Techniques for improving the performance of distributed video coding

Abdalbassir Abou-Elailah

École Doctorale
d'Informatique,
Télécommunications
et Électronique de Paris

Thèse

présentée pour obtenir le grade de docteur
de Télécom ParisTech

Spécialité : **Signal et Images**

Abdalbassir ABOU-ELAILAH

Techniques d'amélioration des performances de compression dans le cadre du codage vidéo distribué.

—

Techniques for improving the performance of distributed video coding.

Soutenue le 14 Décembre 2012 devant le jury composé de :

Touradj	Ebrahimi	Professeur, EPFL	Président
Fernando	Pereira	Professeur, Instituto Superior Técnico	Rapporteurs
Lina	Karam	Professeur, Arizona State University	
Eric	Nassor	Manager, Canon Research Centre France	Examinateur
Frédéric	Dufaux	Directeur de recherche CNRS, Télécom ParisTech	Directeurs de Thèse
Marco	Cagnazzo	Maître de conférence, Télécom ParisTech	
Béatrice	Pesquet-Popescu	Professeur, Télécom ParisTech	
Joumana	Farah	Professeur, Université Saint-Esprit De Kaslik	

Remerciements

Je tiens à adresser mes chaleureux remerciements à mon directeur de thèse Dr. Frédéric DUFAUX, pour avoir dirigé d'une façon continue mes activités de recherche. J'ai profondément apprécié les discussions constructives avec lui ainsi que la confiance qu'il m'a accordée pendant ce travail de thèse. J'ai également apprécié ses relectures, remarques, commentaires et corrections de mes travaux de recherche.

Je remercie très sincèrement mon co-directeur de thèse Dr. Marco CAGNAZZO pour la proposition de mon sujet de thèse, pour l'encadrement de mes activités de recherche et pour la riche compréhension qu'il m'a apportée du codage vidéo distribué.

Je tiens également à remercier Prof. Béatrice PESQUET-POPESQU, chef du groupe MultiMedia à Telecom Paristech, pour son accueil chaleureux dans son groupe de recherche, pour l'encadrement professionnel de mes activités de recherche et pour ses relectures et remarques constructives de mes travaux.

Je souhaite remercier très chaleureusement Prof. Joumana FARAH, pour avoir encadré mon travail de recherches à distance, pour sa confiance, ses encouragements et ses conseils techniques et pratiques. J'ai particulièrement apprécié les discussions avec elle, les relectures et les corrections précieuses de mes articles et de mon manuscrit.

Mes remerciements s'adressent aux rapporteurs : Prof. Lina KARAM et Prof. Fernando PEREIRA et aux examinateurs : Prof. Touradj EBRAHIMI (président du jury) et Dr. Eric NASSOR qui avoir accepté d'évaluer et de rapporter mon travail de recherches, et pour leur participation au Jury de soutenance.

Je remercie aussi Prof. Michel KIEFFER et Dr. Joel JUNG pour avoir effectué l'évaluation de mi-parcours de mon travail de thèse.

Je tiens à remercier Eng. Julien LE TANOUR, pour son aide et sa collaboration dans la compréhension et l'application des courbes élastiques.

Je voudrais également remercier le personnel technique et administratif de Telecom Paristech, pour leur support pendant ma thèse.

Ma gratitude s'adresse également à mes collègues durant ma thèse à Telecom Paricteh : Thomas, Giovanni, Claudio, Rafael, Mounir, Irina, Manel, Elie, Hamlet, Giuseppe, Marc, Paul et Alper qui m'ont aidé à répondre à un certain nombre de questions pratiques, techniques et scientifiques.

Enfin, un grand merci à mes parents, mes frères, mes sœurs et mes amis qui sont

toujours à mes côtés, et qui m'ont continuellement soutenu pendant ces années de travail de thèse.

Abdalbassir ABOU-ELAILAH

Résumé

Le codage vidéo distribué (DVC) est une technique récemment proposée dans le cadre du codage et de la transmission des séquences vidéo, et qui convient surtout à une nouvelle classe d'applications telles que la surveillance vidéo sans fil, les réseaux de capteurs multimédia, les caméras PC sans fil, les téléphones mobiles et les appareils-photos numériques. Ces applications nécessitent en effet un encodeur de faible complexité, avec la possibilité d'un décodeur de complexité élevée. DVC présente plusieurs avantages : d'abord, la complexité peut être distribuée entre l'encodeur et le décodeur. Deuxièmement, le DVC est robuste aux erreurs, car un codeur de canal y est incorporé. En DVC, une information adjacente (Side Information ou SI) est estimée au décodeur en se basant sur les trames décodées disponibles, et utilisée pour le décodage et la reconstruction des autres trames.

Dans cette thèse, nous proposons de nouvelles techniques qui permettent d'améliorer la qualité de l'information adjacente. Tout d'abord, le raffinement itératif de l'information adjacente est réalisé après le décodage de chaque sous-bande DCT, en utilisant la trame Wyner-Ziv (WZ) partiellement décodée (appelée PDWZF) avec les trames de référence. De plus, dans cet algorithme, une nouvelle approche est proposée qui permet d'adapter la fenêtre de recherche au niveau de mouvement courant entre la trame WZ et les trames de référence, en se basant sur la PDWZF obtenue après le décodage de la première sous-bande DCT. Ensuite, une nouvelle méthode de génération de l'information adjacente est proposée, qui utilise l'estimation des vecteurs de mouvement dans les deux sens et le raffinement Quad-tree. En outre, en vue d'améliorer la qualité des trames WZ décodées pour les grandes tailles de GOP (Group Of Pictures), un algorithme basé sur les trames adjacentes décodées est proposé, qui utilise une zone de recherche adaptative et une taille de bloc variable.

Une autre contribution de cette thèse concerne la fusion des estimations globale et locale. Les paramètres globaux sont calculés au codeur en utilisant l'algorithme SIFT. Ces paramètres globaux sont transmis au décodeur pour y être utilisés dans l'estimation de l'information adjacente globale. Ensuite, de nouvelles approches sont proposées afin de combiner les estimations de mouvement globale et locale. Dans la première approche, la fusion se base sur les différences entre les blocs correspondants. Dans la seconde, la technique SVM (Support Vector Machine) est utilisée pour combiner les deux informations adjacentes. En plus, des algorithmes sont proposés pour améliorer la fusion au cours du

décodage, par l'exploitation de la PDWZF et des coefficients DC décodés. En outre, les objets segmentés des trames de référence sont utilisés dans la combinaison des estimations de mouvement globale et locale, en utilisant les courbes élastiques et la compensation de mouvement basée-objets.

De nombreuses simulations ont été effectuées pour tester les performances des techniques proposés et qui montrent des gains importants par rapport au codeur classique DISCOVER. Par ailleurs, les performances de DVC obtenues en appliquant les algorithmes proposés surpassent celles de H.264/AVC Intra et H.264/AVC No motion pour les séquences testées. En plus, l'écart vis-à-vis de H.264/AVC Inter avec une configuration IB...IB est considérablement réduit.

Abstract

Distributed Video Coding (DVC) is a recently proposed paradigm in video communication, which fits well emerging applications such as wireless video surveillance, multimedia sensor networks, wireless PC cameras, and mobile cameras phones. These applications require a low complexity encoding, while possibly affording a high complexity decoding. DVC presents several advantages: First, the complexity can be distributed between the encoder and the decoder. Second, the DVC is robust to errors, since it uses a channel code. In DVC, a Side Information (SI) is estimated at the decoder, using the available decoded frames, and used for the decoding and reconstruction of other frames.

In this Ph.D thesis, we propose new techniques in order to improve the quality of the SI. First, successive refinement of the SI is performed after each decoded DCT band, using a Partially Decoded WZF (PDWZF), along with the reference frames. Moreover, in this refinement approach an adaptive search area algorithm is also proposed, that allows adapting the search area to the current motion between the WZF and the reference frames, using the PDWZF obtained after decoding the first DCT band. Then, a new scheme for SI generation based on backward, forward motion estimations, and Quad-tree refinement is proposed. Furthermore, in the aim of enhancing the quality of the decoded WZFs for larger GOP sizes, an algorithm based on adjacent decoded frames is investigated, using an adaptive search area and a variable block size.

Another contribution of this thesis concerns a fusion of global and local SI. Global parameters are estimated at the encoder using the Scale-Invariant Feature Transform (SIFT) algorithm. These global parameters are sent to the decoder to estimate the global SI. Then, new methods for combining global and local motion estimations are proposed, to further improve the SI. In the first approach, the differences between the corresponding blocks are used to combine the global and local SI frames. In the second approach, Support Vector Machine (SVM) is used to combine the two SI frames. In addition, algorithms are proposed to refine the fusion during the decoding process by exploiting the PDWZF and the decoded DC coefficients. Furthermore, the foreground objects are used in the combination of the global and local motion estimations, using elastic curves and foreground objects motion compensation.

Extensive experiments have been conducted showing that important gains are obtained by the proposed techniques compared to the classical DISCOVER codec. In addition, the

performance of DVC applying the proposed algorithms outperforms now the performance of H.264/AVC Intra and H.264/AVC No motion for tested sequences. Besides that, the gap with H.264/AVC in an Inter IB...IB configuration is significantly reduced.

Résumé en français

Introduction

La compression est une tâche essentielle dans les systèmes de communication; elle vise la réduction du volume des données à stocker ou à transmettre. Elle reste une étape de traitement indispensable, malgré le fait que le débit des réseaux continue de croitre. Récemment, dans le rapport Cisco [1], il a déclaré que le débit total de toutes les formes de vidéo sur IP atteindra bientôt 86 % du trafic IP global. De plus, ce rapport montre qu'en 2016, 1.2 millions de minutes de contenu vidéo traversera chaque seconde le réseau.

Par conséquent, la visualisation des données de haute qualité en temps réel et le stockage d'énormes quantités de données dans moins d'espace deviennent des défis majeurs. Concernant le codage vidéo numérique, les efforts de normalisation de l'ISO/IEC MPEG-x et de l'UIT-T H.26x sont principalement basés sur la Transformée en Cosinus Discrète (DCT) et le codage prédictif intra-trame et inter-trame, en vue de comprimer les séquences vidéos. En plus, le Codage Vidéo à Haute Efficacité (HEVC) est actuellement en développement et se présente comme un successeur du codeur H.264/AVC. HEVC permet de réduire de moitié la bande passante de la vidéo. Dans tous ces systèmes classiques de compression vidéo, l'encodeur exécute un grand nombre d'opérations à cause de l'étape d'estimation de mouvement dans le codage des trames prédictives, afin d'exploiter les corrélations spatiale et temporelle. Par contre, le décodeur utilise simplement les vecteurs de mouvement reçus pour reconstruire l'image décodée. Ce schéma de conception asymétrique est tout à fait adapté pour certaines applications, telles que la télévision numérique, le téléchargement sur des mobiles à partir de serveurs, ... etc.

Le codage vidéo distribué (DVC) est une technique récente qui permet de transmettre des vidéos avec une répartition flexible de la complexité de calcul entre l'encodeur et le décodeur. En particulier, DVC permet un encodage avec une très faible complexité. Cette propriété est particulièrement intéressante pour une nouvelle classe d'applications caractérisées par une liaison montante, telles que les réseaux de capteurs de faible puissance, les caméras de surveillance sans fil ou encore les appareils de communication mobile.

Le DVC est basé sur le théorème de Slepian et Wolf [2] : étant données deux sources corrélées X et Y, avec Y l'information adjacente, comprimée à sa limite entropique $H(Y)$, X peut être transmise à un débit très proche de l'entropie conditionnelle $H(X|Y)$. Le

codage Wyner-Ziv (WZ) [3] est une application de ce concept au cas du codage source avec pertes.

Le codeur de DISCOVER [4, 5] est l'un des codeurs les plus efficaces proposés pour le DVC, et qui est basé sur le schéma de Stanford [6]. Dans ce codeur, les trames de la séquence vidéo sont divisées en deux groupes : les trames clés et les trames WZ (WZFs). Le groupe d'images (GOP) est défini comme la distance entre deux trames clés consécutives. Les trames clés sont directement encodées et décodées en mode intra (H.264/AVC intra). Pour les WZFs, une transformation DCT est préalablement appliquée, suivie d'une quantification uniforme. Les valeurs quantifiées sont ensuite séparées en plans de bits qui sont encodés avec un code LDPCA (Low-Density Parity Check Accumulate code) ou avec un Turbo-code. Au décodeur, une information adjacente (SI) est générée par une interpolation temporelle compensée en mouvement (MCTI) [7] des trames précédemment décodées. La SI est considérée comme une version bruitée de la WZF encodée. Finalement, la SI est exploitée dans le décodeur du canal, conjointement avec les bits de parité des WZFs obtenus par des requêtes successives via un canal de retour, afin de reconstruire les plans de bits, et, par conséquent, la séquence vidéo décodée.

Malgré des progrès considérables ces dernières années dans le cadre de DVC, les performances débit-distorsion (RD) restent en deçà des attentes. En effet, les performances du système restent très dépendantes de la qualité de la SI générée au décodeur. A ce but, nous concentrons ce travail autour de l'amélioration de la SI, en proposant plusieurs approches permettant d'améliorer les performances du DVC. Les approches proposées sont présentées dans ce manuscrit comme suit:

- **Chapitre 3 - Amélioration progressive de l'information Adjacente :** Dans ce chapitre, la méthode proposée consiste à améliorer itérativement la SI après le décodage de chaque sous-bande DCT. Dans ce cas, la SI initiale est estimée en utilisant la technique MCTI. Plus spécifiquement, à chaque itération, le décodeur se sert de la trame WZ partiellement décodée (PDWZF), conjointement avec les trames de référence adjacentes, afin d'améliorer la SI. Pour cette dernière opération, la fiabilité des vecteurs de mouvement est tout d'abord vérifiée. Ensuite, les vecteurs considérés comme suspects sont recalculés par une estimation de mouvement bidirectionnelle. Dans ce cadre, nous proposons deux algorithmes différents pour la phase de ré-estimation des vecteurs de mouvement des blocs suspects. Enfin, le mode de compensation de mouvement optimal est sélectionné.

- **Chapitre 4 - Techniques pour l'amélioration de l'information adjacente :** Dans ce chapitre, nous proposons trois techniques différentes en vue d'améliorer la SI, et par conséquent les performances du DVC. La première approche vise à générer la SI en utilisant une estimation des vecteurs du mouvement dans les deux sens et un raffinement quad-tree.

Ensuite, une nouvelle approche consiste à adapter la fenêtre de recherche au mouvement courant entre la WZF et les trames de référence. Cette adaptation est effectuée après le décodage de la première sous-bande DCT en utilisant la PDWZF et les trames de référence. La SI est enfin raffinée après le décodage de chaque sous-bande DCT en utilisant la fenêtre de recherche adaptée.

Finalement, nous proposons une nouvelle approche qui consiste à ré-estimer la SI après le décodage de toutes les WZFs à l'intérieur du GOP courant, en utilisant la trame décodée et les trames précédente et suivante, pour une large taille de GOP (4 et 8). Plus spécifiquement, nous utilisons une taille variable des blocs durant cette estimation. Cette méthode permet d'améliorer de façon significative la qualité de la SI et, par conséquent, la qualité de la trame décodée.

- **Chapitre 5 - Fusion de l'estimation de mouvement globale et locale :** Dans ce chapitre, nous proposons une nouvelle méthode qui consiste à combiner deux estimations de mouvement globale et locale au décodeur, en vue d'améliorer la SI finale. L'estimation globale, appelée GMC SI, est réalisée en utilisant les paramètres d'une transformation affine envoyés par l'encodeur. Ces paramètres sont estimés en se basant sur un matching des composantes SIFT (Scale-Invariant Feature Transform) entre la trame WZ et les deux trames de référence. L'estimation locale est générée en appliquant la technique MCTI comme dans le codeur de DISCOVER. L'approche proposée vise à combiner les deux estimations GMC SI et MCTI SI en utilisant deux techniques. La première consiste à fusionner les deux SI en se basant sur les différences entre les blocs correspondants dans la SI globale et la SI locale. La deuxième technique utilise les machines à vecteurs de support (Support Vector Machine ou SVM) pour combiner l'estimation de mouvement globale et locale.

  Dans la suite, nous proposons des approches qui consistent à exploiter la PDWZF et les coefficients DC décodés, afin d'améliorer la fusion de GMC SI et MCTI SI. Les premières approches visent à améliorer la combinaison de GMC SI et MCTI SI après le décodage de la première sous-bande DCT (bande DC). D'autres approches améliorent MCTI SI ainsi que la fusion, après le décodage de chaque sous-bande DCT.

- **Chapitre 6 - Fusion basée sur l'estimation des objets:** Dans ce chapitre, nous proposons une nouvelle méthode de fusion des estimations globale et locale, basée sur l'estimation des objets dans la SI en utilisant les objets des trames de référence. Tout d'abord, nous considérons que les objets des trames de référence sont déjà segmentés au décodeur et nous nous intéressons à la combinaison du mouvement global et local. En premier lieu, les courbes élastiques sont utilisées afin d'estimer les contours des objets dans la SI en utilisant les courbes des objets de référence. Ensuite, les contours estimés sont utilisés afin de générer des masques. Les pixels à l'intérieur des masques

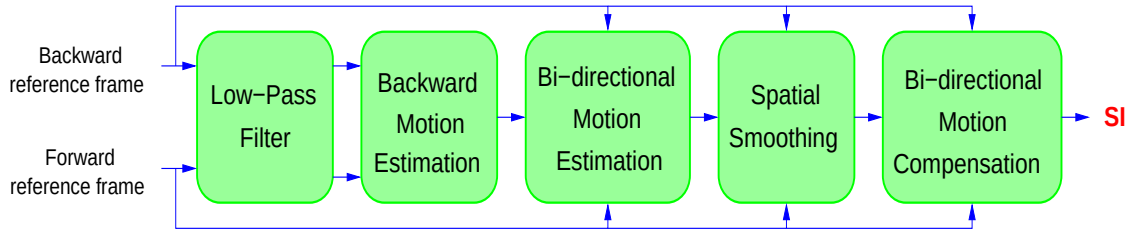

Figure 1: Modules de technique MCTI.

sont sélectionnés de MCTI SI et les pixels du fond de la SI sont sélectionnés de GMC SI.

De plus, des approches basées sur l'estimation locale sont appliquées aux objets des trames de référence, afin d'estimer les objets dans la SI. Entre autres, la technique MCTI est appliquée aux objets de référence pour générer les objets dans la SI. En se basant sur les objets estimés, deux approches sont utilisées pour la combinaison.

- **Annexe:** Au début de cette thèse, nous avons testé une approche basée sur les signaux à taux d'innovation fini, en vue d'utiliser cette approche dans le cadre de DVC mono-vue et ensuite multi-vue. L'approche proposée est appropriée pour les séquences vidéo où la relation entre les trames est une transformation affine. Dans le but d'estimer les performances de cette approche, nous avons créé une séquence vidéo à partir d'un seul objet de façon à ce que la relation entre les trames soit une transformation affine. Concernant les résultats obtenus, nous avons remarqué que cette approche apporte un gain important pour les séquences vidéo synthétiques contenant un seul objet. Par contre, les performances diminuent considérablement lorsque des séquences réelles sont utilisées.

I - Interpolation Temporelle Compensée en Mouvement (MCTI) [7]

La Figure 1 montre les modules de la technique MCTI, largement utilisée dans le cadre DVC afin de générer la SI. Cette technique est composée de 4 modules principaux.

- **Filtre passe-bas :** Un filtre passe-bas est appliqué aux trames de référence afin de réduire le bruit.
- **Estimation monodirectionnelle du champ :** Les vecteurs de mouvement sont estimés pour des blocs de taille 16×16, en utilisant la différence absolue moyenne pondérée (WMAD).
- **Estimation bidirectionnelle du champ :** Les vecteurs de mouvement monodirectionnels sont utilisés afin d'estimer, pour chaque bloc de la SI, un champ de vecteurs bidirectionnels. Pour cette raison, pour chaque bloc dans la SI, le vecteur de

mouvement monodirectionnel le plus proche de ce bloc est divisé par symétrie et utilisé comme vecteur de mouvement bidirectionnel. En outre, une étape de raffinement est appliquée au champ des vecteurs bidirectionnels. Ensuite, les blocs 16×16 sont divisés en blocs de taille 8×8 et l'étape de raffinement est appliquée.

- **Filtrage Médian :** Les vecteurs de mouvement bidirectionnels obtenus sont régularisés en utilisant un filtrage médian.
- **Compensation de mouvement bidirectionnelle :** Les vecteurs de mouvement bidirectionnels sont utilisés pour la compensation du mouvement afin de générer la SI.

II - Amélioration Progressive de l'Information Adjacente

Dans cette section, nous décrivons une nouvelle approche qui permet d'améliorer progressivement la SI après le décodage de chaque sous-bande DCT, dans le cadre d'un système de DVC dans le domaine transformée. Spécifiquement, la SI initiale est estimée en utilisant la technique MCTI et utilisée pour le décodage de la première sous-bande DCT. La PDWZF obtenue est exploitée à chaque itération, conjointement avec les trames de référence, afin d'améliorer les vecteurs de mouvement estimés. La technique proposée pour l'amélioration de la SI est composée de trois étapes: la détection des vecteurs de mouvement erronés, la correction de ces vecteurs de mouvement et la sélection du mode de compensation de mouvement. L'approche proposée est similaire à celle exposée en [8]. Toutefois, dans le schéma proposé, la SI est progressivement améliorée après le décodage de chaque sous-bande, contrairement à [8], où elle n'est révisée qu'après le décodage de toutes les sous-bandes.

Les trois étapes principales de la méthode, avec les améliorations proposées, sont plus amplement détaillées comme suit :

- Détection des vecteurs de mouvement erronés : Les vecteurs de mouvement ne sont pas toujours fiables en présence de mouvements complexes ou rapides, ou d'occlusions. Afin d'identifier ces vecteurs suspects, la différence absolue moyenne (MAD) est estimée pour un bloc donné entre la PDWZF et la SI actuelle, et comparée à un seuil T_1 :

$$\mathrm{MAD}(\mathrm{PDWZF}, \mathrm{SI}(\mathbf{MV})) < T_1 \tag{1}$$

où $\mathbf{MV}$=(MVx, MVy) est le vecteur de mouvement candidat. Si la condition requise par l'équation (1) est satisfaite, le vecteur de mouvement est considéré comme fiable et sera conservé. Dans ce qu'on appellera l'algorithme II, ces vecteur fiables sont seulement raffinés deux fois durant le décodage des sous-bandes DCT (après le

décodage de la première sous-bande DCT et après le décodage de toutes les sous-bandes DCT), dans une petite fenêtre de recherche de ± 2 pixels, à une précision de demi-pixel près. A défaut, il est considéré comme étant peu fiable et sera corrigé à l'étape suivante.

- Correction des vecteurs de mouvement erronés : Pour améliorer les vecteurs de mouvement suspects, ces derniers sont à nouveau calculés par une estimation de mouvement bidirectionnelle et un critère MAD. Plus précisément, pour le bloc considéré dans la PDWZF, le vecteur de mouvement qui minimise le MAD avec la trame de référence précédente est calculé. De manière similaire, un vecteur de mouvement entre le bloc considéré dans la PDWZF et la trame de référence suivante est estimé. Ces vecteurs de mouvement bidirectionnels sont estimés en utilisant l'un de deux algorithmes I et II:

  ▷ **Alg. I** : Dans cet algorithme, pour chaque bloc de 8×8 pixels dans la WZF, le vecteur de mouvement est estimé dans une fenêtre de recherche de ± 16 pixels, à une précision au pixel près.

  ▷ **Alg. II** : Dans cet algorithme, pour chaque bloc de 8×8 pixels dans la WZF, le vecteur de mouvement est estimé en utilisant un bloc étendu de $(8+n) \times (8+n)$ pixels, à une précision aux deux pixels près, dans une fenêtre de recherche de ± 16 pixels. Ensuite, le vecteur de mouvement obtenu est raffiné dans une petite fenêtre de recherche de ± 3 pixels, à une précision de demi-pixel près.

- Sélection du mode de compensation de mouvement : L'objectif de cette étape est de générer une compensation de mouvement optimale en sélectionnant le bloc le plus semblable au bloc courant parmi trois sources : la trame de référence précédente (mode BACKWARD), la trame de référence suivante (mode FORWARD) et la moyenne bidirectionnelle compensée en mouvement des trames de référence précédente et suivante (BIDIRECTIONAL). La sélection est effectuée comme suit:

$$\begin{cases} \text{if } |\text{MAD}_f - \text{MAD}_b| < T_2 \\ \quad \{\text{mode} = \text{BIDIRECTIONAL}\} \\ \text{else if } \text{MAD}_f < \text{MAD}_b \\ \quad \{\text{mode} = \text{FORWARD}\} \\ \text{else} \\ \quad \{\text{mode} = \text{BACKWARD}\} \end{cases}$$

  où MAD_b et MAD_f sont les différences absolues moyennes entre le bloc considéré dans la PDWZF et les blocs correspondants dans les trames de référence précédente et suivante, respectivement, et T_2 constitue un seuil.

Résultats des simulations

Afin d'évaluer les performances du schéma proposé, nous avons réalisé de nombreuses simulations, en utilisant des conditions identiques à celles de DISCOVER [4, 5]. Nous avons utilisé les séquences de test 'Stefan', 'Foreman', 'Bus', 'Coastguard', 'Soccer' et 'Hall' avec une résolution QCIF et un échantillonnage à 15 images/sec.

Le paramètre T_1 joue un rôle important dans notre système. En particulier, il détermine la complexité de notre méthode. Ainsi, pour $T_1 = 0$, on ré-estime tous les vecteurs de mouvement pour tous les blocs, même si les vecteurs sont fiables. Pour $T_1 = +\infty$, tous le vecteurs de mouvement sont considérés comme fiables et conservés, ce qui est équivalent à DISCOVER. Concernant le paramètre T_2, l'utilisation du mode "bidirectionnel - $T_2 > 0$" est toujours meilleur que le mode "unidirectionnel - $T_2 = 0$". En outre, la taille du bloc étendu $(8 + n) \times (8 + n)$ peut encore améliorer les performances, avec une augmentation de la charge de calcul. Nous avons déterminé que les performances de notre système sont optimales pour $T_1 = 4$, $T_2 = 5$ et $n = 4$.

Les valeurs du PSNR et du débit de VISNET II [9], Martins *et al.* [10], la méthode proposée Alg. I et Alg. II, par rapport à DISCOVER, sont montrées dans le Tableau 1, en utilisant la métrique de Bjontegaard [11], pour des longueurs de GOP = 2, 4 et 8. On peut observer que les performances de notre méthode (Alg.I et Alg. II) sont constamment supérieures à DISCOVER, VISNET II [9] et Martins *et al.* [10], pour toutes les tailles de GOP.

L'algorithme Alg. II peut apporter un gain comparé à l'algorithme Alg. I pour toutes les séquences testées et pour toutes les tailles de GOP. La méthode proposée Alg. II apporte une amélioration sur les performances RD jusqu'à 1.05 dB, avec une réduction de débit de 17.65 % par rapport à DISCOVER, pour GOP = 2. Les améliorations sont encore plus significatives pour GOP = 4 et GOP = 8: jusqu'à 2.19 dB avec une réduction de débit de 32.65 %, et 3.02 dB avec une réduction de débit de 41.88 % respectivement.

Table 1: Performances RD de VISNET II [9], Martins *et al.* [10], l'Alg. I et Alg. II pour les séquences *Stefan*, *Foreman*, *Bus*, *Coastguard*, *Soccer* et *Hall*, pour les tailles de GOP 2, 4 et 8, comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
VISNET II [9]						
Δ_R (%)	3.51	-2.41	6.10	1.63	-6.59	1.56
Δ_{PSNR} [dB]	-0.22	0.14	-0.34	-0.08	0.36	-0.11
Martins *et al.* [10]						
Δ_R (%)	-5.25	-6.54	-2.69	-0.98	-9.45	-0.45
Δ_{PSNR} [dB]	0.33	0.40	0.16	0.05	0.54	0.03
Alg. I						
Δ_R (%)	-6.81	-11.11	0.12	-2.00	-14.56	-0.38
Δ_{PSNR} [dB]	0.43	0.69	-0.01	0.10	0.85	0.03
Alg. II						
Δ_R (%)	**-14.06**	**-16.29**	**-4.50**	**-2.24**	**-17.65**	**-1.34**
Δ_{PSNR} [dB]	**0.93**	**1.05**	**0.27**	**0.11**	**1.05**	**0.10**
GOP = 4						
VISNET II [9]						
Δ_R (%)	-0.08	-9.36	2.57	-0.78	-10.01	0.88
Δ_{PSNR} [dB]	0.00	0.53	-0.14	0.03	0.58	-0.05
Martins *et al.* [10]						
Δ_R (%)	-13.38	-16.96	-7.37	-4.26	-14.63	-1.96
Δ_{PSNR} [dB]	0.85	1.04	0.45	0.18	0.90	0.12
Alg. I						
Δ_R (%)	-17.90	-24.33	-7.99	-7.33	-20.78	-2.17
Δ_{PSNR} [dB]	1.16	1.53	0.48	0.31	1.30	0.13
Alg. II						
Δ_R (%)	**-27.84**	**-32.65**	**-15.82**	**-11.94**	**-25.08**	**-4.24**
Δ_{PSNR} [dB]	**1.93**	**2.19**	**0.99**	**0.52**	**1.61**	**0.27**
GOP = 8						
VISNET II [9]						
Δ_R (%)	-1.76	-14.05	-0.68	-8.44	-11.37	-5.36
Δ_{PSNR} [dB]	0.11	0.82	0.05	0.36	0.68	0.33
Martins *et al.* [10]						
Δ_R (%)	-18.36	-23.96	-12.66	-9.67	-17.68	-6.99
Δ_{PSNR} [dB]	1.23	1.54	0.81	0.43	1.13	0.42
Alg. I						
Δ_R (%)	-23.02	-32.52	-14.08	-16.35	-23.12	-8.97
Δ_{PSNR} [dB]	1.56	2.17	0.90	0.73	1.50	0.54
Alg. II						
Δ_R (%)	**-34.13**	**-41.88**	**-22.83**	**-24.21**	**-28.16**	**-11.04**
Δ_{PSNR} [dB]	**2.51**	**3.02**	**1.53**	**1.14**	**1.88**	**0.68**

III - Techniques pour l'amélioration de l'information adjacente

L'estimation du mouvement dans deux sens

Dans le cadre du projet DISCOVER, la SI est estimée par la technique MCTI qui s'est avérée être l'une des plus efficaces parmi les méthodes existantes. En revanche, nous proposons une nouvelle approche qui vise à améliorer la qualité de la SI. Cette approche est basée sur deux estimations du champ de vecteurs. La figure 2 montre le schéma de l'approche proposée pour la génération de la SI. L'algorithme commence par estimer les vecteurs de mouvement dans les deux sens, pour une taille de bloc $B_0 \times B_0$. Concernant le premier sens, les vecteurs de mouvement sont estimés de la trame de référence précédente (BRF)

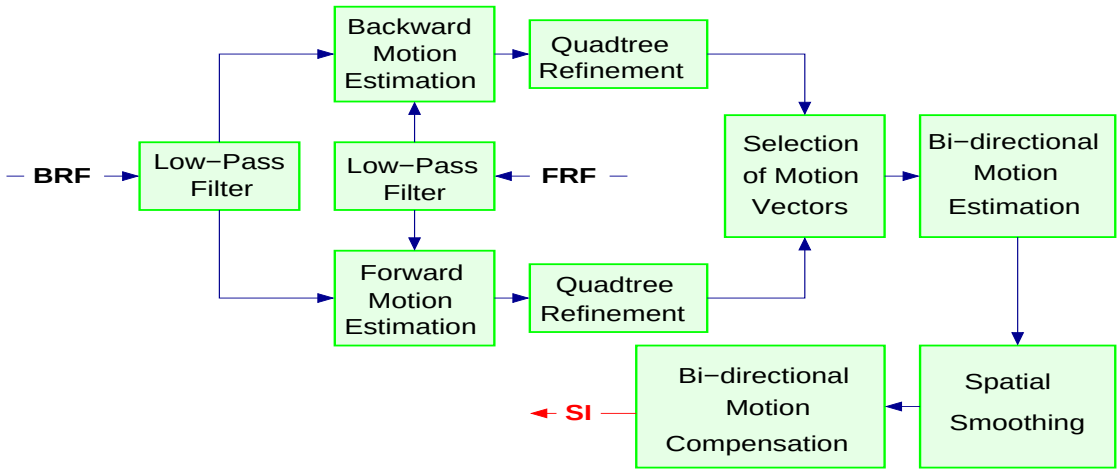

Figure 2: Approche proposée pour la génération de la SI.

Table 2: PSNR moyen de la SI obtenue en utilisant la méthode proposée (SIG) et la technique MCTI.

SI Average PSNR [dB]						
sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
MCTI	22.57	29.31	24.72	31.43	22.05	35.66
SIG	**23.83**	**29.97**	**27.14**	**32.35**	**22.75**	**36.22**
GOP = 4						
MCTI	21.28	27.58	23.48	29.85	20.81	34.51
SIG	**22.24**	**28.10**	**25.68**	**30.75**	**21.42**	**35.03**
GOP = 8						
MCTI	20.64	26.24	22.53	28.75	20.15	33.69
SIG	**21.47**	**26.69**	**24.61**	**29.59**	**20.70**	**34.04**

vers la trame de référence suivante (FRF). Pour l'autre sens, les vecteurs de mouvement sont estimés de la FRF vers la BRF. Ensuite, chaque bloc est divisé en quatre blocs de $B_1 \times B_1$ ($B_1 = \frac{B_0}{2}$) pixels et ces blocs héritent des vecteurs de mouvement estimés pour les blocs de $B_0 \times B_0$ pixels. Dans une étape suivante, une approche Quadtree est utilisée afin de sélectionner, pour un bloc donné b, un vecteur de mouvement parmi les vecteurs adjacents, en se basant sur le critère MAD. La même procédure est itérée de façon à arriver à une taille de bloc de $B_M \times B_M$ pixels. Ensuite, parmi les deux estimations (correspondant aux deux sens), le vecteur de mouvement qui minimise la MAD est retenu. Finalement, l'estimation bidirectionnelle du champ, le filtrage médian et la compensation de mouvement bidirectionnelle sont appliqués afin de générer la SI en utilisant un bloc étendu.

Le PSNR moyen de la SI est indiqué dans le tableau 2, pour la méthode proposée et la technique MCTI, pour toutes les séquences et les différentes tailles de GOP. Un gain significatif est observé avec la méthode proposée pour toutes les séquences de test et toutes les tailles de GOP. Le gain atteint $1,26$ dB et $2,42$ dB pour les séquences Stefan et Bus respectivement, pour GOP $= 2$.

Dans le tableau 3, nous montrons les performances RD de la méthode proposée par

Table 3: Performances RD pour les séquences *Stefan*, *Foreman*, *Bus*, *Coastguard*, *Soccer* et *Hall*, comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
Δ_R [%]	-8.32	-3.17	-13.50	-4.22	-4.95	-1.55
Δ_{PSNR} [dB]	0.50	0.17	0.79	0.21	0.27	0.12
GOP = 4						
Δ_R [%]	-9.28	-2.66	-21.16	-11.70	-6.28	-2.76
Δ_{PSNR} [dB]	0.54	0.14	1.24	0.47	0.34	0.19
GOP = 8						
Δ_R [%]	-9.06	-3.58	-22.10	-16.93	-5.67	-6.07
Δ_{PSNR} [dB]	0.54	0.16	1.31	0.71	0.33	0.32

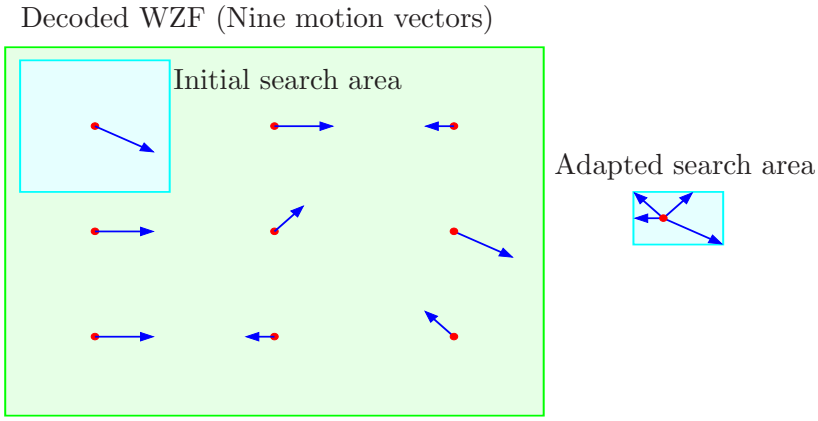

Figure 3: Les vecteurs de mouvement obtenus sont utilisés afin d'adapter la fenêtre de recherche.

rapport à DISCOVER, en utilisant la métrique de Bjontegaard [11]. La méthode proposée permet d'améliorer les performances pour toutes les séquences. Pour la séquence Bus, nous arrivons à une amélioration du PSNR de $1,31$ dB par rapport à MCTI, pour GOP $= 8$.

Recherche de mouvement adaptative

Nous proposons une nouvelle approche qui consiste à adapter la fenêtre de recherche selon le mouvement entre la WZF et les trames de référence, afin d'améliorer successivement la SI. Au début, la taille de la fenêtre de recherche est initialisée en fonction de la distance entre les trames de référence. Après le décodage de la première sous-bande DCT, une PDWZF est construite. N blocs sont sélectionnés dans la PDWZF en utilisant un échantillonnage uniforme. Ensuite, les vecteurs de mouvement sont estimés pour ces blocs entre la PDWZF et la BRF (et FRF). Ces vecteurs estimés sont utilisés afin d'adapter la fenêtre de recherche dans les quatre directions.

La figure 3 montre la méthode proposée concernant l'adaptation de la fenêtre de recherche. Plus précisément, les maximums des vecteurs de mouvement obtenus dans les quatre directions sont utilisés pour adapter la fenêtre de recherche initiale. Ainsi, la fenêtre de recherche est adaptée d'une manière conforme au mouvement courant entre la WZF et les trames de référence. Finalement, la SI est améliorée après le décodage de chaque sous-

Table 4: Performances RD pour les séquences *Stefan*, *Foreman*, *Bus*, *Coastguard*, *Soccer* et *Hall* comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
Alg. II [12]						
Δ_R (%)	-14.06	-16.29	-4.50	-2.24	-17.65	-1.34
Δ_{PSNR} [dB]	0.93	1.05	0.27	0.11	1.05	0.10
Proposed						
Δ_R (%)	**-17.31**	**-16.53**	**-4.91**	**-2.28**	**-18.96**	**-1.36**
Δ_{PSNR} [dB]	**1.16**	**1.07**	**0.30**	**0.11**	**1.14**	**0.10**
GOP = 4						
Alg. II [12]						
Δ_R (%)	-27.84	-32.65	-15.82	-11.94	-25.08	-4.24
Δ_{PSNR} [dB]	1.93	2.19	0.99	0.52	1.61	0.27
Proposed						
Δ_R (%)	**-34.44**	**-33.88**	**-16.42**	**-12.15**	**-27.31**	**-4.27**
Δ_{PSNR} [dB]	**2.51**	**2.30**	**1.03**	**0.53**	**1.78**	**0.27**
GOP = 8						
Alg. II [12]						
Δ_R (%)	-34.13	-41.88	-22.83	-24.21	-28.16	-11.04
Δ_{PSNR} [dB]	2.51	3.02	1.53	1.14	1.88	0.68
Proposed						
Δ_R (%)	**-41.98**	**-43.55**	**-26.13**	**-24.40**	**-31.43**	**-11.22**
Δ_{PSNR} [dB]	**3.29**	**3.19**	**1.78**	**1.15**	**2.15**	**0.68**

bande DCT en utilisant l'Alg. II, mais la fenêtre de recherche adaptée est utilisée dans cette approche.

Les performances RD de la méthode proposée sont montrées pour les séquences Stefan, Foreman, Bus, Coastguard, de soccer et Hall dans le tableau 4, en comparaison avec DISCOVER, en utilisant la métrique de Bjontegaard [11]. La première ligne représente les résultats de la technique précédente (Alg. II), *i.e.*, une fenêtre de recherche constante de ± 16 pixels est utilisée quelque soit la distance entre les trames de référence. Il est clair que notre méthode proposée permet d'atteindre un gain significatif par rapport à DISCOVER, en particulier pour les séquences contenant un mouvement rapide, comme les séquences Stefan et Foreman.

Pour la séquence Stefan avec une GOP = 8, Alg. II peut atteindre un gain de $2,51$ dB par rapport à DISCOVER. La méthode proposée permet un gain significatif de $3,29$ dB par rapport à DISCOVER. En outre, le temps de décodage est considérablement réduit, même pour des séquences contenant un mouvement lent, grâce à l'adaptation de la fenêtre de recherche au mouvement courant.

Ré-estimation de l'information adjacente pour un long GOP

Notons par $\mathbf{I}_k$ la WZF qu'il faut l'estimer et $\hat{\mathbf{I}}_k$ la WZF décodée. Pour GOP = 2, les trames de référence précédemment décodées $\hat{\mathbf{I}}_{k-1}$ et $\hat{\mathbf{I}}_{k+1}$ sont simplement utilisées pour estimer la SI. Ensuite, la SI estimée est utilisée pour décoder la WZF et obtenir la WZF décodée $\hat{\mathbf{I}}_k$. Pour GOP = 4, la figure 4 montre les étapes d'interpolation des WZFs.

Il faut noter que la qualité de la SI pour $\mathbf{I}_k$ est pire que celles de $\mathbf{I}_{k-1}$ et $\mathbf{I}_{k+1}$, à cause

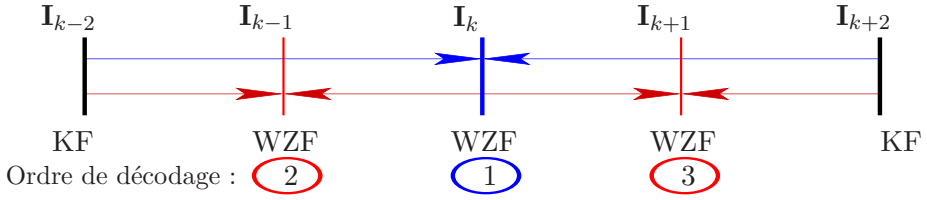

Figure 4: Étapes d'interpolation pour une GOP = 4.

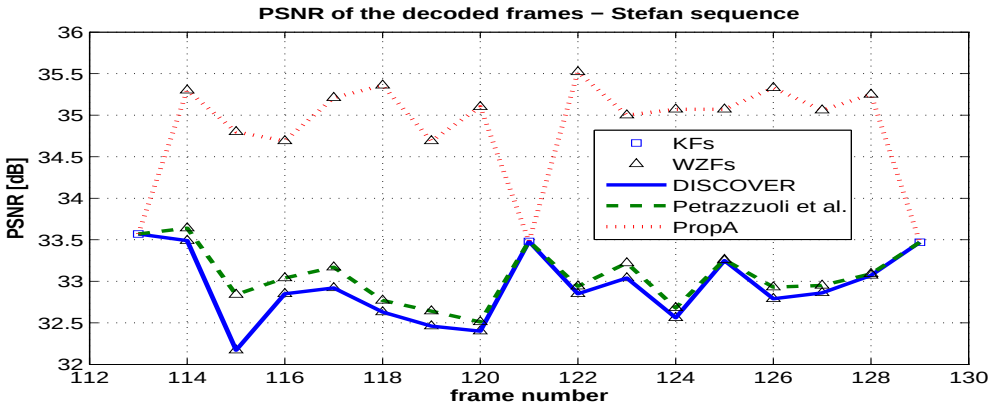

Figure 5: PSNR des trames décodées pour deux GOPs à partir de la trame 113, pour la séquence Stefan, avec une taille de GOP = 8.

de la distance entre les trames de référence. Pour cette raison, nous proposons une nouvelle approche qui vise à améliorer la qualité de la SI après le décodage de toutes les WZFs à l'intérieur du GOP. Dans la littérature, une approche [13] est proposée qui consiste à appliquer la technique MCTI sur les WZFs décodées $\hat{\mathbf{I}}_{k-1}$ et $\hat{\mathbf{I}}_{k+1}$ afin de ré-estimer la SI pour la trame $\mathbf{I}_k$, sans utiliser la trame décodée $\hat{\mathbf{I}}_k$.

Dans notre approche, nous proposons d'exploiter la trame décodée $\hat{\mathbf{I}}_k$, avec les trames $\hat{\mathbf{I}}_{k-1}$ et $\hat{\mathbf{I}}_{k+1}$, pour estimer la SI de la trame $\mathbf{I}_{k-1}$. La même procédure est également appliquée aux trames $\mathbf{I}_{k-1}$ et $\mathbf{I}_{k+1}$. Nous utilisons une fenêtre de recherche adaptative et une taille de bloc variable lors de l'estimation de la SI.

Dans le cas où la méthode proposée est appliquée sur les trames décodées par DISCOVER, l'approche est désignée par 'PropA'. Nous l'avons aussi appliquée sur les trames décodées obtenues par Alg. II. Dans ce cas, l'approche est nommée 'PropB'.

La figure 5 montre le PSNR des trames décodées de la séquence Stefan pour DISCOVER, la méthode proposée dans [13] et le PropA. La technique proposée PropA permet d'obtenir un gain significatif par rapport à DISCOVER et [13].

Les performances RD sont indiquées pour les différentes séquences dans le tableau 5, en comparaison avec DISCOVER, en utilisant la métrique de Bjontegaard, pour les GOPs = 2 et 4. Nous représentons les performances de la méthode proposée dans [13], la méthode proposée PropA, l'Alg. II et la méthode proposée PropB.

On remarque que l'approche proposée propA apporte toujours un gain comparé à

Table 5: Performances RD pour les GOPs = 2 et 4 comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 4						
Petrazzuoli et al. [13]						
Δ_R (%)	-4.49	-5.77	-5.65	-4.29	-3.20	-0.84
Δ_{PSNR} [dB]	0.23	0.28	0.30	0.18	0.15	0.04
PropA						
Δ_R (%)	**-17.42**	**-21.31**	**-8.79**	**-7.25**	**-18.45**	**-3.96**
Δ_{PSNR} [dB]	**1.00**	**1.07**	**0.43**	**0.30**	**0.88**	**0.23**
Alg. II						
Δ_R (%)	-30.35	-36.96	-17.90	-12.25	-28.12	-4.92
Δ_{PSNR} [dB]	1.91	2.19	1.00	0.52	1.59	0.29
PropB						
Δ_R (%)	**-35.53**	**-39.88**	**-19.70**	**-14.78**	**-32.69**	**-6.17**
Δ_{PSNR} [dB]	**2.30**	**2.39**	**1.09**	**0.63**	**1.85**	**0.37**
GOP = 8						
Petrazzuoli et al. [13]						
Δ_R (%)	-8.30	-12.02	-11.56	-9.38	-6.53	-2.33
Δ_{PSNR} [dB]	0.45	0.56	0.60	0.38	0.28	0.09
PropA						
Δ_R (%)	**-22.16**	**-28.01**	**-15.26**	**-13.92**	**-23.49**	**-6.91**
Δ_{PSNR} [dB]	**1.27**	**1.40**	**0.74**	**0.56**	**1.10**	**0.32**
Alg. II						
Δ_R (%)	-37.80	-48.18	-26.65	-26.90	-32.82	-13.00
Δ_{PSNR} [dB]	2.46	2.98	1.53	1.16	1.86	0.71
PropB						
Δ_R (%)	**-44.98**	**-53.03**	**-31.44**	**-31.99**	**-39.51**	**-16.10**
Δ_{PSNR} [dB]	**3.07**	**3.38**	**1.83**	**1.40**	**2.26**	**0.87**

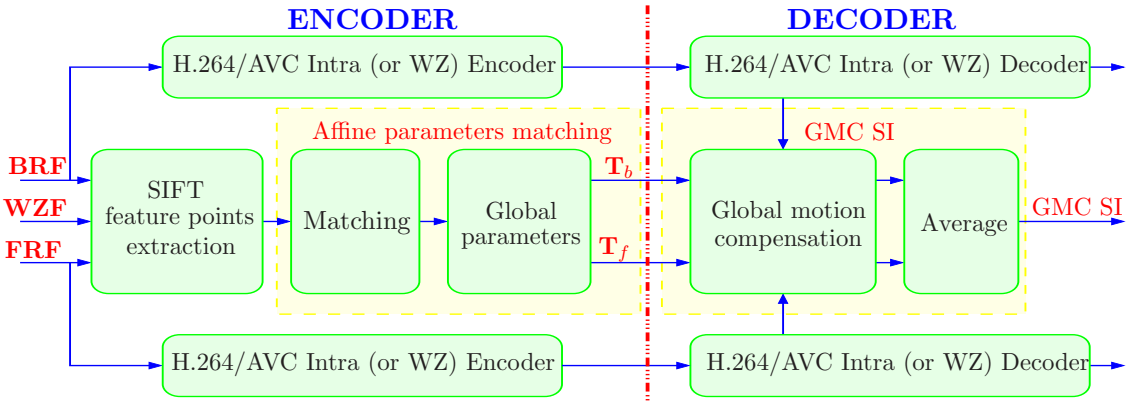

Figure 6: Schéma de la technique proposée GMC.

DISCOVER et [13], et l'approche PropB apporte un gain comparé à Alg. II, en particulier pour les séquences contenant un mouvement rapide.

IV - Fusion de l'estimation du mouvement global et local

Cette approche consiste à combiner l'estimation du mouvement global et local dans le cadre de DVC. D'abord, l'estimation du mouvement local est tout simplement générée en utilisant la technique MCTI comme dans le codeur de DISCOVER. Cette SI est appelée MCTI SI.

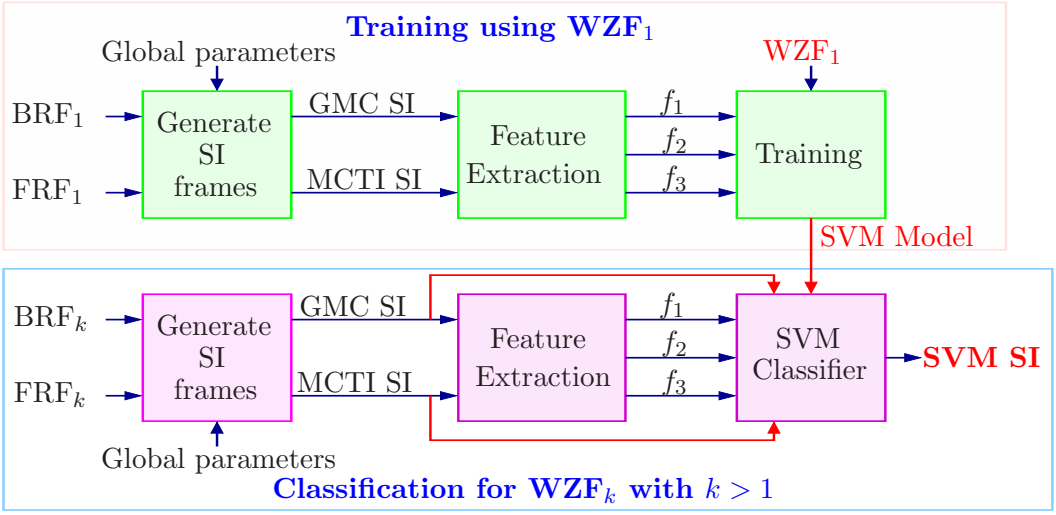

Figure 7: Schéma de la méthode proposée basée sur SVM pour générer des SVM SI.

En revanche, l'estimation du mouvement global est réalisée suivant un schéma proposé et décrit dans la figure 6: Les points caractéristiques de la WZF et des trames de référence sont extraits en utilisant l'algorithme SIFT. Un algorithme efficace est en plus appliqué afin de supprimer les points qui sont localisés dans les objets en mouvement. Ensuite, ces points caractéristiques sont utilisés afin d'estimer les paramètres globaux entre la WZF et les trames de référence. Soient T_b et T_f les transformations globales entre la WZF et les trames de référence précédente (BRF) et suivante (FRF), respectivement. Les paramètres estimés sont encodés et envoyés au décodeur où T_b et T_f sont respectivement appliquées à la BRF et FRF décodée. Les trames transformées sont moyennées pour obtenir la nouvelle SI, appelée GMC SI.

Dans l'étape suivante, nous proposons une nouvelle approche qui vise à améliorer la SI finale en combinant GMC SI et MCTI SI. Cette approche est basée sur les différences $\mathrm{SAD_{GMC}}$ et $\mathrm{SAD_{MCTI}}$ entre les blocs correspondants des trames de référence compensées en mouvement, respectivement dans MCTI SI et GMC SI. La fusion binaire de GMC SI et MCTI SI, appelée 'SADbin', est définie par:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI(b)} & \text{if } \mathrm{SAD_{GMC}} < \mathrm{SAD_{MCTI}} \\ \text{MCTI SI(b)} & \text{otherwise} \end{cases} \tag{2}$$

Par ailleurs, une fusion linéaire de GMC SI et MCTI SI, nommée 'SADlin', est définie comme suit:

$$\mathrm{SI}(b) = \frac{\mathrm{SAD_{MCTI}} \cdot (\text{GMC SI}) + \mathrm{SAD_{GMC}} \cdot (\text{MCTI SI})}{(\mathrm{SAD_{GMC}} + \mathrm{SAD_{MCTI}})} \tag{3}$$

Dans ce qui suit, nous proposons une nouvelle approche en vue d'améliorer la combinaison de l'estimation du mouvement global et local au décodeur. Dans ce but, les machines à vecteurs de support sont utilisées pour fusionner les MCTI SI et GMC SI.

