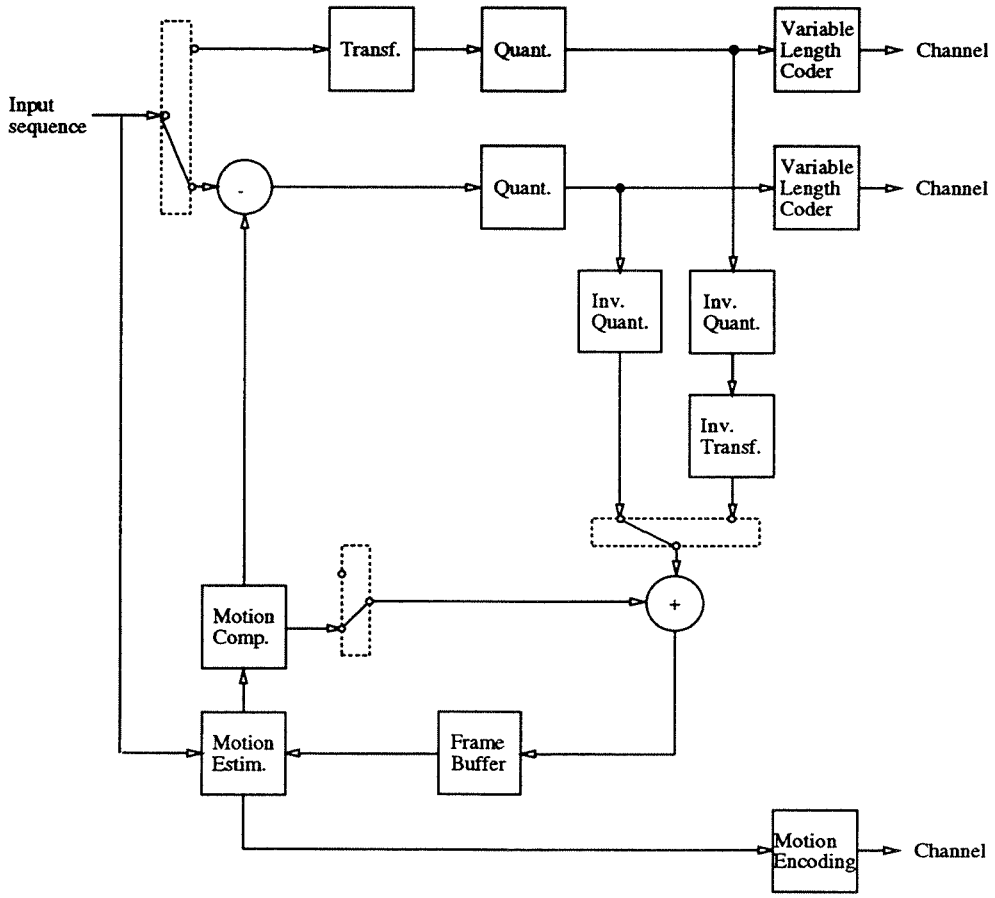

Figure 3.4: Scheme $\mathcal{B}$: motion compensated interframe DPCM coding, encoder block diagram.

3.3.3 Scheme $\mathcal{C}$: motion compensated segmentation-based coding

The scheme $\mathcal{C}$ is based on a morphological segmentation of the DFD [131, 105]. As observed in [32, 33], the correlation is very low in the DFD. Therefore, transform coding of the

DFD performs poorly and an approach based on segmentation is an attractive alternative. In this scheme, the DFD is segmented by morphological operators and only regions with a significant energy are coded. The pixel values in these regions are uniformly quantized and entropy coded, whereas the contours information is coded by a chain code [132]. The intraframe mode coding (I-picture), the quantization and the entropy coder are identical to the first scheme. Figure 3.5 shows the encoder block diagram.

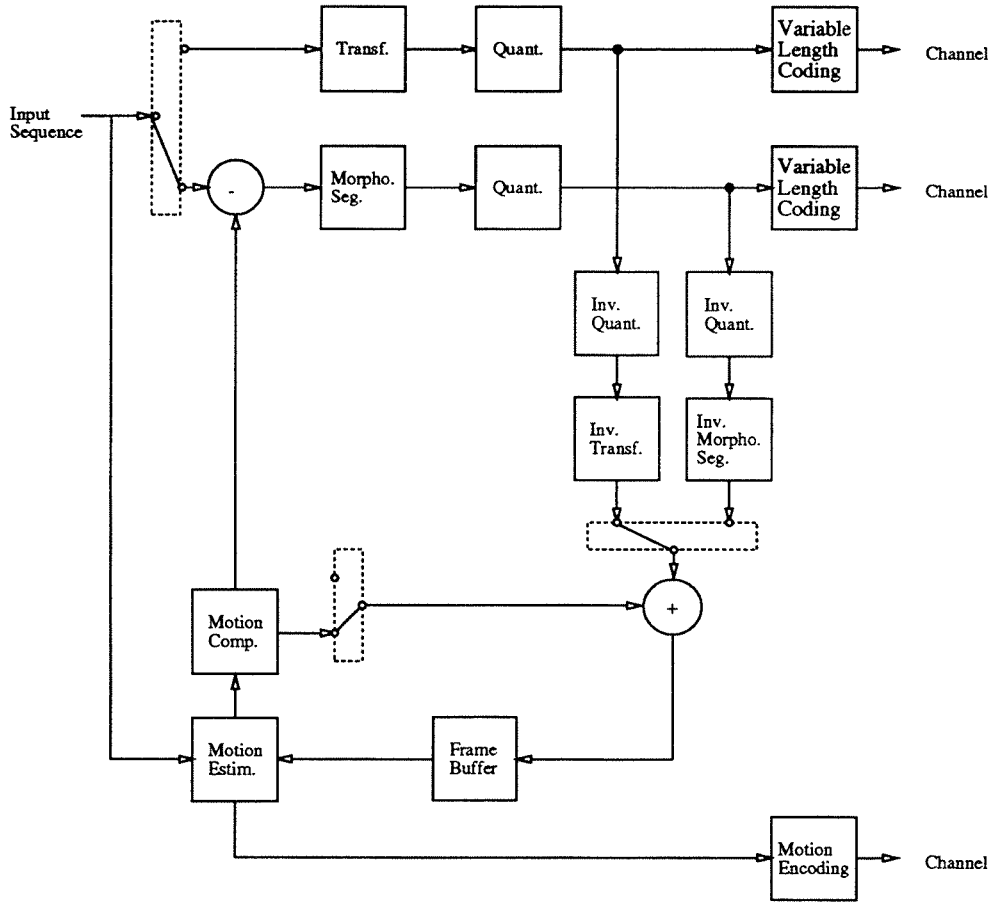

Figure 3.5: Scheme $\mathcal{C}$: motion compensated segmentation-based coding, encoder block diagram.

3.4 Summary

In this chapter, criteria to evaluate motion estimation algorithms have been defined. The DFD energy and the motion vectors entropy are two interesting measures. However, simulations within a video coding scheme have to be carried out to reliably compare

motion estimation techniques. Three test sequences have been selected, which are difficult in terms of motion. They have been converted from interlaced to progressive scan in order to avoid problems relative to interlacing. Finally, three test coding schemes have been defined. Due to the very different coding of the DFD, more reliable results, and therefore conclusions, are expected.

Chapter 4

Multigrid block matching motion estimation

4.1 Introduction

As natural scenes frequently contain motion at different scales, it is natural to introduce multi-level motion estimation algorithms. These techniques are based on the principle that large range displacements are estimated on large-scale structures, with low accuracy but robustness, and that short range displacements are estimated on small-scale structures, with high accuracy. An additional gain of the multi-level motion estimation techniques, compared to monoresolution ones, is that they require a much lower computational complexity.

Taking into account the above considerations, multi-level motion estimation techniques have been proposed by Glazer [88], Enkelmann [89], Baaziz [133], Anandan [14], and Bierling [15]. These algorithms differ in the motion estimation technique used (gradient-based or matching), in the way to build a multi-level structure (e.g. Gaussian pyramid, Laplacian pyramid, ...), in the control strategy (coarse-to-fine or more complex) and in the operators to map the motion vectors from one level to the other one.

More precisely, the algorithms proposed in [88, 89] use a gradient-based motion estimation, whereas the one in [133] uses a pel-recursive technique and the ones in [14, 15] use a block matching technique.

Various multi-level structures are discussed in [133], including low-pass Gaussian pyramids [134, 135], band-pass Laplacian pyramids [134] and wavelet pyramids. The algorithms in [88, 89] are based on a Gaussian pyramid and the one in [14] on a Laplacian pyramid, whereas in [15] images are low-pass filtered by local average. It should be pointed out that although the low-pass filtered image can be subsampled, this tends to produce undesirable aliasing.

The coarse-to-fine control strategy is the most widely used in these multi-level algorithms [88, 89, 133, 14, 15]. This strategy consists in a coarse estimation of the motion field on the lowest resolution level, and an iterative refinement of this estimate on the high resolution levels. In [89], a strategy is described which allows returns to lower resolution levels when the current estimate is unsatisfactory. This improvement avoids to be trapped by wrong initial estimates.

Finally, operators are required to map the motion vectors from one level to the consecutive one: interpolation in the case of coarse-to-fine transfer and extrapolation in the case of fine-to-coarse transfer. For instance, in a dyadic configuration the interpolation propagates the motion vector of a parent pixel or block on the coarse level to four children pixels or blocks on the fine level. The simplest interpolation is to duplicate the parent estimate (possibly rescaled), as in [88]. A bilinear interpolation is preferred in [89, 15],

and an overlapped projection in which each child selects the best initial condition among its parent pixel and three neighboring pixels is performed in [14]. In [133], which is based on a pel-recursive algorithm, the parent pixel motion vector is added to the candidate estimates of the monoresolution pel recursion.

In this chapter, we propose a new multi-level motion estimation technique: a multigrid block matching algorithm. It is based on the same principle that the above described motion estimation techniques. However it includes some new features which distinguish it. The formalism of multigrid theory applied to optimization problems provides a mathematical description of the algorithm. The multi-level structure is built on a set of grids with different resolutions (multigrid structure). The control strategy includes both coarse-to-fine and fine-to-coarse transfers. It prevents the optimization procedure to be trapped in local minima. Extrapolation for fine-to-coarse transfer of the motion field is performed by a median filter. In the interpolation for coarse-to-fine transfer each child block selects the best initial motion vector among four parent blocks. These extrapolation and interpolation operators avoid the propagation of wrong motion vectors estimates throughout the multigrid levels. Due to the multigrid structure, the algorithm allows to solve the conflicting requirements on the matching window size. Therefore, the resulting motion fields are simultaneously smooth and accurate.

As far as hardware implementation is concerned, a study of an earlier version of the multigrid block matching algorithm has been carried out in [136, 137, 138], showing the feasibility of a very efficient hardware implementation.

It should be noted that the above algorithm is only a first step towards the locally adaptive multigrid block matching technique which will be introduced in Chap. 5. This improved algorithm will adapt to the specific geometry of the image sequence by taking large grid sizes in uniform areas and small grid sizes in regions with details. This way, the contradiction on the block size will be overcome as well, the algorithm resulting in more accurate motion vectors and a lower overhead information.

This chapter is structured as follows. First, the principle of multigrid techniques is explained in Sec. 4.2. The multigrid block matching motion estimation technique is described in Sec. 4.3. Its most important features are discussed in details. Simulation results are presented in Sec. 4.4. Finally, based on the simulation results a conclusion is drawn in Sec. 4.5.

4.2 Multigrid techniques - history

Multigrid techniques have been developed to improve the convergence speed of classical numerical methods to solve discretized partial differential equations [139, 140, 141]. In this section, first the fundamental multigrid idea is introduced. Then, an adaptive local mesh refinement is discussed, which improves the classical algorithms. Finally, the application of multigrid techniques to discrete-state optimization problems is described.

4.2.1 Multigrid techniques to solve discretized partial differential equations

Let us consider the following discrete linear elliptic problem (boundary value problem)

$$\mathcal{L}_0 u_0 = f_0 \ , \tag{4.1}$$

where $\mathcal{L}_0$ is a linear operator and f_0 a function, both defined on a grid Ω_0, and u_0 is the unknown. This type of problem is very time consuming to solve numerically. Multigrid techniques [139, 140, 141] provide an efficient way to increase the convergence speed of classical algorithms. The basic idea of multigrid techniques is that the solution u_0 is a linear combination of several components, each having a specific scale.

The multigrid structure is composed of a sequence of increasingly coarser grids Ω_l, with $l = 0, 1, \ldots, L$, characterized by a mesh size Δ_l. Linear operators $\mathcal{L}_l$ are defined on Ω_l, as well as interpolation operators I_{l+1}^l for coarse-to-fine transfer and extrapolation operators I_l^{l+1} for fine-to-coarse one.

In this development, the 2-grid case is considered for simplicity. Supposing u_0^k an approximation of the exact solution u_0 in Eq. (4.1), and v_0^k the corresponding error

$$u_0 = u_0^k + v_0^k \ , \tag{4.2}$$

and defining the difference d_0^k as

$$d_0^k = \mathcal{L}_0 u_0 - \mathcal{L}_0 u_0^k \ , \tag{4.3}$$

from the linearity of $\mathcal{L}_0$, the following correction equation is obtained

$$\mathcal{L}_0 v_0^k = d_0^k \ . \tag{4.4}$$

A multigrid cycle consists of the following procedure:

1. an approximation u_0^k of the solution is computed on Ω_0 by a relaxation method,
2. the difference d_0^k is given by $d_0^k = f_0 - \mathcal{L}_0 u_0^k$,

3. the difference is restricted on Ω_1 (fine-to-coarse transfer) $d_1^k = I_0^1 d_0^k$,

4. the equation for the correction is solved on Ω_1, $\mathcal{L}_1 v_1^k = d_1^k$,

5. the correction is interpolated on Ω_0 (coarse-to-fine transfer) $v_0^k = I_1^0 v_1^k$

6. the new approximation is obtained $u_0^{k+1} = u_0^k + v_0^k$

The motivation of the method is that the relaxation dumps the high frequency errors. Consequently the correction equation can be solved on a coarser grid. The above 2-grid case is straightforwardly generalized to n-grid. Figure 4.1 illustrates different multigrid configurations.

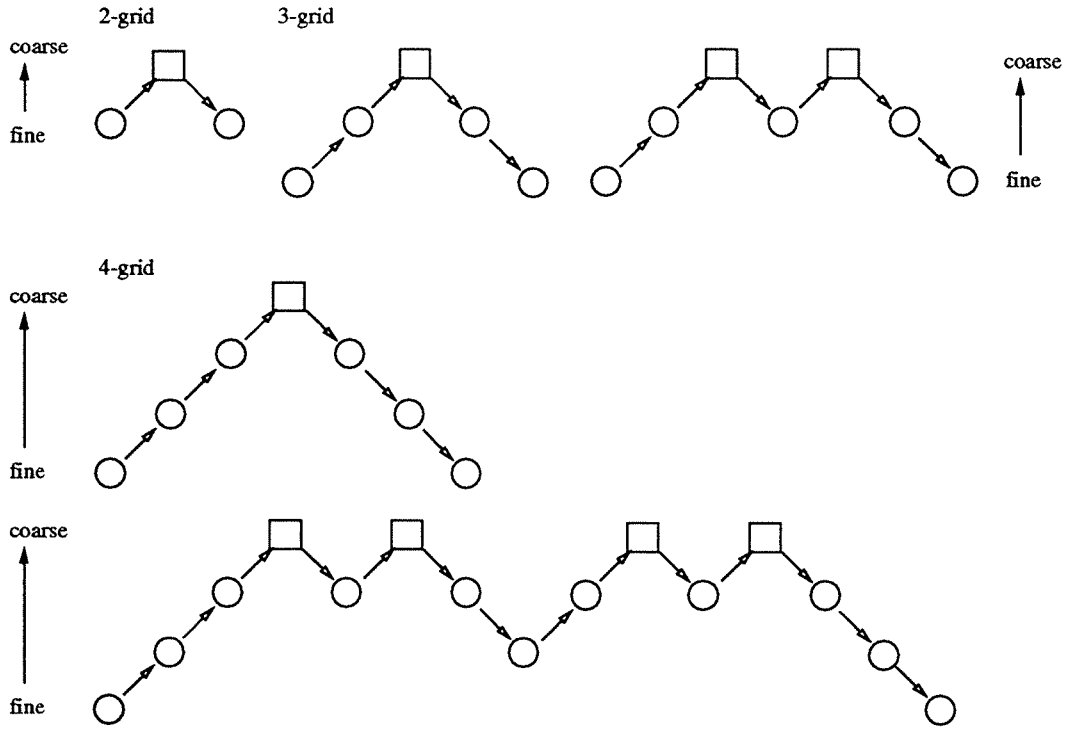

Figure 4.1: Multigrid cycles for different configurations: o = relaxation, □ = solving, ↗= fine-to-coarse, ↘= coarse-to-fine.

In the above development, the operators $\mathcal{L}_l$ have been supposed linear. A similar derivation is obtained in case of non-linear operators. Considering the 2-grid case, and the problem defined by Eq. (4.1), where $\mathcal{L}_0$ is now a non-linear operator, Eq. (4.3) becomes

$$d_0^k = \mathcal{L}_0 u_0 - \mathcal{L}_0 u_0^k = \mathcal{L}_0(u_0^k + v_0^k) - \mathcal{L}_0 u_0^k \ , \tag{4.5}$$

which leads to the following correction equation

$$\mathcal{L}_0 w_0^k = d_0^k + \mathcal{L}_0 u_0^k \ , \tag{4.6}$$

with $w_0^k = u_0^k + v_0^k$.

A multigrid cycle is similar to the procedure described above for the linear case, except that the step number 4 is changed to:

4. the equation for the correction is solved on Ω_1, $\mathcal{L}_1 w_1^k = d_1^k + \mathcal{L}_1 u_1^k$, and the correction term is given by $v_1^k = w_1^k - u_1^k$.

A further improvement of multigrid techniques consists in the introduction of an adaptive local mesh refinement [142]. Based on an error estimate, the mesh size Δ_l is varying locally. For instance, the mesh size is large in regions where the solution is smooth (small error), whereas it is small where the solution has discontinuities (large error). This way, the multigrid structure adapts to the specific problem and its geometry.

Several image analysis problems lead to a variational formulation, and consequently its associated partial differential equations system. The latter is solved by numerical relaxation methods. Therefore the above described multigrid techniques are straightforwardly applicable to a wide range of image analysis and vision problems, for instance the lightness and the shape-from-shading problems [143], as well as optical flow estimation by gradient-based techniques [88, 143, 89].

4.2.2 Multigrid techniques to solve optimization problems

Multigrid techniques have also been applied to solve discrete-state optimization problems [144]. These problems consist in minimizing a cost function, and are commonly solved by an iterative procedure. A solution is iteratively refined by a series of transitions. These latter correspond to a change in the state variables describing the system, and are accepted whenever they decrease the cost function. The difficulty of this optimization procedure is to avoid to be trapped in local minima. A way to overcome this difficulty is given by simulated annealing techniques [145], which allow, with a low probability, transitions increasing the cost function.

Another approach is provided by a multigrid algorithm. This approach is based on a process at different scales, and on the principle that a large-scale transition is accepted only after calculating its effect at all finer scales. This idea is motivated by the typical topology of local minima. Large-scale transitions allow to escape from a local minimum and its attraction basin, and to reach a new basin. To decide whether to prefer the new basin to the previous one, it is necessary to reach its minimum. This is achieved by

calculating the fine-scale effects of the large-scale transition. At this stage, a meaningful comparison with the previous minimum is obtained. The principle is used recursively, as the cost function is composed of attraction basin within attraction basin.

In [144], this method is applied to the spin-glass problem. In Sec. 4.3, its application to block matching motion estimation is presented.

4.3 Multigrid block matching motion estimation

Motivated by the desire of a multi-level motion estimation algorithm, a multigrid block matching is presented in this chapter. It is based on the multigrid theory presented in Sec. 4.2 and its application to optimization problems.

The accuracy and the reliability of the motion estimate define conflicting requirements on the matching window size (see Sec. 2.3.6). Based on a multigrid structure composed of a set of grids with different resolution, the multigrid block matching overcomes these conflicting requirements. Large displacements are estimated robustly on the coarse grids with large matching windows, and small displacements are solved accurately on the fine grids with small matching windows.

Hence, the algorithm leads to both smooth and accurate motion fields. It avoids discontinuities in the motion compensated frames and noise in the motion vectors. Furthermore, it is quasi-optimal in minimizing the matching criterion, when compared to the full-search block matching. Therefore, the DFD and the overhead information are more efficiently coded. In addition, the multigrid algorithm requires a greatly decreased computational complexity when compared to the full-search block matching.

The multigrid structure, the control strategy and the interpolation/extrapolation operators are discussed in details in the following.

4.3.1 Multigrid structure

Following the same development than in Sec. 4.2, a multigrid formulation of the block matching technique can be derived.

The multigrid structure is composed of increasingly coarser grids $\Omega_l, l = 0, 1, \ldots, L$, characterized by a mesh size Δ_l. For simplicity, the cells defined by the grid are assumed square and dyadic (i.e. $\Delta_{l+1} = 2\Delta_l$). Figure 4.2 illustrates the 3-grid structure. The finer grid Ω_0 corresponds to the final resolution of the block-based motion field.

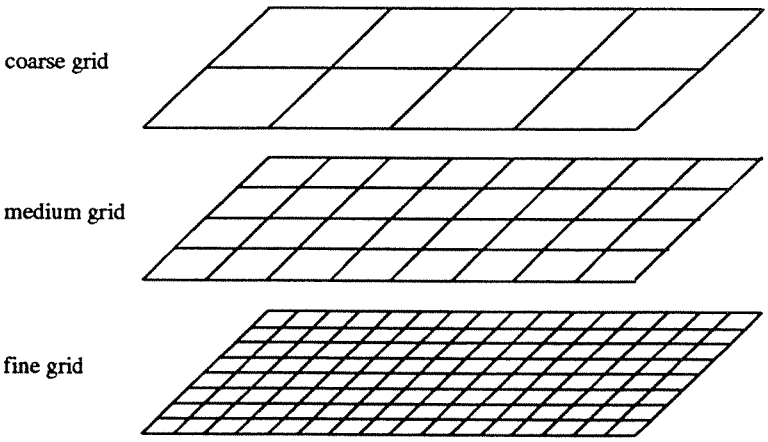

Figure 4.2: The 3-grid multigrid structure.

In the following development, the 2-grid case is considered. The derivation is easily generalized to n-grid. Let us define $B_l(i,j)$ the block partition on Ω_l, b_l being the corresponding block size, $\mathcal{W}_l$ the matching window on Ω_l, of size $w_l \times w_l$ pixels, $\mathcal{S}_l$ the search area on Ω_l, and the matching criterion

$$f_l(\vec{d}) = \frac{1}{w_l^2} \sum_{\vec{r} \in \mathcal{W}_l} \| I_l(\vec{r},t) - \hat{I}_l(\vec{r}-\vec{d},t-\Delta t) \| , \tag{4.7}$$

where $I_l(\vec{r},t)$, respectively $\hat{I}_l(\vec{r},t-\Delta t)$, is the image, respectively the reconstructed image, on the level l.

In the proposed algorithm, the blocks $B_l(i,j)$ correspond to the cells defined by the grid Ω_l, i.e. $b_l = \Delta_l$, $\forall\ l$. At each level, the matching window size w_l is chosen equal to the block size b_l. In order to obtain more robust estimates on the low resolution levels, images $I_l(\vec{r},t)$ and $\hat{I}_l(\vec{r},t-\Delta t)$ could be low-pass filtered (possibly subsampled) version of respectively the original image $I(\vec{r},t)$ or the reconstructed image $\hat{I}(\vec{r},t-\Delta t)$, as proposed in [88, 89, 15]. It appears in our simulations that this low-pass filter does not bring any improvements. Consequently, in this algorithm the original image and the reconstructed image are used at all levels

$$I_l(\vec{r},t) = I(\vec{r},t) \quad \text{and} \quad \hat{I}_l(\vec{r},t-\Delta t) = \hat{I}(\vec{r},t-\Delta t), \quad \forall\ l\ . \tag{4.8}$$

Therefore, the proposed algorithm relies on a multigrid structure and not on a multiresolution representation.

The block matching motion estimation problem defined on Ω_0 is

$$\forall\ \vec{r} \in B_0(i,j)\ ,\quad \vec{d}_0(\vec{r}) = \vec{d}_0(i,j) = \arg \min_{\vec{d}\ \in\ \mathcal{S}_0} f_0(\vec{d})\ . \tag{4.9}$$

If $\vec{d}_0^{\,k}$ is an approximation of the solution $\vec{d}_0$, and $\vec{\epsilon}_0^{\,k}$ the corresponding error,

$$\vec{d}_0 = \vec{d}_0^{\,k} + \vec{\epsilon}_0^{\,k}\ , \tag{4.10}$$

the optimization problem defined by Eq. (4.9) is equivalent to

$$\vec{\epsilon}_0^{\,k} = \arg \min_{\vec{\epsilon}\ \in\ \mathcal{S}_0'} f_0(\vec{d}_0^{\,k} + \vec{\epsilon}\,)\ . \tag{4.11}$$

The last equation is projected and solved on Ω_1,

$$\vec{\epsilon}_1^{\,k} = \arg \min_{\vec{\epsilon}\ \in\ \mathcal{S}_1'} f_1(\vec{d}_1^{\,k} + \vec{\epsilon}\,)\ , \tag{4.12}$$

with

$$\vec{d}_1^{\,k} = I_0^1\ \vec{d}_0^{\,k}\ , \tag{4.13}$$

where I_0^1 is the extrapolation operator for fine-to-coarse transfer. The new approximation is given by

$$\vec{d}_1^{\,k+1} = \vec{d}_1^{\,k} + \vec{\epsilon}_1^{\,k}\ . \tag{4.14}$$

The latter is down projected on Ω_0,

$$\vec{d}_0^{\,k+1} = I_1^0\ \vec{d}_1^{\,k+1}\ , \tag{4.15}$$

where I_1^0 is the interpolation operator for coarse-to-fine transfer. This last equation gives the new approximation of the solution on Ω_0.

The above described multigrid structure allows to solve the conflicting requirements on the matching window size. Large displacements are estimated robustly on the coarse grids with large matching windows, and small displacements are solved accurately on the fine grids with small matching windows. Taking into account this remark, large (respectively small) search areas are chosen on the coarse (respectively fine) grids.

The control flow within the multigrid structure is defined by the control strategy. The latter should prevent the procedure to be trapped in a local minimum.

The up- and down-conversion operators map the motion field between two grids in the iterative estimation and refinement of the motion vectors during the multigrid process. Their high importance is pointed out by simulations. Both should guarantee smooth and

robust motion fields. Thus, they should incorporate a spatial consistency of the motion field and avoid wrong motion vectors estimates, due to local minima of the matching criterion, to propagate throughout the multigrid levels.

In the remaining of this section, first some features of the multigrid structure specific to the considered implementation are discussed. Next the control strategy, as well as the up- and down-conversion operators for fine-to-coarse, respectively coarse-to-fine, transfer are addressed in details.

4.3.2 Multigrid structure - implementation choice

The following implementation choice has been made in the multigrid block matching algorithm for the remaining of this chapter.

A 3-grid structure as illustrated in Fig. 4.2 is considered. The smaller block size is 8×8 pixels, i.e. $\Delta_0 = 8$, $\Delta_1 = 16$, and $\Delta_2 = 32$. The choice of the final block size being 8×8 pixels is a good compromise between the motion estimation precision and the amount of overhead information. The 3-grid structure allows large displacements and requires low computational cost.

In order to reduce the computational complexity of the algorithm, a fast search technique, the modified n-step search, is applied at each level. Figure 4.3 illustrates the modified 3-step search. At the first step, the 9 locations defined by the set $(0, \pm 2^{n-1})$ are evaluated. The best estimate is the initial point for the next step, and at the i^{th} step, the 8 locations defined by the set $(0, \pm 2^{n-i})$ around the initial point are evaluated. Consequently, the resulting maximum displacement of the n-step search is $2^n - 1$. In order to evaluate large displacements on coarse grids, and small displacements on fine grids, a $(l+2)$-step search is performed on the l^{th} grid $(l = 0, 1, 2)$. The fast search technique combined with the multigrid structure allows to estimate large displacements with a very low computational complexity, when compared to a monoresolution full-search block matching algorithm.

The above fast search technique leads to motion vectors with one pixel accuracy. In some applications, a higher accuracy is desired. In this case, a post-processing which refines the one pixel accuracy motion vectors to a fractional pixel precision is performed. This operation requires the image intensity to be interpolated at fractional pixel locations. In the proposed algorithm, this interpolation is performed by bilinear interpolation. It is to note that this post-processing increases significantly the computational complexity. In Sec. 4.4, experiments investigate the optimal accuracy for the motion vectors.

Both the MAE and the MSE measures can be used in the matching criterion. In Sec. 4.4, simulation results comparing MAE and MSE performances motivate the choice of the

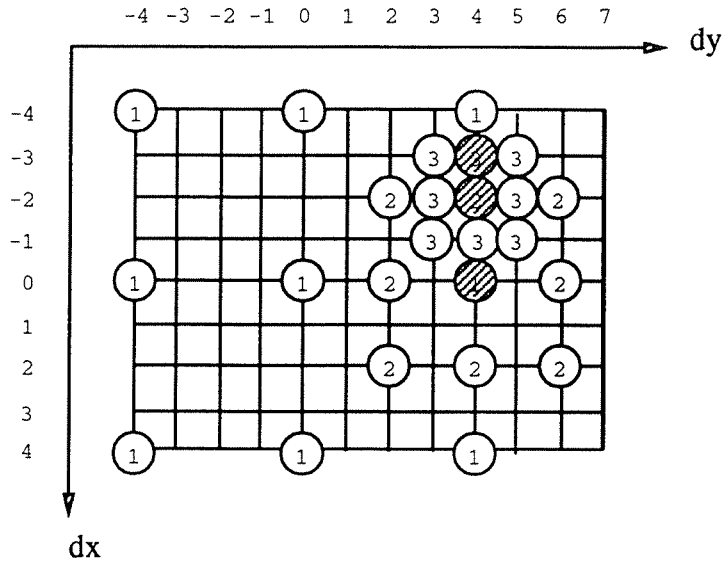

Figure 4.3: Modified 3-step search: example for a displacement dx=-3 and dy=4, and 25 search positions (the numbers $i = 1, \ldots, 3$ in circle indicate the search points at step i, the shaded ones indicate the displacement after step i).

former, due to its lower complexity and its easier hardware implementation.

Table 4.1 summarizes the characteristics of the multigrid structure.

grid	block size	match. window	search	step 1	step 2	step 3	step 4	max. displ.
Ω_0	8×8	8×8	2-step	± 2	± 1			± 3
Ω_1	16×16	16×16	3-step	± 4	± 2	± 1		± 7
Ω_2	32×32	32×32	4-step	± 8	± 4	± 2	± 1	± 15

Table 4.1: The multigrid structure.

4.3.3 Control strategy

The control strategy defines the control flow within the multigrid structure. The classical control strategy, which consists in a coarse-to-fine procedure, has been adopted in the multi-level motion estimation algorithms proposed in [88, 89, 133, 14, 15]. In [89] returns to low resolution levels are possible when the current estimate is unsatisfactory.

In the proposed multigrid block matching algorithm, two control strategies are considered: a coarse-to-fine and a fine-to-coarse-to-fine, illustrated in Fig. 4.4.

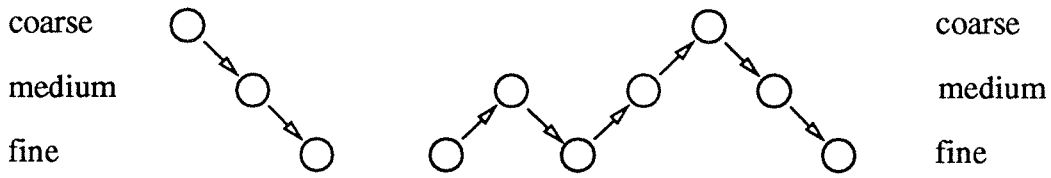

Figure 4.4: Control strategies: left) coarse-to-fine, right) fine-to-coarse-to-fine (o = n-step search, $\nearrow$= fine-to-coarse transfer, $\searrow$= coarse-to-fine transfer).

The coarse-to-fine procedure is motivated by the fact that coarse but robust estimates are obtained on the coarse grids, which are then accurately refined on the fine grids. The drawback of this technique is that large scales are solved first, and are taken as initial conditions to solve smaller scales. Consequently, if an incorrect estimate is obtained on a coarse level, it propagates to the finer levels.

In Chap. 2, Fig. 2.7 showed typical matching functions. For the sequence "Table Tennis", the function is smooth and monotonic. In this case, the coarse-to-fine strategy reaches easily the absolute minimum. However, for the sequence "Mobile Calendar", the function is noisy with a very narrow minimum. In this case, if the procedure begins on the coarsest grid, the n-step search is favorite to provide a displacement estimate far from the exact solution. Then, the refinement on the finer levels is unable to recover from the wrong initial estimate, and the procedure is trapped in a local minimum.

In order to overcome the above described problem, a fine-to-coarse-to-fine strategy is proposed, as illustrated in Fig. 4.4. This control strategy allows to solve independently each motion scales. Indeed, in a coarse-to-fine strategy, large scales are solved first, then small scales. Unfortunately, a small correction on a fine grid cannot recover from a wrong large displacement on a coarse grid. This observation motivates to solve first the small scales, then the large scales, as in the fine-to-coarse-to-fine strategy. When a wrong estimate is obtained on a fine grid, it does not affect the estimation on the coarser grids, as at these levels large displacements allow to escape from a local minimum and its attraction basin. Conversely, when small scales are accurately solved on the fine grid, the correction displacements are estimated as zero on the coarser grids.

With the control strategies described in Fig. 4.4, and the considered multigrid structure characterized by Table 4.1, the maximum displacement is ± 25 pixels for the coarse-to-fine control and ± 45 pixels with the fine-to-coarse-to-fine one.

Simulation results comparing coarse-to-fine and fine-to-coarse-to-fine control strategies are presented in Sec. 4.4.

4.3.4 Up-conversion

The up-conversion operator maps the motion field between two grids in the fine-to-coarse transfer. Figure 4.5 illustrates the correspondence of the block partitions $B_l(i,j)$ and $B_{l+1}(i,j)$ on Ω_l and Ω_{l+1} respectively.

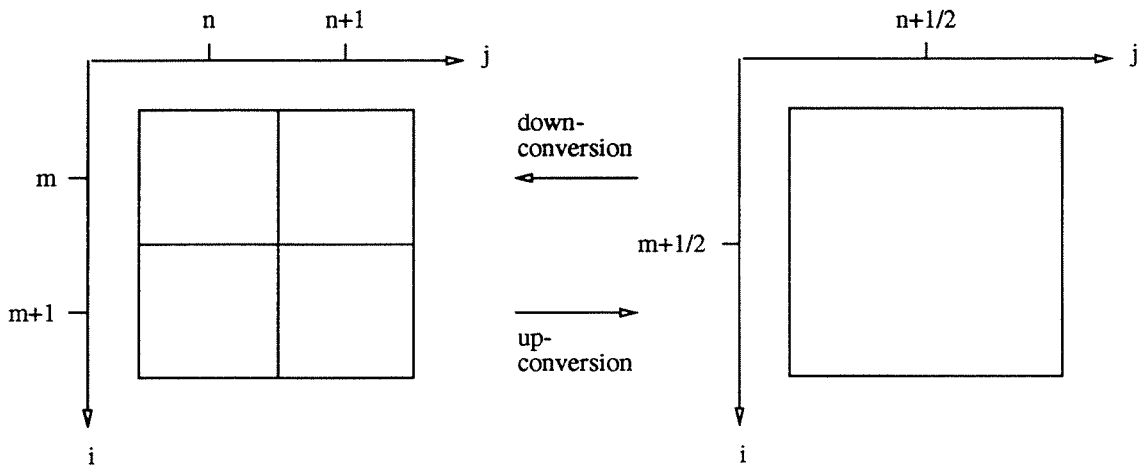

Figure 4.5: Block partition: left) $B_l(i,j)$ on Ω_l, right) $B_{l+1}(i,j)$ on Ω_{l+1}.

In the up-conversion from Ω_l to Ω_{l+1}, according to the dyadic multigrid structure, a new unique motion vector is generated on Ω_{l+1} from the four corresponding values on the finer gird Ω_l. The four displacements $\vec{d}_l(m,n)$, $\vec{d}_l(m,n+1)$, $\vec{d}_l(m+1,n)$ and $\vec{d}_l(m+1,n+1)$ corresponding respectively to the blocks $B_l(m,n)$, $B_l(m,n+1)$, $B_l(m+1,n)$ and $B_l(m+1,n+1)$ on the fine grids are projected on the block $B_{l+1}(m+\frac{1}{2},n+\frac{1}{2})$ on the coarse grid (see Fig. 4.5), the initial condition for the latter block, $\vec{d}_{l+1}(m+\frac{1}{2},n+\frac{1}{2})$, being given by the up-conversion operator.

The simplest operator is to take the new value as the mean of the four precedent values

$$\vec{d}_{l+1}(m+\frac{1}{2},n+\frac{1}{2}) = \frac{1}{4}\left(\vec{d}_l(m,n)+\vec{d}_l(m,n+1)+\vec{d}_l(m+1,n)+\vec{d}_l(m+1,n+1)\right) . \tag{4.16}$$

A more effective operator is to consider the new value as the median of these same four values

$$\vec{d}_{l+1}(m+\frac{1}{2},n+\frac{1}{2}) = \text{MED}[\ \vec{d}_l(m,n),\vec{d}_l(m,n+1),\vec{d}_l(m+1,n),\vec{d}_l(m+1,n+1)\ ] . \tag{4.17}$$

In our case, the median is applied independently on the two components of the displacement vectors

$$d_{x,l+1}(m+\tfrac{1}{2},n+\tfrac{1}{2}) = \mathrm{MED}[\, d_{x,l}(m,n), d_{x,l}(m,n+1), \\ d_{x,l}(m+1,n), d_{x,l}(m+1,n+1)] \;, \tag{4.18}$$

$$d_{y,l+1}(m+\tfrac{1}{2},n+\tfrac{1}{2}) = \mathrm{MED}[\, d_{y,l}(m,n), d_{y,l}(m,n+1), \\ d_{y,l}(m+1,n), d_{y,l}(m+1,n+1)] \;. \tag{4.19}$$

The median is not uniquely defined for an even number of samples. In our case, if $\{x_1, x_2, x_3, x_4\}$ is a set of ordered samples, the median is defined as the mean of the second and third ordered samples

$$\mathrm{MED}[x_1, x_2, x_3, x_4] = \frac{x_2 + x_3}{2} \;. \tag{4.20}$$

Figure 4.6 illustrates the up-conversion operation and compares median and mean values. In this example, the median assigns an effective motion vector to the coarse block, whereas the mean gives an irrelevant estimate. The superiority of the median over the mean value is clearly shown in this illustration. In Sec. 4.4, simulation results comparing both up-conversion operators leads to the same conclusion.

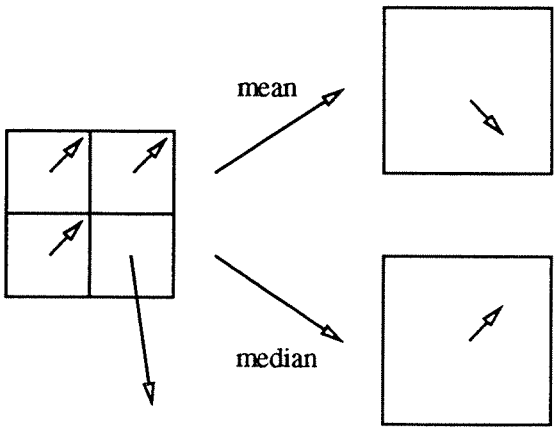

Figure 4.6: Up-conversion: mean compared to median.

4.3.5 Down-conversion

The down-conversion considers the inverse procedure, namely four new motion vectors are generated on Ω_l from the corresponding unique value on the coarser grid Ω_{l+1}. With the notation of Fig. 4.5, the four displacement vectors $\vec{d}_l(m,n)$, $\vec{d}_l(m,n+1)$, $\vec{d}_l(m+1,n)$ and $\vec{d}_l(m+1,n+1)$ have to be estimated from $\vec{d}_{l+1}(m+\frac{1}{2},n+\frac{1}{2})$. The down-conversion

can also take into account estimates issued from neighboring blocks on Ω_{l+1}.

The simplest method duplicates four times the motion vector of the coarser level, as performed in [88].

$$\vec{d}_l(m,n) = \vec{d}_{l+1}(m+\frac{1}{2}, n+\frac{1}{2}) \ , \tag{4.21}$$

$$\vec{d}_l(m,n+1) = \vec{d}_{l+1}(m+\frac{1}{2}, n+\frac{1}{2}) \ , \tag{4.22}$$

$$\vec{d}_l(m+1,n) = \vec{d}_{l+1}(m+\frac{1}{2}, n+\frac{1}{2}) \ , \tag{4.23}$$

$$\vec{d}_l(m+1,n+1) = \vec{d}_{l+1}(m+\frac{1}{2}, n+\frac{1}{2}) \ . \tag{4.24}$$

A bilinear interpolation is preferred in [89, 15]. Applied in the multigrid structure, it leads to the following down-conversion

$$\begin{aligned} \vec{d}_l(m,n) &= \tfrac{9}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}) + \tfrac{3}{16}\,\vec{d}_{l+1}(m-\tfrac{3}{2}, n+\tfrac{1}{2}) \\ &+ \tfrac{3}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n-\tfrac{3}{2}) + \tfrac{1}{16}\,\vec{d}_{l+1}(m-\tfrac{3}{2}, n-\tfrac{3}{2}) \ , \end{aligned} \tag{4.25}$$

$$\begin{aligned} \vec{d}_l(m,n+1) &= \tfrac{9}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}) + \tfrac{3}{16}\,\vec{d}_{l+1}(m-\tfrac{3}{2}, n+\tfrac{1}{2}) \\ &+ \tfrac{3}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{5}{2}) + \tfrac{1}{16}\,\vec{d}_{l+1}(m-\tfrac{3}{2}, n+\tfrac{5}{2}) \ , \end{aligned} \tag{4.26}$$

$$\begin{aligned} \vec{d}_l(m+1,n) &= \tfrac{9}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}) + \tfrac{3}{16}\,\vec{d}_{l+1}(m+\tfrac{5}{2}, n+\tfrac{1}{2}) \\ &+ \tfrac{3}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n-\tfrac{3}{2}) + \tfrac{1}{16}\,\vec{d}_{l+1}(m+\tfrac{5}{2}, n-\tfrac{3}{2}) \ , \end{aligned} \tag{4.27}$$

$$\begin{aligned} \vec{d}_l(m+1,n+1) &= \tfrac{9}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}) + \tfrac{3}{16}\,\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{5}{2}) \\ &+ \tfrac{3}{16}\,\vec{d}_{l+1}(m+\tfrac{5}{2}, n+\tfrac{1}{2}) + \tfrac{1}{16}\,\vec{d}_{l+1}(m+\tfrac{5}{2}, n+\tfrac{5}{2}) \ . \end{aligned} \tag{4.28}$$

A more efficient operator consists for each of the four subblocks to select the best initial condition among the motion vectors estimated in a neighborhood on the coarse level. This idea is similar to the overlapped pyramid projection proposed in [14]. The selection is based on the matching criterion, and the neighborhood is defined as the four blocks the closest to the considered subblock. In the following, this down-conversion operator is referred to as the best initial condition in a neighborhood.

More precisely, the down-conversion by selecting the best initial condition in a neighborhood is defined as follows. For the subblock $B_l(m,n)$, the set $\mathcal{N}_{m,n}$ of the displacement vectors obtained in a neighborhood of the subblock is defined as

$$\begin{aligned} \mathcal{N}_{m,n} = \{ &\vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}), \vec{d}_{l+1}(m-\tfrac{3}{2}, n+\tfrac{1}{2}), \\ &\vec{d}_{l+1}(m+\tfrac{1}{2}, n-\tfrac{3}{2}), \vec{d}_{l+1}(m-\tfrac{3}{2}, n-\tfrac{3}{2})\} \ . \end{aligned} \tag{4.29}$$

Similarly, for the subblocks $B_l(m, n+1)$, $B_l(m+1, n)$ and $B_l(m+1, n+1)$, the set is defined respectively as

$$\begin{aligned}\mathcal{N}_{m,n+1} = \{ & \vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}), \vec{d}_{l+1}(m-\tfrac{3}{2}, n+\tfrac{1}{2}), \\ & \vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{5}{2}), \vec{d}_{l+1}(m-\tfrac{3}{2}, n+\tfrac{5}{2})\} ,\end{aligned} \tag{4.30}$$

$$\begin{aligned}\mathcal{N}_{m+1,n} = \{ & \vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}), \vec{d}_{l+1}(m+\tfrac{5}{2}, n+\tfrac{1}{2}), \\ & \vec{d}_{l+1}(m+\tfrac{1}{2}, n-\tfrac{3}{2}), \vec{d}_{l+1}(m+\tfrac{5}{2}, n-\tfrac{3}{2})\} ,\end{aligned} \tag{4.31}$$

$$\begin{aligned}\mathcal{N}_{m+1,n+1} = \{ & \vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{1}{2}), \vec{d}_{l+1}(m+\tfrac{1}{2}, n+\tfrac{5}{2}), \\ & \vec{d}_{l+1}(m+\tfrac{5}{2}, n+\tfrac{1}{2}), \vec{d}_{l+1}(m+\tfrac{5}{2}, n+\tfrac{5}{2})\} .\end{aligned} \tag{4.32}$$

Finally, the initial estimate of each subblock is selected in the following way

$$\vec{d}_l(m, n) = \arg \min_{\vec{d} \in \mathcal{N}_{m,n}} f_l(\vec{d}) , \tag{4.33}$$

$$\vec{d}_l(m, n+1) = \arg \min_{\vec{d} \in \mathcal{N}_{m,n+1}} f_l(\vec{d}) , \tag{4.34}$$

$$\vec{d}_l(m+1, n) = \arg \min_{\vec{d} \in \mathcal{N}_{m+1,n}} f_l(\vec{d}) , \tag{4.35}$$

$$\vec{d}_l(m+1, n+1) = \arg \min_{\vec{d} \in \mathcal{N}_{m+1,n+1}} f_l(\vec{d}) . \tag{4.36}$$

Figure 4.7 illustrates the down-conversion and compares the duplication, the bilinear interpolation and the selection of the best initial condition in a neighborhood. In this example, the central block, which contains two different objects (indicated with stripes) moving in different directions (indicated with arrows), is shown after down-conversion. In this case, the down-conversion by duplication assigns an irrelevant estimate to the upper-right subblocks. The result is the propagation into fine levels of block artifacts due to the use of large block sizes on the coarser levels. The bilinear interpolation leads to more effective estimates, nevertheless it does not overcome the above drawback. By exploiting the spatial consistency of the motion field, the method selecting the best initial condition in a neighborhood is able to assign to the upper-right subblock a motion vector issued from a neighboring block, overcoming the above problem. Therefore, this method prevents the propagation of block artifacts. Furthermore, this down-conversion operator eliminates the propagation of wrong estimates, due to local minima in the matching criterion, by recovering from the neighboring values. Finally, the computational overload introduced is low.