Chaque SI est divisée en blocs de taille 4×4 pixels. Ces derniers sont considérés comme appartenant à l'une de deux classes. Les variables de discrimination suivantes sont définies: $f_1 = \mathrm{SAD}_{\mathrm{GMC}}$, $f_2 = \mathrm{SAD}_{\mathrm{MCTI}}$ et $f_3 = \mathrm{SAD}_{\mathrm{GMC}} - \mathrm{SAD}_{\mathrm{MCTI}}$. Pour l'étape d'apprentissage, la première WZF est encodée et décodée en utilisant H.264/AVC en mode Intra, comme pour les trames clés. Cette étape consiste à générer un modèle pour la classification des deux SI, de façon à permettre la prédiction d'une classe pour chaque bloc. La figure 7 montre le schéma de cette approche. La classe prédite est utilisée pour sélectionner le bloc de MCTI SI ou GMC SI (fusion binaire appelée 'SVMbin') pour la SI finale comme suit:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } d > 0 \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{4}$$

Par ailleurs, une combinaison linéaire des deux SI par SVM (nommée 'SVMlin') peut être réalisée comme suit:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } d > T \\ \text{MCTI SI} & \text{if } d < (-T) \\ \frac{(T+d)\cdot\text{GMC SI}+(T-d)\cdot\text{MCTI SI}}{2\cdot T} & \text{if } |d| \leq T \end{cases} \tag{5}$$

Les valeurs du PSNR et du débit des approches GMC, SADbin, SADlin, SVMbin, SVMlin et Oracle par rapport à DISCOVER sont montrées dans le Tableau 6, en utilisant la métrique de Bjontegaard [11] pour des longueurs de GOP = 2, 4 et 8. L'approche 'Oracle' vise à estimer la limite supérieure qui peut être atteinte en combinant GMC SI et SI MCTI, en utilisant la WZF originale (cette approche n'est pas applicable en pratique). On peut observer que les performances des approches proposées pour la fusion sont constamment supérieures à celles de DISCOVER, pour toutes les tailles de GOP.

L'approche proposée SVMlin peut apporter un gain par rapport aux autres approches, pour toutes les séquences sauf Stefan. D'ailleurs, l'approche GMC présente la meilleure performance pour la séquence Stefan.

Dans ce qui suit, nous proposons quelques approches qui permettent d'améliorer la fusion de GMC SI et MCTI SI durant le décodage de la WZF. D'abord, la PDWZF est utilisée après le décodage de la première sous-bande DCT pour la combinaison des deux SI. Cette approche, appelée 'FsPF', est simplement définie comme suit:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } \mathrm{SGMC}_1 < \mathrm{SMCTI}_1 \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{6}$$

où SGMC_1 et SMCTI_1 représentent respectivement les différences entre la PDWZF et GMC SI et MCTI SI après le décodage de la première sous-bande DCT.

Alternativement, les coefficients DC décodés peuvent être utilisés avec la PDWZF pour améliorer la fusion de GMC SI et MCTI SI après le décodage de la première sous-bande DCT. La figure 8 montre le schéma de cette approche, appelée 'DCFsPF'. Dans ce cas, la

Table 6: Performances RD de GMC, SADbin, SADlin, SVMbin, SVMlin et Oracle pour les séquences *Stefan*, *Foreman*, *Bus* et *Coastguard*, pour les tailles de GOP 2, 4 et 8, comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

Method	**GMC**	**SADbin**	**SADlin**	**SVMbin**	**SVMlin**	**Oracle**
GOP = 2						
Stefan						
Δ_R (%)	-23.67	-22.39	-19.69	**-23.63**	-23.26	-25.46
Δ_{PSNR} [dB]	1.64	1.54	1.33	**1.65**	1.62	1.80
Foreman						
Δ_R (%)	-8.51	-7.51	-8.71	-10.90	**-11.47**	-13.57
Δ_{PSNR} [dB]	0.52	0.46	0.53	0.68	**0.72**	0.86
Bus						
Δ_R (%)	6.14	-12.10	-9.28	-12.17	**-12.75**	-15.71
Δ_{PSNR} [dB]	-0.34	0.76	0.57	0.76	**0.80**	1.00
Coastguard						
Δ_R (%)	10.02	-4.40	-3.01	-4.90	**-5.24**	-7.43
Δ_{PSNR} [dB]	-0.47	0.22	0.15	0.25	**0.26**	0.38
GOP = 4						
Stefan						
Δ_R (%)	**-42.38**	-39.59	-34.48	-41.59	-40.80	-44.52
Δ_{PSNR} [dB]	**3.21**	2.94	2.45	3.16	3.07	3.44
Foreman						
Δ_R (%)	-21.89	-15.14	-17.62	-22.59	**-23.44**	-28.50
Δ_{PSNR} [dB]	1.35	0.90	1.06	1.41	**1.47**	1.84
Bus						
Δ_R (%)	-1.09	-23.60	-20.05	-23.83	**-24.59**	-29.14
Δ_{PSNR} [dB]	0.07	1.55	1.29	1.58	**1.63**	1.97
Coastguard						
Δ_R (%)	8.53	-13.26	-11.18	-14.89	**-15.49**	-20.08
Δ_{PSNR} [dB]	-0.35	0.58	0.48	0.66	**0.69**	0.91
GOP = 8						
Stefan						
Δ_R (%)	**-49.34**	-46.05	-40.30	-48.27	-47.60	-51.89
Δ_{PSNR} [dB]	**3.96**	3.61	3.00	3.89	3.80	4.26
Foreman						
Δ_R (%)	-30.51	-20.88	-23.41	-30.36	**-31.25**	-36.90
Δ_{PSNR} [dB]	1.99	1.28	1.45	1.98	**2.05**	2.51
Bus						
Δ_R (%)	-8.60	-28.23	-25.10	-28.64	**-29.67**	-34.63
Δ_{PSNR} [dB]	0.54	1.97	1.71	2.01	**2.09**	2.49
Coastguard						
Δ_R (%)	-2.37	-22.47	-20.19	-25.02	**-25.69**	-31.72
Δ_{PSNR} [dB]	0.10	1.04	0.92	1.18	**1.21**	1.55

fusion est améliorée comme suit (L_{GMC} et L_{MCTI} sont définies dans Figure 8):

si $L_{\text{GMC}} < L_{\text{MCTI}}$ et $\text{SGMC}_1 < \text{SMCTI}_1$
- La fusion de ce bloc est choisi de GMC SI

sinon

si $L_{\text{MCTI}} < L_{\text{GMC}}$ et $\text{SMCTI}_1 < \text{SGMC}_1$
- La fusion de ce bloc est choisi de MCTI SI

sinon
- La fusion de ce bloc est la moyenne de GMC SI et SI MCTI

Dans ce qui suit, les approches proposées visent à améliorer la SI après le décodage de chaque sous-bande DCT. La première approche permet de fusionner GMC SI et MCTI SI

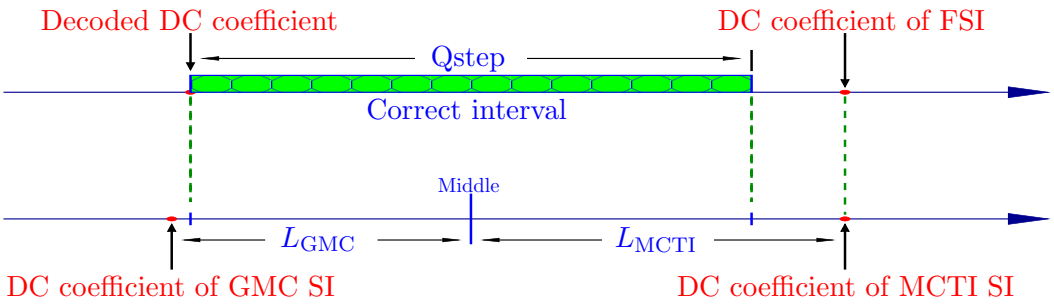

Figure 8: Amélioration de la fusion en utilisant les coefficients DC décodés.

en se basant sur la PDWZF comme suit:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } \mathrm{S}_{\mathrm{GMC}} < \mathrm{S}_{\mathrm{MCTI}} \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{7}$$

où $\mathrm{S}_{\mathrm{GMC}}$ et $\mathrm{S}_{\mathrm{MCTI}}$ représentent respectivement les différences entre la PDWZF et GMC SI et MCTI SI, après le décodage de chaque sous-bande DCT. Cette approche est appelée 'FsPFAll'.

La seconde approche consiste tout d'abord à améliorer MCTI SI en utilisant Alg. II, ensuite la fusion de GMC SI et MCTI SI améliorée est faite en exploitant la PDWZF comme suit:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } \mathrm{S}_{\mathrm{GMC}_k} < \mathrm{S}_{\mathrm{MCTI}_k} \\ \text{MCTI SI}_k & \text{otherwise} \end{cases} \tag{8}$$

où $\mathrm{S}_{\mathrm{GMC}_k}$ et $\mathrm{S}_{\mathrm{MCTI}_k}$ représentent respectivement les différences entre la PDWZF et GMC SI et MCTI SI améliorée (en utilisant Alg. II), après le décodage de chaque sous-bande DCT. Cette approche est nommée 'FsIter'.

Les valeurs du PSNR et du débit des approches Alg. II, SADbin, FsPF, DCFsPF, FsPFAll et FsIter, par rapport à DISCOVER, sont montrées dans le Tableau 7, en utilisant la métrique de Bjontegaard [11], pour des longueurs de GOP = 2, 4 et 8. On peut observer que l'amélioration de la fusion apporte toujours un gain comparé à la première fusion SADbin de GMC SI et MCTI SI. Il est clair que l'utilisation des coefficients DC décodés peut apporter un gain comparé à FsPF (où seule la PDWZF est utilisée).

La méthode proposée FsIter peut permettre d'obtenir un gain significatif par rapport à DISCOVER et Alg. II, pour toutes les séquences et toutes les tailles de GOP. Le gain atteint 4, 59 dB, lorsque Alg. II permet d'obtenir un gain de 2, 51 dB, pour la séquence Stefan, avec un GOP = 8. Par ailleurs, FsIter permet une amélioration importante par rapport à la première fusion SADbin, en particulier pour les séquences Foreman et Stefan.

Les valeurs du PSNR et du débit des approches Alg. II, SADbin, DCFsPF, FsIter, H.264/AVC Intra, H.264/AVC No motion et H.264/AVC Inter par rapport à DISCOVER sont montrées dans le Tableau 8, en utilisant la métrique de Bjontegaard [11], pour des longueurs de GOP = 2, 4 et 8. On peut noter que les performances de FsIter s'approchent de

Table 7: Performances RD de Alg. II, SADbin, FsPF, DCFsPF, FsPFAll et FsIter pour les séquences *Stefan*, *Foreman*, *Bus* et *Coastguard*, pour les tailles de GOP 2, 4 et 8, comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

Method	**Alg. II**	**SADbin**	**FsPF**	**DCFsPF**	**FsPFAll**	**FsIter**
GOP = 2						
Stefan						
Δ_R (%)	-14.06	-22.39	-22.92	**-23.45**	-23.87	**-26.51**
Δ_{PSNR} [dB]	0.93	1.54	1.59	**1.63**	1.67	**1.89**
Foreman						
Δ_R (%)	-16.29	-7.51	-8.99	**-12.24**	-11.39	**-19.68**
Δ_{PSNR} [dB]	1.05	0.46	0.55	**0.77**	0.71	**1.30**
Bus						
Δ_R (%)	-4.50	-12.10	-13.19	**-14.23**	-14.46	**-15.32**
Δ_{PSNR} [dB]	0.27	0.76	0.83	**0.90**	0.92	**0.98**
Coastguard						
Δ_R (%)	-2.24	-4.40	-4.69	**-6.07**	-6.03	**-7.68**
Δ_{PSNR} [dB]	0.11	0.22	0.24	**0.31**	0.31	**0.40**
GOP = 4						
Stefan						
Δ_R (%)	-27.84	-39.59	-40.55	**-41.68**	-42.25	**-46.11**
Δ_{PSNR} [dB]	1.93	2.94	3.03	**3.14**	3.22	**3.65**
Foreman						
Δ_R (%)	-32.65	-15.14	-19.28	**-25.28**	-23.99	**-38.12**
Δ_{PSNR} [dB]	2.19	0.90	1.17	**1.60**	1.52	**2.69**
Bus						
Δ_R (%)	-15.82	-23.60	-25.83	**-27.73**	-27.74	**-30.89**
Δ_{PSNR} [dB]	0.99	1.55	1.72	**1.87**	1.88	**2.13**
Coastguard						
Δ_R (%)	-11.94	-13.26	-14.30	**-17.59**	-16.57	**-21.60**
Δ_{PSNR} [dB]	0.52	0.58	0.63	**0.79**	0.74	**1.00**
GOP = 8						
Stefan						
Δ_R (%)	-34.13	-46.05	-47.44	**-48.99**	-49.58	**-53.98**
Δ_{PSNR} [dB]	2.51	3.61	3.76	**3.93**	4.03	**4.59**
Foreman						
Δ_R (%)	-41.88	-20.88	-26.20	**-33.21**	-31.52	**-47.86**
Δ_{PSNR} [dB]	3.02	1.28	1.66	**2.21**	2.10	**3.65**
Bus						
Δ_R (%)	-22.83	-28.23	-31.73	**-34.31**	-34.18	**-40.40**
Δ_{PSNR} [dB]	1.53	1.97	2.25	**2.47**	2.48	**3.04**
Coastguard						
Δ_R (%)	-24.21	-22.47	-23.93	**-28.53**	-26.98	**-34.51**
Δ_{PSNR} [dB]	1.14	1.04	1.12	**1.37**	1.29	**1.74**

celles de H.264/AVC Inter, et qu'elles surpassent celles de H.264/AVC Intra et H.264/AVC No motion, pour toutes les tailles de GOP et toutes les séquences testées.

Table 8: Performances RD de Alg. II, SADbin, DCFsPF, FsIter, H.264/AVC Intra, H.264/AVC No motion, H.264/AVC Inter pour les séquences *Stefan*, *Foreman*, *Bus* et *Coastguard*, pour les tailles de GOP 2, 4 et 8, comparées au codeur de DISCOVER, en utilisant la métrique de Bjontegaard.

Method	**Alg. II**	**SADbin**	**DCFsPF**	**FsIter**	**H.264/AVC Intra**	**H.264/AVC No motion**	**H.264/AVC Inter**
GOP = 2							
Stefan							
Δ_R (%)	-14.06	-22.39	-23.45	**-26.51**	-6.01	-12.10	-38.97
Δ_{PSNR} [dB]	0.93	1.54	1.63	**1.89**	0.42	0.72	3.18
Foreman							
Δ_R (%)	-16.29	-7.51	-12.24	**-19.68**	6.17	-16.77	-35.90
Δ_{PSNR} [dB]	1.05	0.46	0.77	**1.30**	-0.41	1.13	2.73
Bus							
Δ_R (%)	-4.50	-12.10	-14.23	**-15.32**	2.33	0.02	-31.23
Δ_{PSNR} [dB]	0.27	0.76	0.90	**0.98**	-0.13	-0.02	2.20
Coastguard							
Δ_R (%)	-2.24	-4.40	-6.07	**-7.68**	30.18	9.92	-17.15
Δ_{PSNR} [dB]	0.11	0.22	0.31	**0.40**	-1.44	-0.49	1.04
GOP = 4							
Stefan							
Δ_R (%)	-27.84	-39.59	-41.68	**-46.11**	-25.40	-27.20	-58.30
Δ_{PSNR} [dB]	1.93	2.94	3.14	**3.65**	1.78	1.77	5.22
Foreman							
Δ_R (%)	-32.65	-15.14	-25.28	**-38.12**	-12.28	-30.39	-58.93
Δ_{PSNR} [dB]	2.19	0.90	1.60	**2.69**	0.68	2.08	5.07
Bus							
Δ_R (%)	-15.82	-23.60	-27.73	**-30.89**	-12.18	-10.33	-49.87
Δ_{PSNR} [dB]	0.99	1.55	1.87	**2.13**	0.75	0.57	3.87
Coastguard							
Δ_R (%)	-11.94	-13.26	-17.59	**-21.60**	26.01	14.42	-35.73
Δ_{PSNR} [dB]	0.52	0.58	0.79	**1.00**	-1.04	-0.64	2.06
GOP = 8							
Stefan							
Δ_R (%)	-34.13	-46.05	-48.99	**-53.98**	-36.34	-36.13	-61.58
Δ_{PSNR} [dB]	2.51	3.61	3.93	**4.59**	2.78	2.56	5.64
Foreman							
Δ_R (%)	-41.88	-20.88	-33.21	**-47.86**	-28.48	-37.93	-67.93
Δ_{PSNR} [dB]	3.02	1.28	2.21	**3.65**	1.93	2.66	6.40
Bus							
Δ_R (%)	-22.83	-28.23	-34.31	**-40.40**	-27.19	-23.96	-49.98
Δ_{PSNR} [dB]	1.53	1.97	2.47	**3.04**	1.86	1.50	3.94
Coastguard							
Δ_R (%)	-24.21	-22.47	-28.53	**-34.51**	0.86	2.00	-50.31
Δ_{PSNR} [dB]	1.14	1.04	1.37	**1.74**	0.04	-0.11	3.01

V - Fusion basée sur l'estimation des objets

Dans cette section, nous proposons une nouvelle méthode qui consiste à combiner l'estimation globale et locale en utilisant l'estimation des objets extraits. On désigne par R_F et R_B les trames de référence précédente et suivante, respectivement. Les objets F_B^i et F_F^i ($i = 1, 2, ..., N_o$, N_o est le nombre des objets) sont supposés être déjà segmentés dans les trames de référence précédente et suivante respectivement. En outre, les masques des objets M_B^i et M_F^i sont générés à partir des objets extraits comme suit:

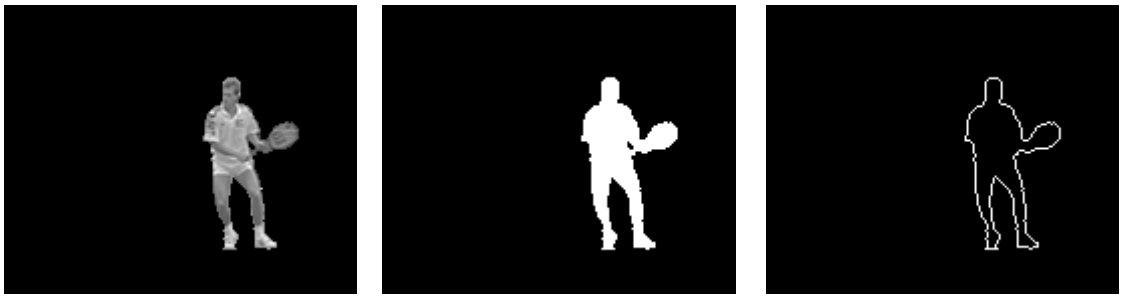

Figure 9: L'objet (F), le masque (M), et le contour (β) de la trame numéro 1 de la séquence Stefan.

Figure 10: GMC SI avant et après l'élimination des défauts (la moyenne entre les pixels des objets et du fond est évitée).

$$\begin{cases} M_B^i(x,y) = \begin{cases} 0 & \text{if } F_B^i(x,y) = 0 \\ 1 & \text{otherwise} \end{cases} \\ M_F^i(x,y) = \begin{cases} 0 & \text{if } F_F^i(x,y) = 0 \\ 1 & \text{otherwise} \end{cases} \end{cases} \tag{9}$$

Ensuite, les contours sont extraits en utilisant les masques. On considère β_B^i et β_F^i les représentations des contours des trames précédente et suivante respectivement. La figure 9 montre l'objet (F), le masque (M) généré à partir de l'objet et le contour (β) de l'objet pour la trame numéro 1 de la séquence Stefan.

En premier lieu, les objets sont utilisés afin de supprimer les défauts aux alentours des objets dans la GMC SI. La figure 10 montre la GMC SI avant et après l'élimination de cet effet. En fait, les objets sont utilisés afin d'éviter la moyenne entre les pixels des objets et du fond. Dans ce cas, seuls les pixels de fond sont utilisés pour la GMC SI.

Les contours sont considérés comme étant des courbes, et les courbes élastiques [14] sont appliquées aux courbes des objets dans les trames de référence précédente et suivante, $\beta_b^i(t)$ et $\beta_f^i(t)$, afin de prédire les courbes $\beta_e^i(t)$ de la SI. La figure 6.8 montre la courbe précédente $\beta_b^i(t)$ de la trame numéro 1 de la séquence Stefan, la courbe suivante $\beta_f^i(t)$ de la trame numéro 5 et les trois courbes prédites $\beta_e^i(t)$ en appliquant les courbes élastiques.

Les courbes prédites sont utilisées pour générer le masque M_e des objets de la SI.

Figure 11: La courbe précédente $\beta_b^i(t)$ (gauche, trame numéro 1 de la séquence Stefan), la courbe suivante $\beta_f^i(t)$ (droite, trame numéro 5) et les trois courbes estimées $\beta_e^i(t)$ (les courbes intermédiaires).

Ensuite, le masque M_e est utilisé afin de combiner MCTI SI et GMC SI comme suit (méthode appelée 'FusElastic'):

$$\text{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_e(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{10}$$

De plus, nous proposons d'appliquer la technique MCTI sur les objets de référence afin d'estimer les objets dans la SI. Ensuite, le masque M_{MCTI} est généré à partir des objets estimés. En plus, l'union de tous les objets forme une trame F_{MCTI}. A partir du masque M_{MCTI} et de la trame FMCTI, les SI globale et locale peuvent être combinées comme suit (méthode 'FoMCTI'):

$$\text{SI}(x,y) = \begin{cases} F_{\text{MCTI}}(x,y) & \text{if } M_{\text{MCTI}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{11}$$

Alternativement, la fusion de MCTI SI et GMC SI par le masque M_{MCTI} peut se faire comme suit (méthode 'FoMCTI2'):

$$\text{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_{\text{MCTI}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{12}$$

Dans ce qui suit, nous proposons une nouvelle approche qui consiste à estimer les objets dans la SI en utilisant les objets des trames de référence. La figure 12 montre l'approche proposée pour estimer les objets F_{FOMC}^i. Le masque M_{FOMC} est ensuite généré à partir

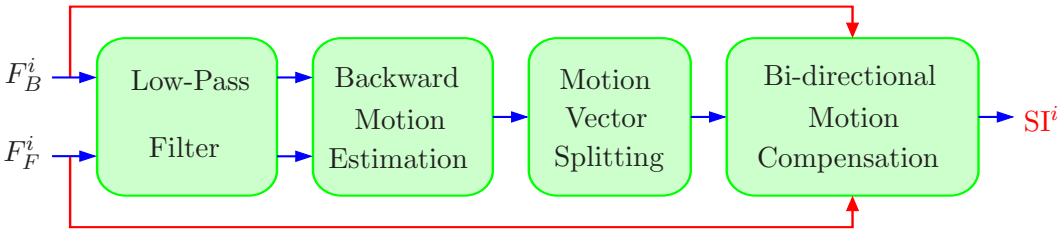

Figure 12: Méthode proposée pour l'estimation des objets.

des objets F^i_{FOMC}. Ce masque est utilisé pour combiner les mouvements global et local de deux façons possibles. La première méthode consiste à combiner GMC SI avec les objets estimés F_{FOMC} comme suit (méthode 'BmEst'):

$$\text{SI}(x,y) = \begin{cases} F_{\text{FOMC}}(x,y) & \text{if } M_{\text{FOMC}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{13}$$

La seconde méthode consiste à combiner GMC SI et MCTI SI en utilisant M_{FOMC} comme suit (méthode 'BmMCTI'):

$$\text{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_{\text{FOMC}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{14}$$

Les performances RD des méthodes proposées SADbin, FusElastic, BmEst, BmMCTI, FoMCTI et FoMCTI2 sont montrées pour les différentes séquences dans le tableau 9, en comparaison avec DISCOVER, en utilisant la métrique Bjontegaard [11], pour des tailles de GOP de 2, 4 et 8.

Toutes les méthodes de fusion proposées permettent d'apporter un gain par rapport à DISCOVER. FusElastic est supérieure à SADbin pour les séquences Stefan et Foreman avec une taille de GOP de 2 et 4, et pour toutes les séquences de test pour une taille de GOP de 8. Le gain est à $4,6$ dB par rapport à DISCOVER et $0,55$ dB par rapport à SADbin, pour la séquence ????, avec une taille de GOP de 8. La perte est à $0,04$ dB par rapport à SADbin pour la séquence Bus, avec une taille de GOP de 2. Les autres méthodes de fusion apportent presque les mêmes gains par rapport à DISCOVER.

Table 9: Performances RD pour différentes séquences, comparées au DISCOVER, en utilisant la métrique de Bjontegaard pour toutes les tailles de GOP.

Method	**SADbin**	**FusElastic**	**BmEst**	**BmMCTI**	**FoMCTI**	**FoMCTI2**	**Oracle**
GOP = 2							
Stefan							
Δ_R (%)	-17.97	-19.72	**-20.06**	-19.98	**-20.05**	-19.79	-20.38
Δ_{PSNR} [dB]	1.23	1.36	**1.39**	1.38	**1.39**	1.37	1.41
Foreman							
Δ_R (%)	-7.58	**-9.65**	-8.51	**-9.67**	-8.37	**-9.70**	-10.07
Δ_{PSNR} [dB]	0.45	**0.59**	0.52	**0.59**	0.49	**0.59**	0.61
Bus							
Δ_R (%)	-12.94	-12.51	-10.25	**-13.34**	-10.75	-11.25	-14.51
Δ_{PSNR} [dB]	0.79	0.75	0.61	**0.80**	0.64	0.68	0.87
Coastguard							
Δ_R (%)	-4.60	-4.32	-4.34	**-4.74**	-4.40	-4.33	-5.36
Δ_{PSNR} [dB]	0.23	0.22	0.22	**0.24**	0.22	0.21	0.27
GOP = 4							
Stefan							
Δ_R (%)	-40.66	-45.18	-45.73	-45.74	-45.80	**-45.71**	-46.42
Δ_{PSNR} [dB]	2.93	3.38	3.42	3.44	3.44	**3.45**	3.51
Foreman							
Δ_R (%)	-15.54	**-21.72**	-20.91	-21.81	-20.34	**-21.93**	-22.41
Δ_{PSNR} [dB]	0.90	**1.33**	1.25	1.32	1.19	**1.33**	1.36
Bus							
Δ_R (%)	-25.95	-25.97	-24.10	**-27.45**	-22.19	-23.67	-28.60
Δ_{PSNR} [dB]	1.60	1.57	1.41	**1.67**	1.34	1.40	1.78
Coastguard							
Δ_R (%)	-14.91	-16.48	-16.37	**-16.59**	-16.24	-15.70	-17.94
Δ_{PSNR} [dB]	0.61	0.68	0.68	**0.69**	0.67	0.65	0.75
GOP = 8							
Stefan							
Δ_R (%)	-51.56	-55.95	**-57.12**	-57.04	**-57.10**	-56.94	-57.84
Δ_{PSNR} [dB]	4.05	4.60	**4.72**	4.72	**4.73**	4.72	4.83
Foreman							
Δ_R (%)	-22.29	**-31.24**	-30.09	-31.01	-29.12	-30.78	-31.80
Δ_{PSNR} [dB]	1.29	**1.93**	1.84	1.92	1.76	1.91	1.97
Bus							
Δ_R (%)	-32.07	-32.82	-31.58	**-34.16**	-27.87	-28.53	-35.50
Δ_{PSNR} [dB]	2.04	2.07	1.97	**2.19**	1.72	1.74	2.31
Coastguard							
Δ_R (%)	-26.32	-29.50	**-30.37**	-29.73	-29.48	-28.19	-31.32
Δ_{PSNR} [dB]	1.10	1.24	**1.27**	1.26	1.23	1.18	1.35

VI - Conclusions

Dans le cadre de cette thèse, nous proposons différentes techniques pour améliorer les performances du DVC. Nous commençons par un raffinement successif de la SI après le décodage de chaque sous-bande DCT. Deux algorithmes sont proposés pour le raffinement des vecteurs de mouvement. Ensuite, nous proposons une nouvelle approche qui permet de générer la SI en se basant sur une estimation des vecteurs de mouvement dans les deux sens et sur le raffinement quad-tree. En outre, une approche visant à adapter la fenêtre de recherche après le décodage de la première sous-bande DCT est proposée. Dans cette approche, la SI est raffinée après le décodage de chaque sous-bande DCT, en utilisant la fenêtre de recherche adaptée. De plus, une nouvelle approche visant à ré-estimer la SI après le décodage de toutes les WZFs, dans le cas d'une large taille de GOP, est proposée.

Par la suite, la combinaison de l'estimation globale et locale est réalisée par le biais de différentes techniques proposées en vue d'améliorer la SI finale. La première technique permet d'exploiter les différences entre les blocs correspondants dans les deux estimations pour la fusion. La seconde technique vise à utiliser le SVM pour combiner les deux SIs. Ensuite, le raffinement de la fusion durant le processus de décodage est proposé en utilisant les coefficients DC décodés et la PDWZF. Enfin, nous proposons plusieurs nouvelles approches qui consistent à exploiter les objets pour la combinaison de l'estimation globale et locale. Plus spécifiquement, les courbes élastiques et l'estimation locale des objets sont utilisées dans ce genre de fusion.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ALF	Adaptive Loop Filter
BRF	Backward Reference Frame
BCH	Bose-Chaudhuri-Hocquenghem
BMA	Block Matching Algorithm
CAVLC	Context-Adaptive Variable-Length Codes
CABAC	Context-Based Arithmetic Coding
CRC	Cyclic Redundancy Check
DCT	Discrete Cosine Transform
DISCOVER	DIStributed COding for Video sERvices
DSC	Distributed Source Coding
DVC	Distributed Video Coding
DP	Dynamic Programming
DPCM	Differential Pulse-Code Modulation
INSI	Initial Side Information
FPS	Frames per Second
FRI	Finite Rate of Innovation
FRF	Forward Reference Frame
GMC	Global Motion Compensation
GOP	Group Of Pictures
HEVC	High Efficiency Video Coding
LCEC	Low Complexity Entropy Coding

PRISM	Power-efficient, Robust, hIgh compression, Syndrome-based Multimedia coding
KL	Karhunen-Loeve
LDPC	Low Density Parity-Check
LDPCA	Low Density Parity-Check Accumulate
PDF	Probability Density Function
PDWZF	Partially Decoded Wyner-Ziv Frame
PSNR	Peak Signal-to-Noise Ratio
QI	Quantization Index
QP	Quantization Parameter
QS	Quantization Step
JVT	Join Video Team
KF	Key-Frame
MAD	Mean Absolute Difference
MCTI	Motion-Compensated Temporal Interpolation
MV	Motion Vector
PB	PetaByte
RD	Rate-Distortion
RLE	Run-Length Encoding
SA	Search Area
SI	Side Information
SIFT	Scale-Invariant Feature Transform
SIG	Side Information Generation
SMV	Symmetric Motion Vectors
SRV	Square Root Velocity
SSD	Sum Square Distance

SVD	Singular Value Decomposition
SVM	Support Vector Machine
MPEG	Moving Picture Experts Group
VCEG	Video Coding Experts Group
VoIP	Voice over IP
VoD	Video-on-Demand
WMAD	Weighted Mean Absolute Difference
WWVC	Witsenhausen-Wyner Video Coding
WZ	Wyner-Ziv
WZF	Wyner-Ziv Frame

Chapter 1

Introduction

The task of compression is essential to communication systems, since it allows to efficiently store or transmit data. Despite the fact that broadband networks continue to grow, a stage of compression is always essential. Fig. 1.1 shows the expected growth of all consumer Internet traffic (Cisco report [1]): Internet video, file charing, Web/Email/Data, Online gaming and Voice over IP (VoIP), in PetaBytes (PB = 10^{15} Bytes) per month, over the years 2011 to 2016. The sum of all forms of IP video such as Internet video, IP Video-on-Demand (VoD), video files exchanged through file sharing, video-streamed gaming, and video conferencing will ultimately reach 86 percent of total IP traffic. Moreover, Only Internet video (excluding file sharing and gaming) will account for 55 percent of consumer Internet traffic in 2015 (see Fig. 1.1). Besides, this Cisco report also states that every second, 1.2 million minutes of video content will cross the network in 2016.

Therefore, the challenge is to visualize high quality data in real time and to store huge amounts of data in less space. For digital video coding, the standardization efforts of ISO/IEC MPEG-x and ITU-T H.26x are mainly based on the Discrete Cosine Transform (DCT) and inter-frame and intra-frame predictive coding. In addition, High Efficiency Video Coding (HEVC) is currently being studied as a successor to H.264/AVC and will soon become an international standard. In all these standards, the encoder exploits the spatial and temporal redundancy existing in a video sequence. In this case, the encoder is significantly more complex than the decoder (with a typical factor of 5 to 10 [15]). This kind of architecture is well-suited for applications where the video sequence is encoded once and decoded many times, such as in broadcasting or video streaming.

In the recent years, this architecture has been challenged by several emerging applications such as wireless video surveillance, multimedia sensor networks, wireless PC cameras, and mobile cameras phones. In these new applications, it is essential to have a low complexity encoding, while possibly affording a high complexity decoding.

Distributed Video Coding (DVC) is a recently proposed paradigm in video communication that fits well these scenarios, since it enables the exploitation of the similarities among successive frames at the decoder side, making the encoder less complex. Consequently, the

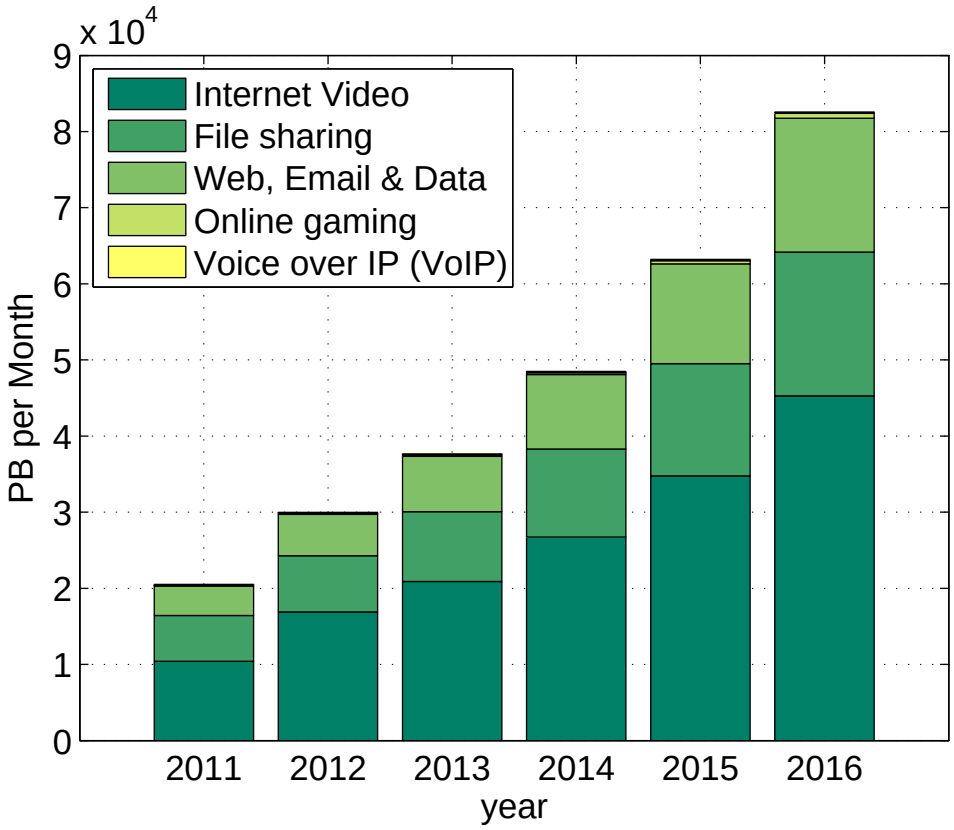

Figure 1.1: Growing of all consumer Internet traffic: Internet video, file charing, Web/Email/Data, Online gaming and Voice over IP (VoIP), in PetaBytes (PB) per month, over the years 2011 to 2016.

complex tasks of motion estimation and compensation are shifted to the decoder. From information theory, the Slepian-Wolf theorem for lossless compression [2] states that it is possible to encode correlated sources (let us call them X and Y) independently and decode them jointly, while achieving the same rate bounds that can be attained in the case of joint encoding and decoding. The Wyner-Ziv (WZ) theorem [3] extends the Slepian-Wolf one to the case of lossy compression of X when Side Information (SI) Y is available at the decoder.

Based on these theoretical results, practical implementations of DVC have been proposed [6, 16]. The European project DISCOVER [4, 5] came up with one of the most efficient and popular existing architectures. The DISCOVER codec is based on the Stanford scheme [6]. More specifically, the sequence images are split into two sets of frames, key frames (KFs) and Wyner-Ziv frames (WZFs). The Group of Pictures (GOP) of size n is defined as a set of frames consisting of one KF and $n-1$ WZFs. The KFs are independently encoded and decoded using Intra coding techniques such as H.264/AVC Intra mode or JPEG2000. The WZFs are separately transformed and quantized, and a systematic channel code is applied to the resulting coefficients. Only the parity bits are kept, and sent

to the decoder upon request. This can be seen as a Slepian-Wolf coder applied to the quantized transform coefficients. At the decoder, the reconstructed reference frames are used to compute the Side Information (SI), which is an estimation of the WZF being decoded. In order to produce the SI, Motion-Compensated Temporal Interpolation (MCTI) [7] is commonly used. Finally, a channel decoder uses the parity information to correct the SI, thus reconstructing the WZF. Straightforwardly, generating a more accurate SI is very important, since it would result in a reduced amount of parity information requested by the decoder through the return channel. At the same time, the quality of the decoded WZF would be improved during reconstruction.

The goal in terms of compression efficiency would be to achieve a coding performance similar to the best available hybrid video coding schemes; but DVC has not reached the performance level of classical inter frame coding yet. This is in part due to the quality of the SI, which has a strong impact on the final Rate-Distortion (RD) performance.

In this Ph.D. thesis, we aim at improving the estimation of the SI by proposing different approaches. First, a technique consisting in progressively refining the estimation of the SI after each decoded DCT band is proposed. Then, an adaptive search area algorithm is introduced. Afterwards, a new method based on backward and forward motion estimation for SI generation is proposed. Finally, different techniques for the fusion of the global and local motion estimations are given to derive the SI. In more details, the manuscript is organized as follows:

- **Chapter 2 - State of the art :** In this chapter, we give a brief introduction on the video coding standards such as H.264/AVC and HEVC. Then, we describe in detail the origins of distributed source coding and DVC. Specifically, the PRISM, DISCOVER (*i.e.* based on Stanford scheme [6]) and VISNET II architectures are described, together with the MCTI technique.

  Chapters 3 to 6 present the contributions of this thesis.

- **Chapter 3 - Successive refinement of side information generation :** In this chapter, we propose a new approach for enhancing the SI in transform-domain DVC. This solution consists in progressively improving the SI after each decoded DCT-band. It is particularly efficient for high motion regions and in the case where KFs are separated by a significant number of WZFs. We first start by generating an Initial Side Information (INSI) by using the backward and forward reference frames similarly to the SI generated in the DISCOVER codec (MCTI technique). The decoder reconstructs a Partially Decoded WZF (PDWZF) by correcting the INSI with the parity bits of the first DCT-band. Then, the PDWZF, along with the backward and forward reference frames, is exploited to refine the INSI. The refinement approach consists of three modules: Suspicious Vector Detection, Refinement, Mode Selection and Motion Compensation. More specifically, in the module that consists in refining the motion vectors, we propose two different algorithms for enhancing the quality of

the final decoded WZF. Finally, we correct this refined SI with the parity bits of the next DCT-band, and the same procedure is repeated for the decoding of all DCT bands. Experimental results show that the proposed method allows a significant gain compared to DISCOVER codec, VISNET II codec, and previous techniques.

- **Chapter 4 - Side information improvement techniques :** In this chapter, we first propose a new scheme for SI generation. This scheme consists in estimating backward and forward motion vectors using quad-tree refinement. Then, reliable motion vectors are selected from the backward and forward estimations.

  Second, we propose a new method for improving the SI using an adaptive search area, along with our successive refinement technique. In the proposed approach, variable search areas are initially set according to the temporal distance between the neighboring reference frames. After generating the INSI, the decoder reconstructs a PDWZF by correcting the INSI with the parity bits of the first DCT-band. Afterwards, the PDWZF is used to adapt the initial search area, along with the backward and forward reference frames. Furthermore, the adapted search area is used to refine the INSI. Finally, the improved SI is corrected with the parity bits of the next DCT-band and so on.

  For large GOP sizes, it is known that the quality of the central SI is worse than the quality of the lateral ones, because the reference frames used for estimating the central WZF are farther apart. The consequence is that the PSNR of the decoded frames fluctuates within the GOP. For this reason, we propose a new approach to re-estimate the SI using the already decoded WZF and the adjacent decoded frames (WZF or KF). During the re-estimation procedure, an adaptive search area and a variable block size are also used. Finally, the WZFs are reconstructed with an improved quality, using the same parity bits sent during the first step.

- **Chapter 5 - Fusion of global and local motion estimation :** In this chapter, we propose a new approach that consists in combining global and local motion estimation to improve the SI. The local SI is directly estimated using the MCTI technique. For global SI, the feature points of the reference frames and the original WZF are extracted by applying the Scale-Invariant Feature Transform (SIFT) [17] algorithm. Afterwards, the matching is carried out between the feature points of WZF and reference frames. Then, in order to remove the points that exist on individual objects of the frame (local motion), an algorithm robust to outliers is proposed. Moreover, an affine model is estimated between the WZF and the reference frames using the corresponding feature points. The global parameters are sent to the decoder in order to generate the global SI.

  Furthermore, the global and local SI frames are combined using two approaches, in order to improve the estimation of the SI. The first approach aims at using the

differences between the corresponding blocks in the global and local SI frames, in the combination process. The second one consists in using Support Vector Machine (SVM) to combine the two estimations. Afterwards, the decoded DC coefficients and the PDWZF are used to improve the combination during the decoding process. Experimental results show that the proposed techniques can achieve a significant gain compared to DISCOVER codec. In addition, we show that the performance of the proposed methods is better than the performance of H.264/AVC Intra and H.264/AVC No motion for several test sequences.

- **Chapter 6 - Fusion based on foreground objects estimation :** In this chapter, we propose new methods to combine the global and local motion estimation using the foreground objects of the reference frames. These foreground objects are used, in order to generate the foreground objects in the SI frames, while the background pixels are directly taken from the global SI. First, we propose a new method based on elastic curves [14, 18] in order to estimate the foreground object contours in the SI frame. Based on the estimated contours, the fusion of the global and local SI is performed.

  Second, the MCTI technique is directly applied on the backward and forward foreground objects in order to generate the foreground objects in the SI frame. Furthermore, we propose an approach based on foreground object motion estimation to generate the foreground objects, using the backward and forward foreground objects. Based on the estimated objects, two approaches are proposed to combine global and local motion estimations.

Furthermore, conclusions are drawn in **Chapter 7**, with the perspectives and the producing papers during this PhD thesis are shown in **Publications**.

Finally, in **Annex A**, signals with finite rate of innovation are introduced. Furthermore, we show the application of such signals in DVC and prove that the utilization of these signals in DVC can allow a gain in performance for video sequences containing fixed background, but fails when a real sequence is used.

Chapter 2

State of the art

In this chapter, we recall the main concepts and depict the state of the art for the thwo topics of this thesis manuscript: video compression and Distributed Source Coding (DSC).

We first give a brief introduction of video coding concepts. Then, we describe the most efficient among current standards H.264/AVC and its successor HEVC.

Afterwards, the principle of distributed source coding is described and we show the fundamental results of the Slepian-Wolf theorem for lossless compression and Wyner-Ziv theorem for lossy compression. Based on these theoretical results, many practical systems using the concept of distributed video coding (DVC) have been developed in the recent years. We illustrate the main architectures. We also discuss the motion compensated temporal interpolation technique used in DVC.

Contents

2.1 Video Coding Concepts

The Human Visual System can form, transmit and analyze $10 - 12$ separate images per second and perceive them individually [19]. This principle is at the basis of the video sequence representation, since an illusion of continuity is created if more than 12 images

per second are shown. For this reason, a rate of capture of 15 to 30 images per second is usually considered, in order to represent a visual scene by a digital video sequence. This rate is referred to as the number of Frames per Second (FPS). FPS can be 60 in some cases when the video sequence is shown on the screen.

Normally, an uncompressed digital video sequence requires a very large bitrate. Therefore, compression of video (*i.e.* reducing the number of bits used in its representation) is necessary in practice. The compression ratio is defined as:

$$\text{Compression Ratio} = \frac{\text{Uncompressed Size}}{\text{Compressed Size}}. \tag{2.1}$$

Lossless compression [20] of images and video is not very efficient since it only gives a moderate amount of compression. The JPEG-LS compression standard [21] (image lossless compression) achieves a compression ratio of 3 to 4 times. However, several types of data contain statistical redundancy and can be effectively compressed using lossless compression.

Lossy compression [20] is necessary in order to achieve a high compression ratio for digital video at the expense of a loss of visual quality, since the decompressed video is not identical to the original one. Lossy compression in video is based on the principle of eliminating some elements without significantly affecting the viewer's perception of visual quality.

In image coding, spatial correlation is exploited in order to reduce the size of the image. The spatial correlation represents the correlation among neighboring samples. Commonly, the spatial correlation is significant within an image. Many techniques can be used for lossless image compression such as Run-Length Encoding (RLE), Huffman coding, Arithmetic coding, dictionary-based coding, ... etc, that also allow to exploit statistical redundancy. On the other side, many methods can be used in lossy image compression such as: chroma subsampling, quantization of the transformed coefficients (by applying first a Discrete Cosine Transform (DCT), a wavelet transform), ... etc. These transformations are commonly followed by quantization and entropy encoding. DCT is very used in image and video coding.

In a digital video sequence, correlation among successive frames is referred to as temporal correlation. In general, both spatial correlation (within a specific image) and temporal correlation (among successive frames) are present in a video sequence. The temporal correlation is often high, especially if the FPS is high and/or if the video motion is slow. These correlations must be exploited in order to compress the video sequence. Thus, video compression plays an important role in transmission and storage of video data. The basic aim of the compression process is to remove temporal and spatial redundancies existing in digital video sequences.

Two large groups called Moving Picture Experts Group (MPEG) and Video Coding Experts Group (VCEG) have been created in order to produce video coding standards.

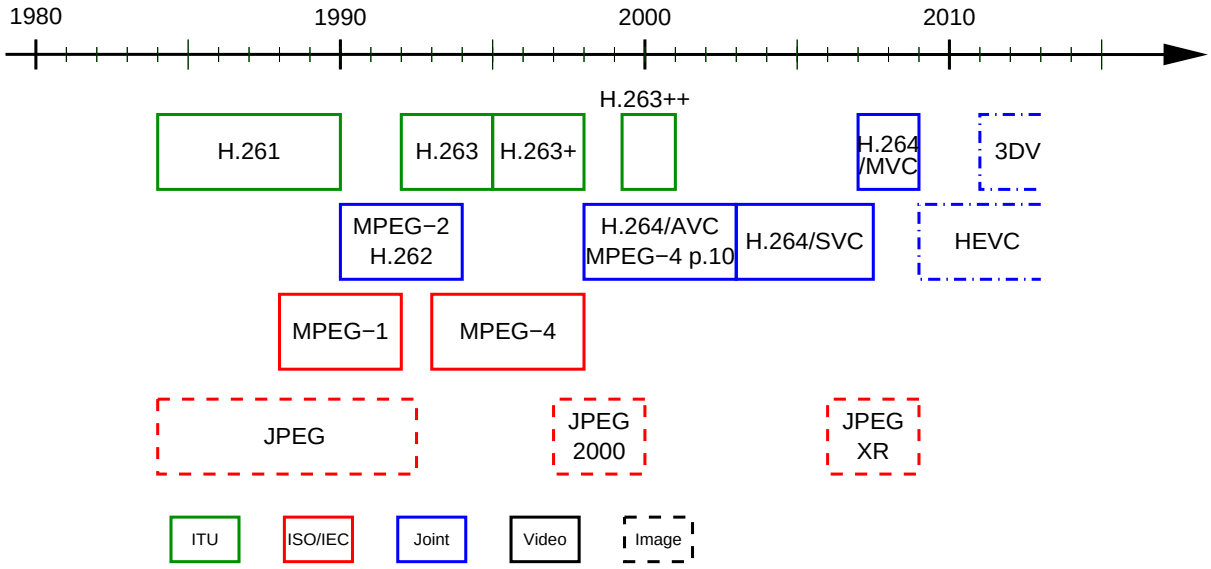

Figure 2.1: Chronology of international image and video coding standards

MPEG developed MPEG-1 and two successful MPEG-2 [22] and MPEG-4 [23] standards for coding video and audio. These standards (MPEG-2 and MPEG-4) are widely used for communication and storage of video sequences. The first used video telephony standard H.261 [24] and its successor H.263 [25] were developed by the standardization efforts of VCEG. The two groups MPEG and VCEG created the Joint Video Team (JVT), in order to generate the international standard H.264/AVC (also referred to as H.264/MPEG-4 Part 10) and more recently HEVC (High Efficiency Video Coding). Fig. 2.1 shows the chronology of the international image and video coding standards.

2.2 H.264/AVC Video compression

H.264/AVC is a standard for video compression [15, 26], which was jointly developed by MPEG and VCEG. Nowadays, H264/AVC is one of the most commonly used formats for video compression and storage. Most of the basic modules of H.264/AVC such as prediction, transform, quantization and entropy encoding (except loop filter) are present in previous standards like MPEG1-4 and H26x, but the important changes in H.264/AVC occur in the details of each functional module.

Fig. 2.2 shows an encoder block diagram of H.264/AVC. The pictures of a video sequence are split into the two categories of Intra-picture and Inter-picture. The first picture of the sequence is coded in Intra mode. In such mode, the encoder only exploits the spatial correlation in the picture by choosing neighboring samples to estimate a good prediction to the current sample. On the other hand, Inter coding is based on inter-picture temporal prediction such as motion compensation using previously decoded pictures. In this case, the encoder tries to estimate motion vectors and identify the reference pictures to predict

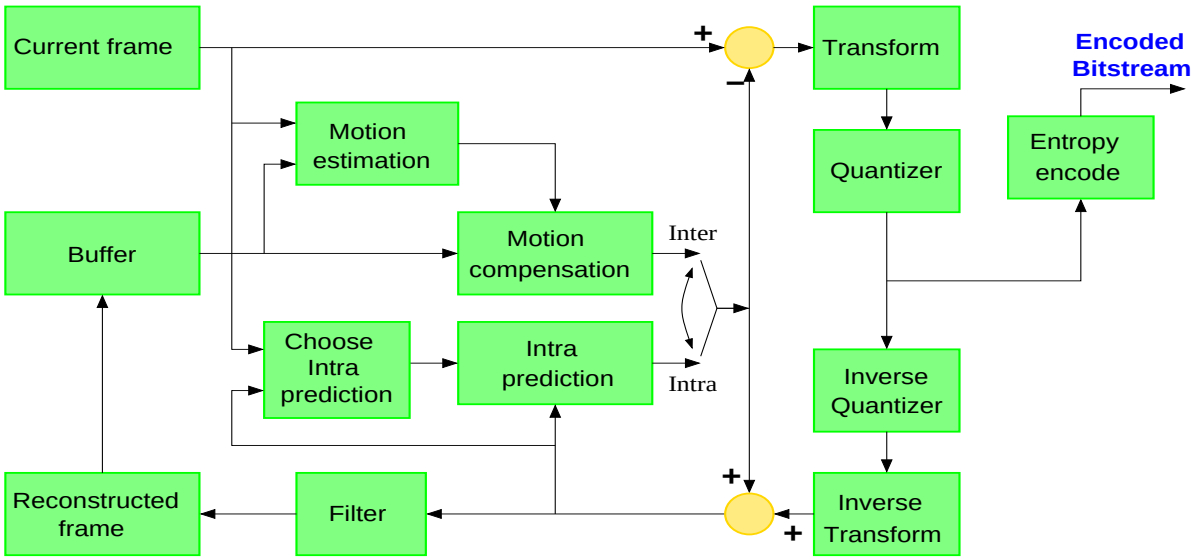

Figure 2.2: H.264/AVC encoder

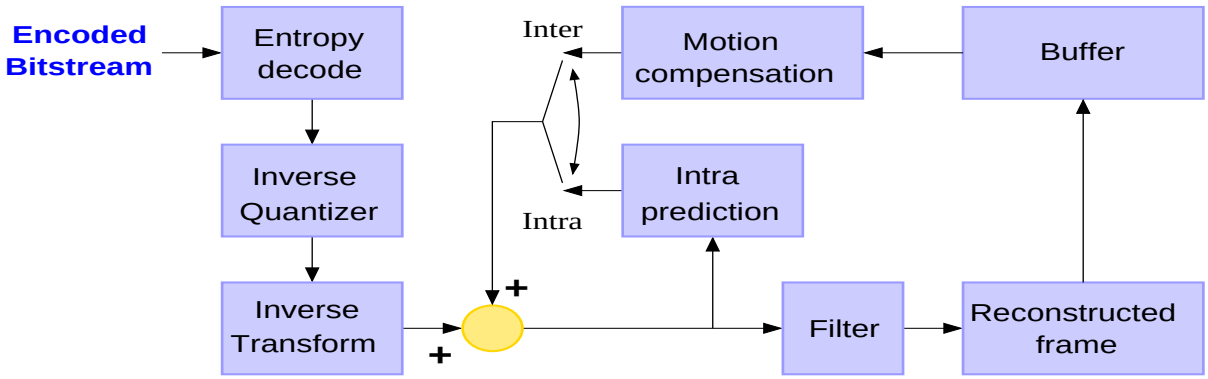

Figure 2.3: H.264/AVC decoder

the samples of each block. In both Intra and Inter prediction, the encoder computes a residual of the prediction. This residual is defined as the difference between the original samples and the predicted samples. The residual is transformed and the obtained transform coefficients are scaled and approximated using scalar quantization. Finally, the quantized coefficients are entropy encoded.

Fig. 2.3 shows the block diagram of an H.264/AVC decoder. Note that the encoder incorporates a model of the decoding process, since it computes the same prediction values that are computed in the decoder for the prediction of samples in the current picture. Thus, both the encoder and the decoder have the same prediction values of the samples. At the receiver, lossless decoding is first carried out, and the syntax elements are recovered. In particular, the quantized coefficients are inverse scaled and inverse transformed, in order to form the approximated residual. It also performs the prediction process using the motion data and the reference samples. Then, it adds the approximated residual to the prediction of the samples. Finally, the obtained picture is fed into a deblocking filter.

In H.264/AVC, as in prior standards, YCbCr color space together with reducing the sampling resolution of the Cb and Cr chroma information is used. Component Y is called luma and represents brightness. Cb and Cr represent the blue-difference and red-difference chroma components. The human visual system is more sensitive to luma than chroma. For this reason, H.264/AVC typically uses (it is not mandatory and some profiles can encode 4:4:4) a 4:2:0 sampling (*i.e.* the chroma component has one fourth the number of samples of the luma component), with a precision of 8 bits per sample.

In practice, H.264/AVC video compression covers a large range of applications such as digital video broadcast, internet streaming, video on demand, etc. Furthermore, H.264/AVC allows a significant bit rate reduction compared to the previous standards. In this section, we describe the main tools of the H.264/AVC standard.

- **Subdivision of a picture into macroblocks and slices** - Each picture of a video sequence is split into small pictures area of 16×16 samples of the luma component and 8x8 samples of each of the two chroma components. These small pictures are referred to as macroblocks. The samples of a macroblock are either spatially or temporally predicted. The prediction residual is subdivided into blocks. An integer transform is applied to each block and the transform coefficients are quantized and entropy coded.

  A slice is a group of successive macroblocks (it may contain one or more macroblocks). H.264/AVC supports five slice-coding types. An I slice contains only intra-coded macroblocks that are predicted from previously decoded samples in the same slice. A P slice can contain inter-coded macroblocks that are predicted from samples in previously decoded pictures, as well as intra coded macroblocks and, eventually, skipped macroblocks. A B slice contains inter-coded macroblocks that are predicted from previous and subsequent decoded pictures. The remaining two slice types are Switching P and Switching I slices, which are specified for efficient switching between bit-streams coded at various bit rates.

- **Spatial Intra Prediction** - In Intra mode, prediction samples of a block are formed based on neighboring previously encoded and reconstructed samples. The prediction process is performed for each 4×4 block ($\text{Intra}_{4\times4}$) or 16×16 macroblocks ($\text{Intra}_{16\times16}$). 9 prediction modes for $\text{Intra}_{4\times4}$, 4 prediction modes for $\text{Intra}_{16\times16}$ and four modes for chroma components can be used. I_{PCM} mode allows the direct transmission of the samples values, without prediction or transformation. The $\text{Intra}_{4\times4}$ mode is well suited for the encoding of the picture parts with significant details. On the contrary, $\text{Intra}_{16\times16}$ mode is more suited for coding smooth areas in a picture.

  Fig. 2.4 shows the nine modes of 4×4 block. The samples **A-H** (above) and **I-M** (left) have previously been encoded and reconstructed and are available in both the encoder and the decoder. These samples are considered as reference samples. The

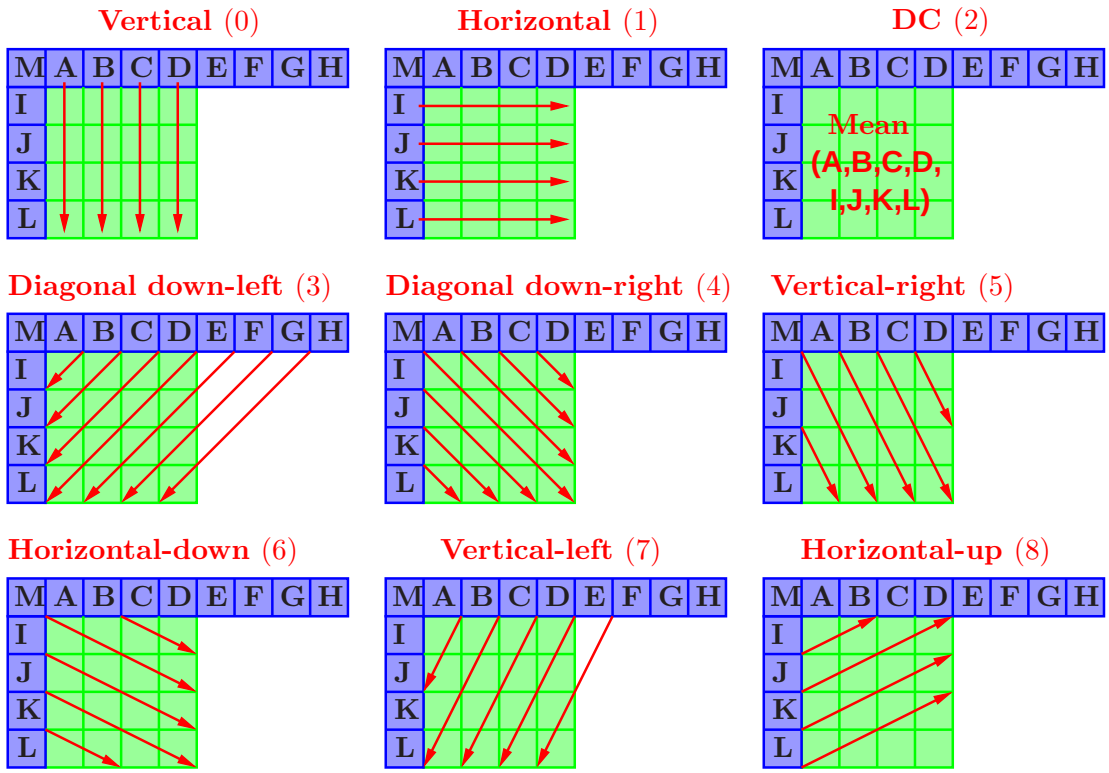

Figure 2.4: 4×4 luma prediction modes.

samples of a 4×4 block (green block) are computed based on the samples **A-M**. For mode 0, the samples are vertically extrapolated from **A, B, C** and **D**. For mode 1, the samples are horizontally extrapolated from **I, J, K** and **L**. For mode 2, the samples are predicted by the mean of samples **A-D** and **I-L**. In this case, all samples in the block have the same predicted value. For modes $3-8$, the predicted samples of the block are estimated using a weighted average of the reference samples **A-M**. The prediction of the samples for $\text{Intra}_{16\times16}$ and chroma components operates in a similar way.

- **Motion-Compensated prediction** - Inter prediction in H.264/AVC estimates a prediction of the samples in the current block using one or more previously encoded and decoded pictures. Fig. 2.5 shows a range of block sizes (16×16, 16×8, 8×16, 8×8, 8×4, 4×8 and 4×4) that are included in H.264/AVC. These partitions give rise to a large number of possible combinations within each macroblock. H.264/AVC supports quarter-sample resolution in the estimation of the motion vectors for luma component and one-eighth-sample resolution for the chroma components.

  Fig. 2.6 illustrates multi-picture motion-compensated prediction. That is, more than one prior coded picture can be used as reference for motion-compensated prediction in H.264/AVC. Thus, both the encoder and the decoder have to store the reference pictures in a multi-frame buffer.

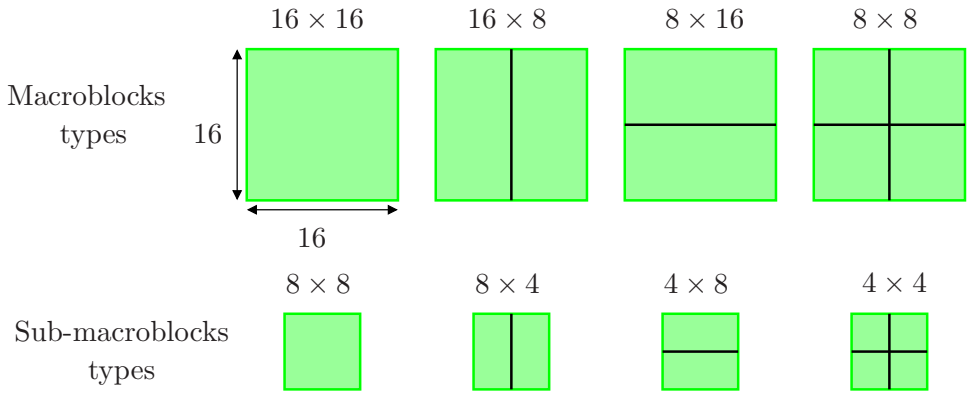

Figure 2.5: Partitioning of a macroblock (top) and a sub-macroblock (bottom) for motion-compensated prediction.

- **Transform, Scaling, and Quantization** - H.264/AVC standard uses three transforms depending on the type of the residual data. First, a Hadamard transform is used for the 4×4 array of luma DC coefficients in intra macroblocks in 16×16 mode. In this case, the transform matrix is defined as:

$$H_{4\times4} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}$$

Second, a Hadamard transform for the 2×2 array of chroma DC coefficients is used. The matrix of this transform is given by:

$$H_{2\times2} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Finally, for all other 4×4 blocks in the residual data, an integer DCT with similar properties to that of a normal 4×4 DCT is used. The transform matrix is given by:

$$H_{\mathrm{DCT}} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix}$$

Since the inverse integer DCT consists in exact integer operations, inverse-transform mismatches are avoided. In H.264/AVC, 52 values of a Quantization Parameter (QP) are defined. The Quantization Step (QS) is doubled by 2 for each increment of 6 in the value of QP. For an input residual block $\mathbf{X}$, the transformed block $\mathbf{Y}$ is obtained

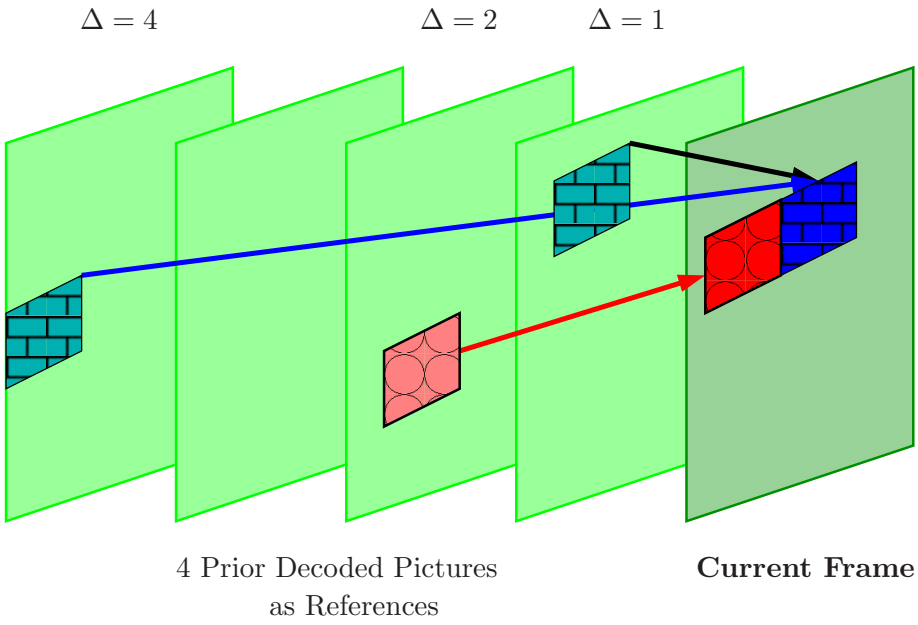

Figure 2.6: Multi-frame motion compensation. In addition to the motion vector, also picture reference parameters (Δ) are transmitted. The concept is also extended to B slices.

by:

$$\mathbf{Y} = H\mathbf{X}H^T, \tag{2.2}$$

where H is the matrix transform and H^T the transpose matrix of H. The transform coefficients are scaled and quantized depending on the selected value of QP. Then, the quantized coefficients are scanned in a zig-zag fashion and further entropy encoded.

- **In-Loop Deblocking Filter** - Block-based coding results in visually noticeable discontinuities along the block boundaries due to coarse quantization at low bit rates. For this reason, H.264/AVC includes an adaptive deblocking filter in the prediction loop. The filter consists in reducing the blockiness without much affecting the sharpness of the content. As a result, the subjective quality is significantly improved, and a rate reduction of $5-10\%$ can be achieved, at the same objective quality of non-filtered video.

- **Profiles and Levels** - Four Profiles are defined in H.264/AVC, each supporting a particular set of coding functions and specifying what is required of an encoder or a decoder that complies with the given Profile. Fig. 2.7 shows the relationships between the Profiles and the coding tools supported by the H.264/AVC standard. It is clear that the Baseline Profile is a subset of the Extended Profile, but not of the Main Profile.

  The Baseline Profile includes intra and inter-coding of I-slices and P-slices and entropy coding with Context-Adaptive Variable-Length Codes (CAVLC). The Main

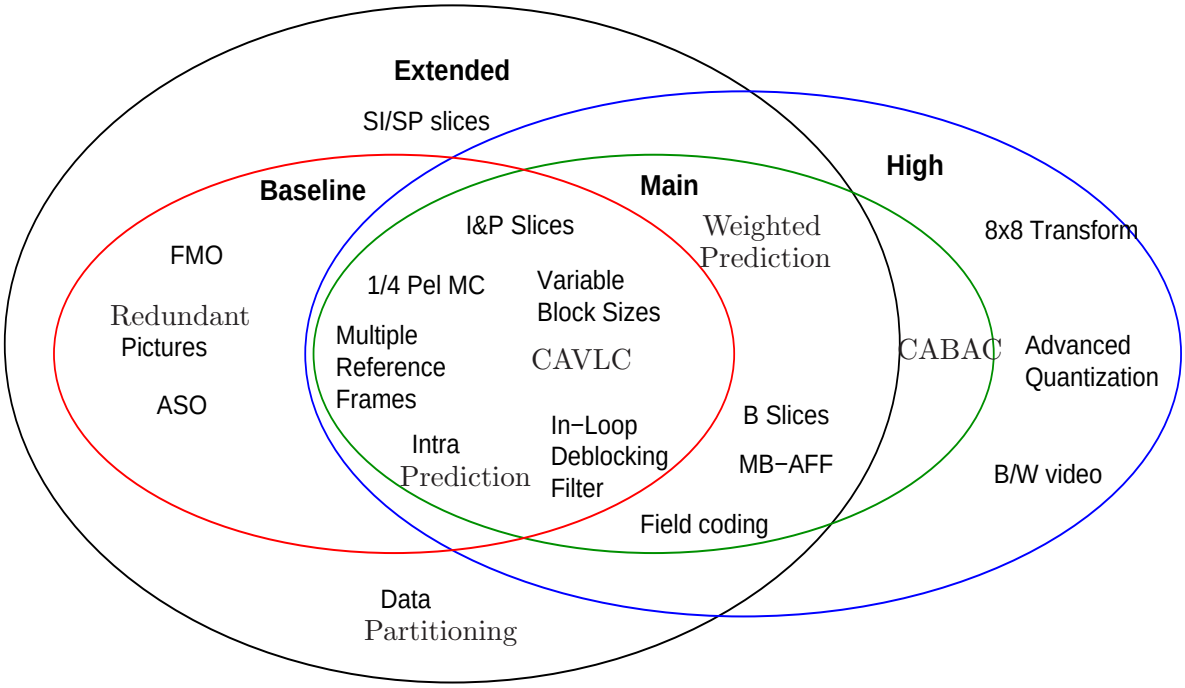

Figure 2.7: Four Profiles in H.264.

Profile includes support for interlaced video, inter-coding using B-slices, inter-coding using weighted prediction and entropy coding using Context-Based Arithmetic Coding (CABAC). The Extended Profile does not support interlaced video or CABAC, but adds modes to enable efficient switching between coded bitstreams and improved error resilience (Data Partitioning). However, each Profile has sufficient flexibility to support a wide range of applications.

2.3 HEVC Video compression

MPEG and VCEG are currently developping a new video compression standard called High Efficiency Video Coding (HEVC). Comparing to the actual standard H.264/AVC, HEVC [27, 28] aims at achieving a bitrate reduction of 50% with comparable image quality. This new standard is predicted to appear in January 2013. It should be able to balance computational complexity, coding efficiency, robustness to errors and delay time depending on the application.

The HEVC standard has the same basic video coding architecture as the previous video coding standard H.264/AVC. Its architecture is based on:

- Block hybrid coding scheme - Advanced intra and inter coding modes
- Motion compensated prediction
- Transform coding with high efficiency entropy coding - Context Adaptive Binary

Arithmetic Coding (CABAC)

- Loop filter - Deblocking filter or Adaptive Loop Filter (ALF)

Moreover, the major differences between HEVC and H.264/AVC standards are related to:

- Flexible quad-tree partitioning structure with a larger block size 64×64
- New partitions - Coding units, Prediction units and Transform units

2.4 Distributed Source Coding

Distributed Source Coding (DSC) is a compression paradigm focusing on coding two or more dependent random signals in a distributed manner. Each signal is independently encoded and the bitstreams are sent to a single decoder. The decoder aims at performing a joint decoding, in order to exploit correlation among dependent signals.

Let X and Y be two discrete sources. For lossless coding, the encoding rates when X and Y are independently encoded are lower-bounded by their marginal entropy:

$$\begin{cases} R_X \geq H(X) \\ R_Y \geq H(Y) \end{cases} \tag{2.3}$$

In this case, the correlation between X and Y is not exploited neither at the encoder, nor at the decoder. Thus, the total transmission rate R associated to the independent encoding and decoding of X and Y is given by:

$$R = R_X + R_Y, \tag{2.4}$$

where R_X and R_Y are respectively the encoding rates of X and Y. On the other side, the encoding rate, when both the encoding and the decoding of X and Y are performed jointly (Fig. 2.8), is:

$$\begin{cases} R \geq H(X,Y) \\ H(X,Y) \leq H(X) + H(Y) \end{cases} \tag{2.5}$$

$H(X,Y)$ is the joint entropy of X and Y. In this configuration, if the source X is transmitted with a rate R_X, the source Y can be transmitted with a rate $R_Y = H(X,Y) - R_X$, without having any loss at the decoder.

In distributed coding, the encoding of X and Y is performed independently and the decoding is performed jointly (Fig. 2.9). From information theory, the Slepian-Wolf theorem for lossless coding [2] states that it is possible to encode X and Y independently and decode them jointly, achieving the same rate bounds which can be attained in the case of

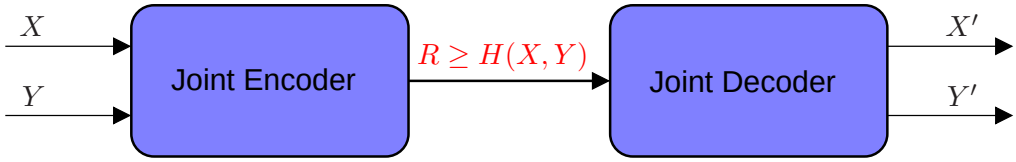

Figure 2.8: Traditional coding paradigm.

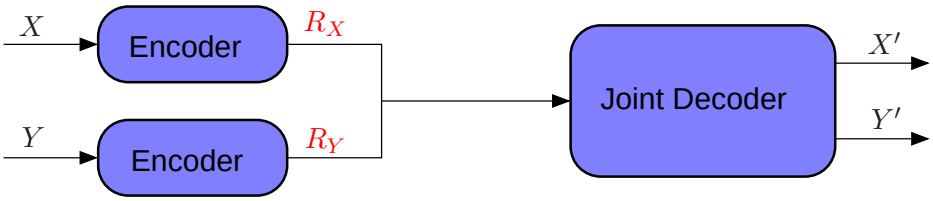

Figure 2.9: Distributed source coding paradigm.

joint encoding and decoding. This situation is showed in Fig. 2.10; the encoding rates can be:

$$\begin{cases} R_X \geq H(X|Y) \\ R_Y \geq H(Y|X) \\ R = R_X + R_Y \geq H(X,Y) \end{cases} \tag{2.6}$$

The Slepian-Wolf theorem can be considered in practical applications that use channel coding. In the case of two statistically dependent sources X and Y, Y can be considered as a noisy version of X. Therefore, Y can be written as follows:

$$Y = X + \text{Error} \tag{2.7}$$

The process of encoding and decoding of X is described as follows: X is fed to a channel encoder and only the generated parity information is sent to the decoder. At the decoder, Y is considered to be a noisy version of X. Thus, the received parity information of X is concatenated with Y and fed into a channel decoder. This process consists in correcting the errors in Y (the difference between X and Y) using the parity information of X. In this case, the encoding rate of the source X can be limited between two boundaries as follows:

$$H(X|Y) \leq R_X \leq H(X) \tag{2.8}$$

The Wyner-Ziv (WZ) theorem [3] extends the Slepian-Wolf theorem to the case of lossy compression. Fig. 2.11 illustrates the basic WZ architecture. At the decoder, a Side Information (SI) Y is estimated, in order to constitute a noisy version of X.

The distortion D between the original information X and the decoded X' can be defined as follows:

$$D = \mathbb{E}[d(X, X')] \tag{2.9}$$

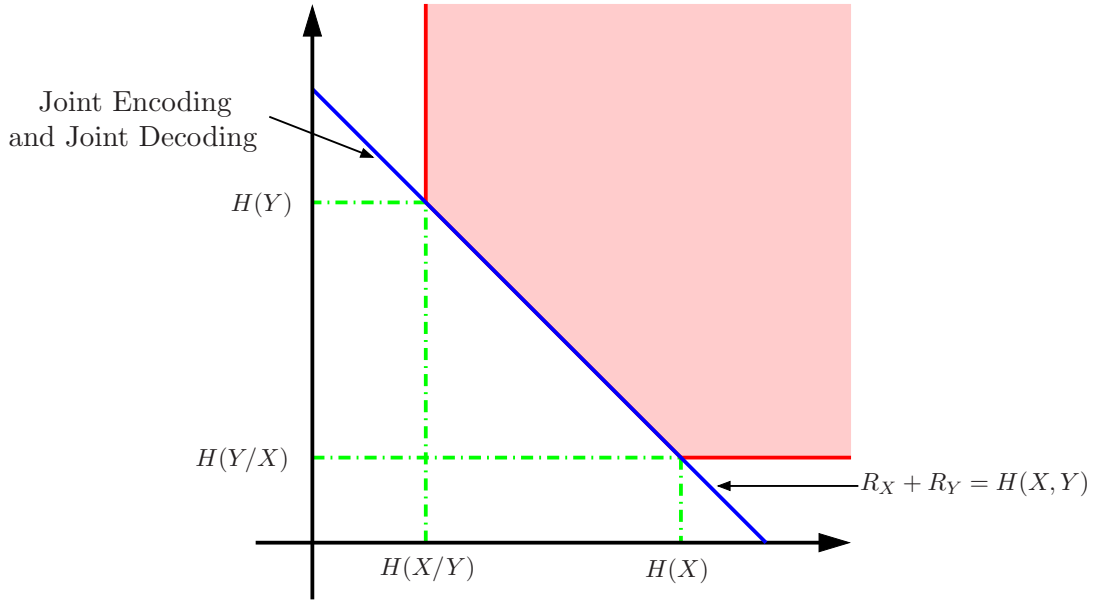

Figure 2.10: Achievable rate region following the Slepian-Wolf theorem.

where d is a distortion metric (such as the mean squared error). Let $R^{WZ}_{X|Y}(D)$ be the compression rate when the SI is available only at the decoder. However, if the SI is available at both the encoder and the decoder, the compression rate is denoted as $R_{X|Y}(D)$. In this situation, Wyner and Ziv proved that:

$$R^{WZ}_{X|Y}(D) - R_{X|Y}(D) \geq 0. \tag{2.10}$$

Therefore, when the statistical dependency is exploited only at the decoder, the minimum rate to transmit X at the same distortion D increases compared to the case where the statistical dependency is exploited at both the encoder and the decoder. However, Wyner and Ziv proved that the rate loss can be zero in the case where X and Y are jointly Gaussian and a mean-square distortion metric is used.

Based on these results, the authors in [29] demonstrated the equality in the compression rate for sequences defined by the sum of arbitrarily distributed SI and independent Gaussian noise. Zamir[30] proved that the rate loss is less than 0.5 bits per sample, when the SI is exploited only at the decoder side, compared to the case where both the encoder and decoder exploit the SI.

In conclusion, for the systems that exploit the statistical dependency only at the decoder (to moderate the encoder complexity), the Slepian-Wolf and WZ theorems prove that it is possible to achieve the same rate as the systems where the dependency is exploited at the encoder and the decoder. For lossy compression, this is valid only if the two sequences are jointly Gaussian and a mean square error distortion metric is considered.

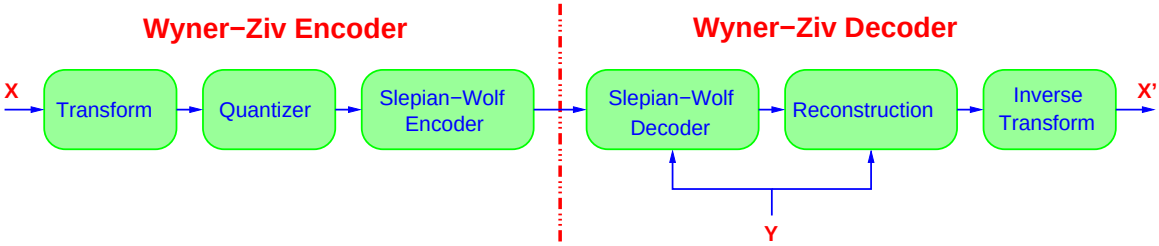

Figure 2.11: Block diagram of the basic Wyner-Ziv codec.

2.5 Distributed Video Coding

Distributed Video Coding (DVC) is a new paradigm in video communication which is based on the principles of DSC. The theories of Slepian-Wolf and WZ can be applied to the transmission of a video sequence. The idea behind DVC is to exploit the temporal correlation among successive frames of a video sequence in the decoding process, making the encoding one less complex.

In video coding standards like ISO/IEC MPEG and ITU-T H.26x, motion estimation and compensation are performed at the encoder in order to achieve high rate-distortion performance, while the decoder can directly use the received motion vectors to decode the sequence. This architecture makes the encoder much more complex than the decoder [15]. This asymmetry in complexity is well-suited for applications where the video sequence is encoded once and decoded many times, such as broadcasting or video-on-demand streaming systems. However, some recent applications [31] such as wireless video surveillance, multimedia sensor networks, wireless PC cameras, and mobile cameras phones require a low complexity encoding, while possibly affording a high complexity decoding.

DVC fits well these scenarios since it enables the exploitation of the similarities among successive frames at the decoder side, making the encoder less complex. Thus, the task of motion estimation and compensation is shifted from the encoder to the decoder.

Based on Slepian-Wolf and WZ theories, practical implementations of DVC have been proposed in PRISM scheme [16, 32] for multimedia transmission on wireless networks using syndrome coding. In [33], Aaron *et al.* reported the first results on a WZ coding scheme for motion video that is useful for systems that require simple encoders but can handle more complex decoders. In this scheme, the encoder performs scalar quantization and turbo encoding, whereas the decoder aims at executing turbo decoding using an interpolated version of the original WZF. Afterwards, Aaron *et al.* [34] proposed an architecture similar to [33], but with the major difference of introducing transform coding at the encoder. This new scheme leads to a better coding efficiency compared to [33]. In [35], hash information is extracted from the WZF being encoded and sent to help the decoder in the motion estimation task. In [36], a modified three dimensional recursive search block matching is proposed to increase the accuracy of the SI. The European project DISCOVER [4, 5] came up with one of the most efficient and popular existing architectures, which is based on

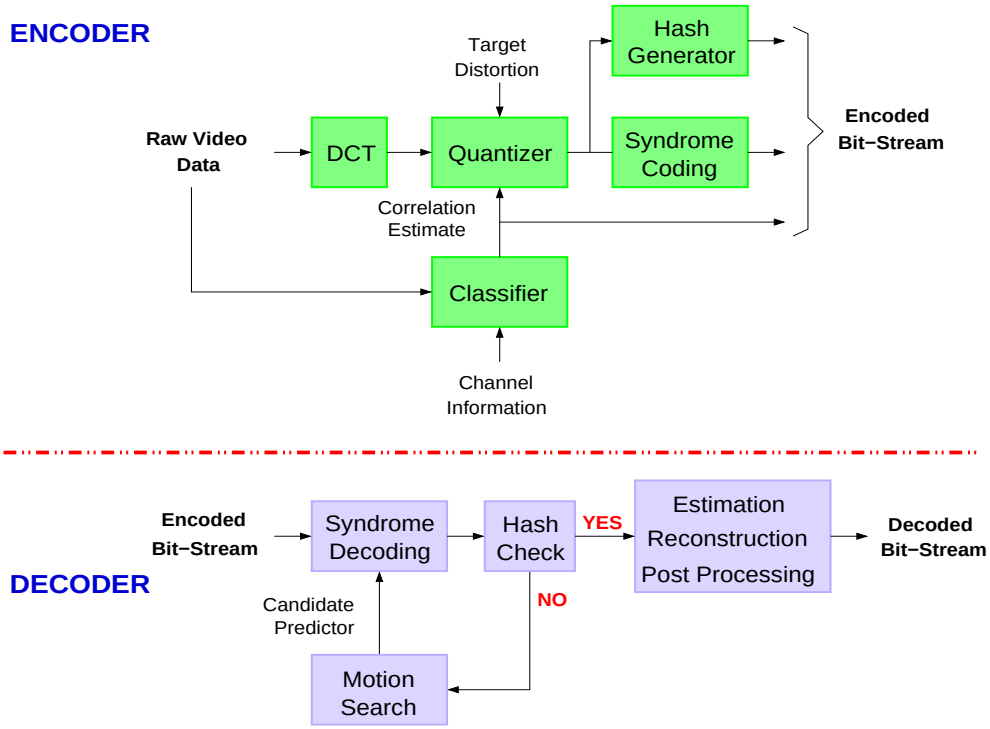

Figure 2.12: PRISM scheme.

Stanford DVC scheme [6]. More recently, VISNET II Project developed a transform domain WZ coding scheme [9] based on DICOVER codec, by adding several new tools. Moreover, Chien *et al.* [37] proposed a DVC system with an RD-based adaptive quantization scheme. In this system, the RD estimation is performed at the decoder without adding complexity to the encoder.

In this section, PRISM codec, DISCOVER codec, VISNET II codec and motion compensated temporal interpolation technique are described.

2.5.1 PRISM Architecture

DISCUS [38, 39] is one of the first efficient code designs investigated in order to realize WZ coding. The authors in [40] present a theoretic study of a WZ coding scheme with SI at the decoder. Furthermore, they prove that a WZ system can be developed with a minor loss in rate-distortion performance, with respect to conventional predictive video coding. Based on these works, PRISM (Power-efficient, Robust, hIgh compression, Syndrome-based Multimedia coding) scheme [16, 32] is created. Encoding and decoding processes of PRISM architecture are described in this section. The scheme of the PRISM codec is depicted in Fig. 2.12. In the PRISM scheme, the frames are divided into blocks of 8×8 pixels. The main modules of the PRISM encoding are listed in the following.

- **Decorrelation Transform -** The discrete cosine transform (DCT) is first applied on the 8×8 block to approximate the Karhunen-Loeve (KL) transform of the correlation

noise innovations process between the source vector and its SI counterpart. The transformed coefficients are then arranged in a 1-D vector doing a zig-zag scan on the 2-D block.

- **Quantization -** The transformed coefficients are then quantized with a given quantization step size, which is estimated according to a desired reconstruction quality.

- **Classification -** This step consists in exploiting the correlation between the source and the SI, in order to classify the bit planes of the block into three categories: skipped bit plane, WZ encoding and entropy encoding. In PRISM encoder, the reference block is estimated according to the computational capability of the encoder. Thus, two cases for SI generation are investigated. The first case aims directly at using the previous frame as SI (No motion search). This case is suitable for low power encoders. The second one consists in finding the most similar block in the previous frame to the target block by carrying out high-complexity motion search. The correlation between the source coefficient of the block X_i $(i = 1, 2, ..., 64)$ and the coefficient of the reference block in the SI Y_i $(i = 1, 2, ..., 64)$ is used to estimate the number of most significant bit planes of the quantized X_i that can be inferred from the SI Y_i. The selected most significant bit planes are not transmitted, since they will be recovered at the decoder from the SI. The most significant of the remaining least significant bit planes are WZ encoded and the least significant of them are entropy encoded, since these bit planes can not be recovered from the SI. For correlation noise, a set of 15 thresholds T_i $(i = 0, 1, ..., 14)$ and 16 classes C_i $(i = 0, 1, ..., 15)$ are defined. The index i which indicates the class of Laplacian correlation noise is determined according to the scalar mean-squared error E between X_i and Y_i. Thus, C_i is chosen if $T_{i-1} \leq E < T_i$.

- **Syndrome Encoding -** As mentioned in the classification procedure, the least significant bit planes are encoded using WZ encoding and entropy encoding. For WZ encoding, the bits are encoded using the parity check matrix of a linear error correction code. The simple Bose-Chaudhuri-Hocquenghem (BCH) [41] block codes are used since the length of the bitstream is small.

- **Hash generation -** A cyclic redundancy check (CRC) checksum is used as a "signature" of the quantized codeword sequence. The CRC aims at finding the best predictor at the decoder. Note that the CRC needs to be sufficiently strong so as to act as a reliable signature for the codeword sequence. For this reason, 16 bit-checksum is used.

The block diagram of the PRISM decoder is depicted in Fig. 2.12. The encoder sends the motion vectors when the decoder does not have to do the motion search. The main decoder modules are described.

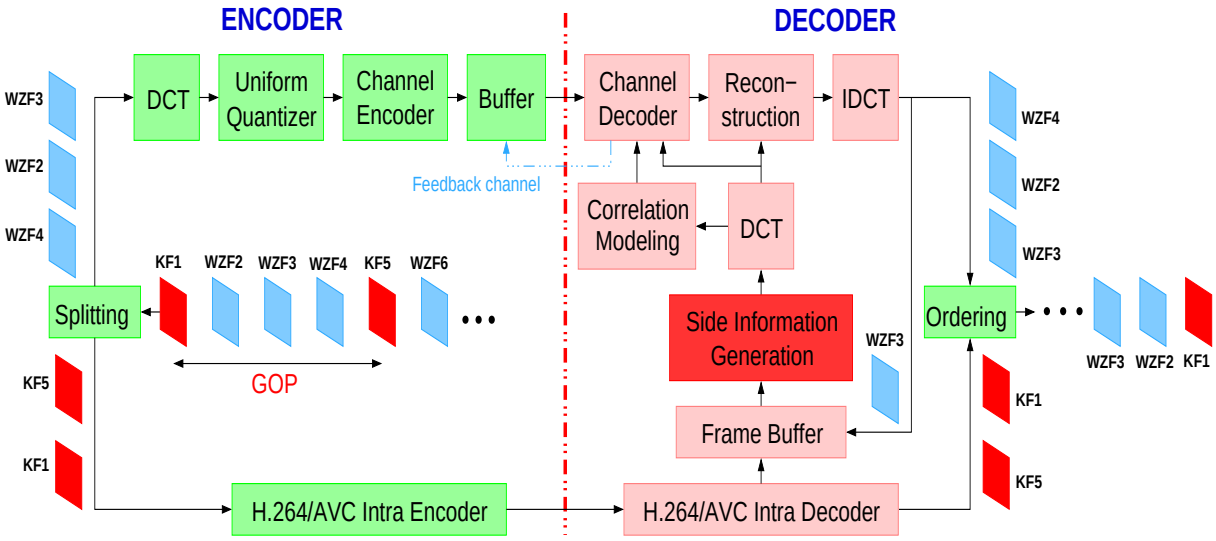

Figure 2.13: Stanford architecture used in this manuscript (example for GOP = 4).

- **Generation of Side Information (Motion Search) -** The decoder generates a set of candidates by performing a motion search. These candidates are tried one by one to decode the received syndrome. The one that fits the hash check module is selected as the best predictor.

- **Syndrome decoding -** There are two steps in the syndrome decoding. The first step consists in decoding the entropy coded bit planes. The second one aims at finding the closest codeword to the SI within the specified coset.

- **Hash Check -** If the checksum of the decoded block matches the transmitted hash, the decoding is declared to be successful. Otherwise, a next candidate predictor is obtained using the motion search module and the decoding procedure is repeated. When the best predictor is detected, the syndrome decoding process recovers the base quantization intervals for the coefficients that are syndrome encoded.

- **Estimation and Reconstruction -** The recovered quantized codeword sequence is used along with the best predictor to obtain the best reconstruction of the source. The best mean squared estimate is used from the predictor and the quantized codeword to obtain the source reconstruction.

- **Inverse Transform -** The inverse zig-zag scan operation is carried out on the reconstructed coefficients to obtain a 2-D block. Then, the inverse transform is applied to obtain the reconstructed pixels.

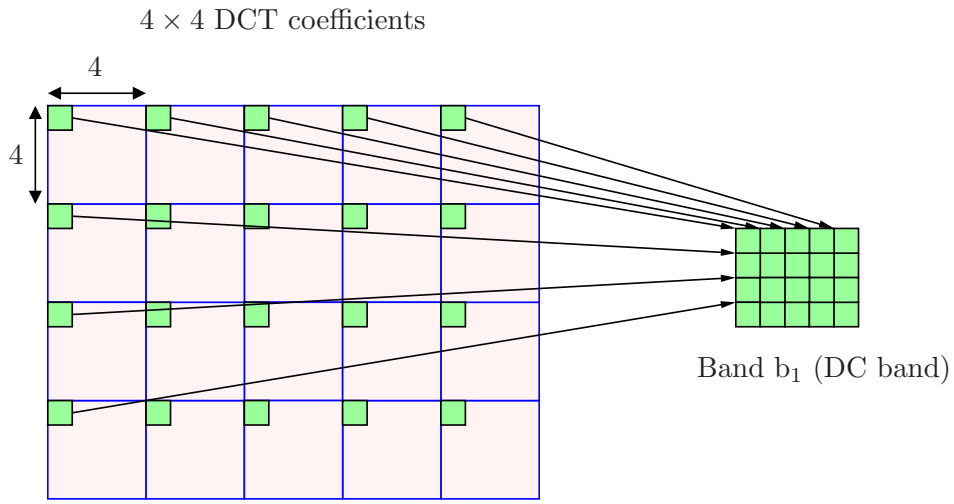

Figure 2.14: Generating the DC band (Band b_1) from the 4×4 DCT coefficients.

2.5.2 DISCOVER Architecture

In this section, we present the DISCOVER codec [4, 5] whose architecture is depicted in Fig. 2.13. The DISCOVER project came up with one of the most efficient DVC schemes, and the most widely used as a reference in this domain. It is based on the Stanford scheme [6]. More specifically, it is based on transform domain WZ coding. First, the video sequence is divided into WZ frames (WZFs) and key frames (KFs). The Group of Pictures (GOP) of size n is defined as a set of frames consisting of one KF and $n-1$ WZFs.

The KFs are directly encoded using H.264/AVC Intra. The modules of the WZF encoding procedure are detailed in the following.

- **Transform and Quantization:** First, the WZF is transformed using a 4×4 integer Discrete Cosine Transform (DCT). The integer DCT coefficients of the whole WZF are then organized into 16 bands, indicated by b_k with $k \in [1, 16]$, according to their position within the 4×4 blocks (Fig. 2.14). The low frequency information (*i.e.* the DC coefficients) are placed in the first band $k = 1$, and the others coefficients are grouped in the AC bands $k = 2, 3, ..., 16$. Next, each DCT coefficients band b_k is uniformly quantized with 2^{M_k} levels (where the number of reserved bits M_k depends on the band k). Fig. 2.15 shows the number of levels for each band, for eight different encoding rates with quantization matrices QI $= 1, 2, ..., 8$.

  The data range for the DC coefficients band is assumed to be $[0, 2^{M_{max}})$. Thus, the range of the n-th quantization interval using a uniform scalar quantizer is:

$$I_n = [n \times \Delta,\ (n+1) \times \Delta), \tag{2.11}$$

Band number

QI Values	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	16	8	8	0	0	0	0	0	0	0	0	0	0	0	0	0
2	32	8	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	32	8	8	4	4	4	0	0	0	0	0	0	0	0	0	0
4	32	16	16	8	8	8	4	4	4	4	0	0	0	0	0	0
5	32	16	16	8	8	8	4	4	4	4	4	4	4	0	0	0
6	32	16	16	8	8	8	8	8	8	8	4	4	4	4	4	0
7	64	32	32	16	16	16	8	8	8	8	4	4	4	4	4	0
8	128	64	64	32	32	32	16	16	16	16	8	8	8	4	4	0

Figure 2.15: Various 4×4 quantization matrices (one per line) corresponding to eight rate-distortion points. For each QI, the number of levels is given for the 16 bands.

In this case, Δ represents the DC quantization step and is defined as:

$$\Delta = 2^{(M_{max} - M_1)} \tag{2.12}$$

where M_1 is the number of bits reserved for each quantized value of the DC band.

A dead-zone quantizer with doubled zero interval is used for the AC coefficients. In this case, the dynamic range $[-R^k_{max}, +R^k_{max})$ is separately estimated for each band b_k with $k > 1$. The quantization step is defined as:

$$\Delta^k = \frac{2 \times R^k_{max}}{2^{M_k}} \tag{2.13}$$

where M_k is the number of bits reserved for each band b_k with $k > 1$. Furthermore, the quantization intervals are defined as:

$$\mathrm{I}^k_n = \begin{cases} [(n-1) \times \Delta^k,\ n \times \Delta^k) & \text{if } n < 0 \\ [-\Delta^k,\ \Delta^k) & \text{if } n = 0 \\ [n \times \Delta^k,\ (n+1) \times \Delta^k) & \text{if } n > 0 \end{cases} \tag{2.14}$$

The quantization indices n of each DCT band b_k are split and then organized into M_k bit planes and fed to the channel encoder.

- **Channel encoder:** The resulting quantized symbols (associated to the DCT band b_k) are split into bit planes. For a given band, the bits of the same significance are grouped together in order to form the corresponding bitplane array which is

then independently encoded using a channel encoder. The latter, also known as the Slepian-Wolf encoder, is a rate-compatible Low Density Parity-Check Accumulate (LDPCA) code or a turbo code. Each bit plane is successively fed into the channel encoder in order to compute a separate set of parity bits, while the systematic bits are discarded. The parity information is then stored in a buffer and progressively sent in chunks, upon request by the decoder, through the feed-back channel. The encoder estimates a minimum number of accumulated syndromes to be sent per bit plane and per band, in order to reduce the number of accumulated syndrome requests from the decoder. Furthermore, an 8-bit CRC sum of the encoded bit plane is also transmitted to assist the decoder in detecting residual errors.

The KFs are directly decoded using H.264/AVC Intra. The decoded KFs are used to generate a SI for the WZF being decoded. The modules of the decoding process of the WZF are described in the following.

- **Side Information Generation:** The quality of the SI has a strong impact on the performance of DVC. Fig. 2.16 shows all necessary interpolations for a GOP size 4. For example, during the interpolation of WZF $\mathbf{F}_2$, the forward and backward reference frames are KFs $\mathbf{F}_0$ and $\mathbf{F}_4$. For the interpolation of $\mathbf{F}_1$, the reference frames are the KF $\mathbf{F}_0$ and the previously decoded WZF $\mathbf{F}_2$. This hierarchical interpolation order has been shown to be optimal for a GOP of size 4 [42]. In the DISCOVER scheme, the frame interpolation framework is composed of four modules to obtain high quality SI [7] (preceded by low-pass filtering of the reference frames in order to improve the motion vectors reliability): forward motion estimation between the previous and next reference frames, bi-directional motion estimation to refine the motion vectors, spatial smoothing of motion vectors in order to achieve higher motion field spatial coherence (reduction of the number of false motion vectors), and finally bi-directional motion compensation. This technique is more detailed in Section 2.5.4.

- **Channel decoder:** A block-based 4×4 DCT is carried out over the generated SI in order to obtain the DCT coefficients which can be seen as a noisy version of the WZF DCT coefficients. In order to model the error distribution between corresponding DCT bands of SI and WZF, the DISCOVER codec uses a Laplacian distribution [43, 44]. This distribution is defined as:

$$f_Z(z) = \frac{-\alpha}{2} e^{|z|} \tag{2.15}$$

where $z = \mathrm{WZ}(x, y) - \mathrm{SI}(x, y)$, (x, y) is the current position within the WZF and α is the Laplacian distribution parameter defined as:

$$\alpha = \sqrt{\frac{2}{\sigma^2}} \tag{2.16}$$

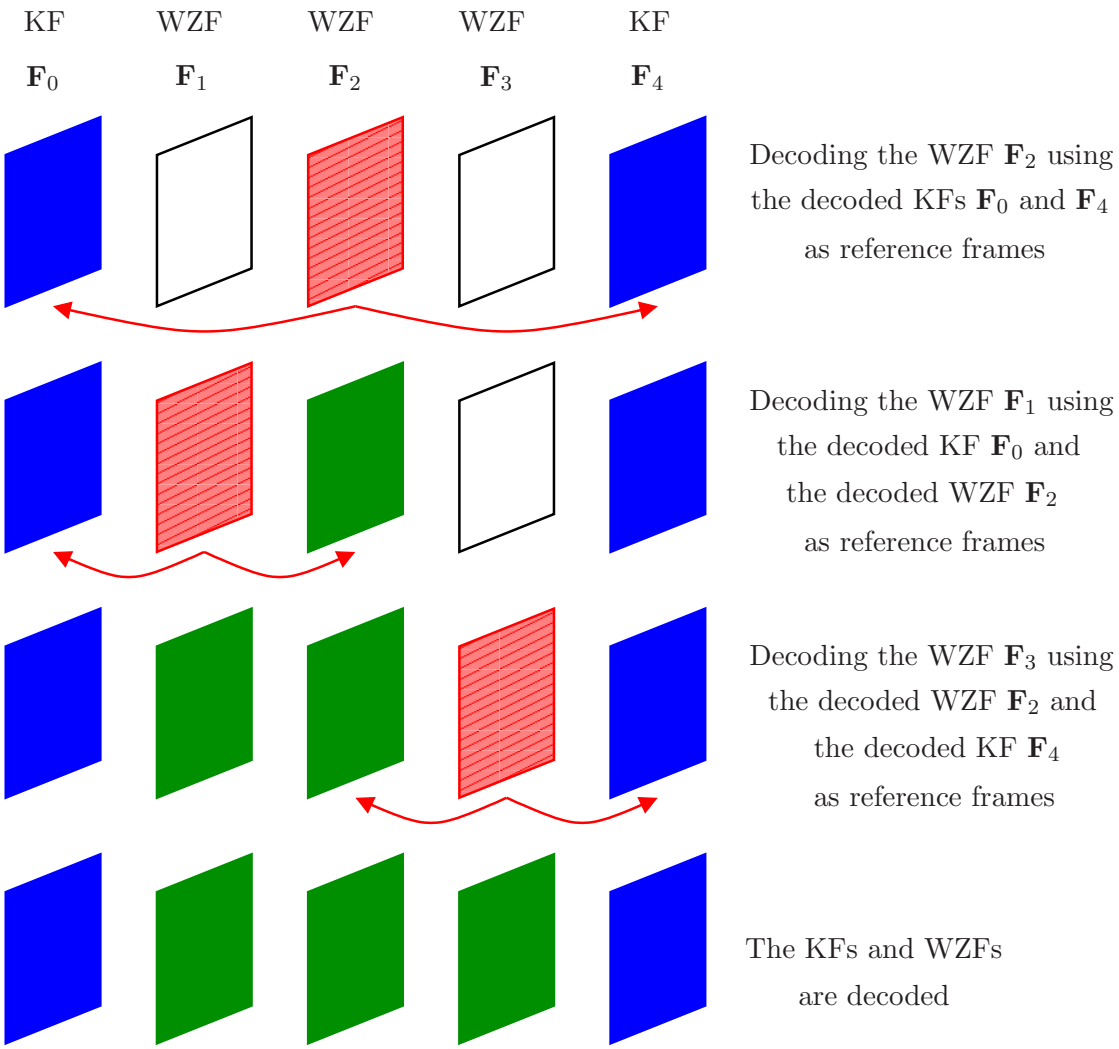

Figure 2.16: Interpolation steps for a GOP size 4.

where σ^2 is the variance of the residual between the WZF and the generated SI. α can be estimated at three levels: frame level, block level or pixel level [44].

In the DISCOVER codec, the Laplacian parameter is estimated on-line at the decoder. Since the original WZF is not available at the decoder, the residual frame R between the motion compensated reference frames is used to estimate the Laplacian parameter. This residual frame is defined as:

$$\mathrm{R}(x,y) = \frac{\mathrm{FRF}(x+v_x, y+v_y) - \mathrm{BRF}(x-v_x, y-v_y)}{2} \tag{2.17}$$

where $\mathrm{FRF}(x+v_x, y+v_y)$ and $\mathrm{BRF}(x-v_x, y-v_y)$ represent the forward and backward motion compensated frames. The α parameter is computed using the residual frame R. Once the SI DCT coefficients and the residual statistics for a given DCT band b_k are known, the Slepian-Wolf decoder corrects the bit errors using the parity bits of

WZF requested through the feedback channel.

- **Reconstruction and inverse transform:** The reconstruction is performed by using the SI DCT coefficients and the decoded DCT coefficients. Let i be the decoded quantization index and y the SI DCT coefficient. $\{z_0, z_1, ..., z_i, ..., z_{M-1}\}$ denote the M quantizer levels. The quantization step size is $\Delta = z_{i+1} - z_i$, since the quantization is uniform. The reconstruction step [45] consists in computing the expectation $\hat{x} = \mathbb{E}[X|X \in B_i, y]$, where $B_i = [z_i, z_{i+1})$ is the quantization interval corresponding to the index i. This expectation is defined as:

$$\hat{x} = \mathbb{E}[X|X \in B_i, y] = \frac{\int_{z_i}^{z_{i+1}} x f_{X|Y}(x|y)dx}{\int_{z_i}^{z_{i+1}} f_{X|Y}(x|y)dx} \tag{2.18}$$

where $f_{X|Y}(x|y)$ is the conditional probability density function (pdf) of X given Y. In DISCOVER codec, the Laplacian model is defined as:

$$f_{X|Y}(x|y) = \frac{\alpha}{2} e^{-\alpha|x-y|} \tag{2.19}$$

where α is the model parameter. Finally, $\hat{x}$ can be written as:

$$\hat{x} = \begin{cases} z_i + \frac{1}{\alpha} + \frac{\Delta}{1-e^{\alpha\Delta}} & \text{if } y < z_i \\ y + \frac{(\gamma+\frac{1}{\alpha})e^{-\alpha\gamma} - (\delta+\frac{1}{\alpha})e^{-\alpha\delta}}{2-(e^{-\alpha\gamma}+e^{-\alpha\delta})} & \text{if } y \in B_i \\ z_{i+1} - \frac{1}{\alpha} - \frac{\Delta}{1-e^{\alpha\Delta}} & \text{if } y \geq z_{i+1} \end{cases} \tag{2.20}$$

where $\gamma = y - z_i$ and $\delta = z_{i+1} - y$. $\hat{x} = \frac{z_i + z_{i+1}}{2}$ when $\alpha = 0$. On the other side, when $\alpha \to \infty$, $\hat{x}$ approaches tone of the following values:

$$\hat{x} = \begin{cases} z_i & \text{if } y < z_i \\ y & \text{if } y \in B_i \\ z_{i+1} & \text{if } y \geq z_{i+1} \end{cases} \tag{2.21}$$

Finally, the inverse 4×4 DCT transform is carried out, and the entire frame is restored in the pixel domain.

2.5.3 VISNET II Architecture

The objective of this codec is to improve the RD performance of DISCOVER. VISNET II DVC codec [9] provides the same modules as DISCOVER, but many advanced tools are added. In the following, we present the new tools of this codec compared to DISCOVER codec.

- **Iterative Reconstruction** - After decoding all DCT bands using turbo codes (as in DISCOVER codec), the decoder reconstructs a decoded WZF using the SI and

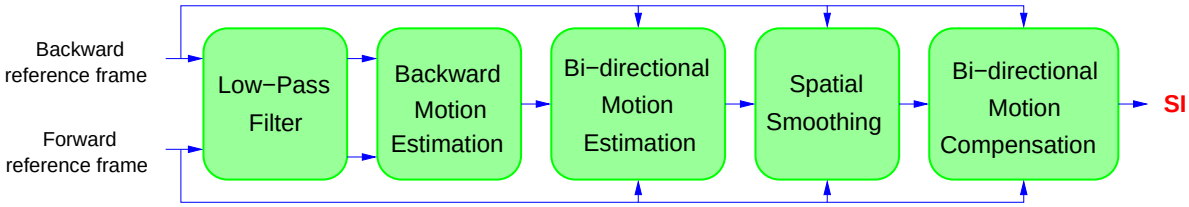

Figure 2.17: Modules of MCTI technique.

the decoded DCT coefficients. This decoded WZF has a higher quality than the SI and can therefore be exploited to generate again the SI with an improved quality [8]. This procedure is based on the refinement of the motion vectors and the reference frame mode selection (backward, forward, and bidirectional prediction are allowed). Afterwards, the reconstruction can be performed again with the improved SI and higher quality frame is obtained.

- **Deblocking Filter -** The deblocking filter [46] is used in order to improve both subjective and objective qualities of the WZFs. The filter is inserted in the SI loop, *i.e.* the frame generated by the filter is used as a reference in the SI generation process.

2.5.4 Motion Compensated Temporal Interpolation technique

The SI is commonly generated using MCTI [47]. This technique is used in DISCOVER and VISNET II codecs [4][9]. Figure 2.17 shows the modules of the MCTI technique. The frame interpolation framework is composed of four modules.