The example presented in Fig. 4.7 shows the advantage of the down-conversion by selecting the best initial condition in a neighborhood compared to the duplication and the bilinear interpolation. Simulation results in Sec. 4.4 confirm this observation.

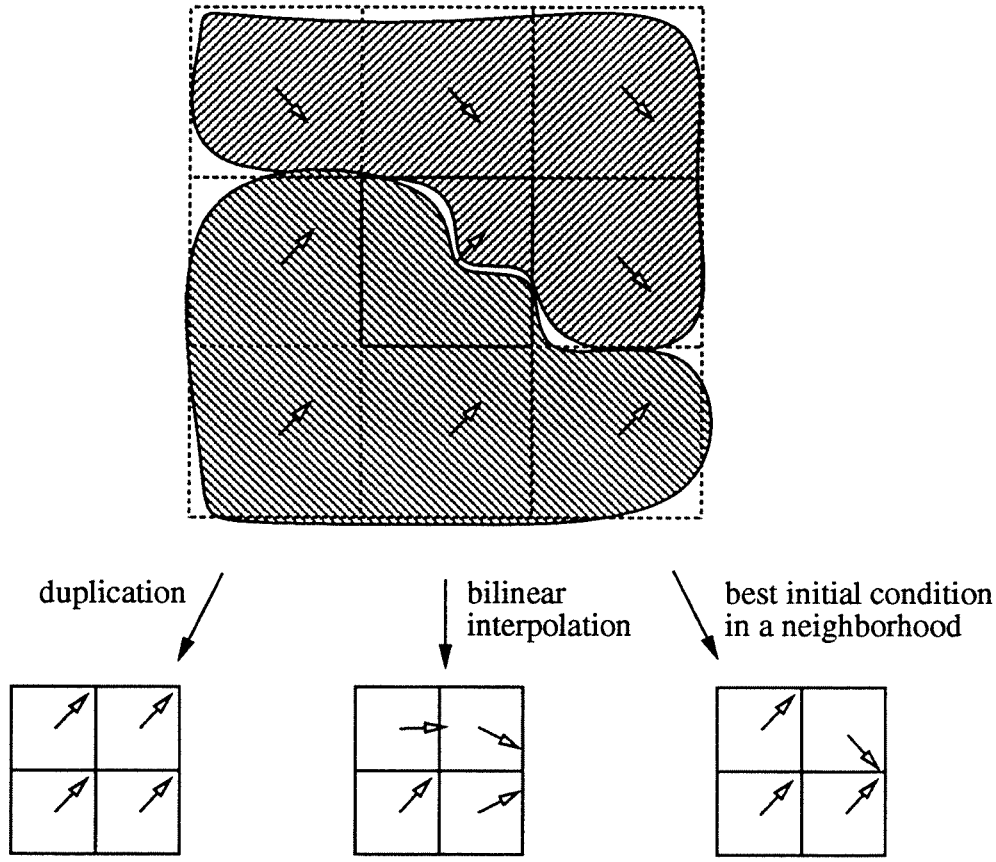

Figure 4.7: Down-conversion: comparison for the central block of duplication, bilinear interpolation and best initial condition in a neighborhood.

4.3.6 Complexity

The computational complexity of the multigrid block matching is much lower than the one of the monoresolution full-search technique.

With the multigrid structure described by Table 4.1, if the image size is $M \times N$ pixels, the number of match positions to be evaluated is given as follows.

The number of positions on the grid Ω_0 is

$$p_0 = \frac{M}{8} \times \frac{N}{8} \times (9+8) \ , \tag{4.37}$$

on Ω_1

$$p_1 = \frac{M}{16} \times \frac{N}{16} \times (9 + 8 + 8) \ , \tag{4.38}$$

and on Ω_2

$$p_2 = \frac{M}{32} \times \frac{N}{32} \times (9 + 8 + 8 + 8) \ . \tag{4.39}$$

With the coarse-to-fine control strategy, the total number of match positions is given by

$$p_{cf} = p_2 + p_1 + p_0 \ , \tag{4.40}$$

whereas with the fine-to-coarse-to-fine one, it is

$$p_{fcf} = p_2 + 3 \times p_1 + 3 \times p_0 \ . \tag{4.41}$$

For an image in the format CCIR-601 (see Table 3.2), i.e. of size 576×704 pixels, the total number of match positions for the two control strategies is respectively

$$p_{cf} = 160\ 380 \quad \text{and} \quad p_{fcf} = 455\ 004 \ . \tag{4.42}$$

In comparison, for the monoresolution full-search block matching with a block size of 8×8 pixels and a maximum displacement of ± 25 pixels (i.e. corresponding to the coarse-to-fine control), the number of positions to be evaluated is

$$p_{\pm 25} = \frac{M}{8} \times \frac{N}{8} \times (2 \times 25 + 1)^2 = 16\ 479\ 936 \ , \tag{4.43}$$

and for a maximum displacement of ± 45 pixels (i.e. equivalent to the fine-to-coarse-to-fine control), it is

$$p_{\pm 45} = \frac{M}{8} \times \frac{N}{8} \times (2 \times 45 + 1)^2 = 52\ 468\ 416 \ . \tag{4.44}$$

Consequently, the multigrid block matching algorithm decreases by approximately a factor 100 the number of match positions to be evaluated, when compared to the monoresolution full-search technique, for an identical maximum displacement. This result is extremely important for software simulations. However, it cannot be deduced straightforwardly that the hardware implementation will be more efficient. Nevertheless, as already mentioned, a study of an earlier and simpler version of the proposed multigrid algorithm has been carried out in [136, 137, 138] and has shown the feasibility of an efficient hardware implementation.

4.4 Simulation results

In this section, first results in terms of DFD energy, motion vectors entropy and CPU time are presented. These results motivate the choices made concerning the matching criterion, the control strategy and the up- and down-conversion operators. Furthermore, it gives a first insight on the performance of the multigrid approach when compared to the monoresolution full-search block matching algorithm

Next, experiments investigate the optimal precision of the motion vectors accuracy. The comparison is based on the DFD energy, the motion vectors entropy as well as bit rate versus PSNR.

To completely assess the performance of a motion estimation algorithm, simulations within a coding scheme have to be carried out, for the reasons exposed in Chap. 3. Therefore, multigrid and full-search block matching motion estimation performances are finally compared within video coding schemes.

Video coding simulations are carried out with the three schemes described in Chap. 3:

- Scheme $\mathcal{A}$: motion compensated wavelet transform-based (see Sec. 3.3.1),
- Scheme $\mathcal{B}$: motion compensated interframe DPCM (see Sec. 3.3.2),
- Scheme $\mathcal{C}$: motion compensated segmentation-based (see Sec. 3.3.3).

All simulations have been carried out on the sequences "Mobile Calendar", "Table Tennis" and "Flower Garden" (see Sec. 3.2.3) in the format CCIR-601 progressive (see Table 3.2). Results in terms of CPU time refer to a Cray-2 supercomputer.

4.4.1 Comparison between Mean Absolute Error and Mean Square Error for the matching criterion

In this simulation, the MAE and MSE measures for the matching criterion are compared. Figure 4.8 shows the DFD energy for the multigrid algorithm (coarse-to-fine control and down-conversion by selecting the best initial condition in a neighborhood) using both MAE and MSE. Motion vectors have one pixel accuracy. It appears clearly that both measures gives identical performances. Readers may be surprised that the MAE gives sometimes even lower DFD energy than the MSE. This is due to multigrid structure and the fact that different results are obtained on the coarse levels, therefore it is possible for the MAE measure to reach sometimes lower DFD energy than the MSE measure.

As a conclusion of the above simulation results, the MAE measure is chosen for the remaining of this dissertation, due to its lower computational complexity.

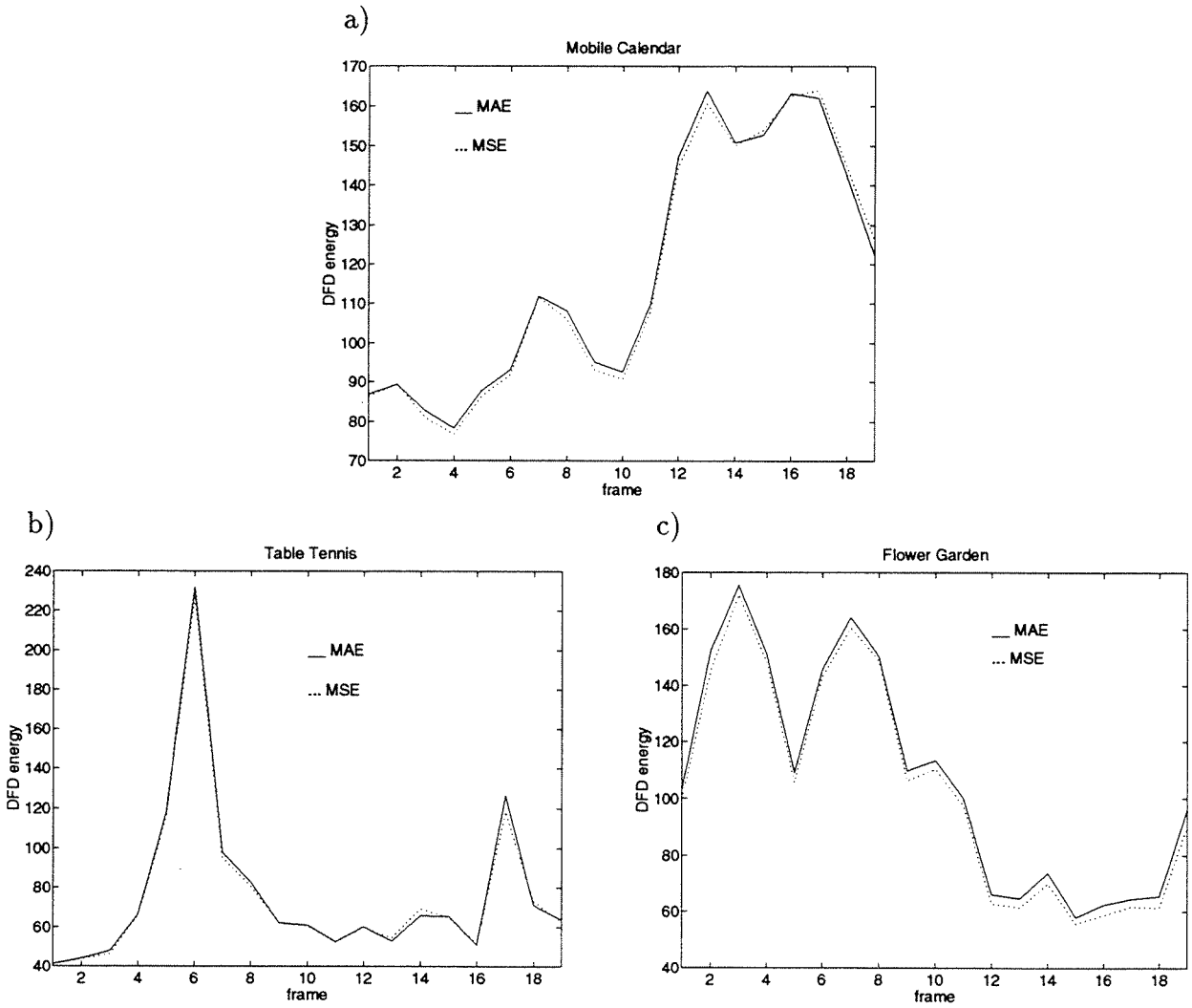

Figure 4.8: DFD energy: comparison between MAE and MSE for the matching criterion, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

4.4.2 Comparison between coarse-to-fine and fine-to-coarse-to-fine control strategies

Both coarse-to-fine and fine-to-coarse-to-fine control strategies have been described in Sec. 4.3.3. This simulation compares the performances of these latter. The results below have been obtained with motion vectors of one pixel accuracy.

Figure 4.9 shows the DFD energy for the multigrid algorithm when using each of these controls. The simulation have been carried out using the simplest up- and down-conversion operators, namely up-conversion by mean and down-conversion by duplication. For the two sequences "Table Tennis" and "Flower Garden", the fine-to-coarse-to-fine control brings only small improvements when compared to the coarse-to-fine control. However, for the sequence "Mobile Calendar" the improvement is significant. This is due to the characteristic matching functions obtained with this sequence, which contain many local minima. In this specific case, the coarse-to-fine control algorithm is trapped in a local minimum, whereas the fine-to-coarse-to-fine control reaches a minimum very close to the global one.

As a conclusion of the simulation results presented in this section, the fine-to-coarse-to-fine control strategy outperforms the simpler coarse-to-fine one, when the up- and down-conversion are performed by mean and duplication respectively.

4.4.3 Comparison between up- and down-conversion operators

Up-conversion by mean and median, as well as down-conversion by duplication, bilinear interpolation and selection of the best initial condition in a neighborhood have been discussed in Sec. 4.3.4 and 4.3.5. Simulation results in terms of the DFD energy, with motion vectors of one pixel accuracy, are shown in this section.

Figure 4.10 compares the down-conversion operators in the coarse-to-fine multigrid block matching algorithm. For the three sequences, the down-conversion by selecting the best initial condition is clearly superior to the classical methods, namely duplication or bilinear interpolation. The gain is obtained thanks to the capability of this efficient down-conversion operator to avoid both the propagation of block artifacts between levels, as well as the propagation of wrong estimates due to local minima of the matching function. It is to note that the down-conversion by bilinear interpolation performs very closely to the duplication method, and consequently does not lead to significant improvements compared to the latter.

Figure 4.11 compares up- and down-conversion operators in the fine-to-coarse-to-fine multigrid technique. The higher performance is achieved by the combination of median filte-

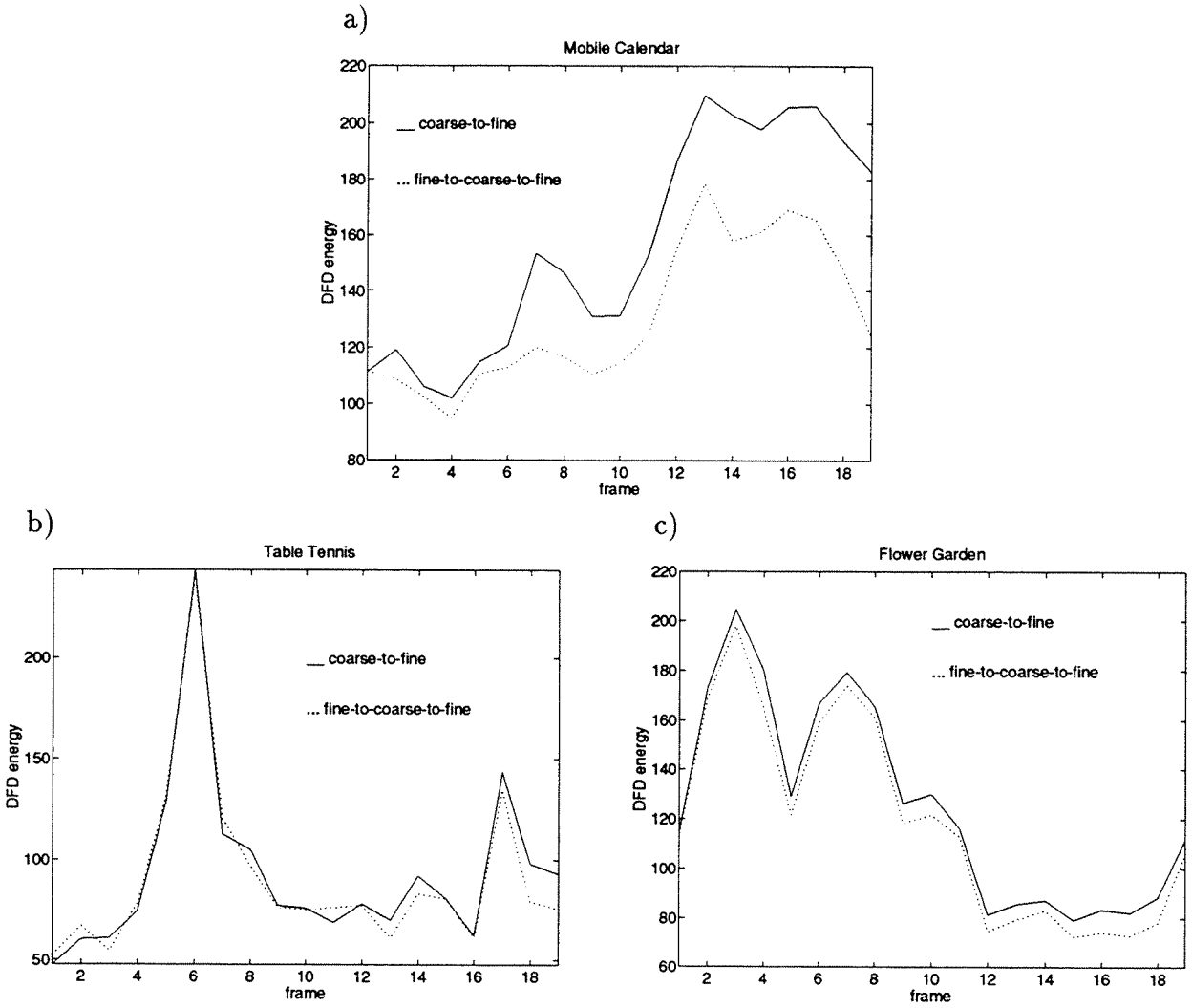

Figure 4.9: DFD energy: comparison between coarse-to-fine and fine-to-coarse-to-fine control strategies, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

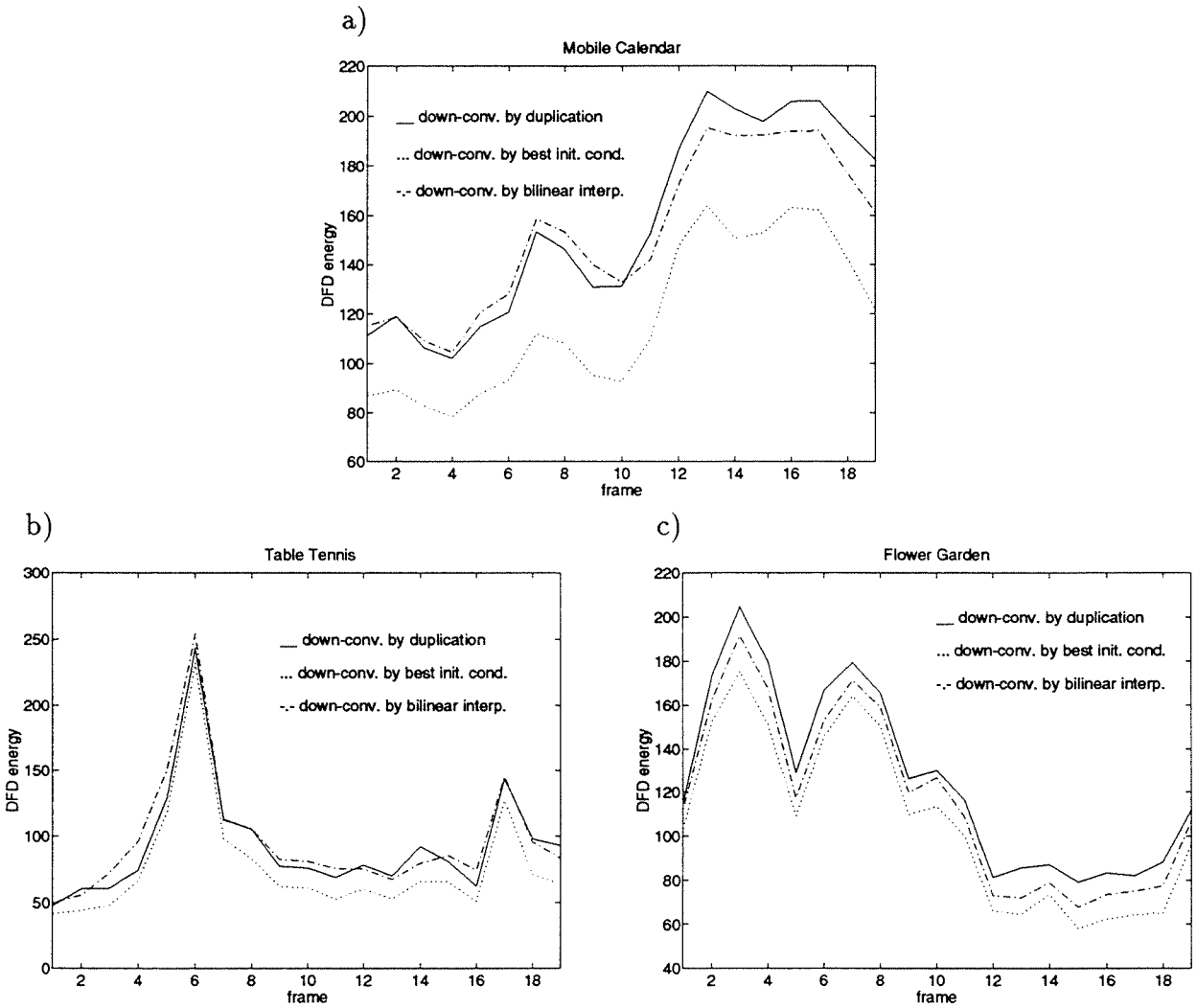

Figure 4.10: DFD energy: comparison in the coarse-to-fine multigrid algorithm between down-conversion by duplication, bilinear interpolation and selection of the best initial condition in a neighborhood, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

ring for up-conversion and selection of the best initial condition for down-conversion. It outperforms both the up-conversion by mean and the down-conversion by duplication or bilinear interpolation, and confirms the arguments given in Sec. 4.3.4 and 4.3.5.

As a consequence of the results presented in this section, the up-conversion by median and down-conversion by selecting the best initial condition in a neighborhood are adopted for the remaining of this dissertation.

4.4.4 Comparison between the multigrid and the full-search block matching motion estimation techniques in terms of DFD energy, motion vectors entropy and CPU time

In this section, the multigrid block matching motion estimation is compared with the full-search technique, in terms of DFD energy, motion vectors entropy and CPU time.

The simulation shows the DFD energy and the motion vectors entropy for the full-search, the coarse-to-fine multigrid and the fine-to-coarse-to-fine multigrid techniques. It should be clear that up-conversion and down-conversion are always performed by median and selection of the best initial condition in a neighborhood respectively. Motion vectors are limited to one pixel accuracy. The full-search simulation is carried out with a block size of 8×8 pixels and a maximum displacement of ± 25 pixels (corresponding to the coarse-to-fine multigrid algorithm).

Figure 4.12 compares the three motion estimation algorithms in terms of DFD energy. It appears clearly that both multigrid techniques are quasi-optimal in terms of minimizing the DFD energy, when compared to the full-search algorithm. It may be surprising to readers that the multigrid techniques reach sometimes even lower DFD energy that the full-search which is optimal in this sense. This is explained by the fact that the three algorithms are minimizing the MAE measure, and not the MSE, whereas the DFD energy is compared. In addition, as the fine-to-coarse-to-fine control leads to a maximum displacement of ± 45 pixels, which is larger than the maximum displacement of the full-search technique, it may lead to lower DFD energy in the presence of large motion.

Another important result is that both the coarse-to-fine and the fine-to-coarse-to-fine controls have now similar performances, even though we showed in Sec. 4.4.2 the higher performance of the latter algorithm. It is explained as follows. In Sec. 4.4.2, the improvement of the fine-coarse-to-fine control, using up-conversion by mean and down-conversion by duplication, was due to its ability to avoid to be trapped in local minima. However, the down-conversion by selecting the best initial condition in a neighborhood has the same capability. Therefore the coarse-to-fine control combined with this more efficient down-conversion operator reaches identical performances compared to the more complex

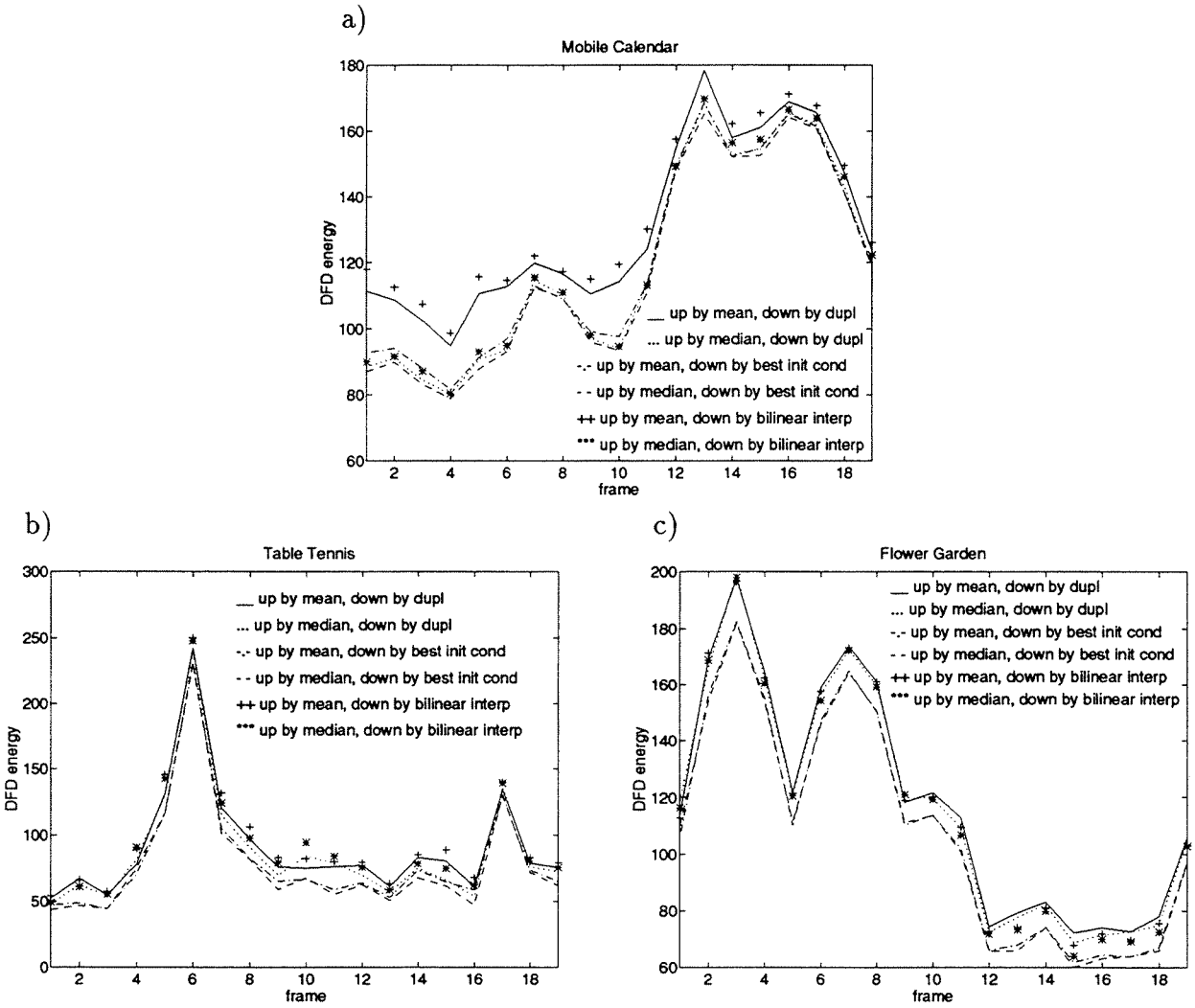

Figure 4.11: DFD energy: comparison in the fine-to-coarse-to-fine multigrid algorithm between up-conversion by mean and median, and down-conversion by duplication, bilinear interpolation and selection of the best initial condition in a neighborhood, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

fine-to-coarse-to-fine control, making the latter useless in this context.

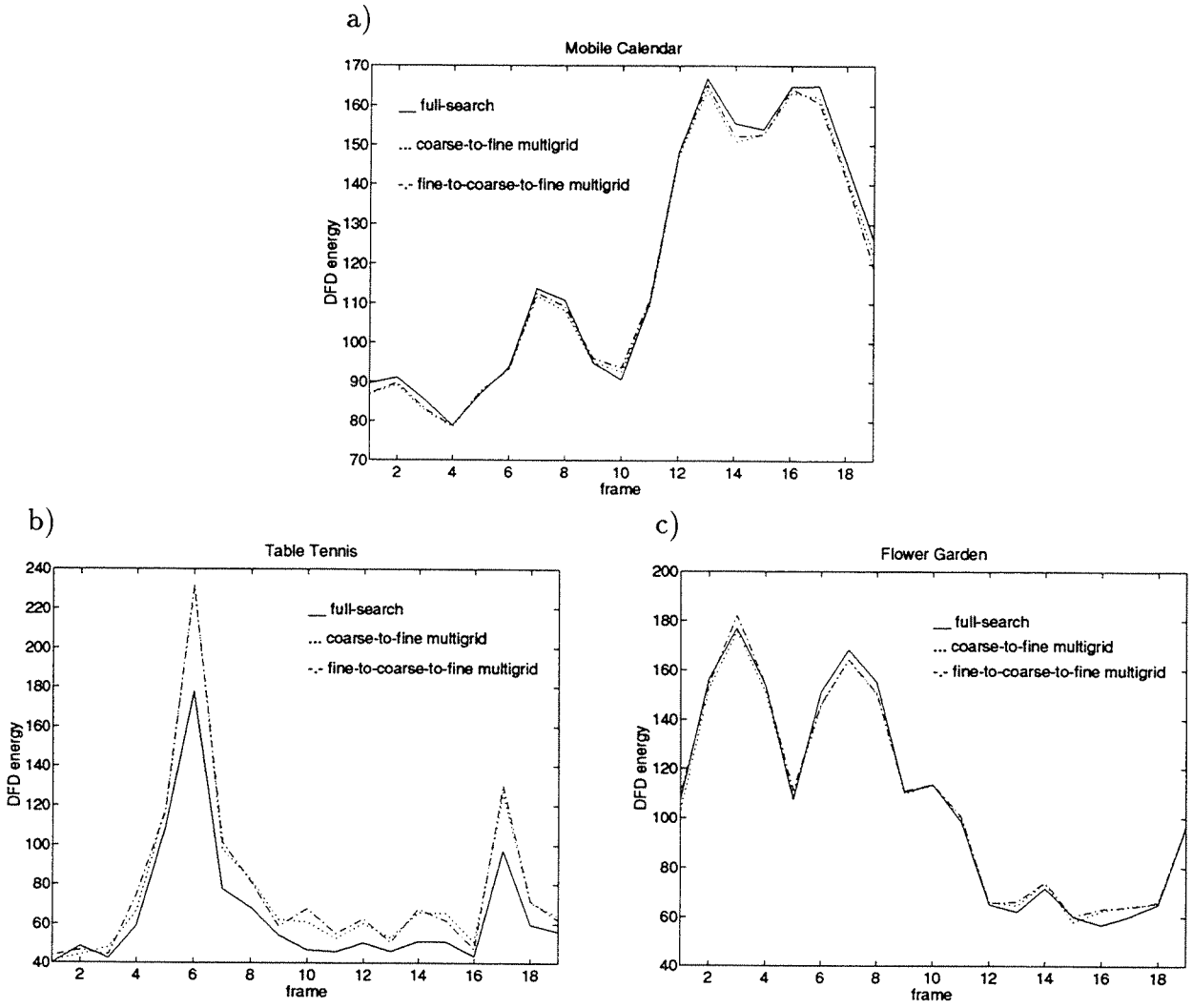

Figure 4.12: DFD energy: comparison between full-search, coarse-to-fine multigrid and fine-to-coarse-to-fine multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

Figure 4.13 shows the motion vectors entropy for the three same motion estimation algorithms. The multigrid approaches lead to lower motion vectors entropy, in other words lower overhead information, compared to the full-search technique. The gain is approximately 1 bit per motion vector, and represents between 20% to 30% saving for the sequences "Mobile Calendar" and "Flower Garden" and 10% to 20% for the sequence "Table Tennis". In [48], motion fields resulting from multigrid and full-search block matching techniques have been coded with a composite source model, showing a gain ranging from

1 to 1.5 bit per motion vector in favor of the multigrid algorithm, confirming consequently the above results. The gain is obtained thanks to the capability of the multigrid techniques to provide smooth and robust motion fields, close to the true motion in the scene.

Figure 4.14 illustrates the last remark. It shows the motion field obtained for the sequence "Mobile Calendar", using either the full-search technique or the coarse-to-fine multigrid algorithm. One can see that the motion vectors resulting from the full-search technique are noisy and differ significantly from the true motion in the scene, whereas the motion vectors obtained with the multigrid algorithm are smooth and close to the true motion.

Another important feature of a motion estimation algorithm is its computational complexity. The number of match positions required by the full-search and the multigrid algorithms have been evaluated in Sec. 4.3.6. The CPU time per frame required by the three above motion estimation algorithms (on a Cray-2 supercomputer) is given in Table 4.2. Clearly, the multigrid techniques represent a large saving in terms of computation time.

algorithm	max. displ.	CPU [sec.]
full-search	± 25	863
coarse-to-fine multigrid	± 25	14
fine-to-coarse-to-fine multigrid	± 45	32

Table 4.2: CPU time per frame required by the full-search, the coarse-to-fine multigrid and the fine-to-coarse-to-fine multigrid algorithms.

From this simulation, the following conclusions can be drawn. The multigrid algorithms, when compared to the classical full-search technique, are quasi-optimal in terms of minimizing the DFD energy, lead to lower overhead information due to smoother motion fields, and require a much lower computational complexity. Furthermore, the coarse-to-fine multigrid algorithm leads to similar performances than the fine-to-coarse-to-fine one, for a lower complexity. Therefore, the former is preferred in the remaining of this dissertation, and is simply referred to as the multigrid algorithm in the following.

4.4.5 Comparison between the multigrid block matching motion estimation and the fast search techniques

Due to the difficulty to directly compare the multigrid algorithm with the fast search techniques described in Sec. 2.3.6, the latter are compared with the full-search technique.

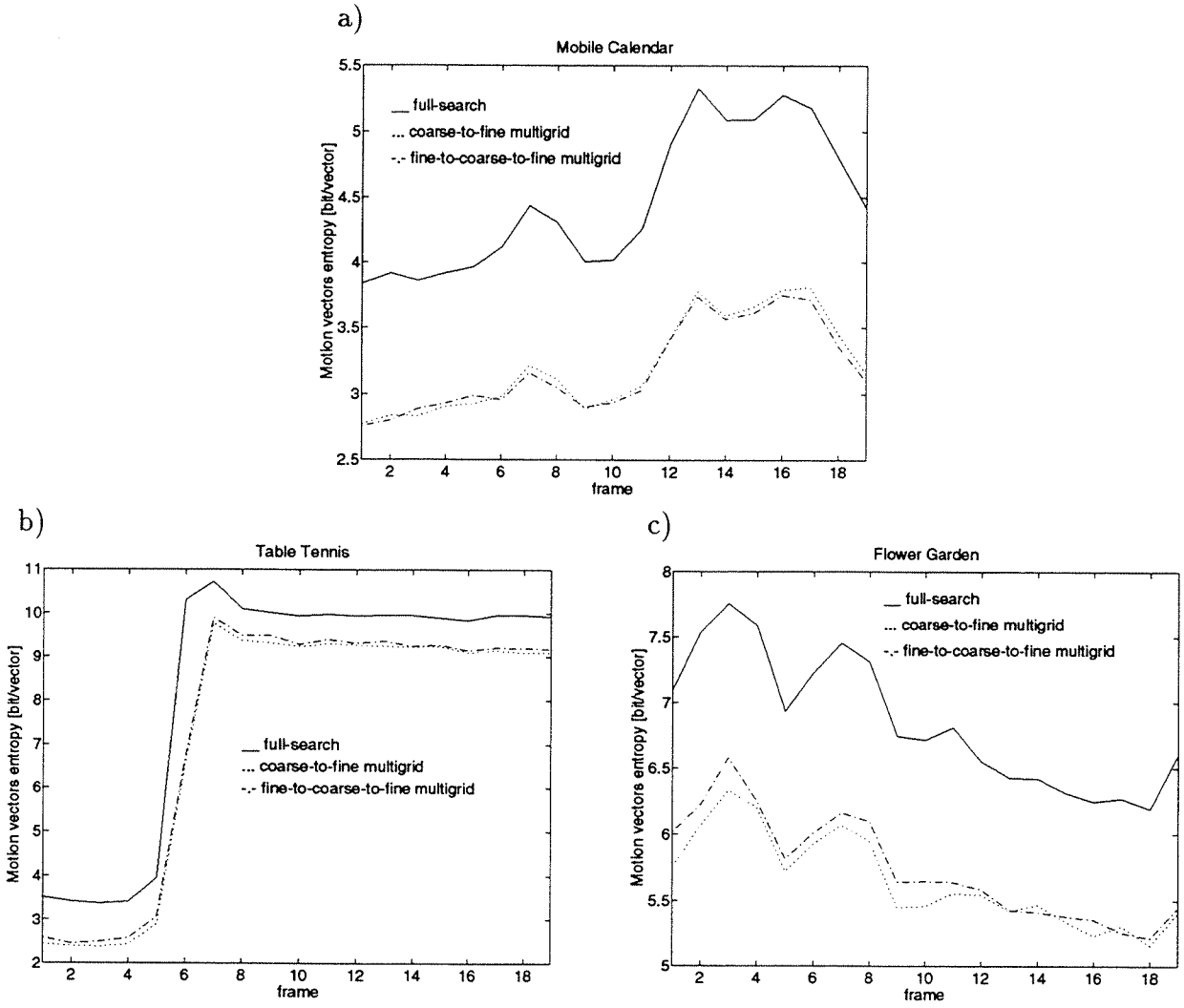

Figure 4.13: Motion vectors entropy: comparison between full-search, coarse-to-fine multigrid and fine-to-coarse-to-fine multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

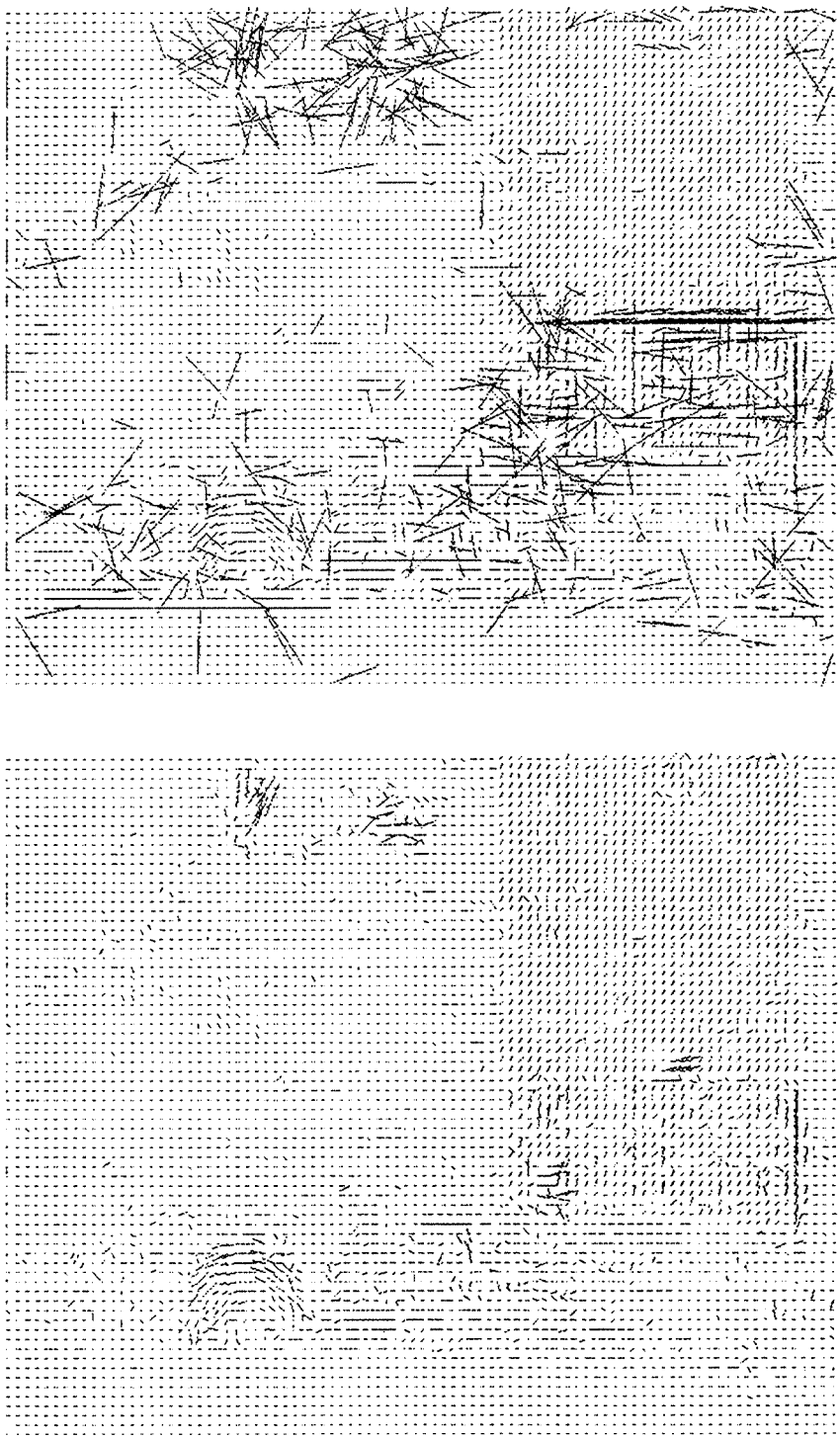

Figure 4.14: Motion field needle diagram for "Mobile Calendar": top) full-search, bottom) coarse-to-fine multigrid.

Thus, the performances of both the fast search and the multigrid techniques relative to the full-search are compared.

In Sec. 2.3.6, three fast search block matching techniques have been described: the $2D$-logarithmic search [9], the 3-step search [94] and the conjugate direction search [96]. A direct comparison between these fast search techniques and the multigrid algorithm would be unfair, as the former have been designed to estimate only small displacements (for instance ± 6 pixels for the 3-step search), whereas the latter estimates displacements up to ± 25 pixels. Therefore, a comparison between the fast search techniques and the full-search algorithm is preferred in this section. From the relative performances of the fast search techniques compared to full-search on the one side, and the relative performance of the multigrid algorithm compared to full-search on the other side, some conclusions can be drawn.

Figure 4.15 compares the above mentioned three fast search techniques with the full-search method in terms of DFD energy. The simulations are carried out with a block size of 8×8 pixels, a maximum displacement of ± 6 pixels and one pixel accuracy motion vectors. The full-search outperforms significantly the three other methods. The $2D$-logarithmic search and the 3-step search have comparable performances in average, the $2D$-logarithmic search performing better on "Mobile Calendar" and worse on "Table Tennis" and "Flower Garden". Finally, the conjugate direction search performs significantly less efficiently than the other methods.

The number of match positions depends on the input images for both the $2D$-logarithmic search and the conjugate direction search. Therefore, the comparative computational complexity cannot be study as in Sec. 4.3.6. Nevertheless, the CPU time obtained for the four motion estimation algorithms evaluated in Fig. 4.15 is given in Table 4.3.

algorithm	CPU [sec.]
full-search	57.8
3-step	9.7
$2D$-logarithmic	7.5
conjugate direction	5.1

Table 4.3: CPU time per frame required by the full-search, $2D$-Logarithmic search, 3-step search and conjugate direction search.

As a conclusion of this simulation, none of the three fast search techniques, namely the $2D$-logarithmic search, the 3-step search and the conjugate direction search leads to performances comparable to those provided by the full-search in terms of DFD energy. As the

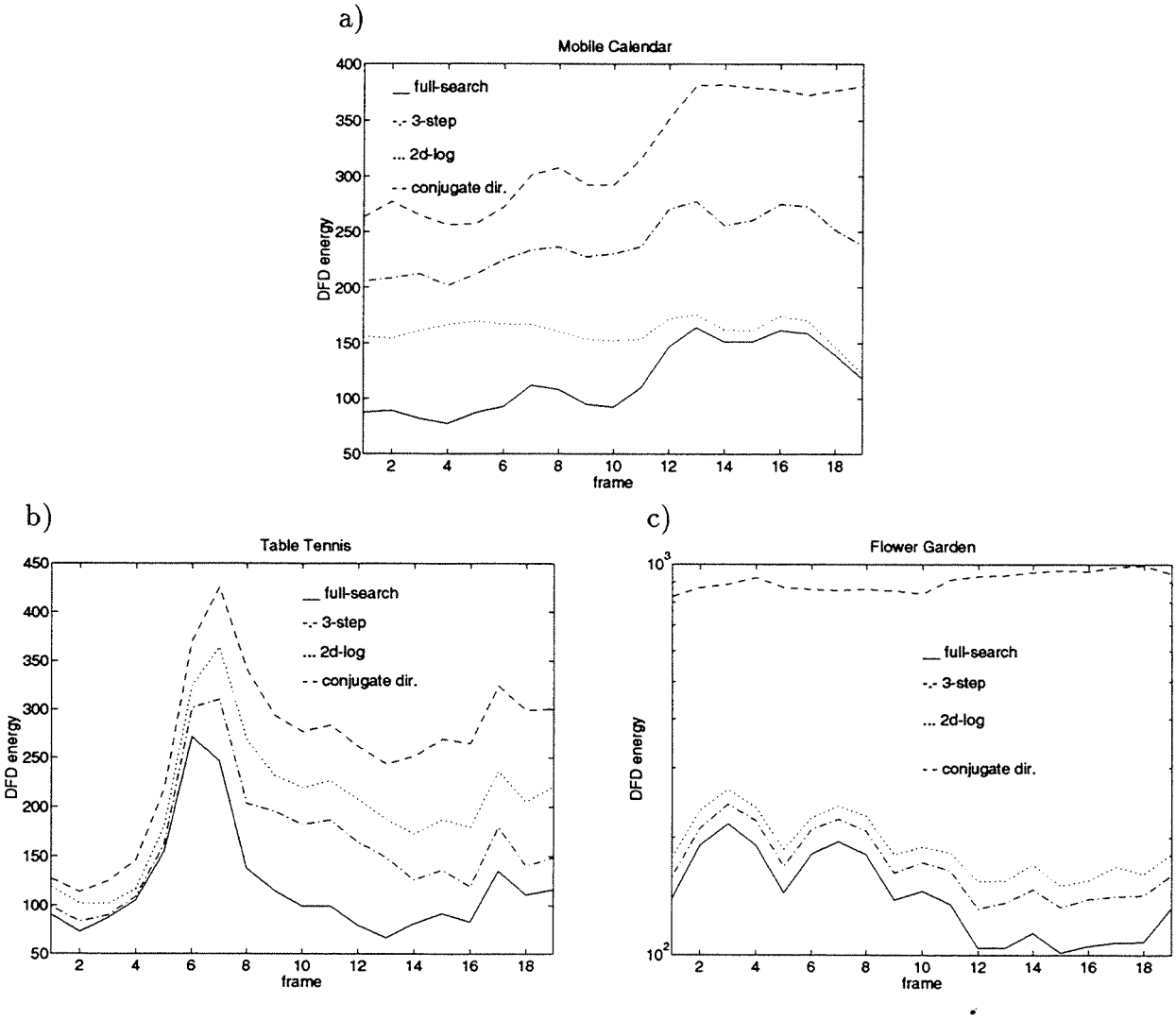

Figure 4.15: DFD energy: comparison between full-search, $2D$-logarithmic search, 3-step search and conjugate direction search, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

multigrid algorithm is quasi-optimal in reaching the minimum DFD energy, its superiority compared to the above classical fast search techniques is therefore shown.