- **Low-Pass Filter** - The backward and forward reference frames are low-pass filtered in order to improve the motion vectors reliability and to reduce noise.
- **Backward Motion Estimation** - A block matching algorithm is employed in order to estimate the motion between the forward and backward reference frames. This stage provides a coarse estimation of the motion field using a block size for the matching of 16×16 pixels, within a search area (**SA**) of ± 32 pixels, in 2 pixels accuracy. Fig. 2.18 shows the backward motion estimation. In the block matching algorithm, the determination of the similarity between the target block b and the shifted block by the motion vector $\mathbf{v} \equiv (v_x, v_y) \in \mathbf{SA}$ is estimated using the Weighted Mean Absolute Difference (WMAD) criterion, as follows:

$$\mathrm{WMAD}(\mathrm{b}, \mathbf{v}) = \frac{1}{16^2} \sum_{x=x_0}^{x_0+16} \sum_{y=y_0}^{y_0+16} |\mathrm{FR}(x,y) - \mathrm{BR}(x+v_x, y+v_y)| \, (1 + \lambda \|\mathbf{v}\|) \quad (2.22)$$

where BR and FR are the filtered backward and forward reference frames, (x_0, y_0) is the up-left pixel of the block b and λ is a penalty factor which allows to penalize

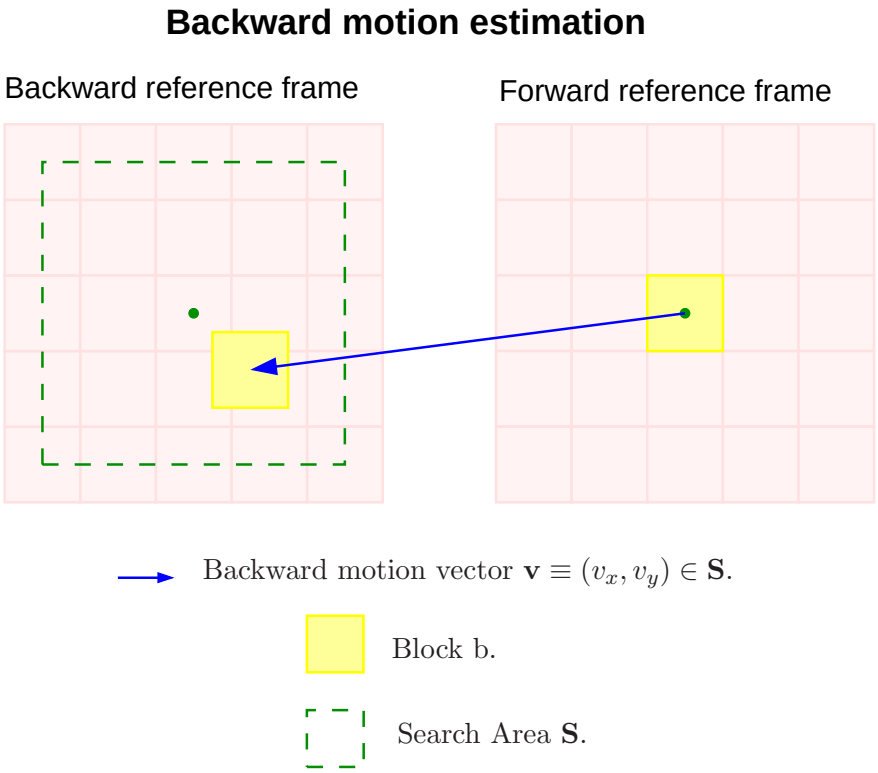

Figure 2.18: Backward motion estimation.

the MAD by the length of the motion vector $\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2}$. λ is empirically set to 0.05. This penalty term aims at avoiding large motion vectors errors. The block matching algorithm aims at selecting the best motion vector $\mathbf{V}_\mathrm{b}$ for the block b by minimizing the WMAD as follows:

$$\mathbf{V}_\mathrm{b} = \arg \min_{\mathbf{v_i} \in \mathbf{SA}} \mathrm{WMAD}(\mathrm{b}, \mathbf{v}_i) \tag{2.23}$$

- **Bidirectional Motion Estimation** - A bidirectional motion estimation algorithm is employed to refine the motion vectors obtained in the forward motion estimation step. Fig 2.19 shows the bi-directional motion estimation procedure. First, for each block b in the WZF, the distances $\mathbf{d}_i^\mathrm{b}$ between the center of the block b and the center of each motion vector $\mathbf{v}_i$ are computed. The motion vector $\mathbf{v}^\mathrm{b}$ which gives the smallest distance is selected for block b. Then, the bidirectional motion estimation technique aims at dividing the motion vector $\mathbf{v}^\mathrm{b}$ towards the backward and forward reference frames. The forward and backward motion vectors for the block b in the WZF are respectively taken as: $\mathbf{u}^\mathrm{b} = -\frac{1}{2}\mathbf{v}^\mathrm{b}$ and $\mathbf{w}^\mathrm{b} = -\mathbf{u}^\mathrm{b} = \frac{1}{2}\mathbf{v}^\mathrm{b}$.

  The bidirectional motion vectors ($\mathbf{u}^\mathrm{b}$, $-\mathbf{u}^\mathrm{b}$) are refined within a small search range. The block size used for the matching is initially set to 16×16 pixels, with an adaptive

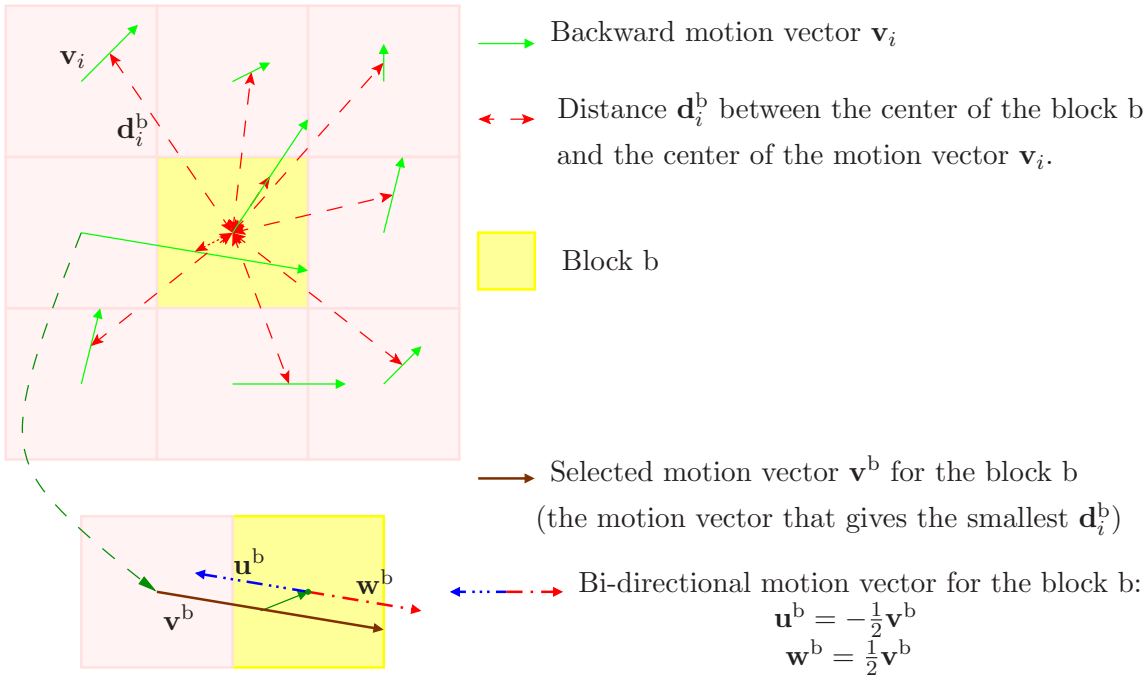

Figure 2.19: Bidirectional motion estimation procedure.

search range (**ASR**), in half-pixel accuracy. Let $\mathbf{r}^{\mathrm{b}}$ be the small motion vector that must be added to the bidirectional motion vector. $\mathbf{r}^{\mathrm{b}}$ can be obtained as:

$$\mathbf{r}^{\mathrm{b}} = \arg \min_{\mathbf{r} \in \mathbf{ASR}} \mathrm{WMAD}_{\mathrm{bid}}(\mathrm{b}, \mathbf{u}^{\mathrm{b}} + \mathbf{r}) \tag{2.24}$$

with

$$\mathrm{WMAD}_{\mathrm{bid}}(\mathrm{b}, \mathbf{r}) = \frac{(1 + \lambda \|\mathbf{r}\|)}{16^2} \sum_{x=x_0}^{x_0+16} \sum_{y=y_0}^{y_0+16} |\mathrm{BRF}(x + r_x, y + r_y) - \mathrm{FRF}(x - r_x, y - r_y| \tag{2.25}$$

Then, the block size is reduced to a finer 8×8 pixels and the 8×8 blocks inherit the bidirectional motion vectors of 16×16 blocks. Finally, the bidirectional motion vectors of 8×8 blocks are refined applying the same refinement procedure of 16×16 blocks (*i.e.* an adaptive search range is used, in half-pixel accuracy). Note that the final bidirectional motion vector (after the refinement) of the block b (8×8 pixels) is referred to as $\mathbf{h}^{\mathrm{b}}$.

- **Spatial Motion Smoothing** - In order to achieve higher motion field coherence, spatial smoothing algorithms are used to reduce the number of false motion vectors. For each block b, the spatial motion smoothing algorithm takes into account the neighboring bidirectional motion vectors as candidates for the block b (Fig 2.20). The weighted median vector field [48] is used to select the bidirectional motion

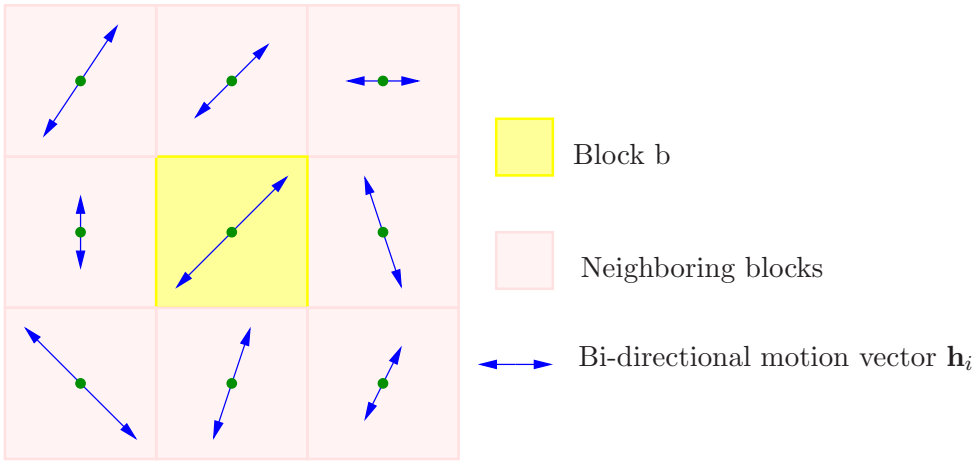

Figure 2.20: Neighboring bidirectional motion vectors of the block h.

vector from the candidate bidirectional motion vectors as follows:

$$\mathbf{s}^{\mathrm{b}} = \arg \min_{k=1,2,\ldots,\mathrm{N}} \left(\sum_{i=1}^{\mathrm{N}} a_i ||\mathbf{h}_k - \mathbf{h}_i|| \right), \tag{2.26}$$

with

$$a_i = \frac{1}{8^2} \sum_{x=x_0}^{x_0+8} \sum_{y=y_0}^{y_0+8} |\mathrm{BRF}(x + h_{ix}, y + h_{iy}) - \mathrm{FRF}(x - h_{ix}, y - h_{iy})| \tag{2.27}$$

where N = 9 is the number of neighboring blocks.

- **Bidirectional Motion Compensation** - Once the final bidirectional motion vectors $\mathbf{s}^{\mathrm{b}}$ are estimated, the SI can be interpolated using bidirectional motion compensation as follows:

$$\begin{cases} \text{if } \mathbf{p} + \mathbf{s}^{\mathrm{b}} \notin \mathbf{SP} \\ \quad \mathrm{SI}(\mathbf{p}) = \mathrm{FRF}(\mathbf{p} - \mathbf{s}^{\mathrm{b}}) \\ \text{otherwise} \\ \quad \text{if } \mathbf{p} - \mathbf{s}^{\mathrm{b}} \notin \mathbf{SP} \\ \quad\quad \mathrm{SI}(\mathbf{p}) = \mathrm{BRF}(\mathbf{p} + \mathbf{s}^{\mathrm{b}}) \\ \quad \text{otherwise} \\ \quad\quad \mathrm{SI}(\mathbf{p}) = \frac{1}{2}(\mathrm{BRF}(\mathbf{p} + \mathbf{s}^{\mathrm{b}}) + \mathrm{FRF}(\mathbf{p} - \mathbf{s}^{\mathrm{b}})) \end{cases}$$

where $\mathbf{s}^b$ and $-\mathbf{s}^b$ are the bidirectional motion vectors associated with the position $\mathbf{p} = (x, y)$ toward the BRF and FRF respectively, and **SP** is defined as:

$$\mathbf{SP} = \{(m, n) : 0 < m < (M - 1) \text{ and } 0 < n < (N - 1)\} \tag{2.28}$$

where M and N are the dimensions of the frame.

2.6 Conclusion

Distributed video coding aims at giving the decoder the task of exploiting the correlation among successive frames. The major difference with respect to predictive video coding is shifting the computational load from the encoder to the decoder. In DVC, the interpolation of the original Wyner-Ziv frame at the decoder can significantly impact the rate-distortion performance.

In this chapter, we showed the main modules used in the predictive video coding standards such as H.264/AVC. Then, the main architectures of distributed video coding (PRISM, DISCOVER and VISNET II) were presented. We also explained the main steps of the motion-compensated temporal interpolation technique to generate the side information. In the upcoming chapter, we will focus on refinement techniques that aim at improving the quality of SI generation in DVC.

Chapter 3

Successive Refinement of Side Information Generation

Contents

In Distributed Video Coding (DVC), the Side Information (SI) is commonly generated by Motion-Compensated Temporal Interpolation (MCTI) of the neighboring reference frames at the decoder side. The SI quality has a strong impact on the final Rate-Distortion (RD) performance of the codec. Indeed, the channel decoder corrects the errors in the SI using the parity bits sent by the encoder via the feedback channel. Furthermore, the SI is used to generate the correlation noise model necessary for the decoding process. A more accurate approximation of the true distribution of the correlation noise (*i.e.*, the estimated SI becomes closer to the original WZF) reduces the necessary amount of parity information requested by the decoder through the feedback channel. Moreover, the reconstruction module in the decoder uses the SI coefficients along with the decoded WZF coefficients to obtain the decoded WZF. Therefore, a more accurate estimation of the original WZF can

also enhance the final quality of the decoded WZF. For all these reasons, a great importance has been given to the problem of SI estimation, which has led to many works that aim at improving the accuracy of the SI.

In this chapter, we present a new approach that consists of a successive refinement of the SI, after the decoding of each DCT band. This approach allows to improve the accuracy of motion compensation between reference frames and progressively generate a new SI. This new SI is closer to the original Wyner-Ziv frame (WZF). In other words, we aim to successively exploit the available information of the WZF during the decoding process, in order to improve the accuracy of the SI. Then, the enhanced SI is used to decode the remaining information. Consequently, the number of requested bits for decoding the remaining DCT bands can be reduced. The proposed SI improvement technique consists of three steps that are applied after decoding each DCT band. First, we detect the suspicious motion vectors using the available information at the decoder. Second, we re-estimate the motion vectors for those suspicious blocks. Third, we compute a new SI based on three possible modes: Backward, Forward and Bidirectional mode. At this point, we propose two different algorithms that can be implemented in the block matching phase of the refinement process.

First, in Section 3.1, we present the existing methods used for improving the estimation of the SI. Second, in Section 3.2, we describe the proposed technique for successive refinement of the SI after each decoded DCT band and the correlation noise model. Third, experimental results are shown in Section 3.3. Finally, conclusions are drawn in Section 3.4.

3.1 State of the art

DVC has not reached the performance level of classical inter frame coding yet. This is in part due to the quality of the SI, which has a strong impact on the final RD performance. Several works have been proposed in order to enhance the SI. An approach proposed by Aaron et al. [35] consists in sending a hash of the original WZF being decoded in order to enhance the interpolation of the SI at the receiver. The hash code for a frame block consists of a small subset of the quantized DCT coefficients. For each block in the current WZF, the distance between the block's hash towards that of the corresponding block in the previous frame is estimated at the decoder. If the measured distance is greater than a threshold, the hash of the original WZ block is requested from the transmitter. In this system, only the previous decoded frame is used to generate the SI. Therefore, it is referred to as a low-delay system.

In [49], the authors proposed a bidirectional hash motion estimation framework by including a coding of DCT hash with zero-motion, a combination of trajectory-based motion interpolation with hash-based motion estimation, and an adaptive selection of the DCT bands that are sent to the decoder to guide the estimation of the SI.

In [50, 51], solutions are proposed for SI enhancement by merging several SI at the

decoder using a genetic algorithm. The fusion is done based on hash information adaptively transmitted from the encoder.

However, all these techniques demand that some additional information (the hash) be sent through the channel. Other techniques for SI improvement exist that can avoid this overhead. They are based on the successive refinement of the SI. A solution proposed by J. Ascenso et al. [52] for pixel domain DVC uses a motion compensated refinement of the SI successively after each decoded bit plane, in order to achieve a better reconstruction of the decoded WZF. In [53], the authors propose a novel DVC successive refinement approach to improve the motion compensation accuracy and the SI. More specifically, the technique consists in encoding each frame successively by WZ coding of multiple layers generated using the N-Queen sub-sampling pattern. Then, in the receiver, the reconstruction of each layer is performed to refine the motion vectors as well as the SI during the decoding process.

The authors in [54] propose an iterative motion compensated interpolation technique for pixel-domain DVC. The turbo decoder is executed several times for decoding the current WZF and a refined SI is generated each time. For each aligned block in the partially decoded picture, the most similar block is searched for in a number of sources (the past frame, the future frame, the motion-compensated average of the past and the future frame, and the generated SI using MCTI technique). The gain of this technique compared to the existing DVC codec is 0.15 dB in RD performance, for the first 100 frames of the Foreman sequence, with QCIF resolution at 30 Hz.

In [55], Adikari *et al.* propose a bitplane level SI refinement solution using luminance and chrominance information in pixel-domain DVC. This solution consists in refining the SI after each decoded bitplane. In [56], Weerakkody *et al.* propose a spatial-temporal refinement algorithm for pixel-domain DVC. This approach consists in extending [55] to iteratively improve the initial SI obtained by motion extrapolation. This comprises interleaving the initial SI for error estimation and flagging, followed by de-interleaving and filling of the flagged bits with an alternate iterative use of spatial and temporal prediction techniques. The authors in [57] propose a technique for unsupervised learning of forward motion vectors during the decoding of a frame with reference to its previous reconstructed frame, based on the Expectation Maximization algorithm.

In [10, 58], solutions are proposed for transform-domain DVC based on the successive refinement of the SI after each decoded DCT band. The method in [58] uses only the already available SI frame as a reference frame to further refine the SI for decoding the next band. More specifically, the SI refinement consists in three modules. First, the current reconstructed frame is used to define which blocks are worthwhile to be selected in the SI for refinement. Second, the initial SI is used as a reference frame to find the SI candidates to the selected blocks within a given window. Then, the candidate blocks are used to refine the SI for decoding the next DCT band.

The authors in [8] propose a solution for transform-domain DVC, which refines the SI

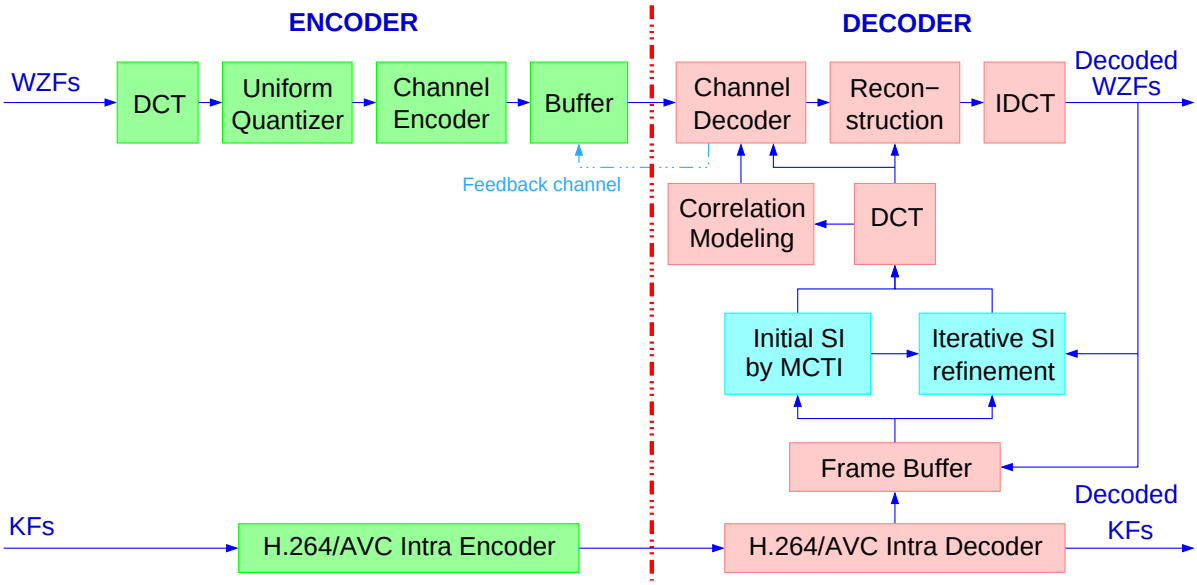

Figure 3.1: Proposed SI refinement procedure.

after the decoding of all DCT bands, in order to improve reconstruction. In VISNET II codec [9], the refinement process of the SI is also carried out after decoding all DCT bands, and a deblocking filter is used to improve the decoded WZFs. High-order motion interpolation has been proposed [59] in order to cope with object motion with non-zero acceleration. This approach consists in using more than two reference frames to interpolate the current WZF.

3.2 Proposed method for SI refinement

The block diagram of our proposed codec architecture is depicted in Figure 3.1. It is based on the DISCOVER codec [4, 5]. The Initial SI (INSI) is first computed by MCTI with spatial motion smoothing exactly as in DISCOVER codec. The LDPC parity bits of the first band (DC band) are then used by the turbo-decoder to correct the corresponding DCT coefficients in INSI; the obtained decoded frame is denoted as Partially Decoded WZF (PDWZF). Here, the two adjacent reference frames and the PDWZF are used in order to improve the SI interpolation using two different approaches, which will be detailed afterwards. Then, the obtained decoded frame after the first improvement of the INSI is used as a new PDWZF in order to improve the SI for decoding the next DCT band, and so on after each decoded DCT band.

Let SI_1 be the INSI generated for decoding the first DCT band b_1 and $PDWZF_1$ the PDWZF obtained after decoding the band b_1. The $PDWZF_1$ is used to detect the suspicious motion vectors in SI_1. Then, the $PDWZF_1$ is used along with the backward and forward reference frames to re-estimate the suspicious motion vectors. Finally, a refined SI_2 is generated using the obtained motion vectors and used to decode the second DCT

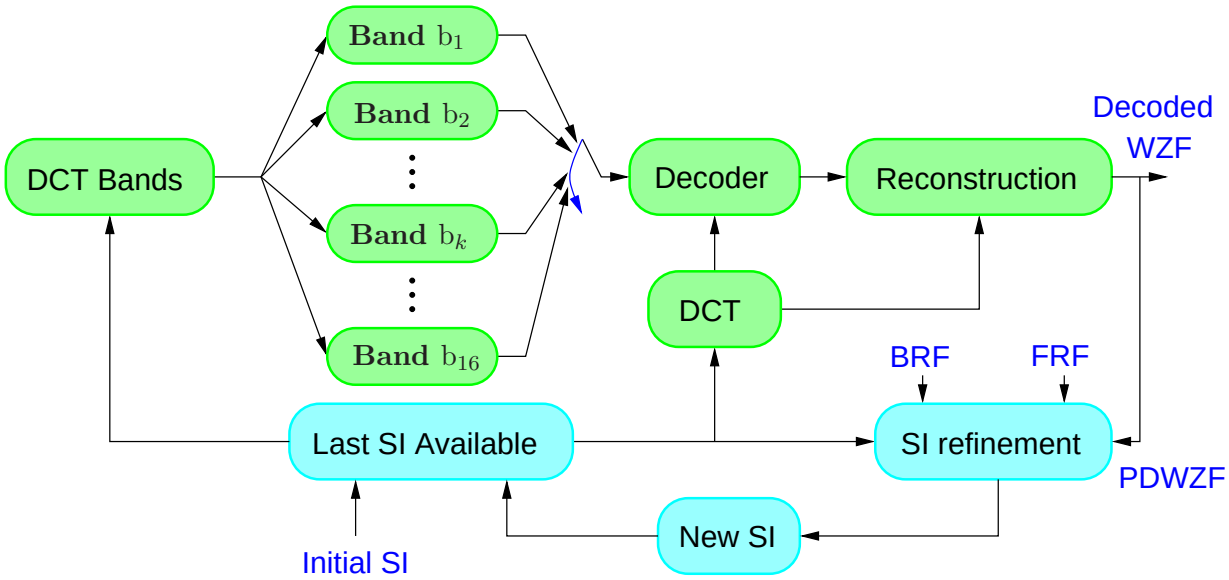

Figure 3.2: Proposed technique for successive refinement of SI and WZF decoding.

band b_2, and so on until decoding all DCT bands. The proposed scheme for this procedure is illustrated in Fig. 3.2.

For each WZF, NB bands are successively decoded (NB represents the total number of encoded DCT bands, which depends on the value of Quantization Index (QI)). Let R_i be the necessary rate for decoding the band b_i. In this case, the total rate R is defined as follows:

$$R = \sum_{i=1}^{NB} R_i, \text{ with } R_i \geq H(X_{b_i}|Y^i_{b_i}) \tag{3.1}$$

where $Y^i_{b_i}$ and X_{b_i} represent the band b_i of the SI_i and of the original WZF respectively. However, the INSI is not changed during the decoding process in DISCOVER codec. In this case, the total rate R_{DIS} can be estimated as follows:

$$R_{DIS} = \sum_{i=1}^{NB} R_i^{DIS}, \text{ with } R_i^{DIS} \geq H(X_{b_i}|Y^1_{b_i}) \tag{3.2}$$

where Y^1 represents the INSI SI_1. In the proposed method, we aim at improving the SI after decoding each DCT band. Therefore, the rate for decoding the band b_i is reduced compared to DISCOVER codec $R_i \leq R_i^{DIS}$, since $H(X_{b_i}|Y^i_{b_i}) \leq H(X_{b_i}|Y^1_{b_i})$.

At the same time, the quality of the decoded WZF will be improved during the reconstruction, since the quality of the final SI is significantly enhanced compared to the INSI. It is important to note that the reconstruction is based on the decoded DCT coefficients and the DCT coefficients of the SI. Therefore, an improved SI can reduce the necessary rate for decoding the WZF and enhance the quality of the final decoded WZF.

The proposed scheme for SI enhancement after decoding a given band is illustrated

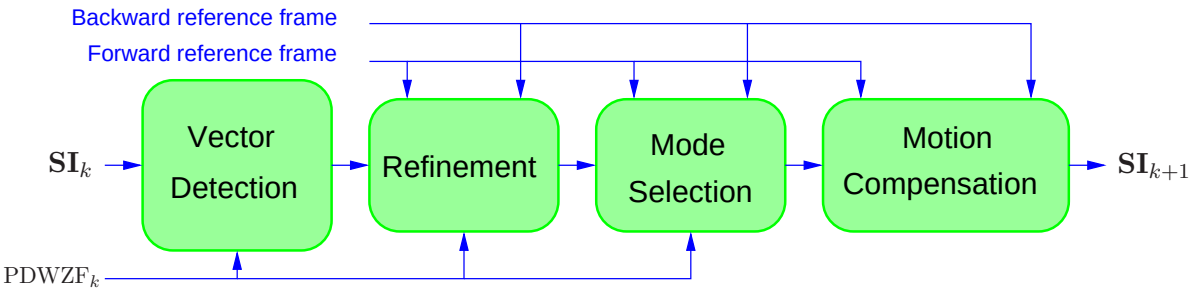

Figure 3.3: Proposed SI refinement procedure.

in Fig. 3.3. It consists of three steps that use the PDWZF, along with the forward and backward reference frames, to improve the SI: suspicious vector detection based on a matching criterion, motion vector refinement and smoothing, mode selection, and motion compensation. In this chapter, we propose two algorithms to refine the SI, denoted as Algorithms I and II. Algorithm I is similar to the method in [8]. However, it has been improved in such a way that the SI is progressively refined after the decoding of each DCT band. Moreover, both the matching criterion and the mode selection have been modified, resulting in improved performances. In Algorithm II, a different algorithm is applied in the motion vector refinement module.

Let **MVB** and **MVF** be the backward and forward motion vector for a block b respectively. These motion vectors are used to generate the INSI using the backward and forward reference frames. $\mathbf{MVB} = -\mathbf{MVF}$, since they are determined by the MCTI technique [7]. The size of the block in MCTI is 8×8 pixels. The proposed method consists in refining the **MVB** and **MVF** independently after decoding each DCT band. In this section, the proposed method is described in details and the difference between the two algorithms is shown in each module.

3.2.1 Vector Detection

In order to exploit the spatial-temporal correlations to enhance the estimated motion vectors, the proposed method uses a matching criterion which is based on the Mean Absolute Difference (MAD). The MAD between the frames F_1 and F_2, for a block b, is defined as:

$$\mathrm{MAD}(\mathbf{P_0}, \mathrm{F}_1, \mathrm{F}_2) = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |\mathrm{F}_1(i + x_0, j + y_0) - \mathrm{F}_2(i + x_0, j + y_0)| \qquad (3.3)$$

where $\mathbf{P_0} = (x_0, y_0)$ is the coordinate of the top-left pixel for the current block b, which has M rows and N columns.

The motion vectors estimated by MCTI for sequences with low motion are close to the true motion. However, false motion vectors may occur in sequences with high motion and occlusions. In order to identify suspicious vectors, the MAD is calculated between the

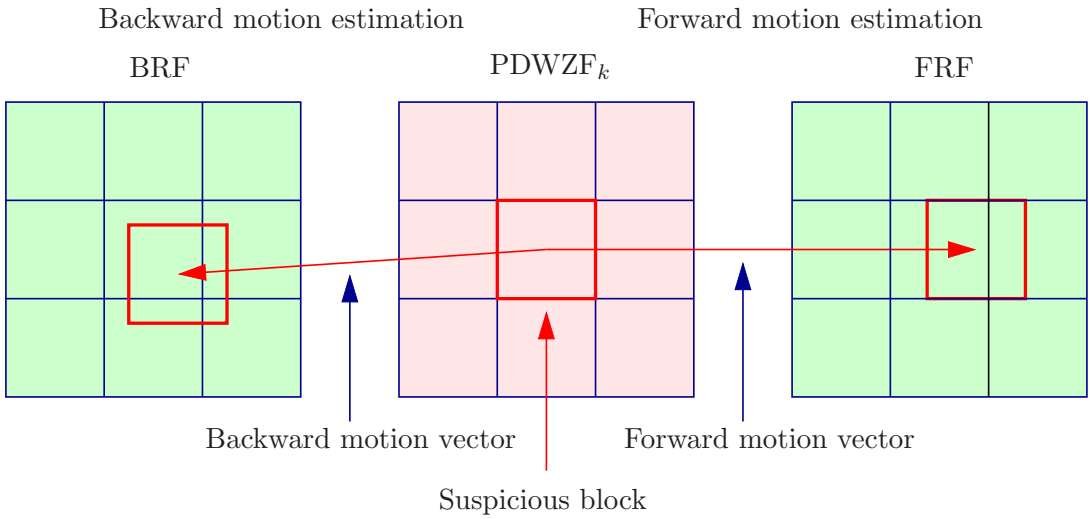

Figure 3.4: Estimation of the backward and forward motion vectors using the PDWZF and the reference frames for the suspicious blocks.

PDWZF$_k$ and the SI$_k$ (the last refinement of the SI) and compared to a threshold T_1:

$$\mathrm{MAD}(\mathbf{P_0}, \mathrm{PDWZF}_k, \mathrm{SI}_k) < T_1, \tag{3.4}$$

where $\mathbf{P_0}$ is the top left point for the processed block b. Here, we present the two algorithms based on the MAD of the block b.

- **Algorithm I** - If the condition defined in Eq. (3.4) is satisfied, the estimation is considered to be true (*e.g.*, the motion vectors **MVB** and **MVF** for this block are not modified). Otherwise, the vectors are identified as suspicious vectors and will be further refined.

- **Algorithm II** - If Eq. (3.4) is satisfied for the block b, the motion vectors **MVB** and **MVF** are refined independently within a small search area of ±2 pixels. This refinement is only applied two times during the decoding of DCT bands, the first time being after the decoding of the first ban d, and the second one after the decoding of all DCT bands. This step consists in relaxing the symmetric bidirectional motion vectors constrained in MCTI and allows a small refinement of the estimated motion vectors. Otherwise, the vectors are considered to be suspicious and will be further refined.

3.2.2 Motion vector refinement

In order to refine the motion vectors that are identified as suspicious vectors, the PDWZF$_k$ and the reference frames are used to re-estimate the motion vectors for those selected blocks. For the current block in the PDWZF$_k$, the block motion estimation determines,

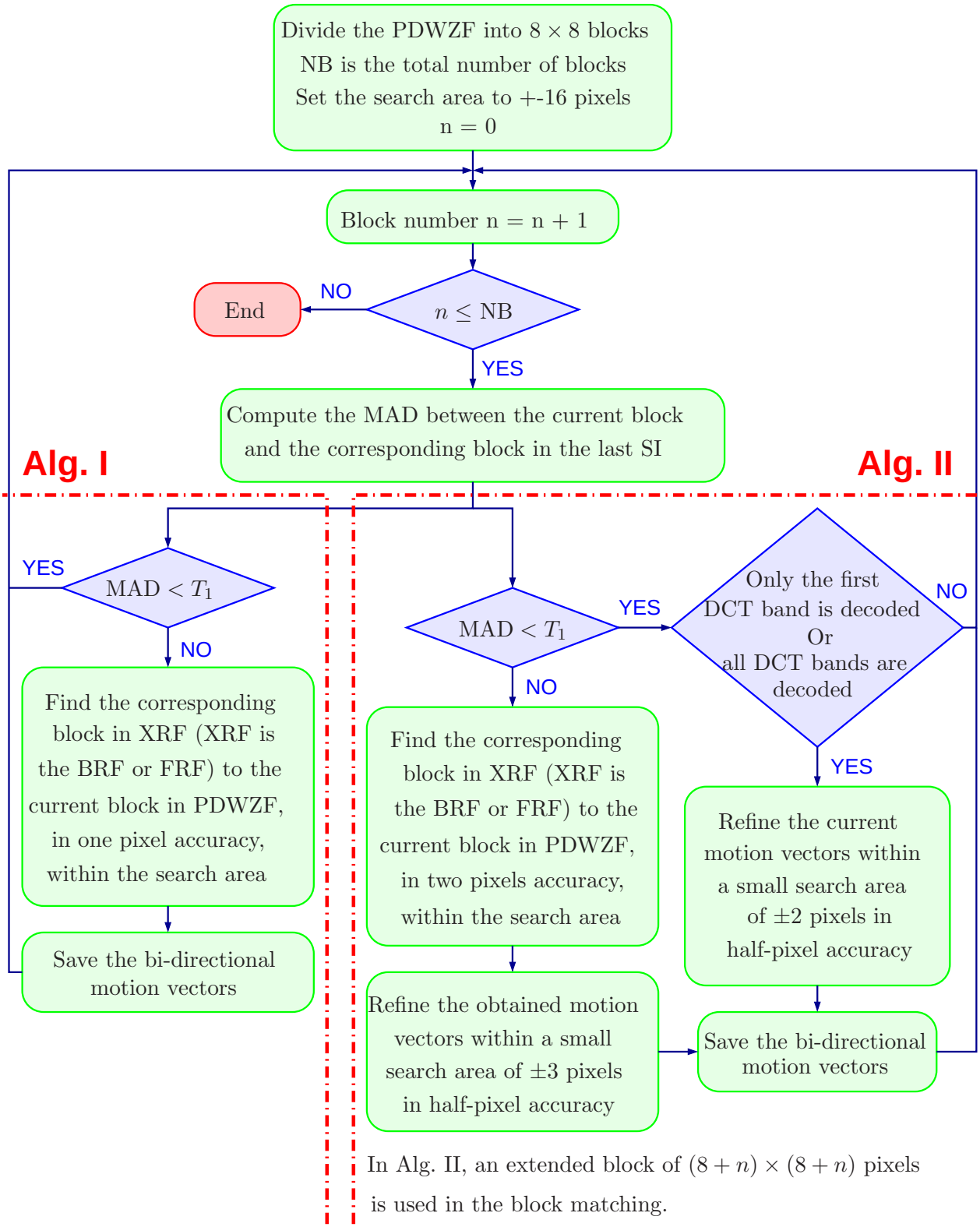

Figure 3.5: Proposed algorithms

among all candidate blocks within a search area ($\mathbf{S}$) in the BRF (or FRF), the most similar one to the current block (see Fig. 3.4). The backward and forward motion vectors, for the block b in the PDWZF_k, are obtained as follows:

$$\mathbf{MVX} = \arg \min_{\mathbf{MV} \in \mathbf{S}} \mathrm{MAD}(\mathbf{P_0}, \mathrm{PDWZF}_k, \mathrm{XRF}, \mathbf{MV}) \tag{3.5}$$

with

$$\begin{aligned} &\mathrm{MAD}(\mathbf{P_0}, \mathrm{PDWZF}_k, \mathrm{RXF}, \mathbf{MV}) = \\ &\tfrac{1}{M \times N} \textstyle\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |\mathrm{PDWZF}_k(i + x_0, j + y_0) - \mathrm{XRF}(i + x_0 + \mathrm{MV}_x, j + y_0 + \mathrm{MV}_y)| \end{aligned} \tag{3.6}$$

where $\mathbf{MV} = (\mathrm{MV}_x, \mathrm{MV}_y)$ represents the candidate motion vector and XRF represents the BRF or the FRF. $\mathbf{MV}$ is the backward motion vector if XRF is the BRF and the forward motion vector if XRF is the FRF. In this module, two different algorithms are carried out in order to re-estimate the motion vectors that are identified as suspicious vectors. As for the motion vectors that are identified as being true, they are only refined in the second algorithm. These proposed algorithms are presented below and illustrated in Fig. 3.5.

Algorithm I : This algorithm searches for the most corresponding block in the XRF (XRF is the backward or forward reference frame) within a search area of ± 16 pixels in one pixel accuracy. The obtained motion vectors are considered to be the bi-directional motion vectors of the current block in the PDWZF_k.

Algorithm II : Even though the block size is 8×8 pixels, an extended block of $(8+n) \times (8+n)$ pixels is considered in the block matching step of this refinement algorithm. First, the motion vectors between the current block in PDWZF_k and XRF(XRF is the backward or forward reference frame) are re-estimated within a search area of ± 16 pixels in two pixels accuracy. The obtained motion vectors are then refined within a search area of ± 3 pixels in half-pixel accuracy. As for the vectors that were identified as true motion vectors, they are only refined two times in this algorithm, within a search area of ± 2 pixels in half-pixel accuracy, once after decoding the first DCT band, and another time after decoding all DCT bands.

It can be verified that, in terms of the computational load, the two algorithms almost have the same complexity in finding the corresponding blocks.

3.2.3 Motion compensation mode selection

The objective of this step is to generate a motion-compensated estimate by selecting the most similar block to the current block from three sources: the BRF (BACKWARD MODE), the FRF (FORWARD MODE), and the bi-directional motion-compensated average of the backward and forward reference frames (BIMODE). The decision among these

modes is taken according to the following equations:

$$\begin{cases} \text{if } |\text{MAD}_f - \text{MAD}_b| < T_2 \\ \quad \text{MODE=BIMODE} \\ \text{otherwise} \\ \quad \text{if MAD}_f < \text{MAD}_b \\ \quad\quad \text{MODE=FORWARD MODE} \\ \quad \text{otherwise} \\ \quad\quad \text{MODE=BACKWARD MODE} \end{cases} \tag{3.7}$$

where T_2 is a threshold, MAD_b and MAD_f are the estimated mean absolute differences between the current block (in PDWZF_k) and the corresponding blocks (e.g. the blocks that minimize the MAD) in the backward and forward reference frames respectively.

The refinement SI obtained after decoding the band b_k is used at the WZ decoder as a new SI for the decoding of the next band b_{k+1}, and so forth for all bands of the WZF being decoded. Then, after decoding all bands, a new SI is generated to perform the reconstruction step and obtain the final WZF.

3.2.4 Correlation noise model

The channel decoder (Turbo code or LDPCA code) uses the noise distribution model between the original WZF and the SI in the decoding process. The correlation noise model can be estimated offline or online. The offline estimation consists in using the original WZF and the SI to determine the model parameters. Consequently, the offline correlation noise model can provide the upper performance that can be achieved by the decoder. However, it is impractical, since it requires that either the original WZF be available at the decoder or the SI be available at the encoder. Online correlation noise model, on the other hand, aims at estimating the distribution parameter only using the available information at the decoder. The commonly used methods in this context [43, 44] use the motion compensated residual between the reference frames to model the correlation noise.

At the beginning, the estimated motion vectors by the MCTI technique are used to generate the residual frame [44] as follows:

$$\text{R}^1(x,y) = \frac{\text{FRF}(x + \mathbf{MVF}^1_x, y + \mathbf{MVF}^1_y) - \text{BRF}(x + \mathbf{MVB}^1_x, y + \mathbf{MVB}^1_y)}{2} \tag{3.8}$$

where $\mathbf{MVB}^1 = (\mathbf{MVB}^1_x, \mathbf{MVB}^1_y)$ and $\mathbf{MVF}^1 = (\mathbf{MVF}^1_x, \mathbf{MVF}^1_y)$ represent the backward and forward motion fields generated respectively by MCTI. The two motion fields $\mathbf{MVB}^1$ and $\mathbf{MVF}^1$ are symmetrical (*i.e.* $\mathbf{MVB}^1 = -\mathbf{MVF}^1$). This symmetry of the motion field can disturb the estimation of the SI for sequences containing irregular motion. Therefore, we improve successively the accuracy of the correlation noise model after each decoded DCT band: First, the residual frame R^1 is used for estimating the correlation noise

Table 3.1: Rate-distortion performance gain of Alg. I for *Stefan* and *Foreman* sequences, compared to the DISCOVER codec, for different values of T_2 ($T_1 = 4$), using Bjontegaard metric.

Alg. I < T1 = 4>						
	$T_2 = 0$	$T_2 = 2$	$T_2 = 5$	$T_2 = 8$	$T_2 = 10$	$T_2 = 12$
GOP = 2						
Stefan						
Δ_R (%)	0.51	-4.92	-7.14	-7.23	-7.21	-7.25
Δ_{PSNR} [dB]	-0.04	0.29	0.42	0.44	0.43	0.43
Foreman						
Δ_R (%)	-9.08	-11.79	-12.29	-12.33	-12.10	-12.14
Δ_{PSNR} [dB]	0.51	0.67	0.69	0.69	0.69	0.69

Table 3.2: Rate-distortion performance gain of Alg. II for *Stefan* and *Foreman* sequences, compared to the DISCOVER codec, for different values of T_2 ($T_1 = 4$ and $n = 4$), using Bjontegaard metric.

Alg. II < T1 = 4 and n = 4>						
	$T_2 = 0$	$T_2 = 2$	$T_2 = 5$	$T_2 = 8$	$T_2 = 10$	$T_2 = 12$
GOP = 2						
Stefan						
Δ_R (%)	-8.29	-13.53	-15.05	-15.02	-14.95	-14.79
Δ_{PSNR} [dB]	0.49	0.83	0.93	0.93	0.93	0.92
Foreman						
Δ_R (%)	-15.37	-17.73	-18.17	-18.16	-18.16	-18.03
Δ_{PSNR} [dB]	0.89	1.04	1.06	1.06	1.06	1.06

model to serve in the decoding of the first DCT band. Then, a PDWZF is reconstructed to improve the accuracy of the SI by enhancing the motion fields $\mathbf{MVB}^1$ and $\mathbf{MVF}^1$. Let $\mathbf{MVB}^2$ and $\mathbf{MVF}^2$ be the backward and forward motion fields used for generating the SI_2 ($\mathbf{MVB}^2$ and $\mathbf{MVF}^2$ are computed separately). Now, the two motion fields $\mathbf{MVB}^2$ and $\mathbf{MVF}^2$ are not symmetrical (*i.e.* they have been adapted to the current motion in the sequence using the PDWZF). These motion fields are used to generate a new residual frame R^2, which is used to create the correlation noise model for decoding the next DCT band. The same procedure is repeated after decoding each DCT band.

3.3 Experimental results

In order to evaluate the performance of the proposed algorithms, we performed extensive simulations, adopting the same test conditions as described in DISCOVER [4, 5]: test video sequences Foreman (150 frames), Soccer (150 frames), Coastguard (150 frames) and Hall (165 frames) are at QCIF spatial resolution and sampled at 15 frames/sec. We also added to the test sequences Stefan (150 frames) and Bus (75 frames).

3.3.1 Parameter tuning

The parameter T_1 plays an important role in the proposed method, since it determines the execution time of the decoding process as well as the achieved performance improvement. $T_1 = \infty$ corresponds to the case where no vector is considered as erroneous, which is equivalent to the DISCOVER codec. On the other hand, when $T_1 = 0$, all blocks are

Table 3.3: Rate-distortion performance gain of Alg. II for *Stefan* and *Foreman* sequences, compared to the DISCOVER codec, for different values of n ($T_1 = 4$ and $T_2 = 5$), using Bjontegaard metric.

Alg. II < T1 = 4 and T2 = 5>					
	$n = 0$	$n = 2$	$n = 4$	$n = 6$	$n = 8$
GOP = 2					
Stefan					
Δ_{R} (%)	-11.17	-13.47	-15.05	-15.89	-16.43
Δ_{PSNR} [dB]	0.69	0.83	0.93	0.99	1.02
Foreman					
Δ_{R} (%)	-15.87	-17.28	-18.17	-18.45	-18.32
Δ_{PSNR} [dB]	0.91	1.01	1.06	1.09	1.08

Table 3.4: Rate-distortion performance gain of Alg. I for *Stefan* and *Foreman* sequences, for GOP sizes of 2 and 8, compared to the DISCOVER codec, for different values of T_1 ($T_2 = 5$), using Bjontegaard metric.

Alg. I - T2 = 5							
	$T_1 = 0$	$T_1 = 2$	$T_1 = 4$	$T_1 = 6$	$T_1 = 8$	$T_1 = 10$	$T_1 = 12$
GOP = 2							
Stefan							
Δ_{R} (%)	-5.51	-6.05	-7.14	-8.07	-7.74	-7.02	-6.03
Δ_{PSNR} [dB]	0.32	0.36	0.42	0.48	0.46	0.41	0.35
Foreman							
Δ_{R} (%)	-10.60	-12.11	-12.29	-10.30	-8.34	-6.79	-5.40
Δ_{PSNR} [dB]	0.58	0.68	0.69	0.58	0.46	0.37	0.29
GOP = 8							
Stefan							
Δ_{R} (%)	-24.34	-24.50	-25.23	-25.27	-24.38	-22.86	-20.96
Δ_{PSNR} [dB]	1.45	1.48	1.53	1.52	1.46	1.35	1.23
Foreman							
Δ_{R} (%)	-38.76	-39.63	-37.82	-33.35	-28.85	-24.15	-20.58
Δ_{PSNR} [dB]	2.21	2.29	2.14	1.84	1.54	1.26	1.04

Table 3.5: Rate-distortion performance gain of Alg. II for *Stefan* and *Foreman* sequences, for GOP sizes of 2 and 8, compared to the DISCOVER codec, for different values of T_1 ($T_2 = 5$ and $n = 4$), using Bjontegaard metric.

Alg. II < T2 = 5 and n = 4 >							
	$T_1 = 0$	$T_1 = 2$	$T_1 = 4$	$T_1 = 6$	$T_1 = 8$	$T_1 = 10$	$T_1 = 12$
GOP = 2							
Stefan							
Δ_{R} (%)	-14.85	-15.03	-15.05	-14.64	-13.94	-13.24	-12.70
Δ_{PSNR} [dB]	0.92	0.93	0.93	0.90	0.86	0.82	0.78
Foreman							
Δ_{R} (%)	-18.32	-18.48	-18.17	-17.62	-17.35	-16.97	-16.74
Δ_{PSNR} [dB]	1.07	1.09	1.06	1.03	1.00	0.98	0.96
GOP = 8							
Stefan							
Δ_{R} (%)	-37.85	-37.99	-37.80	-36.91	-35.89	-34.69	-33.55
Δ_{PSNR} [dB]	2.47	2.48	2.46	2.39	2.30	2.21	2.13
Foreman							
Δ_{R} (%)	-48.86	-48.86	-48.18	-46.92	-46.08	-45.18	-44.43
Δ_{PSNR} [dB]	3.06	3.06	2.98	2.87	2.79	2.71	2.64

considered as erroneous and will be refined after each decoded DCT band. Moreover, the size of the extended block $(8 + n) \times (8 + n)$ can further improve the performance, at the cost of an increase in the decoding computational load.

As for the parameter T_2, Tables 3.1 and 3.2 show the RD performance gain of the

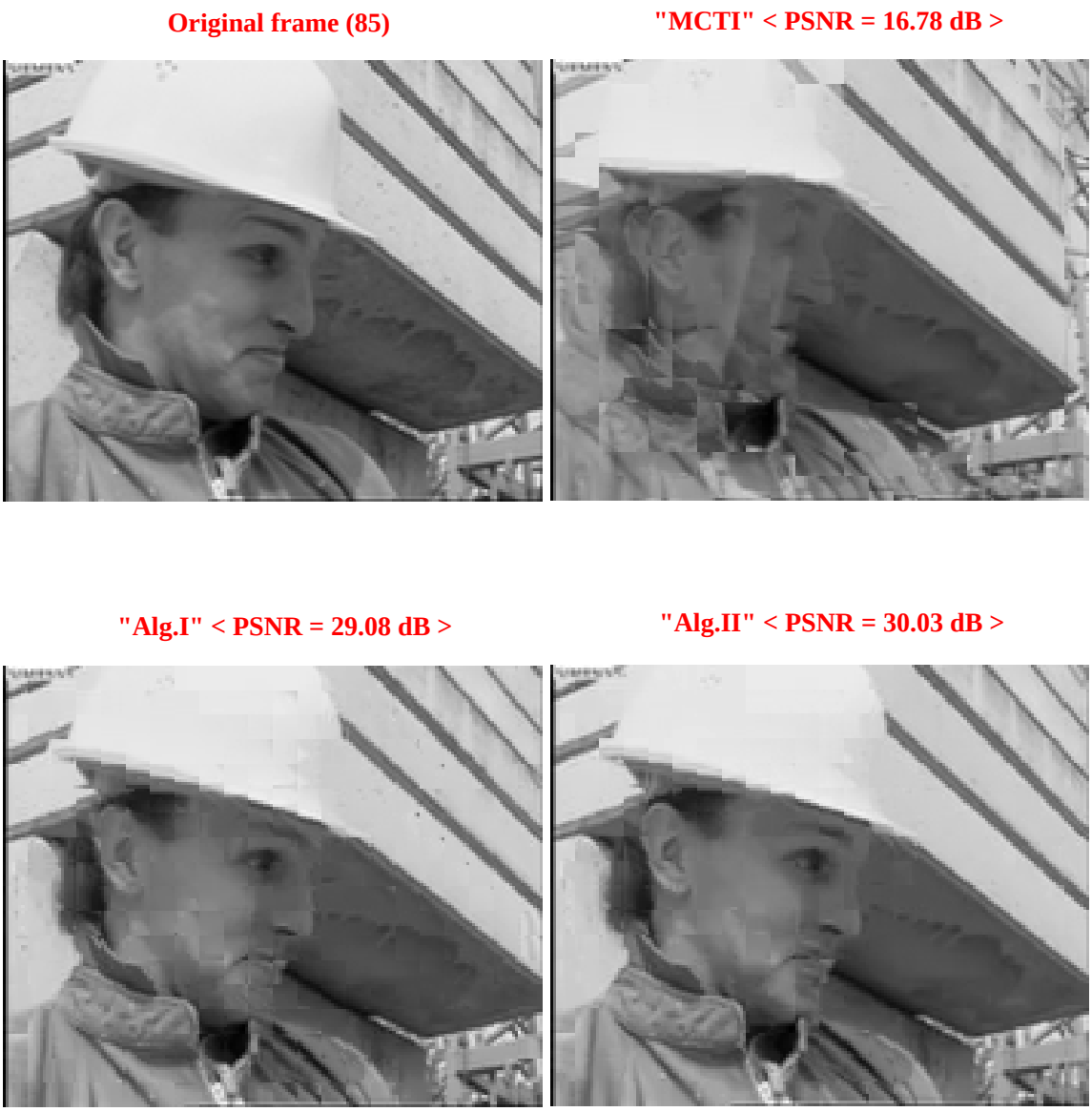

Figure 3.6: Visual result of the SI estimated by MCTI, and the final SI obtained by the proposed algorithms Alg. I and Alg. II, for frame number 85 of Foreman sequence, for a GOP size of 8 (QI = 8).

proposed algorithms I and II respectively, for different values of T_2, with respect to DISCOVER codec. T_1 is set to 4 for the two algorithms, and n is set to 4 for Alg. II. As we can see, the mode 'bidirectional ($T_2 > 0$)' is better than the mode 'unidirectional ($T_2 = 0$)'. However, as the value of this parameter has a limited incidence on the RD performance improvement, it was set to $T_2 = 5$ in our simulations.

Concerning the parameter n that determines the size of the extended block in Alg. II, we show in Table 3.3 the RD performance of the proposed Alg. II for different values of n. In these simulations, T_1 and T_2 are set to 4 and 5 respectively. It is clear that the RD

Table 3.6: Average PSNR of the INSI estimated by MCTI technique and the final SI obtained by the proposed algorithms, for a GOP size equal to 2, 4 and 8 (QI = 8).

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
MCTI [dB]	22.78	29.38	25.37	31.47	22.13	35.81
Alg. I [dB]	25.66	33.46	27.29	32.23	29.65	36.62
Alg. II [dB]	26.61	34.45	27.78	32.35	29.53	36.84
GOP = 4						
MCTI [dB]	21.44	27.64	24.00	29.91	20.87	34.64
Alg. I [dB]	25.14	32.66	26.77	31.14	28.79	35.95
Alg. II [dB]	25.97	33.63	27.33	31.49	28.66	36.10
GOP = 8						
MCTI [dB]	20.78	26.29	22.95	28.82	20.20	33.68
Alg. I [dB]	24.83	32.03	26.17	30.49	28.32	35.38
Alg. II [dB]	25.63	32.93	26.69	30.88	28.19	35.49

performance for $n > 0$ (an extended block is used) is better than the RD performance for $n = 0$. Furthermore, the RD performance for $n = 4$, $n = 6$ and $n = 8$ is almost the same. On the other side, the computational load of the decoding process is much higher for $n = 8$. For this reason, we have set $n = 4$ in the rest of the simulations.

For the important parameter T_1, Tables 3.4 and 3.5 show the RD performance of the proposed algorithms Alg. I and Alg. II respectively. It is clear that the performance tends to be similar for $T_1 = 0$, $T_1 = 2$ and $T_1 = 4$, but it decreases starting at $T_1 > 4$. This means that fewer blocks tend to be selected for the re-estimation. Consequently, in the simulations, we have set the values $T_1 = 4$, along with $T_2 = 5$ and $n = 4$, because of the high performance gain and the low computational load achieved for these values.

3.3.2 SI assessment

Fig. 3.6 shows the visual result of the SI for frame number 85 of Foreman sequence, for a GOP size of 8 (QI = 8). The SI obtained by MCTI technique has a poor quality, as shown in this figure (top-right - PSNR = 16.78 dB). On the contrary, the successive refinement of the SI after decoding each DCT band can significantly enhance the quality of the SI. The final SI frames obtained by the proposed algorithms Alg. I and Alg. II are significantly improved compared to the INSI estimated by MCTI technique.

Table 3.6 shows the average PSNR of the SI obtained with the MCTI technique and the final SI obtained by the proposed techniques Alg. I and Alg. II, for different sequences and different GOP sizes. The successive refinement of the SI by applying Algorithms I and II can significantly improve the quality of the SI.

3.3.3 RD performance assessment of the proposed methods

Fig. 3.7 shows the visual results of the decoded frames obtained by different methods, for frame number 125 of Foreman sequence, for a GOP equal to 8. The decoded frame obtained by DISCOVER codec contains block artifacts. On the contrary, the decoded frame obtained

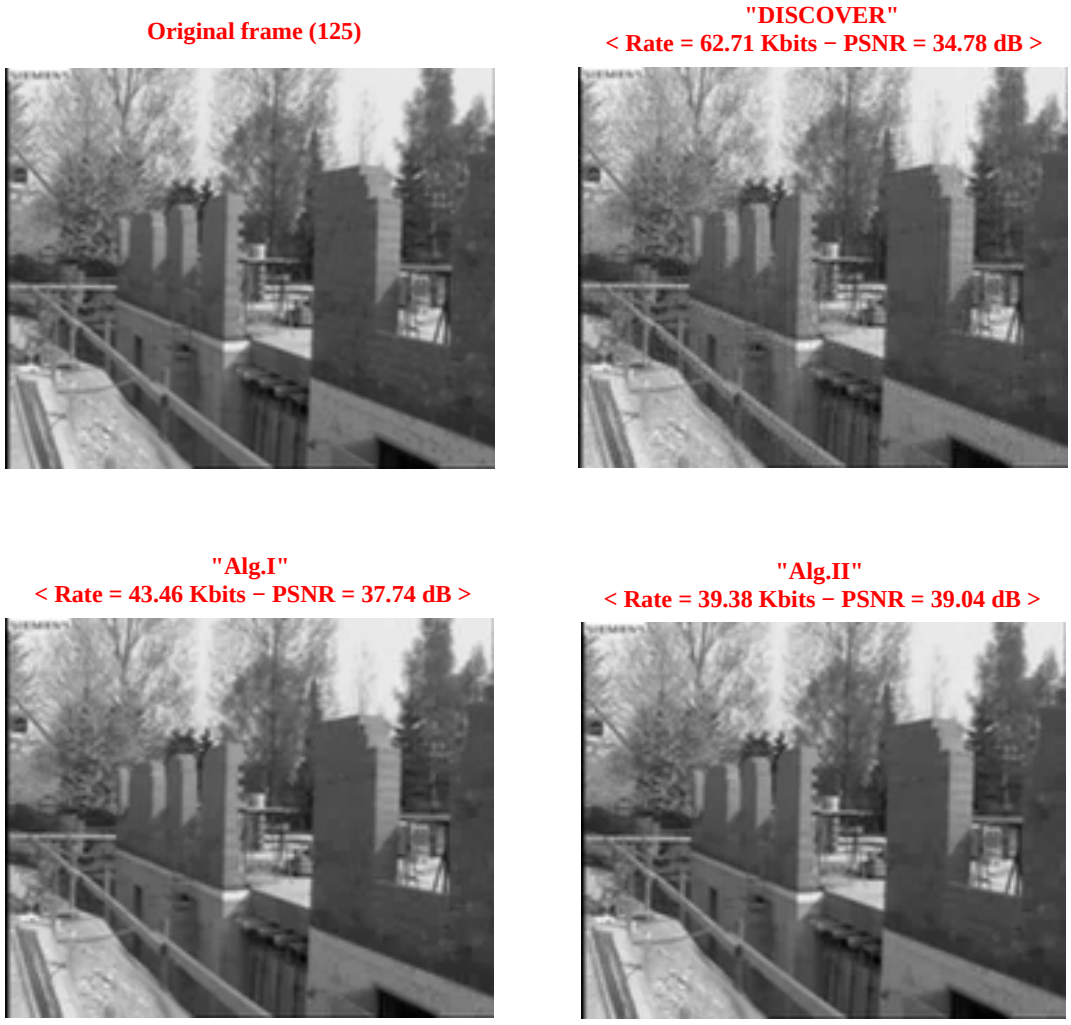

Figure 3.7: Visual results of the decoded frames that are obtained by the proposed methods (Alg. I and II) and DISCOVER codec, for frame number 125 of Foreman sequence.

by the proposed algorithm Alg. I has a better quality(up to 3 dB improvement), with less requested bits (from 62.71 Kbits down to 43.46 Kbits). Moreover, the proposed algorithm Alg. II allows a significant enhancement compared to the DISCOVER codec. The gain is up to 4.26 dB with a bit reduction of 37 % (the rate decreases from 62.71 Kbits down to 39.38 Kbits).

The RD performance of the proposed algorithms Alg. I and Alg. II, along with those of VISNET II codec [9] and Martins *et al.* [10], are shown in Table 3.7, for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, with different GOP sizes (2, 4 and 8), in comparison to DISCOVER codec, using Bjontegaard metric [11].

The first proposed algorithm (Alg. I) respectively gives a gain of up to 0.43, 0.69, 0.01, 0.1, 0.85 and 0.03 dB and an average rate reduction of 6.81, 11.11, 0.12, 2, 14.56 and 0.38 % for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, compared

Table 3.7: Rate-distortion performance gain of VISNET II codec [9], Martins *et al.* [10], Alg. I and Alg. II for *Stefan*, *Foreman*, *Bus*, *Coastguard*, *Soccer* and *Hall* sequences, for GOP sizes of 2, 4 and 8, compared to DISCOVER codec, using Bjontegaard metric.

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
VISNET II [9]						
Δ_R (%)	3.51	-2.41	6.10	1.63	-6.59	1.56
Δ_{PSNR} [dB]	-0.22	0.14	-0.34	-0.08	0.36	-0.11
Martins *et al.* [10]						
Δ_R (%)	-5.25	-6.54	-2.69	-0.98	-9.45	-0.45
Δ_{PSNR} [dB]	0.33	0.40	0.16	0.05	0.54	0.03
Alg. I						
Δ_R (%)	-6.81	-11.11	0.12	-2.00	-14.56	-0.38
Δ_{PSNR} [dB]	0.43	0.69	-0.01	0.10	0.85	0.03
Alg. II						
Δ_R (%)	**-14.06**	**-16.29**	**-4.50**	**-2.24**	**-17.65**	**-1.34**
Δ_{PSNR} [dB]	**0.93**	**1.05**	**0.27**	**0.11**	**1.05**	**0.10**
GOP = 4						
VISNET II [9]						
Δ_R (%)	-0.08	-9.36	2.57	-0.78	-10.01	0.88
Δ_{PSNR} [dB]	0.00	0.53	-0.14	0.03	0.58	-0.05
Martins *et al.* [10]						
Δ_R (%)	-13.38	-16.96	-7.37	-4.26	-14.63	-1.96
Δ_{PSNR} [dB]	0.85	1.04	0.45	0.18	0.90	0.12
Alg. I						
Δ_R (%)	-17.90	-24.33	-7.99	-7.33	-20.78	-2.17
Δ_{PSNR} [dB]	1.16	1.53	0.48	0.31	1.30	0.13
Alg. II						
Δ_R (%)	**-27.84**	**-32.65**	**-15.82**	**-11.94**	**-25.08**	**-4.24**
Δ_{PSNR} [dB]	**1.93**	**2.19**	**0.99**	**0.52**	**1.61**	**0.27**
GOP = 8						
VISNET II [9]						
Δ_R (%)	-1.76	-14.05	-0.68	-8.44	-11.37	-5.36
Δ_{PSNR} [dB]	0.11	0.82	0.05	0.36	0.68	0.33
Martins *et al.* [10]						
Δ_R (%)	-18.36	-23.96	-12.66	-9.67	-17.68	-6.99
Δ_{PSNR} [dB]	1.23	1.54	0.81	0.43	1.13	0.42
Alg. I						
Δ_R (%)	-23.02	-32.52	-14.08	-16.35	-23.12	-8.97
Δ_{PSNR} [dB]	1.56	2.17	0.90	0.73	1.50	0.54
Alg. II						
Δ_R (%)	**-34.13**	**-41.88**	**-22.83**	**-24.21**	**-28.16**	**-11.04**
Δ_{PSNR} [dB]	**2.51**	**3.02**	**1.53**	**1.14**	**1.88**	**0.68**

to the DISCOVER codec, with a GOP size of 2. The gains become more important for larger GOP sizes. It is clear that the performance of the algorithm Alg. I is better than DISCOVER for all GOP sizes.

Moreover, the second algorithm (Alg. II) can achieve a gain of up to 0.93, 1.05, 0.27, 0.11, 1.05 and 0.1 dB and an average rate reduction of 14.06, 16.29, 4.5, 2.24, 17.65 and 1.34 % for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, compared to the DISCOVER codec. The algorithm Alg. II always allows a gain with respect to the algorithm Alg. I. It is clear that the gain in RD performance increases with the GOP length. In this case, classical interpolation techniques for SI generation become less effective.

The proposed algorithm Alg. II allows a significant gain of 3.02 dB for Foreman sequence compared to DISCOVER codec, for a GOP size of 8. Moreover, the performance of

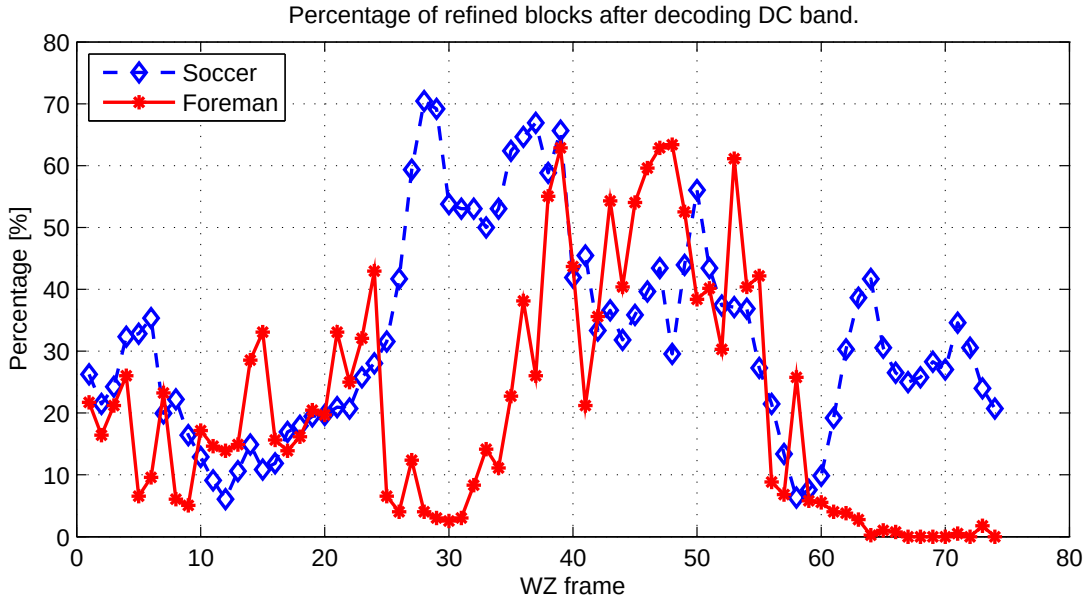

Figure 3.8: Percentage of refined blocks after decoding the DC band for Soccer and Foreman sequences with a GOP size 2.

the proposed algorithms is better than both VISNET II codec [9] and Martins *et al.* [10] for all test sequences.

For sequences containing slow motion such as Coastguard and Hall, the gains between the proposed algorithms and the DISCOVER codec are smaller and range between 0.03 and 1.14 dB.

Fig. 3.8 indicates the percentage of refined blocks, that is, the percentage of blocks that are identified as having suspicious motion vectors, in one execution of the refinement procedure after decoding the DC band, for Soccer and Foreman sequences, with a GOP size equal to 2. It is clear that the percentage of refined blocks increases with the motion level within the video sequence. For the Foreman sequence, the percentage tends to zero due to the low motion at the end of the sequence.

In Fig. 3.9, we show the average rate and the average PSNR of the DISCOVER codec and the proposed algorithms Alg. I and Alg. II for Foreman sequence, for a GOP size of 8 (QI = 8). The proposed algorithms allow a reduction in the rate compared to DISCOVER codec for all test sequences. At the same time, the quality of the decoded frames is improved compared to DISCOVER codec. The proposed algorithm Alg. II leads the best performance (*i.e.*, the highest quality of the decoded frames at the lowest bit rate).

Figs 3.10 and 3.11 show the RD performance of DISCOVER codec, the proposed algorithms, H.264/AVC Intra and H.264/AVC No motion for all test sequences, for GOP sizes of 2 and 8. The difference between the performance of DISCOVER codec and H.264/AVC No motion is up to 1.15 dB for Foreman sequence for a GOP size 2. In this case, the proposed algorithm can achieve the same performance as H.264/AVC No

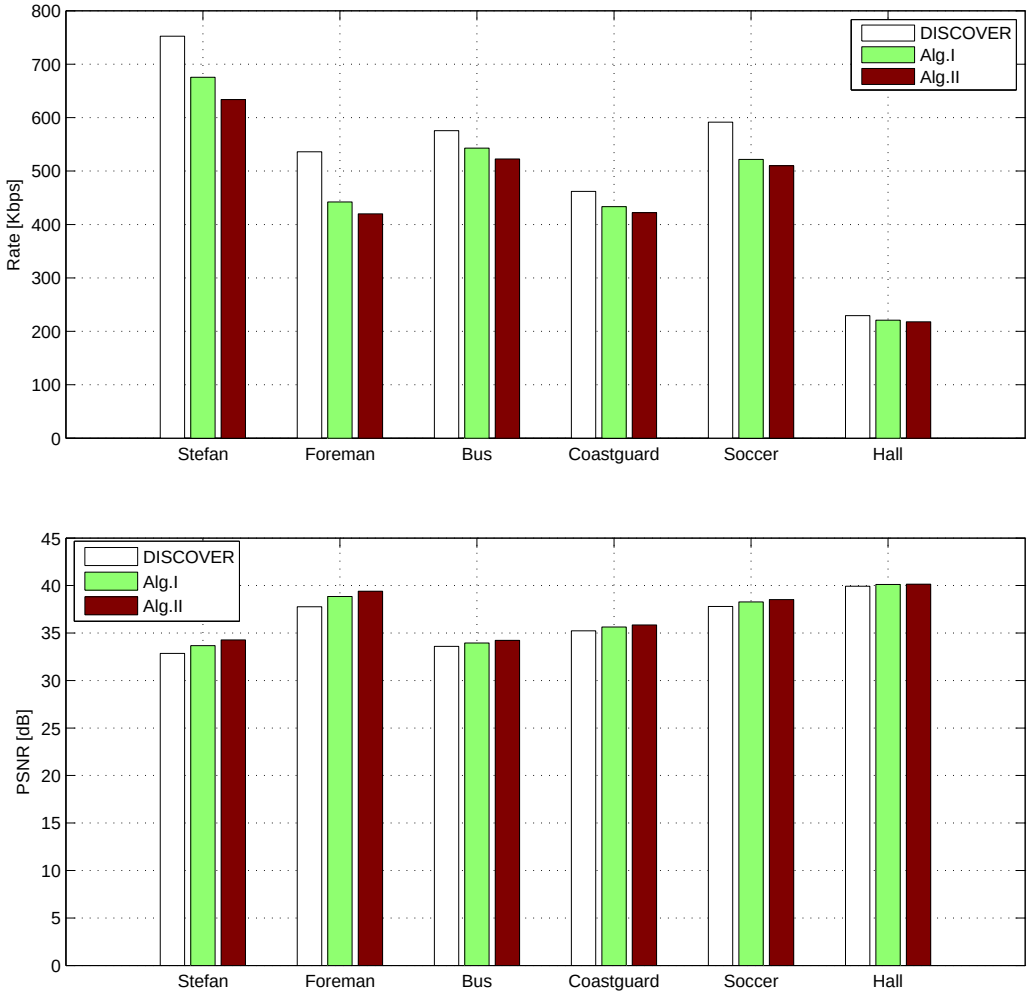

Figure 3.9: Average rate (Kbps) and PSNR (dB) of DISCOVER and the proposed algorithms, for all test sequences, for a GOP size of 8 (QI = 8).

motion, whereas the performance of DISCOVER codec is close to H.264/AVC Intra. For a GOP size 8, the proposed algorithm (Alg. II) gives a gain up to 0.15 dB compared to H.264/AVC No motion. For some sequences, the proposed algorithm Alg. II leads to the best RD performance, and for the other sequences, the proposed algorithm can reduce the gap between DISCOVER codec and H.264/AVC No motion.

In Fig. 3.12, we show the performance of DISCOVER codec and Alg. II for all GOP sizes and for all test sequences. For the Foreman sequence, the performance loss, using DISCOVER codec, exceeds 1.5 dB, when the GOP size increases from 2 to 4 and from 4 to 8. In our proposed algorithm, this loss is less than 0.1 dB, when the GOP size increases from 2 to 4, and less than 0.4 dB between the GOP sizes 2 and 8, despite a big difference of 3 dB in the case of DISCOVER codec. For all test sequences, the gap between the performances for different GOP sizes is also significantly reduced compared

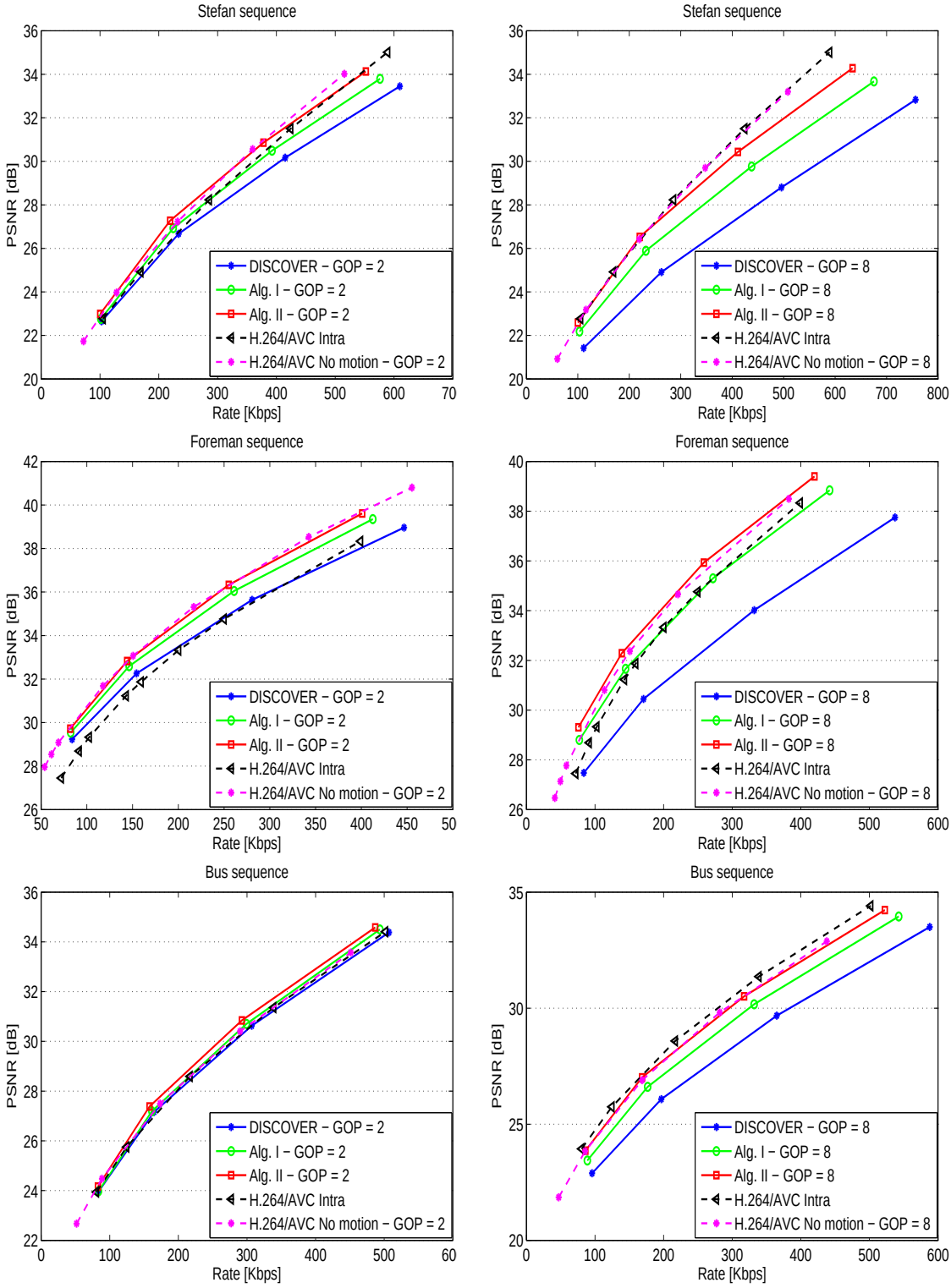

Figure 3.10: RD performance of DISCOVER, proposed algorithms, H.264/AVC Intra and H.264/AVC No motion for GOP sizes of 2 and 8, for Stefan, Foreman and Bus sequences.

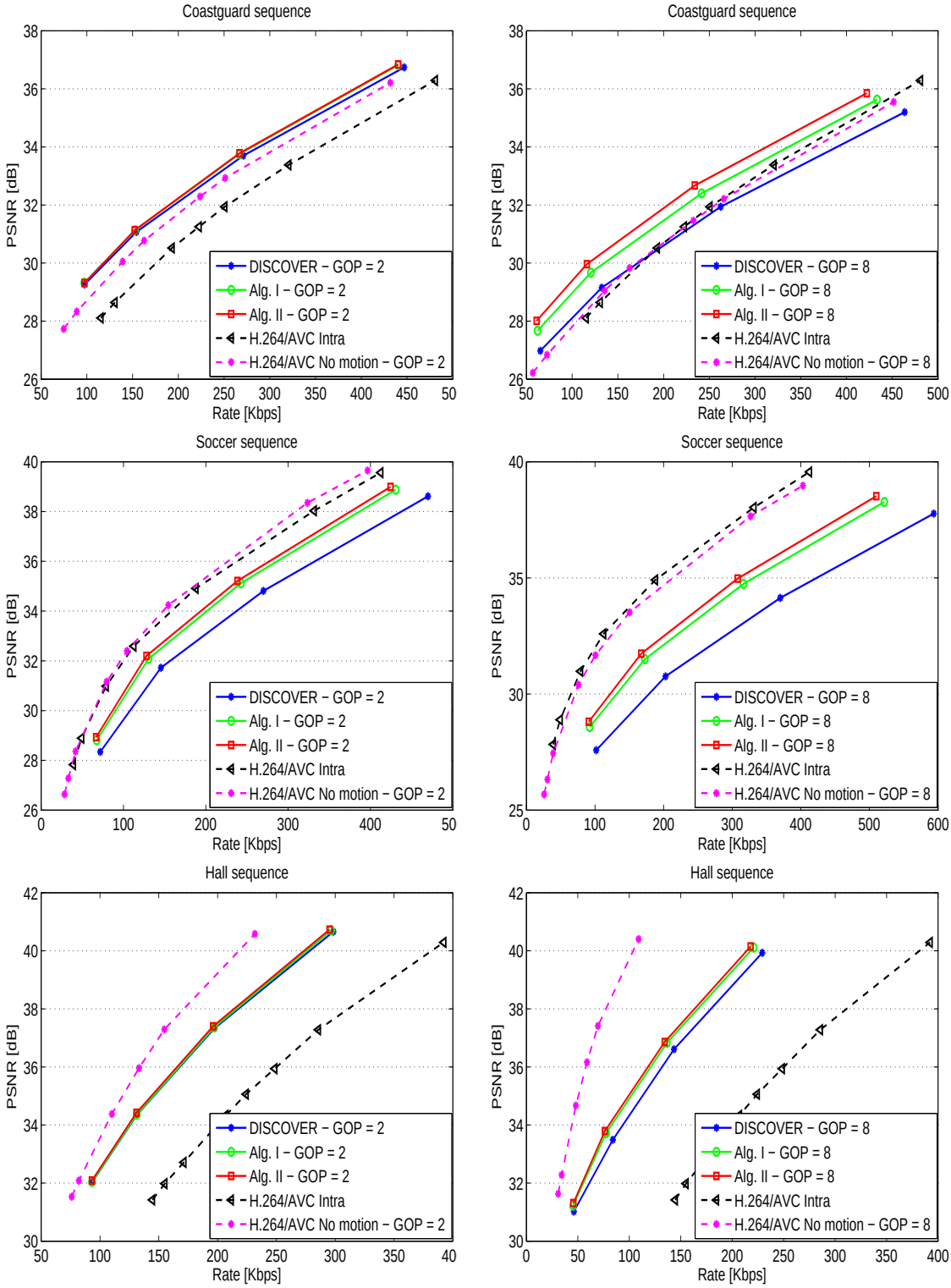

Figure 3.11: RD performance of DISCOVER, proposed algorithms, H.264/AVC Intra and H.264/AVC No motion for GOP sizes of 2 and 8, for Coastguard, Soccer and Hall sequences.

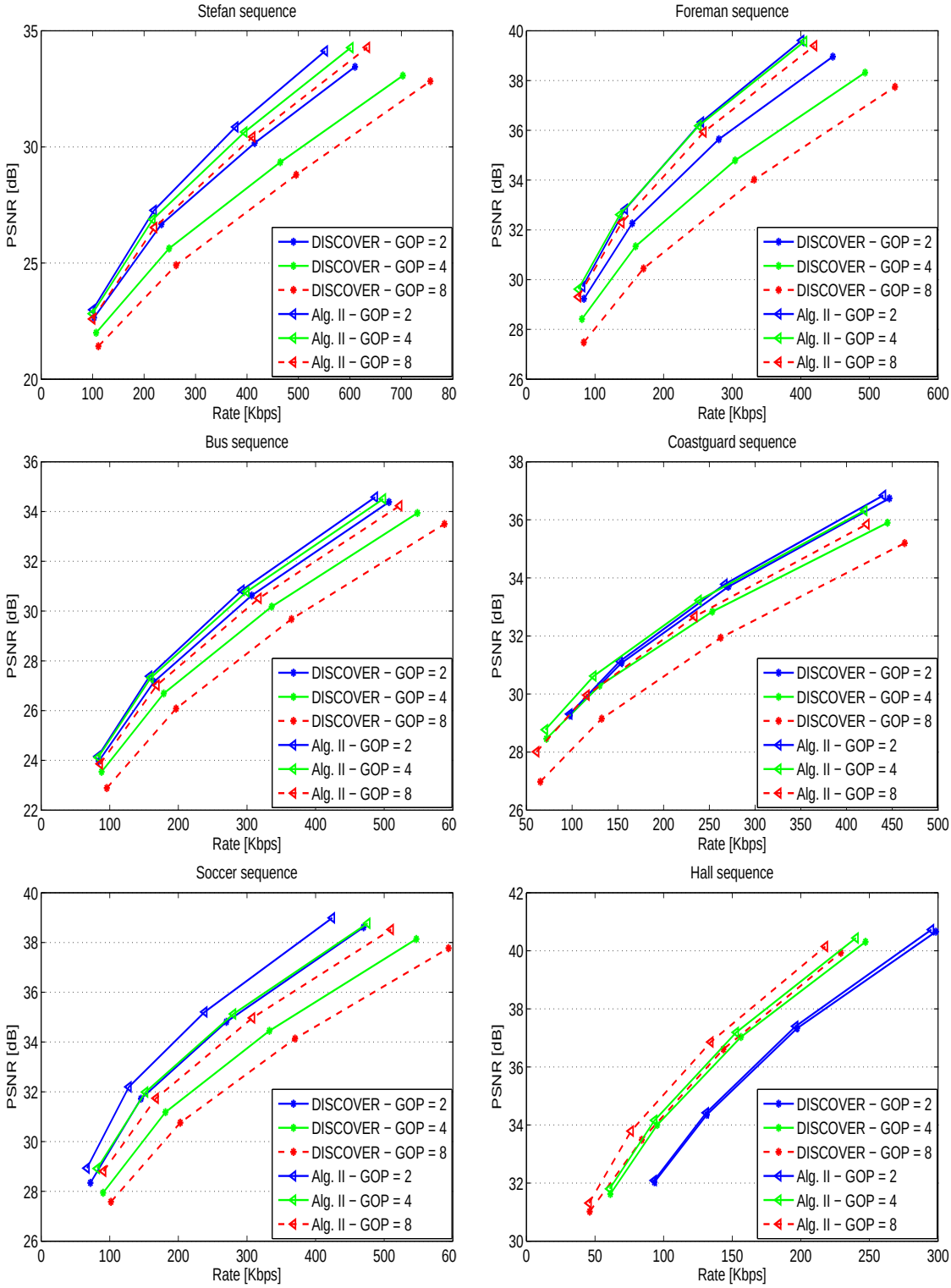

Figure 3.12: RD performance of the DISCOVER codec and the proposed algorithm Alg. II for all GOP sizes.

to DISCOVER codec.

It can be concluded that the performance gains are more substantial for high motion sequences and for long GOP sizes. They are mainly associated with the proposed algorithms for successive refinement of the SI interpolation after each decoded DCT band, which is the major contribution of our present work with respect to the reference codec.

3.4 Summary

In this chapter, we introduced two new techniques for the successive refinement of the SI using the PDWZF based on the successive decoding of the DCT bands. The initial SI is generated using MCTI technique. Then, the decoder reconstructs a PDWZF after the decoding of the first DCT band. The latter is used in order to detect the suspicious motion vectors in the last SI. Furthermore, the motion vectors of the suspicious blocks are re-estimated using two different algorithms.

As a consequence, a new SI is generated for decoding the next DCT band. Then, a new PDWZF is reconstructed and the same steps are carried out to improve the quality of the SI, after each decoded DCT band. The successive refinement of the SI can significantly improve the performance of DVC, since it generally results in a reduced amount of parity information requested by the decoder through the return channel. At the same time, the quality of the decoded WZF is improved during reconstruction.

Experimental results showed that our proposed method can achieve a gain in RD performance of up to 1.08 dB for a GOP size of 2 and 3.05 dB for longer GOP sizes, compared to DISCOVER codec, especially when the video sequence contains high motion. The improvement becomes even more important as the GOP size increases.

The material in this chapter was published in:

1 A. Abou-Elailah, J. Farah, M. Cagnazzo, B. Pesquet-Popescu, and F. Dufaux “Improved Side Information Generation for Distributed Video Coding”, *European Workshop on Visual Information Processing (EUVIP)*, July 2011, Paris, France.

2 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu “Successive Refinement of Motion Compensated Interpolation for Transform-Domain Distributed Video Coding”, *European Signal Processing Conference (EUSIPCO)*, August 2011, Barcelona, Spain.

3 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu “Amélioration Progressive de l’Information Adjacente Pour le Codage Vidéo Distribué”, *GRETSI*, September 2011, Bordeaux, France.

Chapter 4

Side information improvement techniques

Contents

The Side Information (SI) is commonly generated using the Motion-compensated temporal interpolation (MCTI) technique [7]. In fact, the SI is not accurate when the temporal distance between the neighboring reference frames increases or when the video sequence contains fast motion. In this chapter, three approaches are proposed for SI improvement in transform-domain DVC.

First, we propose a new method that uses backward and forward motion estimation to enhance the generation of the initial SI (INSI). It consists in selecting reliable motion vectors from the backward and forward estimations.

Second, we propose a new approach that allows improving the SI using an adaptive motion search area. This solution is based on our successive refinement technique [12], previously explained in the former chapter, which consists in progressively improving the SI after each decoded DCT-band. However, until now a constant search area has been used to refine the SI after each decoded DCT-band, regardless of the distance between the reference frames. This method achieves a significant gain, compared to DISCOVER codec, for sequences containing fast motion, as well as for long duration GOPs. In the second proposed approach, variable search areas are initially set according to the temporal distance between the neighboring reference frames. We first start by generating an INSI by using the backward and forward reference frames, similarly to the SI generated in DISCOVER codec. The decoder then reconstructs a Partially Decoded Wyner-Ziv Frame (PDWZF) by correcting the INSI with the parity bits of the first DCT-band. Afterwards, the PDWZF, along with the backward and forward reference frames, is used to adapt the initial search area. Furthermore, the adapted search area is used to refine the INSI. Finally, we correct this improved INSI with the parity bits of the next DCT-band and we repeat the same procedure to decode all DCT-bands of the current WZF.

A third method that we propose aims at refining the SI for large GOP sizes. In these conditions, it is known that the central SI is of worse quality w.r.t. the lateral ones, because the reference frames used for estimating the central WZF are farther apart. The consequence is that the PSNR of the decoded frames fluctuates within the GOP. Therefore, we propose to re-estimate the SI using the already decoded WZF and the adjacent decoded frames (WZF or KF). During the re-estimation procedure, an adaptive search area and a variable block size are also used. Finally, the WZFs are reconstructed with an improved quality, using the same parity bits sent during the first step.

This chapter is structured as follows. First, the first approach is introduced in Section 4.1 with experimental results and analysis. The second approach based on successive refinement of the SI using an adaptive motion search area is described in Section 4.2 with the obtained results. The third approach of re-estimating the SI using the already decoded WZF, along with the neighboring decoded frames, is illustrated in Section 4.3. Finally, conclusions are presented in Section 4.4.

4.1 Backward and forward motion estimation

In this section, we first present the main existing methods for SI generation. Then, we describe the proposed method for SI generation. Finally, the experimental results are shown.

4.1.1 Related work

The SI is generally estimated using interpolation/extrapolation of the available information at the decoder side, since the reconstructed WZF is not available. In [33], the SI is simply generated by averaging the two reference frames. For each pixel p in the interpolated SI, the two pixels at the same position in the Backward reference frame (BRF) and the Forward reference frame (FRF) are averaged:

$$\mathrm{SI}(p) = \frac{\mathrm{BRF}(p) + \mathrm{FRF}(p)}{2} \tag{4.1}$$

This method is very simple and can be efficient for sequences containing low motion. When the sequence contains medium or high motion, this simple averaging of the two frames leads to a bad quality SI. For this reason, the authors propose in the same paper [33] a more elaborated technique based on Symmetric Motion Vectors (SMV) interpolation. For a given block in the interpolated SI, the motion vector from the BRF to the SI is assumed to be the same as the motion vector from the SI to FRF. Thus, this method consists in finding the best candidate motion vector that gives the smallest Sum Square Distance (SSD) for the current block b:

$$\mathrm{SSD}(\mathbf{v}) = \sum \left(\mathrm{BRF}(p - \mathbf{v}) - \mathrm{FRF}(p + \mathbf{v})\right)^2 \tag{4.2}$$

where $\mathbf{v}$ is a candidate motion vector. The SMV interpolation was used in many works [6, 50, 60–63]since it allows for a better interpolation, compared to simple averaging, when motion activity is present in the sequence. However, the SMV method is not efficient for sequences exhibiting complex and high motion. For this reason, Aaron *et al.* proposed a technique [34] that aims at using the SMV to obtain symmetrical bidirectional motion estimation. Then, smoothness constraints on the estimated motion vectors are applied. In addition, overlapped block motion compensation is performed.

In [64], the edge information of the decoded frames is used to improve the accuracy of the SI. However, the largest advance for SI generation was proposed in [7] and has been used in the DISCOVER and VISNET II projects. This technique is called Motion-Compensated Temporal Interpolation (MCTI) and is composed of four modules: forward motion estimation, bi-directional motion estimation, spatial smoothing of motion vectors and bi-directional motion compensation (it is fully described in Chapter 2). In [65], the authors propose a novel frame interpolation method that allows improving the MCTI using a block-adaptive matching algorithm. This technique enables block size adaptation to local motion activity within the reference frames. Furthermore, Huang and Forchhammer [66] proposed an algorithm that consists of a variable block size, based on Y, U and V components motion fields, and an adaptive weighted overlapped block motion compensation .

In this section, we describe the proposed method for SI generation based on backward

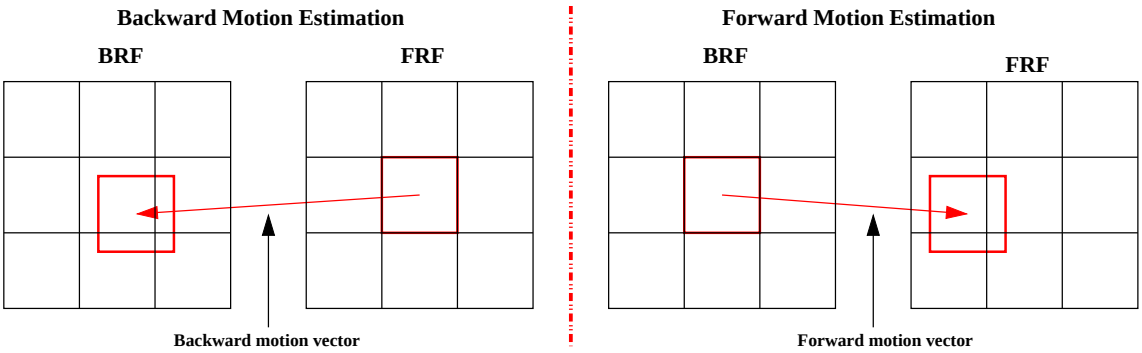

Figure 4.1: Backward and forward motion estimation.

and forward motion estimation. We also use a quadtree refinement approach to predict the motion vectors from a coarse motion estimation. The obtained results are compared to the DISOVER codec.

4.1.2 Proposed method

In the DISCOVER codec, the SI is generated using MCTI technique. Here, we propose a new approach to generate the SI, which is based on backward and forward motion estimation. We refer to backward motion estimation when we search in the BRF to find the most similar block to the target block in FRF and to forward motion estimation when the most similar block to the target block in BRF is found in FRF (see Fig. 4.1).

Fig. 4.2 shows an example of four parts A, B, C and D in the backward and forward reference frames. Parts A and B are defined in BRF, and parts C and D are defined in FRF. As shown in the figure, parts A and D cannot be found in FRF and BRF respectively. Conversely, parts B and C can be found in FRF and BRF respectively. While the motion vectors for the blocks in part D cannot be reliable in backward motion estimation, the motion vectors for the same block positions can be reliable and accurate in forward motion estimation (the part B can be found in FRF). Similarly, the motion vectors for the blocks in part A are not reliable in forward motion estimation. In order to find the correct motion vectors for all blocks, we propose a new method that consists in combining backward and forward motion estimations. The proposed SI generation is depicted in Fig. 4.3. This new approach is described as follows:

- **Low-Pass Filtering:** The reference frames are padded and low-pass filtered in order to improve the motion vectors reliability.
- **Backward and Forward Motion Estimation:** A block matching algorithm is applied to estimate the backward and forward motion vector fields. These motion estimations are computed with a block size $BS_0 \times BS_0$, a search area ($\mathbf{S}$) of $\pm SA_0$ pixels, and a step size of N_0 pixels. In the block matching algorithm, the Weighted Mean

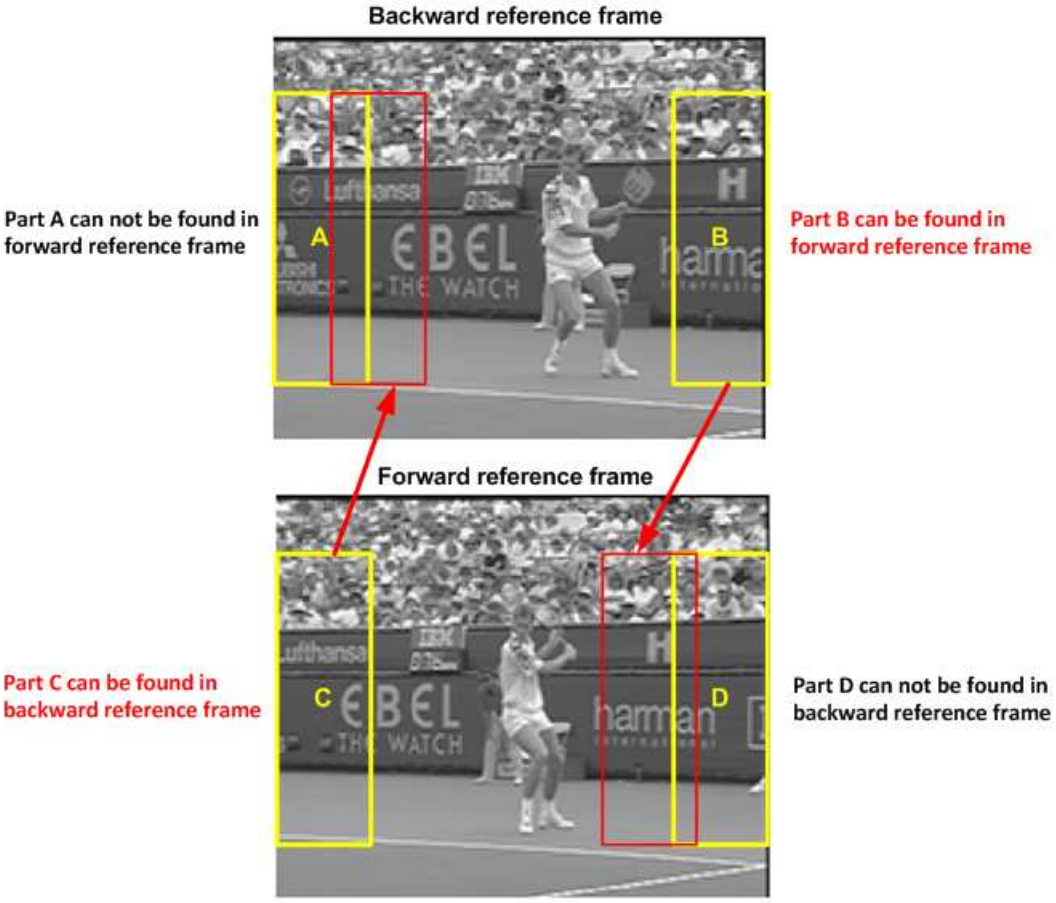

Figure 4.2: An example of backward and forward reference frames.

Absolute Difference (WMAD) criterion is used to compute the similarity between the target block b in the frame F_1 (F_1 can be the BRF in forward motion estimation or the FRF in backward motion estimation) and the shifted block in the frame F_2 (F_2 can be the BRF in backward motion estimation or the FRF in forward motion estimation) by the motion vector $\mathbf{v} \equiv (v_x, v_y) \in \mathbf{S}$, as follows:

$$\mathrm{WMAD}(\mathrm{b}, \mathbf{v}) = \frac{1}{\mathrm{BS}_0^2} \sum_{x=x_0}^{x_0+\mathrm{BS}_0} \sum_{y=y_0}^{y_0+\mathrm{BS}_0} |\mathrm{F}_1(x, y) - \mathrm{F}_2(x + v_x, y + v_y)| \left(1 + \lambda \sqrt{v_x^2 + v_y^2}\right) \tag{4.3}$$

where (x_0, y_0) is the up-left pixel of the block b and λ is a penalty factor which allows to penalize the MAD by the length of the motion vector $\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2}$. λ is empirically set to 0.01. The objective of the block matching algorithm is to find the block in F_2 most similar to the target block in F_1. In other words, the algorithm aims at obtaining the best motion vector $\mathbf{V}_\mathrm{b}$ for the block b by minimizing the WMAD as follows:

$$\mathbf{V}_\mathrm{b} = \arg \min_{\mathbf{v_i} \in \mathbf{S}} \mathrm{WMAD}(\mathrm{b}, \mathbf{v}_i). \tag{4.4}$$

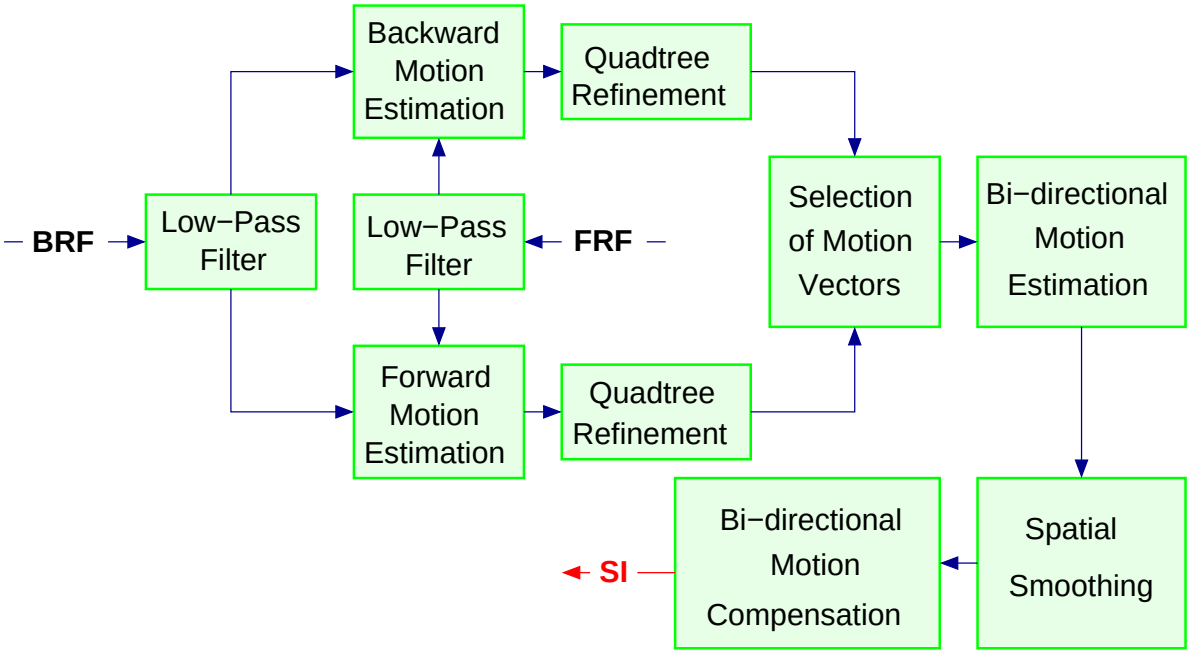

Figure 4.3: Proposed SI generation.

The obtained motion vectors are then refined in N_1 ($N_1 < N_0$) pixel(s) accuracy within a search area of $\pm SA_1$ pixels ($SA_1 \ll SA_0$). The MAD criterion is used in the refinement process. Let $\mathbf{V}_b^{\text{back}}$ and $\mathbf{V}_b^{\text{for}}$ be the obtained backward and forward motion vectors for the block b respectively, and MAD_{back} and MAD_{for} the mean absolute differences corresponding to the motion vectors $\mathbf{V}_b^{\text{back}}$ and $\mathbf{V}_b^{\text{for}}$ respectively. We now aim at replacing the false motion vectors that can be obtained at the borders of the image by reliable ones (for example, the obtained motion vectors for the blocks that form the part D in backward motion estimation are not reliable). The motion vectors can be improved as follows:

$$
\begin{cases}
\textbf{if } |\text{MAD}_{\text{back}} - \text{MAD}_{\text{for}}| < T_b \\
\quad \text{The motion vectors } \mathbf{V}_b^{\text{back}} \text{ and } \mathbf{V}_b^{\text{for}} \text{ are considered to be reliable} \\
\textbf{otherwise} \\
\quad \textbf{if } \text{MAD}_{\text{back}} < \text{MAD}_{\text{for}} \\
\quad\quad \text{The motion vector } \mathbf{V}_b^{\text{for}} \text{ is dropped} \\
\quad \textbf{otherwise} \\
\quad\quad \text{The motion vector } \mathbf{V}_b^{\text{back}} \text{ is dropped}
\end{cases}
$$

where T_b is a threshold.

- **Quad-tree Refinement:** The backward and forward motion vectors are obtained for each $BS_0 \times BS_0$ block. The objective of this step is to estimate the motion vectors for blocks of size $BS_1 \times BS_1$ ($BS_{i+1} = BS_i/2$) based on the obtained motion vectors of

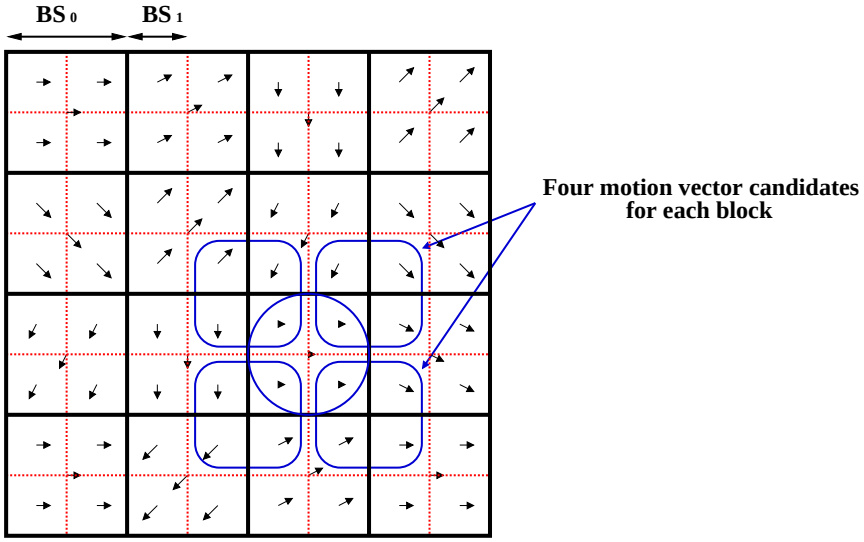

Figure 4.4: Motion vectors candidates for each $BS_1 \times BS_1$ block.

the $BS_0 \times BS_0$ blocks from the previous step. Thus, each $BS_0 \times BS_0$ block is divided into four $BS_1 \times BS_1$ blocks. First, each four $BS_1 \times BS_1$ blocks inherit the motion vector of the corresponding $BS_0 \times BS_0$ block (see Fig. 4.4). Then, for every $BS_1 \times BS_1$ block b, the motion vectors of the neighboring blocks are taken into account to select the most accurate one. As shown in Fig. 4.4, one motion vector among the four different candidates $\mathbf{v}_n$ $(n = 1, ..., 4)$ is selected for the block b according to:

$$\mathbf{v}_\mathrm{b} = \arg\min_{\mathbf{v}_n} \mathrm{MAD}(\mathrm{b}, \mathbf{v_n}). \tag{4.5}$$

The motion vector for each $BS_1 \times BS_1$ block is computed using Eq. 4.5. Then, the $BS_1 \times BS_1$ block is split into four $BS_2 \times BS_2$ blocks. The motion vectors for the $BS_2 \times BS_2$ blocks are computed using the same procedure. Finally, the same procedure can be repeated until obtaining the motion vectors for $BS_M \times BS_M$ blocks $(M > 1)$.

- **Selection of Motion Vectors:** This step consists in selecting the best motion vector $\widetilde{\mathbf{v}}_\mathrm{b}$ (backward or forward motion vector) for each block b ($BS_M \times BS_M$). Let $\mathrm{MAD}'_{\mathrm{back}}$ and $\mathrm{MAD}'_{\mathrm{for}}$ be the mean absolute difference corresponding to the motion vectors $\mathbf{V}_b^{\mathrm{back}}$ and $\mathbf{V}_b^{\mathrm{for}}$ respectively. The selection of the best motion vector is done as follows:

$$\begin{cases} \textbf{if } \mathrm{MAD}'_{\mathrm{back}} < \mathrm{MAD}'_{\mathrm{for}} \\ \quad \widetilde{\mathbf{v}}_\mathrm{b} = \mathbf{V}_b^{\mathrm{back}} \\ \textbf{otherwise} \\ \quad \widetilde{\mathbf{v}}_\mathrm{b} = -\mathbf{V}_b^{\mathrm{for}} \end{cases}$$

- **Bi-directional Motion Estimation:** First, we aim at splitting the obtained motion vectors to estimate bi-directional motion vectors for the blocks in WZF. For each block w in WZF, the distances between the center of the block w and the center of each obtained motion vector are computed. The closest motion vector to the block w is selected. Then, the selected motion vector is associated to the center of the block w, and divided by symmetry to obtain the bidirectional motion field. Second, the bidirectional motion vectors are refined within a small search area $\mathbf{S}'$ of ±SSR pixels in half-pixel accuracy. Let $(\mathbf{r}_b, -\mathbf{r}_b)$ be the bidirectional motion vector for the block b in WZF. A small displacement $\mathbf{d}_b^{\text{ref}}$ is added to the bidirectional motion vector during the refinement process. This displacement is obtained as follows:

$$\mathbf{d}_{\mathbf{b}}^{\text{ref}} = \arg\min_{\mathbf{d}\in\mathbf{S}'} \text{MAD}(\text{b}, \mathbf{r}_b, \mathbf{d}) \tag{4.6}$$

with

$$\begin{aligned}&\text{MAD}(\text{b}, \mathbf{r}, \mathbf{d}) = \tfrac{1}{\text{L}^2} \textstyle\sum_{x=x_0}^{x_0+\text{L}} \sum_{y=y_0}^{y_0+\text{L}} \\ &|\text{BRF}(x + r_{bx} + d_x, y + r_{by} + d_y) - \text{FRF}(x - r_{bx} - d_x, y - r_{by} - d_y)|\end{aligned} \tag{4.7}$$

where $\mathbf{d} \equiv (d_x, d_y)$ and $\mathbf{r}_b \equiv (r_{bx}, r_{by})$. Even though the size of the block is $\text{BS}_M \times \text{BS}_M$, an extended block of $\text{L} \times \text{L}$ ($\text{L} = \text{BS}_M + \text{EX}$) is used to compute the MAD in the refinement process.