4.4.6 Comparative results on the motion vectors sub-pixel accuracy

A point which has been mentioned without being investigated up to now is the suitable accuracy of the motion vectors. As explained previously, the block matching techniques define displacements of one pixel accuracy. However, a higher precision can be obtained. This stage is usually implemented as a post-processing where the one pixel accuracy displacement vectors are refined to a fractional pixel precision. For this purpose, the image intensity has to be interpolated at fractional pixel locations. The latter operation is commonly performed by bilinear interpolation.

Obviously, the highest the accuracy of the motion vectors, the lowest the DFD energy, the price to pay being a higher rate overhead information. Furthermore, it should be pointed out that the sub-pixel accuracy refinement increases significantly the computational complexity. In this section, in a first stage, results in terms of DFD energy and motion vectors entropy are presented. In a second stage, results in terms of bit rate versus PSNR are given. The simulation is carried out with the multigrid block matching motion estimation.

The optimal accuracy of the motion vectors depends on the image sequence format, the target bit rate and the subsequent coding technique applied on the DFD. In [146], results showing the improvement of half pixel accuracy compared to one pixel accuracy for CCIR 601 test sequences are reported. In [147], a comparison between different sub-pixel accuracies is carried out for HDTV sequences. Based on the bit rate to transmit the frames, experimental results show that a precision higher than 1/4 pixel accuracy is useless. Nevertheless, as the overhead information to transmit motion vectors is not taken into account, these results do not allow yet to draw a conclusion on the optimal accuracy.

Figures 4.16 and 4.17 show respectively the DFD energy and the motion vectors entropy for the multigrid block matching motion estimation technique with an accuracy of 1, 1/2, 1/4, 1/8 and 1/16 pixel accuracy. In Fig. 4.16, it appears clearly that the 1/2 pixel precision arises significantly the performance of the motion estimation compared to 1 pixel accuracy. In comparison, the gain due to 1/4 pixel precision is much less important. Finally, precision higher than 1/4 pixel is completely useless, as the performance saturates, in accordance with the observation made in [147]. In Fig. 4.17, the transmission cost of the motion vectors, estimated by their entropy, increases continuously with the increasing accuracy of the motion vectors. The improvement from $1/2^n$ to $1/2^{n+1}$ $(n = 0, \ldots, 3)$ in the displacements accuracy leads to an increased overhead information ranging from 1 to 2 bit per motion vector. From these results, we conclude that the motion vectors

accuracy should be limited to 1/4 pixel. Again, simulations within a video coding scheme have to be carried out to completely assess the motion vectors accuracy question.

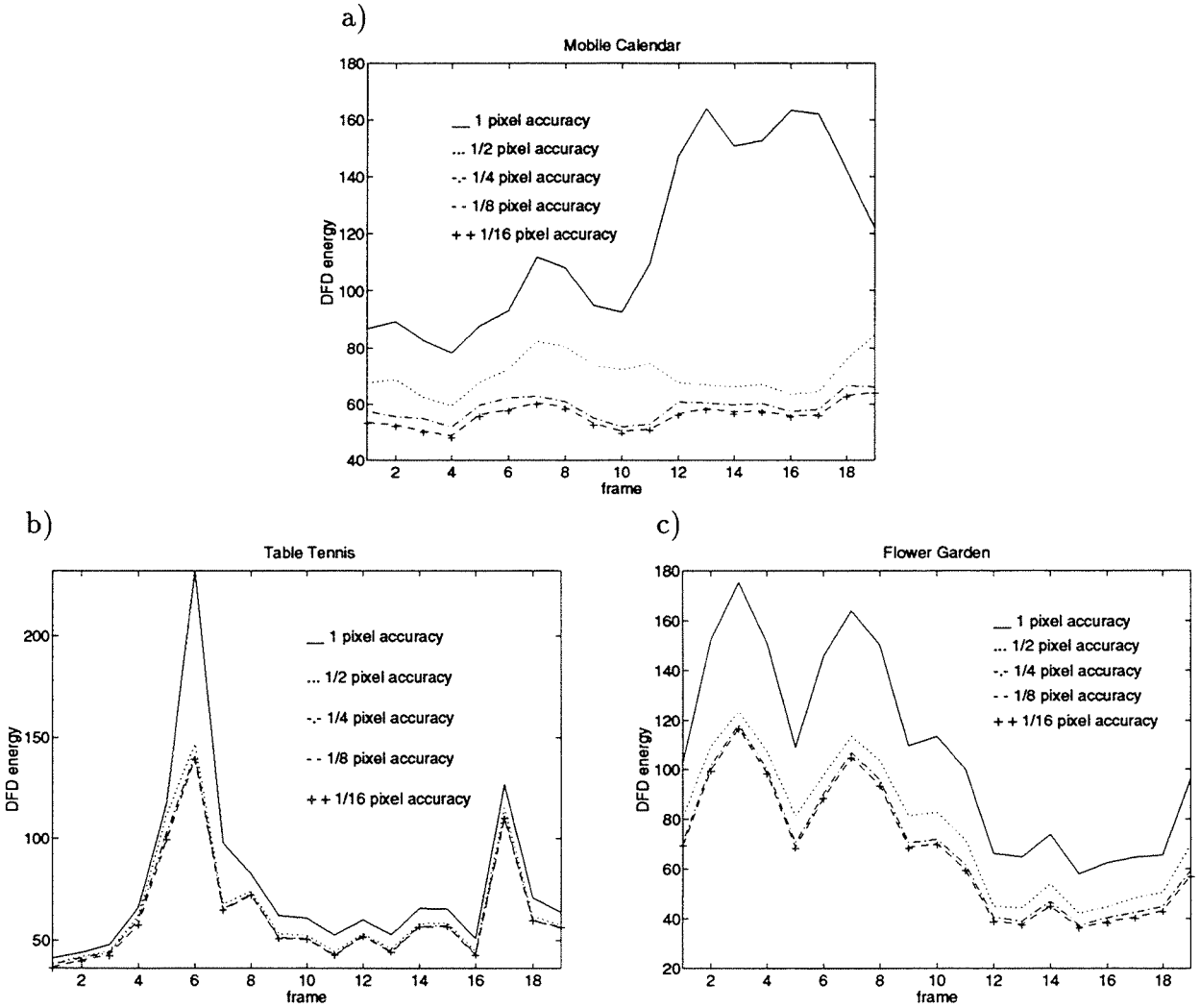

Figure 4.16: DFD energy: multigrid block matching, comparison between 1, 1/2, 1/4, 1/8 and 1/16 pixel accuracy, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

Figures 4.18, 4.19 and 4.20 compare the performances of the 1, 1/2 and 1/4 pixel accuracy multigrid algorithms in the three test video schemes. The schemes $\mathcal{A}$ and $\mathcal{B}$ give similar relative results between the different degrees of sub-pixel accuracy. For the sequence "Table Tennis", the gain due to 1/2 pixel precision ranges from 1.5 to 2.5 dB, or from 2 to 3 Mb/s, whereas the 1/4 pixel precision does not lead to further improvement. For the sequences "Mobile Calendar" and "Flower Garden", the improvement arising from the

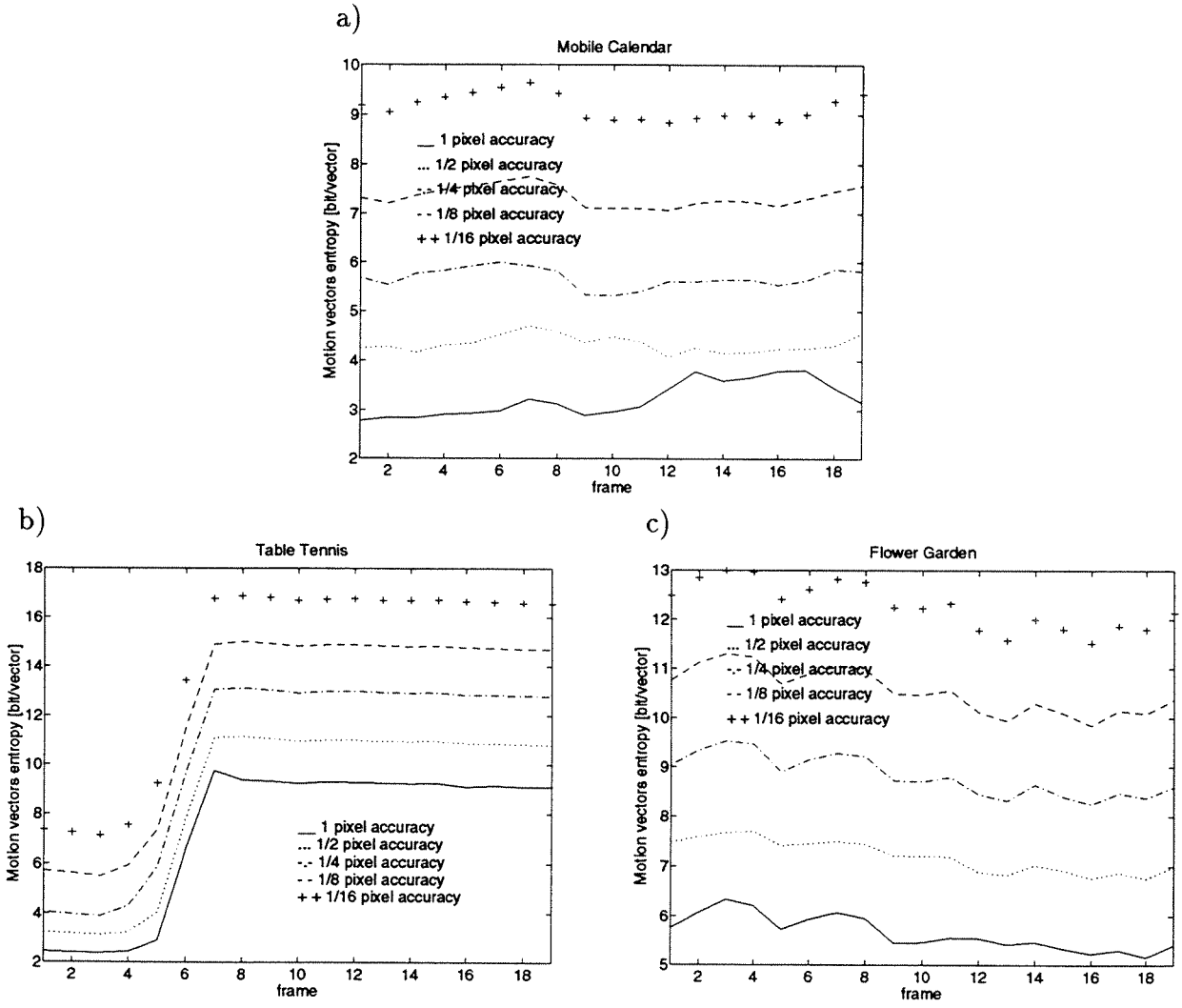

Figure 4.17: Motion vectors entropy: multigrid block matching, comparison between 1, 1/2, 1/4, 1/8 and 1/16 pixel accuracy, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

1/2 pixel accuracy is larger, ranging from 2.5 to 3.5 dB, or 2 to 3 Mb/s. In the range from 34 to 35 dB, the 1/4 pixel accuracy leads to small improvement compared to 1/2 pixel accuracy, of approximately 0.5 dB or 0.5 Mb/s. Nevertheless, this improvement decreases for a lower PSNR, becoming negligible for a PSNR around 30 dB. The coding scheme $\mathcal{C}$ leads to slightly different results. The 1/2 pixel accuracy arises the larger improvement for "Table Tennis", ranging from 2.5 to 2.75 dB or approximately 2.5 Mb/s, whereas the improvement is between 1.5 and 2 dB for the two other sequences, or around 2 Mb/s. With this third scheme, for the three sequences the 1/4 pixel accuracy leads to negligible improvement compared to 1/2 pixel accuracy.

As a conclusion of this simulation comparing different degrees of sub-pixel accuracy for the motion vectors, it appears that the half pixel precision is the best compromise between computational complexity and coding performance. It leads to an improvement ranging from 1.5 to 3.5 dB for the different sequences and the different coding schemes used in the simulation, when compared to one pixel accuracy. The 1/4 pixel accuracy leads only to insignificant further improvements for an increased computational cost.

4.4.7 Comparison between the multigrid and the full-search block matching motion estimation techniques in terms of bit rate and PSNR

Up to now, the comparison between the multigrid and the full-search block matching algorithms has been bounded to DFD energy, motion vectors entropy and CPU time. The following simulation are carried out with the three test video coding schemes.

The multigrid algorithm includes coarse-to-fine control and down-conversion by selecting the best initial condition in a neighborhood. The full-search block matching is carried out with a block size of 8×8 pixels and a maximum displacement of ± 15 pixels, lower than the maximum allowed with the multigrid technique (± 25 pixels), but representing a compromise between computational complexity and performances. In the two motion estimation algorithms, motion vectors have a half pixel accuracy.

Figures 4.21, 4.22 and 4.23 compares the bit rate for an identical reconstructed quality (in terms of PSNR) while using the multigrid and the full-search block matching for the three video coding schemes respectively. Table 4.4 summarizes the performances corresponding to Figs. 4.21, 4.22 and 4.23 in terms of bit rate and PSNR.

The first observation is that both multigrid and full-search block matching motion estimation performs closely to each other in terms of coding. The full-search is slightly better on the sequence "Table Tennis". However, for the sequence "Mobile Calendar" and "Flower Garden", the multigrid algorithm is more effective than the full-search, the gain ranging

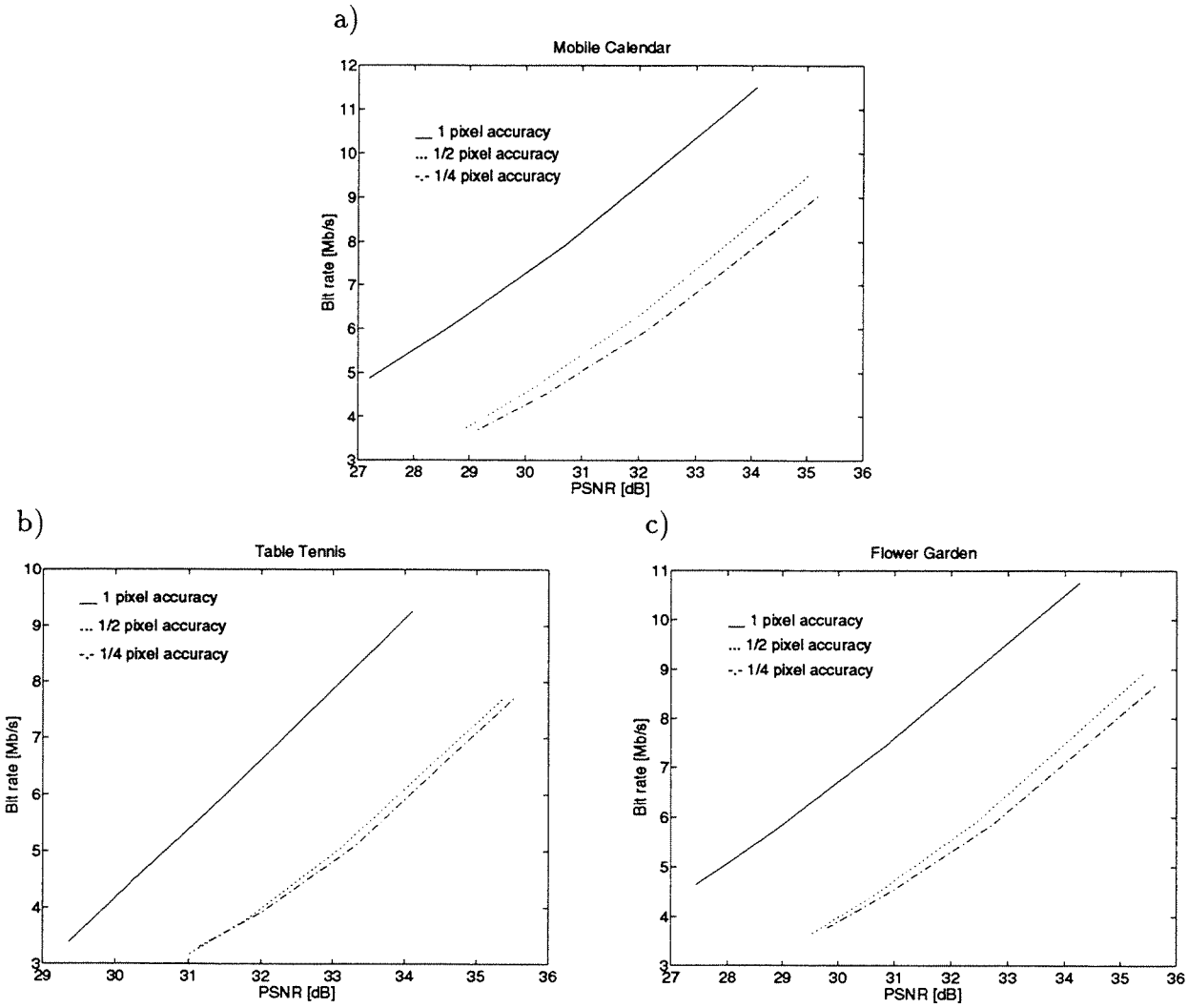

Figure 4.18: Bit rate versus PSNR for the coding scheme $\mathcal{A}$: comparison between 1, 1/2 and 1/4 pixel accuracy for the multigrid algorithm, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

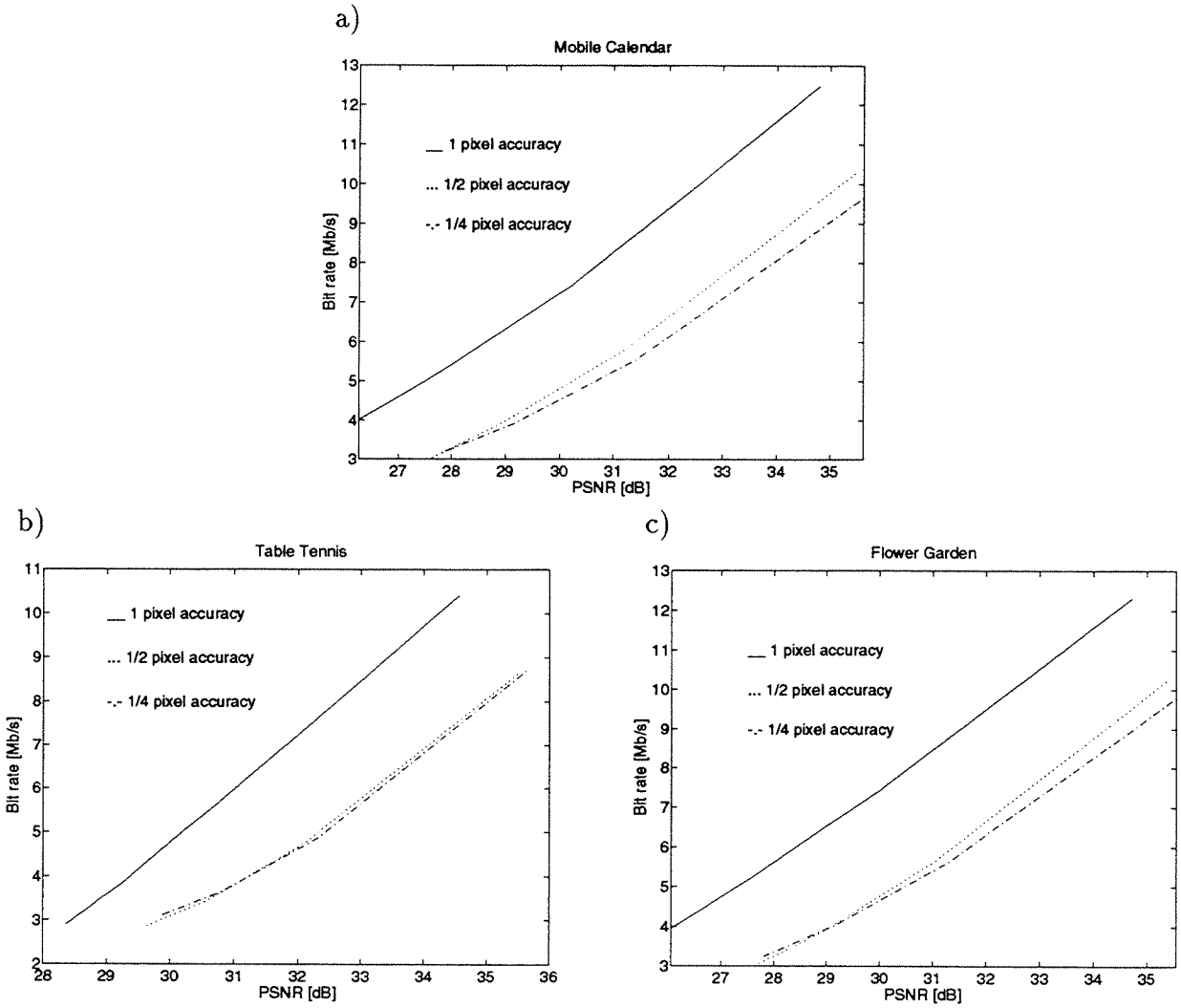

Figure 4.19: Bit rate versus PSNR for the coding scheme $\mathcal{B}$: comparison between 1, 1/2 and 1/4 pixel accuracy for the multigrid algorithm, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

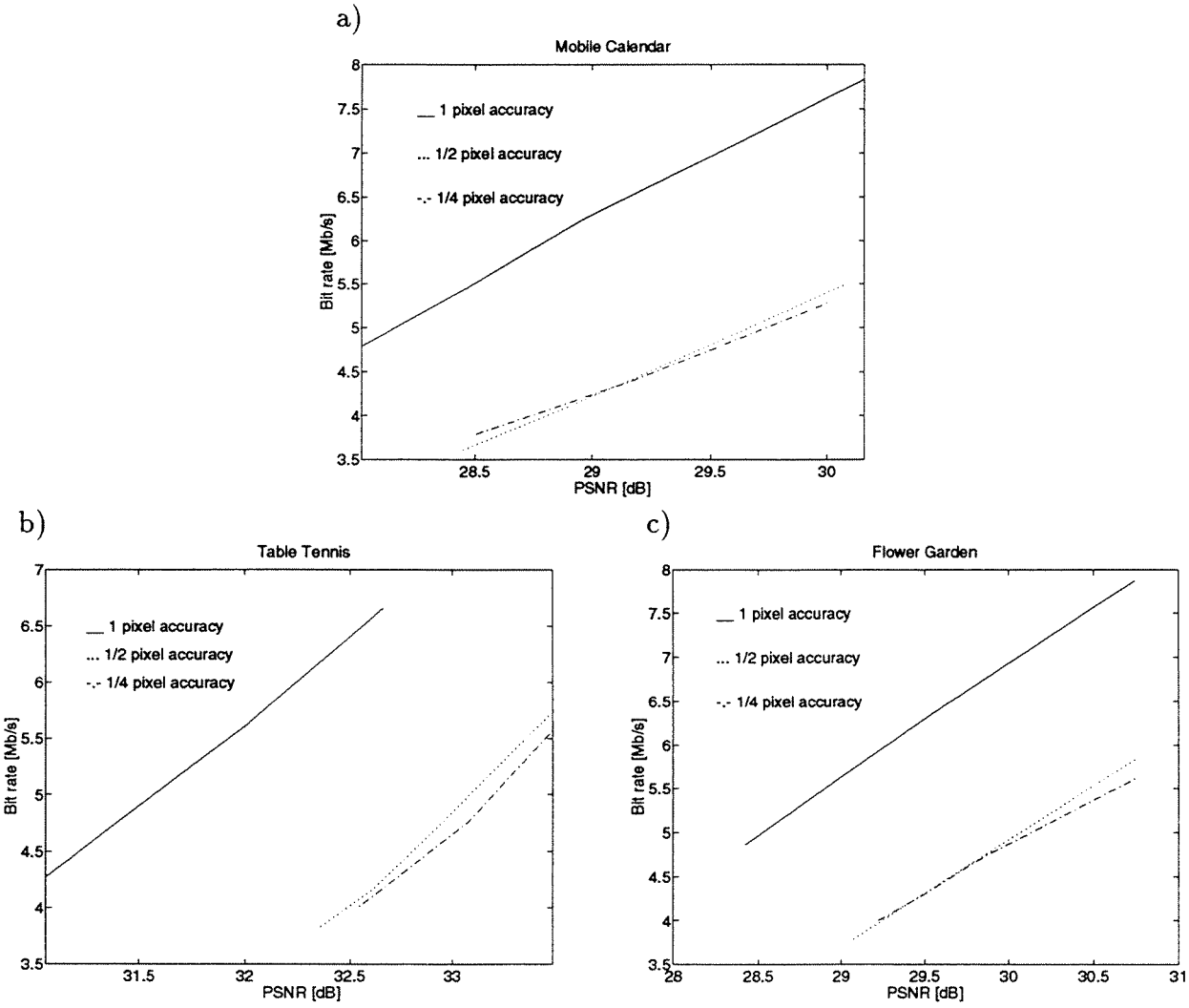

Figure 4.20: Bit rate versus PSNR for the coding scheme $\mathcal{C}$: comparison between 1, 1/2 and 1/4 pixel accuracy for the multigrid algorithm, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

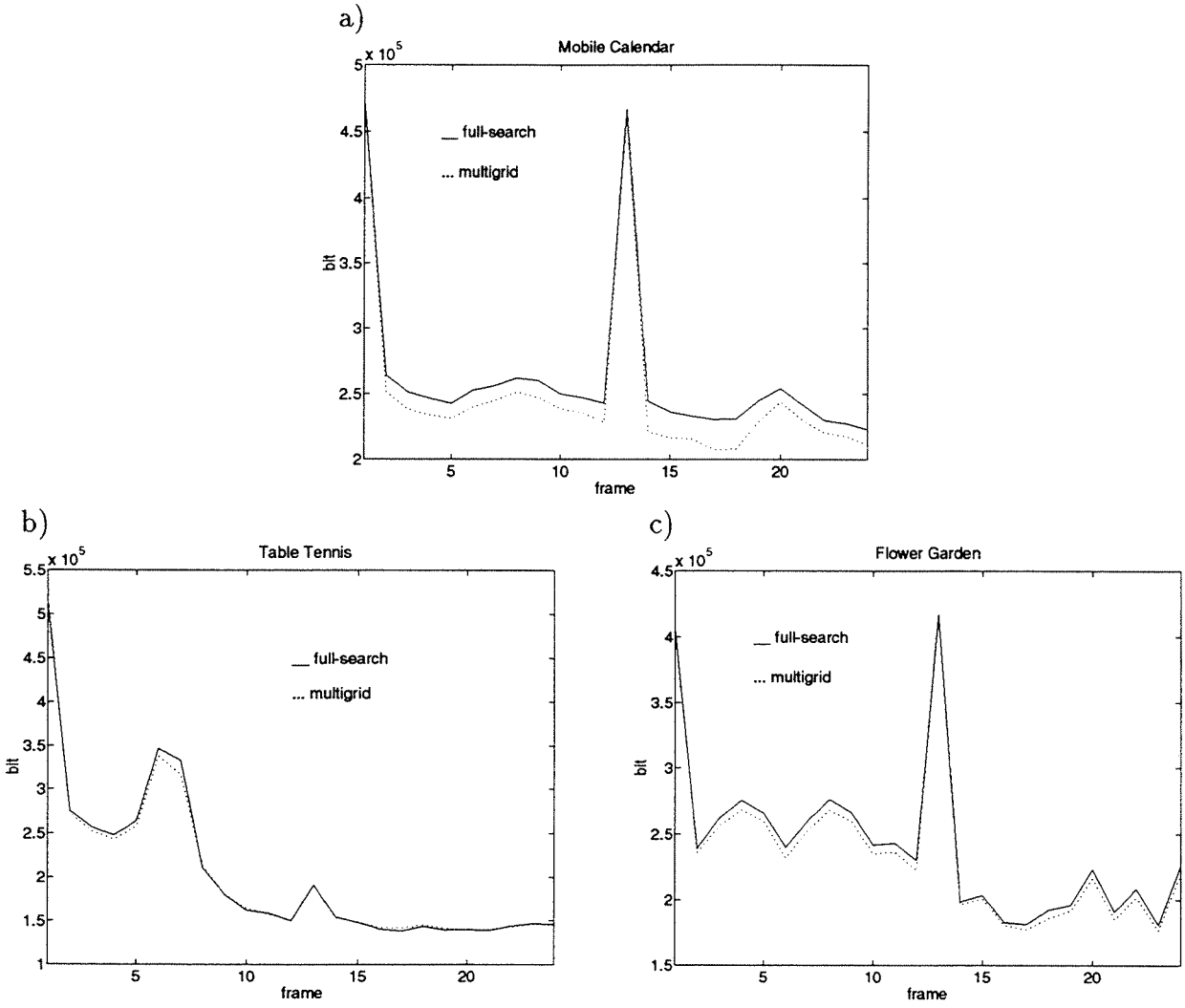

Figure 4.21: Bit rate for the scheme $\mathcal{A}$: comparison between full-search and fine-to-coarse-to-fine multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

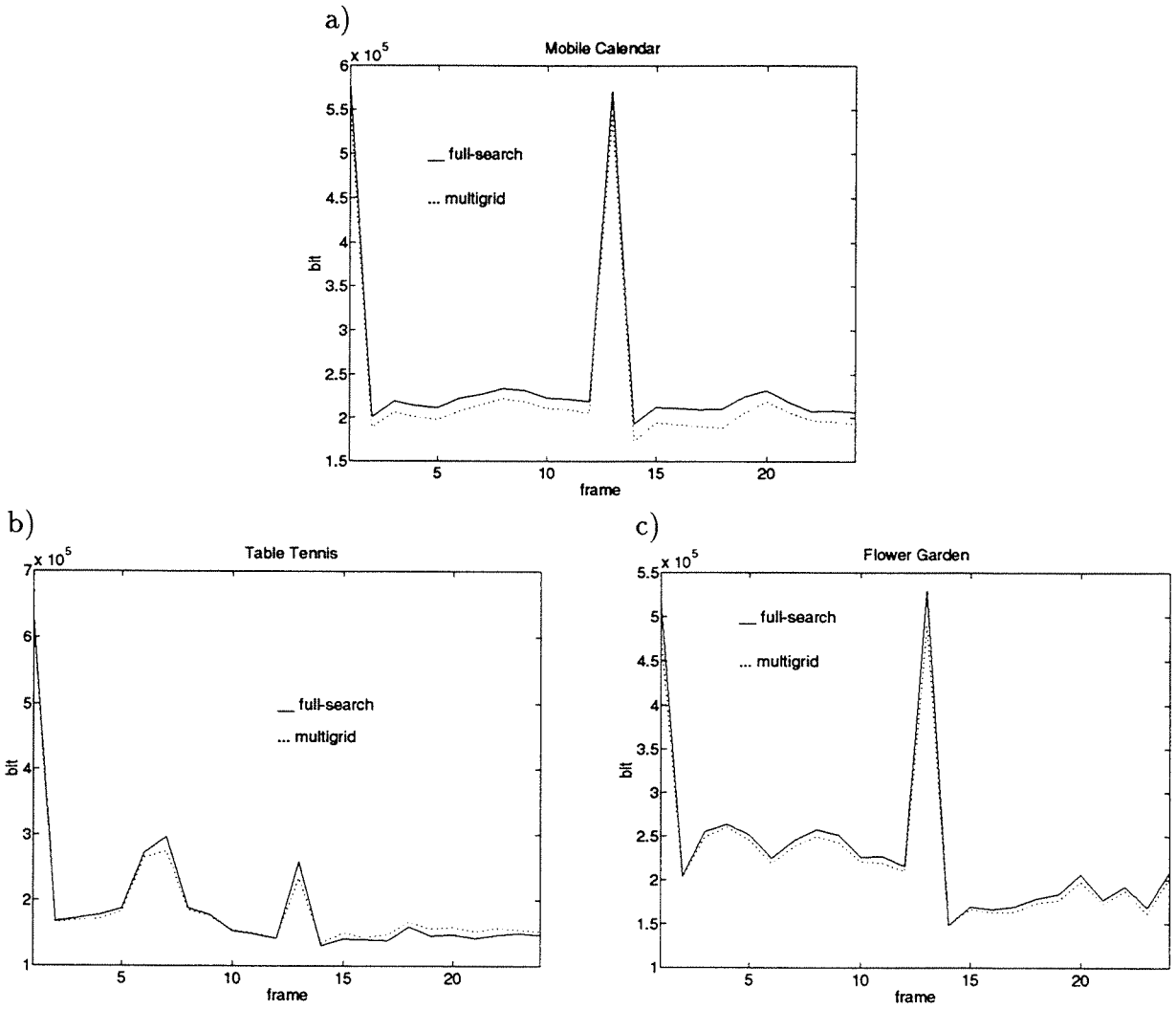

Figure 4.22: Bit rate for the scheme $\mathcal{B}$: comparison between full-search and fine-to-coarse-to-fine multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

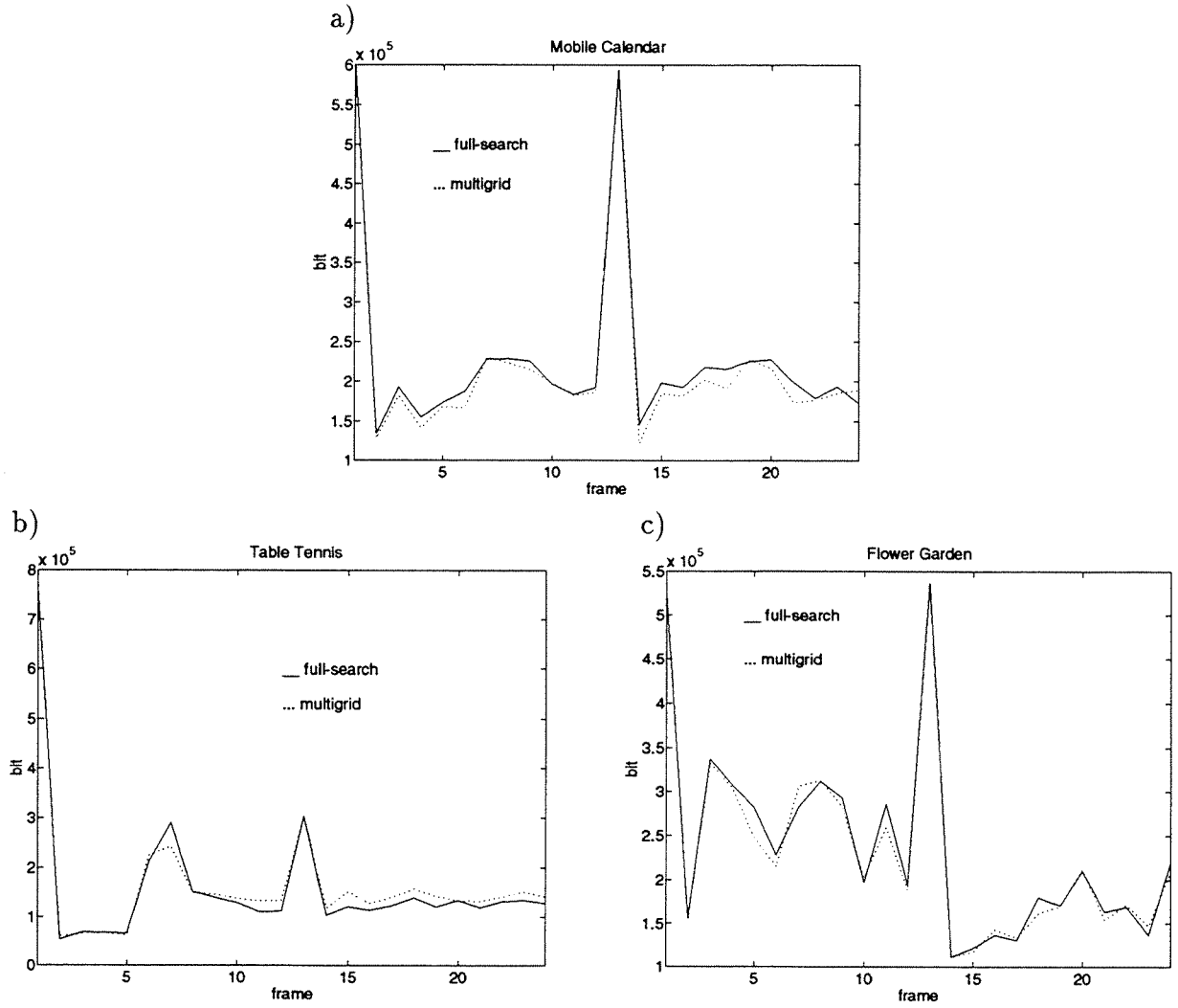

Figure 4.23: Bit rate for the scheme $\mathcal{C}$: comparison between full-search and fine-to-coarse-to-fine multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

		intra+inter [Mb/s]	motion [Mb/s]	total [Mb/s]	PSNR [dB]
scheme $\mathcal{A}$					
Mobile	full-search	5.548	1.017	6.565	31.92
Calendar	multigrid	5.411	0.834	6.245	31.94
Table	full-search	3.587	1.475	5.062	33.13
Tennis	multigrid	3.622	1.414	5.036	33.09
Flower	full-search	4.779	1.261	6.040	32.44
Garden	multigrid	4.739	1.175	5.914	32.45
scheme $\mathcal{B}$					
Mobile	full-search	5.116	1.031	6.147	31.20
Calendar	multigrid	4.918	0.841	5.759	31.17
Table	full-search	3.264	1.484	4.748	32.31
Tennis	multigrid	3.323	1.429	4.752	32.11
Flower	full-search	4.576	1.290	5.866	31.09
Garden	multigrid	4.467	1.189	5.656	31.06
scheme $\mathcal{C}$					
Mobile	full-search	4.636	1.047	5.683	30.01
Calendar	multigrid	4.622	0.859	5.481	30.07
Table	full-search	2.522	1.464	3.986	32.59
Tennis	multigrid	2.760	1.418	4.178	32.62
Flower	full-search	4.624	1.297	5.921	30.80
Garden	multigrid	4.627	1.200	5.827	30.74

Table 4.4: Comparison between full-search and multigrid algorithms in schemes $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$: bit rate corresponding respectively to intraframe and interframe, motion vectors and total (all expressed in Mb/s), as well as PSNR (in dB).

from 0.1 to 0.4 Mb/s, corresponding approximately to 2% to 7% saving. This gain is obtained both on the motion vectors information (as motion fields are smoother), and on the DFD coding (as the DFD have less artificial discontinuities).

Another important aspect is visual quality. Due to the smoother motion field, avoiding artificial discontinuities in the DFD, the multigrid algorithm provides higher visual quality, when compared to the full-search. However this point is very difficult to assess in practice for two reasons. First, the evaluation of the visual quality of a reconstructed sequence is a complex task, and in particular the PSNR does not provide an accurate measure. Second, it is difficult to distinguish between intrinsic coding artifacts due to the quantization and artifacts due to the motion estimation.

As a conclusion of this simulation, even though the multigrid algorithm provides more accurate motion vectors in the sense of the motion in the scene, when compared to the full-search block matching technique, it leads only to a small improvement in a video coding scheme. The last observation is not surprising. In fact, the multigrid algorithm prevents artificial discontinuities in the DFD but does not improve the DFD energy minimization. Therefore, although it provides a higher visual quality, it leads only to a small gain in the DFD coding. Furthermore, the gain on the motion side information, due to smoother motion fields, is low as the latter represents only a fraction of the overall bit rate. Consequently, the most significant gain of the multigrid algorithm is the decreased computational complexity when compared to full-search. To reach higher coding performances, a more sophisticated multigrid motion estimation algorithm is required. For this purpose, in Chap. 5, a locally adaptive multigrid technique is presented.

The above presented results are bounded to medium bit rate applications and CCIR 601 format. Similar figures have been obtained when applying the multigrid motion estimation to HDTV and low bit rate coding [103, 53, 104].

4.5 Summary

In this chapter, a multigrid block matching motion estimation technique has been proposed and widely analyzed. It is based on the multigrid theory initially developed in the field of numerical analysis. The technique iteratively refines the motion field solution on a set of grids with different resolutions. Problems relative to the control strategy to iterate the process throughout the multigrid structure, as well as the up- and down-conversion operators to map the motion field estimate between two consecutive grid levels have been addressed. Due to the multigrid structure, the conflicting requirements on the matching window size are overcome. Hence, the method generates smooth and robust motion fields. Furthermore, the computational complexity is greatly reduced when compared to

full-search block matching motion estimation.

Extensive simulations have been carried out and the following conclusions have been drawn. The MAE and the MSE matching criteria perform similarly whereas the MAE requires a lower complexity. The fine-to-coarse-to-fine control strategy, the up-conversion by median filtering and the down-conversion by selecting the best initial condition among the motion vectors estimated in a neighborhood reach the highest performances. The half-pixel accuracy corresponds to the best compromise between performances and complexity. Finally, the multigrid algorithm is quasi-optimal in terms of minimizing the DFD energy, when compared to the full-search technique. In contrast, none of the classical fast search techniques leads to performances close to those provided by the full-search in this respect. Furthermore, the multigrid algorithm generates smooth and robust motion fields close to the true motion in the scene. It results in a lower entropy of the motion vectors and consequently a lower amount of side information. Besides, the multigrid approach requires a greatly decreased computational complexity. Nevertheless, the multigrid algorithm leads only to small improvements in terms of coding performances.

Chapter 5

Locally adaptive multigrid block matching motion estimation

5.1 Introduction

The multigrid block matching motion estimation technique proposed in Chap. 4 aims at overcoming the conflicting requirements on the matching window size, thus providing simultaneously smooth and accurate motion fields. Even though this method represents a large saving in terms of computation time when compared to full-search block matching, it leads only to a small gain in terms of coding efficiency.

In order to reach higher performances, a locally adaptive multigrid block matching motion estimation is proposed in this chapter. This method is based on the idea of local mesh refinement proposed in [142] to improve multigrid techniques to solve partial differential equations. It shares some similarities with the variable size block matching motion estimation introduced by Chan *et al.* in [16].

The block-based nature of the block matching techniques represents a serious drawback. The assumption that the motion within each block is uniform is not valid for blocks which include the boundary of a moving object. For those blocks, the motion compensated prediction is poor, and even though moving edges constitute only a small part of the entire image, the human visual system is very sensitive to their degradation [148]. Furthermore, the method results in block artifacts in the motion compensated frame and consequently in the DFD. When the latter is coded using a subband decomposition, a wavelet transform or a segmentation based technique, these block artifacts decrease significantly the efficiency of the coding technique. Therefore, small blocks are required to maintain the accuracy of the motion field. However, decreasing the block size increases the side information to transmit the motion vectors.

In order to overcome the above conflicting requirements on the block size, a variable size block matching motion estimation is proposed in [16]. The decomposition in blocks of varying sizes is carried out by a quad-tree decomposition [149]. Since a quad-tree decomposition produces a hierarchy of nested meshes, it is natural to include it in a multigrid algorithm. Consequently, the locally adaptive multigrid block matching algorithm proposed in this chapter combines a quad-tree decomposition with a multigrid structure. Therefore, it adds the advantage of the locally varying block size to the benefits of the multigrid algorithm, distinguishing it from the algorithm described in [16]. The technique is referred to as the adaptive multigrid algorithm in the following, whereas the multigrid algorithm refers to the technique without adaptation described previously in Chap. 4.

The locally adaptive multigrid block matching generates large blocks in uniform areas and small blocks on moving objects boundaries. Hence, it leads to more accurate motion fields with a decreased overhead information. Furthermore, the algorithm keeps the advantages of the multigrid algorithm, namely smooth and robust motion fields as well

as low computational complexity.

This chapter is structured as follows. First, the locally adaptive multigrid structure is described in Sec. 5.2. The split procedure as well as the criterion to decide whether to split a block are discussed in more details. Simulation results are presented in Sec. 5.3. Finally, based on the simulation results a conclusion is drawn in Sec. 5.4.

5.2 Adaptive multigrid structure and quad-tree decomposition

Let us describe now in more details the adaptive multigrid structure and the quad-tree decomposition.

In the adaptive multigrid algorithm, the multigrid structure is made locally adaptive by a quad-tree decomposition. Figure 5.1 illustrates an example of such an adaptive structure. It corresponds to the pruning of the multigrid structure shown in Fig. 4.2.

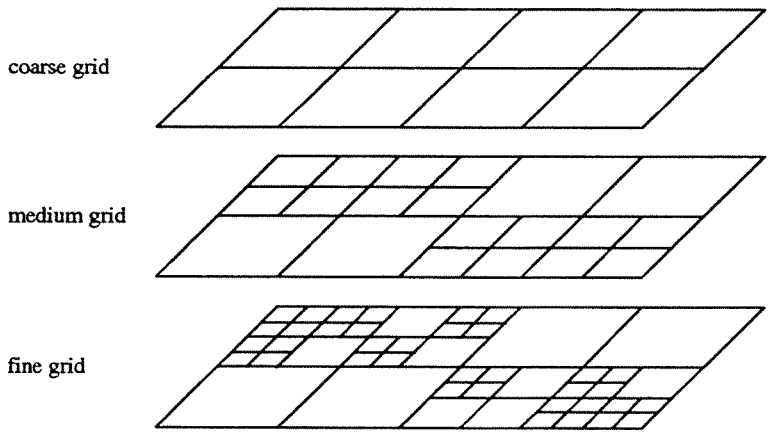

Figure 5.1: Example of a 3-grid adaptive multigrid structure.

The decomposition is carried out by a split procedure. The algorithm is coarse-to-fine. It starts by estimating a motion vector for each block of the coarsest grid. Then, based on an estimate of the current solution reliability, the mesh is split only in specific regions and the corresponding motion vectors are refined. At this stage, different segmentation decision rules can be used, for instance blocks where large matching errors occur are split. The down-conversion is carried out by the best initial condition in a neighborhood (see Sec. 4.3). The refinement is iterated until a satisfactory accurate solution (in the sense of the segmentation decision rule) is obtained or a minimum block size is reached. The

algorithm results in small grid sizes in areas containing details and large grid sizes in the uniform ones. Consequently, the conflicting requirements on the block size are solved, and the method leads globally to more accurate motion vectors with a reduced overhead information.

The segmentation information, namely the quad-tree, should be sent to the decoder as side information. Nevertheless, as one bit per node of the tree is sufficient to completely define the segmentation, it represents a very low amount of information.

Depending on the segmentation rule, the algorithm can significantly improves the motion vectors accuracy and consequently the prediction (many blocks are split) or reduces the amount of motion vectors to be transmitted (few blocks are split). Even though it is very easy to gain on one of these two terms, prediction or side information, the challenge of motion estimation is to achieve an overall gain.

For this purpose, we will compare two different adaptive multigrid structures. The first one corresponds to the pruning of the multigrid structure described in Sec. 4.3 and Table 4.1. The second one consists of four grids, a level with a block size of 4×4 pixels and a 2-step search being added. Table 5.1 summarizes the characteristics of these two different adaptive multigrid structures (the numbers refer to the blocks which are split). The resulting maximum displacements are ± 25 and ± 28 pixels for the first and second structures respectively. When compared to the full-search block matching or the multigrid algorithm without adaptation, the first structure aims at decreasing the motion side information to be transmitted, whereas the second one can potentially decrease both the amount of motion vectors and the DFD information.

	grid	block size	match. window	search	max. displ.
structure 1					
	Ω_0	8×8	8×8	2-step	± 3
	Ω_1	16×16	16×16	3-step	± 7
	Ω_2	32×32	32×32	4-step	± 15
structure 2					
	Ω_0	4×4	4×4	2-step	± 3
	Ω_1	8×8	8×8	2-step	± 3
	Ω_2	16×16	16×16	3-step	± 7
	Ω_3	32×32	32×32	4-step	± 15

Table 5.1: The adaptive multigrid structure: two different configurations.

5.2.1 Segmentation decision rule

The above split procedure requires a criterion to evaluate the accuracy of the current motion vectors. More precisely it requires a rule to decide whether to split a block. Simulations point out the high importance of this criterion and its significant influence on the overall performance of the motion estimation procedure.

A simple method to evaluate the accuracy of a solution is to use the matching error. Blocks for which this error is above a preset threshold are defined as unsatisfactory and are further split:

- If the MAE (or another error measure) of the motion compensated block is above a preset threshold T, the block is split.

$$\mathrm{MAE}_{\mathrm{nosplit}} > T \Rightarrow \text{split} \ . \tag{5.1}$$

Of course, more generally the threshold could depend on the multigrid level. The method defined by Eq. (5.1) has been successfully applied in [102, 150, 16].

An important difficulty of this approach is to determine the value of the threshold T. It is clear that a low threshold value leads to a more accurate segmentation, in other words more accurate motion vectors, but to a higher side information. Conversely, a high threshold value results in poorer motion vectors, but requires a lower amount of overhead information. Therefore there is an optimal trade-off to find. In addition, the optimality depends on the type of application and on the target bit rate. Consequently, several trials are required to set a satisfactory threshold value, nevertheless this algorithm is not workable in practice. Furthermore, the above criterion does not guarantee that the extra-cost to send more motion parameters is worth the improvement of the DFD coding.

Taking into account the above remarks, in Chap. 7 a criterion is proposed to control the segmentation in order to reach the optimal bit allocation between motion parameters and DFD information. This criterion compares the extra-cost to send additional motion parameters with the gain obtained on the DFD side to decide whether to split a block. Nevertheless, in the remaining of this chapter, the decision rule defined by Eq. (5.1) is used.

Figure 5.2 depicts an example of the final segmentation obtained on the sequence "Table Tennis" while using the adaptive multigrid algorithm (structure 2) and the segmentation criterion defined by Eq. (5.1). The algorithm clearly generates large blocks in uniform area and small blocks on the player's arm, the bat and the ball.

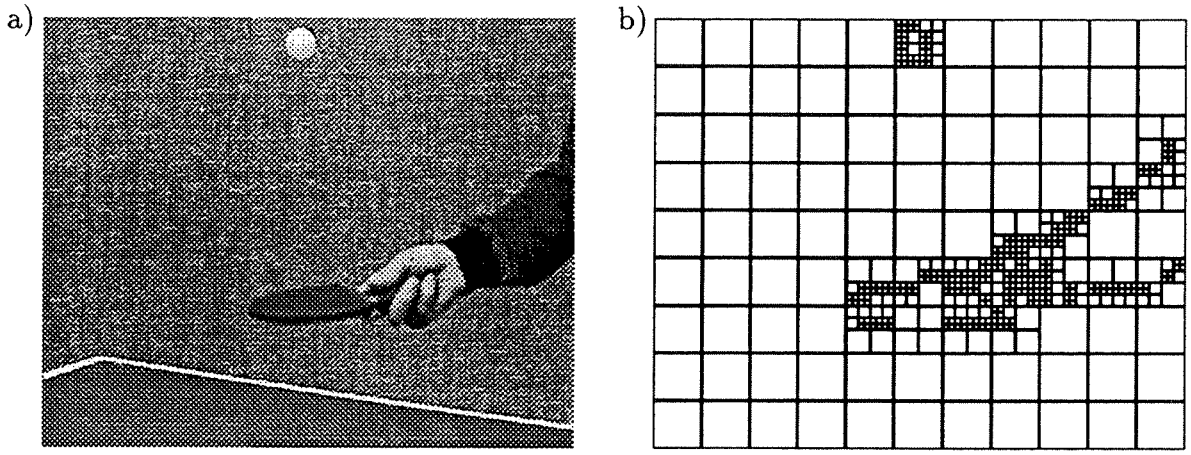

Figure 5.2: A frame of a) "Table Tennis" and b) the corresponding final grid.

5.3 Simulation results

Comparative results between the adaptive multigrid, the multigrid and the full-search algorithms are presented now. Results are expressed in terms of DFD energy, number of motion vectors and bit rate. Simulations have been carried out on the same sequences and with the same coding schemes as in Sec. 4.4.

5.3.1 Comparison between the multigrid and the locally adaptive multigrid block matching motion estimation techniques in terms of DFD energy and number of motion vectors

The simulation compares the multigrid algorithm with and without adaptation in terms of DFD energy and the number of motion vectors. Figures 5.3 and 5.4 shows the DFD energy obtained with both methods. The adaptive multigrid algorithm is either in the first or second configuration, and with a threshold of $T = 2, 4, 6, 8, 10$. Figures 5.5 and 5.6 shows the corresponding number of motion vectors to be transmitted.

An observation which can be done from this simulation is the problem of the trade-off between the amount of information relative to DFD (i.e. the motion estimation accuracy) on the one side and the motion information on the other side. Depending on the threshold value, one of these two terms can be decreased, resulting in an increase of the second one. This remark underlines the importance of the threshold.

This simulation clearly show that the first structure has the capability to decrease signifi-

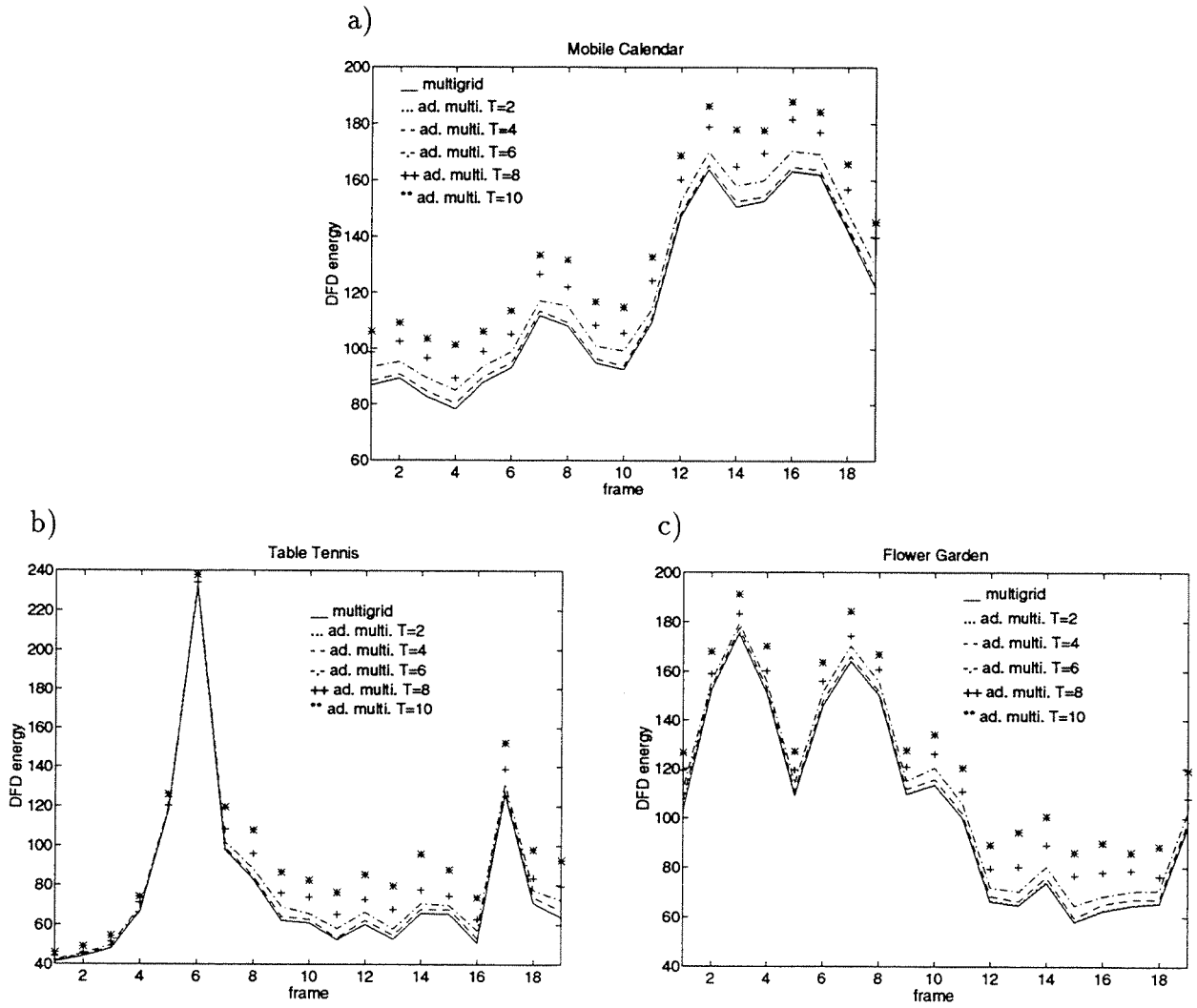

Figure 5.3: DFD energy: comparison between adaptive multigrid (structure 1) and multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

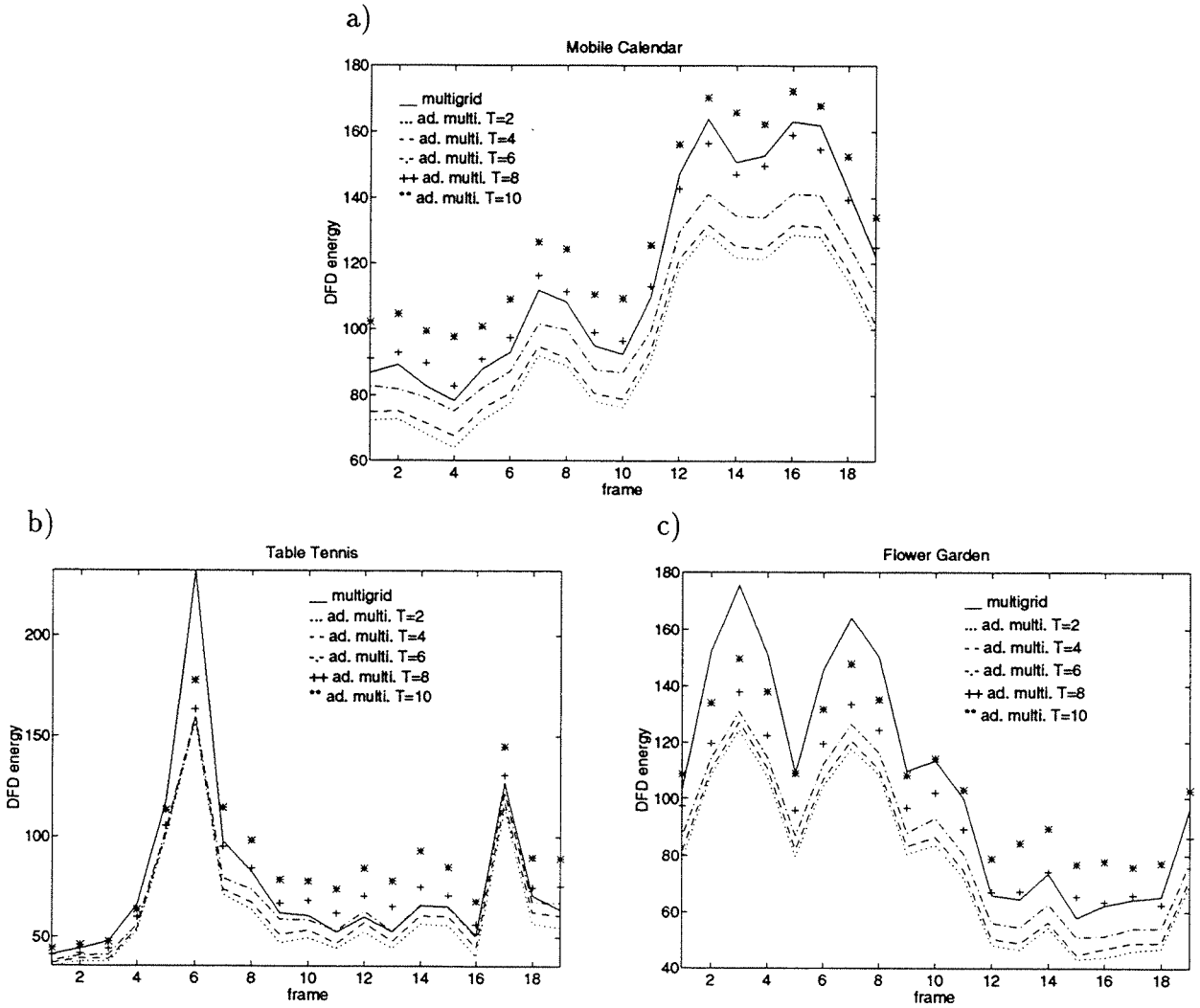

Figure 5.4: DFD energy: comparison between adaptive multigrid (structure 2) and multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

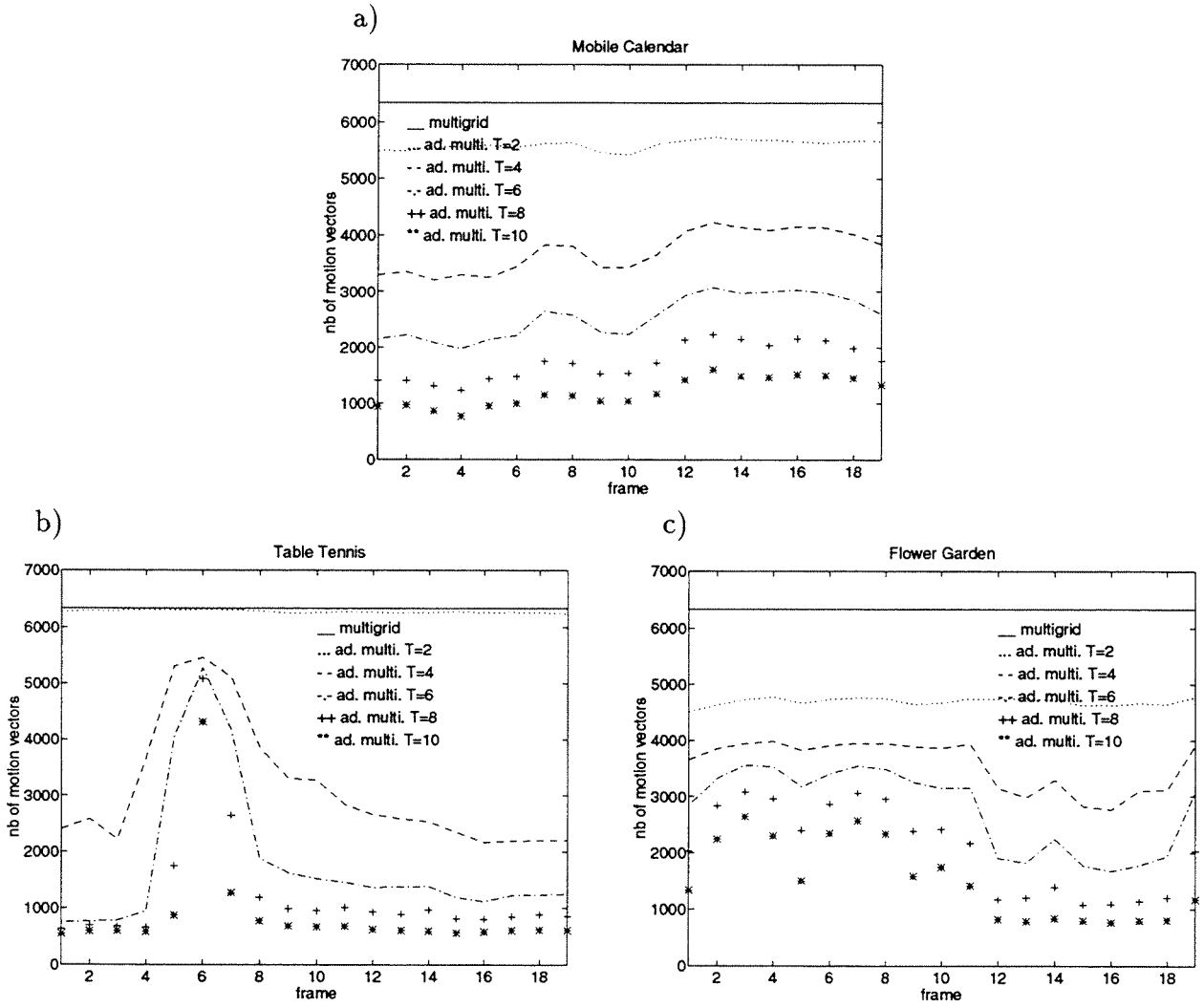

Figure 5.5: Number of motion vectors: comparison between adaptive multigrid (structure 1) and multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

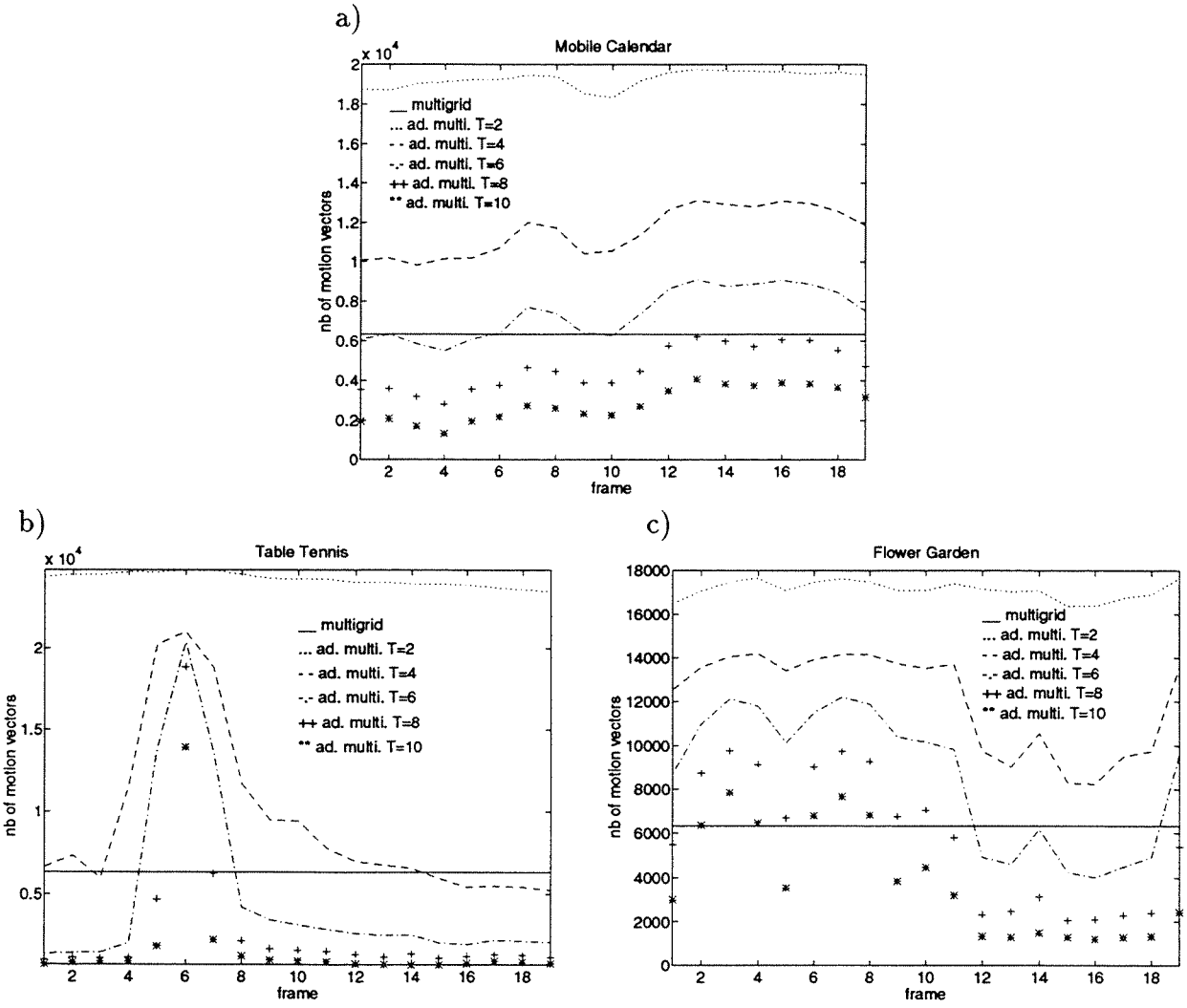

Figure 5.6: Number of motion vectors: comparison between adaptive multigrid (structure 2) and multigrid algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

cantly the overhead information (the number of motion vectors is greatly reduced) while keeping an accurate prediction (the DFD energy is only slightly increased). The performance of the second structure depends strongly on the threshold. Whereas for a high threshold value it performs similarly to the first structure (reduction of the side information), for a small threshold value, the second structure allows to improve the motion compensated prediction (decreased DFD energy) without significantly increasing the number of motion vectors.

5.3.2 Comparison between the locally adaptive multigrid and the full-search block matching motion estimation techniques in terms of bit rate and PSNR

The above results do not allow to assess the performance of the adaptive multigrid motion estimation in terms of coding. Therefore, simulations are now carried out with the three coding schemes $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ described in Chap. 3.

Figures 5.7, 5.8 and 5.9 compares the bit rate obtained while using the full-search block matching and the adaptive multigrid motion estimation (structures 1 and 2). The full-search algorithm is applied with a block size of 8×8 pixels and a maximum displacement of ± 15 pixels. The threshold value in the adaptive multigrid method has been set after several trials, even though it is not workable in practice. In both configurations, the threshold has been set to $T = 6$. In the first structure, it corresponds to a decreased side information for an identical motion accuracy, whereas in the second one, it leads to an improved prediction for a comparable amount of motion information. Therefore, this choice shows both properties of the adaptive multigrid technique: the ability to reduce the side information or to improve the motion estimation. It should underlined that with a higher threshold value, the second structure allows also to decrease the overhead information while keeping an identical motion accuracy, thus performing similarly to the first structure.

Table 5.2 summarizes the results corresponding to Figs. 5.7, 5.8 and 5.9. It indicates the bit rate relative to the intraframe and the DFDs coding, the motion side information and the split information, as well as the PSNR of the reconstructed sequence.

A first remark before analyzing these results is that the PSNR is a poor measure of the visual quality in the context of the adaptive multigrid motion estimation. Actually, the latter algorithm aims at improving the coding of moving edges. As edges are perceptually very important [148], it results in a significantly higher visual quality of the reconstructed sequence. Nevertheless, edges constitute only a small portion of the entire image and thus the gain in terms of PSNR is low (the PSNR is even sometimes decreased as uniform areas, which are less important perceptually are less accurately predicted). Despite this

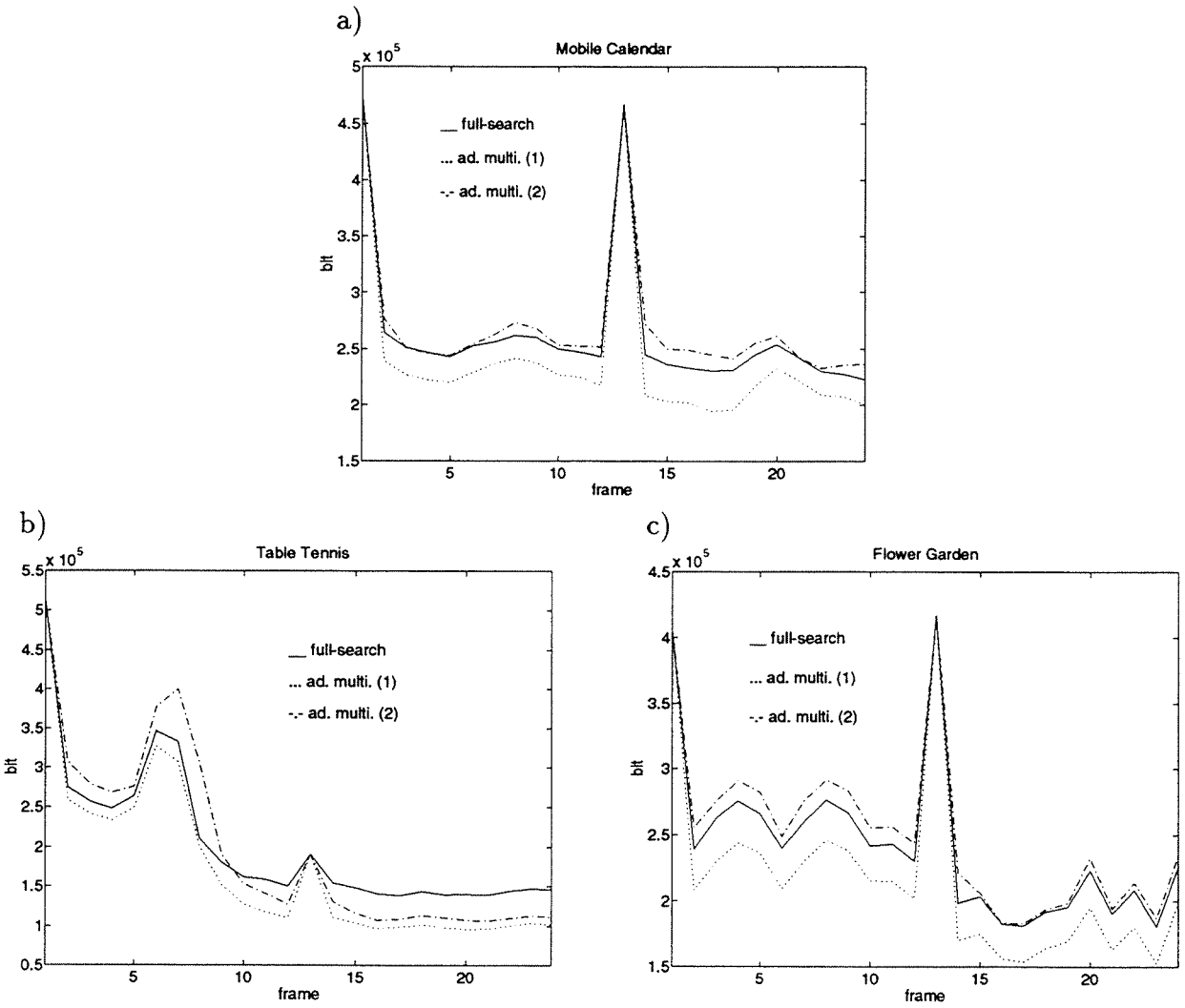

Figure 5.7: Bit rate for the scheme $\mathcal{A}$: comparison between full-search and adaptive multigrid algorithms (structures 1 and 2), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

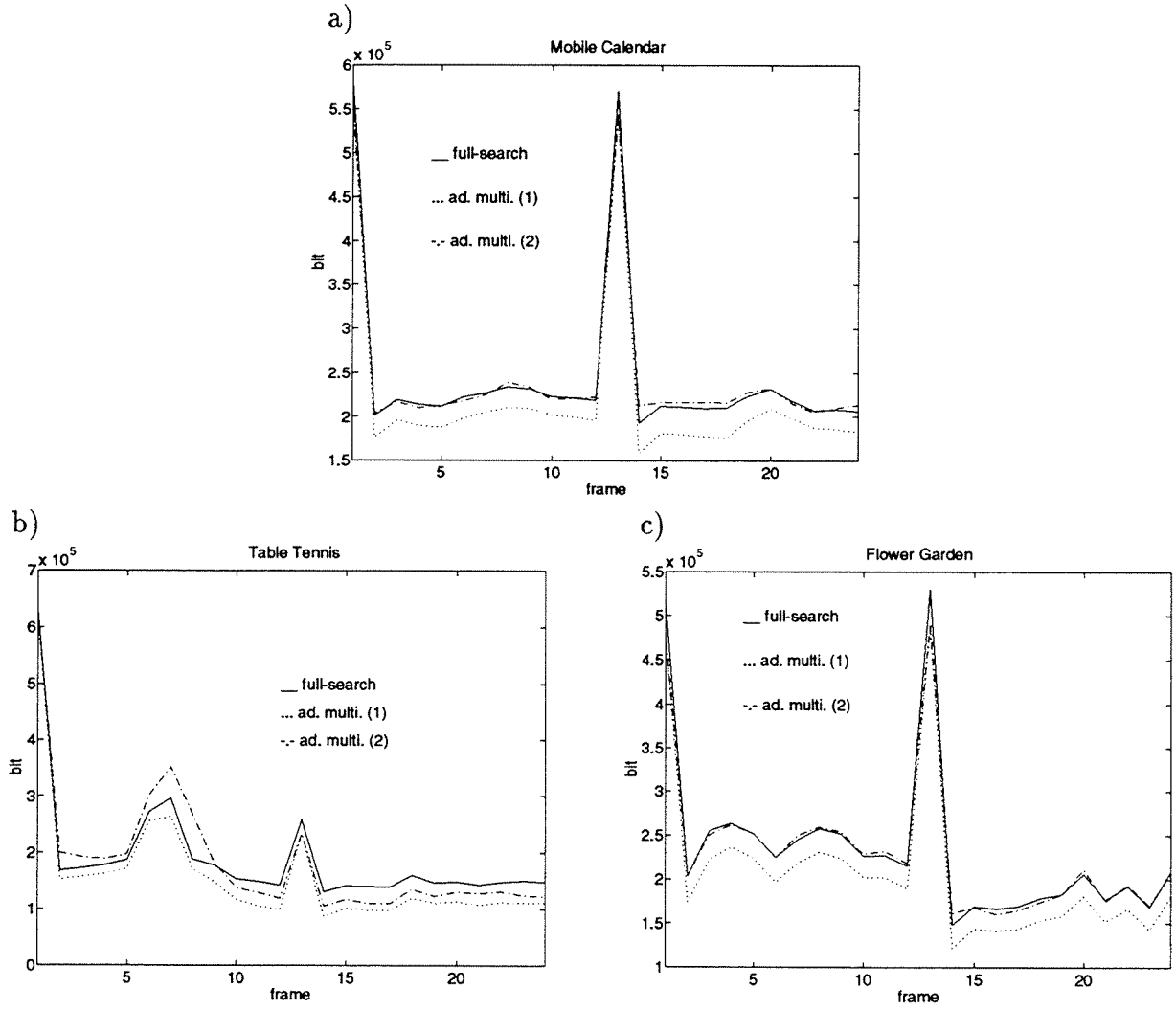

Figure 5.8: Bit rate for the scheme $\mathcal{B}$: comparison between full-search and adaptive multigrid algorithms (structures 1 and 2), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

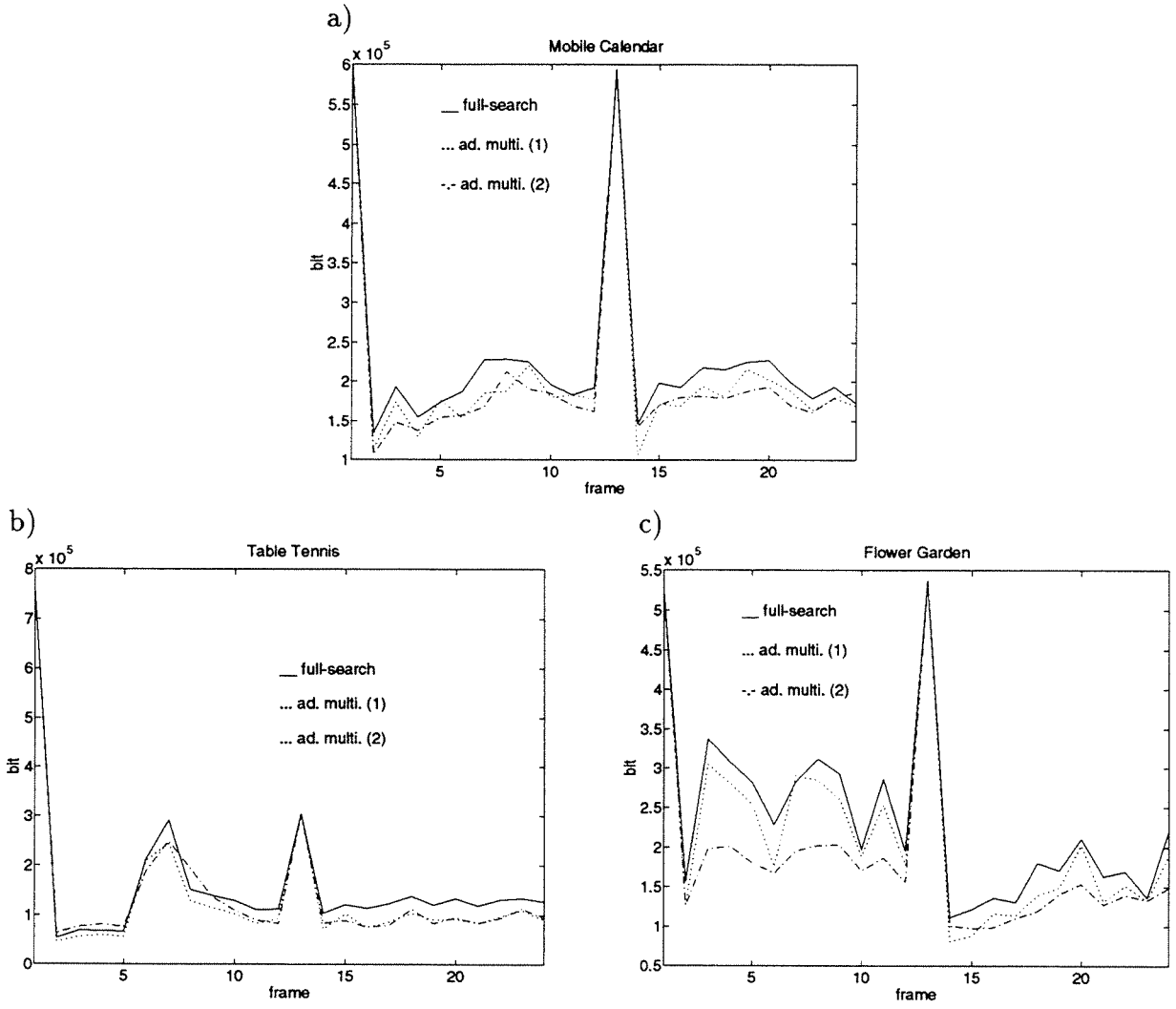

Figure 5.9: Bit rate for the scheme $\mathcal{C}$: comparison between full-search and adaptive multigrid algorithms (structures 1 and 2), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

		intra+inter [Mb/s]	motion [Mb/s]	split [Mb/s]	total [Mb/s]	PSNR [dB]
scheme $\mathcal{A}$						
Mobile Calendar	full-search	5.548	1.017		6.565	31.92
	ad. multi. (1)	5.543	0.413	0.033	5.989	31.79
	ad. multi. (2)	5.215	1.440	0.103	6.758	31.94
Table Tennis	full-search	3.587	1.475		5.062	33.13
	ad. multi. (1)	3.798	0.480	0.028	4.306	32.80
	ad. multi. (2)	3.429	1.450	0.080	4.959	32.99
Flower Garden	full-search	4.779	1.261		6.040	32.44
	ad. multi. (1)	4.832	0.525	0.031	5.388	32.33
	ad. multi. (2)	4.354	1.818	0.107	6.279	32.49
scheme $\mathcal{B}$						
Mobile Calendar	full-search	5.116	1.031		6.147	31.20
	ad. multi. (1)	5.034	0.451	0.035	5.520	30.96
	ad. multi. (2)	4.554	1.481	0.106	6.141	31.38
Table Tennis	full-search	3.264	1.484		4.748	32.31
	ad. multi. (1)	3.398	0.549	0.029	3.976	31.56
	ad. multi. (2)	3.009	1.538	0.084	4.631	32.00
Flower Garden	full-search	4.576	1.290		5.866	31.09
	ad. multi. (1)	4.572	0.571	0.032	5.175	30.85
	ad. multi. (2)	3.762	1.928	0.110	5.800	31.47
scheme $\mathcal{C}$						
Mobile Calendar	full-search	4.636	1.047		5.683	30.01
	ad. multi. (1)	4.717	0.472	0.035	5.224	29.91
	ad. multi. (2)	3.435	1.576	0.110	5.121	29.46
Table Tennis	full-search	2.522	1.464		3.986	32.59
	ad. multi. (1)	2.899	0.458	0.027	3.384	31.93
	ad. multi. (2)	2.236	1.236	0.075	3.547	31.97
Flower Garden	full-search	4.624	1.297		5.921	30.80
	ad. multi. (1)	4.750	0.579	0.032	5.361	30.50
	ad. multi. (2)	2.528	1.958	0.110	4.596	29.97

Table 5.2: Comparison between full-search and adaptive multigrid (structures 1 and 2) in schemes $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$: bit rate corresponding respectively to intraframe and interframe, motion vectors, split and total (all expressed in Mb/s), as well as PSNR (in dB).

severe drawback, the PSNR is commonly used in coding due to the lack of perceptually reliable visual quality measures.

With regard to the first structure, the following conclusions can be drawn from the results. For the three schemes, the method leads to a significantly decreased bit rate when compared to full-search block matching. The gain ranges from 0.5 to 0.8 Mb/s which represents about 10% saving for "Mobile Calendar" and "Flower Garden" and 15% for "Table Tennis". This gain is obtained due to the greatly reduced motion information, whereas the DFDs coding cost increases unsignificantly. Concerning the reconstructed sequence quality, the PSNR is slightly decreased. However, the visual quality is identical to the one obtained with full-search block matching.

As far as the second structure is concerned, conclusions are very different. First the schemes $\mathcal{A}$ and $\mathcal{B}$ on the one side, and $\mathcal{C}$ on the other side exhibit very different behaviors. For the schemes $\mathcal{A}$ and $\mathcal{B}$, the bit rate obtained while using the adaptive multigrid algorithm is comparable to the one reached by the full-search block matching (difference ranging from -2% to +4%). The gain achieved on the DFD side is balanced by the increased side information. The PSNR is slightly increased for the sequence "Mobile Calendar" and "Flower Garden", but decreased for the sequence "Table Tennis". However, for the three sequences, moving edges are sharper and the visual quality is higher. Results obtained with the scheme $\mathcal{C}$ are very different. As in this scheme, a segmentation of the DFD is performed and only regions of high energy are coded, the improvement of the motion compensated prediction introduces a very important gain on the DFDs side. Consequently, the adaptive multigrid algorithm leads to a gain in terms of bit rate of about 0.5 Mb/s for "Mobile Calendar" and "Table Tennis" (10% saving) and 1.3 Mb/s for "Flower Garden" (22% saving). The reconstructed sequences have a slightly decreased PSNR (as fewer regions of the DFD have been coded), however moving edges are sharp and the visual quality is high.

The above results clearly demonstrate the efficiency of the locally adaptive multigrid block matching motion estimation. Significant gains have been achieved either in terms of bit rate and/or visual quality, when compared to full-search block matching. The choice between the two proposed configurations depends on the subsequent coding technique, as well as the type of application and the target bit rate. The problem relative to the optimal threshold selection remains unsolved yet. It will be addressed in Chap. 7. The above results are limited to medium bit rate coding, experimental results obtained when applying the method to HDTV and low bit rate coding can be found in [103, 53, 104].

5.4 Summary

In this chapter, an adaptive multigrid structure has been introduced. It generates small blocks in detailed areas and large blocks in uniform ones. Therefore it solves the conflicting requirements on the block size, and allows an improved motion estimate with a decreased side information.

Simulation results show the ability of the algorithm to either greatly decrease the side information while keeping an identical prediction accuracy, or to significantly improve the prediction without increasing the overhead information. A significant gain up to 22% in terms of bit rate is reported while using various sequences and different coding schemes. Furthermore, the method leads to sharper moving edges and therefore to an enhanced visual quality of the reconstructed sequence.

Chapter 6

Segmentation of the motion field based on vector quantization

6.1 Introduction

Block matching motion estimation techniques are widely used in video coding, due to their simplicity, their ease of hardware implementation and their low overhead information to transmit the motion vectors. However, they produce motion fields of poor resolution, especially on the edge of moving objects. Moreover, they tend to introduce annoying block artifacts in the motion compensated frame. In block-based coding schemes such as those using the DCT (e.g. MPEG-I [5, 30], MPEG-II [6, 31] and H.261 [7]), as the block boundaries of the motion estimation correspond to the boundaries of the transform support, the block artifacts does not constitute a limitation. Nevertheless, the latter assessment is no longer true for coding schemes performing a subband decomposition, a wavelet transform or a segmentation of the DFD. In this case, the block-based nature of the motion estimation represents a serious drawback and results in a significantly decreased performance of the DFD coding technique.

In order to overcome the problems related to the block-based nature of the block matching motion estimation techniques, different approaches have been investigated (see Chap. 2). Two simple methods are overlapped windows [106, 107, 108, 109] and control grid interpolation [110, 111]. Although the latter have shown their efficiency, more promising approaches consider motion field segmentation.

When performing a segmentation of the motion field, accurate motion estimation and low cost overhead information are two conflicting requirements. It is straightforward that a very precise segmentation leads to an accurate motion field and thus a low DFD energy, but also to a high overhead segmentation information. Conversely, a coarse segmentation entails a low overhead information, but a high energy DFD. The challenge of the motion field segmentation techniques lies undoubtedly in providing simultaneously an accurate segmentation (i.e. an accurate motion field), and low cost overhead information.

In variable size block matching [16, 102, 53] (e.g. the locally adaptive multigrid block matching algorithm, see Chap. 5), segmentation is performed by means of a split technique. The simplicity of the segmentation and the low overhead bit rate required to represent the contour information are the main advantages of this approach. Nevertheless, the segmented regions are still constrained to square blocks, which obviously is not a good approximation of moving objects. Another approach estimates in a first stage a block-based motion field, and then segments blocks corresponding to moving edges [112]. In order to improve the procedure, the segmentation information of the previous frame is also taken into account. Finally, object-based motion estimation algorithms have been proposed [75, 76, 77]. These methods take into account the fact that the scene is composed of objects. Motion estimation is performed based on this model, resulting in a more accurate evaluation of the motion field. Clearly, this representation should be coded and

transmitted together with the respective motion information. In particular, the contour information of the different regions has to be efficiently represented. In [76], entropy coding of vertex coordinates is performed, while in [77], a chain code is used. Depending on the complexity of the scene and on the target bit rate, the overhead segmentation information may become important.

In this chapter, a new method to segment block-based motion field is proposed [113, 114]. It is based on VQ used for segmentation. The method overcomes the block-based constraint of block matching motion estimation techniques (or any block-based motion estimation). Once a block-based motion field has been evaluated, blocks corresponding to moving edges are segmented. Thus, a more accurate motion compensated prediction is obtained along boundaries of moving objects. It is consistent with the sensitivity of the human visual system to degradation along edges in an image [148]. In order to keep the bit rate required to represent the contour information at a reasonable level, the segmentation is approximated by a finite set of different patterns, and this information is coded by a VQ technique. Hence, the algorithm provides higher resolution motion field without significantly increasing the overhead information. Consequently, the use of the VQ-based segmentation technique in a video coding scheme leads to a globally decreased bit rate as well as a higher visual quality.

The chapter is structured as follows. The motion field segmentation technique is described in Sec. 6.2. The VQ-based segmentation algorithm, simplifying hypotheses, and the criterion to select blocks to be segmented are discussed. Simulation results are presented in Sec. 6.3. Finally, Sec. 6.4 draws the conclusions.

6.2 Segmentation of the motion field

In this section, the VQ-based segmentation technique is described in more details. The method starts from a block-based motion field initially evaluated by a block matching motion estimation technique (or more generally by any block-based motion estimation algorithm). By segmenting blocks for which the block-based motion model fails, one improves the accuracy of the corresponding motion vectors.

More precisely, blocks corresponding to moving edges are selected and undergo a segmentation in N regions ($N = 2, 3, \ldots, N_{\max}$), where a region means a set of connected pixels. A motion vector is assigned to each segmented region, this motion vector being chosen among the ensemble Ω of the motion vectors in the neighborhood of the considered block. Two non-adjacent regions may share the same motion vector. The segmentation pattern defined by the association of each pixel to one motion vector is coded by means of a VQ technique and transmitted as overhead information. The upper limit of the num-

ber of segmented regions in each block, the strategy to define the neighborhood and the constraints on the segmented regions have to be chosen in order to balance the amount of side information, the computational requirements as well as quality enhancement of the motion field. Furthermore, a criterion is required to decide whether to segment a block.

6.2.1 VQ-based segmentation

A *vector quantizer* is defined as the mapping of input data vectors into a discrete set of possible output vectors or *codevectors* belonging to a previously generated *codebook* [151, 152]. The codevector is chosen such as to provide the best match with the input vector using a minimum distortion rule. The codebook is commonly generated by using a training set representative of the data to be encoded (a well-known algorithm for this procedure is the LBG algorithm [153]). The advantage of the VQ technique, when compared to a *scalar quantizer* (SQ), lies in its capability to exploit the dimensionality of the data vectors. Key elements for high performances are the codebook generation, the codebook size and the vector size.

In the proposed VQ based segmentation algorithm, the codevectors are by definition different segmentation patterns approximating the motion field boundaries. In other words, the codevectors $\vec{c}$ represent the mapping of each pixel to the different motion vectors in the set Ω. The segmentation patterns are thus restricted to the codevectors of the codebook. Depending on the number of segmented regions N, different codebooks Γ_N are considered, where Γ_N includes γ_N codevectors.