- **Spatial Smoothing:** The obtained bidirectional motion vectors may sometimes present low spatial coherence. The spatial smoothing algorithm aims at achieving higher motion field spatial coherence, by reducing the number of suspicious bidirectional vectors. For each block b, the spatial motion smoothing algorithm considers the neighboring bidirectional motion vectors as candidates for the block b. The weighted median vector field [48] is used to select the best bidirectional motion vector from the candidate bidirectional motion vectors as follows:

$$\mathbf{s}_b = \arg \min_{k=1,2,\ldots,\text{Nb}} \left(\sum_{i=1}^{\text{Nb}} a_i ||\mathbf{r}_k - \mathbf{r}_i|| \right), \tag{4.8}$$

with

$$a_i = \frac{1}{\text{L}^2} \sum_{x=x_0}^{x_0+\text{L}} \sum_{y=y_0}^{y_0+\text{L}} |\text{BRF}(x + r_{ix}, y + r_{iy}) - \text{FRF}(x - r_{ix}, y - r_{iy})| \tag{4.9}$$

Nb is the number of neighboring blocks, and $\text{L} = \text{BS}_M + \text{EX}$.

- **Bi-directional Motion Compensation:** Once the final bidirectional motion vectors are estimated, the SI can be interpolated using bidirectional motion compensation as follows:

$$\text{SI}(\mathbf{p}) = \frac{1}{2}(\text{BRF}(\mathbf{p} + \mathbf{s}_\text{b}) + \text{FRF}(\mathbf{p} - \mathbf{s}_\text{b})), \tag{4.10}$$

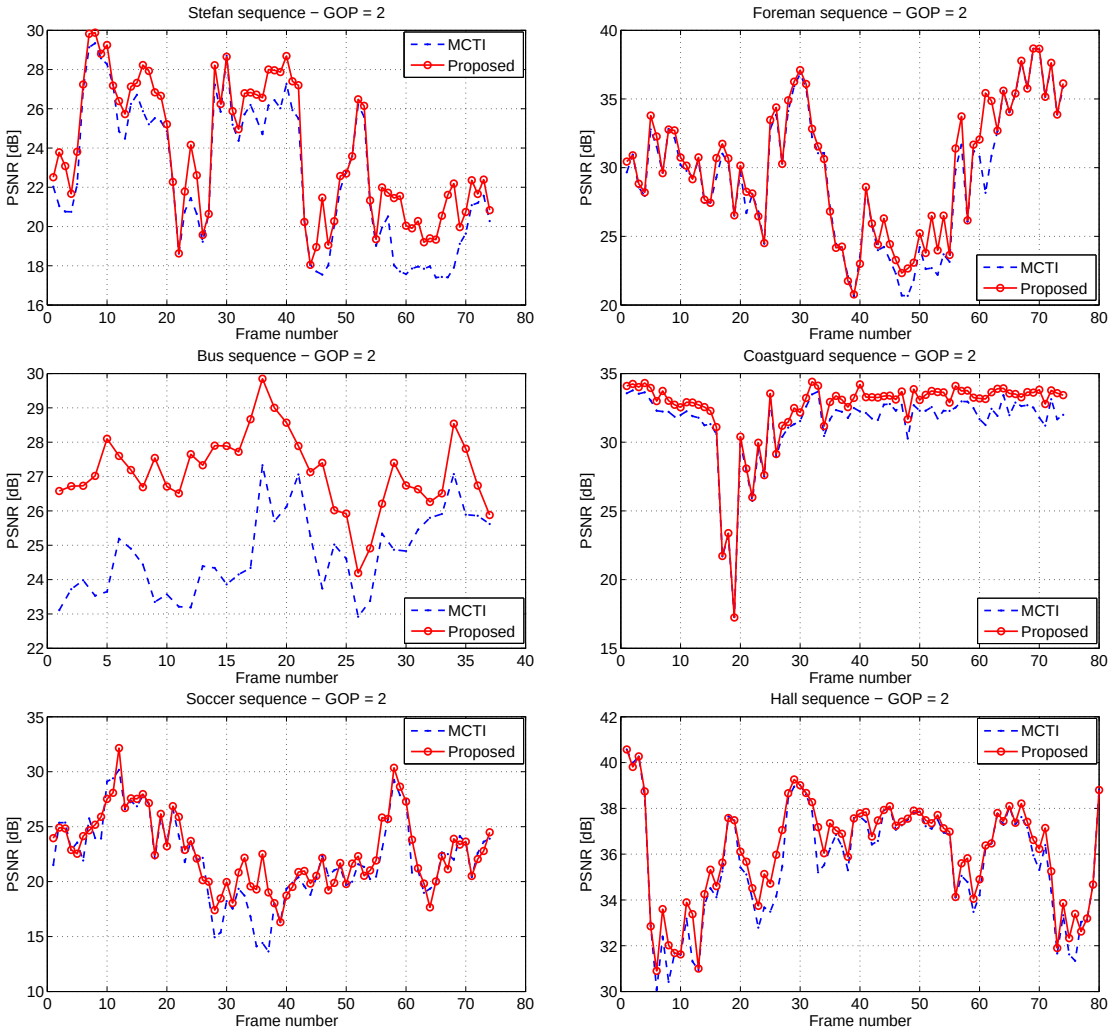

Figure 4.5: PSNR of the proposed SI generation (SIG) and the MCTI SI generation techniques for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, for a GOP size of 2.

where $\mathbf{s}_b$ and $-\mathbf{s}_b$ are the bidirectional motion vectors, associated to the position $\mathbf{p} = (x, y)$ toward the BRF and FRF respectively.

4.1.3 Experimental results

With the aim of evaluating the performance of the proposed methods, we performed extensive simulations, adopting the same test conditions as described in DISCOVER[4] , *i.e.* test video sequences are at QCIF spatial resolution and sampled at 15 frames/sec.

The parameters of the proposed method are set as follows in the experiments: $\mathrm{BS}_0 = 32$ pixels, $\mathrm{BS}_M = 4$ pixels, $\mathrm{SA}_0 = 48$ pixels, $\mathrm{SA}_1 = 3$ pixels, SSR = 1.5 pixels, $\mathrm{N}_0 = 2$ pixels, $\mathrm{N}_1 = 1$ pixel, and EX = 4 pixels. The results of the proposed method are compared to DISCOVER codec.

Table 4.1: Average PSNR of the SI obtained with the proposed method and the MCTI technique.

SI Average PSNR [dB]						
sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
MCTI	22.57	29.31	24.72	31.43	22.05	35.66
SIG	**23.83**	**29.97**	**27.14**	**32.35**	**22.75**	**36.22**
GOP = 4						
MCTI	21.28	27.58	23.48	29.85	20.81	34.51
SIG	**22.24**	**28.10**	**25.68**	**30.75**	**21.42**	**35.03**
GOP = 8						
MCTI	20.64	26.24	22.53	28.75	20.15	33.69
SIG	**21.47**	**26.69**	**24.61**	**29.59**	**20.70**	**34.04**

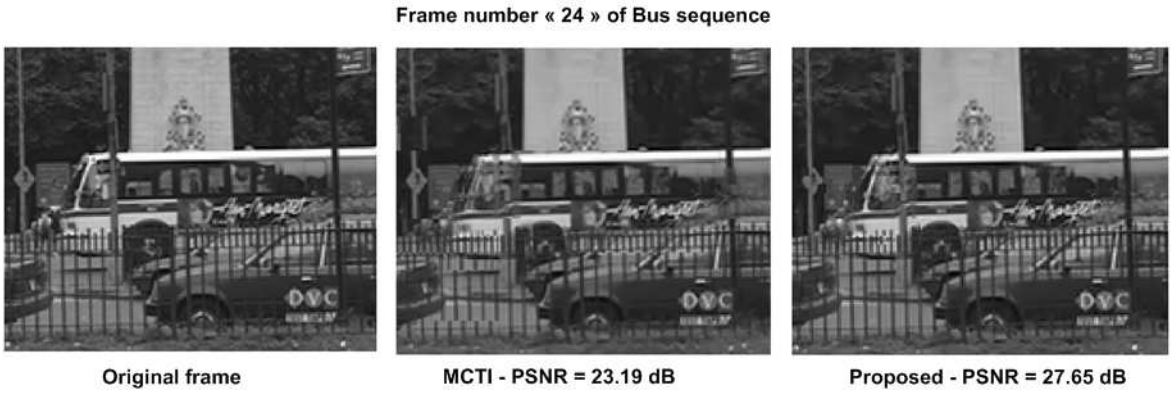

Figure 4.6: Visual result of the SI generated by the proposed method and the MCTI technique, for frame number 24 of Bus sequence, with a GOP size of 2.

SI performance assessment

The quality of the generated SI, estimated in terms of the PSNR obtained with the proposed method and with the MCTI technique, is shown in Fig. 4.5, for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, for a GOP size of 2. As shown in this figure, the proposed method consistently achieves a gain compared to MCTI technique. The gain is considerable in the case of Bus sequence.

The average PSNR of the SI is shown in Table 4.1, for the proposed method and the MCTI technique, for all sequences and different GOP sizes. A significant gain is observed with the proposed method for all test sequences and all GOP sizes. The gain reaches 1.26 dB and 2.42 dB for Stefan and Bus sequences respectively, for a GOP size of 2.

Fig. 4.6 shows the visual result of the SI estimated by the proposed and the MCTI techniques, compared to the original frame, for frame number 24 of Bus sequence, with a GOP size of 2. The SI generated using the MCTI technique contains many block artifacts (PSNR = 23.19 dB), whereas the proposed method presents a much better quality, with a gain of 4.45 dB compared to MCTI.

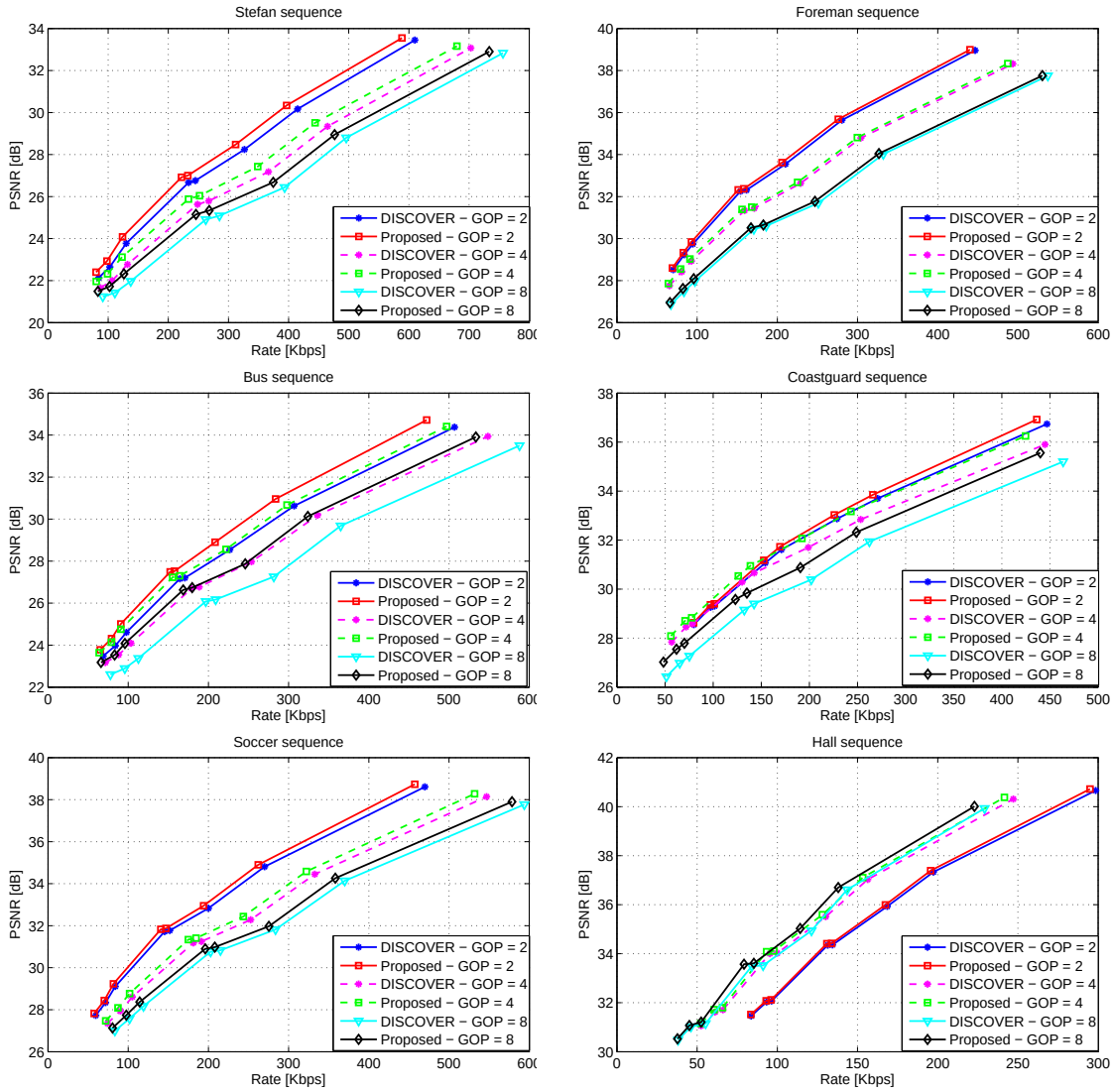

Figure 4.7: RD performance comparison between the proposed method SIG and DISCOVER codec for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, for all GOP sizes.

Rate-Distortion performance

Fig. 4.7 shows the RD performance of the proposed method and DISCOVER codec for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, for all GOP sizes. The proposed method clearly outperforms the MCTI technique.

In Table 4.2, we show the RD performance of the proposed method compared to DISCOVER codec, using the Bjontegaard metric [11]. At the decoder side, the computational complexity is increased with respect to the MCTI technique: in particular, two motion estimations are needed for the first step. This is justified, as the proposed method achieves a significant rate reduction. For instance, for Bus sequence, we reach a PSNR improvement of 1.31 dB and a bit reduction of 22.1% w.r.t MCTI, for a GOP size equal to 8.

Table 4.2: RD performance gain for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences towards DISCOVER codec, using Bjontegaard metric.

sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
Δ_R [%]	-8.32	-3.17	-13.50	-4.22	-4.95	-1.55
Δ_{PSNR} [dB]	0.50	0.17	0.79	0.21	0.27	0.12
GOP = 4						
Δ_R [%]	-9.28	-2.66	-21.16	-11.70	-6.28	-2.76
Δ_{PSNR} [dB]	0.54	0.14	1.24	0.47	0.34	0.19
GOP = 8						
Δ_R [%]	-9.06	-3.58	-22.10	-16.93	-5.67	-6.07
Δ_{PSNR} [dB]	0.54	0.16	1.31	0.71	0.33	0.32

4.1.4 Conclusion

In this part, we propose a new method for SI generation in transform-domain DVC. The proposed technique consists in enhancing the generation of the initial SI using a combination of backward and forward motion estimation. First, the backward and forward motion vectors are computed for large blocks of 32×32 pixels. Then, the blocks are split into four blocks and each new block inherits the estimated motion vector. A quadtree refinement is carried out to select the best candidate motion vector from the neighboring vectors. Afterwards, the same procedure is applied until the size of the block becomes 4×4 pixels. Then, the best motion vector is selected for each 4×4 block. Finally, spatial smoothing and motion compensation are performed to obtain the SI.

The SI generated by the proposed method is always better than the one obtained using the MCTI technique, for all test sequences and all GOP sizes. The gain reaches 1.26 dB and 2.42 dB for Stefan and Bus sequences respectively, for a GOP size of 2. Furthermore, the proposed method allows a gain in RD performance of up to 1.31 dB and a bit reduction of 22.1% compared to DISCOVER codec.

4.2 Adaptive motion search

In this section, we propose a new method that consists in adapting the motion search area using the PDWZF and the reference frames. This adapted search area is then used by the decoder to successively refine the SI by carrying out our refinement technique (called algorithm II) [12], previously described in chapter 2.

4.2.1 Proposed method

The block diagram of our proposed codec architecture is depicted in Fig. 4.8. It is based on the DISCOVER codec [4][5]. The INSI is first computed using the MCTI technique with spatial motion smoothing, exactly as in DISCOVER codec. The LDPC parity bits of the first band (DC band) are used to correct the corresponding DCT coefficients in the INSI and obtain the PDWZF_1.

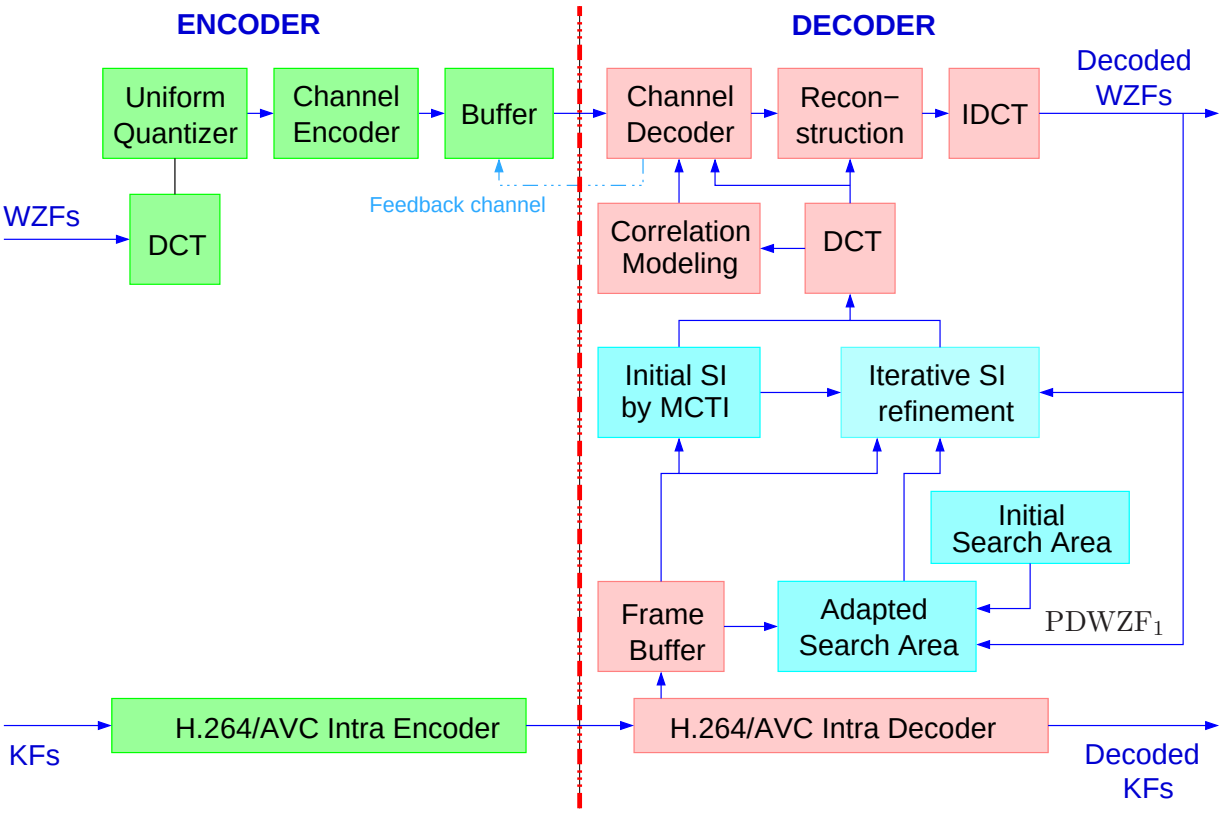

Figure 4.8: Overall structure of the proposed DVC codec

Let d be the distance between the previous and next reference frames. For example, for a GOP size of 8 (sequence of frames KF_1 WZF_1 WZF_2 WZF_3 WZF_4 WZF_5 WZF_6 WZF_7 KF_2), d equals 8 for WZF_4, equals 4 for WZF_2 and WZF_6, and equals 2 for WZF_1, WZF_3, WZF_5, and WZF_7. We first set the initial motion search area according to the distance d. A search area of $RA_4 = \pm 80$ pixels is initially set if d equals 8 (the temporal distance between the current WZF and the previous (or next) reference frame is 4), a search area of $RA_2 = \pm 56$ pixels is set if d equals 4, and a search area of $RA_1 = \pm 32$ pixels is set if d equals 2. These initial search areas are empirically set according to the observed motion in typical test sequences. These large search areas are especially efficient when the current GOP exhibits high motion. However, the computational complexity is significantly increased, which could constitute a useless additional burden in the case of slow motion in the current GOP, without significant enhancement in the SI, compared to a small search area

Therefore, an adaptation of the initial search areas to the current motion level is necessary. For this purpose, N points are selected in the $PDWZF_1$ (using uniform sampling). Fig. 4.9 shows $N = 20$ selected points in the $PDWZF_1$. Each one of those selected points represents the center of a large block of 24×24 pixels. These blocks are used along with the previous and next reference frames in order to adapt the search area. The block matching

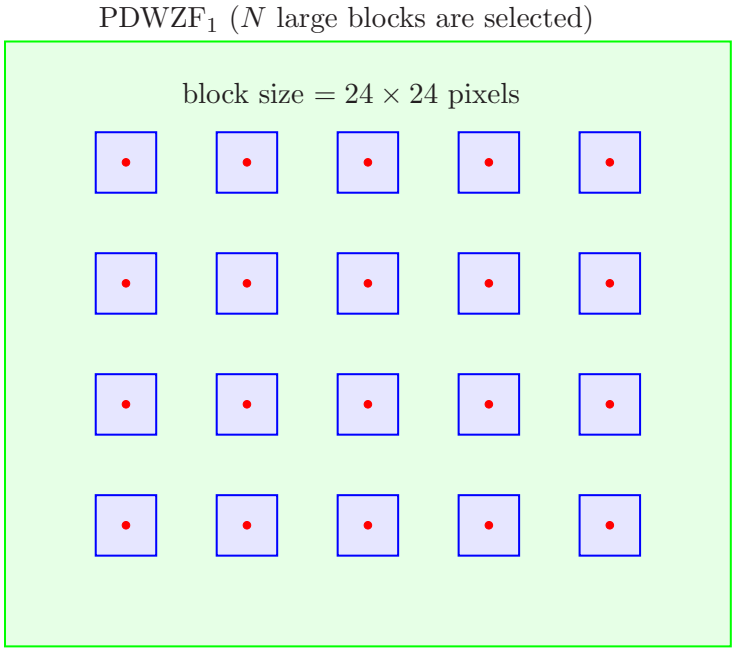

Figure 4.9: The selected points in the PDWZF$_1$ after decoding the first DCT band.

between the PDWZF$_1$ and the previous (and next) reference frame is carried out for those selected blocks in two pixels accuracy, within the search area defined by RA$_i$, $i = 1, 2$, or 4, depending on the distance d between the reference frames . When a selected block belongs to a homogeneous region, the MAD is almost the same for all candidate blocks in this region. In order to avoid obtaining false large motion vectors in these homogeneous areas, the MAD computed during the matching procedure is penalized (MAD$_{pen}$) by the length of the motion vector $\mathbf{m} = (m_x, m_y)$ using:

$$\text{MAD}_{pen} = \text{MAD} \times \left(1 + \lambda\sqrt{m_x^2 + m_y^2}\right), \tag{4.11}$$

where the penalty parameter λ is empirically set to 0.008 if $d = 8$, to 0.012 if $d = 4$ and to 0.02 if $d = 2$, after preliminary tests. This penalty allows to avoid the large estimation errors that can occur with large search areas, when a selected block belongs to a homogeneous region.

The initial search area is adapted in the four directions (left, right, top and bottom) according to the N obtained motion vectors as follows: the maxima of the obtained motion vectors in the four directions are used to adapt the initial search area, as depicted in Fig. 4.10. Let us define the search area by the four parameters SA$_r$, SA$_l$, SA$_t$ and SA$_b$, which represent the distance between the center and the right, left, top, and bottom points, respectively, attained by the search area. The initial search area always has a squared shape, whereas the shape of the adapted search area can be rectangular, depending on the obtained motion vectors. The parameters are adapted according to the N obtained

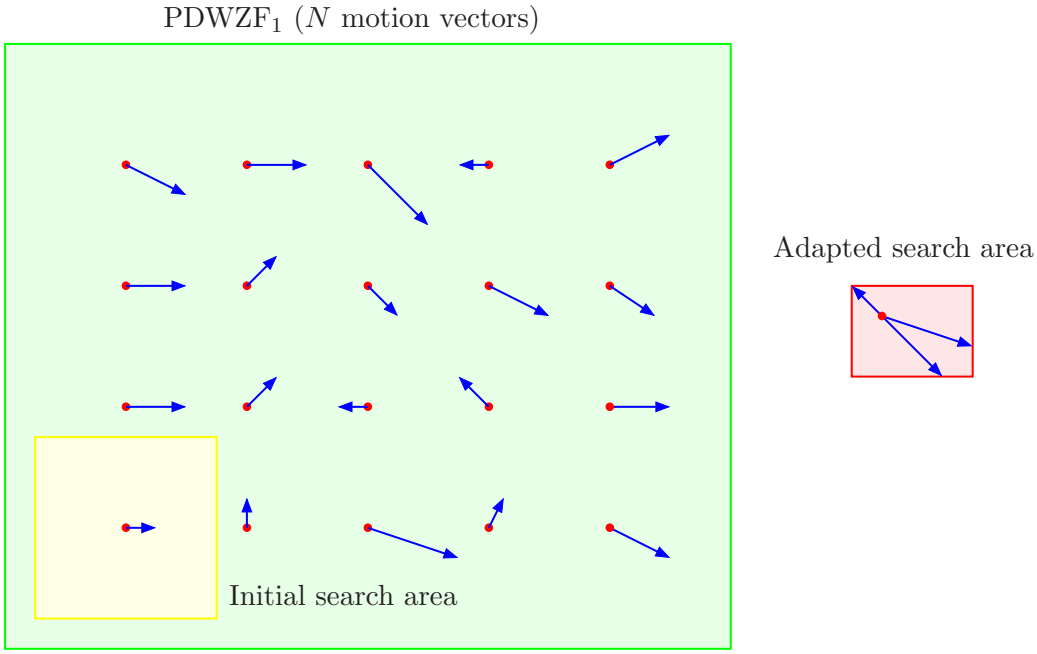

Figure 4.10: The obtained motion vectors used to adapt the search area.

motion vectors $\mathbf{m}_i = (m_{ix}, m_{iy})$ $(i = 1, 2, ..., N)$ as follows:

$$\begin{cases} \text{SA}_r &= \max\limits_i(m_{ix}), \text{ if } m_{ix} > 0 \\ \text{SA}_l &= -\min\limits_i(m_{ix}), \text{ if } m_{ix} < 0 \\ \text{SA}_t &= \max\limits_i(m_{iy}), \text{ if } m_{iy} > 0 \\ \text{SA}_b &= -\min\limits_i(m_{iy}), \text{ if } m_{iy} < 0 \end{cases}$$

The adapted search area is then used to re-estimate the suspicious motion vectors.

As explained in the previous chapter, the suspicious motion vectors are detected by computing the MAD between the last SI (SI_k) and the last PDWZF (PDWZF_k):

$$\text{MAD}(\mathbf{P_0}, \text{PDWZF}_k, \text{SI}_k) < T_1, \tag{4.12}$$

where $\mathbf{P_0}$ represents the top left point for the processed block b. If Eq. (4.12) is satisfied, the corresponding vectors are considered to be true motion vectors. In this case, the motion vectors are refined within a small search area of ± 2 pixels, in half-pixel accuracy. Note that this refinement is only applied two times during the decoding of DCT bands, the first time after decoding first DCT band, and the second one after decoding all DCT bands. Otherwise, the motion vectors are identified as suspicious vectors and will be re-estimated.

The bi-directional motion vectors are re-estimated for those selected blocks using the adapted search area, in two pixels accuracy, with an extended block of $(8 + n) \times (8 + n)$ pixels. Afterwards, the obtained motion vectors are refined within a small search area of

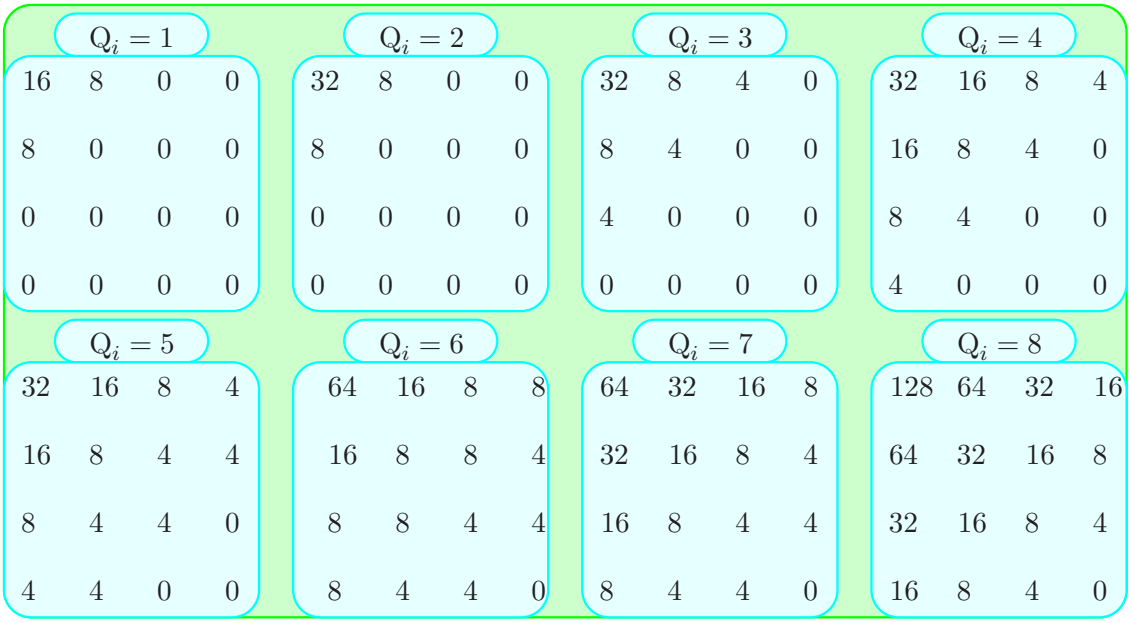

Figure 4.11: Eight 4×4 quantization matrices corresponding to different rate-distortion points.

± 3 pixels, in half-pixel accuracy.

Similarly to the proposed method in Chapter 2, three modes are used to generate a motion-compensated estimate of the new SI: the BRF (BACKWARD MODE), the FRF (FORWARD MODE), and the bi-directional motion-compensated average of the backward and forward reference frames (BIMODE). The decision among these modes is performed according to Eq. 3.7.

Fig. 4.11 shows eight 4×4 quantization matrices corresponding to various rate-distortion points. The value at position k (L_k) within a given 4×4 quantization matrix indicates the number of quantization levels associated with the DCT coefficients band b_k. The DCT coefficients band b_k is uniformly quantized with $L_k = 2^{M_k}$ levels, where M_k decreases with k. Note that the value 0 in the quantization matrix means that no WZ parity bits are transmitted for the corresponding band (for instance, $b_k = 0$ when $k > 4$ in the quantization matrix for $Q_i = 1$). Indeed, we can see that M_k becomes less significant after the first three DCT bands. For this reason, the SI is refined after each decoded DCT band if $k < 4$, and after each pair of decoded DCT bands otherwise.

4.2.2 Experimental results

In order to evaluate the performance of the proposed method, we performed extensive simulations, adopting the same test conditions as described in DISCOVER [4][5], *i.e.* test video sequences are at QCIF spatial resolution and sampled at 15 frames/sec. The obtained results are compared to the DISCOVER codec and to our previous successive refinement technique (Alg. II [12]) from Chapter 2.

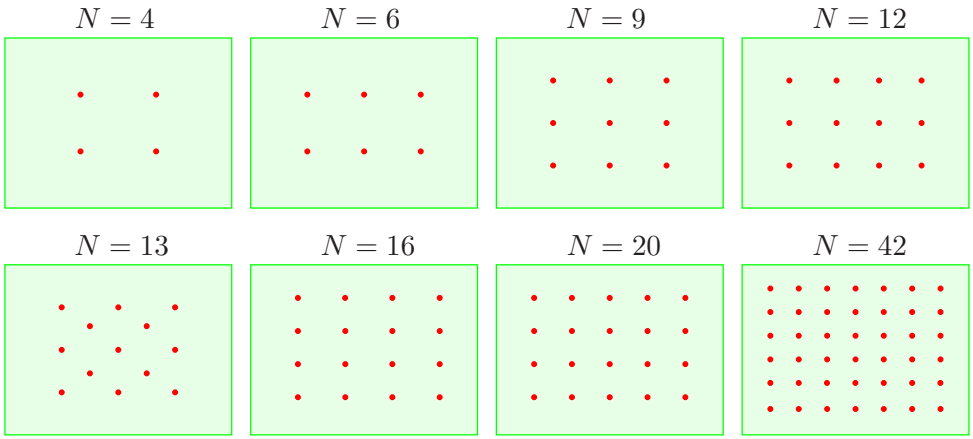

Figure 4.12: Distribution of the selected blocks in the PDWZF_1 after the decoding of the first DCT band, for different values of N ($N = 4, 6, 9, 12, 13, 16, 20$ and 42).

Table 4.3: RD performance gain of the proposed method for different values of N compared to DISCOVER codec using Bjontegaard metric [11], for Foreman and Stefan sequences, for a GOP size of 8.

Proposed < T1 = 4, T2 = 5 and n = 4 >								
	$N = 4$	$N = 6$	$N = 9$	$N = 12$	$N = 13$	$N = 16$	$N = 20$	$N = 42$
GOP = 8								
Stefan								
Δ_R (%)	-42.60	-43.87	-45.14	-45.59	-45.67	-45.69	**-46.04**	**-45.99**
Δ_{PSNR} [dB]	2.86	2.99	3.12	3.15	3.15	3.16	**3.19**	**3.19**
Foreman								
Δ_R (%)	-49.39	-49.83	-49.86	-50.10	-50.00	-50.08	**-50.08**	**-50.11**
Δ_{PSNR} [dB]	3.08	3.12	3.12	3.14	3.14	3.14	**3.14**	**3.15**

Parameter tuning

Fig. 4.12 shows the distribution of the selected blocks in the PDWZF_1 after the decoding of the first DCT band, for several values of N ($N = 4, 6, 9, 12, 13, 16, 20$ and 42), using uniform sampling. We show in Table 4.3 the RD performance gain of the proposed method for each value of N, for the Foreman and Stefan sequences, for a GOP size of 8. It is clear that the RD performance is slightly increased with N. In our simulations, N is set to 20.

The parameter T_1 in Eq. 4.12 has a great impact on the performance of the proposed method, since it determines the computational load and the achieved performance improvement. $T_1 = \infty$ corresponds to the case where no block is considered as erroneous, which is equivalent to the DISCOVER codec. On the other hand, when $T_1 = 0$, all blocks are considered as erroneous and will be refined. As for the parameter T_2, it was set to $T_2 = 5$ in our simulations, after preliminary tests. In fact, the parameter T_2 has a limited influence on the RD performance improvement.

In Table 4.4, the RD performance of the proposed method and Alg. II is shown for different values of T_1 and n, in comparison to DISCOVER codec, using the Bjontegaard metric [11], for a GOP size of 8. In this table, the percentage of the decoding time complexity compared to DISCOVER codec is also shown. It is computed by the ratio (in %)

Table 4.4: Rate-Distortion performance gain and decoding complexity for *Stefan* and *Foreman*, compared to DISCOVER codec, for different values of T_1 and n, for a GOP size of 8.

	Extended block $(8+n)\times(8+n)$ - $T_2=5$					
	$T_1=4$		$T_1=6$		$T_1=9$	$T_1=12$
	$n=0$	$n=4$	$n=0$	$n=4$	$n=4$	$n=4$
	Stefan sequence					
	Proposed					
Δ_R (%)	-36.04	**-41.98**	-35.02	**-40.26**	-38.64	-37.15
Δ_{PSNR} [dB]	2.72	**3.29**	2.65	**3.08**	2.91	2.7
Complexity (%)	79	**85**	76	**77**	77	80
	Alg. II					
Δ_R (%)	-28.6	**-34.13**	-28.22	**-33.97**	-32.38	-30.68
Δ_{PSNR} [dB]	2.07	**2.51**	2.02	**2.42**	2.26	2.12
Complexity (%)	100	**132**	91	**101**	97	96
	Foreman sequence					
	Proposed					
Δ_R (%)	-40.49	**-43.55**	-38.79	**-42.27**	-40.61	-39.4
Δ_{PSNR} [dB]	2.96	**3.19**	2.71	**2.99**	2.81	2.67
Complexity (%)	69	**75**	71	**74**	76	78
	Alg. II					
Δ_R (%)	-40.49	**-41.88**	-37.03	**-40.6**	-39.14	-37.89
Δ_{PSNR} [dB]	2.74	**3.02**	2.58	**2.87**	2.72	2.59
Complexity (%)	75	**89**	71	**78**	76	77

between the computational load of our technique and the one required by the DISCOVER codec as follows:

$$\text{Complexity (\%)} = 100 \times \frac{\text{Decoding time required in the proposed technique}}{\text{Decoding time required in DISCOVER}} \tag{4.13}$$

It is clear that for $T_1 = 4$ and $n = 4$, the proposed method and Alg. II achieve the best RD performance. For Stefan, the proposed method can reduce the decoding time by 15% compared to DISCOVER codec. On the contrary, the decoding time is increased by 32% for Alg. II, due to the high motion in this sequence. As we can see, the proposed method reduces the computational load for all values of T_1, due to the appropriate adaptation of the search area to the motion level.

In the case of the Foreman sequence, Alg. II achieves a significant gain compared to DISCOVER codec, and the decoding time is reduced. On the other hand, with the proposed method, the gain becomes more effective and the decoding time is further reduced. In the remaining simulations, we have set $T_1 = 4$ and $n = 4$ since these values allowed very good performances and a low computational burden. In fact, the search area size being sometimes increased compared to ± 16 pixels and the block size being extended to 12×12, the improvement of the SI not only significantly reduces the amount of requested parity bits through the feedback channel, but it also reduces the decoder processing time, by decreasing the number of necessary decoding runs in the iterative receiver.

SI performance assessment

Figs. 4.13 and 4.14 show the visual results of the original frame, the SI estimated by MCTI (DISCOVER), the final SI estimated by Alg. II after decoding all DCT bands, and the final

Figure 4.13: Visual result comparisons among the original frame (top-left), the SI estimated by the MCTI technique (top-right), the final SI estimated by Alg. II (bottom-left), and the final SI estimated by the proposed method (bottom-right), for frame number 95 of Foreman sequence, for a GOP size of 8 (QI = 8).

SI estimated by the proposed method, for frame number 95 of Foreman and frame number 115 of Stefan, for a GOP equal to 8. The SI frame obtained by MCTI contains block artifacts. On the contrary, the SI frames obtained by Alg. II and the proposed method present a much better quality.

For frame number 95 of Foreman (Figure 4.13), the proposed method allows an improvement up to 11 dB compared to MCTI technique, and an improvement up to 2.76 dB compared to the final SI estimated in Alg. II. For the final decoded WZFs of these SI frames, the proposed method achieves a gain up to 2 dB compared to DISCOVER codec, with less requested bits, down from 46.39 Kbits to 36.22 Kbits, and a gain up 0.67 dB compared to Alg. II, with a bit rate decrease from 40.30 Kbits to 36.22 Kbits.

For frame number 115 of Stefan (Figure 4.14), the final SI obtained by the proposed method allows a gain up to 8.25 dB compared to MCTI technique, and an improvement of up to 5.26 dB compared to Alg. II. For the final decoded WZFs of these SI frames, the proposed method achieves a gain up to 2.41 dB compared to DISCOVER codec, with a

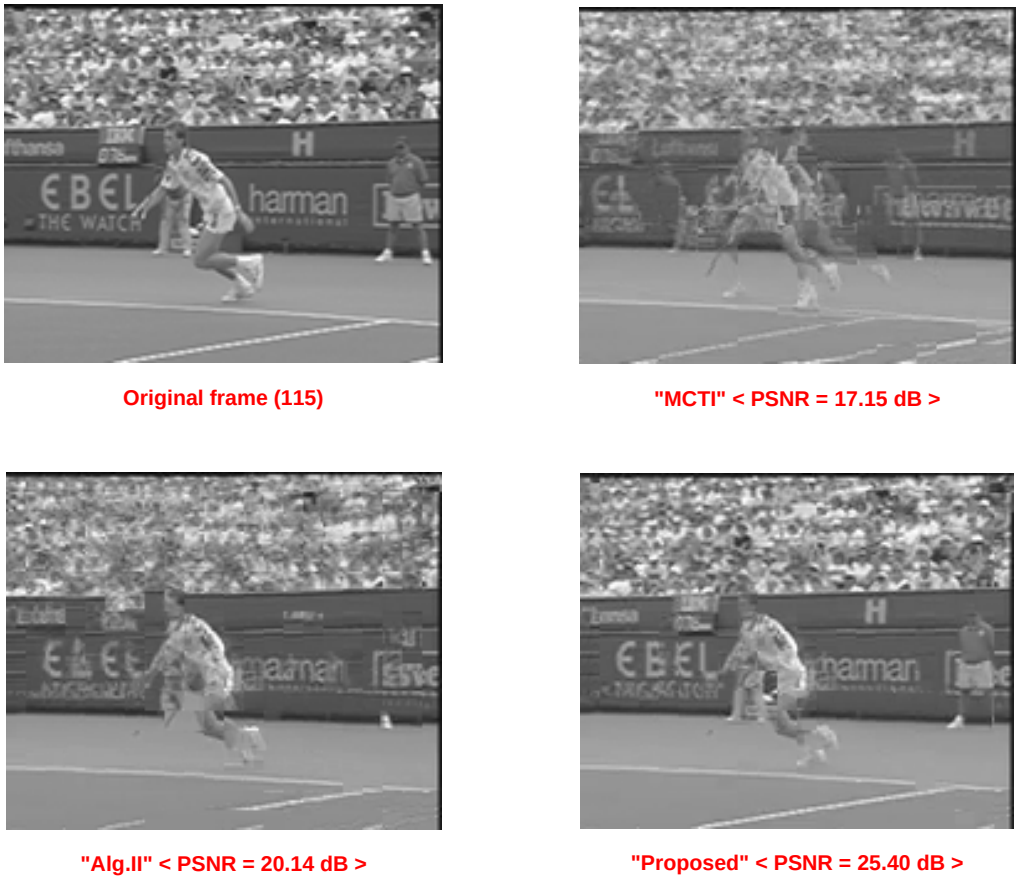

Figure 4.14: Visual result comparisons among the original frame (top-left), the SI estimated by the MCTI technique (top-right), the final SI estimated in Alg. II (bottom-left), and the final SI estimated by the proposed method (bottom-right), for frame number 115 of Stefan sequence, for a GOP size of 8 (QI = 8).

bit rate decrease from 61.75 Kbits to 47.93 Kbits, and a gain up 2.12 dB compared to Alg. II, with a bit rate decrease from 59.75 Kbits to 47.93 Kbits.

Rate-Distortion performance

The RD performance of the proposed method is shown for the Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences in Table 4.5, in comparison to the DISCOVER codec, using the Bjontegaard metric [11] for different GOP sizes (2, 4 and 8). The first row represents the performance of our previous technique (Alg. II), *i.e.*, a constant search area of ± 16 pixels is used regardless of the distance between the reference frames. It is clear that our proposed method achieves a significant gain compared to DISCOVER codec, especially for sequences containing high motion such as Stefan and Foreman sequences.

For Stefan sequence with a GOP size of 8, Alg. II can achieve a gain up 2.51 dB with a rate reduction up to 34.13 % compared to DISCOVER codec. The proposed method

Table 4.5: Rate-Distortion performance gain for *Stefan*, *Foreman*, *Bus*, *Coastguard*, *Soccer* and *Hall* sequences compared to DISCOVER codec, using Bjontegaard metric [11].

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 2						
Alg. II [12]						
Δ_R (%)	-14.06	-16.29	-4.50	-2.24	-17.65	-1.34
Δ_{PSNR} [dB]	0.93	1.05	0.27	0.11	1.05	0.10
Proposed						
Δ_R (%)	**-17.31**	**-16.53**	**-4.91**	**-2.28**	**-18.96**	**-1.36**
Δ_{PSNR} [dB]	**1.16**	**1.07**	**0.30**	**0.11**	**1.14**	**0.10**
GOP = 4						
Alg. II [12]						
Δ_R (%)	-27.84	-32.65	-15.82	-11.94	-25.08	-4.24
Δ_{PSNR} [dB]	1.93	2.19	0.99	0.52	1.61	0.27
Proposed						
Δ_R (%)	**-34.44**	**-33.88**	**-16.42**	**-12.15**	**-27.31**	**-4.27**
Δ_{PSNR} [dB]	**2.51**	**2.30**	**1.03**	**0.53**	**1.78**	**0.27**
GOP = 8						
Alg. II [12]						
Δ_R (%)	-34.13	-41.88	-22.83	-24.21	-28.16	-11.04
Δ_{PSNR} [dB]	2.51	3.02	1.53	1.14	1.88	0.68
Proposed						
Δ_R (%)	**-41.98**	**-43.55**	**-26.13**	**-24.40**	**-31.43**	**-11.22**
Δ_{PSNR} [dB]	**3.29**	**3.19**	**1.78**	**1.15**	**2.15**	**0.68**

allows a significant gain up to 3.29 dB with a rate reduction of 41.98% w.r.t. DISCOVER codec. For the other sequences, the proposed method achieves a smaller, but still notable, improvement compared to Alg. II, while the decoding time is significantly reduced, even for sequences containing slow motion, since the search area is adapted to the current motion in the sequence.

Figs. 4.15 and 4.16 show the RD performance curves of the DISCOVER codec, the Alg. II, the proposed method, H.264/AVC Intra, and H.264/AVC No motion, for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, for GOP sizes of 2 and 8. The proposed method can achieve a slight gain compared to H.264/AVC No motion for Stefan, Foreman, Bus and Coastguard sequences. In addition, it can reduce the gap with H.264/AVC No motion for Soccer and Hall sequences.

4.2.3 Conclusion

Successive refinement of the SI using an adaptive motion search area is described in this section, based on the sequential decoding of the DCT bands. The partially decoded frame after decoding the first DCT band is used to adapt the initial motion search area. The adapted search area is then used to progressively refine the SI, along with the previous and next reference frames, after each decoded DCT band.

Experimental results showed that our proposed method can achieve a gain in RD performance of up to 0.78 dB for a GOP size of 8, compared to Alg. II, and 3.29 dB compared to DISCOVER codec, especially when the video sequence contains high motion. The proposed method allows an improvement in the final SI that can reach 5.6 dB, compared to

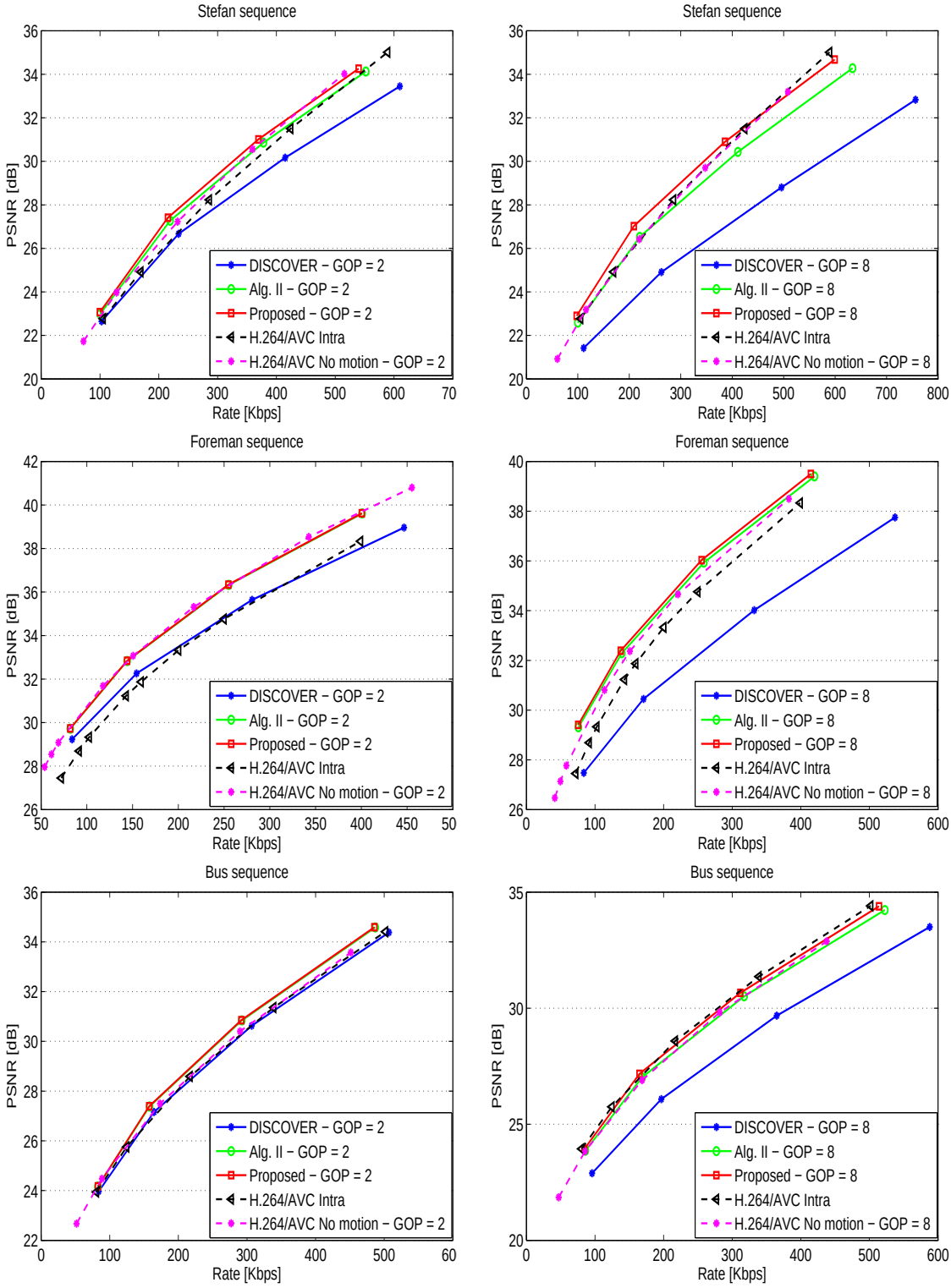

Figure 4.15: RD performance comparison among the DISCOVER codec, the Alg. II, the proposed method, H.264/AVC Intra, and H.264/AVC No motion, for Stefan, Foreman and Bus sequences, for GOP sizes of 2 and 8.

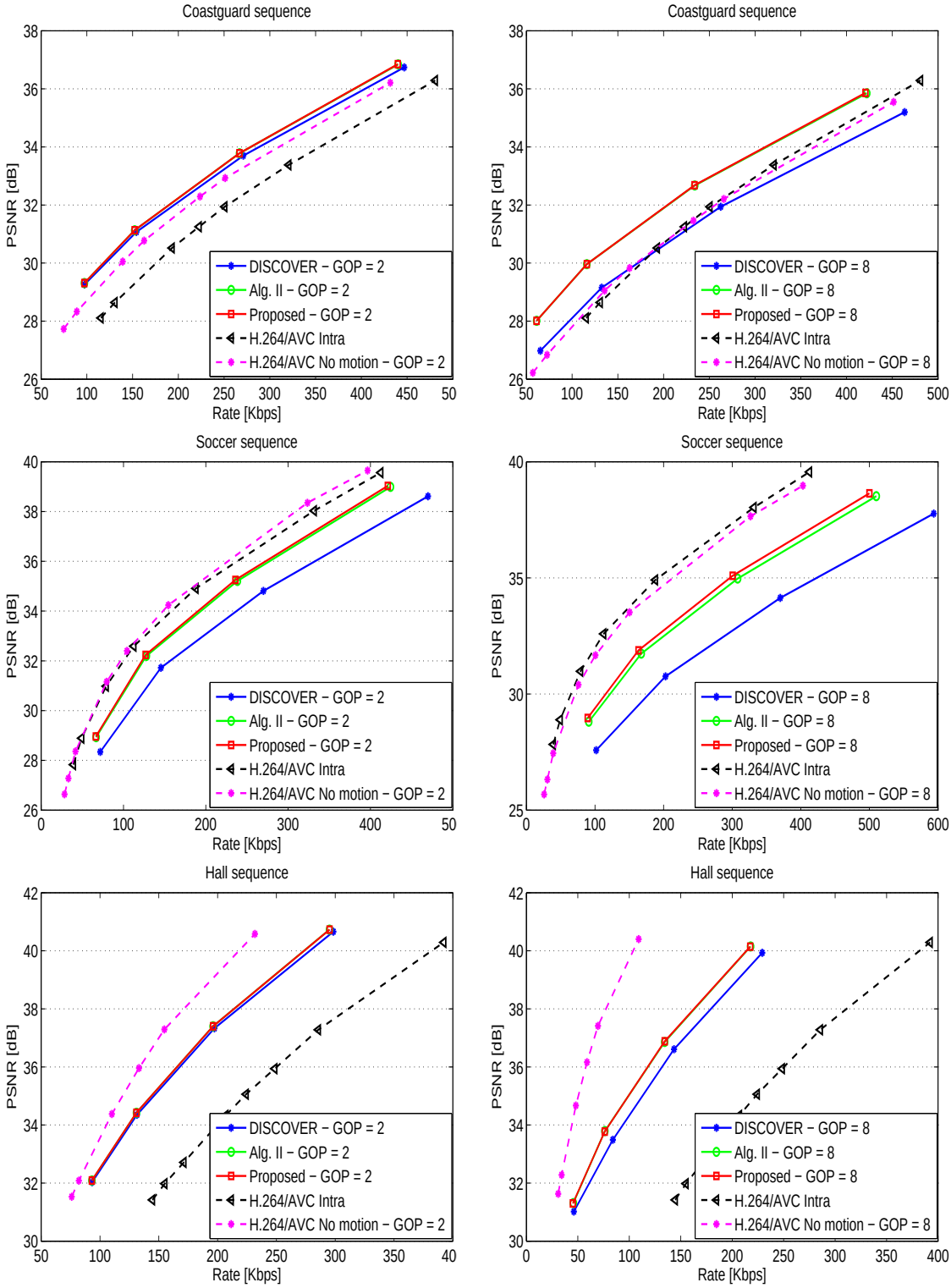

Figure 4.16: RD performance comparison among the DISCOVER codec, the Alg. II, the proposed method, H.264/AVC Intra, and H.264/AVC No motion, for Coastguard, Soccer and Hall sequences, for GOP sizes of 2 and 8.