Given a certain N, the segmentation assigning for each of the N regions in a block a displacement vector is expressed by the following mapping:

$$\vec{d}(\vec{c},\vec{r}) \mapsto \begin{cases} \vec{d}_1 & \text{if } \vec{r} \in \text{region } 1 \\ \vec{d}_2 & \text{if } \vec{r} \in \text{region } 2 \\ \vdots & \\ \vec{d}_N & \text{if } \vec{r} \in \text{region } N \ , \end{cases} \tag{6.1}$$

where

$$\vec{c} \in \Gamma_N \ , \tag{6.2}$$

and

$$\vec{d}_1, \vec{d}_2, \ldots, \vec{d}_N \in \Omega \ . \tag{6.3}$$

In the segmentation process, the codevector providing the best match in the sense of the block matching measure (in our case MAE) is selected. In other words, the segmentation of a block in N regions is carried out by minimizing

$$\min_{\vec{c} \in \Gamma_N,\ \vec{d}_1,\ \vec{d}_2,\ ...,\ \vec{d}_N \in \Omega} \left(\sum_{\vec{r} \in \mathcal{W}} \| I(\vec{r}, t) - I(\vec{r} - \vec{d}(\vec{c}, \vec{r}), t - \Delta t) \| \right) . \tag{6.4}$$

This is accomplished through an exhaustive search of all the elements of the codebook and all the combinations of motion vectors.

The segmentation information is transmitted to the decoder by sending the *codeword* corresponding to the respective pattern. The resulting number of overhead bits per segmented block includes the segmentation pattern, the number of segmented regions and the motion vectors corresponding to the segmented regions. More precisely, the amount of information to transmit the segmentation pattern depends on the codebook size and is given by

$$[\log_2(\gamma_N)] , \tag{6.5}$$

with $[x]$ the smallest integer greater than x. The information to indicate for each block the number of segmented regions ($N = 1$ i.e. no segmentation or $N = 2, \ldots, N_{\max}$) is expressed in bits by

$$[\log_2(N_{\max})] . \tag{6.6}$$

Assuming that the block-based motion vectors (i.e. the elements of Ω) have already been transmitted, the N motion vectors assigned to the respective N regions are specified by indicating which elements of Ω have been selected. If Ω includes ω elements, it represents a coding cost (in bits) of

$$[\log_2(\omega^N)] . \tag{6.7}$$

Consequently, for each block (segmented or not), the side information due to the algorithm is given by

$$R_{\text{side}} = [\log_2(N_{\max})] \quad [\text{bit}] , \tag{6.8}$$

whereas for each segmented blocks, the additional information to represent the segmentation information (for a given N) is defined by

$$R_{\text{seg}}^{(N)} = [\log_2(\gamma_N)] + [\log_2(\omega^N)] \quad [\text{bit}] . \tag{6.9}$$

6.2.2 Efficient implementation under realistic hypotheses

Under realistic hypotheses, the above segmentation method can be simplified to be efficiently implementable in a video coding system.

The basic hypothesis in a block matching motion estimation technique is that the block size is small compared to the size of the objects present in the scene. This hypothesis justifies the block-based motion field model which assigns a single motion vector to a block of pixels. Nevertheless, the model is not valid when an object boundary lies within a block. However, this hypothesis introduces the following simplifications in the method proposed in Sec. 6.2.1.

- The block segmentation is limited to two regions, i.e. $N_{max} = 2$. On the one hand only a few blocks contain actually more than two regions. On the other hand, the segmentation of a block in three or more regions leads to a large number of different patterns, in other words to a large codebook. Therefore it requires a large amount of overhead information as well as a high computational complexity. Furthermore, as the number of segmented regions for each block has always to be specified to the decoder (see Eq. (6.8)), it globally increases the overhead information.

- The elements of the motion vectors set Ω are defined as the initial motion vector obtained for the considered block as well as the ones obtained for 4-connected blocks. With the 4-connectivity choice, Ω contains five elements, i.e. $\omega = 5$ (see Fig. 6.1). The fact of considering in the set Ω global vectors obtained blockwise, without taking into account possible segmentation of surrounding blocks, is motivated by the hypothesis that objects are large compared to block size. Therefore, it is assumed that among 4-connected blocks at least one of them belongs completely to the same physical object as the segmented region and provides an efficient motion vector for this region. The last remark, as well as the desire to keep a low overhead information, justifies the choice of 4-connectivity rather than 8-connectivity.

- As far as the codebook is concerned, a common strategy to built it is to evaluate the statistical distribution of the different patterns on a training set representative of the motion field segmentation in natural scenes. The efficiency of this method depends on the size, as well as on the statistical relevance of the resulting codebook. To avoid training, the generation of synthetic codebooks has been preferred. The boundaries separating the two regions is limited, for instance to straight lines or splines (more precisely to the approximation of lines or splines on a discrete sampling grid). To further decrease the size of the resulting codebooks, less relevant or less frequent patterns can be discarded. Two codebooks have been designed, the first one constraints the boundary between two regions to a straight line (Fig. 6.2 illustrates some codevectors of this codebook), whereas the second one constraints the latter to a second degree polynomial.

Given the above assumptions, the general equations in Sec. 6.2.1 can be simplified. Eq. (6.1) becomes

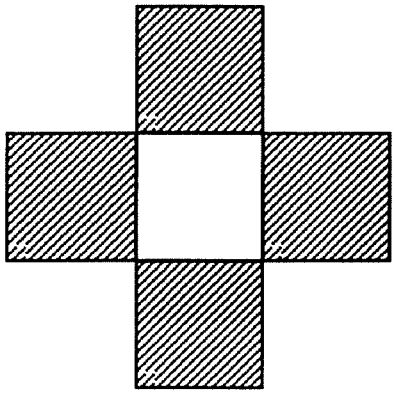

Figure 6.1: A block and its 4-connected blocks (indicated with stripes) defining the set Ω.

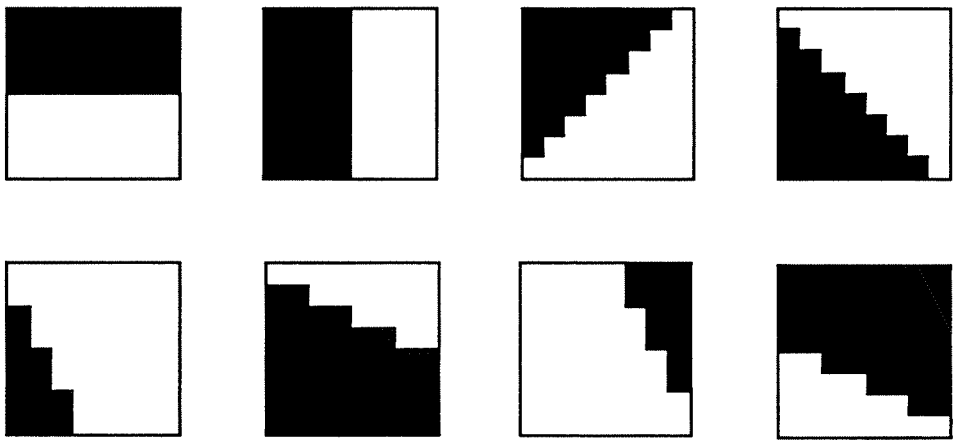

Figure 6.2: Example of codevectors of the codebook constraining the boundary to straight line.

$$\vec{d}(\vec{c},\vec{r}) \mapsto \begin{cases} \vec{d}_1 & \text{if } \vec{r} \in \text{region 1} \\ \vec{d}_2 & \text{if } \vec{r} \in \text{region 2} , \end{cases} \tag{6.10}$$

where $\vec{c} \in \Gamma$ and the codebook Γ, which contains γ codevectors, defines the set of available segmentation patterns with two connected regions. The displacement vector of each region is determined by Eq. (6.4). The latter is simplified to become

$$\min_{\vec{c} \in \Gamma,\ \vec{d}_1,\vec{d}_2 \in \Omega} \left(\sum_{\vec{r} \in \mathcal{W}} \| u(\vec{r},t) - u(\vec{r} - \vec{d}(\vec{c},\vec{r}), t - \Delta t) \| \right) . \tag{6.11}$$

With the above hypotheses, the overhead information due to the algorithm is deduced from Eqs. (6.8) and (6.9).

As $\vec{d}_1$ and $\vec{d}_2$ have to be different (otherwise there would be no need to segment), Eq. (6.7) can be written

$$[\log_2(\omega \cdot (\omega - 1))] . \tag{6.12}$$

Given $N_{max} = 2$ and $\omega = 5$, Eqs. (6.8) and (6.9) become

$$R_{\text{side}} = 1 \quad [\text{bit}] , \tag{6.13}$$

and

$$R_{\text{seg}}^{(2)} = [\log_2(\gamma)] + 5 \quad [\text{bit}] . \tag{6.14}$$

Figure 6.3 illustrates the limit of the block-based motion estimation techniques whenever a block encompasses regions moving in two different directions. Figure 6.4 illustrates the result of the above simplified VQ-based segmentation applied to the previous case. In this illustration, the proposed segmentation reaches the optimal solution, even with the simplifications introduced.

6.2.3 Segmentation decision rule

The above algorithm is designed to segment blocks for which the block-based motion model fails, in other words those corresponding to moving edges. Furthermore, segmentation has to be controlled in order to limit the overhead information. This problem shares similarities with the splitting decision rule in the locally adaptive multigrid algorithm (see Chap. 5)

A simple criterion to control the segmentation is as follows. Blocks with a DFD energy (actually a matching error) higher than a preset threshold are assumed to contain boundaries of moving objects. Consequently, they are selected for segmentation. Hence, the criterion to decide whether to segment a block is:

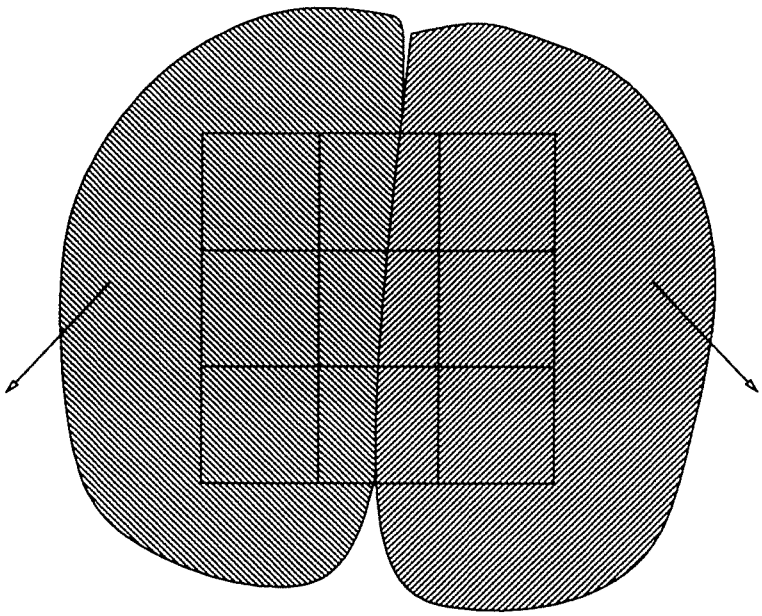

Figure 6.3: Limitation of the block-based motion estimation when the boundary between two objects (indicated with stripes) moving in different directions (indicated with arrows) lies inside blocks.

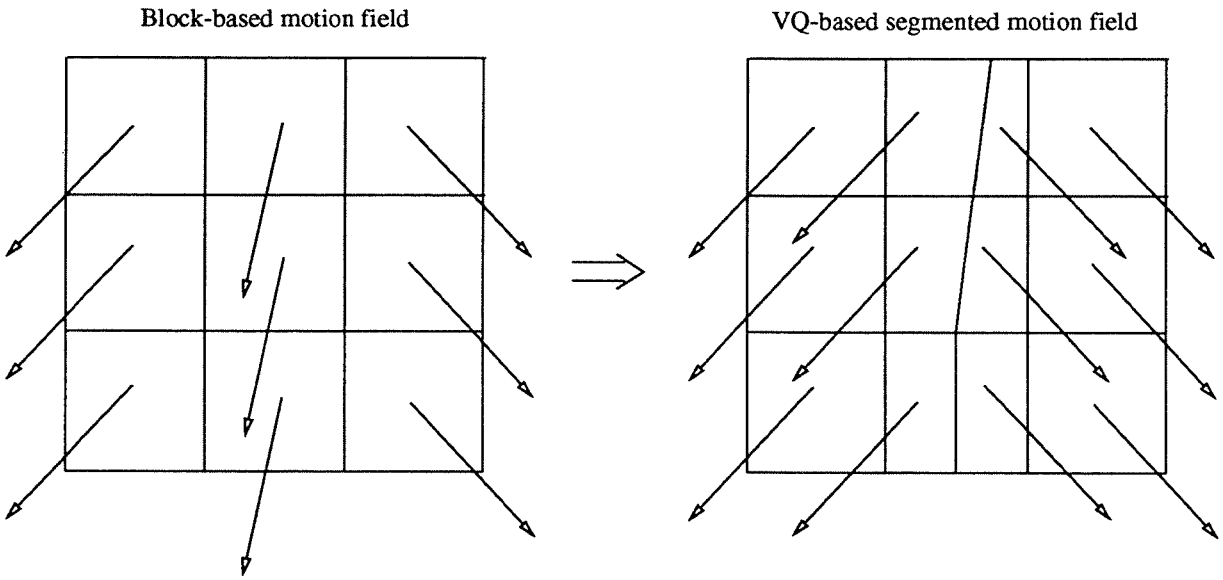

Figure 6.4: Initial block-based motion field and after segmentation.

$$\mathrm{MAE}_{\mathrm{noseg}} > T \Rightarrow \text{segmentation} , \tag{6.15}$$

where T is the threshold and $\mathrm{MAE}_{\mathrm{noseg}}$ is the mean absolute error of the DFD block without segmentation. The criterion defined by Eq. (6.15) is identical to the one defined by Eq. (5.1). Consequently, it has the same drawback: to select an adequate value of the threshold. Furthermore, the criterion does not guarantee that the gain on the bit rate obtained thanks to the segmentation is worth the extra cost to transmit the segmentation information. As already mentioned, the above problem is addressed in Chap. 7. Nevertheless, the criterion defined by Eq. (6.15) is used for the remaining of the chapter. In practice, the threshold is set in percent of segmented blocks.

6.3 Simulation results

In order to evaluate the performances of the segmentation algorithm previously described, simulation results are presented in this section. Performances are evaluated in terms of the DFD energy, as well as bit rate versus PSNR. Experiments have been carried out on the three sequences "Mobile Calendar", "Table Tennis" and "Flower Garden". Due to the high computational complexity of the VQ-based segmentation algorithm, the CIF format (see Table 3.1) has been preferred to the CCIR 601.

The VQ-based segmentation algorithm could be used to segment motion fields resulting from either the multigrid or the locally adaptive multigrid block matching techniques introduced in Chap. 4 and 5 respectively. However, in order to remain very general motion field processed by the classical full-search block matching algorithm are considered in this section.

With the CIF format, the highest performances are achieved with a block size of 16×16 pixels (instead of 8×8 pixels previously with the CCIR 601 format). In the next simulations, the maximum displacement of the full-search block matching technique is ± 15 pixels with one pixel accuracy.

Two codebooks have been artificially designed. The first one constraints the boundary between two regions to a straight line, whereas the second one constraints the latter to a second degree polynomial. The resulting codebooks contain 1275 and 10226 codevectors respectively, and are referred to as the line-codebook and the polynomial-codebook respectively. Figure 6.5 shows all the codevectors of the line-codebook, each block corresponding to a distinct codevectors. Figure 6.6 illustrates similarly some of the codevectors of the polynomial-codebook.

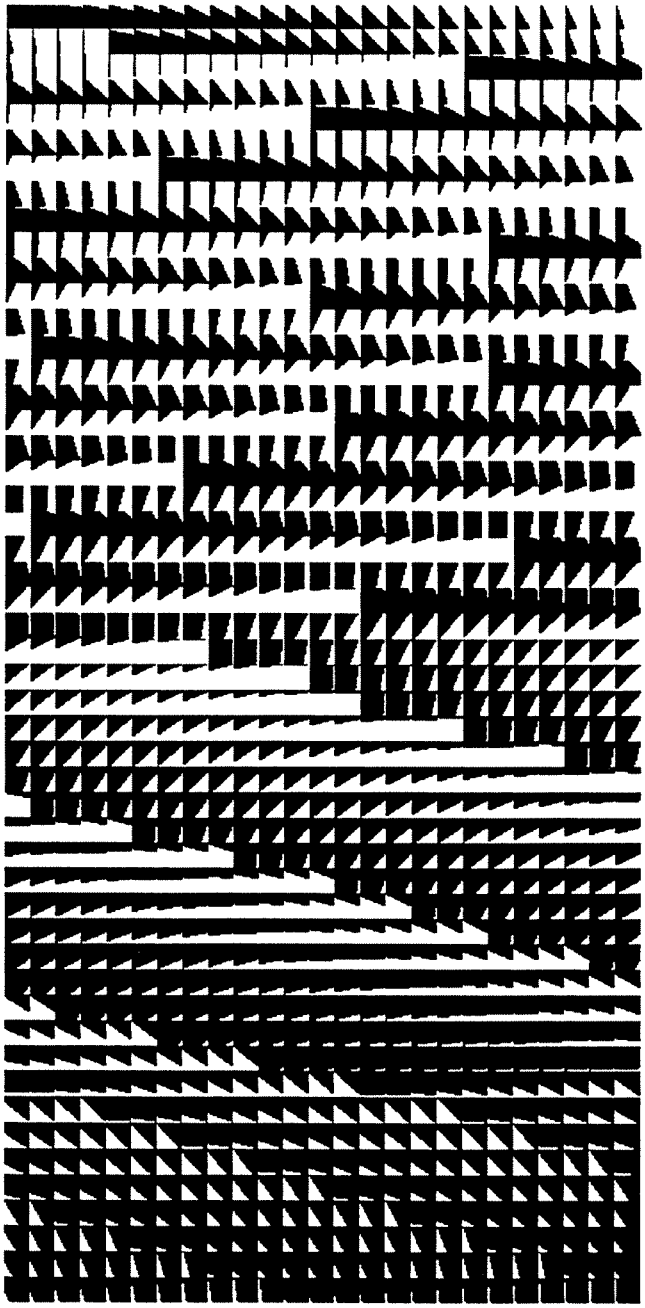

Figure 6.5: All the codevectors in the line-codebook.

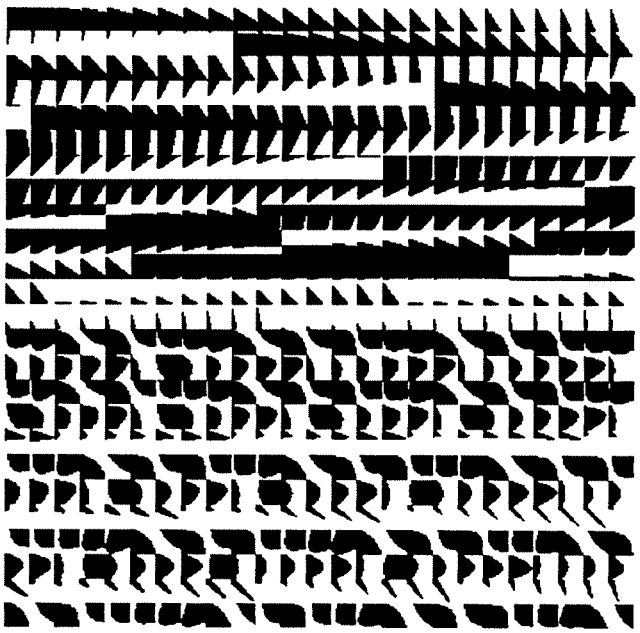

Figure 6.6: Some codevectors in the polynomial-codebook.

6.3.1 Comparison between the VQ-based segmentation and the full-search block matching in terms of DFD energy

Figure 6.7 compares the full-search block matching and the VQ-based segmentation algorithm in terms of DFD energy. Both the line-codebook and the polynomial-codebook are used with different thresholds: line-codebook and threshold of 10%, 20% or 30%, polynomial-codebook and threshold of 10%.

The segmentation algorithm outperforms significantly the full-search block matching. The gain is as high as 15% for "Mobile Calendar", 35% for "Table Tennis" and 50% for "Flower Garden". The polynomial-codebook does not further improve the performances when compared to the line-codebook, whereas it introduces a much larger complexity and a larger overhead information. Consequently, it has been discarded in the next experiments. As far as the threshold is concerned, the increase of the percentage of blocks selected for segmentation improves logically the performances. However, the gain is obtained at a cost of an increased overhead information and computation time.

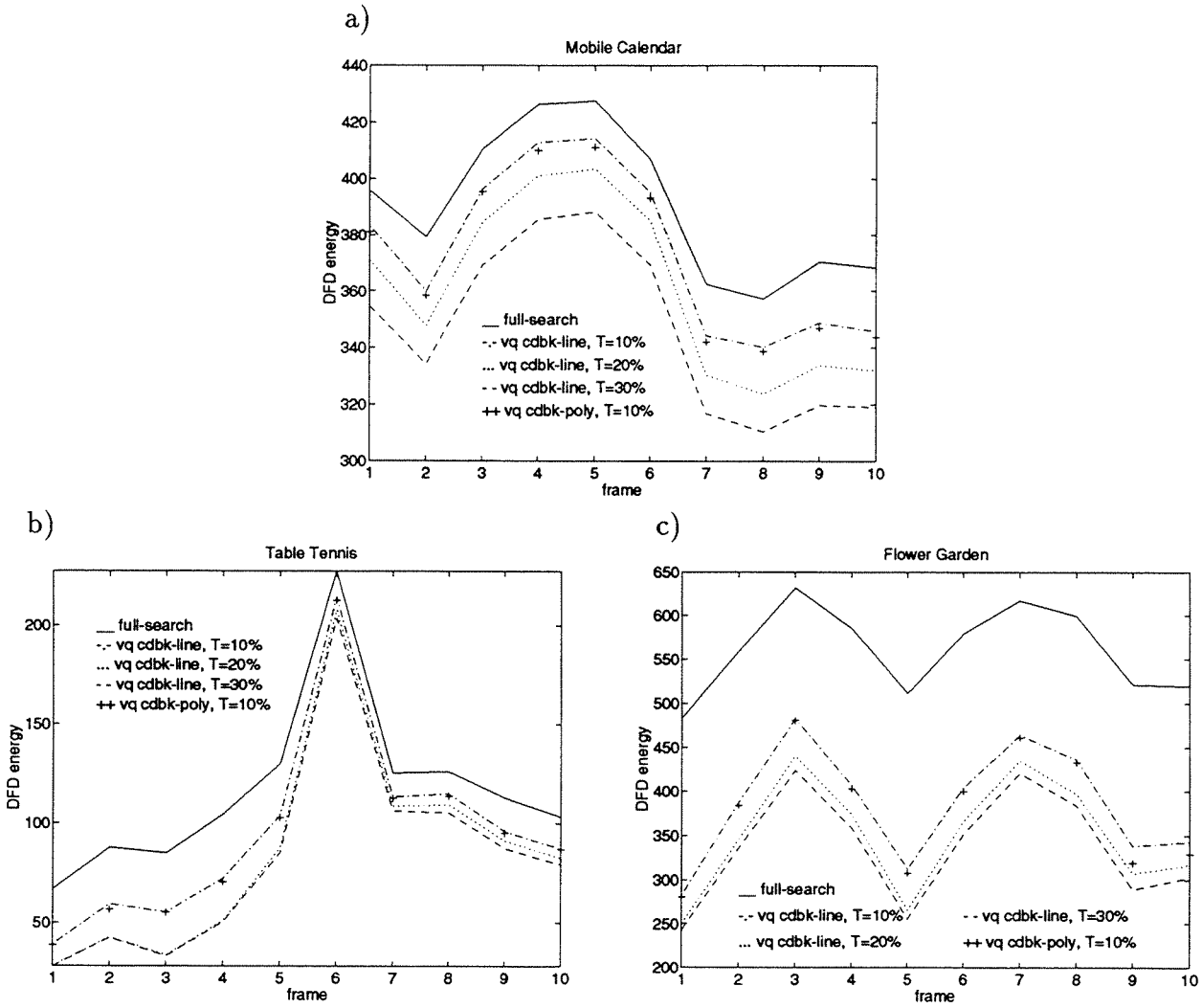

Figure 6.7: DFD energy: comparison between full-search and VQ segmentation with either the line-codebook and T=10%, 20%, 30% or the polynomial-codebook and T=10%, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

6.3.2 Comparison between the VQ-based segmentation and the full-search block matching in terms of visual quality

Figure 6.8 illustrates the VQ-based segmentation for the sequence "Table Tennis". It shows the blocks segmented with a threshold of $T = 20\%$, as well as the motion field segmentation obtained with the proposed VQ-based algorithm. The segmented blocks correspond clearly to the moving edges: the bat, the player's arm and the ball. Moreover, one can clearly notice that the segmentation of the motion field corresponds to the segmentation of the moving edges. Blocks on the left and bottom borders have also been segmented, due to the interpretation of the image border as an object boundary in these areas.

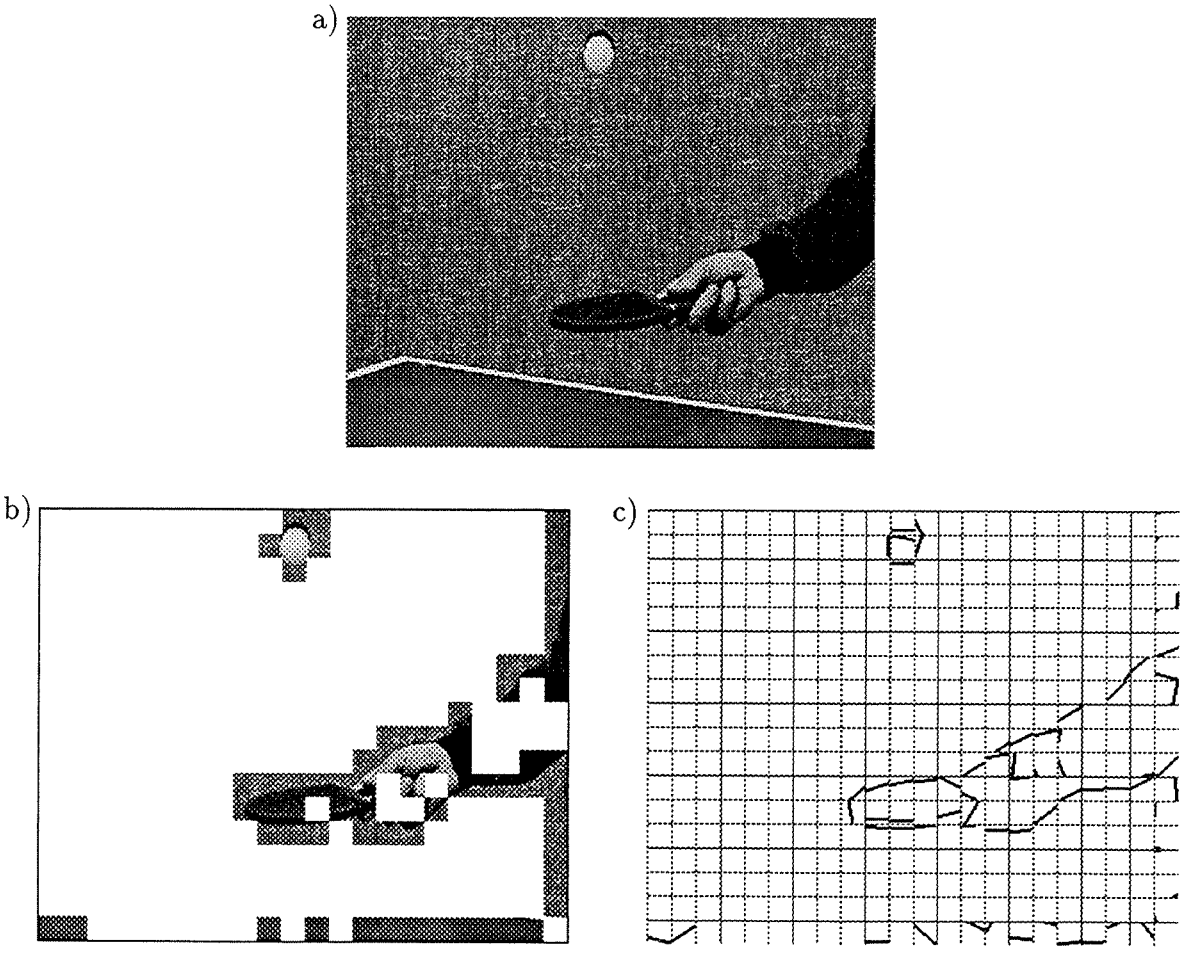

Figure 6.8: A frame of a) "Table Tennis", b) the corresponding mask of segmented blocks and c) the corresponding motion field segmentation.

From the above results, a higher visual quality is expected while using the VQ-based

motion field segmentation algorithm. Figure 6.9 shows an original portion of the sequence "Table Tennis", as well as the motion compensated prediction obtained while using full-search block matching motion estimation and the proposed segmentation algorithm. Whereas Fig. 6.9.b exhibits strong block artifacts, especially on the edges of the bat and the hand, in Fig. 6.9.c these block artifacts have been removed and boundaries are smooth and sharp, providing a more accurate prediction of the original frame. Consequently a greatly higher visual quality of the reconstructed sequence is obtained.

a) b) c)

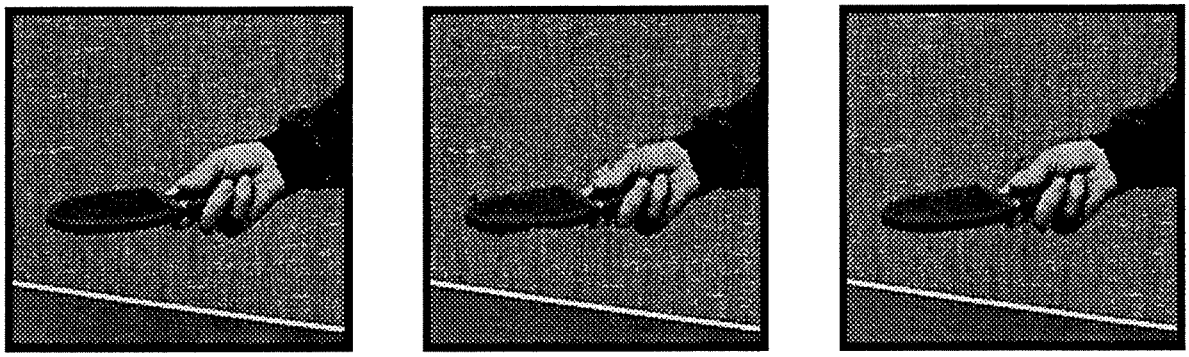

Figure 6.9: "Table Tennis": a) original, b) motion compensated prediction using full-search block matching, c) motion compensated prediction using the proposed segmentation algorithm.

6.3.3 Comparison between the VQ-based segmentation and the full-search block matching in terms of bit rate and PSNR

The next simulations have been carried out with the three coding schemes $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ introduced in Chap. 3. Figures 6.10, 6.11 and 6.12 show the bit rate versus PSNR characteristics when the motion estimation is estimated either by full-search block matching or by the proposed VQ-based segmentation technique. The line-codebook is used and the threshold has been set at its optimal value after several trials, resulting in a threshold of $T = 50\%, 35\%$ and 40% for the sequences "Mobile Calendar", "Table Tennis" and "Flower Garden" respectively. The different threshold values are explained by the very different content of the three scenes.

The comparison is made based on the bit rate and PSNR, even though the latter is a poor measure of the visual quality of an image sequence. The human visual system is indeed extremely sensitive to degradation along edges [148], although they form only a

small part of the entire image. As the proposed motion estimation technique improves the compensation along moving edges, it achieves a significantly higher visual quality, but leads only to a small increase of the PSNR.

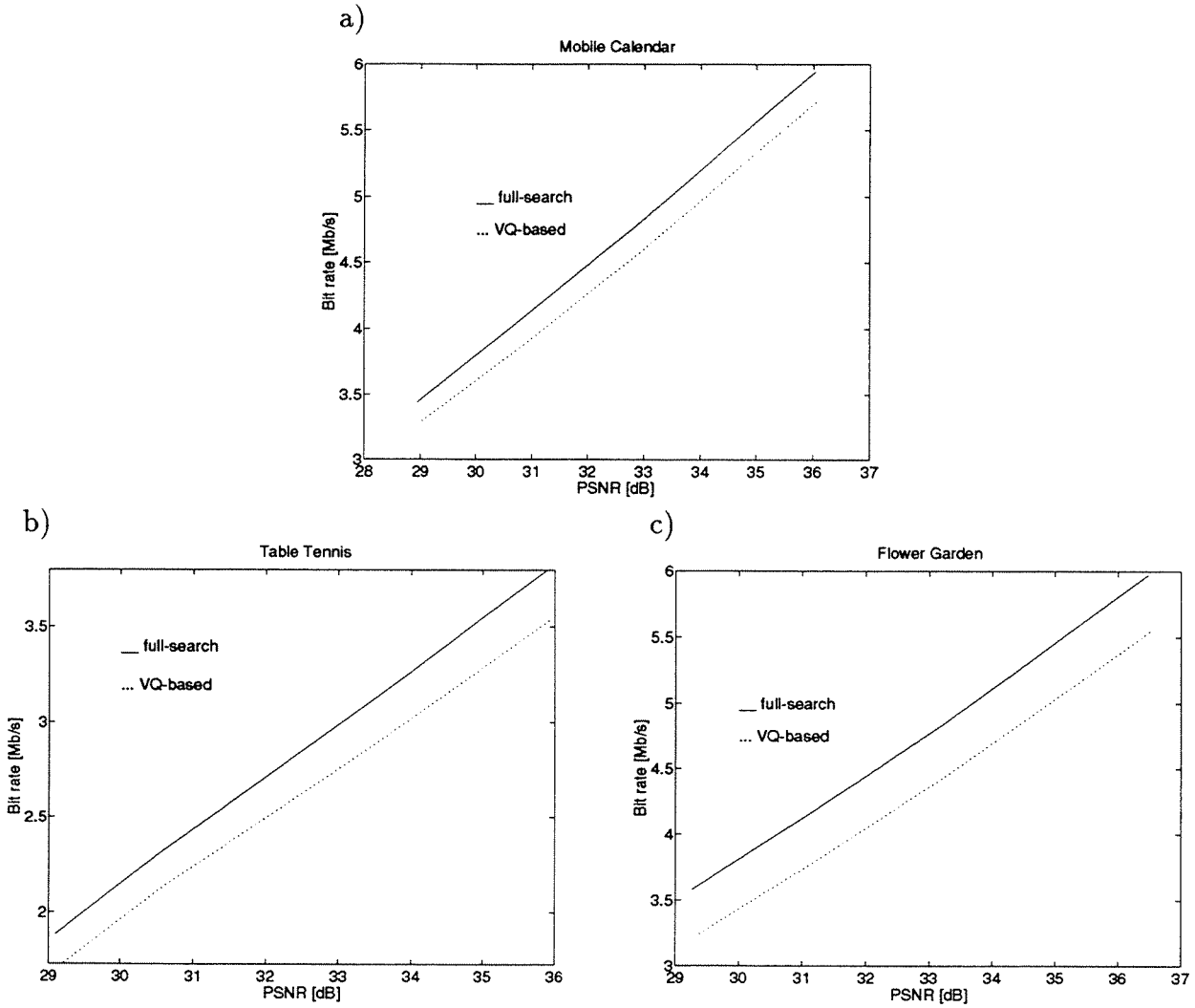

Figure 6.10: Bit rate versus PSNR for the scheme $\mathcal{A}$: comparison between full-search and VQ-based segmentation algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

The following conclusions can be drawn from the above results. For the three schemes and the three sequences, the VQ-based segmentation technique significantly outperforms the full-search block matching. The gain is the most significant for the sequences "Flower Garden" and "Table Tennis", and for the schemes $\mathcal{B}$ and $\mathcal{C}$. The saving in terms of bit rate is ranging from 0.4 to 0.5 Mb/s (7% to 15 %) for "Flower Garden", 0.2 to 0.3 Mb/s (6%

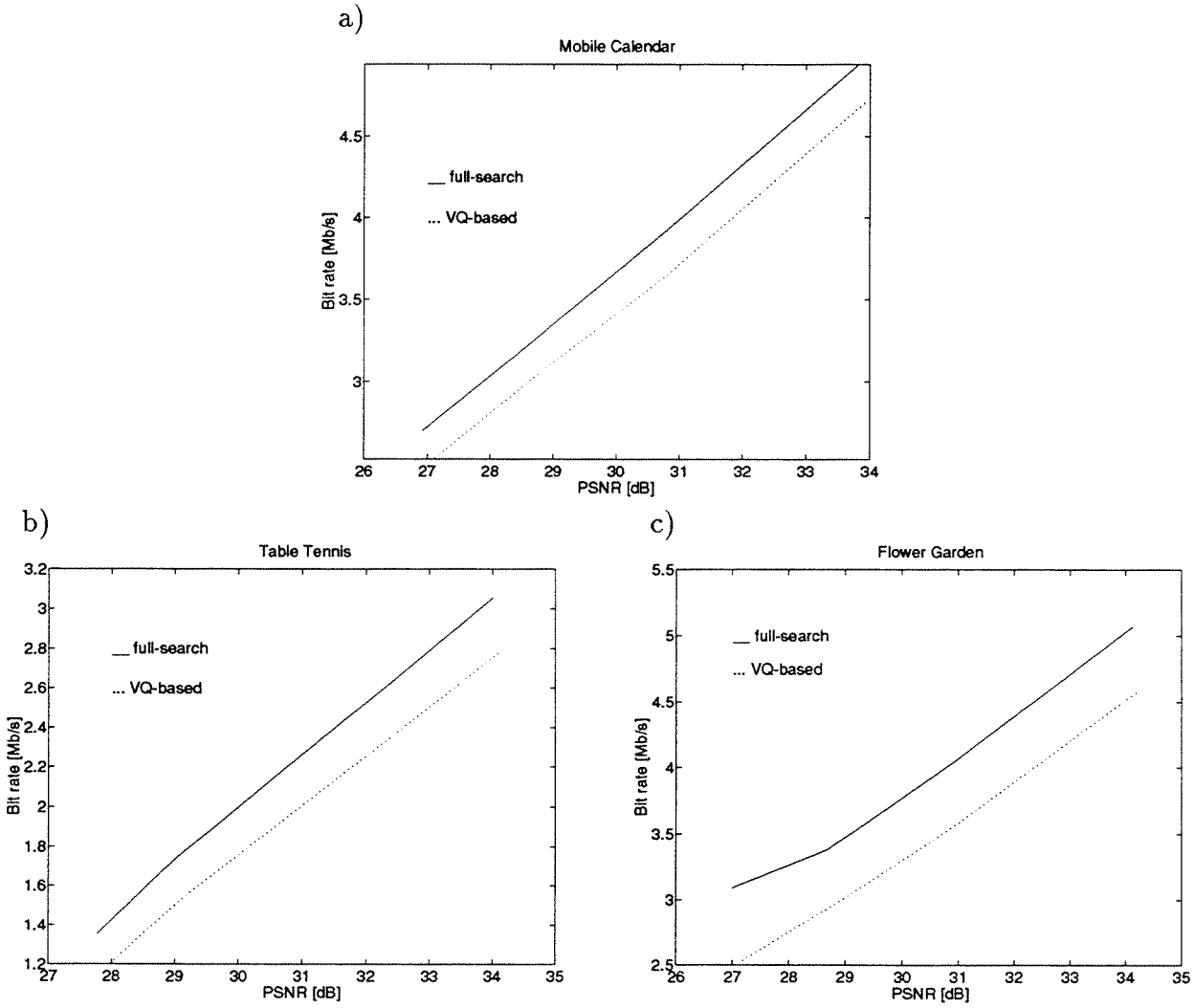

Figure 6.11: Bit rate versus PSNR for the scheme $\mathcal{B}$: comparison between full-search and VQ-based segmentation algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

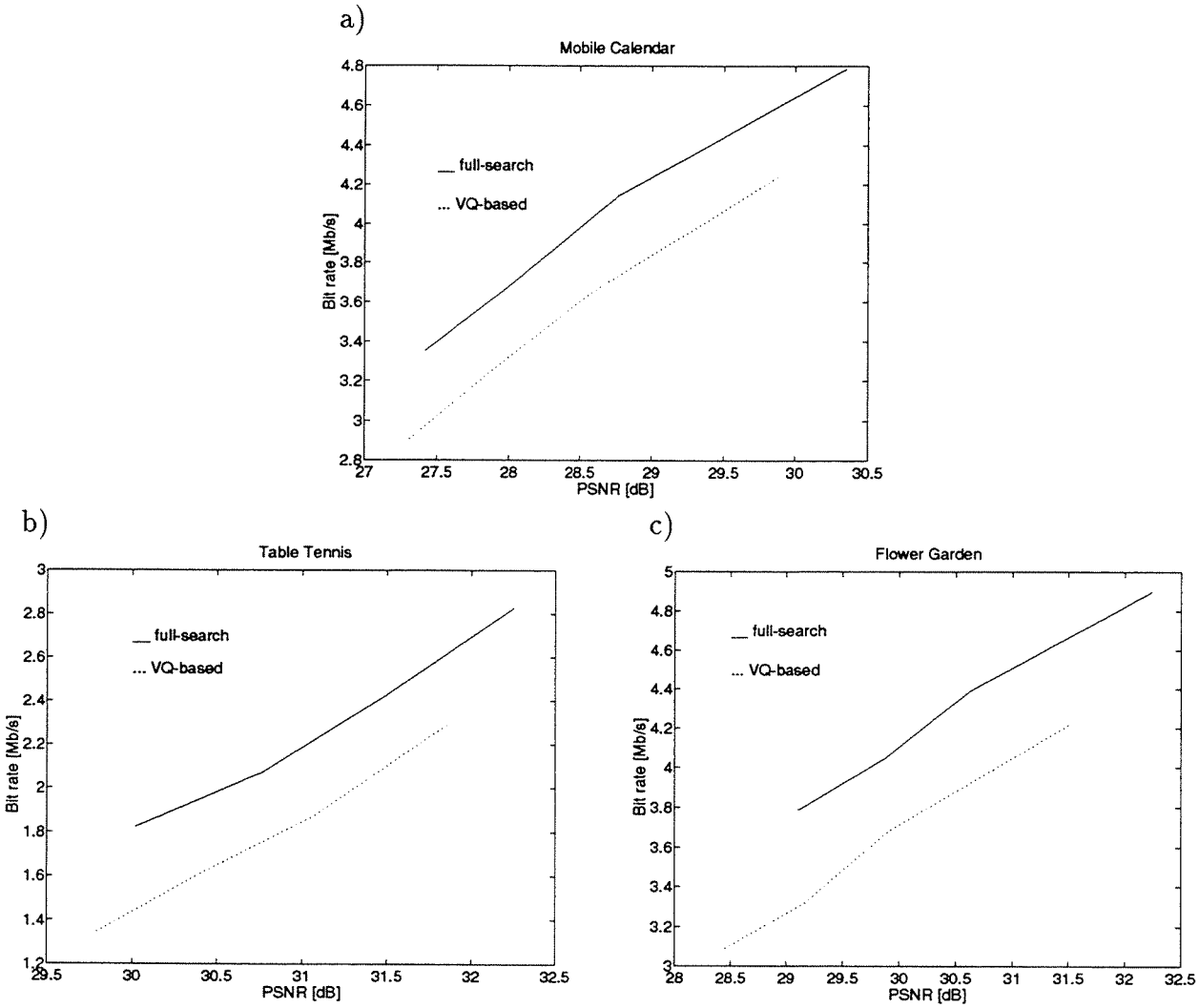

Figure 6.12: Bit rate versus PSNR for the scheme $\mathcal{C}$: comparison between full-search and VQ-based segmentation algorithms, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

to 20%) for "Table Tennis", and 0.25 to 0.4 Mb/s (5% to 13%) for "Mobile Calendar". It corresponds to a gain in terms of PSNR ranging from 0.5 to 1.5 dB. Furthermore, the visual quality of the reconstructed sequences is greatly enhanced by the segmentation algorithm, in particular moving edges are sharper. Consequently, the efficiency of the VQ-based segmentation technique to improve the motion estimation is clearly shown.

6.4 Summary

In this chapter, a technique to segment motion fields has been introduced. It overcomes the problems related to the block-based nature of the block matching motion estimation techniques. The method segments, by means of a VQ technique, blocks corresponding to moving edges for which failures of the block-based motion model appear. Thus, the motion field accuracy is increased in these important regions. Therefore, the segmentation technique reduces the bit rate to code the DFD and leads to higher visual quality of the decoded sequence by preserving the sharpness of moving edges. Besides, the VQ technique is efficient to represent and code the segmentation pattern. Consequently, the overhead information is not significantly increased.

Simulations have been carried out with various sequences and coding schemes. They have demonstrated the ability of the method to significantly improve the motion estimation technique performances. Saving ranging from 0.2 to 0.5 Mb/s (5% to 20%) in terms of bit rate, or gain ranging from 0.5 to 1.5 dB in terms of PSNR have been achieved. Moreover, reconstructed sequences exhibit greatly enhanced visual quality.

Chapter 7

Entropy criterion to optimize motion compensation

7.1 Introduction

As in video coding applications the motion information has to be transmitted to the decoder, the motion estimation algorithms developed in this field have to be distinguished from those used in applications such as the analysis, interpolation or restoration of image sequences. Whereas in the latter cases the motion estimation techniques aim at estimating in the most precise fashion the motion parameters, in the former case the amount of both motion and prediction error information have to be taken into account. It is straightforward that a very precise motion estimation leads on the one hand to a low coding cost of the DFD, but on the other hand to a high overhead motion information. Conversely, a coarse motion estimation requires a low side information, but entails a poor motion compensated prediction. Therefore, a trade-off between the two terms has to be found. Little research effort has been devoted to this subject in the literature.

This chapter introduces an approach to achieve an optimal bit allocation between respectively the motion and the DFD components [154]. The optimum is reached through the minimization of the sum of the corresponding transmission costs. Therefore, the use of the method in a coding scheme achieves a minimization of the total bit rate. It is applicable to a wide variety of motion estimation algorithms and to any classical motion compensated coding scheme. Furthermore, the method avoids the setting of parameters controlling the motion estimation algorithm. In particular, it provides an efficient procedure to control the segmentation in both the locally adaptive multigrid block matching motion estimation (see Chap. 5) and the VQ-based segmentation of the motion field (see Chap. 6), overcoming the problem of a threshold setting.

The global bit rate optimization procedure demands an estimation of the transmission cost for both the DFD and the motion components. First, the low spatial correlation of the DFD pixels permits to consider it as a 0th-order Markov process. Furthermore, by the very nature of the prediction process, the DFD exhibits a characteristic distribution which allows its modeling as a Laplacian probability density function (PDF). Hence an analytical expression to estimate the entropy of the DFD, namely its coding cost, can be derived. With regard to motion information, its cost is most of the time straightforward and computationally easy to estimate. Therefore, the minimization of the sum of the coding costs corresponding to the two components reaches the optimal trade-off. This minimization defines the so-called entropy criterion.

The chapter is structured as follows. The need to control the motion estimation as well as examples of applications are discussed in Sec. 7.2, introducing the entropy criterion. The statistical modeling of the DFD is studied in Sec. 7.3. The application of the entropy criterion to the locally adaptive multigrid block matching motion estimation is described in Sec. 7.4, and to the VQ-based segmentation of the motion field in Sec. 7.5. Simulation

results are given to show the improvement due to the use of this criterion. Finally, Sec. 7.6 draws the conclusions.

7.2 Control of the motion estimation - entropy criterion

An important and unsolved question in the field of video coding is the appropriate accuracy of the motion field. Even though it is straightforward to improve the motion compensated prediction, it does not mean that the gain obtained on the DFD side is worth the cost of extra side information. Therefore, to achieve an overall gain in terms of coding performances is a much more challenging problem.

More precisely, in most motion estimation algorithms parameters have to be set in order to control the estimation process. Numerous examples can be found. For instance, these parameters could be a threshold in a segmentation procedure, the density of the motion vectors (e.g. the block size in a block matching technique), the precision of the motion vectors (e.g. 1 or 1/2 pixel accuracy), or the choice of the motion model (e.g. translational, affine, ...). In all the above examples, these parameters decide the bit rate sharing between the DFD component on the one side and the motion information (including the segmentation if any) on the other side.

The proposed method minimizes the global bit rate R as a trade-off between the amount of motion parameters R_{motion} and the DFD information R_{DFD}. In order to obtain a global optimization of the coding scheme, an expression of the total bit rate

$$R = R_{\text{motion}} + R_{\text{DFD}} \ , \tag{7.1}$$

is introduced. The minimization procedure demands an estimate of the two terms of the right hand side in Eq. (7.1). The difficulty lies undoubtedly in the latter operation.

As far as R_{motion} is concerned, it corresponds to the whole motion information. Hence, it includes the motion vectors, e.g. translation or rotation, as well as the segmentation if any. Most generally, the transmission cost of the motion vectors can be evaluated by the mere computation of their entropy. Concerning the bit rate required to represent the segmentation, it depends on the specific motion estimation algorithm.

With regard to the DFD transmission cost, it can be estimated as follow. Assuming an entropy coding, the DFD bit rate is given by

$$R_{\text{DFD}} = n_{\text{DFD}} \cdot H_{\text{DFD}} \ , \tag{7.2}$$

where n_{DFD} is the number of considered pixels and H_{DFD} their entropy.

The entropy H_{DFD} remains to be evaluated. For this purpose a source model is developed in Sec. 7.3. Ideally, a high order statistical model should be used. In this case, Eq. (7.2) remains valid when a transform coding technique is applied to the DFD (as in MPEG-I [5, 30], MPEG-II [6, 31], H.261 [7] or in the scheme $\mathcal{A}$ introduced in Chap. 3). Nevertheless, observations have shown that the correlation is very low in the DFD [32, 33]. Consequently, in practice a 0th- or 1st-order model to compute the entropy provides a good approximation of the DFD coding cost. Furthermore, when the DFD is directly entropy coded without transform (as in the scheme $\mathcal{B}$, see Chap. 3) the order of the statistical model used to estimate the entropy in Eq. (7.2) and the order of the entropy coder applied in the coding scheme should be equal in order to be consistent.

Once the transmission cost of the different components composing the bit rate have been estimated, the entropy criterion can be defined as follow.

- If $\vec{n_1}$ and $\vec{n_2}$ correspond to two parameters states of the motion estimation algorithm, the one providing the lower total bit rate is preferred.

$$\begin{aligned} R_{\mathrm{motion}}(\vec{n_2}) + R_{\mathrm{DFD}}(\vec{n_2}) < R_{\mathrm{motion}}(\vec{n_1}) + R_{\mathrm{DFD}}(\vec{n_1}) \\ \Rightarrow \vec{n_2} \text{ is preferred to } \vec{n_1} \ . \end{aligned} \tag{7.3}$$

In the locally adaptive multigrid block matching algorithm (see Chap. 5), $\vec{n_1}$ and $\vec{n_2}$ correspond to the situations without and with splitting respectively, whereas in the VQ-based motion field segmentation technique (see Chap. 6), they correspond to the cases without and with segmentation by VQ respectively.

The above criterion minimizes the bit rate by optimizing the motion estimation procedure. As far as the visual quality is concerned, the following remarks can be done. First, the motion estimation improvement is likely to lead to a large gain on moving edges which correspond to regions difficult to predict accurately. Therefore the criterion is accepting more precise motion parameters in these visually important regions. Second, the bit rate saving due to the criterion can be used to enhance the visual quality of the reconstructed sequence (e.g. by decreasing the quantization step size in the DFD coding).

7.3 Statistical model of the DFD

7.3.1 Basic definitions

In order to estimate the DFD entropy, in other words its coding cost, a statistical model has to be defined. To take into account the correlation between pixels, a high order

statistical model should be used. Hence, the *nth-order entropy* is defined as [17]

$$H_n = -\sum_{i0,i1,\ldots,in} p(x_{i0}, x_{i1}, \ldots, x_{in}) \cdot \log_2 \left(p(x_{i0}, x_{i1}, \ldots, x_{in})\right) \ , \tag{7.4}$$

where x_i denotes the symbols of the source, $p(x_{i0}, x_{i1}, \ldots, x_{in})$ expressed the joint probability of $x_{i0}, x_{i1}, \ldots, x_{in}$, and the summation is carried out over all possible n-tuples.

With the above entropy definition, Eq. (7.2) represents the lower bound of the DFD bit rate. Furthermore, it should be noted that the last affirmation remains true even when a transform is applied on the DFD.

The correlations between the image pixels is represented by the correlation matrix. The $N \times N$ *correlation matrix* is defined as [155]

$$\mathcal{R} = \begin{pmatrix} r_{0,0} & r_{0,1} & \cdots & r_{0,N-1} \\ r_{1,0} & r_{1,1} & \cdots & r_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ r_{N-1,0} & r_{N-1,1} & \cdots & r_{N-1,N-1} \end{pmatrix} \ , \tag{7.5}$$

where

$$r_{i,j} = E[I(i) \cdot I(j)] \ . \tag{7.6}$$

In the above equation, $E[\ ]$ is the expected value and $I(i)$ denotes the image intensity at pixel i, the image being represented in vector form.

The $N \times N$ *covariance matrix* is defined as [155]

$$\mathcal{K} = \begin{pmatrix} k_{0,0} & k_{0,1} & \cdots & k_{0,N-1} \\ k_{1,0} & k_{1,1} & \cdots & k_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ k_{N-1,0} & k_{N-1,1} & \cdots & k_{N-1,N-1} \end{pmatrix} \ , \tag{7.7}$$

where

$$k_{i,j} = E\left[\{I(i) - E[I(i)]\} \cdot \{I(j) - E[I(j)]\}\right] \ . \tag{7.8}$$

Finally, the *variance* is given by

$$\sigma_{i,i}^2 = k_{i,i} \ . \tag{7.9}$$

If the image is *wide-sense stationary* [156], the correlation is expressed as

$$r_{i,j} = r_{|i-j|} = E[I(n) \cdot I(n + |i - j|)] \ , \tag{7.10}$$

and the covariance as

$$k_{i,j} = k_{|i-j|} = r_{|i-j|} - \mu^2 \ , \tag{7.11}$$

with $\mu = E[I(n)]$. Hence, the matrices $\mathcal{R}$ and $\mathcal{K}$ becomes of Toepliz form [157]. Normalizing the correlation r_i with respect to r_0, where $r_0 = \sigma^2 = E$ represents the energy E of the image,

$$\rho_i = \frac{r_i}{r_0} \ , \tag{7.12}$$

we obtained the correlation matrix

$$\mathcal{R} = \sigma^2 \begin{pmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{N-1} \\ \rho_1 & 1 & \ddots & \ddots & \vdots \\ \rho_2 & \ddots & 1 & \ddots & \rho_2 \\ \vdots & \ddots & \ddots & \ddots & \rho_1 \\ \rho_{N-1} & \cdots & \rho_2 & \rho_1 & 1 \end{pmatrix} . \tag{7.13}$$

7.3.2 Correlation in the DFD

Experimental results reported in [32, 33] have shown that the correlation is very low in the DFDs. Table 7.1 compares typical values of the energy E and the correlations ρ_1 and ρ_2 for frames and DFDs. The measurements have been carried out on the sequences "Mobile Calendar", "Flower Garden" and "Table Tennis" in CIF format (see Table 3.1). The correlation has been computed along the lines. It appears clearly that the correlation is very high in frames ($\cong 1$) and very low in DFDs, confirming the above mentioned works.

		$\sigma^2 = E$	ρ_1	ρ_2
Mobile Calendar	frame	16201.56	0.96	0.94
	DFD	339.85	-0.11	-0.12
Table Tennis	frame	15573.33	0.99	0.99
	DFD	28.37	0.18	-0.02
Flower Garden	frame	36741.32	0.99	0.98
	DFD	227.06	0.34	-0.03

Table 7.1: Typical energy E and correlations ρ_1 and ρ_2 for frames and DFDs.

Figure 7.1 shows the correlation as a function of the pixel index, i.e. $I(m,n) \cdot I(m,n+1)$ as a function of (m,n), in a frame, respectively a DFD, of the sequence "Flower Garden". The correlation in the frame remains constantly high throughout the image. Some features of the scene, in particular the tree can be recognized. In the DFD, the correlation is

mainly zero, with few peaks of high correlation in areas where the motion model failed. Whereas the statistic is almost stationary in the frame, it is highly non-stationary in the DFD.

Two major conclusions can be drawn from the above results. First, the classical intraframe coding methods, such as transform techniques, which aim at decorrelating the pixels, performs poorly when applied to the DFDs. Segmentation-based techniques, such as those proposed in [32, 33] as well as in [131] (scheme $\mathcal{C}$, see Chap. 3), seem a promising alternative. Second, in order to characterize DFDs a high order statistical model is useless and a memoryless source model is sufficient. In particular, the 0th-order entropy gives an accurate approximation of the DFD coding cost.

7.3.3 Memoryless Laplacian model for the DFD

From the above results, it has been concluded that a memoryless source model is sufficient for the DFD. The latter remains to be defined. The Laplacian PDF

$$p(X = x) = \frac{1}{\sqrt{2}\sigma} \exp\left(-\frac{\sqrt{2}|x|}{\sigma}\right) , \tag{7.14}$$

where σ is the Laplacian standard deviation and x is a realization of the random variable X, has been shown to closely matches the measured PDF of most differential signals [17].

Figure 7.2 depicts the histograms obtained for frames and DFDs of the sequences "Mobile Calendar", "Table Tennis" and "Flower Garden". For the DFDs, both the measured PDF and the theoretical Laplacian PDF are indicated. Whereas for the frames the histograms do not show any particular distribution, for the DFDs the Laplacian PDF provides a good model of the measured PDF. Consequently, it will be applied in the following to model the DFD statistic.

7.3.4 Entropy and energy of a Laplacian PDF

Assuming a Laplacian model of the DFD, its energy E and 0th-order entropy H are given, in both the continuous and uniform quantization cases, by the following analytical formula.

DFD entropy - continuous case
The energy E and entropy H of a Laplacian PDF are respectively

$$E = \int_{-\infty}^{\infty} x^2 p(x) dx = \sigma^2 , \tag{7.15}$$

$$H = -\int_{-\infty}^{\infty} p(x) \log_2(p(x)) dx = \log_2(\sqrt{2}\sigma e) , \tag{7.16}$$

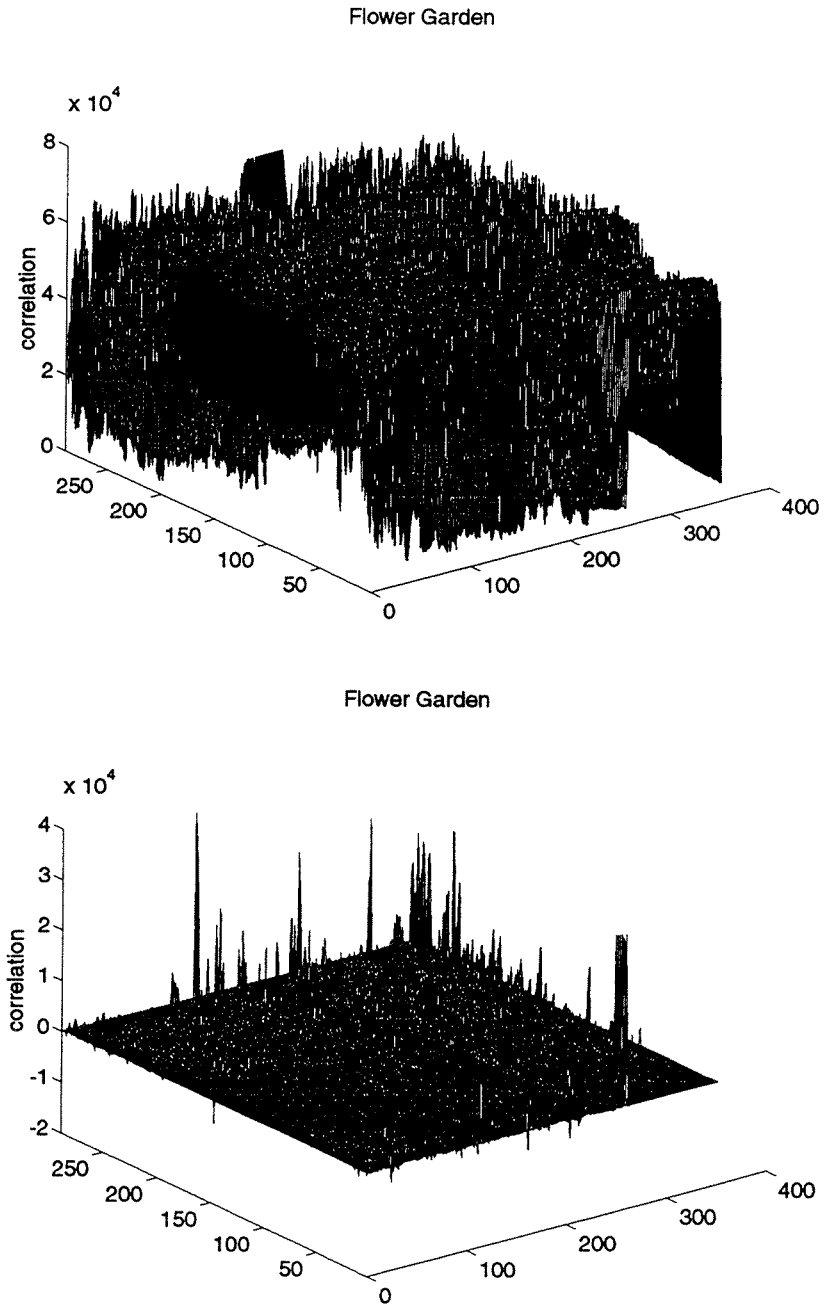

Figure 7.1: "Flower Garden", correlation function of the pixel index: $I(n) \cdot I(n+1)$ function of n, top) frame, bottom) DFD (the coordinate point (0,0) corresponds to the upper left corner of the image).

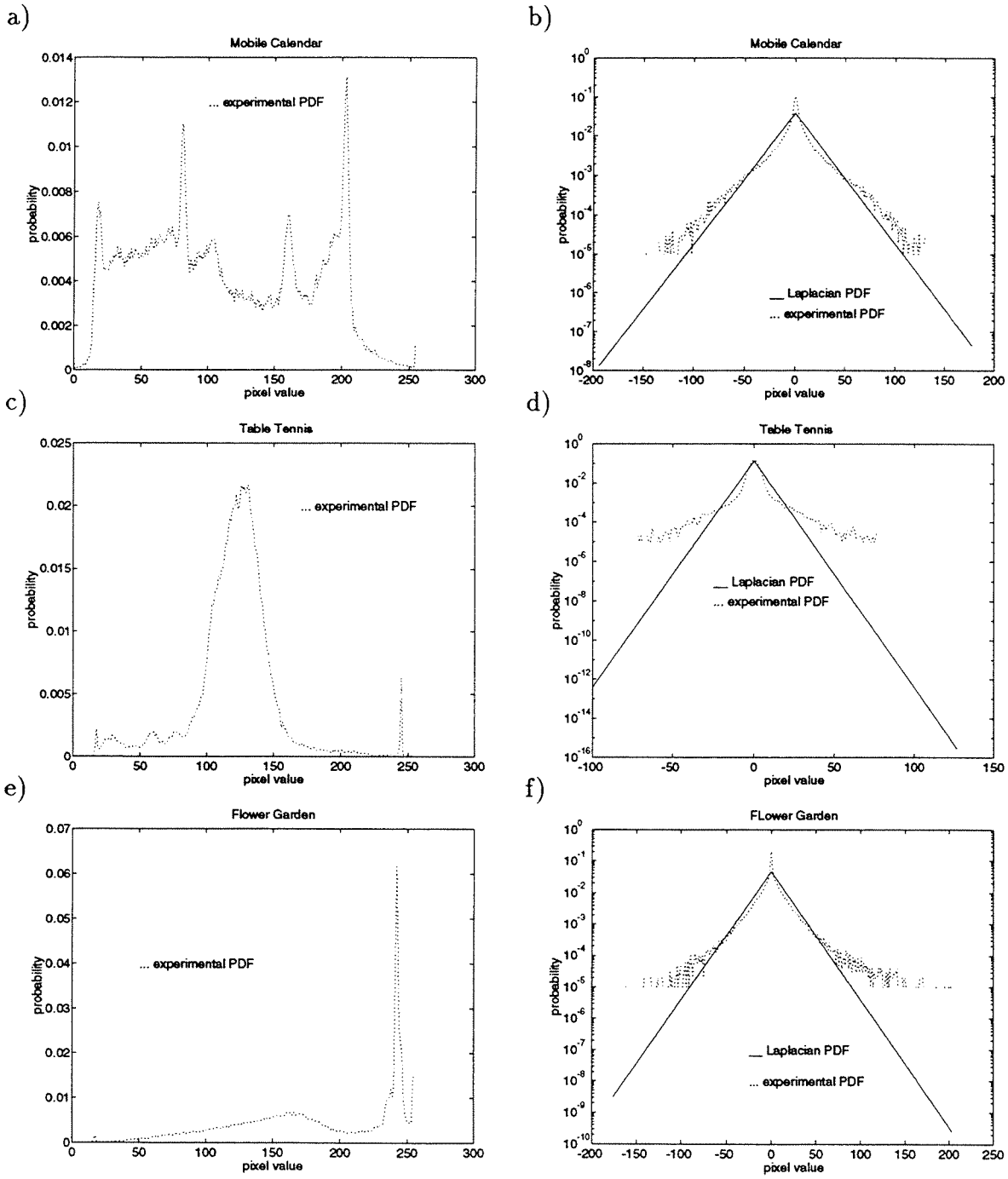

Figure 7.2: Histograms of pixel values probabilities, a) frame of "Mobile Calendar", b) DFD of "Mobile Calendar", c) frame of "Table Tennis", d) DFD of "Table Tennis", e) frame of "Flower Garden", f) DFD of "Flower Garden".

DFD entropy - uniform quantization case

In a lossy compression scheme, higher compression ratios are obtained by a coarse quantization of the DFD pixel values. In the case of a uniform quantization of the Laplacian distribution with a quantization step size Q, the PDF is given by

$$p(i) = \begin{cases} \int_{-Q/2}^{Q/2} \frac{1}{\sqrt{2}\sigma} \exp\left(-\frac{\sqrt{2}|nQ+x|}{\sigma}\right) dx & \text{if } i = nQ, n = 0, \pm 1, \pm 2, \ldots \\ 0 & \text{otherwise} , \end{cases} \tag{7.17}$$

namely

$$p(0) = 1 - \exp\left(-\frac{\alpha}{2}\right) \tag{7.18}$$

$$p(nQ) = \sinh\left(\frac{\alpha}{2}\right) \exp\left(-\alpha|n|\right) \quad |n| > 0 , \tag{7.19}$$

where $\alpha = \sqrt{2}Q/\sigma$. The energy E of the uniformly quantized Laplacian PDF is given by

$$\begin{aligned} E &= \sum_{n=-\infty}^{\infty} (nQ)^2 p(nQ) = Q^2 \sinh\left(\frac{\alpha}{2}\right) \sum_{n=-\infty}^{\infty} n^2 \exp\left(-\alpha|n|\right) \\ &= 2Q^2 \sinh\left(\frac{\alpha}{2}\right) \sum_{n=1}^{\infty} n^2 \exp\left(-\alpha n\right) . \end{aligned} \tag{7.20}$$

Using d'Alembert criterion [158]

- for the positive terms series $\sum_{n=1}^{\infty} u_n$, with $u_n \geq 0$,

  if $\lim_{n\to\infty} \frac{u_{n+1}}{u_n} < 1 \Rightarrow$ the series converges ,

it is easy to show that the series converges. We have also that

$$\sum_{n=1}^{\infty} n^2 \exp\left(-\alpha n\right) = \frac{\partial^2}{\partial \alpha^2} \sum_{n=1}^{\infty} \exp\left(-\alpha n\right) \tag{7.21}$$

with the geometric series

$$\sum_{n=1}^{\infty} \exp\left(-\alpha n\right) = \frac{1}{e^{\alpha} - 1} . \tag{7.22}$$

Therefore, we obtain

$$E = 2Q^2 \sinh\left(\frac{\alpha}{2}\right) \frac{e^{\alpha}(e^{\alpha} + 1)}{(e^{\alpha} - 1)^3} . \tag{7.23}$$

The derivation of the entropy H of the quantized Laplacian PDF is similar

$$\begin{aligned} H &= -\sum_{n=-\infty}^{\infty} p(nQ)\log_2(p(nQ)) \\ &= -p(0)\log_2(p(0)) \\ &\quad -\left(\sum_{n=-\infty}^{-1}+\sum_{n=1}^{\infty}\right)\left\{\sinh\left(\frac{\alpha}{2}\right)\exp\left(-\alpha|n|\right)\log_2\left(\sinh\left(\frac{\alpha}{2}\right)\exp\left(-\alpha|n|\right)\right)\right\} \\ &= -p(0)\log_2(p(0)) \\ &\quad -2\sinh\left(\frac{\alpha}{2}\right)\left\{\log_2\left(\sinh\left(\frac{\alpha}{2}\right)\right)\sum_{n=1}^{\infty}\exp\left(-\alpha n\right)+\frac{\alpha}{\ln 2}\sum_{n=1}^{\infty}(-n)\exp\left(-\alpha n\right)\right\} . \end{aligned} \tag{7.24}$$

It is easy to prove that both series converge using d'Alembert criterion. We also have

$$\sum_{n=1}^{\infty}(-n)\exp\left(-\alpha n\right)=\frac{\partial}{\partial\alpha}\sum_{n=1}^{\infty}\exp\left(-\alpha n\right)=\frac{-e^{\alpha}}{\left(e^{\alpha}-1\right)^2} . \tag{7.25}$$

The expression for the entropy H is straightforward

$$H=-p(0)\log_2(p(0))-2\sinh\left(\frac{\alpha}{2}\right)\left\{\log_2\left(\sinh\left(\frac{\alpha}{2}\right)\right)\frac{1}{\mathrm{e}^{\alpha}-1}+\frac{\alpha}{\ln 2}\frac{-\mathrm{e}^{\alpha}}{(\mathrm{e}^{\alpha}-1)^2}\right\} . \tag{7.26}$$

In the entropy criterion, the entropy of the DFD needs to be evaluated in order to estimate the required bit rate for this component. This entropy can be obtained, according to the modeling of the PDF, from the energy. As the latter is obtained from unquantized pixel values (actually quantized with a step size of 1), the standard deviation σ can be derived through Eq. (7.15). Thus, this value is introduced in Eq. (7.26) in order to obtain the entropy H corresponding to the quantized energy E. Therefore, by computing the energy of the pixels (straightforward if the matching criterion in the block matching algorithm is the MSE), the entropy can be derived, and this last value is introduced in the entropy criterion.

The relationship between the energy E and the entropy H (Eqs. (7.23) and (7.26)) is now experimentally validated. As no direct analytical expression exists, a numerical approach is used. Figure 7.3 shows the comparison between the numerically computed curve and experimental points obtained using various sequences, the quantization step size being $Q = 5$. Similar results have been obtained with various quantization step sizes. A good match between theoretical and experimental results is observed.

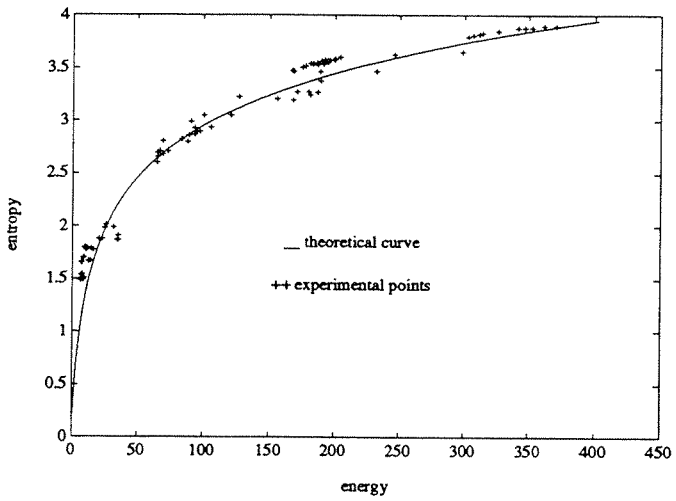

Figure 7.3: Validation of the energy-entropy analytical formula for $Q = 5$.

7.4 Application to locally adaptive multigrid block matching motion estimation

In order to illustrate the efficiency of the entropy criterion, the latter is applied to control the split procedure in the locally adaptive multigrid block matching motion estimation technique presented in Chap. 5.

The adaptive multigrid block matching technique generates a locally varying block size. This coarse segmentation is carried out by a quad-tree decomposition. Blocks for which the accuracy of the obtained motion vector is not sufficient are further split. The corresponding motion vector is refined on a finer grid until a satisfactory accuracy is achieved or a minimum block size is reached. The criterion to decide whether to split a block determines both the accuracy of the motion field and the amount of overhead information. Consequently, it influences significantly the overall performance of the motion estimation procedure.

A simple criterion defined by Eq. (5.1) and applied in [102, 150, 16] as well as in Chap. 5 is the following:

- If the mean absolute error (MAE) (or another error measure) of the motion compensated block is above a preset threshold T, the block is split.

$$\mathrm{MAE}_{\mathrm{nosplit}} > T \;\Rightarrow \text{split} \; . \tag{7.27}$$

However, the above criterion does not guarantee that the extra-cost to send more motion parameters is worth the gain of decreasing the DFD energy. Furthermore, it requires to determine an appropriate value of the threshold T.

By applying the entropy criterion described in Sec. 7.2, the split can be controlled in order to reach the optimal bit allocation between motion parameters and DFD information. Moreover, it overcomes the problem of setting a threshold. The entropy criterion (see Eq. (7.3)) is applied blockwise and can be written as follows:

- If the extra-cost to send additional motion parameters is worth the gain obtained on the DFD side, then the block is split.

$$n \cdot (H_{\text{DFD nosplit}} - H_{\text{DFD split}}) > 4 \cdot H_{\vec{v}\ \text{split}} - H_{\vec{v}\ \text{nosplit}} \Rightarrow \text{split} , \tag{7.28}$$

  where n is the number of pixels in the block, $H_{\text{DFD split}}$ and $H_{\text{DFD nosplit}}$ are their entropy with/without split respectively, and $H_{\vec{v}\ \text{split}}$ and $H_{\vec{v}\ \text{nosplit}}$ the entropy of the motion vectors with/without split respectively.

In the algorithm, the amount of information to transmit the segmentation information, i.e. the quad-tree, is negligible. Therefore, the extra-cost is only represented by an increased number of motion vectors. The factor 4 is due to the fact that, in case of splitting, four motion vectors are transmitted for the block (quad-tree segmentation) instead of one. The entropy of the DFD block pixels is evaluated through Eq. (7.26), while the entropy of the motion vectors is estimated from the vectors already available.

7.4.1 Simulation results

Simulations are now carried out in order to compare the performances of the segmentation control procedures defined by Eqs. (7.27) and (7.28) in the locally adaptive multigrid block matching motion estimation technique. The latter is used in both structure 1 and 2 (see Chap. 5 and Table 5.1), the experimental conditions being identical to those in Sec. 5.3.

Simulations are performed with the schemes $\mathcal{A}$ and $\mathcal{B}$ (see Chap. 3). The scheme $\mathcal{B}$ is consistent with the memoryless source model of the DFD introduced in Sec. 7.3. More precisely, as this model assumes no correlation between pixels of the DFD, any transform is consequently useless. However, in practice a low correlation remains in the DFD which is thus coded by a transform technique as in the scheme $\mathcal{A}$. Nevertheless, due to the low correlation, the 0th-order entropy still accurately approximates the DFD coding cost. Therefore the entropy criterion remains meaningful in this scheme as well, as we shall see.

Figures 7.4 and 7.5 show the bit rate as a function of the threshold for the segmentation control procedure defined by Eq. (7.27) as well as the bit rate obtained with the entropy criterion (Eq. (7.28)) when using the locally adaptive multigrid block matching motion estimation technique in the structure 1 with the schemes $\mathcal{A}$ and $\mathcal{B}$ respectively. Figures 7.6 and 7.7 show similar results for the structure 2.

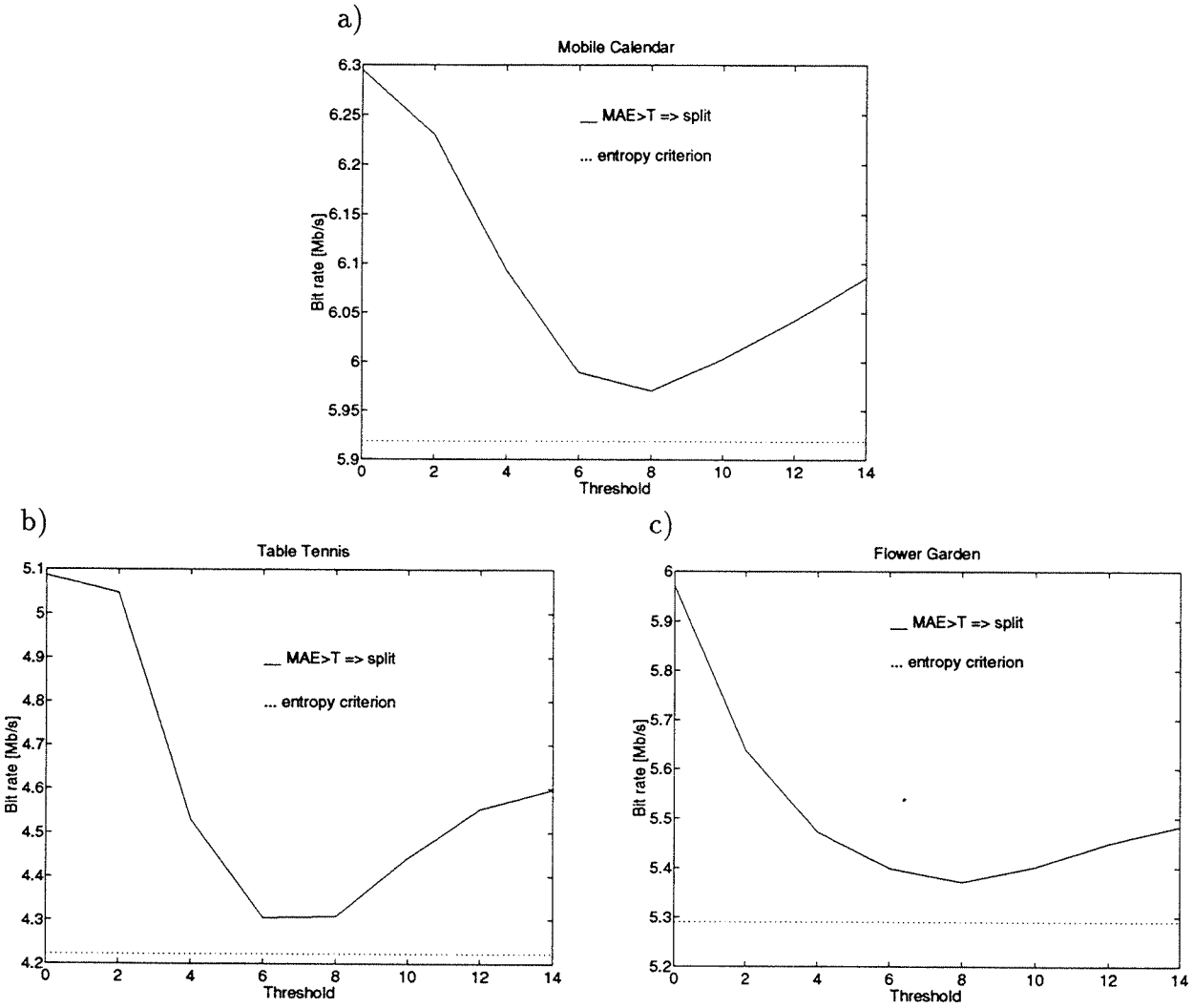

Figure 7.4: Scheme $\mathcal{A}$: comparison between Eqs. (7.27) and (7.28) to control the segmentation in the locally adaptive multigrid block matching motion estimation (structure 1), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

The first observation is that the criterion defined by Eq. (7.27) shows a characteristic behavior. On the one hand and for a small threshold value, too many blocks are split, resulting in a high overhead motion information compared to a small gain in terms of

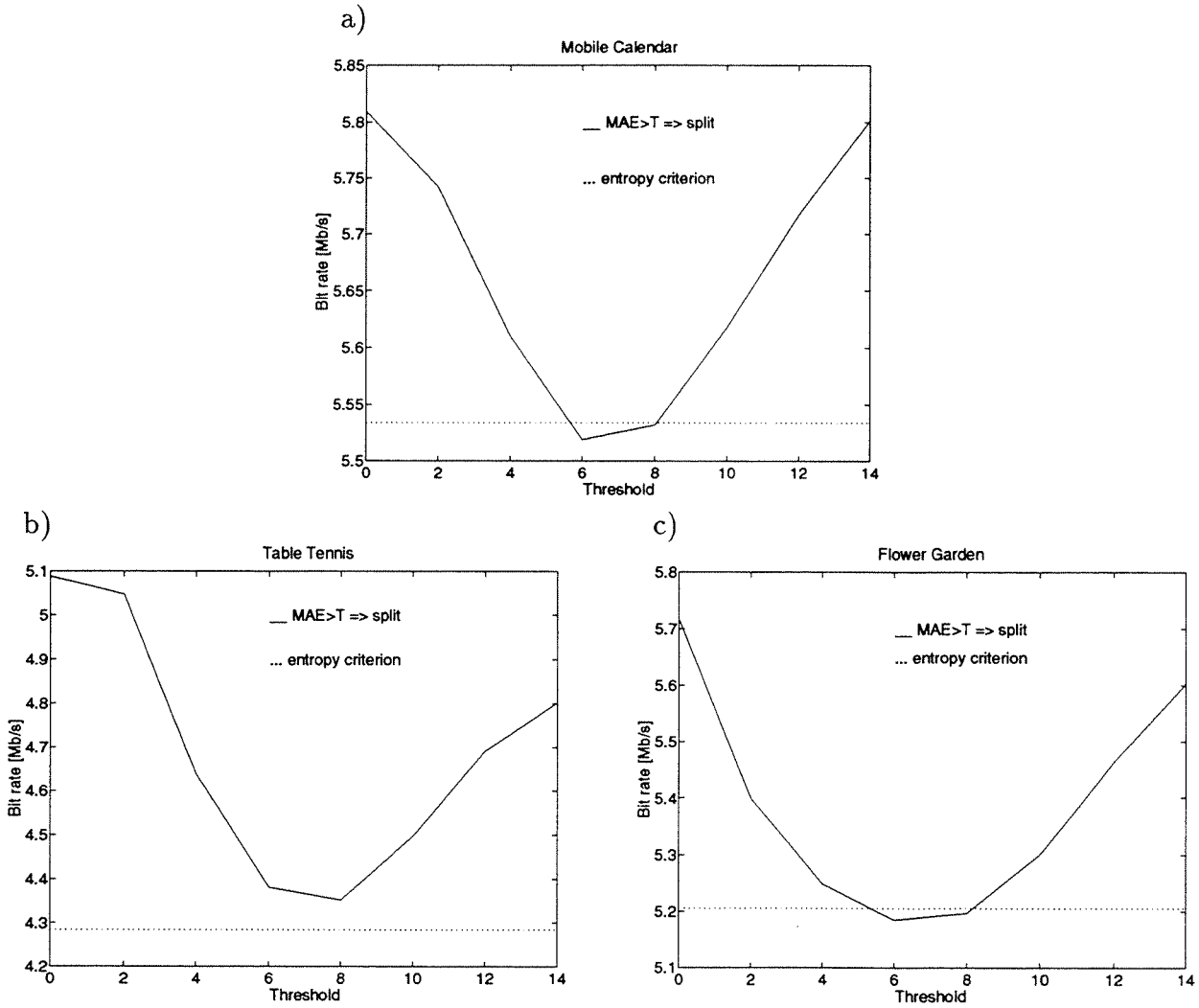

Figure 7.5: Scheme $\mathcal{B}$: comparison between Eqs. (7.27) and (7.28) to control the segmentation in the locally adaptive multigrid block matching motion estimation (structure 1), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

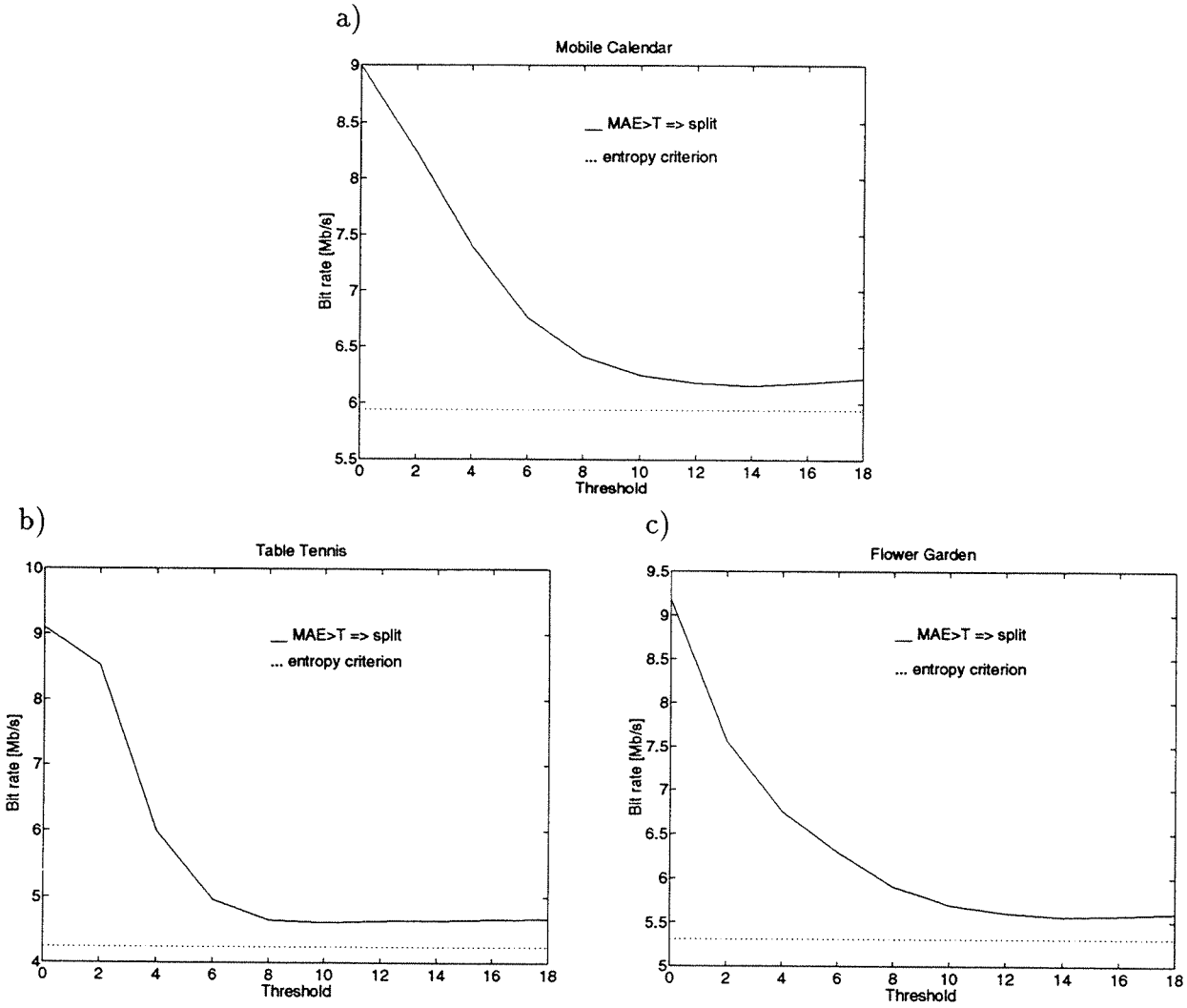

Figure 7.6: Scheme $\mathcal{A}$: comparison between Eqs. (7.27) and (7.28) to control the segmentation in the locally adaptive multigrid block matching motion estimation (structure 2), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

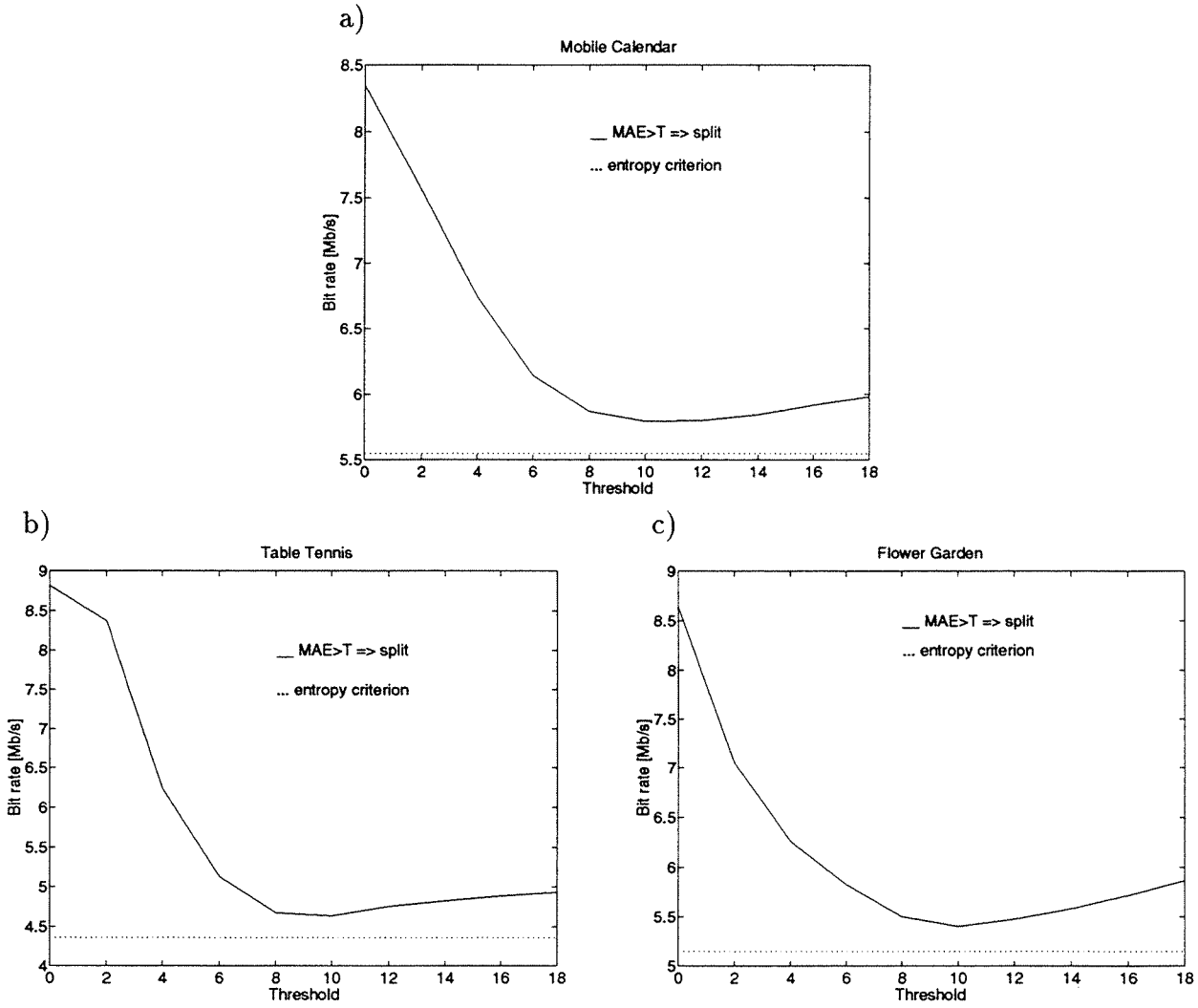

Figure 7.7: Scheme $\mathcal{B}$: comparison between Eqs. (7.27) and (7.28) to control the segmentation in the locally adaptive multigrid block matching motion estimation (structure 2), a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

the bit rate to transmit the DFD. On the other hand and for a high threshold value, too few blocks are split, reducing the efficiency of the adaptive multigrid block matching motion estimation technique. Between these two extremes, there is an optimal threshold value which gives the best performance. However, the arbitrary nature of this optimum does not allow to predetermined it. Only several trials could lead to this value, but this method is not feasible in practice.

The second and most important observation is that the entropy criterion (Eq. (7.28)) always outperforms the criterion based on a threshold (Eq. (7.27)), in both cases with and without transform applied on the DFD. Even though the gain is small when an optimal threshold value is used in Eq. (7.27), it becomes significant for a non-optimal one. Furthermore, as the entropy criterion overcomes the problem of setting the threshold value, it leads to a workable algorithm for practical applications.

As far as the reconstructed sequence quality is concerned, the following observation is done. On the one hand, the visual quality depends strongly on the threshold in the criterion defined by Eq. (7.27). On the other hand, the entropy criterion always segments moving edges, and leads therefore to a constantly high visual quality.

As a conclusion of these simulations, the superiority of the entropy criterion defined by Eq. (7.28) is clearly demonstrated in the locally adaptive multigrid block matching motion estimation technique. It leads to a decreased bit rate and a high visual quality. Furthermore, it does not require any parameters or threshold setting.

7.5 Application to VQ-based segmentation of the motion field

The entropy criterion is now applied to a different algorithm, the VQ-based segmentation of the motion field introduced in Chap. 6.

In the VQ-based segmentation technique, blocks corresponding to moving edges, for which the motion model fails, undergo a segmentation in two regions. Each of the resulting regions is assigned a different motion vector. The criterion to decide whether to segment a block controls the precision of the motion estimation as well as the amount of side information to represent the segmentation. Therefore, it influences greatly the effectiveness of the method.

A simple criterion to control the segmentation, defined by Eq. (6.15) and applied in Chap. 6 is as follows:

- Blocks with a MAE (or another error measure) higher than a preset threshold T are assumed to contain boundaries of moving objects and are segmented.

$$\mathrm{MAE}_{\mathrm{noseg}} > T \Rightarrow \text{segmentation} . \tag{7.29}$$

As already mentioned, the above criterion suffers two drawbacks: first it requires to select an adequate value of the threshold, second it does not guarantee that the gain on the bit rate obtained thanks to the segmentation is worth the extra cost to transmit the segmentation information.