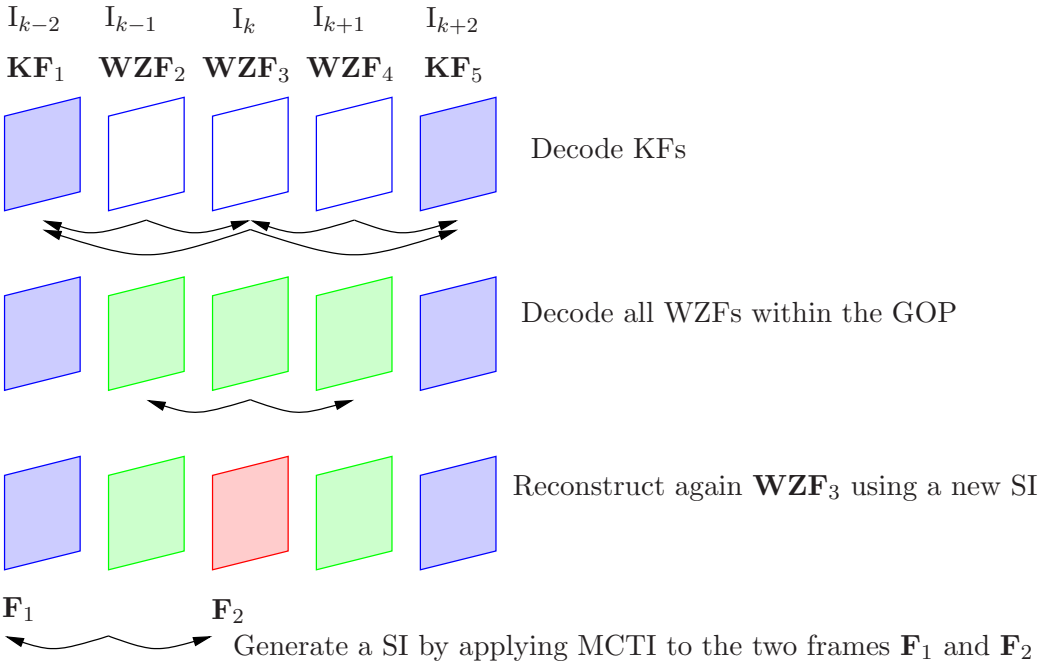

Figure 4.17: WZF estimation for a GOP size of 4.

Alg. II, for some frames. Moreover, the decoding time is significantly reduced by using the adaptive search area.

4.3 Side information re-estimation for long GOP

In this section, we first describe the related work for SI enhancement in the case of large GOP sizes, and then present our proposed method in this context. Finally, we illustrate the experimental results and draw conclusions.

4.3.1 SI construction for large GOP sizes

Let I_k be the WZF that we want to estimate and $\hat{I}_k$ the decoded WZF. The decoding procedure is simple when the GOP size is equal to 2. The frames used for the interpolation are the decoded KFs $\hat{I}_{k-1}$ and $\hat{I}_{k+1}$. Then, this estimation is used to obtain the decoded frame $\hat{I}_k$.

If the GOP size is equal to 4, the WZF I_k is usually estimated using the decoded KFs $\hat{I}_{k-2}$ and $\hat{I}_{k+2}$ (Fig. 4.17). Then, this estimation is used in the turbo-decoder along with the parity bits, thus producing the decoded WZF $\hat{I}_k$. The WZF I_{k-1} is estimated using the decoded frames $\hat{I}_{k-2}$ and $\hat{I}_k$, and I_{k+1} is interpolated using the decoded frames $\hat{I}_k$ and $\hat{I}_{k+2}$.

As a consequence, the quality of the decoded WZFs $\hat{I}_{k-1}$ and $\hat{I}_{k+1}$ is better than the quality of the WZF $\hat{I}_k$. In other words, the PSNR of the decoded WZFs varies within

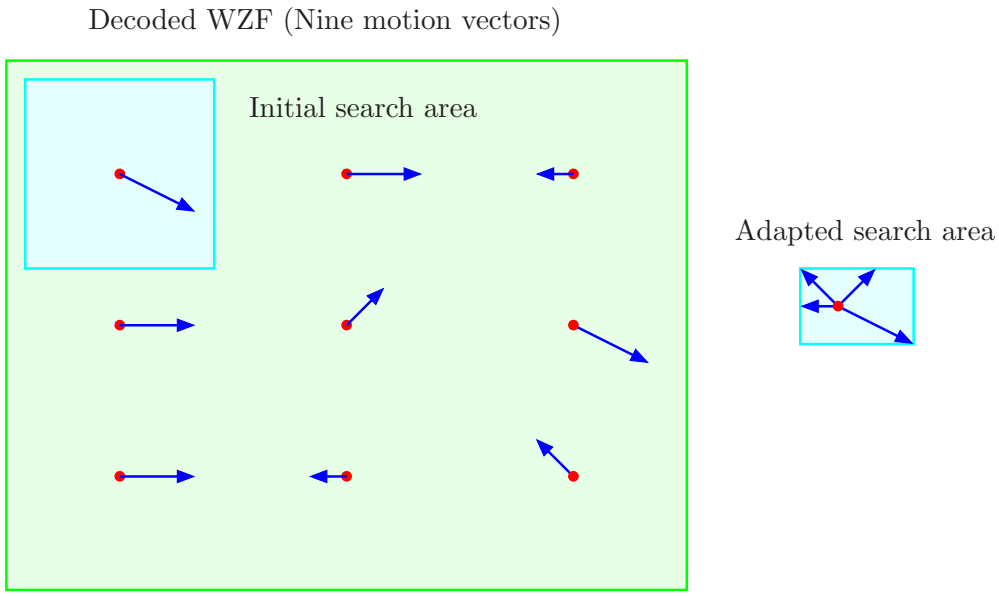

Figure 4.18: The obtained motion vectors used to adapt the search area.

the GOP, depending on the distance between the reference frames (considered for the interpolation of each WZF). The authors in [13] propose to add another step in order to improve the quality of the decoded frame $\hat{I}_k$: Once I_{k-1} and I_{k+1} have been decoded, the frame I_k can be re-estimated by applying the MCTI technique between $\hat{I}_{k-1}$ and $\hat{I}_{k+1}$ (Fig. 4.17). Then, the turbo-decoding and the reconstruction are applied again using the new SI to obtain the final decoded WZF, without requesting any additional parity bits from the encoder. This procedure can be extended for larger GOP sizes. However, in [13], the available decoded frame $\hat{I}_k$ is not used in the re-estimation of the new SI. For this purpose, we propose in Section 4.3.2 to modify this technique in such a way to profit from the already decoded frame $\hat{I}_k$, along with the decoded frames $\hat{I}_{k-1}$ and $\hat{I}_{k+1}$, in the re-estimation of the new SI. The same procedure is also applied for the remaining WZFs.

4.3.2 Proposed method for SI re-estimation

We now propose a new method that consists in improving the decoded WZFs, already obtained by DISCOVER or by our previous techniques Alg. I or Alg. II, when the GOP size is larger than 2. This improvement is achieved by re-estimating the SI using the neighboring decoded frames, with an adaptive search area and a variable block size

In the method proposed by Petrazzuoli et al. [13], the central WZF I_k is re-estimated without using the available decoded frame $\hat{I}_k$. In the proposed method, we re-estimate I_k using the already decoded WZF $\hat{I}_k$, along with the neighboring decoded frames. In particular, we propose an approach to adapt the search area to the real motion between the decoded WZF $\hat{I}_k$ and the previous (and next) decoded frame. This procedure achieves

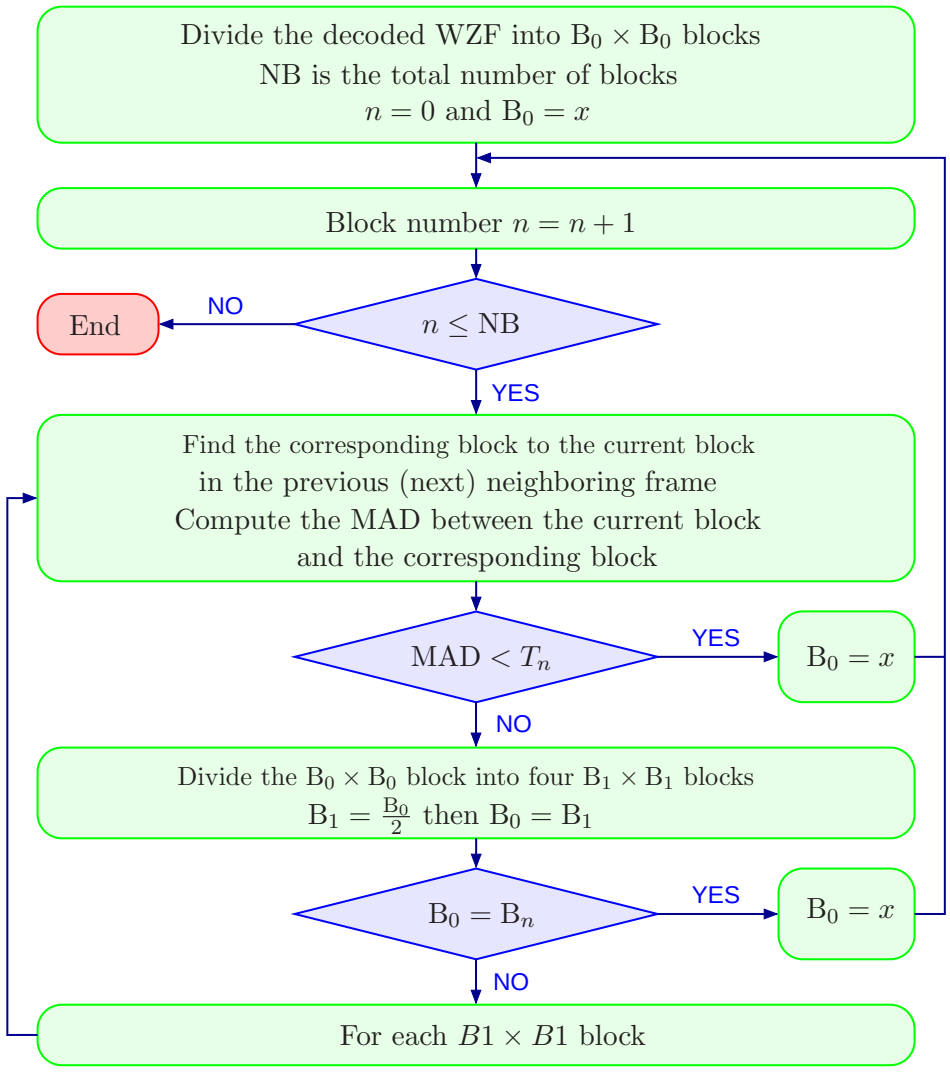

Figure 4.19: Proposed algorithm for the estimation of motion vectors between the decoded WZF and the adjacent frames.

a better estimation for high motion regions. Moreover, it generates more homogeneous motion vectors and reduces the estimation complexity for slow motion regions.

First, N large blocks are selected (using uniform sampling) in the decoded WZF. In the simulations, we empirically set $N = 9$. Matching is then performed for those blocks in order to determine their corresponding blocks in the previous (or next) decoded frame. The same approach already described in the previous section is used to adapt the initial search area (Fig. 4.18), with the difference that in the current refinement technique, the distance between the reference frames is always 2. The initial search area is set to ± 32 pixels and the penalty factor is empirically set to 0.2. Then, the search area is adapted between the current WZF and the previous (and next) decoded frame. The adapted search area will be used to find the motion vectors using a variable block size.

For this purpose, we start by dividing the decoded WZF into $B_0 \times B_0$ blocks. Then, block

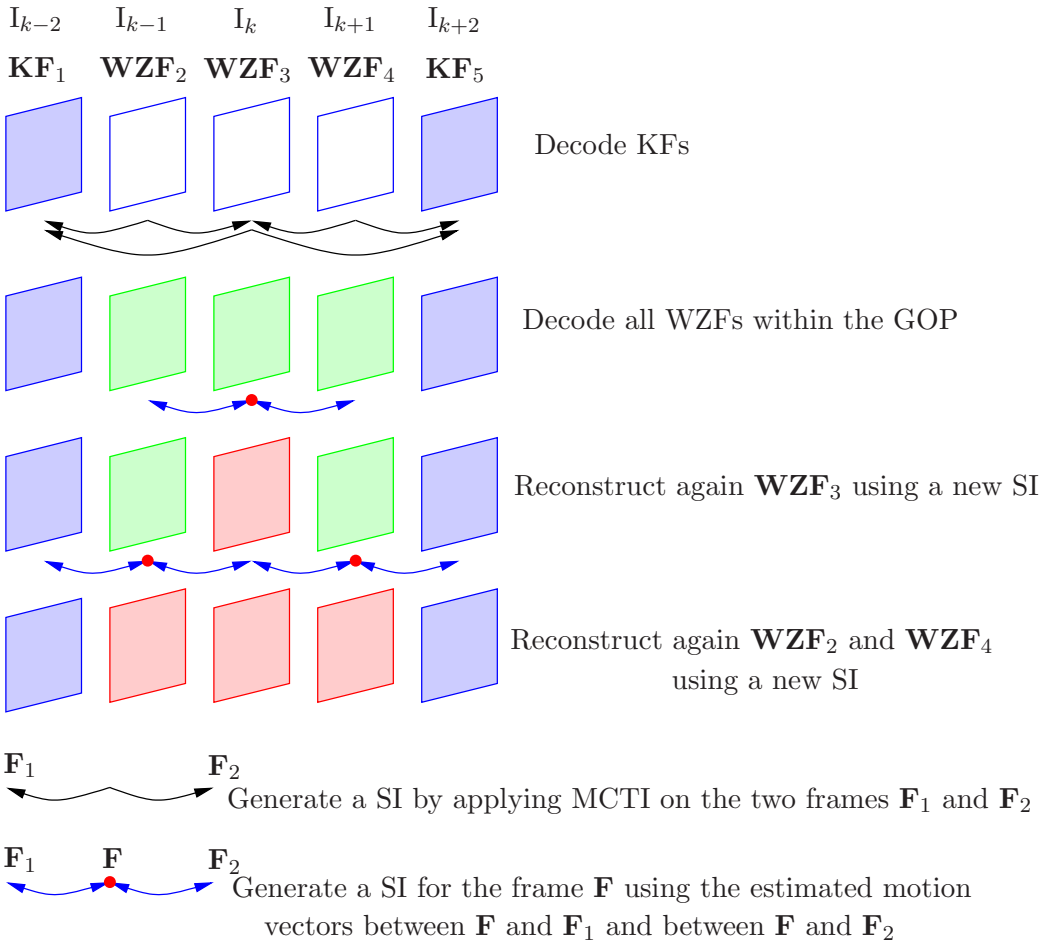

Figure 4.20: Proposed WZF estimation for GOP size = 4.

matching is carried out between the decoded WZF and the previous (and next) decoded frame, using the adapted search area, in order to find the corresponding blocks. The MAD is computed between the current block in the decoded WZF and the corresponding block in the previous (and next) decoded frame. If the obtained MAD is greater than a threshold T_n, this block is split into four $B_1 \times B_1$ blocks ($B_1 = \frac{B_0}{2}$). Then, the motion vectors are estimated for these four $B_1 \times B_1$ blocks using the adapted search area in the same way. The same procedure (Fig. 4.19) is repeated until the block size becomes $B_n \times B_n$ pixels.

Subsequently, the bidirectional motion compensation (using the previous and next decoded frames) is applied to obtain the new SI. Finally, the reconstruction is performed again using the decoded coefficients and the new SI to obtain the final decoded WZF. The proposed method is used to re-estimate a new SI for all WZFs within the current GOP, as shown in Fig. 4.20. For a GOP size of 4, the algorithm starts by re-estimating a new SI for WZF_3 using the previous decoded frame WZF_2 ($\hat{I}_{k-1}$), the available decoded frame WZF_3 ($\hat{I}_k$), and the next decoded frame WZF_4 ($\hat{I}_{k+1}$). Then, the frame WZF_3 is reconstructed using the improved SI. Similarly, the decoded key frame KF_1 ($\hat{I}_{k-2}$), the

Average PSNR of the Final SI [dB]						
Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 4						
MCTI	21.44	27.64	24.00	29.91	20.87	34.64
Petrazzuoli et al.	22.71	29.43	25.53	31.46	22.13	35.96
PropA	**28.79**	**35.02**	**28.66**	**32.39**	**31.36**	**37.77**
Alg. II	25.97	33.63	27.33	31.49	28.66	36.10
PropB	**28.94**	**35.17**	**28.72**	**32.45**	**31.39**	**37.78**
GOP = 8						
MCTI	20.78	26.29	22.95	28.82	20.20	33.68
Petrazzuoli et al.	22.73	29.19	25.43	31.28	22.10	35.76
PropA	**28.89**	**34.86**	**28.66**	**32.26**	**31.28**	**37.57**
Alg. II	25.63	32.93	26.69	30.88	28.19	35.49
PropB	**29.12**	**35.14**	**28.75**	**32.41**	**31.32**	**37.64**

Table 4.6: Average PSNR of the Final SI for GOP sizes equal to 4 and 8 (QI = 8).

already reconstructed frame WZF_3, and the decoded frame WZF_2 are used to generate a new SI for the frame WZF_2. Then, the reconstruction is carried out again to obtain the final decoded frame WZF_2. The same procedure is applied for the WZF_4. As a result, all WZFs within the GOP are improved by applying the proposed approach, without the necessity for a new decoding run (as was done in [13]), and without demanding any additional bits from the encoder.

4.3.3 Experimental results

The proposed SI re-estimation method is applied in two different contexts. When the enhancement is applied on the decoded WZFs obtained by DISCOVER codec, the method is called PropA, and PropB when the improvement is carried out on the decoded WZFs obtained by our previous algorithm Alg. II. Both methods are compared w.r.t. DISCOVER codec. Also, the results obtained by Petrazzuoli et al.[13] and by Alg. II are shown in order to have a complete comparative analysis of the methods.

SI performance assessment

Table 4.6 shows the average PSNR of the final SI, obtained after the refinement process, with the different methods (including the non-refined MTCI). Our proposed techniques lead to a significant improvement in the SI quality for all test sequences. Both proposed PropA and PropB methods show very significant gains compared to DISCOVER, to the method proposed by Petrazzuoli et al. [13], and to Alg. II.

Rate-Distortion performance

Fig. 4.21 shows the PSNR of the decoded frames of Stefan sequence for DISCOVER codec, the method proposed by Petrazzuoli et al. [13], and the proposed technique PropA. We can see that the proposed method achieves a significant gain compared to both DISCOVER and Petrazzuoli et al. [13].

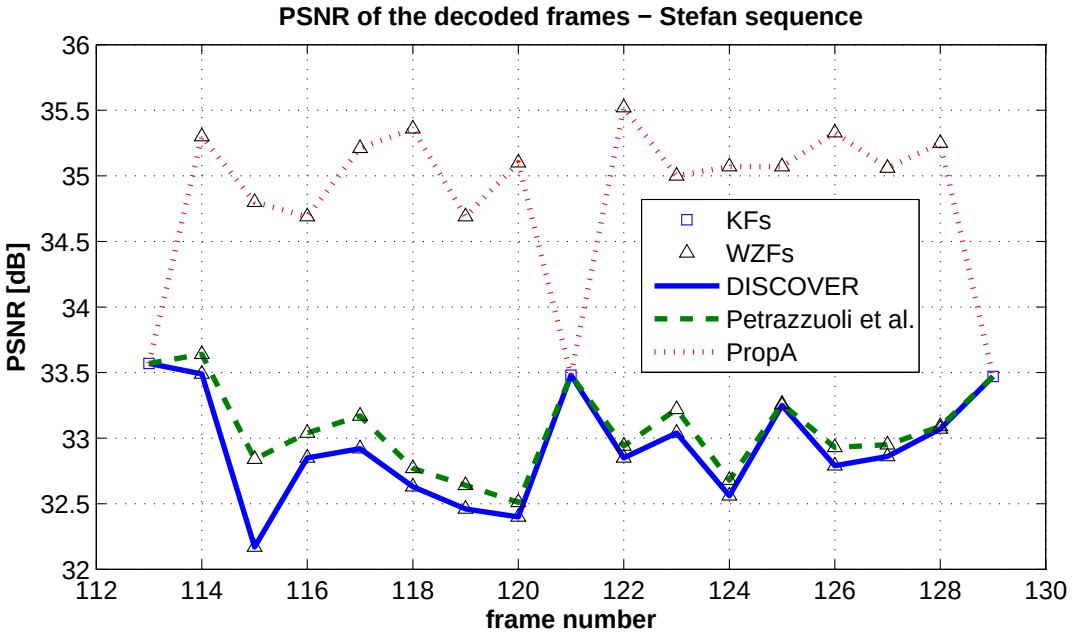

Figure 4.21: PSNR of the decoded frames for two GOPs beginning at frame number 113, from the *Stefan* sequence, for a GOP size of 8.

Table 4.7: Rate-Distortion performance comparison for a GOP size equal to 4 and 8, w.r.t. DISCOVER codec, using Bjontegaard metric.

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**	**Soccer**	**Hall**
GOP = 4						
Petrazzuoli et al. [13]						
Δ_R (%)	-4.49	-5.77	-5.65	-4.29	-3.20	-0.84
Δ_{PSNR} [dB]	0.23	0.28	0.30	0.18	0.15	0.04
PropA						
Δ_R (%)	**-17.42**	**-21.31**	**-8.79**	**-7.25**	**-18.45**	**-3.96**
Δ_{PSNR} [dB]	**1.00**	**1.07**	**0.43**	**0.30**	**0.88**	**0.23**
Alg. II						
Δ_R (%)	-30.35	-36.96	-17.90	-12.25	-28.12	-4.92
Δ_{PSNR} [dB]	1.91	2.19	1.00	0.52	1.59	0.29
PropB						
Δ_R (%)	**-35.53**	**-39.88**	**-19.70**	**-14.78**	**-32.69**	**-6.17**
Δ_{PSNR} [dB]	**2.30**	**2.39**	**1.09**	**0.63**	**1.85**	**0.37**
GOP = 8						
Petrazzuoli et al. [13]						
Δ_R (%)	-8.30	-12.02	-11.56	-9.38	-6.53	-2.33
Δ_{PSNR} [dB]	0.45	0.56	0.60	0.38	0.28	0.09
PropA						
Δ_R (%)	**-22.16**	**-28.01**	**-15.26**	**-13.92**	**-23.49**	**-6.91**
Δ_{PSNR} [dB]	**1.27**	**1.40**	**0.74**	**0.56**	**1.10**	**0.32**
Alg. II						
Δ_R (%)	-37.80	-48.18	-26.65	-26.90	-32.82	-13.00
Δ_{PSNR} [dB]	2.46	2.98	1.53	1.16	1.86	0.71
PropB						
Δ_R (%)	**-44.98**	**-53.03**	**-31.44**	**-31.99**	**-39.51**	**-16.10**
Δ_{PSNR} [dB]	**3.07**	**3.38**	**1.83**	**1.40**	**2.26**	**0.87**

The RD performance is shown for the Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences in Table 4.7, in comparison to the DISCOVER codec, using the Bjontegaard

metric [11], for a GOP size equal to 4 and 8. We represent the performance of the method proposed by Petrazzuoli et al. [13], the proposed method PropA, the Alg. II and the proposed method PropB.

The proposed method PropA, as well as the technique by Petrazzuoli et al. [13], consist in a refinement of the frames decoded by DISCOVER, while PropB is a refinement of the frames obtained by Alg. II. It can be observed that PropA and PropB always achieve a gain w.r.t. the previous works [12, 13] especially for sequences containing high motion.

The gains become even more significant for a GOP size equal to 8. In fact, for PropA, we obtain a bit reduction up to -28.01%, which corresponds to an improvement of 1.4 dB on the decoded frames w.r.t. DISCOVER codec. The proposed method PropB allows a significant gain of up to 3.38 dB, with a rate reduction of 53.03%, compared to the DISCOVER codec. These maxima of performance gain are obtained for the Foreman sequence.

Figs. 4.22 and 4.23 show the RD performance of the DISCOVER codec, the Alg. II, PropB, H.264/AVC Intra, and H.264/AVC No motion, for Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences, for GOP sizes of 4 and 8. The performance of the proposed method PropB is superior to that of H.264/AVC Intra for all test sequences, except for the Soccer sequence with a GOP size of 4, and Bus and Soccer sequences with a GOP size of 8. The proposed method can outperform H.264/AVC No motion for Coastguard and Foreman sequences.

4.3.4 Conclusion

In this section, we proposed a new approach for SI refinement in the case of long duration GOPs. Each decoded WZF is used to adapt the initial motion search area, along with the adjacent decoded frames. The adapted search area is then used to re-estimate the SI, along with the previous and next decoded frames, with a variable block size. Experimental results show that our proposed method can achieve a gain in RD performance of up to 0.61 dB for a GOP size of 8, compared to the Alg. II, 2.8 dB compared to the one proposed by Petrazzuoli et al. [13], and 3.57 dB compared to DISCOVER codec, especially when the video sequence contains high motion.

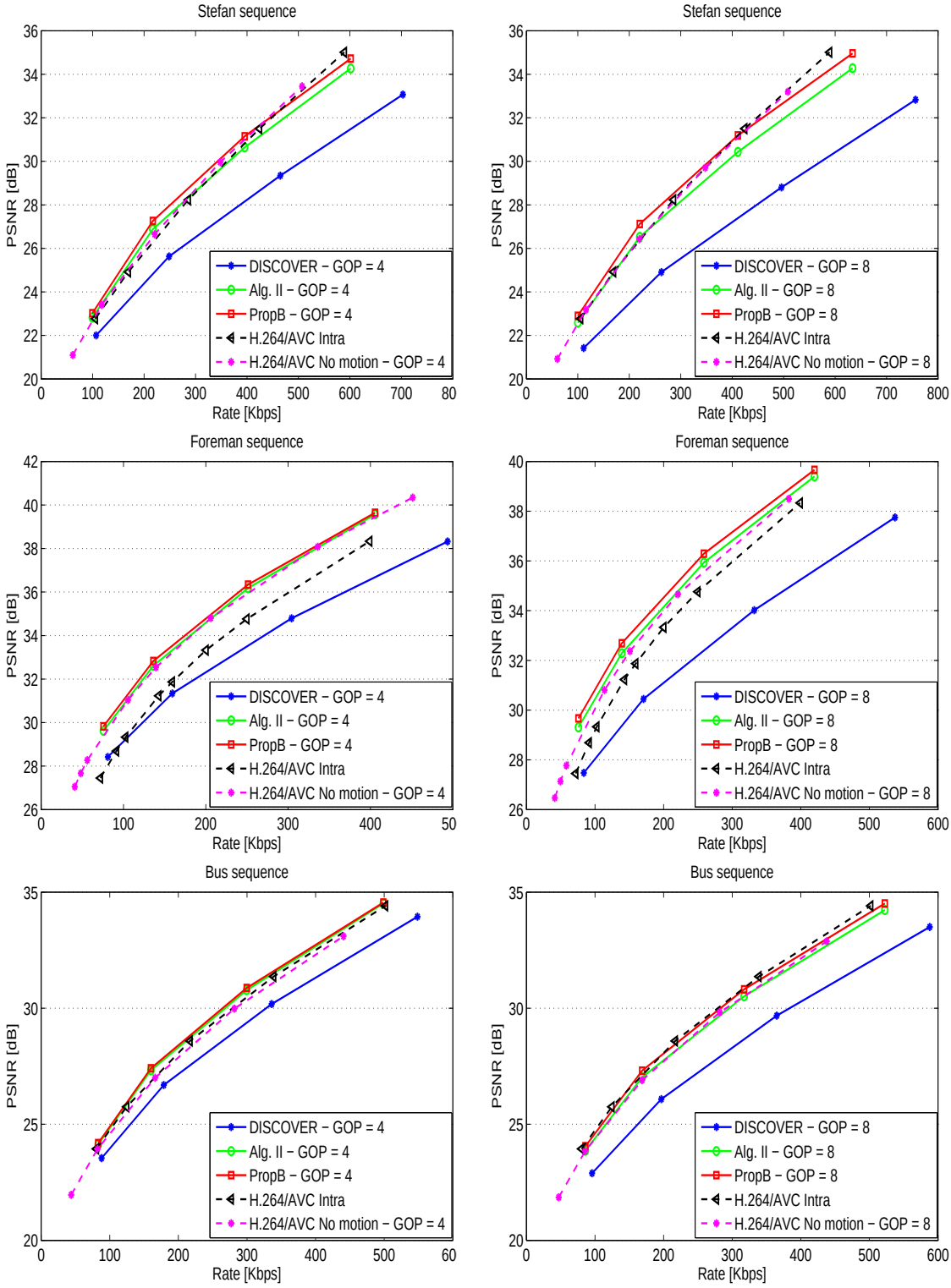

Figure 4.22: RD performance comparison among the DISCOVER codec, the Alg. II, the proposed method, H.264/AVC Intra, and H.264/AVC No motion, for Stefan, Foreman and Bus sequences, for GOP sizes of 4 and 8.

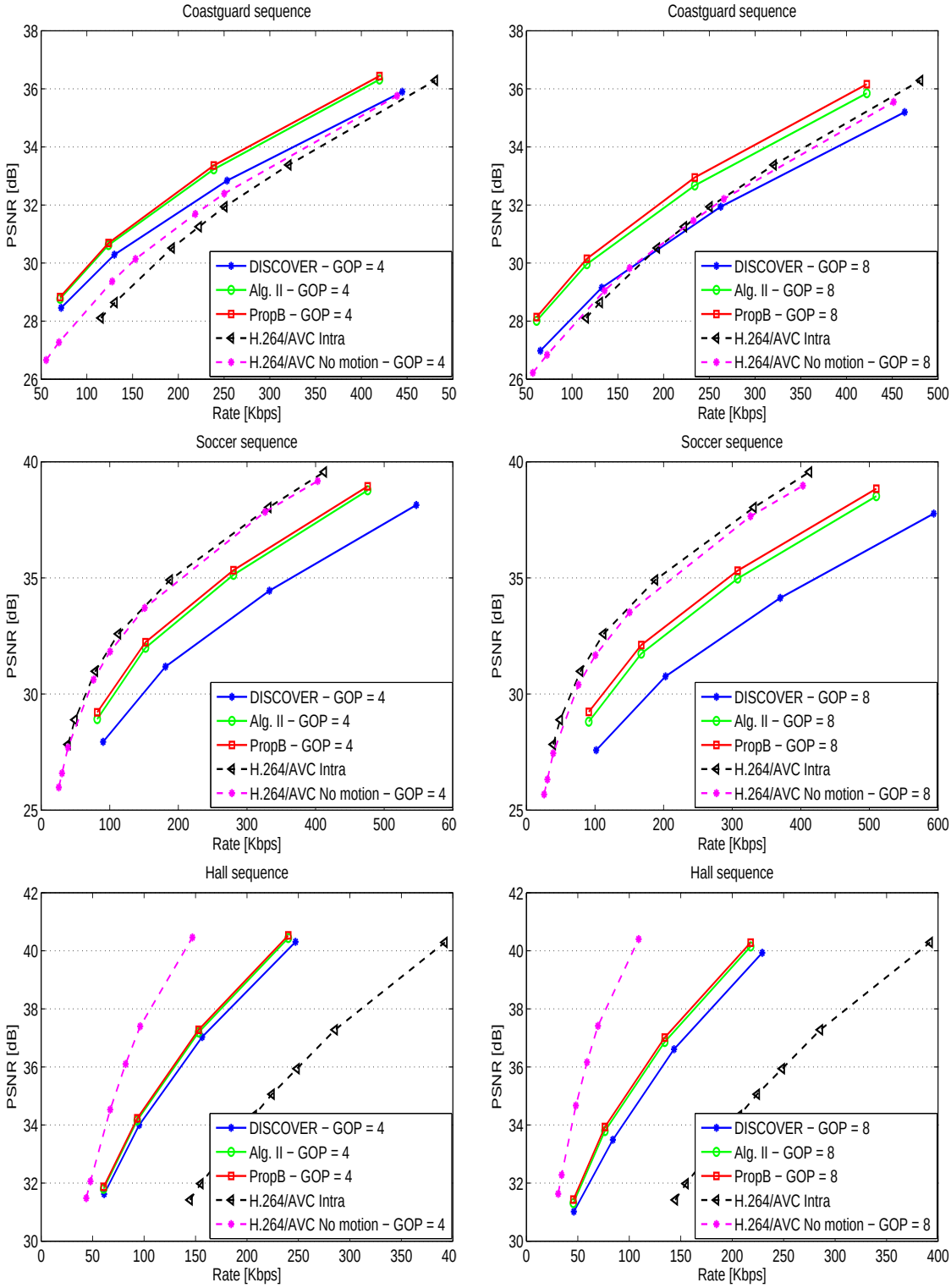

Figure 4.23: RD performance comparison among the DISCOVER codec, the Alg. II, the proposed method, H.264/AVC Intra and H.264/AVC No motion, for Coastguard, Soccer and Hall sequences, for GOP sizes of 4 and 8.

4.4 Conclusions

In this chapter, we proposed three different approaches to improve the quality of the SI in transform-domain DVC. First, a new method is proposed for SI generation based on backward and forward motion estimation. The backward and forward motion fields are first estimated for large blocks of 32×32 pixels. Then, each block is divided into four blocks and a quadtree refinement algorithm is used to select the motion vectors for those blocks from the already estimated motion vectors. The same procedure is repeated until the block size becomes 4×4 pixels. Furthermore, a motion vector is selected among the backward and forward motion vectors for each block. In addition, spatial smoothing and bi-directional motion estimation are applied to refine the motion vectors. Finally, motion compensation is applied using the reference frames to obtain the SI.

In the second part, we proposed a new approach which consists in adapting the search area to the current motion, after decoding the first DCT band, using the PDWZF and the two reference frames. First, N large blocks are uniformly selected in the PDWZF and the block matching is carried out between the PDWZF and the backward (and forward) reference frame. Then, the obtained motion vectors for those large blocks are used to adapt an initial search area. The PDWZF is used along with the backward (and forward) reference frame to successively improve the SI using the adapted search area.

Finally, we proposed a new method to re-estimate the SI using the already decoded WZFs, for long duration GOPs, in transform-domain DVC. First, all WZFs are decoded within the current GOP. Then, the SI is re-estimated using the already decoded WZF along with the neighboring backward and forward decoded frames. More specifically, a variable block size algorithm is used during the re-estimation with the adaptation of the search area. Finally, the reconstruction is carried out again using the enhanced SI to obtain the improved WZF.

All three proposed methods allow consistent performance gains compared to DISCOVER codec and to other techniques that can be found in the literature. They can be applied separately or in a combined way. Even though additional steps are added to the decoder, most of the time the decoding computational load is decreased, compared to DISCOVER codec, due to the fact that the enhanced SI allows for a quicker convergence of the iterative decoding procedure. At the same time, the necessary bit rate for the decoding of the WZFs is also decreased.

The material in this chapter was published in:

1 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu “Successive Refinement of Side Information using Adaptive Search Area for Long Duration GOPs in Distributed Video Coding ”, *19th International Conference on Telecommunications (ICT 2012)*, April, Jounieh, Lebanon.

2 A. Abou-Elailah, G. Petrazzuoli, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-

Popescu “Side Information Improvement in Transform-Domain Distributed Video Coding ”, *SPIE, Applications of Digital Image Processing XXXV Conference*, August 2012, San Diego, CA United States.

Chapter 5

Fusion of global and local motion estimation

Contents

In this chapter, we propose a new method for enhancing the SI in transform-domain DVC. This solution consists in combining global and local SI at the decoder. The global motion parameters are computed at the encoder, based on a low-complexity estimation technique: For a given WZF, feature points of the two original reference frames and of the original WZF are extracted by carrying out the Scale-Invariant Feature Transform (SIFT) [17] algorithm. Then, a matching between these feature points is applied. Next, we aim at finding the matches that belong to the global motion in the scene. For this

purpose, we propose an efficient algorithm that consists in eliminating iteratively the false matches due to local motion, in order to estimate the parameters of a global motion model between the current WZF and the backward or forward reference frame. The parameters of the global model are sent to the decoder in order to generate a SI based on Global Motion Compensation (GMC), and referred to as GMC SI. On the other hand, another SI is estimated using the MCTI technique (local motion estimation) with spatial motion smoothing, exactly as in DISCOVER codec [4][5]. This SI will be called MCTI SI.

Furthermore, two different methods are proposed to combine GMC SI and MCTI SI frames. In the first method, the sum of absolute differences (SAD) between the corresponding blocks in the two estimations is used for the combination of the two SI frames. Based on the obtained SADs, a binary or a linear combination is performed. The second method consists in using Support Vector Machine (SVM) to fuse GMC SI and MCTI SI. In this case, the SVM classifier provides a decision for each block in the WZF. Then, a binary or a linear combination is also performed based on the decision of the SVM classifier.

In addition, we propose an improvement technique for the fusion of GMC SI and MCTI SI, by a refinement of the SI after the decoding of the first DCT band. Starting with the fusion of GMC SI and MCTI SI, the decoder reconstructs a Partially Decoded Wyner-Ziv Frame (PDWZF) by correcting the SI with the parity bits of the first DCT band. Here, two variants of the algorithm are proposed to enhance the first fusion. The first one consists in improving the fusion after decoding the first DCT band, using the PDWZF, while the second uses the PDWZF along with the decoded DC coefficients. It is important to note that these approaches are very efficient in terms of the computational load.

Moreover, we also propose to successively improve the fusion of GMC SI and MCTI SI after the decoding of each DCT band. Here, two variants are also proposed. In the first one, the fusion is simply refined using the PDWZF after decoding each DCT band. The second method consists in improving the fusion using the PDWZF along with the backward and forward reference frames. This method consists in re-estimating the false motion vectors obtained by the MCTI technique, after the decoding of each DCT band, similarly to the refinement techniques previously exposed in the two former chapters.The fusion between GMC SI and MCTI SI is iterated after each improvement of the PDWZF.

This chapter is structured as follows. After recalling the most relevant related work in Section 5.1, the estimation of the GMC SI is introduced in Section 5.2. The proposed techniques for the fusion of GMC SI and MCTI SI are then described in Section 5.3, along with the experimental results. The methods for enhancing the fusion of GMC SI and MCTI SI during the decoding process are presented in Section 5.4 with the experimental results. Finally, conclusions are drawn in Section 5.5.

5.1 Related work

The goal of distributed compression is to achieve a coding efficiency similar to the best available hybrid video coding scheme, while ensuring a lower encoding complexity. However, DVC has not reached the performance level of classical inter frame coding yet. This is in part due to the quality of the SI, which has a strong impact on the final Rate-Distortion (RD) performance.

Commonly, the SI is generated by applying the MCTI technique on consecutive reference frames and already reconstructed WZFs. The quality of the SI is poor in certain regions of the video scene, like in areas of partial occlusions, fast motion, etc. In this case, a hash information may be transmitted to the decoder in order to improve the SI. However, the encoder needs to determine in advance the regions where the interpolation at the decoder would fail. In [35][49], hash information is extracted from the WZF being encoded and sent only for the macroblocks where the sum of squared differences between the previous reference frame and the WZF is greater than a certain threshold. In [67], global motion is estimated at the decoder in order to adapt temporal inter-/extrapolation for SI generation.

In [68], the authors proposed a Witsenhausen-Wyner Video Coding (WWVC) that employs forward motion estimation at the encoder and sends the motion vectors to the decoder to generate the SI. This WWVC scheme achieves better performance than H.264/AVC in noisy networks and suffers a limited loss (up to 0.5 dB compared to H.264/AVC) in noiseless channels. The authors in [69] proposed a novel framework that integrates the graph-based segmentation and matching to generate inter-view SI in Distributed Multiview Video Coding.

The problem of SI fusion has been addressed in Multiview DVC where two SI are usually generated. The first SI (SI_t) is generated from previously decoded frames in the same view, while the second one (SI_v) is estimated using previously decoded frames in adjacent views. The authors in [70] proposed several new techniques for the fusion of SI_t and SI_v . Dufaux [71] proposed a solution that consists in combining SI_t and SI_v using SVM.

In [72][73][74], the authors presented DVC schemes that consist in performing the motion estimation both at the encoder and decoder. In [72], the authors propose a pixel-domain DVC scheme, which consists in combining low complexity bit plane motion estimation at the encoder side, with motion compensated frame interpolation at the decoder side. Improvements are shown for sequences containing fast and complex motion. The authors in [73] present a DVC scheme where the task of motion estimation is performed both at the encoder and decoder. Results have shown that the cooperation of the encoder and decoder can reduce the overall computational complexity while improving coding efficiency. Finally, a DVC scheme proposed by Dufaux *et al.* [74] consists in combining the global and local motion estimations at the encoder. In this scheme, the motion estimation and

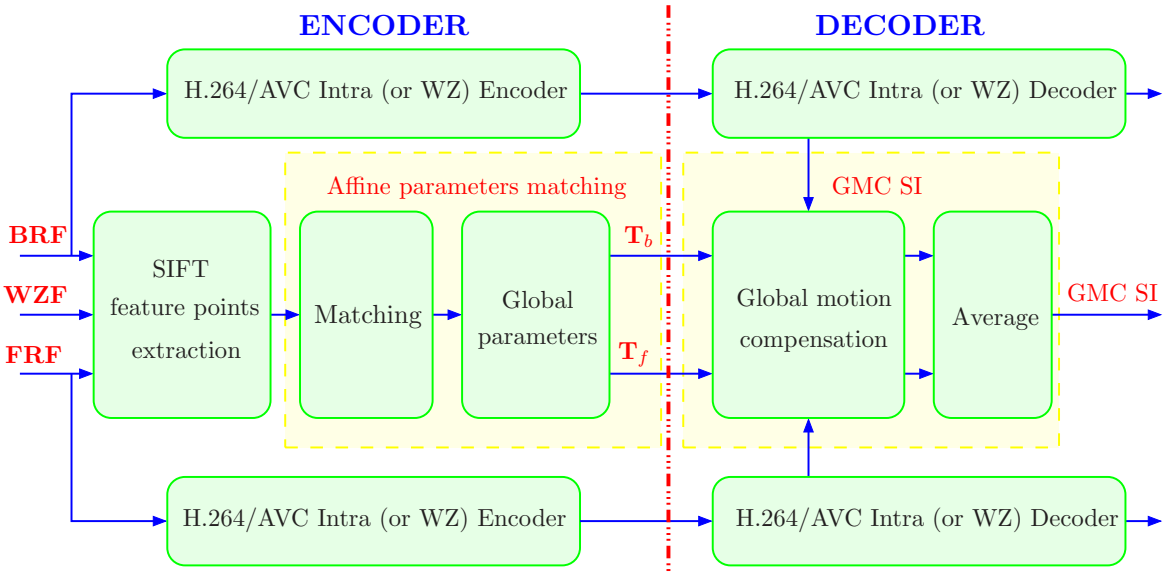

Figure 5.1: Block diagram of the proposed GMC technique.

compensation are performed both at the encoder and decoder.

On the contrary, in the proposed method, the local motion estimation is only performed in the decoder, while the global motion parameters are estimated in the encoder using a low-complexity SIFT algorithm. The global parameters are sent to the decoder to estimate the GMC SI and the combination between the GMC SI and MCTI SI is made at the decoder side.

5.2 Global Motion Estimation and Compensation

In this section, we describe the estimation technique of the global parameters based on the feature points. Then, the generation of the global SI (GMC SI) at the decoder is shown, and a solution is proposed to remedy for eventual black borders in the generated GMC SI.

5.2.1 Global Parameters Estimation

Fig. 5.1 shows the block diagram of the proposed GMC technique. At the encoder side, we extract the feature points of the Backward Reference Frame (BRF), Forward Reference Frame (FRF), and the original WZF. These feature points are extracted by applying the SIFT algorithm [17]. Matching between the feature points of the WZF and the backward and forward reference frames is then applied in order to estimate the global motion parameters defined by the two transforms T_b and T_f, respectively.

In this context, several global motion models are analyzed in order to choose the most suited one for our proposed method. Three parametric models are considered: translational

motion model (two parameters), affine motion model (six parameters), and perspective motion model (eight parameters). The perspective motion model is defined as follows:

$$\begin{cases} u_i &= (a_0 + a_2 x_i + a_3 y_i)/(a_6 x_i + a_7 y_i + 1) \\ v_i &= (a_1 + a_4 x_i + a_5 y_i)/(a_6 x_i + a_7 y_i + 1) \end{cases}$$

where $(a_0, a_1, ..., a_7)$ are the motion parameters, $(x_i,\ y_i)$ denotes the pixel location in the WZF, and $(u_i,\ v_i)$ the corresponding position in the backward or forward reference frame. The affine $(a_6 = a_7 = 0)$ and the translational $(a_2 = a_5 = 1, a_3 = a_4 = a_6 = a_7 = 0)$ models are particular cases of the perspective model.

Afterwards, we carry out an efficient algorithm on the feature matches that estimates the parameters of the model between the WZF and the BRF. This algorithm allows us to remove the false matches, *i.e.*, the matches that exist on individual objects of the scene and correspond to local motion. The motion parameters between the WZF and the FRF are estimated in the same way.

The motion parameters are estimated by minimizing:

$$E = \sum_{i=1}^{N} f(E_i) \tag{5.1}$$

with

$$f(E_i) = \begin{cases} E_i & \text{if } E_i < T \\ 0 & \text{otherwise} \end{cases}$$

where E_i represents the error of feature match number i, and N represents the number of the feature matches between the two frames. In the aim of increasing the robustness of the estimation technique to false feature matches, a threshold T is defined so as to take into account only the most accurate feature matches, corresponding to a fixed percentage.

The error of feature match number i is defined as:

$$E_i = \left[(u_i - r_i)^2 + (v_i - s_i)^2\right]. \tag{5.2}$$

where

$$\begin{cases} r_i &= a_{0e} + a_{2e} x_i + a_{3e} y_i / (a_{6e} x_i + a_{7e} y_i + 1) \\ s_i &= a_{1e} + a_{4e} x_i + a_{5e} y_i / (a_{6e} x_i + a_{7e} y_i + 1) \end{cases}$$

(r_i, s_i) are the coordinates in the backward or forward reference frame, corresponding to the feature point (x_i, y_i) in the WZF, according to the actual estimated parameters $(a_{0e}, a_{1e}, ..., a_{7e})$.

The flowchart diagram of the proposed algorithm for the estimation of the global model parameters is depicted in Fig. 5.2. The algorithm consists of the following steps to estimate the parameters of the two transforms T_b and T_f describing the motion models between the WZF and the backward and forward reference frames, respectively:

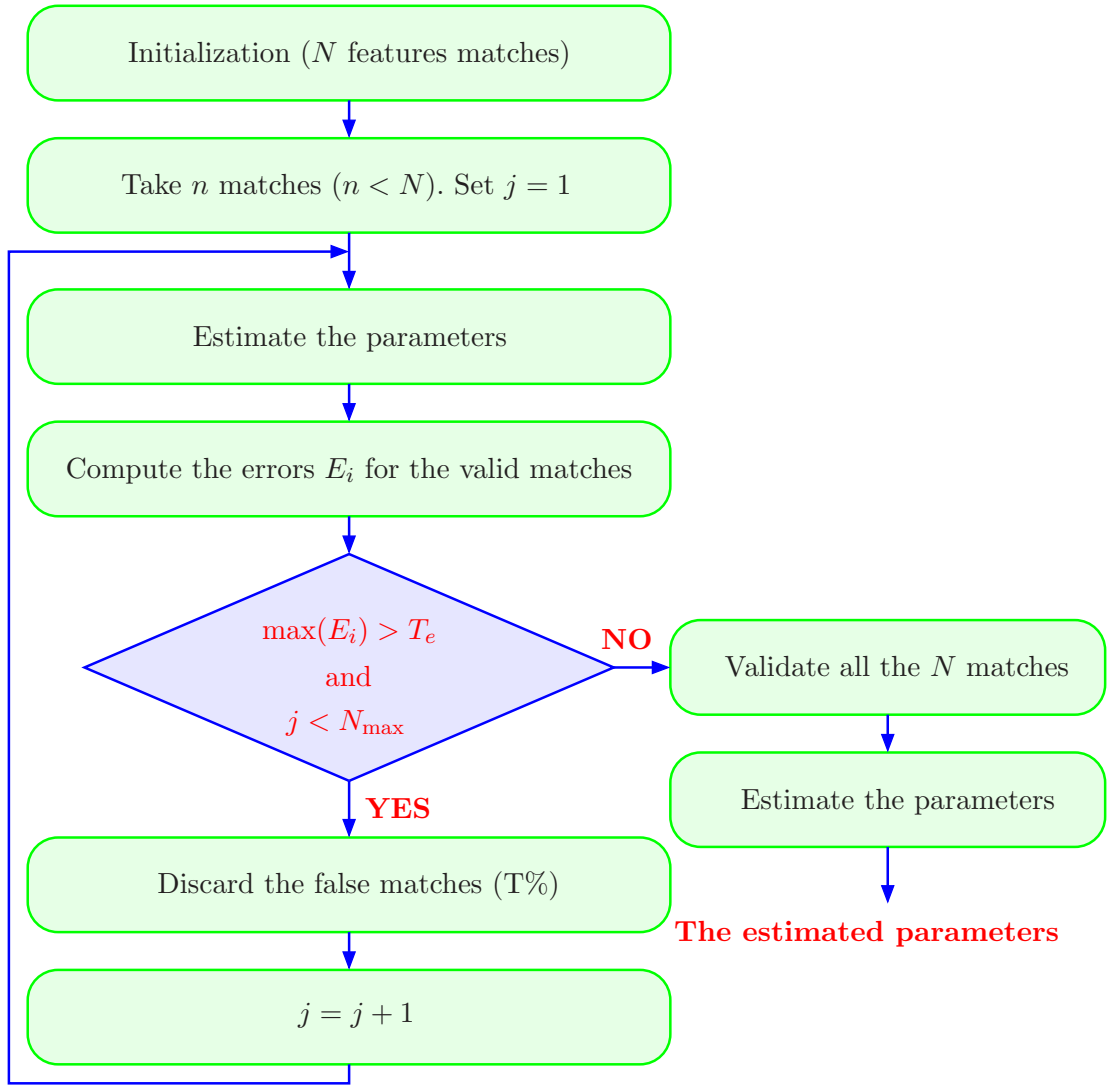

Figure 5.2: Flowchart diagram of the proposed global model parameters estimation.

Step 1 - N feature matches are obtained between the original WZ and the original reference (backward and forward) frame using SIFT algorithm. Typically, a large number of matches are found. However, in the unlikely case where no matches are found (e.g. in the case of a shot cut), the global motion estimation procedure is stopped and only MCTI SI is used.

Step 2 - Commonly, the moving objects appear mostly in the center of the frame. In order to increase the probability of the feature matches belonging to the global motion compared to the local motion, the proposed algorithm takes the feature points that belong to the top and bottom quarters of the frame (n feature matches are taken, $n < N$). This step allows a quick and accurate convergence of the algorithm.

Step 3 - The parameters of the model T_b (respectively T_f) are estimated by minimizing

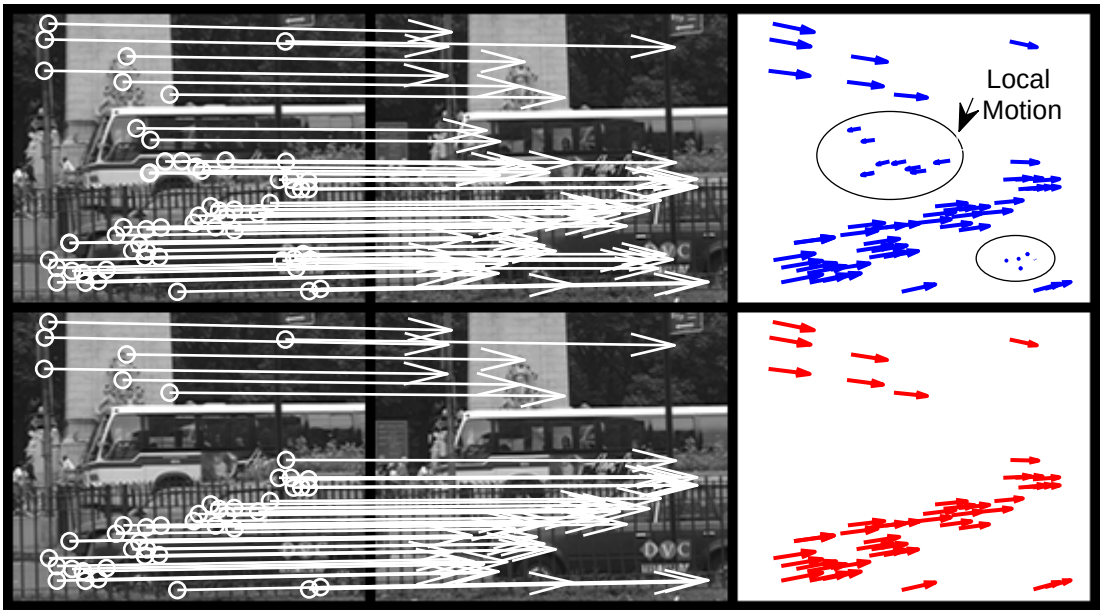

Figure 5.3: The obtained feature matches between frames 17 and 21 of Bus sequence before (blue, top) and after (red, bottom) applying the proposed algorithm.

the Euclidean distance estimated on the n feature matches (Eq. 5.1), *i.e.*, between the feature points in the WZF and the corresponding feature points in the backward or forward reference frame.