When applying the entropy criterion described in Sec. 7.2, the segmentation can be controlled in order to optimize the bit allocation between the motion and segmentation information on the one side and DFD information on the other side. Moreover, it avoids the setting of a threshold. The entropy criterion (see Eq. (7.3)), which is applied blockwise, is straightforwardly derived from Eq. (6.14):

- If the extra-cost to send additional motion and segmentation information is worth the gain obtained on the DFD side, then the block is segmented.

$$n \cdot (H_{\mathrm{DFD\ seg}} - H_{\mathrm{DFD\ noseg}}) < [\log_2(\gamma)] + 5 \Rightarrow \text{segmentation} , \tag{7.30}$$

  where n is the number of pixels in the block, $H_{\mathrm{DFD\ noseg}}$ and $H_{\mathrm{DFD\ seg}}$ are the entropy of the DFD block pixels without/with segmentation respectively, and γ is the codebook size.

The above criterion (Eq. (7.30)) decides whether to segment a block in two regions. In the more general algorithm described in Sec. 6.2.1 which performs a segmentation in N regions, $N = 2, 3, \ldots, N_{\max}$, the entropy criterion could be used simultaneously to decide for each block whether it is segmented and to determine the optimal number of segmented regions N.

7.5.1 Simulation results

The following simulations compare the performances of the criteria defined by Eqs. (7.29) and (7.30) when applied to the segmentation control in the VQ-based motion field segmentation technique. The simulations are carried out with the schemes $\mathcal{A}$ and $\mathcal{B}$ described in Chap. 3, the experimental conditions being identical to those in Sec. 6.3.

Figures 7.8 and 7.9 show the bit rate function of the threshold (expressed in %) using the criterion defined by Eq. (7.29) compared to the proposed entropy criterion defined by Eq. (7.30), in the schemes $\mathcal{A}$ and $\mathcal{B}$ respectively.

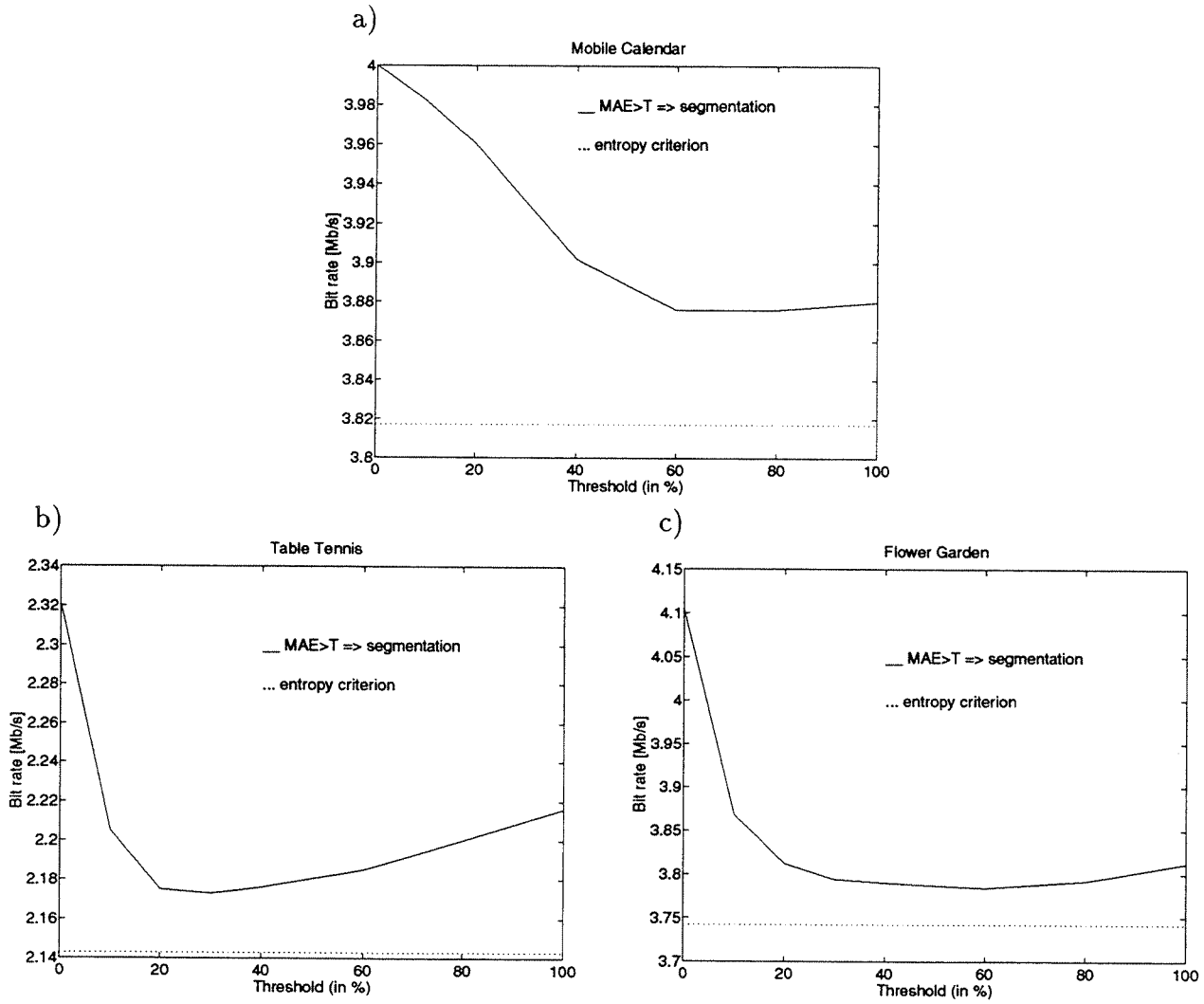

Figure 7.8: Scheme $\mathcal{A}$: comparison between Eqs. (7.29) and (7.30) to control the segmentation in the VQ-based motion field segmentation technique, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

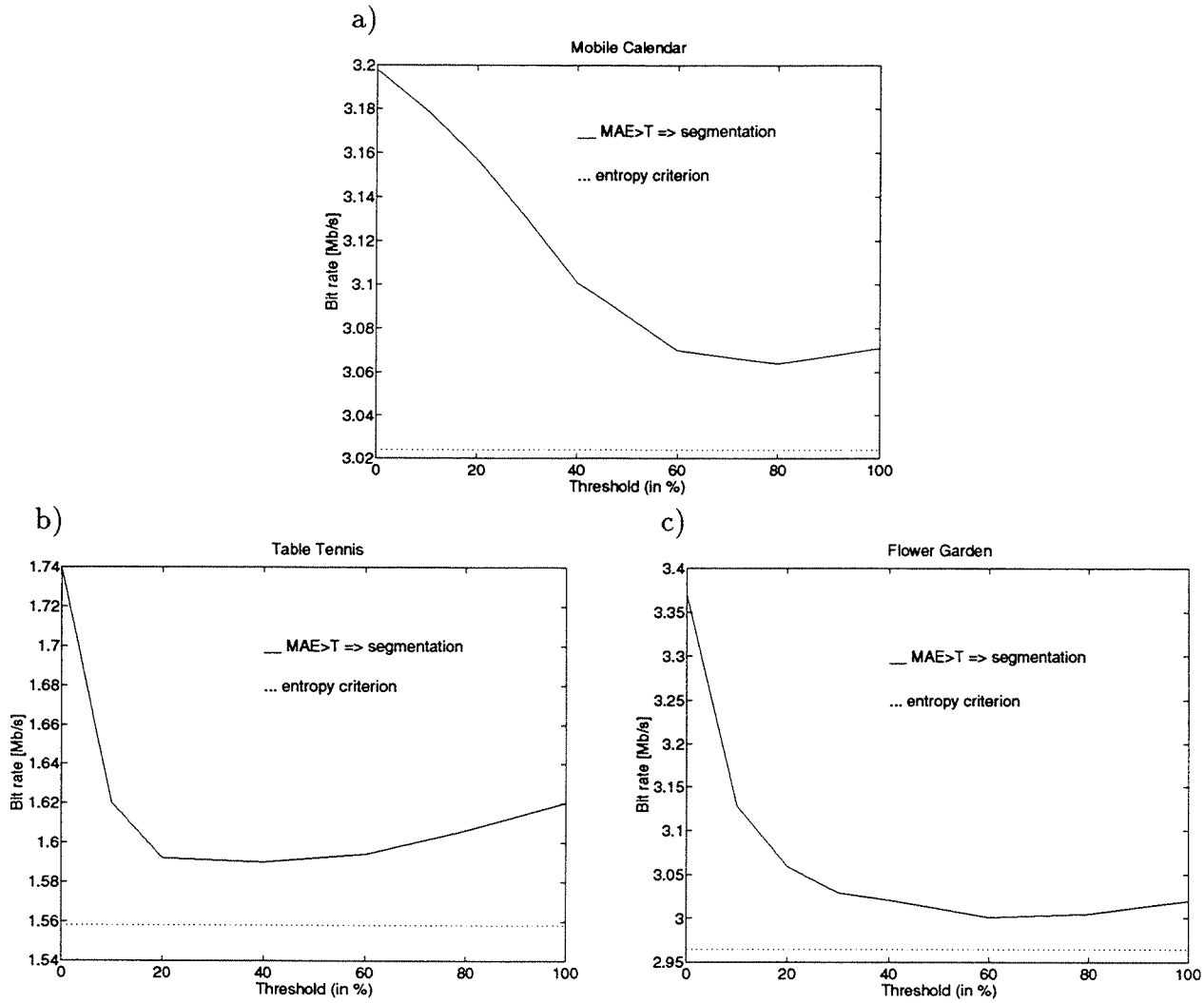

Figure 7.9: Scheme $\mathcal{B}$: comparison between Eqs. (7.29) and (7.30) to control the segmentation in the VQ-based motion field segmentation technique, a) "Mobile Calendar", b) "Table Tennis" and c) "Flower Garden".

From the above results, it appears that the segmentation algorithm controlled by Eq. (7.29) reaches an optimum for an arbitrary threshold value. A lower threshold does not exploit the efficiency of the segmentation technique and a higher one raises the bit rate. The last observation is straightforwardly explained. As more blocks are selected for segmentation all of them for which the segmentation decreases the DFD energy are segmented regardless to the fact that the segmentation leads to a net gain in terms of total bit rate. Consequently the segmentation bit rate increases constantly with an increase of the threshold. For a high threshold value, this increase annihilates the gain obtained by the decrease of the DFD bit rate.

Clearly, the proposed entropy criterion (Eq. (7.30)) outperforms the threshold criterion (Eq. (7.29)), reaching always a lower bit rate. Whereas the former method segments only blocks which lead to a gain in terms of total bit rate therefore controlling efficiently the segmentation process, the latter does not have this concern. Even though the gain in terms of bit rate is small, the entropy criterion avoids the setting of a threshold and leads therefore to a workable algorithm.

Finally, the entropy criterion always segments moving edges and achieves a high visual quality of the reconstructed sequence. In opposite, in the criterion defined by Eq. (7.29) the visual quality depends strongly on the threshold.

To conclude, the superiority of the entropy criterion (Eq. (7.30)) is clearly shown while using the VQ-based segmentation of the motion field. It minimizes the bit rate, leads to a high visual quality, and does not need a threshold setting.

7.6 Summary

In this chapter, a method has been introduced to optimally allocate the bit rate between the motion overhead information on the one side and the DFD component on the other side. The coding costs relative to the different components of the total bit rate are evaluated. Hence, the method minimizes the sum of the above terms. It provides a criterion, referred to as the entropy criterion, to decide the appropriate accuracy of the motion estimation procedure. Resulting from this optimization, a minimization of the total bit rate is achieved.

In order to estimate the coding cost of the DFD, a statistical model of the latter is needed. Experiments have shown that a memoryless source model is adequate and that the Laplacian PDF matches closely the measured PDF of the DFD. Hence, the 0th-order entropy is straightforwardly obtained and consequently the DFD coding cost.

The entropy criterion has been successfully applied to control the segmentation in both the locally adaptive block matching motion estimation (see Chap. 5) and the VQ-based segmentation of the motion field (see Chap. 6). Simulation results show that the entropy criterion reaches always a lower bit rate when compared to a criterion based on a threshold and achieves a constantly high visual quality. Furthermore, it avoids the setting of a threshold and leads to a workable algorithm. Therefore, the superiority of the proposed criterion has been clearly demonstrated.

Chapter 8

Application to generic video coding

8.1 Introduction

As an application of the locally adaptive multigrid block matching motion estimation (see Chap. 5) controlled by the entropy criterion (see Chap. 7), in this chapter a new generic coding scheme is described [53]. It is an improved variant of the Swiss Federal Institute of Technology (EPFL) proposal for MPEG-II presented at the Kurihama evaluation meeting in Japan in November 1991 [159].

Recent standards such as MPEG-I [5, 30] and -II [6, 31] from ISO, which are based on a motion compensated DCT technique, have shown their efficiency in terms of picture quality versus data rate for medium bit rate applications. Nevertheless, important features, such as generic coding, scalability and interactivity, are desired in emerging visual information processing applications [160] (e.g. digital TV and HDTV, video-conference, video-phone, medical imaging, archiving, and multimedia). However, these features are not supported (at least not straightforwardly) in the above standards. Furthermore, the DCT produces annoying block artifacts in low bit rate applications. For instance, this is the case with the recommendation H.261 [7] from CCITT. These observations motivate the study of new coding schemes.

The new generic coding scheme proposed in this chapter aims at overcoming the drawbacks of the above standards. It is based on a motion compensated Gabor-like wavelet transform. The motion estimation is carried out by the locally adaptive multigrid block matching technique (see Chap. 5) controlled by the entropy criterion (see Chap. 7). The scheme works independently from the input sequence resolution and can provide a wide range of bit rates. Furthermore, the bit stream is scalable [160], i.e. it can be partially decoded to reconstruct a lower resolution and lower quality sequence. For these reasons, the scheme is naturally suitable for multimedia applications. Finally, the system supports efficiently both interlaced and progressive scan formats.

The chapter is structured as follows. The general description of the system is given in Sec. 8.2. Next, the main components of the codec are discussed in more details: the Gabor-like wavelet transform in Sec. 8.3, the motion estimation in Sec. 8.4, the motion compensation in Sec. 8.5, the quantization in Sec. 8.6 and the entropy coding in Sec. 8.7. Simulation results obtained when applying the coding scheme to HDTV, TV and videophone applications are given in Sec. 8.8. These results show the efficiency of the system for a wide range of bit rates and very different applications. Finally, Sec. 8.9 draws the conclusions.

8.2 General description of the codec

This section gives a general description of the motion compensated wavelet transform based system for generic video coding. Figures 8.1 and 8.2 represent the block diagram of the coder for progressive and interlaced input formats respectively.

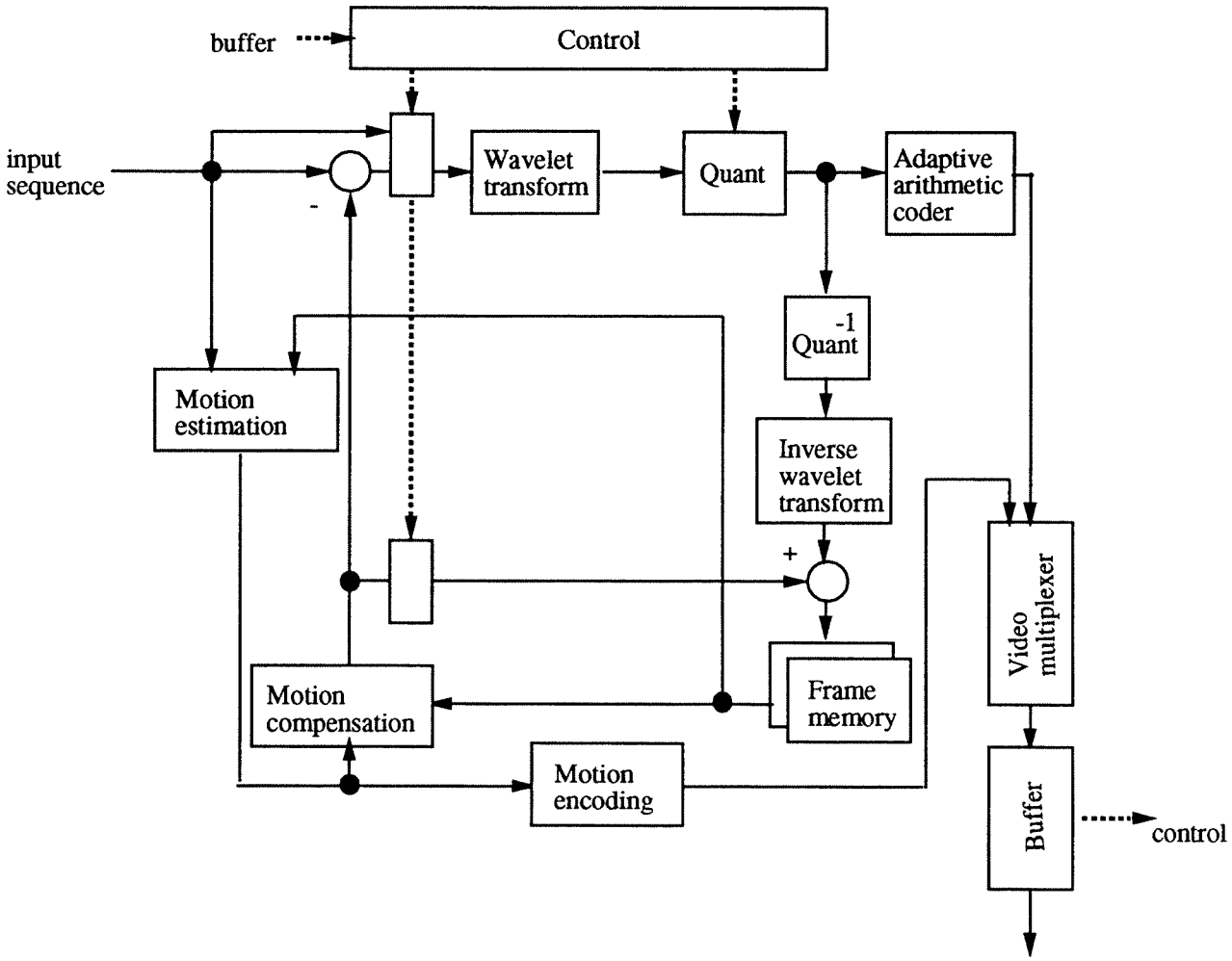

Figure 8.1: Block diagram of the coder for progressive input format.

The spatial correlation is reduced by a Gabor-like wavelet transform, performing octave band partitioning of the spatial-frequency domain. This partition produces a multiresolution structure, which is suitable for generic coding and scalability. Furthermore, the artifacts introduced by a wavelet transform are perceptually more acceptable when compared to the block artifacts characteristic of the DCT.

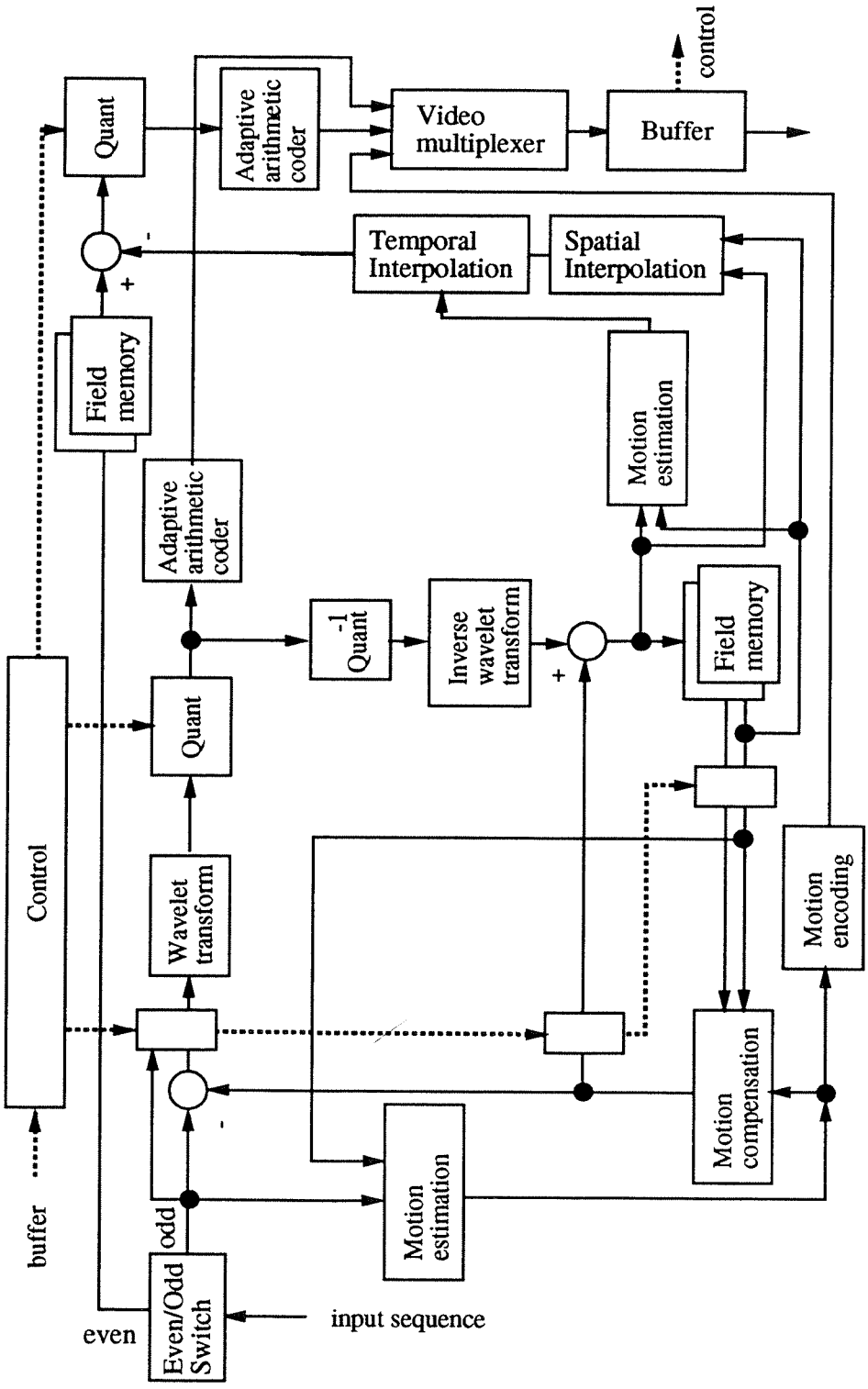

Figure 8.2: Block diagram of the coder for interlaced input format.

The temporal correlation is reduced by motion compensation. The motion estimation is performed by the locally adaptive multigrid block matching technique introduced in Chap. 5. The entropy criterion described in Chap. 7 is applied to control the split procedure.

In the case of progressive input sequences, an interframe coding technique including predictive and interpolative modes is applied. For interlaced sequences, odd fields are directly coded as a progressive sequence, and even fields are predicted by spatial and temporal interpolation of the corresponding decoded odd fields. Therefore, both progressive and interlaced scan formats are efficiently handled.

The coefficients after the wavelet transform are scalar quantized by using a quantization weighting matrix that takes into account the perceptual relevance of components belonging to different subbands.

The output of the quantization stage as well as the motion vectors are entropy coded by the adaptive arithmetic coder proposed in [127].

The chrominance components are treated exactly in the same way as the luminance one with proper quantization parameters. Furthermore, motion vectors are transposed from the luminance component by adapting their amplitudes and their spatial resolution accordingly.

Investigations on the hardware complexity of the proposed scheme show its ease of implementation when compared to other classical techniques [136, 138].

8.3 Gabor-like wavelet transform

One of the main drawbacks in video compression techniques is that transformations or filter banks used for video compression have been primarily designed for still-image compression algorithms. For instance, natural images can be approximated locally as being generated by a first-order Markov stationary process. The energy of signals generated by such a process is optimally compacted by the Karhunen-Loève transform (KLT). As the DCT approximates closely the KLT, its use for still-image coding is justified. However, in most of the image sequence compression techniques, only in the intraframe mode is the actual image directly transformed. In interframe modes, an error image from motion compensated prediction or interpolation is transformed instead. It can be shown that these error images cannot be modeled by a first-order stationary Markov process. Accordingly, the justification of the DCT, as an optimal transform is not valid anymore for these interframe modes.

Therefore, the video coding system presented in this chapter applies a Gabor-like wavelet transform performing octave band partitioning of the spatial-frequency domain. This partition is motivated by typical image statistics as well as by the spatial-frequency sensitivity of the human visual system. Moreover, it produces a multiresolution structure of interest for generic coding and scalability. For instance, by discarding or taking into account a certain number of levels in the multiresolution structure, an HDTV sequence can be decoded by a TV receiver, and vice versa. Furthermore, the use of Gaussian-shaped filters (Gabor functions which are Gaussian modulated by complex exponentials) is motivated due to their optimal localization in the spatial/spatial-frequency domain. Finally, in low bit rate applications, a certain amount of distortion is introduced in the reconstructed data. In this situation, perceptually derived transforms, such as the Gabor-like wavelet, are more suitable. The Gaussian shape of these filters produces less perceptible distortion at high compression [161] when compared to the block artifacts that are typical in block-transform-based techniques.

The transformed coefficients of the input signal are generated by a tree structure and a rectangular separable biorthogonal wavelet transform [125], as shown in Fig. 8.3. The basic structure is a 1-D two channel frequency decomposition. The high-frequency channel filter is obtained from the low-frequency channel by a π shift in a normalized scale. The low-frequency filter is an approximation of a Gaussian, in which the frequency response at points π and $-\pi$ have been set to zero. This guarantees a zero at the DC for the high-frequency filter response. By applying the basic two-channel decomposition in a recursive way on only the high-pass part of each level, all frequency channels except the low-pass one will have a zero at DC. Moreover, to decrease the computational and implementational complexity to simple shift-and-add operations only, analysis and synthesis filters are with coefficients in powers of two. Furthermore, filter banks for image processing applications should be very short and have linear phase characteristic. It is also suitable if they can be implemented in polyphase structure. In some applications in which decoding is performed more often, synthesis filters shorter than analysis ones are desired. The filter bank associated with the Gabor-like wavelet transform have all the above properties. Tables 8.1 and 8.2 show the unnormalized values of the coefficients in the analysis and the synthesis filters. The normalization is taken into account in the quantization stage.

Figure 8.4 shows the performance on a still image of a DCT and the Gabor-like wavelet transform discussed in this section. The input image is "Lena", 512×512 black and white, 8 bit/pixel and the comparison is made in terms of PSNR versus compression. As can be seen, the wavelet transformation outperforms the DCT for high compression. For lower compression, the DCT gives a higher PSNR than the wavelet transformation. However, at this range the wavelet transformation results in higher visual quality.

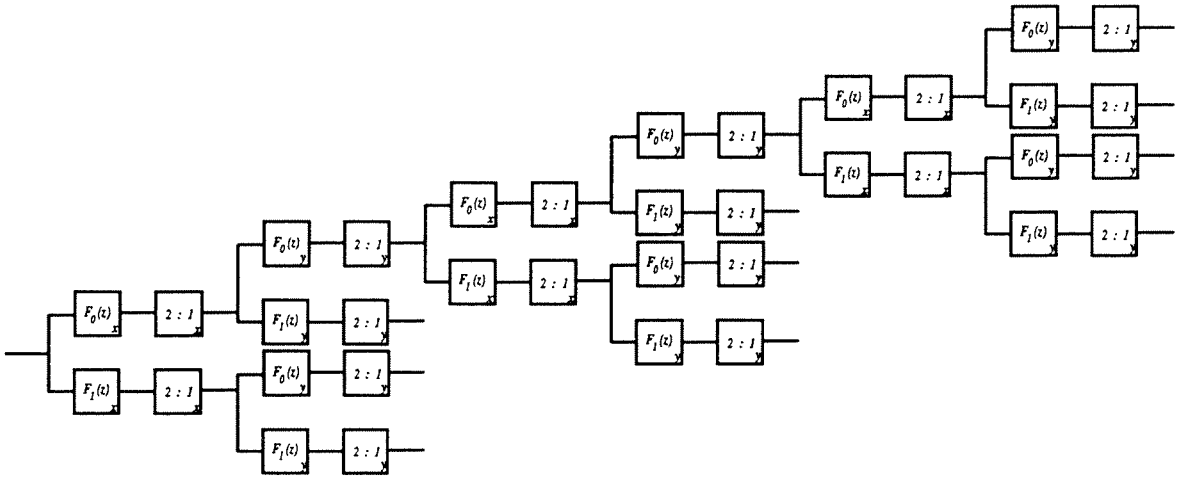

Figure 8.3: Tree structure rectangular separable subband decomposition.

Coefficient	*Value*
$g_0(-3) = g_0(2) = g_1(-3) = -g_1(2)$	2^{-7}
$g_0(-2) = g_0(1) = -g_1(-2) = g_1(1)$	2^{-3}
$g_0(-1) = g_0(0) = g_1(-1) = -g_1(0)$	2^{0}

Table 8.1: The values of coefficients in the synthesis filters

Coefficient	*Value*
$f_0(-5) = f_0(4) = -f_1(-5) = f_1(4)$	2^{-6}
$f_0(-4) = f_0(3) = f_1(-4) = f_1(3)$	0
$-f_0(-3) = -f_0(2) = f_1(-3) = -f_1(2)$	2^{-3}
$-f_0(-2) = -f_0(1) = -f_1(-2) = f_1(1)$	2^{-7}
$f_0(-1) = f_0(0) = -f_1(-1) = f_1(0)$	2^{0}

Table 8.2: The values of coefficients in the analysis filters

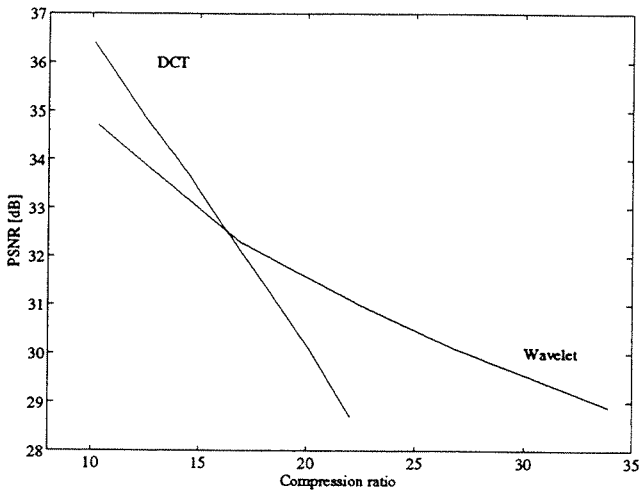

Figure 8.4: Comparison of performances between DCT and the proposed wavelet in terms of PSNR versus compression ratio.

In order to further increase the energy compaction capability of the transform, an adaptive frequency decomposition can be performed. It is known as the wavelet packet transform [162]. Results obtained with the proposed coding scheme when using a wavelet packet transform are given in [103, 104].

8.4 Motion estimation

The locally adaptive multigrid block matching, as described in Chap. 5, including the entropy criterion introduced in Chap. 7, is used for motion estimation. As the algorithm has already been widely discussed in the previous chapters, we will only remind its key points.

The full-search block matching is widely used in the field of video coding. However, the locally adaptive multigrid block matching outperforms this algorithm for the following reasons. First, it generates smoother and more robust motion fields, close to the true motion in the scene. Therefore, it improves the coding of both the DFDs (as they have less discontinuities) and the motion vectors (as they are smoother). Second, it is quasi-optimal in minimizing the DFD energy when compared to an exhaustive search. Third, it requires a greatly decreased computational complexity. This point is very important when carrying out intensive software simulations. Fourth, the local mesh adaptation provides more accurate motion vectors in detailed areas and simultaneously decreases the total

overhead motion information.

Besides, as the generic video system codes very different sequences (e.g. HDTV, TV, video-phone) at a wide range of bit rates, it is important for the motion estimation to be flexible and to adapt to these various applications. In this context, the entropy criterion provides a very efficient control of the split procedure in the locally adaptive multigrid block matching motion estimation. The algorithm reaches always the optimal trade-off between DFD and motion information given the allotted bandwidth. This is not the case for a strategy based on a threshold, in which the threshold requires to be optimally set for each different application.

8.5 Motion compensation

The Gabor-like wavelet transform, as described in Sec. 8.3, achieves spatial decorrelation. To reduce the temporal redundancies as well, an interframe/interfield coding technique is applied. Motion compensation is performed in the image domain, using the motion estimation information. The resulting prediction/interpolation error is spatially transformed by the Gabor-like wavelet transform.

Motivated by the need for video codec to handle both progressive and interlaced formats efficiently the following motion compensated scheme has been adopted.

8.5.1 Progressive scan

In the case of progressive sequences, an interframe coding technique similar to the one used in MPEG-II [6, 31] is applied. A Group of Pictures (GOP) is defined in which the first frame is intraframe coded and the following frames are alternatively coded in interpolative and predictive modes, as shown in Fig. 8.5. This structure avoids error propagation and allows fast random access. Thanks to the smooth motion fields provided by the locally adaptive multigrid block matching motion estimation technique, the same motion vectors are used in the predictive and interpolative modes, reducing the hardware complexity and decreasing the amount of motion side information to be transmitted.

8.5.2 Interlaced scan

The interlaced format introduces difficulties for most video coding schemes. MPEG-II combines field and frame coding [50] (see Chap. 2). However, on the one hand, a simple merging of two fields creates high-frequency artifacts that artificially increase the amount of high spatio-temporal frequency components to be transmitted. On the other hand, coding the fields separately does not exploit efficiently the redundancies between fields.

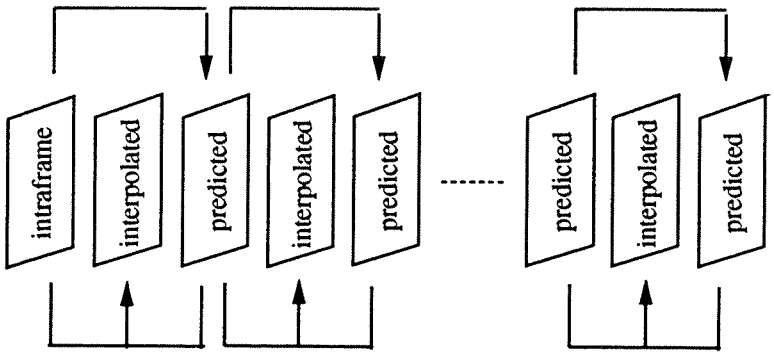

Figure 8.5: Structure of intraframe and interframe modes in a progressive sequence.

These reasons motivate the study of interlaced-to-progressive scan-rate conversion. Techniques based on linear spatial interpolation, nonlinear spatial interpolation [118, 119, 120], median filtering [121] or spatio-temporal interpolation [122] have been proposed. Alternatives have been proposed in [51, 52]. The input signal is separated into odd and even fields. The even fields are directly coded, whereas the odd fields are encoded by motion compensated interfield interpolation.

Similarly, the following technique [53] has been adopted in the proposed codec. Only odd fields are directly coded as a progressive sequence. The even fields are predicted using spatial and temporal interpolation based on the corresponding decoded odd fields. Figure 8.6 illustrates the procedure.

The number of lines in each odd field is doubled by interpolating consecutive lines. This spatial interpolation is performed using a gradient-based compensated interpolation [123, 124], as described in Chap. 3. Next, even fields fields are predicted by motion compensated temporal interpolation between the previous and following spatially interpolated odd fields. The predicted even fields are subtracted from the original fields and the resulting prediction error is quantized, coded and transmitted.

8.6 Quantization

In this section the quantization strategy adopted to code the multiresolution structure of the coefficients is presented. These data result from the Gabor-like wavelet transform and the different coding modes previously described in Sec. 8.3. The quantization approach critically determines the performance of the complete coding scheme. On the one side the different statistical characteristics and perceptual relevances of the coefficients, as well as the existing inter/intraband residual correlations must be taken into account. On the other side, the statistical complexity of the input signal and the feasibility of the codec

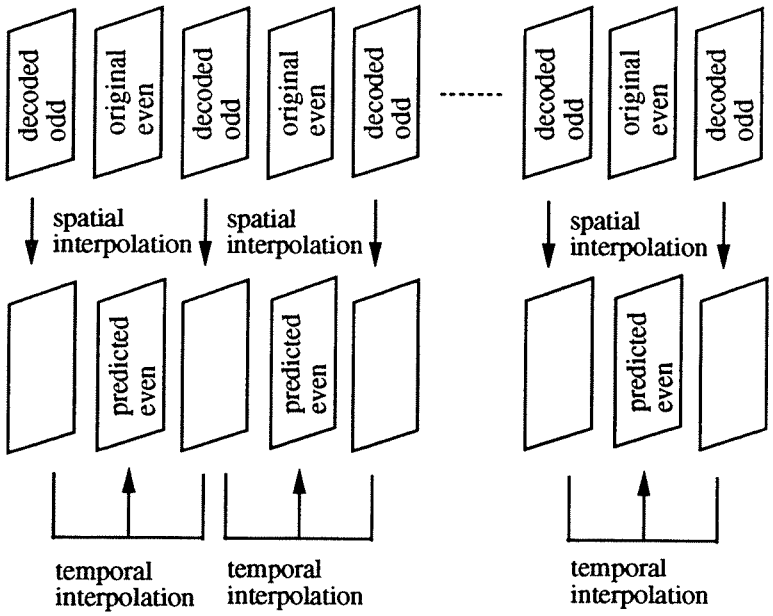

Figure 8.6: Coding of interlaced sequences.

put severe constraints on the complexity of possible quantizers.

A computationally simple solution is the implementation of a scalar quantizer with independent coding of the coefficients [21]. In [163], Berger derives that in subband decomposition the optimal scalar quantization for an ideal entropy coding of coefficients is the uniform quantization. He also shows that, according to the MSE criterion, optimality is reached when the same quantization step size is used for all subbands. However, it is well known that the frequency response characteristic of the human visual system allows high tolerability of quantization/coding error in the higher frequency subbands. Consequently, the above MSE distortion criterion is not relevant from a perceptual point of view. Therefore, a different quantization step size is introduced in each subband. Such steps are chosen to introduce a higher quantization error in the subbands in which the human visual system sensitivity is lower [164].

To take into account the multiresolution pyramidal structure of the Gabor-like wavelet coefficients, the following quantization strategy is applied. The lowest resolution level in the pyramidal data (DC component) is visually very sensitive to distortion and therefore the most important. In addition, the number of coefficients relative to this part is small compared to the other resolution levels. For these reasons, a pulse code modulation (PCM) technique is applied on this part of the data. For the remaining levels, according to the perceptual relevance of the subband coefficients, each subband is multiplied by a

weighting factor. This stage is followed by a uniform scalar quantization with the same quantization in all subbands. Various techniques have been proposed in the literature to evaluate the quantization weighting matrix taking into account the perceptual relevance of quantized coefficients in subbands [164]. In the proposed implementation, the weighting matrix has been empirically determined. Moreover, different weighting matrices have been designed for luminance and chrominance components as well as for the different coding modes. The simulation results obtained with the video codec applying the above quantizer are presented in Sec. 8.8.

An alternative promising technique is a vector quantization (VQ) scheme as proposed in [165]. A multiresolution and frequency-oriented codebook allows to exploit both the intra- and interband correlations. Furthermore, a pyramidal vector-forming technique is applied to follow the pyramidal structure of the data. The vectors are composed of coefficients from different frequency ranges inside the same frequency orientation, thus corresponding to the same spatial location. Finally, in order to take into account the existing correlation between luminance and chrominance components, the chrominance subband coefficients are introduced in the vectors.

8.7 Entropy coding

The coding of quantized coefficients is an important stage in any coder. The Huffman code [166] is widely used in coding. However, the latter is outperformed by another variable length coding strategy known as arithmetic coding [167, 168]. First, arithmetic coding allows to reach a performance very close to the entropy of the source. Second, it can efficiently adapt to changing source statistics. As these two features are not able with Huffman code, arithmetic coding is preferred in the proposed system.

In the proposed coding scheme, the output of the quantization stage is entropy coded by the adaptive arithmetic coder proposed in [127]. The adaptation is performed in the coder by generating and updating an internal histogram of symbols. The decoder is the dual of the coder and is building its own histogram. Therefore, no side information requires to be transmitted for updating the model. To avoid error propagation, the model is reset frequently. The same arithmetic coder, with a different model, is used to code the motion vectors.

8.8 Experimental results

In order to illustrate the generic coding capability of the proposed scheme, three type of applications have been simulated: HDTV, TV and video-phone. The obtained results are

presented in this section.

The HDTV experiments have been performed on the sequence "Un Bel Di" (interlaced, Y: 576 $\times$ 1408 pixels, U & V: 576 $\times$ 704 pixels, subsampling 4:2:2, 50 Hz). The sequence "Flower Garden" in CCIR 601 format has been used for TV results. Finally, the video-phone simulations have been carried out on the sequence "Miss America" in CIF format. The total bit rate corresponding to the different applications are: 30 Mb/s for HDTV, 9 Mb/s for TV and 300 kb/s for video-phone. It corresponds to a compression ratio of approximately 20:1 in the two first cases and 100:1 in the third case.

Figures 8.7, 8.8 and 8.9 illustrates the obtained results for HDTV, TV and video-phone applications respectively. The figures show the total bit rate as well as the PSNR of each component (luminance and chrominance components, odd and even fields) as a function of the frame number.

The results in terms of bit rate illustrate the motion compensation structure. The number of frames in a GOP defines both the random access delay and the control of error accumulation. As the requirements are very different between HDTV and TV on the one side and video-phone on the other side, the GOP has been set to 11 frames in the two first cases and 49 frames in the third case. From the figures, high peaks appear for every intraframe modes which are very expensive in terms of bandwidth. Due to the temporal redundancies reduction, a greatly decreased bit rate is needed for predictive and interpolative modes. The latter mode is slightly less expensive than the former one, showing the superiority of interpolation when compared to extrapolation. The above observations justify the choice of a larger GOP in low bit rate applications.

The PSNR depends on the activity in the scene as well as on the motion compensation structure. However, it remains constantly high showing the good quality of the reconstructed sequences. Moreover, due to the perceptually oriented wavelet transform, the visual quality is high throughout the sequences.

Finally, one can notice that in the sequence "Un Bel Di" the last GOP exhibits very different characteristics when compared to the three first GOPs. It is due to a scene change between the 33rd and 34th frame.

As a conclusions, the simulation results show the generic coding capability of the proposed coding system. High visual quality of the reconstructed sequences have been obtained in applications such as HDTV, TV and video-phone and for bit rates ranging from 30 Mb/s to 300 kb/s.

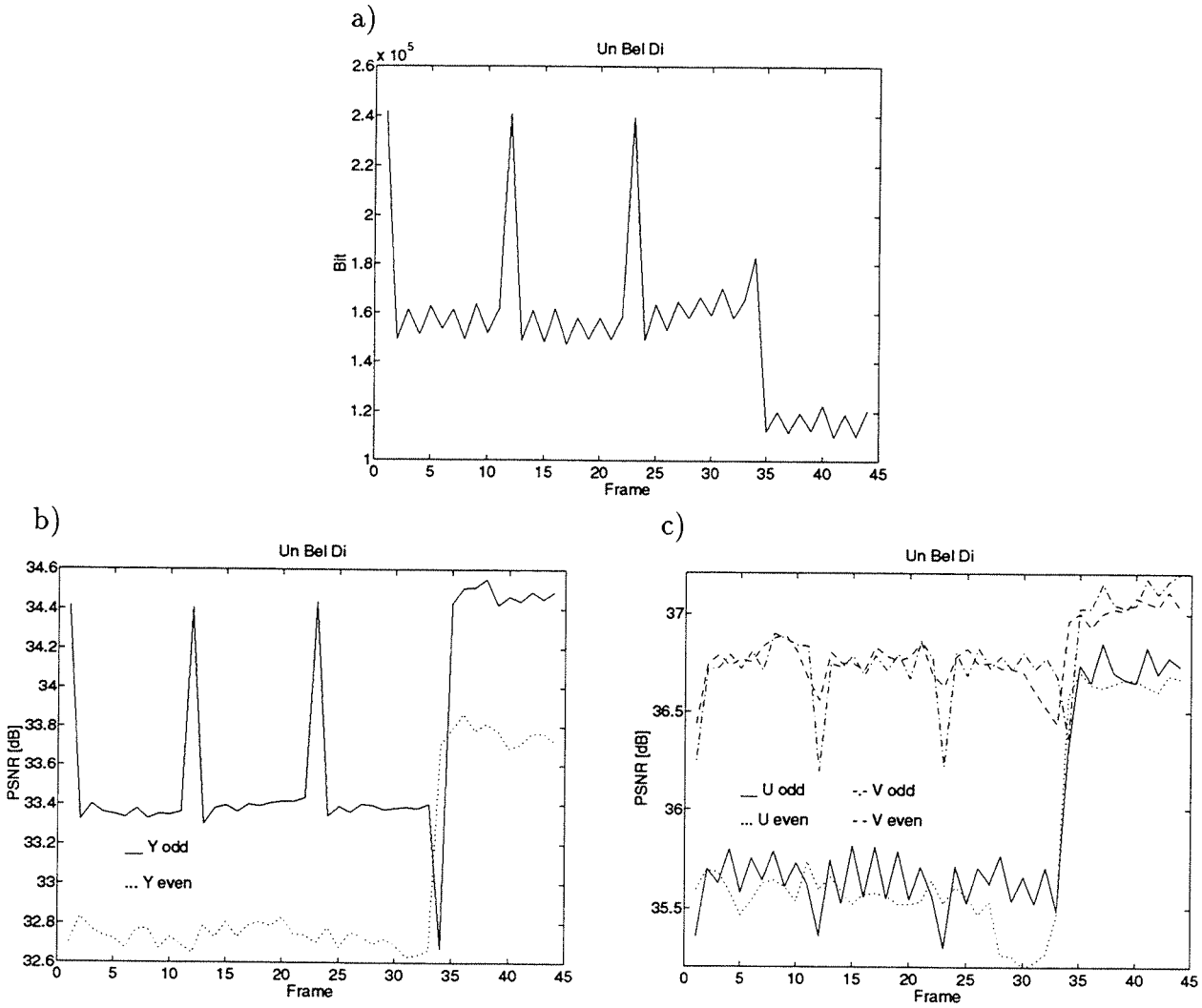

Figure 8.7: HDTV - "Un Bel Di": a) bit rate, b) PSNR luminance and c) PSNR chrominance.

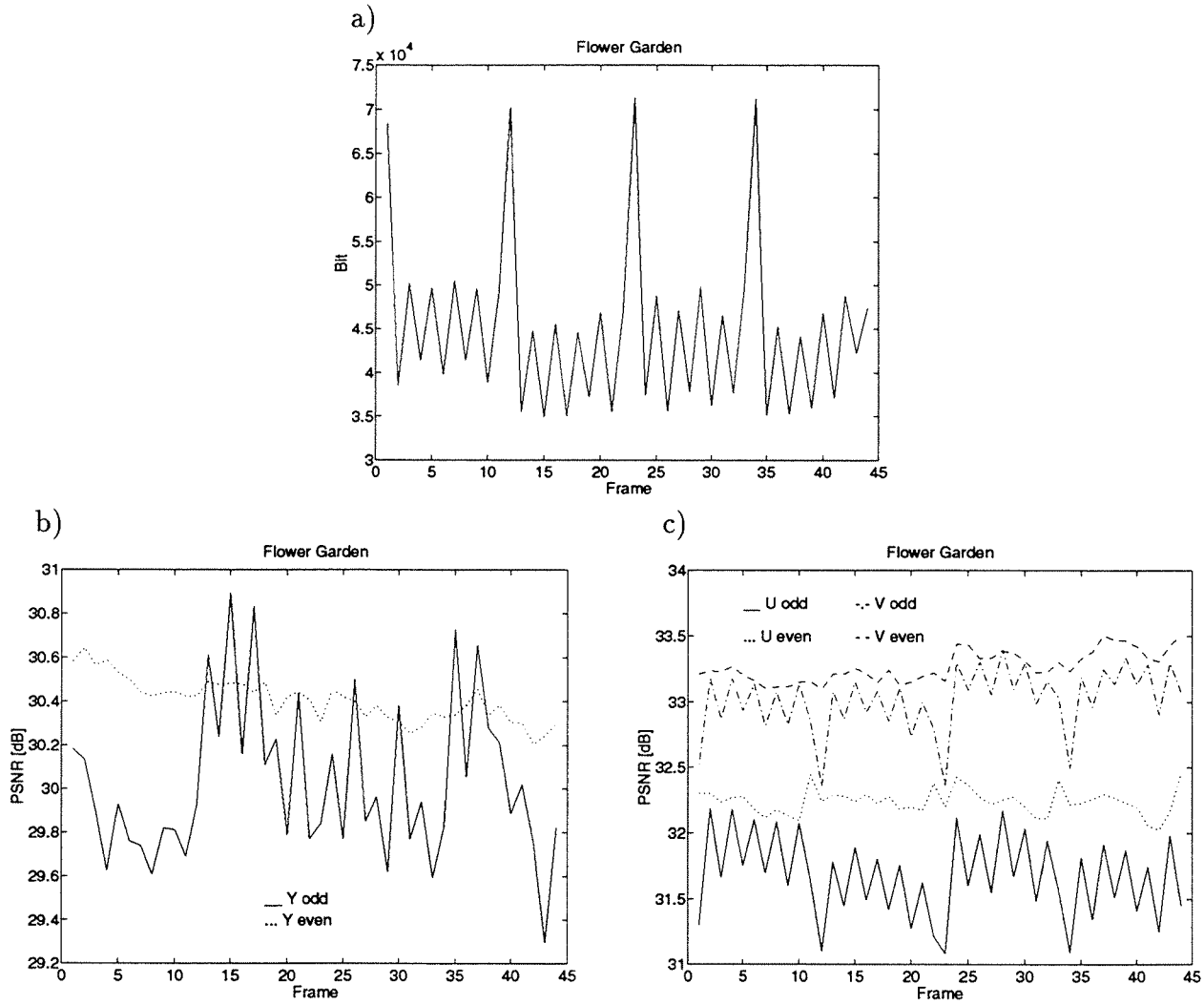

Figure 8.8: TV - "Flower Garden": a) bit rate, b) PSNR luminance and c) PSNR chrominance.

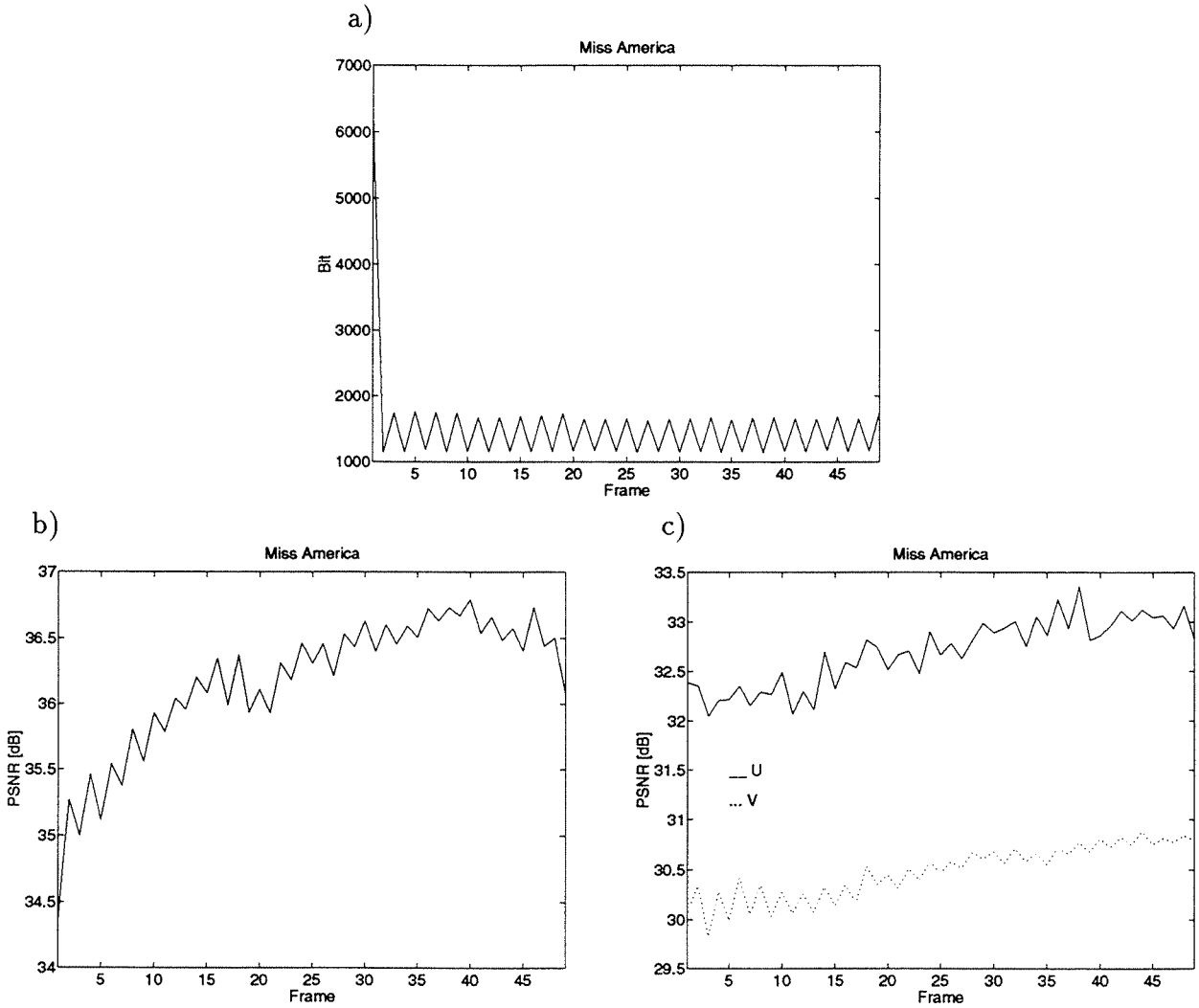

Figure 8.9: Video-phone - "Miss America": a) bit rate, b) PSNR luminance and c) PSNR chrominance.

8.9 Summary

In this chapter, a motion compensated wavelet transform based system for video coding applications has been presented. This codec is characterized by an inherent multiresolution data structure. It works independently from the input format resolution and can handle both interlaced and progressive sequences.

Due to the above features, the system can be used in very different applications. It leads to high performances on a wide range of bit rates. Furthermore, the bit stream can be partially decoded to reconstruct a lower resolution sequence. With these properties, the system is suitable for multimedia applications.

The application of the multigrid block matching motion estimation in the system leads to high performances in terms of both motion field accuracy and low overhead information. Furthermore, the entropy criterion controls efficiently the motion estimation procedure in order to achieve always the optimal segmentation given the allotted bandwidth.

Simulation results show a very good visual quality of the reconstructed sequences for HDTV, TV and video-phone applications and a wide range of bit rate from 30 Mb/s to 300 kb/s.

Chapter 9

Conclusions

9.1 Summary of developments and achievements

This chapter concludes the dissertation by summarizing the main developments and results of this work, and by indicating directions for further research.

This work has addressed the development of motion estimation techniques for image sequence coding applications. In this framework, the aims of a motion estimation algorithm can be defined as follows. First, it provides an accurate motion compensated prediction, especially along moving edges. Second, it requires a low amount of side information. Finally, it generates a robust and smooth motion field close to the true motion in the scene. These aims have been the guideline for the developments studied throughout the dissertation. Block matching motion estimation techniques have been chosen as the baseline of this study, due to their suitability for coding applications. The algorithms proposed throughout the dissertation have aimed at overcoming the drawbacks of the classical block matching methods, taking into account the above listed desired properties.

The state of the art of temporal redundancies reduction techniques and motion estimation algorithms have been first reviewed in Chap. 2. Next, a simulation environment has been defined in Chap. 3 in order to evaluate the performances of motion estimation algorithms.

A multigrid block matching motion estimation has been introduced in Chap. 4. Due to its multigrid structure, robust motion vectors are obtained on the coarse levels and are accurately refined on the finer ones. After the description of the algorithm, its most important features have been addressed in more details, namely the strategy to control the data flow within the multigrid structure and the up-/down-conversion operators to project the motion field from one grid to the consecutive one. Furthermore, its decreased computational complexity when compared to full-search has been demonstrated by computing the number of matching positions required in both cases. Extensive simulations have been carried out to assess the performances of the multigrid block matching algorithm. First, a comparison between the MAE and the MSE matching criteria showed that both perform similarly whereas the MAE requires a lower complexity. Next, the simulation results have shown that the highest performances are achieved by the fine-to-coarse-to-fine control strategy, the up-conversion by median filtering and the down-conversion by selecting the best initial condition among the motion vectors estimated in a neighborhood. Different sub-pixel accuracies of the motion vectors have also been compared. It has been concluded that the half-pixel accuracy corresponds to the best compromise between performances and complexity. Finally, the multigrid block matching algorithm has been compared to the full-search and classical fast search techniques. The multigrid algorithm is quasi-optimal in terms of minimizing the DFD energy, when compared to the full-search technique. In contrast, none of the fast search techniques leads to performances close to those provided by the full-search in this respect. Furthermore, the multigrid algorithm generates smooth

and robust motion fields close to the true motion in the scene. It results in a lower entropy of the motion vectors and consequently a lower amount of side information. Besides, the multigrid approach requires a greatly decreased computational complexity. In terms of coding performances, a gain is obtained both on the motion vectors information as the motion fields are smoother and on the DFD coding as the DFD have less discontinuities. The saving in terms of bit rate is up to 7% when compared to the full-search technique.

In Chap. 5 a local adaptation has been introduced in the multigrid block matching algorithm. By generating small blocks in detailed areas and large blocks in uniform ones, more accurate predictions are obtained along moving edges whereas the side information is globally decreased. After a description of the algorithm, the latter has been experimentally validated. Simulation results have shown the ability of the algorithm to either greatly decrease the side information while keeping an identical prediction accuracy, or to significantly improve the prediction without increasing the overhead information. A significant gain in terms of bit rate has been achieved with various coding schemes and sequences, this gain being up to 22% in the most favorable case. Furthermore, the method leads to sharper moving edges and therefore to an enhanced visual quality of the reconstructed sequence.

A VQ-based segmentation of the motion field has been introduced in Chap. 6. The block-based nature of the block matching motion estimation technique is a serious limitation. In order to relax this constraint, blocks where the block-based motion model fails are segmented in several regions, each of them being assigned a different motion vectors. The segmentation is approximated by a finite set of different patterns, leading to a low overhead information. Simulations results have shown a gain due to the method ranging from 5% to 20% in terms of bit rate. Although the gain in terms of PSNR is ranging from 0.5 to 1.5 dB only, the reconstructed sequences have demonstrated a greatly enhanced visual quality.

Chapter 7 has introduced an entropy criterion to optimize the motion estimation procedure. It is straightforward that a more accurate motion field leads to an improved prediction but needs a higher coding cost. Conversely, a less accurate motion field requires a lower coding cost but generates a poorer prediction. The entropy criterion evaluates the transmission cost relative to both the DFD and the motion information. By minimizing the sum of these two terms, it leads to the optimal motion estimation and compensation. Furthermore, it avoids the setting of arbitrary parameters (e.g. a threshold). The technique has been successfully applied in the locally adaptive multigrid block matching to control the split procedure and in the VQ-based motion field segmentation to decide whether a block is segmented. In both applications, simulations results have demonstrated that the entropy criterion outperforms a classical and simpler method based on a threshold. It always achieves a lower bit rate.

Finally, a generic video coding scheme has been presented in Chap. 8. Recent standards such as MPEG-I and -II have shown their efficiency in terms of picture quality for medium bit rate applications. However, these standards do not support straightforwardly important features such as generic coding, scalability and interactivity desired in emerging multimedia services. Furthermore, schemes based on the DCT produces annoying block artifacts in low bit rate applications. The proposed system is based on a motion compensated Gabor-like wavelet transform. Due to its multiresolution data structure, it is suitable for generic coding and the resulting bit stream is scalable (i.e. it can be partially decoded to reconstruct a lower resolution sequence). Furthermore, the artifacts produced by the wavelet transform are perceptually more acceptable than those of the DCT. For these reasons, the scheme overcomes the above drawbacks of MPEG and H.261 standards. The simulations results have shown a very good visual quality of the reconstructed sequences for HDTV, TV and video-phone applications and a wide range of bit rate from 30 Mb/s to 300 kb/s.

9.2 Possible extensions

Hereafter, a non-exhaustive list of possible extensions of this work is presented.

The first points are tightly linked to the proposed algorithms:

- The locally adaptive multigrid block matching motion estimation technique and the VQ-based motion field segmentation algorithm can be combined together. During the multigrid iterations, a block is either split, segmented by VQ or kept without changes. The algorithm requires the definition of multiple codebooks, depending on the block size at the level where the VQ-based segmentation is performed. It demands also an efficient decision rule to choose for each block between the three different possibilities. For this last operation, the use of the entropy criterion is a promising approach.

- As far as the VQ-based segmentation algorithm is concerned, the two following extensions seem interesting. First, the codebook can be generated by performing a training on segmented patterns (using any segmentation algorithms), instead of the current artificial generation. Second, the segmentation of a block can be extended to more than two regions.

- In order to refine the entropy criterion, a high order statistical model of the DFD can be introduced instead of the memoryless Laplacian model. Besides, the development of a non-stationary high order statistical model of the DFD could lead to

new and efficient techniques to code the DFD.

The following points are more general and represent a long term perspective:

- The algorithms discussed throughout the dissertation are concerned with local motion estimation. Their performances can be improved by the introduction of a global motion estimation. First, the global motion estimation computes the camera motion (e.g. zoom and pan). Then, the local motion estimation evaluates only the residual motion due to the displacement of the objects in the scene.

- In very low bit rate applications, a representation of the scene in terms of objects is more appropriate. In this context, an object-based motion estimation is more suitable. For this purpose, techniques matching features or objects instead of blocks can be defined.

- The motion model commonly adopted in block matching techniques constraints the motion to translations. More complex models can be introduced, for instance including rotations or affine transformations. In particular, those complex models seem more appropriate for object-based motion estimation.

Bibliography

[1] A.N. Netravali and J.O. Limb. Picture coding: a review. *Proc. IEEE*, vol. 68, no. 3, pp. 366-406, March 1980.

[2] A.K. Jain. Image data compression: a review. *Proc. IEEE*, vol. 69, no. 3, pp. 349-389, March 1981.

[3] H.G. Musmann, P. Pirsch, and H.J. Grallert. Advances in picture coding. *Proc. IEEE*, vol. 73, pp. 523-548, April 1985.

[4] ISO/IEC JTC1/SC2/WG10 Joint Picture Expert Group. JPEG technical specification, revision 8. Technical Report JPEG-8-R8, 1990.

[5] ISO/IEC JTC1 CD 11172. Information technology - Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s - Part 2: Coding of moving pictures information. Technical report, 1991.

[6] ISO/IEC JTC1/SC29/WG11 Motion Picture Expert Group. MPEG-II test model 4. Technical report, 1993.

[7] CCITT SG XV. Recommendation H.261 -video codec for audiovisual services at p*64kbit/s. Technical Report COM XV-R37-E, August 1990.

[8] A. Netravali and J.D. Robbins. Motion compensated television coding part I. *Bell Syst. Tech. Journal*, vol. 58, no. 3, pp. 629-668, April 1979.

[9] J.R. Jain and A.K. Jain. Displacement measurement and its application in interframe image coding. *IEEE Trans. Commun.*, vol. COM-29, no. 12, pp. 1799-1808, December 1981.

[10] T.S. Huang, editor. *Image Sequence Analysis.* Springer-Verlag, 1981.

[11] T.S. Huang, editor. *Image Sequence Processing and Dynamic Scene Analysis.* Springer-Verlag, 1983.

[12] J.K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - a review. *Proc. IEEE*, vol. 76, no. 8, pp. 917-935, August 1988.

[13] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artif. Intell.*, vol. 17, pp. 185-193, 1981.

[14] P. Anandan. A unified perspective on computational techniques for the measurement of visual motion. In *IEEE Proc. Int. Conf. Computer Vision*, pages 219–230, London, England, 1987.

[15] M. Bierling. Displacement estimation by hierarchical block matching. In *SPIE Proc. Visual Communications and Image Processing '88*, volume 1001, pages 942–951, Cambridge, MA, November 1988.

[16] M.H. Chan, Y.B. Yu, and A.G. Constantinides. Variable size block matching motion compensation with applications to video coding. *IEE Proc.*, vol. 137, no. 4, pp. 205-212, August 1990.

[17] A.N. Netravali and B.G. Haskell. *Digital Pictures Representation and Compression.* Plenum Press, New York, 1991.

[18] A. Habibi. Comparison of the nth-order DPCM encoder with linear transformations and block quantization techniques. *IEEE Trans. Commun. Tech.*, vol. COM-19, pp. 948-956, December 1971.

[19] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Trans. Comput.*, vol. C-23, pp. 90-93, January 1974.

[20] A.K. Jain. A sinusoidal family of unitary transforms. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-1, pp. 356-365, October 1979.

[21] J.W. Woods and S.D. O'Neil. Subband coding of images. *IEEE Trans. Acoust., Speech, and Signal Proces.*, vol. ASSP-34, no. 5, pp. 1278-1288, October 1986.

[22] S.G. Mallat. Multifrequency channel decomposition of images and wavelet models. *IEEE Trans. Acoust., Speech, and Signal Proces.*, vol. ASSP-37, no. 12, pp. 2091-2110, 1989.

[23] S.G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-11, no. 7, pp. 674-693, 1989.

[24] M. Rabbani and P.W. Jones. *Digital Image Compression Techniques.* SPIE Press, Bellingham, 1991.

[25] M. Kunt, A. Ikonomopoulos, and M. Kocher. Second generation image coding techniques. *Proc. IEEE*, vol. 73, no. 4, pp. 549-575, April 1985.

[26] M. Kunt, M. Benard, and R. Leonardi. Recent results in high compression image coding. *IEEE Trans. Circuits and Syst.*, vol. CAS-34, no. 11, pp. 1306-1336, November 1987.

[27] E. Dubois and S. Sabri. Noise reduction in image sequences using motion compensated temporal filtering. *IEEE Trans. Commun.*, vol. COM-32, no. 7, pp. 826-831, July 1984.

[28] C. Cafforio, F. Rocca, and S. Tubaro. Motion compensated image interpolation. *IEEE Trans. Commun.*, vol. COM-38, no. 2, pp. 215-222, February 1990.

[29] E. Dubois. Motion-compensated filtering of time-varying images. *Multidim. Syst. Sig. Process.*, vol. 3, pp. 211-239, 1992.

[30] D. LeGall. MPEG: A video compression standard for multimedia. *Commun. of the ACM*, vol. 34, no. 4, pp. 47-58, April 1991.

[31] D. LeGall. The MPEG video compression algorithm. *Signal Processing: Image Communications*, vol. 4, no. 2, pp. 129-140, April 1992.

[32] M. Gilge. A high quality videophone coder using hierarchical motion estimation and structure coding of the prediction error. In *SPIE Proc. Visual Communications and Image Processing '88*, volume 1001, pages 864–874, Cambridge, MA, November 1988.

[33] P. Strobach. Tree-structured scene adaptive coder. *IEEE Trans. Commun.*, vol. COM-38, no. 4, pp. 477-486, April 1990.

[34] Y.Q. Zhang and S. Zafar. Motion-compensated wavelet transform coding for color video compression. *IEEE Trans. Circuits and Systems for Video Technology*, vol. CSVT-2, no. 3, pp. 285-296, September 1992.

[35] S. Yao and R.J. Clarke. Motion-compensated wavelet coding of colour images using adaptive vector quantization. In *Proc. Int. Conf. on Image Processing: Theory and Applications*, pages 99–102, San Remo, Italy, June 1993.

[36] P.H. Westerink. *Subband Coding of Images.* PhD thesis, Delft University of Technology, The Netherlands, 1989.

[37] G. Karlsson and M. Vetterli. Three dimensional subband coding of video. In *IEEE Proc. ICASSP'88*, volume II, pages 1100–1103, New York, NY, April 1988.

[38] J.R. Ohm. Temporal domain subband video coding with motion compensation. In *IEEE Proc. ICASSP'92*, volume III, pages 229–232, San Francisco, CA, March 1992.

[39] W. Li and M. Kunt. Block apaptive 3D subband coding of image sequences. In *Proc. Int. Conf. on Image Processing: Theory and Applications*, pages 59–62, San Remo, Italy, June 1993.

[40] A.S. Lewis and G. Knowles. Video compression using 3D wavelet transforms. *Electron. Lett.*, vol. 26, no. 6, pp. 396-398, March 1990.

[41] T. Ebrahimi and M. Kunt. Image sequence coding using a three dimensional wavelet packet and adaptive selection. In *SPIE Proc. Visual Communications and Image Processing '92*, volume 1818, pages 222–232, Boston, MA, November 1992.

[42] M. Hoetter. Differential estimation of the global motion parameters zoom and pan. *Signal Processing*, vol. 16, pp. 249-265, March 1989.

[43] Y.T. Tse and R.L. Baker. Global zoom/pan estimation and compensation for video compression. In *IEEE Proc. ICASSP'91*, volume IV, pages 2725–2728, Toronto, Canada, May 1991.

[44] D. Adolph and R. Buschmann. 1.15 Mbit/s coding of video signals including global motion compensation. *Signal Processing: Image Communication*, vol. 3, nos. 2-3, pp. 259-274, June 1991.

[45] S.F. Wu and J. Kittler. A differential method for simultaneous estimation of rotation, change of scale and translation. *Signal Processing: Image Communication*, vol. 2, no. 1, pp. 69-80, May 1990.

[46] A. Zakhor and F. Lari. Edge based 3-D camera motion estimation with application to video coding. In M.I. Sezan and R.L. Lagendijk, editors, *Motion Analysis and Image Sequence Processing*, pages 89–123. Kluwer Academic Publishers, 1993.

[47] S. Gupta and A. Gersho. Joint motion compensated prediction and interpolation of video sequences. In *IEEE Proc. ICASSP'92*, volume III, pages 457–460, San Francisco, CA, March 1992.

[48] A. Nicoulin and M. Mattavelli. Entropy coding of displacement vector fields using a composite source model. *IEEE Trans. Circuits and Systems for Video Technology*, 1994. Submitted paper.

[49] H.Q. Nguyen and E. Dubois. Representation of motion vector fields for image coding. In *Picture Coding Symposium '90*, pages 8.4.1–8.4.5, Cambridge, MA, March 1990.

[50] A. Puri, R. Aravind, and B. Haskell. Adaptive frame/field motion compensated video coding. *Signal Processing: Image Communication*, vol. 5, nos. 1-2, pp. 39-58, February 1993.

[51] F.-M. Wang and D. Anastassiou. High-quality coding of the even fields based on the odd fields of interlaced video sequences. *IEEE Trans. Circuits and Syst.*, vol. CAS-38, pp. 140-142, January 1991.

[52] K.M. Uz, M. Vetterli, and D. LeGall. Interpolative multiresolution coding of advanced television with compatible subchannels. *IEEE Trans. Circuits and Systems for Video Technology*, vol. CSVT-1, no. 1, pp. 86-99, March 1991.

[53] F. Dufaux, I. Moccagatta, B. Rouchouze, T. Ebrahimi, and M. Kunt. Motion compensated generic coding of video based on multiresolution data structure. *Optical Engineering*, vol. 32, no. 7, pp. 1559-1570, July 1993.

[54] CCIR. Digital television coding parameters for studios, CCIR Recommendation 601-1. *Recommandations and Reports of the CCIR*, 1986.

[55] E.H. Adelson and J.R. Bergen. Spatio-temporal energy models for the perception of motion. *Journal of Optical Society of America*, vol. 2, no. 2, pp. 284-299, February 1985.

[56] D.C. Marr and S. Ullman. Directional selectivity and its use in early visual processing. *Proc. Royal Society London*, vol. B-211, pp. 151-180, March 1981.

[57] S. Ullman. Analysis of visual motion by biological and computer systems. *IEEE Computer*, vol. 14, no. 8, pp. 57-67, August 1981.

[58] D.R. Watson and A.J. Ahumada. Model of human visual-motion sensing. *Journal of Optical Society of America*, vol. 2, no. 2, pp. 322-342, February 1985.

[59] G.B. Townsend. *Pal Colour Television.* Cambridge University Press, 1970.

[60] P.G. Carnt and G.B. Townsend. *Colour Television, volume 2.* Iliffe, London, 1969.

[61] R.S. O'Brien, editor. *Color Television.* Society of motion picture and television engineers, New York, 1970.

[62] A. Verri and T. Poggio. Motion field and optical flow: qualitative properties. MIT, A.I. 917, December 1986.

[63] T.S. Huang and R.Y. Tsai. Image sequence analysis: motion estimation. In T.S. Huang, editor, *Image Sequence Analysis*, pages 104–124. Springer-Verlag, 1981.

[64] N. Mukawa. Estimation of shape, reflectance coefficients and illumination direction from image sequences. In *IEEE Proc. Int. Conf. on Computer Vision*, Osaka, Japan, December 1990.

[65] C.R. Moloney and E. Dubois. Estimation of motion fields from image sequences with illumination variation. In *IEEE Proc. ICASSP'91*, volume IV, pages 2425–2428, Toronto, Canada, May 1991.

[66] A. Pentland. Photometric motion. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-13, no. 9, pp. 879-890, September 1991.

[67] S. Negahdaripour and C.H. Yu. A generalized brightness change model for computing optical flow. In *IEEE Proc. Int. Conf. on Computer Vision*, pages 2–11, Berlin, Germany, May 1993.

[68] J.A. Stuller, A.N. Netravali, and J.D. Robbins. Intraframe television coding using gain and displacement compensation. *Bell Syst. Tech. Journal*, vol. 59, no. 7, pp. 1227-1241, September 1980.

[69] H. Nicolas. *Hiérarchie de modèles de mouvement et méthodes d'estimation associées. Application au codage de séquences d'images.* PhD thesis, University of Rennes I, France, 1992.

[70] N. Mukawa and H. Kuroda. Uncovered background prediction in interframe coding. *IEEE Trans. Commun.*, vol. COM-33, no. 11, pp. 1227-1231, November 1985.

[71] S. Brofferio. Videophone coding using background prediction. In *Proc. EUSIPCO 86*, pages 813–816, The Hague, The Netherlands, September 1986.

[72] D. Hepper and H. Li. Analysis of uncovered background prediction for image sequence coding. In *Picture Coding Symposium '87*, pages 192–193, Stockholm, Sweden, June 1987.

[73] D. Hepper. Efficiency analysis and application of uncovered background prediction in a low bit rate image coder. *IEEE Trans. Commun.*, vol. COM-38, no. 9, pp. 1578-1584, September 1990.

[74] X. Yuan. Hierarchical uncovered background prediction in a low bit-rate video coder. In *Picture Coding Symposium '93*, page 12.1, Lausanne, Switzerland, March 1993.

[75] N. Diehl. Object-oriented motion estimation and segmentation in image sequences. *Signal Processing: Image Communication*, vol. 3, no. 1, pp. 23-56, February 1991.

[76] P. Wagner and B. Girod. Region-based motion field estimation. In *Picture Coding Symposium '93*, page 4.5, Lausanne, Switzerland, March 1993.

[77] H. Sanson. Region based motion estimation and compensation for digital TV sequence coding. In *Picture Coding Symposium '93*, page 4.4, Lausanne, Switzerland, March 1993.

[78] H.G. Musmann, M. Hoetter, and J. Ostermann. Object-oriented analysis-synthesis coding of moving images. *Signal Processing: Image Communication*, vol. 1, no. 2, pp. 117-138, October 1989.

[79] M. Hoetter. Object-oriented analysis-synthesis coding based on moving two-dimensional objects. *Signal Processing: Image Communication*, vol. 2, no. 4, pp. 409-428, December 1990.

[80] B.G. Haskell. Frame-to-frame coding of television pictures using two-dimensional fourier transforms. *IEEE Trans. Inform. Theory*, vol. IT-20, no. 1, pp. 119-120, January 1974.

[81] D.J. Fleet and A.D. Jepson. Velocity extraction without form interpretation. In *Proc. 3rd Workshop on Computer Vision: Representation and Control*, pages 179–185, October 1985.

[82] D.J. Fleet and A.D. Jepson. Computation of normal velocity from local phase information. In *IEEE Proc. Conf. on Computer Vision and Pattern Recognition*, pages 379–386, San Diego, CA, August 1990.

[83] D.J. Heeger. Optical flow using spatio-temporal filters. *Int. Journal of Computer Vision*, vol. 1, no. 4, pp. 279-302, January 1988.

[84] R. Eagleson. Group-theoretic analysis of local flow characteristics while visually tracking a textured surface. In *Proc. Int. Conf. on Image Analysis and Processing*, Positano, Italy, September 1989.

[85] N. Cornelius and T. Kanade. Adapting optical flow to measure object motion in reflectance and x-ray image sequences. In *Proc. ACM SIGGRAPH/SIGART Interdisciplinary Workshop on Motion: Representation and Perception*, pages 50–58, Toronto, Canada, April 1983.

[86] H.-H. Nagel. Constraints for the estimation of displacement vector fields from image sequences. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 945–951, Karlsruhe, Germany, August 1983.

[87] H.-H. Nagel. Image sequence - ten (octal) years - from phenomenology towards a theoretical foundation. In *Proc. Int. Joint Conf. on Pattern Recognition*, pages 1174–1185, Paris, France, October 1986.

[88] F. Glazer. Multilevel relaxation in low-level computer vision. In A. Rosenfeld, editor, *Multiresolution Image Processing Analysis*, pages 312–330. Springer-Verlag, 1984.

[89] W. Enkelmann. Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences. *Computer Graphics and Image Processing*, vol. 43, pp. 150-177, August 1988.

[90] S. Sabri. Movement-compensated interframe prediction for NTSC colour TV signals. Bell Northern Res. Ltd., Montreal, Canada, Internal Rep., September 1982.

[91] C. Cafforio and F. Rocca. The differential method for motion estimation. In T.S. Huang, editor, *Image Sequence Processing and Dynamic Scene Analysis*, pages 104–124. Springer-Verlag, New York, 1983.

[92] D.R. Walker and K.R. Rao. Improved pel-recursive motion compensation. *IEEE Trans. Commun.*, vol. COM-32, no. 10, pp. 1128-1134, October 1984.

[93] J. Biemond, L. Looijenga, D.E. Boekee, and R.H.J.M. Plompen. A pel-recursive Wiener-based displacement estimation algorithm. *Signal Processing*, vol. 13, no. 4, pp. 399-412, December 1987.

[94] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro. Motion compensated interframe coding of video conferencing. In *Proc. Nat. Telecommun. Conf.*, pages G5.3.1–G5.3.5, New Orleans, LA, December 1981.

[95] H.C. Bergmann. Displacement estimation based on the correlation of image segments. In *IEEE Proc. Int. Conf. on Electronic Image Processing*, pages 215–219, York, England, July 1982.

[96] R. Srinivasan and K.R. Rao. Predictive coding based on efficient motion estimation. *IEEE Trans. Commun.*, vol. COM-33, no. 8, pp. 888-896, August 1985.

[97] S. Kappagantula and K.R. Rao. Motion compensated interframe image prediction. *IEEE Trans. Commun.*, vol. COM-33, no. 9, pp. 1011-1015, September 1985.

[98] A. Puri, H.-M. Hang, and D.L. Schilling. An efficient block-matching algorithm for motion-compensated coding. In *IEEE Proc. ICASSP'87*, volume 2, pages 1063–1066, Dallas, TX, April 1987.

[99] M. Ghanbari. The cross-search algorithm for motion estimation. *IEEE Trans. Commun.*, vol. COM-38, no. 7, pp. 950-953, July 1990.

[100] B. Liu and A. Zaccarin. New fast algorithms for the estimation of block motion vectors. *IEEE Trans. Circuits and Systems for Video Technology*, vol. CSVT-3, no. 2, pp. 148-157, April 1993.

[101] F. Dufaux and M. Kunt. Multigrid based motion estimation for interframe image sequence coding. In *Proc. EUSIPCO 92*, pages 1323–1326, Brussels, Belgium, August 1992.

[102] F. Dufaux and M. Kunt. Multigrid block matching motion estimation with an adaptive local mesh refinement. In *SPIE Proc. Visual Communications and Image Processing '92*, volume 1818, pages 97–109, Boston, MA, November 1992.

[103] T. Ebrahimi and F. Dufaux. Efficient hybrid coding of video for low bitrate applications. In *IEEE Proc. Int. Conf. on Commun.*, pages 522–526, Geneva, Switzerland, May 1993.

[104] F. Dufaux, T. Ebrahimi, and M. Kunt. Generic coding of video using packet wavelets and adaptive motion estimation. In *Proc. European Conf. on Circuit Theory and Design*, pages 1235–1240, Davos, Switzerland, September 1993.

[105] W. Li and F. Dufaux. Image sequence coding by multigrid motion estimation and segmentation based coding of prediction errors. In *SPIE Proc. Visual Communications and Image Processing'93*, volume 2094, pages 542–552, Cambridge, MA, November 1993.

[106] H. Watanabe and S. Singhal. Windowed motion compensation. In *SPIE Proc. Visual Communications and Image Processing '91*, volume 1605, pages 582–589, Boston, MA, November 1991.

[107] S. Nogaki and M. Ohta. An overlapped block motion compensation for high quality motion picture coding. In *IEEE Proc. Int. Symp. on Circuits and Systems*, pages 184–187, San Diego, CA, May 1992.

[108] C. Auyeung, J. Kosmach, M. Orchard, and T. Kalafatis. Overlapped block motion compensation. In *SPIE Proc. Visual Communications and Image Processing'92*, volume 1818, pages 561–571, Boston, MA, November 1992.

[109] M. Mattavelli, A. Nicoulin, and G. Fernandez. Overlapped motion compensation in hybrid video coding systems. In *Int. Workshop on HDTV'93*, Ottawa, Canada, October 1993.

[110] G.J. Sullivan and R.L. Baker. Motion compensation for video compression using control grid interpolation. In *IEEE Proc. ICASSP'91*, volume IV, pages 2713–2716, Toronto, Canada, May 1991.

[111] Y. Nakaya and H. Harashima. An iterative motion estimation method using triangular patches for motion compensation. In *SPIE Proc. Visual Communications and Image Processing '91*, volume 1605, pages 546–557, Boston, MA, November 1991.

[112] M.T. Orchard. Predictive motion field segmentation for image sequence coding. *IEEE Trans. Circuits and Systems for Video Technology*, vol. CSVT-3, no. 1, pp. 54-70, February 1993.

[113] I. Moccagatta, F. Dufaux, and M. Kunt. Motion field segmentation and coding by means of vector quantization. In *Picture Coding Symposium '93*, page 4.1, Lausanne, Switzerland, March 1993.

[114] F. Dufaux, I. Moccagatta, F. Moscheni, and H. Nicolas. Vector quantization based motion field segmentation under the entropy criterion. *Submitted to Visual Communication and Image Representation*, 1994.

[115] C.S. Fuh and P. Maragos. Affine models for image matching and motion detection. In *IEEE Proc. ICASSP'91*, volume IV, pages 2409–2412, Toronto, Canada, May 1991.

[116] V. Seferidis and M. Ghanbari. General approach to block-matching motion estimation. *Optical Engineering*, vol. 32, no. 7, pp. 1464-1474, July 1993.

[117] CCIR. Method for subjective assessment of the quality of television pictures. *13th Plenary Assembly, Recommandation 500*, vol. 11, pp. 65-68, 1974.

[118] D. De Vleeschauwer and I. Bruyland. Nonlinear interpolators in compatible HDTV image coding. In L. Chiariglione, editor, *Signal Processing of HDTV*. North-Holland, Amsterdam, 1988.

[119] D.M. Martinez and J.S. Lim. Spatial interpolation of interlaced television pictures. In *IEEE Proc. ICASSP'89*, volume 3, pages 1886–1889, Glasgow, Scotland, May 1989.

[120] D.K. Jensen and D. Anastassiou. Spatial resolution enhancement of images using nonlinear interpolation. In *IEEE Proc. ICASSP'90*, Albuquerque, NM, April 1990.

[121] T. Doyle. Interlaced to sequential conversion for EDTV applications. In L. Chiariglione, editor, *Signal Processing of HDTV*. North-Holland, Amsterdam, 1988.

[122] F.-M. Wang, D. Anastassiou, and A. Netravali. Time-recursive deinterlacing for IDTV and pyramid coding. *Signal Processing: Image Communication*, vol. 2, no. 3, pp. 365-374, October 1990.

[123] F. Dufaux, T. Ebrahimi, I. Moccagatta, T.G. Campbell, A. Geurtz, and M. Kunt. Interlaced to progressive conversion. Technical Report VADIS/A3/TD16, EPFL, April 1991.

[124] B. Rouchouze, F. Dufaux, and M. Kunt. Interlaced image sequence coding for digital TV. In *SPIE Proc. Applications of Digital Image Processing XV*, volume 1771, pages 470–478, San Diego, CA, July 1992.

[125] T. Ebrahimi. *Perceptually Derived Localized Linear Operators: Application to Image Sequence Compression.* PhD thesis, Swiss Federal Institute of Technology, Lausanne, Switzerland, 1992.

[126] T. Ebrahimi, F. Dufaux, I. Moccagatta, B. Rouchouze, P. Cicconi, E. Reusens, and M. Kunt. Hybrid video coding based on an efficient subband/wavelet transform and arithmetic coding for generic applications. In *Picture Coding Symposium '93*, page 11.6, Lausanne, Switzerland, March 1993.

[127] R.M. Witten, I.H. Neal, and J.G. Cleary. Arithmetic coding for data compression. *Commun. of the ACM*, vol. 30, no. 6, pp. 520-540, June 1987.

[128] T. Ebrahimi and M. Kunt. Image compression by Gabor expansion. *Optical Engineering*, vol. 30, no. 7, pp. 873-880, July 1991.

[129] T. Ebrahimi and M. Kunt. Application of a perceptually based localized and fast wavelet transform in image compression. In *IEEE Proc. ICASSP'92*, San Francisco, CA, March 1992.

[130] A.K. Jain. Advances in mathematical models for image processing. *Proc. IEEE*, vol. 69, no. 5, pp. 502-528, May 1981.

[131] W. Li and F.X. Mateo. Segmentation based coding of motion compensated prediction error images. In *IEEE Proc. ICASSP'93*, volume V, pages 357–360, Minneapolis, MN, April 1993.

[132] M. Eden and M. Kocher. On the performance of a contour coding algorithm in the context of image coding Part I: contour segment coding. *Signal Processing*, vol. 8, no. 10, pp. 381-386, 1985.

[133] N. Baaziz. *Approches d'estimation et de compensation de mouvement multirésolutions pour le codage de séquences d'images.* PhD thesis, University of Rennes I, France, 1991.

[134] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Trans. Commun.*, vol. COM-31, no. 4, pp. 482-540, April 1983.

[135] J.L. Crowley and R.M. Stern. Fast computation of the difference of low-pass transform. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, no. 2, pp. 212-222, March 1984.

[136] J. Kowalczuk, R. Hervigo, T. Ebrahimi, M. Mattavelli, D. Mlynek, and M. Kunt. Hardware evaluation of EPFL proposal for MPEG-II. Technical Report 40, ISO-IEC/JTC1/SC29/WG11, Kurihama, Japan, November 1991.

[137] R. Hervigo, J. Kowalczuk, and D. Mlynek. A multiprocessors architecture for HDTV motion estimation system. *IEEE Trans. Consumer Electronics*, vol. 38, no. 3, pp. 690-697, August 1992.

[138] J. Kowalczuk, R. Hervigo, T. Ebrahimi, F. Dufaux, I. Moccagatta, D. Mlynek, and M. Kunt. VLSI implementation for a new video codec. In *Proc. EUSIPCO 92*, pages 1517–1520, Brussels, Belgium, August 1992.

[139] W. Hackbusch and U. Trottenberg, editors. *Multigrid Methods.* Lecture Notes in Mathematics, Springer-Verlag, New York, 1982.

[140] W. Hackbusch. *Multigrid Methods and Applications.* Springer Series in Comp. Math., 1985.

[141] K. Stueben and U. Trottenberg. Multigrid methods: Fundamental algorithms, model problem analysis and applications. In *Proc. of the Conf. on Multigrid Methods*, pages 1–176, Cologne, Germany, November 1981.

[142] R.E. Bank. A-posteriori error estimates, adaptive local mesh refinement and multigrid iteration. In *Proc. of the 2nd European Conf. on Multigrid Methods*, pages 7–22, Cologne, Germany, October 1985.

[143] D. Terzopoulos. Image analysis using multigrid relaxation methods. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, no. 2, pp. 129-139, March 1986.

[144] A. Brandt, D. Ron, and D.J. Amit. Multi-level approaches to discrete-state and stochastic problems. In *Proc. of the 2nd European Conf. on Multigrid Methods*, pages 66–99, Cologne, Germany, October 1985.

[145] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, vol. 220, pp. 671-680, May 1983.

[146] G. Madec. Half pixel accuracy in block matching. In *Picture Coding Symposium '90*, Cambridge, MA, March 1990.

[147] S.L. Iu. Comparison of motion compensation using different degrees of sub-pixel accuracy for interfield/interframe hybrid coding of HDTV image sequences. In *IEEE Proc. ICASSP'92*, volume III, pages 465–468, San Francisco, CA, March 1992.

[148] D.H. Hubel and T. Weisel. Brain mechanisms of vision. *Sci. Amer.*, pages 130–146, September 1979.

[149] S.L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *J. Assoc. Comput. Mach.*, vol. 23, pp. 368-388, 1976.

[150] C. Labit and H. Nicolas. Compact motion representation based on global features for semantic image sequence coding. In *SPIE Proc. Visual Communications and Image Processing '91*, volume 1605, pages 697–709, Boston, MA, November 1991.

[151] R.M. Gray. Vector quantization. *IEEE ASSP Magazine*, vol. 1, no. 2, pp. 4-29, April 1984.

[152] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, 1992.

[153] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84-95, January 1980.

[154] F. Moscheni, F. Dufaux, and H. Nicolas. Entropy criterion for optimal bit allocation between motion and prediction error information. In *SPIE Proc. Visual Communications and Image Processing '93*, volume 2094, pages 235–242, Cambridge, MA, November 1993.

[155] W.K. Pratt. *Digital Image Processing.* Wiley, New York, 1991.

[156] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, New York, 1965.

[157] U. Grenander and G. Szego. *Toepliz Forms and Their Applications.* University of California Press, Berkeley, 1958.

[158] G. Arfken. *Mathematical Methods for Physicists.* Academic Press, 1970.

[159] T. Ebrahimi, F. Dufaux, I. Moccagatta, P. Cicconi, and M. Kunt. EPFL proposal for MPEG-II. Technical Report 40, ISO-IEC/JTC1/SC29/WG11, Kurihama, Japan, November 1991.

[160] A. Lippman. Feature sets for interactive images. *Commun. of the ACM*, vol. 34, no. 4, pp. 93-102, April 1991.

[161] J. G. Daugman. Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression. *IEEE Trans. Acoust., Speech, and Signal Proces.*, vol. ASSP-36, no. 7, pp. 1169-1179, July 1988.

[162] R. Coifman, Y. Meyer, D. Quake, and V. Wickerhauser. Acoustic signal compression with wave packets. In *Wavelet Workshop*, Marseille, France, October 1990.

[163] T. Berger. Optimum quantizers and permutation codes. *IEEE Trans. Inform. Theory*, vol. IT-18, no. 6, November 1972.

[164] B. Macq. *Perceptual transforms and universal entropy coding for an integrated approach to picture coding.* PhD thesis, Université Catholique de Louvain, Louvain-la-Neuve, Belgium, 1989.

[165] I. Moccagatta and M. Kunt. A pyramidal vector quantization approach to transform domain. In *EUSIPCO'92*, pages 1365–1368, Brussels, Belgium, August 1992.

[166] D.A. Huffman. A method for the reconstruction of minimum redundancy codes. *Proc. IRE*, vol. 40, pp. 1098-1101, 1952.

[167] J. Rissanen and G.G. Langdon. Arithmetic coding. *IBM J. Res. Develop.*, vol. 23, no. 2, pp. 149-162, March 1979.

[168] G.G. Langdon. An introduction to arithmetic coding. *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 135-149, March 1984.

Curriculum Vitae

Frédéric Dufaux est né le 17 août 1967 à Bienne en Suisse. Il a obtenu le diplôme d'ingénieur physicien de l'Ecole Polytechnique Fédérale de Lausanne (EPFL) en 1990.

En 1990, il a rejoint le Laboratoire de Traitement des Signaux de l'EPFL en qualité d'assistant et de doctorant. Il a participé à la recherche sur le codage de séquences d'images et à la proposition de l'EPFL pour MPEG-II. Il a également travaillé pour le projet "Massively Parallel Processing Collaboration" (MPPC). Durant l'été 1992, il a été un visiteur aux laboratoires AT&T Bell à Murray Hill, New Jersey.

Ses principaux intérêts de recherche portent sur le traitement et le codage de séquences d'images, l'estimation de mouvement et le calcul parallel. Il est l'auteur ou co-auteur de plusieurs publications scientifiques et possède un brevet. Il est membre de EURASIP et de IEEE.

Frédéric Dufaux was born in Bienne, Switzerland, on August 17, 1967. He received the M.S. in physics from the Swiss Federal Institute of Technology at Lausanne (EPFL) in 1990.

In 1990, he joined the Signal Processing Laboratory in EPFL as a research assistant and a Ph.D. student. He participated to the research on video coding and to the MPEG-II proposal of the EPFL. He worked also on the Massively Parallel Processing Collaboration (MPPC) project. During the summer 1992, he was a visiting researcher at the Advanced Video Technology Department of the AT&T Bell Laboratories, Murray Hill, New Jersey.

His main research interests are in image sequence processing and coding, motion estimation and parallel processing. He is the author or co-author of several research publications and holds one patent. He is a member of EURASIP and IEEE.