Step 4 - The error of each match E_i (n matches) is computed according to Equation (5.2). If the maximum error E_{max} ($E_{max} = max(E_i)$) is greater than a threshold T_e, go to **Step 5**. Otherwise, go to **Step 7**.

Steps 5 and 6 - The feature matches which give the largest errors (the top T% of the distribution E_i) are discarded, and the rest of the feature matches are taken for the next iteration ($i = i + 1$). Then, go to **Step 3**.

Step 7 - The feature matches of the entire frame (N feature matches) are fed into the estimated model to identify the valid feature matches. The feature match that gives an error greater than T_e is considered to be false match (belongs to the local motion) and discarded.

Step 8 - Finally, the algorithm computes once again the parameters of the model T_b (respectively T_f) by taking into account only the valid feature matches (belonging to the global motion) of the entire frame.

In this algorithm, at most N_{max} iterations are carried out. In most cases, the algorithm converges rapidly before the N_{max} iterations. We have empirically chosen $N_{max} = 5$ and $T_e = 1$ in our simulations.

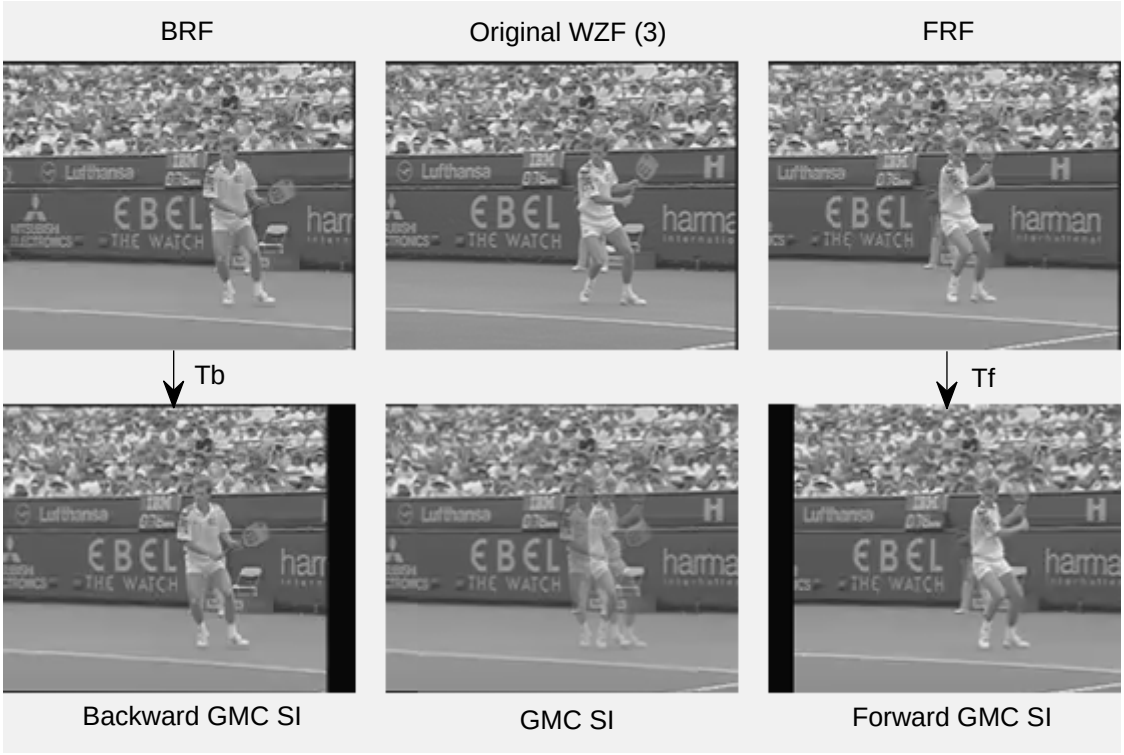

Figure 5.4: SI generated by GMC.

Fig. 5.3 shows the feature matches between the frames no. 17 and 21 of the Bus sequence. The top frames represent the feature matches (blue) obtained by applying the method in [17] (60 feature matches are obtained). The bottom frames represent the feature matches (red) obtained by carrying out our algorithm (14 feature matches are removed). It is clear that the proposed technique discards all feature matches corresponding to local motion.

Once the transforms T_b and T_f have been estimated at the encoder, the computed parameters (4 in case of a translational model, 12 in case of an affine model, or 16 for the perspective model) are sent to the decoder for each WZF.

5.2.2 Global SI Generation

At the decoder side, the parameters of T_b and T_f are respectively applied to the backward and forward decoded reference frames in order to estimate the GMC SI. Similarly to MCTI SI, the GMC SI is obtained by averaging both backward and forward predictions. Fig. 5.4 shows a computation example of a GMC SI; the top left image represents the BRF, the top center image represents the original WZF and the top right image represents the FRF. Then, the bottom left image represents the backward GMC SI, where T_b is applied to the decoded BRF, and the bottom right image represents the forward GMC SI, where T_f is applied to the decoded FRF. Finally, the average between the pixels of the backward and

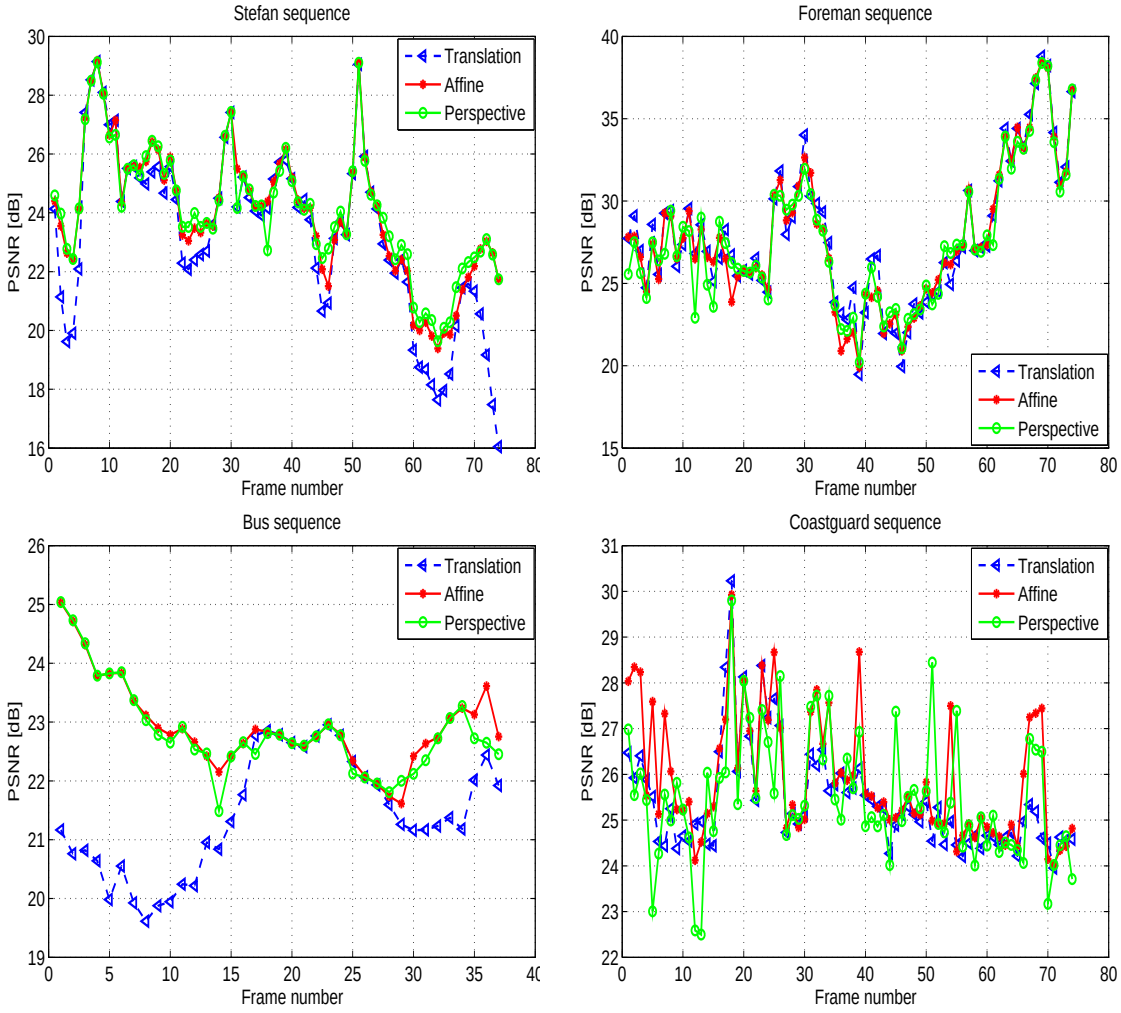

Figure 5.5: PSNR of GMC SI for Stefan, Foreman, Bus and Coastguard sequences, for various global motion models.

the forward GMC SI frames is computed to generate the GMC SI, and is shown at the bottom center. However, when the pixels are black (on the borders of the image due to camera motion) in the backward GMC SI frame, only the pixels of the forward GMC SI frame are taken for the GMC SI, and vice versa:

$$\text{GMC SI}(p) = \begin{cases} \text{BSI}(p) & \text{if FSI}(p) \text{ is black} \\ \text{FSI}(p) & \text{if BSI}(p) \text{ is black} \\ \frac{\text{BSI}(p)+\text{FSI}(p)}{2} & \text{otherwise} \end{cases} \tag{5.3}$$

where BSI and FSI are the backward and forward GMC SI, respectively.

The experimental assessment of the quality of GMC SI, realized for various global motion models, is shown in Figure 5.5 for Stefan, Foreman, Bus, and Coastguard sequences (QCIF, at 15 Hz). As it can be seen from the obtained results, the translational model allows a small gain in the Foreman sequence, but it generally fails when the global mo-

Table 5.1: Final SI average PSNR for GOP size equal to 2 (QI = 8) for Stefan, Foreman, Bus and Coastguard sequences for different global models.

Final SI Average PSNR [dB]				
Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**
GOP = 2				
Translation	23.26	27.84	21.42	25.41
Affine	23.98	**27.66**	**22.93**	**25.91**
Perspective	**24.08**	27.59	22.84	25.49

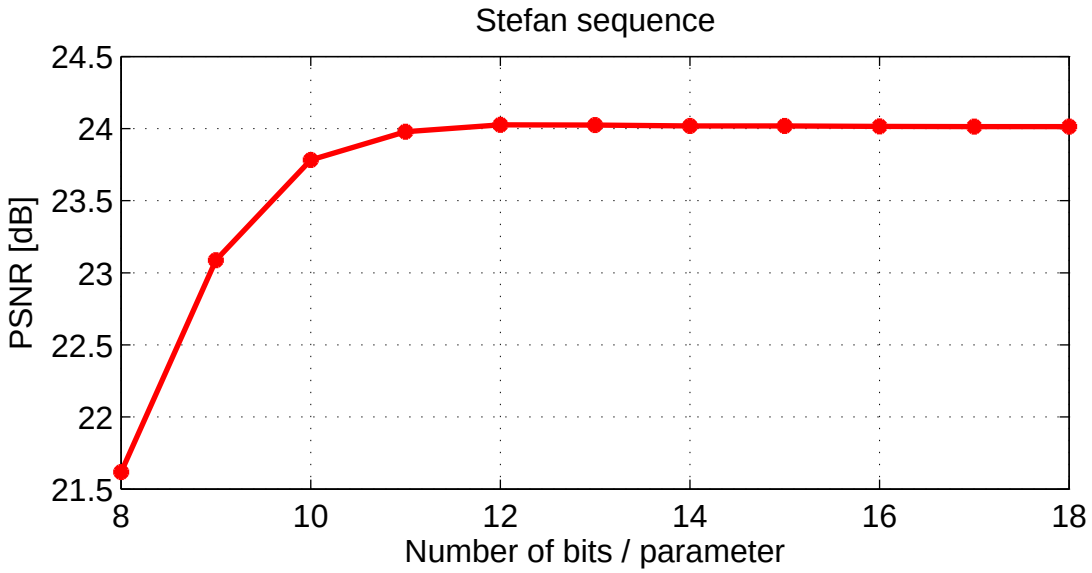

Figure 5.6: Average PSNR of the GMC SI frames in terms of number of bits per parameter (*i.e.*, affine global parameters) for Stefan sequence, for a GOP size of 2.

tion becomes more complicated, as in Stefan and Bus sequences. On the other hand, the perspective model is less robust in the case of noisy matches. Table 5.1 shows the average PSNR of the SI frames, for Stefan, Foreman, Bus, and Coastguard sequences, with the different models. For Stefan sequence, the perspective model can achieve a small gain compared to the affine model. The latter leads to the best PSNR average for Foreman, Bus, and Coastguard sequences. Therefore, the affine model will be adopted for the rest of this chapter.

First, a_2 and a_5 represent the scale parameters, a_3 and a_4 represent the shear parameters and the parameters a_0 and a_1 represent the translation vector between the two frames. In a video sequence, the amount of scaling and shearing between successive frames remains typically small, whereas the translation vector may be large. Fig. 5.6 represents the average PSNR of the GMC SI frames as a function of the number of bits per parameter, for Stefan sequence, for a GOP size of 2. The quality of the GMC SI becomes stationary after 12 bits per parameter. As for the affine parameters transmission, we encode each parameter using 15 bits.

Specifically, the parameters a_2 and a_5 can be written as $1 + s \times f$, where s is the sign of the number and f a positive floating value ($f < 1$). We encode s and f using 1 bit and

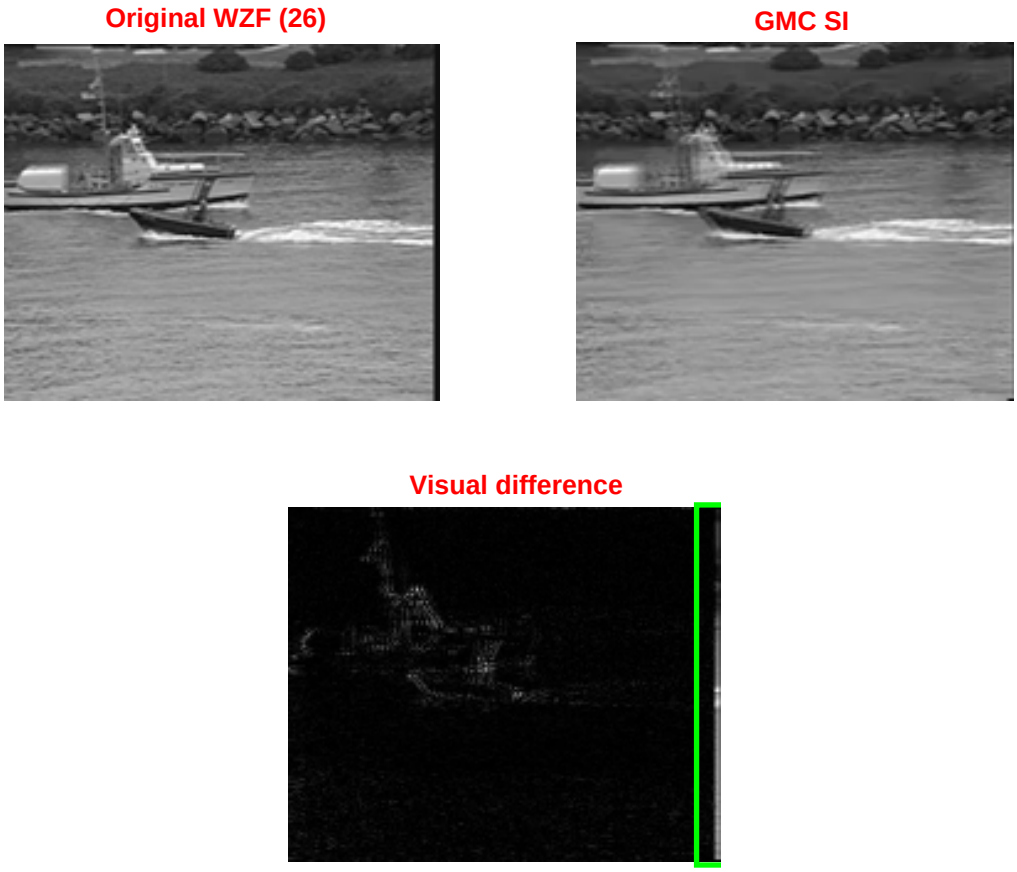

Figure 5.7: Original WZF number 26 of Coastguard sequence, the corresponding estimated GMC SI and the visual difference between GMC SI and the original WZF.

14 bits respectively. The parameters a_3 and a_4 can be written as $s \times f$, where s is the sign of the number and f a positive floating number ($f < 1$). We encode s and f using 1 bit and 14 bits respectively. For the translation parameters a_0 and a_1, the maximum translation between two frames is considered to be ± 128 pixels. Thus, these parameters can be written as $s \times (n + f)$, where s, n, and f represent the sign of the number, an integer number ($n < 128$), and a positive floating number ($f < 1$) respectively. Then, s is encoded using 1 bit, n and f are encoded using 7 bits respectively.

For the case of a video at QCIF resolution and 15 Hz with a GOP size of 2, the supplementary data burden, for the transmission of the affine parameters, will be only 180 ($2 \times 6 \times 15$) bits (15 bits/parameter) per WZF (1.35 kbps). Thus, the bitrate overhead necessary to transmit the global parameters is negligible.

5.2.3 GMC SI borders improvement

The borders of the original frames can be black in some sequences such as the Coastguard sequence. In the estimation process of the GMC SI, the borders of the obtained SI frames

Table 5.2: Average PSNR of the GMC SI before and after border improvement, for all GOP sizes (QI = 8).

Sequence	**Stefan**	**Foreman**	**Bus**	**Coastguard**
GOP = 2				
Initial GMC SI [dB]	24.02	28.42	23.10	25.92
GMC SI [dB]	**25.88**	**30.70**	**23.10**	**29.28**
GOP = 4				
Initial GMC SI [dB]	23.43	27.39	22.53	24.99
GMC SI [dB]	**25.27**	**29.62**	**22.53**	**28.19**
GOP = 8				
Initial GMC SI [dB]	23.12	26.51	21.95	24.52
GMC SI [dB]	**24.85**	**28.62**	**21.95**	**27.50**

cannot be black due to the global shift of the reference frames (*i.e.*, camera motion). Fig. 5.7 shows an example of the GMC SI, for frame number 26 of Coastguard sequence. As we can see, the right borders of the GMC SI is not black, unlike the original WZF. Indeed, this effect perturbs the quality of the estimated SI by GMC. For this reason, we propose an approach that allows improving the quality of the GMC SI for sequences containing black borders.

For this purpose, after the estimation of the GMC SI, the difference between the reference frames at the borders is considered. More specifically, for an image having NR rows and NC columns, a window $\mathbf{w}$ of three pixels is defined as follows:

$$\mathbf{w} = \{(x, y) : x \in [1, 3] \cup [\mathrm{NR} - 3, \mathrm{NR}], y \in [1, 3] \cup [\mathrm{NC} - 3, \mathrm{NC}]\} \tag{5.4}$$

For each block $\mathrm{b} \in \mathbf{w}$ (b represents a block of 1×8 or 8×1 pixels), the average absolute difference between the decoded BRF and FRF is calculated by:

$$\mathrm{DBF_b} = \frac{1}{8} \sum_{p \in \mathrm{b}} |\mathrm{BRF}(p) - \mathrm{FRF}(p)| \tag{5.5}$$

The GMC SI is then updated according to the obtained difference $\mathrm{DBF_b}$ as follows:

$$\mathrm{GMC\ SI}(p) = \begin{cases} \frac{\mathrm{BRF}(p) + \mathrm{FRF}(p)}{2} & \text{if } \mathrm{DBF_b} < \mathrm{Th} \\ \mathrm{GMC\ SI}(p) & \text{otherwise} \end{cases} \tag{5.6}$$

where Th is a threshold (it is empirically set to 30).

Fig. 5.8 shows the PSNR of the initial GMC SI and of the GMC SI after the enhancement of the borders, for Stefan, Foreman, and Coastguard sequences, for GOP sizes of 2 and 8. It is clear that the proposed approach can improve the borders of the GMC SI estimation.

In Table 5.2, we show the average PSNR of the initial GMC SI and of the GMC SI after updating the borders. Border enhancement allows a consistent gain for sequences containing black borders like Stefan, Foreman, and Coastguard sequences. The gain can

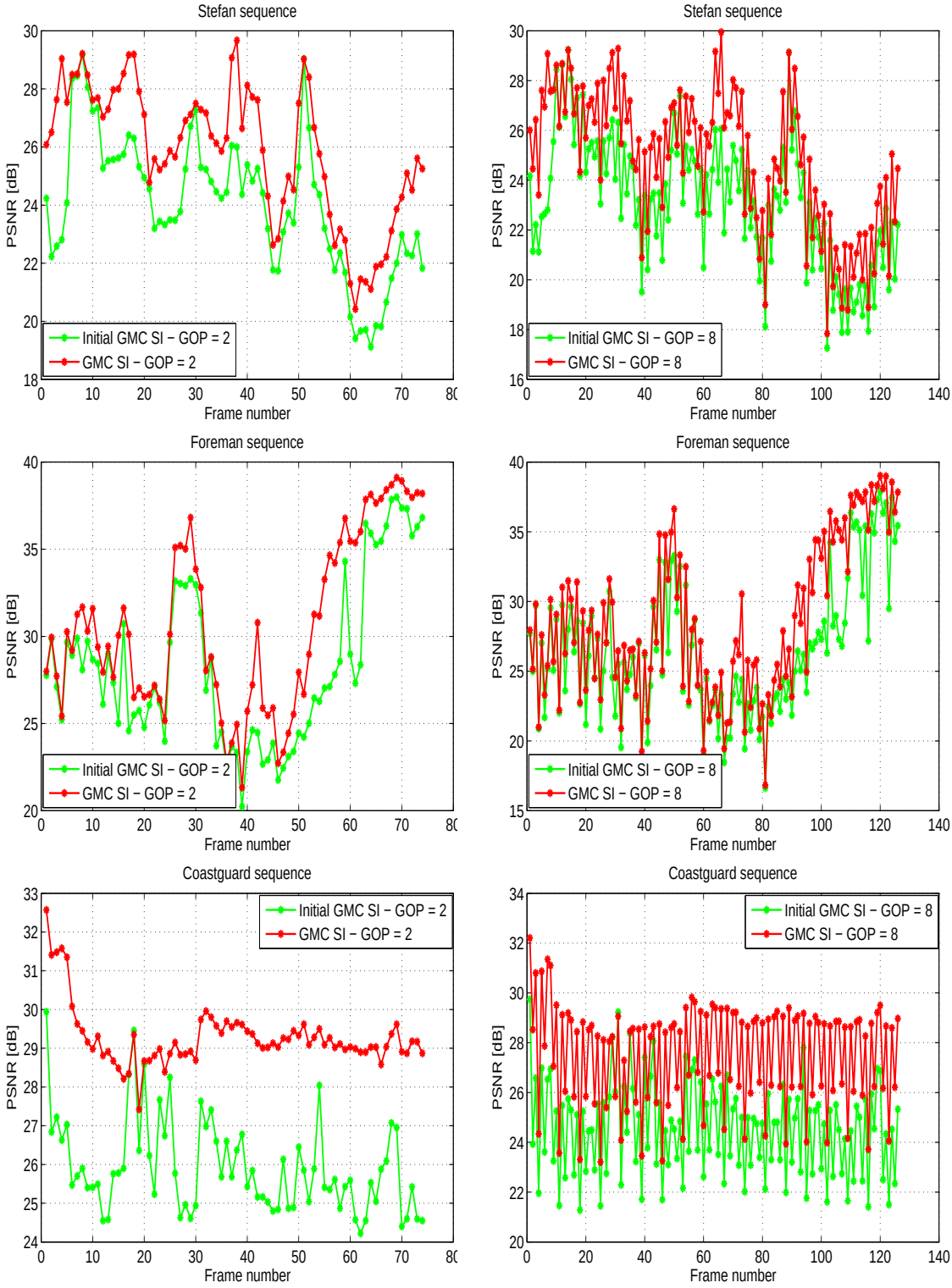

Figure 5.8: PSNR of the initial GMC SI and of the GMC SI after border improvement, for Stefan, Foreman, and Coastguard sequences, for GOP sizes of 2 and 8.

reach 3.36 dB for Coastguard sequence, with a GOP size of 2.

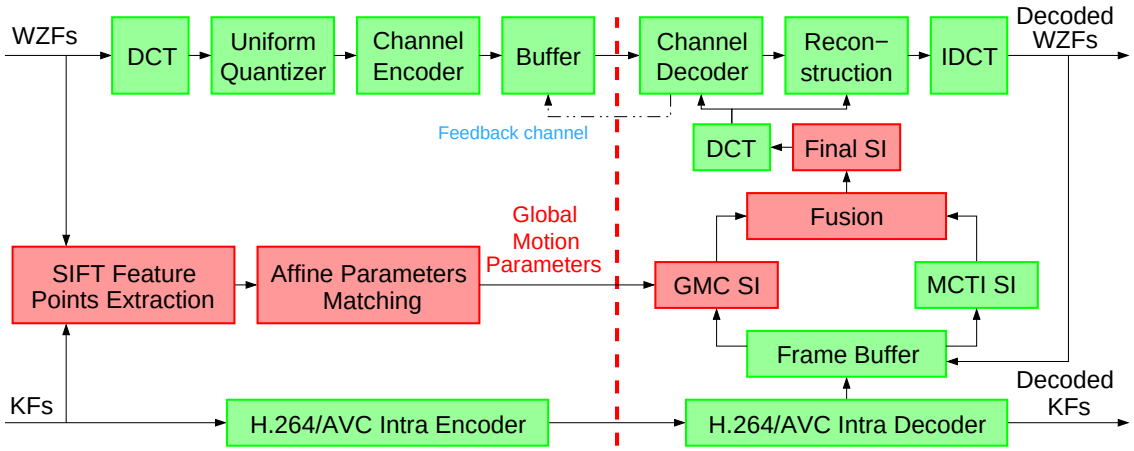

Figure 5.9: Overall structure of the proposed DVC codec based on the combination of GMC SI and MCTI SI.

5.3 Fusion of MCTI SI and GMC SI

The current section deals with the fusion between MCTI SI and GMC SI, both generated at the decoder, as described in the previous section. The proposed scheme is illustrated in Fig. 5.9. Here, two different methods for combining MCTI SI and GMC SI are proposed. The first method is based on the SADs between the corresponding blocks in the two estimations, while the second one uses SVM to combine the two SI frames.

5.3.1 Fusion based on SADs between corresponding blocks

Let us call the backward and forward reference frames respectively R_B and R_F, for short. Moreover, we denote by $\hat{R}_B$ and $\hat{R}_F$ the results of the GMC transforms T_B and T_F applied to R_B and R_F. The GMC SI is simply defined as the average of the frames $\hat{R}_B$ and $\hat{R}_F$. On the other hand, let $\tilde{R}_B$ and $\tilde{R}_F$ be the backward and forward compensated reference frames estimated by the MCTI technique.

The block size adopted for the fusion step is 4×4 pixels. Fig. 5.10 shows the combination of the global and local motion estimations. For a given block B in the current SI, the following steps are carried out:

Step 1 - The global transforms T_b and T_f are applied to the backward and forward reference frames respectively. The corresponding blocks to the current block B are now directly B_{gb} and B_{gf} in the same position of the current block. Then, the SAD between B_{gb} and B_{gf} is computed using an 8×8 window around the blocks B_{gb} and

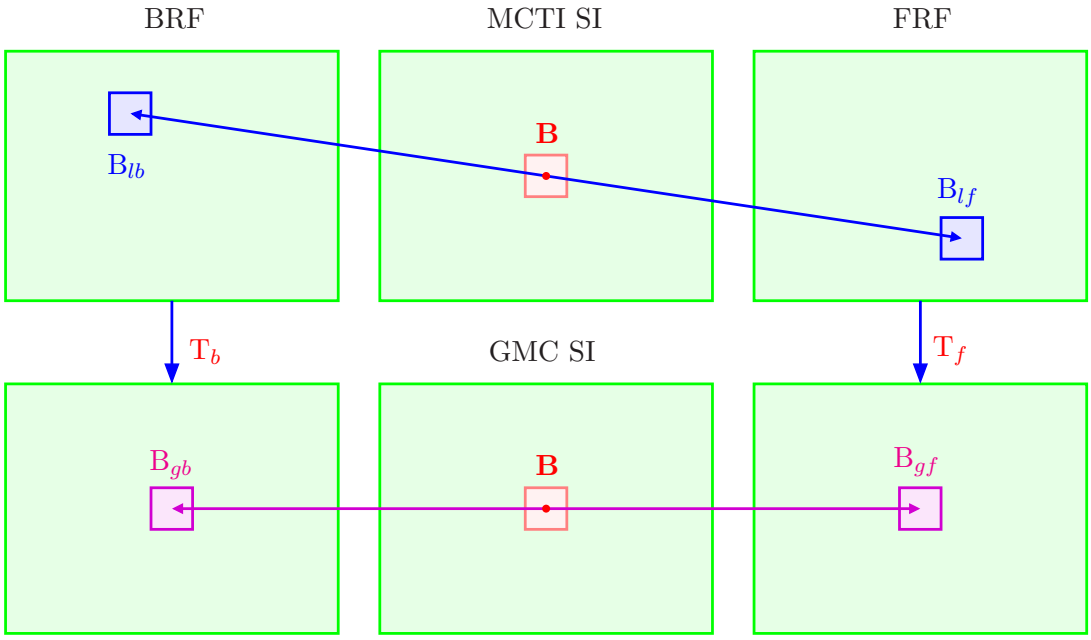

Figure 5.10: Fusion of global and local motion estimations.

B_{gf}:

$$\begin{aligned} \text{SAD}_{\text{GMC}} &= |B_{gb} - B_{gf}| \\ &= \sum_{i=-4}^{3} \sum_{j=-4}^{3} |\hat{R}_F(X_i, Y_j) - \hat{R}_B(X_i, Y_j)| \end{aligned} \tag{5.7}$$

where $(X_i, Y_j) = (x_0 + i, y_0 + j)$, and (x_0, y_0) is the coordinate of the center pixel of the current block B.

Step 2 - The SAD is computed between the corresponding blocks B_{lb} and B_{lf} in the backward and forward reference frames, these blocks being determined by the MCTI technique. The SAD_{MCTI} is computed using a 8×8 window around the blocks B_{lb} and B_{lf}:

$$\begin{aligned} \text{SAD}_{\text{MCTI}} &= |B_{lb} - B_{lf}| \\ &= \sum_{i=-4}^{3} \sum_{j=-4}^{3} |\tilde{R}_F(X_i, Y_j) - \tilde{R}_B(X_i, Y_j)| \end{aligned} \tag{5.8}$$

where $(X_i, Y_j) = (x_0 + i, y_0 + j)$, and (x_0, y_0) is the coordinate of the center pixel of

the current block B.

Step 3 - Based on $\mathrm{SAD}_{\mathrm{GMC}}$ and $\mathrm{SAD}_{\mathrm{MCTI}}$, we propose two methods for combining the global and local motion estimations. The first method consists in a binary fusion of the two SI frames and the second method aims at linearly combining the two SI frames.

SAD binary fusion - The GMC SI and MCTI SI are combined using $\mathrm{SAD}_{\mathrm{GMC}}$ and $\mathrm{SAD}_{\mathrm{MCTI}}$ as follows:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI(b)} & \text{if } \mathrm{SAD}_{\mathrm{GMC}} < \mathrm{SAD}_{\mathrm{MCTI}} \\ \text{MCTI SI(b)} & \text{otherwise} \end{cases} \tag{5.9}$$

This method is referred to as 'SADbin'.

SAD linear fusion - Inspired from [70], a linear fusion of GMC SI and MCTI SI is proposed as follows:

$$\mathrm{SI}(b) = \frac{\mathrm{SAD}_{\mathrm{MCTI}} \cdot (\text{GMC SI}) + \mathrm{SAD}_{\mathrm{GMC}} \cdot (\text{MCTI SI})}{(\mathrm{SAD}_{\mathrm{GMC}} + \mathrm{SAD}_{\mathrm{MCTI}})} \tag{5.10}$$

This method is referred to as 'SADlin'.

At the border of the image, if the pixels of the block B_{gb} or B_{gf} are black due to the shift resulting from the application of the global transforms, the GMC SI is only taken into account to generate the fusion of these blocks (in this case, the pixels in the GMC SI are only taken from B_{gb} if the block B_{gf} is black and vice versa).

The error distribution between the corresponding DCT bands of SI and WZFs is necessary for the Slepian-Wolf decoder, in order to correct the errors in the DCT SI coefficients. Furthermore, an offline process for determining this distribution is not realistic, since it requires either the encoder to recreate the SI or to have the original data available at the decoder. In [44], the correlation noise is estimated online at the decoder, using the residual frame (denoted by RF) between the backward and forward motion compensated reference frames as a confidence measure for the frame interpolation operation. Here, this approach is adopted for the MCTI SI:

$$\mathrm{RF}_{\mathrm{MCTI}}(x, y) = \tilde{R}_F(x, y) - \tilde{R}_B(x, y) \tag{5.11}$$

For GMC SI, the difference between the transformed decoded reference frames (by applying the transforms T_b and T_f) is computed to create the residual frame for the correlation noise:

$$\mathrm{RF}_{\mathrm{GMC}}(x, y) = \begin{cases} 0 & \text{if } \hat{R}_B(x, y) \text{ or } \hat{R}_F(x, y) \text{ is black} \\ \hat{R}_F(x, y) - \hat{R}_B(x, y) & \text{otherwise} \end{cases} \tag{5.12}$$

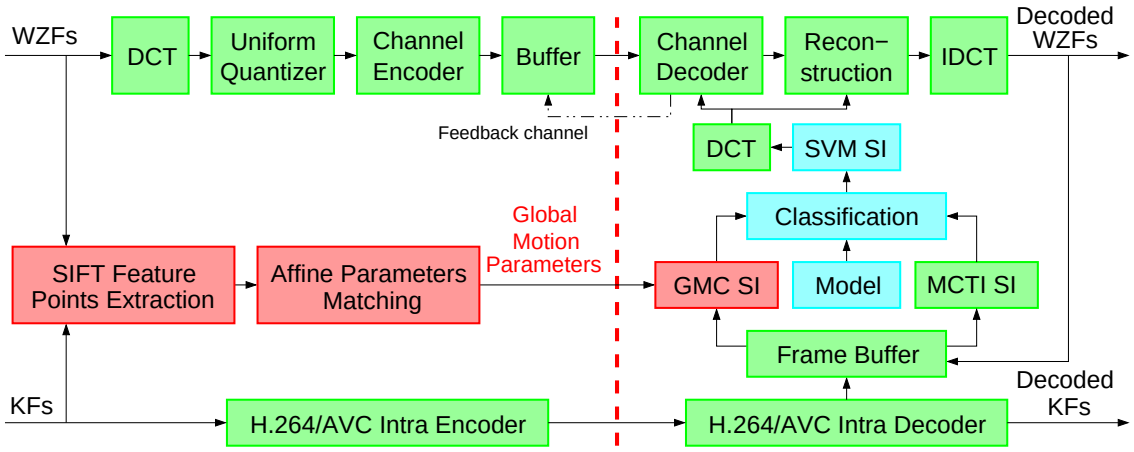

Figure 5.11: Overall structure of the proposed DVC codec based on SVM.

The correlation noise for the new SI (fusion of MCTI SI and GMC SI) is estimated by combining the two residual frames $\mathrm{RF}_{\mathrm{MCTI}}$ and $\mathrm{RF}_{\mathrm{GMC}}$ in the same manner as in Fig. 5.10. In other words, the two residual frames are combined according to the fusion scheme of MCTI SI and GMC SI.

5.3.2 Fusion using Support Vector Machine

The block diagram of our proposed codec architecture is depicted in Fig. 5.11. The three new modules in this system are the Model construction, the Classification, and the generation of SVM SI, which will be described in this section.

Each block in the SI can be predicted from either GMC SI or MCTI SI using the SVM classifier. In this work, we use the $\mathrm{SVM}^{\mathrm{Light}}$ software implementation [75]. We have investigated several kernels in this context, without noticing a significant impact on the system performance. Therefore, a linear kernel will be used hereafter.

First, the training stage used to generate the model is described, along with the classification procedure. Then, the proposed methods are described for the combination of GMC SI and MCTI SI, based on the predicted value by the SVM classifier.

Model and Classification

Fig 5.12 shows the main modules of the proposed SVM method to generate SVM SI (a combination of GMC SI and MCTI SI using SVM). First, we select the most discriminative features to be used in SVM. For this reason, three features are estimated in the proposed method as follows:

$$\begin{cases} f_1 = \mathrm{SAD}_{\mathrm{GMC}} \\ f_2 = \mathrm{SAD}_{\mathrm{MCTI}} \\ f_3 = \mathrm{SAD}_{\mathrm{GMC}} - \mathrm{SAD}_{\mathrm{MCTI}} \end{cases} \tag{5.13}$$

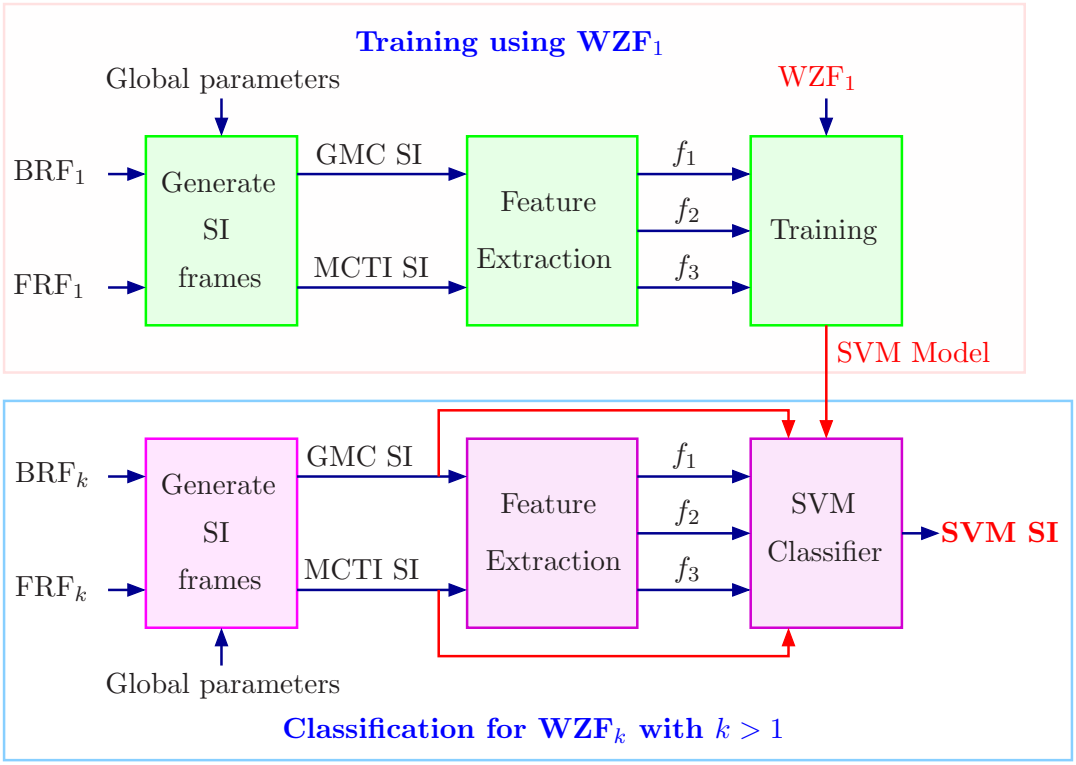

Figure 5.12: Proposed SVM-based combination algorithm to generate SVM SI.

where SAD_{GMC} and SAD_{MCTI} are defined in Eqs. 5.7 and 5.8 respectively. Note that we have considered different types of features, but we retain here the three ones (Eq. 5.13) that give the best results.

In the training stage, the first WZF is encoded using the H.264/AVC Intra mode, as in the encoding of the Key-Frames (KFs). This frame is used to build the model for SVM. For each 4×4 block b, the two SADs between the WZF and the GMC SI and MCTI SI respectively, D_{GMC} and D_{MCTI}, are computed according to:

$$\begin{aligned} D_{\text{GMC}} &= |\text{WZF}(p) - \text{GMC SI}(p)| \\ D_{\text{MCTI}} &= |\text{WZF}(p) - \text{MCTI SI}(p)| \end{aligned} \tag{5.14}$$

The block b is assigned to GMC SI if D_{GMC} is smaller than D_{MCTI} (a label of $+\mathbf{1}$ is set in this case), or to MCTI SI otherwise (with a label of $-\mathbf{1}$). Only the N blocks that give the largest difference D between the two SADs ($D = |D_{\text{GMC}} - D_{\text{MCTI}}|$) are taken in the training stage. This step allows increasing the accuracy of the training stage. In our experiments, N has been empirically set to 300 blocks (about 20% of the total number of blocks). However, the value of N has a moderate impact on the RD performance of the proposed method.

The features (f_1, f_2, and f_3) are computed for those selected blocks ($N = 300$ blocks), and used in the training step, in order to create the first model (*i.e.*, find the hyperplane

that separates the blocks of GMC SI and MCTI SI) for the classification. Next, the classification procedure is carried out on the first WZF using this model. The SVM classifier gives a decision value d for each block (d represents the distance between this block and the separating hyperplane). The predicted value d indicates the class of the block as follows:

$$\begin{cases} \text{if } d > 0 \\ \quad \text{GMC SI class} \\ \text{otherwise} \\ \quad \text{MCTI SI class} \end{cases}$$

The well-classified blocks are defined based on the predicted value d, D_{GMC}, and D_{MCTI}:

$$\begin{cases} \text{if } (d > 0 \text{ and } D_{\text{GMC}} < D_{\text{MCTI}}) \text{ or } (d < 0 \text{ and } D_{\text{GMC}} > D_{\text{MCTI}}) \\ \quad \text{This block is well-classified} \\ \text{otherwise} \\ \quad \text{This block is not well-classified} \end{cases}$$

The blocks that are well-classified are taken into account for a second learning stage, in order to produce the final model. This model will then be used in the classification procedure for all WZFs in the sequence.

In the classification, three features f_1, f_2, and f_3 are computed for each WZF using GMC SI and MCTI SI. The SVM classifier computes a predicted value for each block based on the features and the obtained model.

Based on this value, we define two fusion algorithms. The first algorithm consists of a binary combination of GMC SI and MCTI SI. The second algorithm linearly combines the two SI.

SVM binary fusion - In this method, the value d is directly used to combine the two SI as follows:

$$\text{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } d > 0 \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{5.15}$$

where d represents the classification label at block b. This method is referred to as 'SVM-bin'.

SVM linear fusion - This method aims at linearly combining GMC SI and MCTI SI. The linear combination is defined as follows:

$$\text{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } d > T \\ \text{MCTI SI} & \text{if } d < (-T) \\ \frac{(T+d)\cdot\text{GMC SI}+(T-d)\cdot\text{MCTI SI}}{2\cdot T} & \text{if } |d| \leq T \end{cases} \tag{5.16}$$

where T represents a threshold. In our experiments, T has been empirically set to 3. This method is referred to as 'SVMlin'.

Oracle fusion - This method is impractical, but it aims at estimating the upper

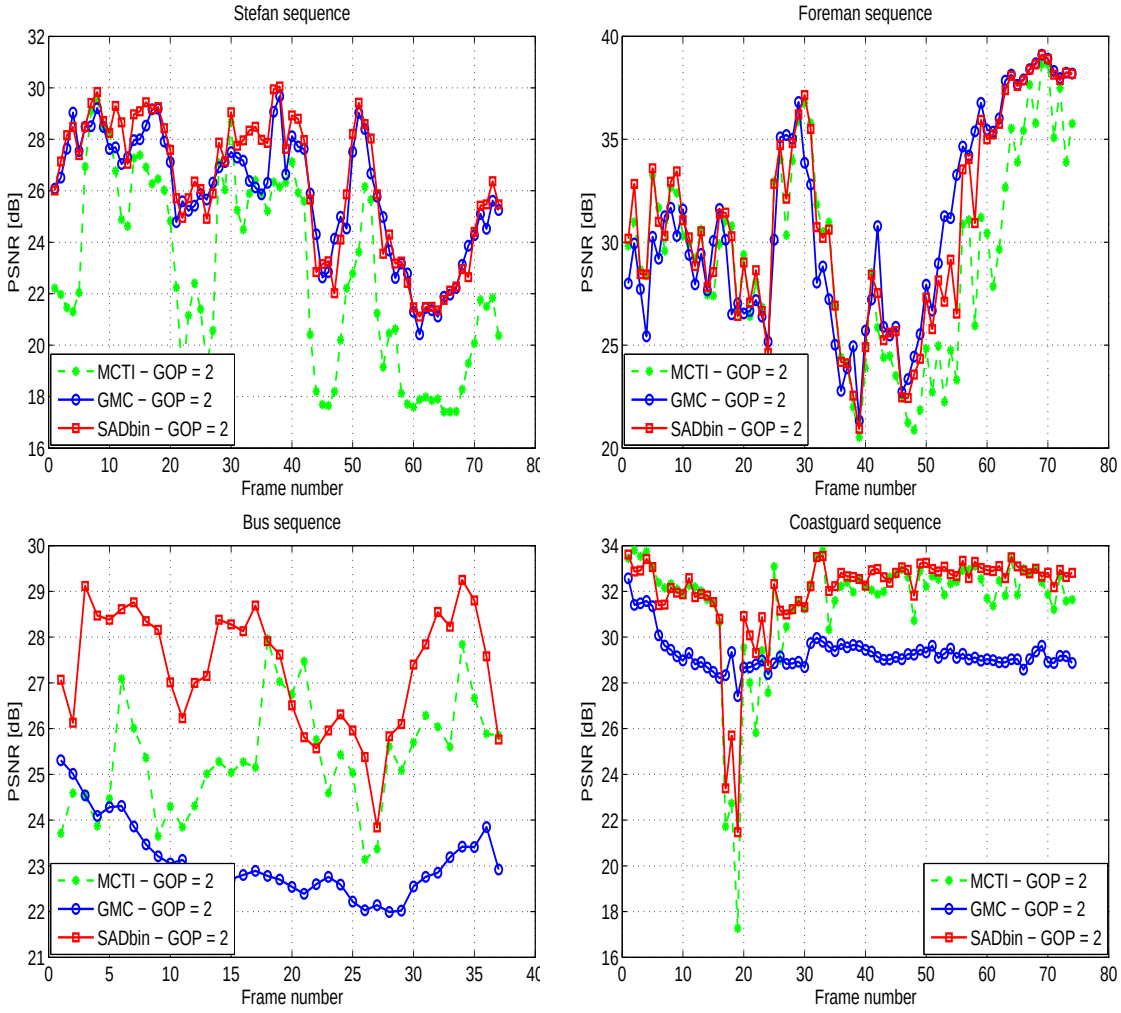

Figure 5.13: PSNR of MCTI SI, GMC SI, and the fusion of MCTI SI and GMC SI (SADbin) for Stefan, Foreman, Bus and Coastguard sequences for a GOP size of 2.

bound limit that can be achieved by combining GMC SI and MCTI SI, using the original WZF. This fusion is defined as follows:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } D_{\mathrm{GMC}} < D_{\mathrm{MCTI}} \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{5.17}$$

where D_{GMC} and D_{MCTI} are defined in Eq. 5.14. This method is referred to as 'Oracle'.

5.3.3 Experimental results

In order to evaluate the performance of the proposed methods, we performed extensive simulations, adopting the same test conditions as described in DISCOVER [4, 5], *i.e.* test video sequences are at QCIF spatial resolution and sampled at 15 frames/sec. The obtained results of the proposed methods SADbin (Eq. 5.9), SADlin (Eq. 5.10), SVMbin (Eq. 5.15) and SVMlin (Eq. 5.16) are compared to the DISCOVER codec, to the GMC,

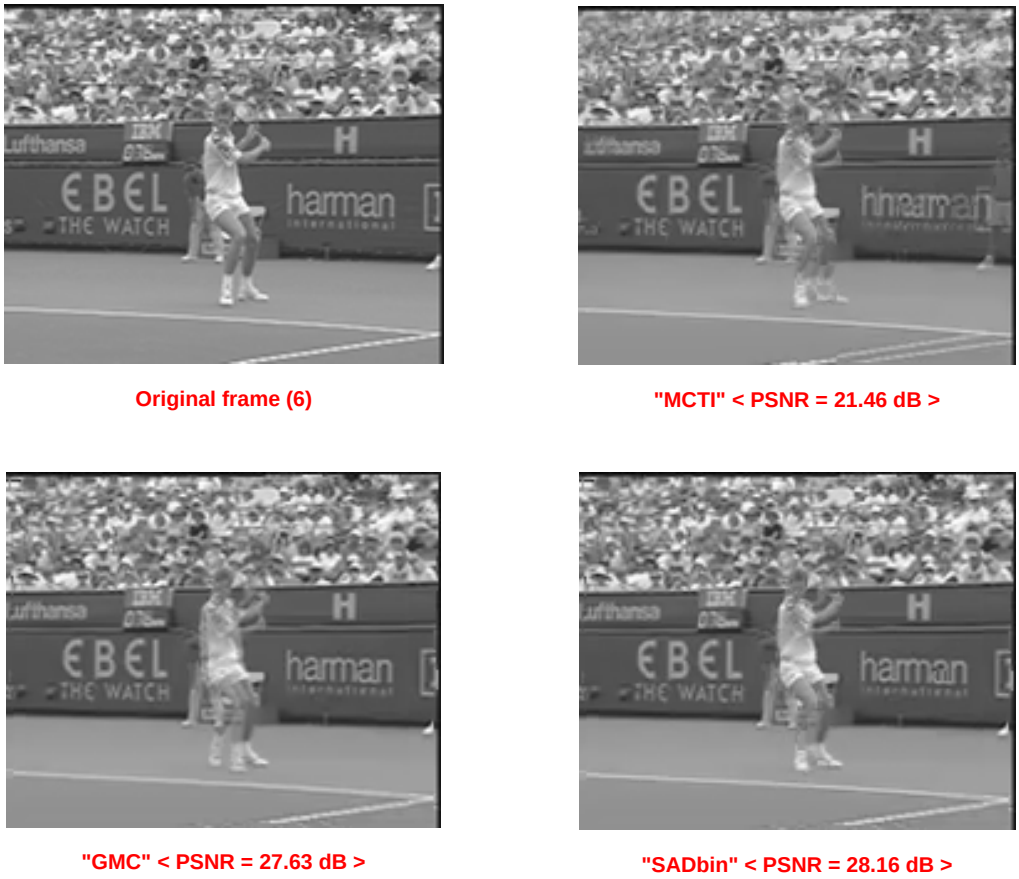

Figure 5.14: Visual quality of the original WZF(top-left), the SI obtained by MCTI SI (top-right), by GMC SI (bottom-left), and by the fusion of MCTI SI and GMC SI (bottom-right), for frame number 6 of Stefan sequence.

and to 'Oracle' fusion (Eq. 5.17).

SI performance assessment

Fig. 5.13 shows the SI PSNR of MCTI SI, GMC SI, and the fusion of the two SI frames (SADbin) for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 2. For Stefan and Foreman sequences, the quality of the GMC SI is better than that of the MCTI SI, in most cases. However, for Bus and Coastguard sequences, the MCTI SI is, most of the time, better than the GMC SI. It is clear that the fusion of global and local motion estimations can achieve the best quality SI in most frames, for all sequences. When the gap between the quality of MCTI SI and GMC SI is high, the fusion method can significantly reduce the gap (i.e. in such conditions, the fusion achieves a performance very close to the best case).

Fig. 5.14 shows the visual quality of the SI for frame number 6 of Stefan sequence. The SI obtained by DISCOVER codec (MCTI) contains block artifacts (top-right - 21.46 dB).

Figure 5.15: Visual quality comparisons among the original frame (top-left), the SI obtained by MCTI SI (top-right), GMC SI (bottom-left), and the fusion of MCTI SI and GMC SI (bottom-right), for frame number 14 of Bus sequence.

On the contrary, the SI obtained by the GMC technique is free from these artifacts (bottom-left - 27.63 dB). However, the foreground object Stefan is poorly estimated in GMC SI. The SI improvement obtained with our fusion technique SADbin (bottom-right - 28.16 dB) is 6.7 dB compared to MCTI. Fig. 5.15 shows the visual quality of the SI for frame number 14 of Bus sequence. In this case, the SI obtained by MCTI is better than the one obtained using GMC. However, the fusion of global and local motion estimations (SADbin) achieves a gain up to 2.75 dB, compared to MCTI, for this frame.

Figure 5.16 shows the percentage of blocks that are taken from the MCTI SI, the GMC SI, and the border (due to camera motion) during the fusion of global and local motion estimations for Stefan, Foreman, Bus, and Coastguard sequences. The border corresponds to the blocks that are taken from only one side in the estimation of GMC SI (from the backward or forward globally compensated reference frame), e.g. when this block is black in the backward (or forward) GMC SI due to camera motion. The percentage of MCTI SI and GMC SI in the fusion of global and local estimations depends on the sequence. It

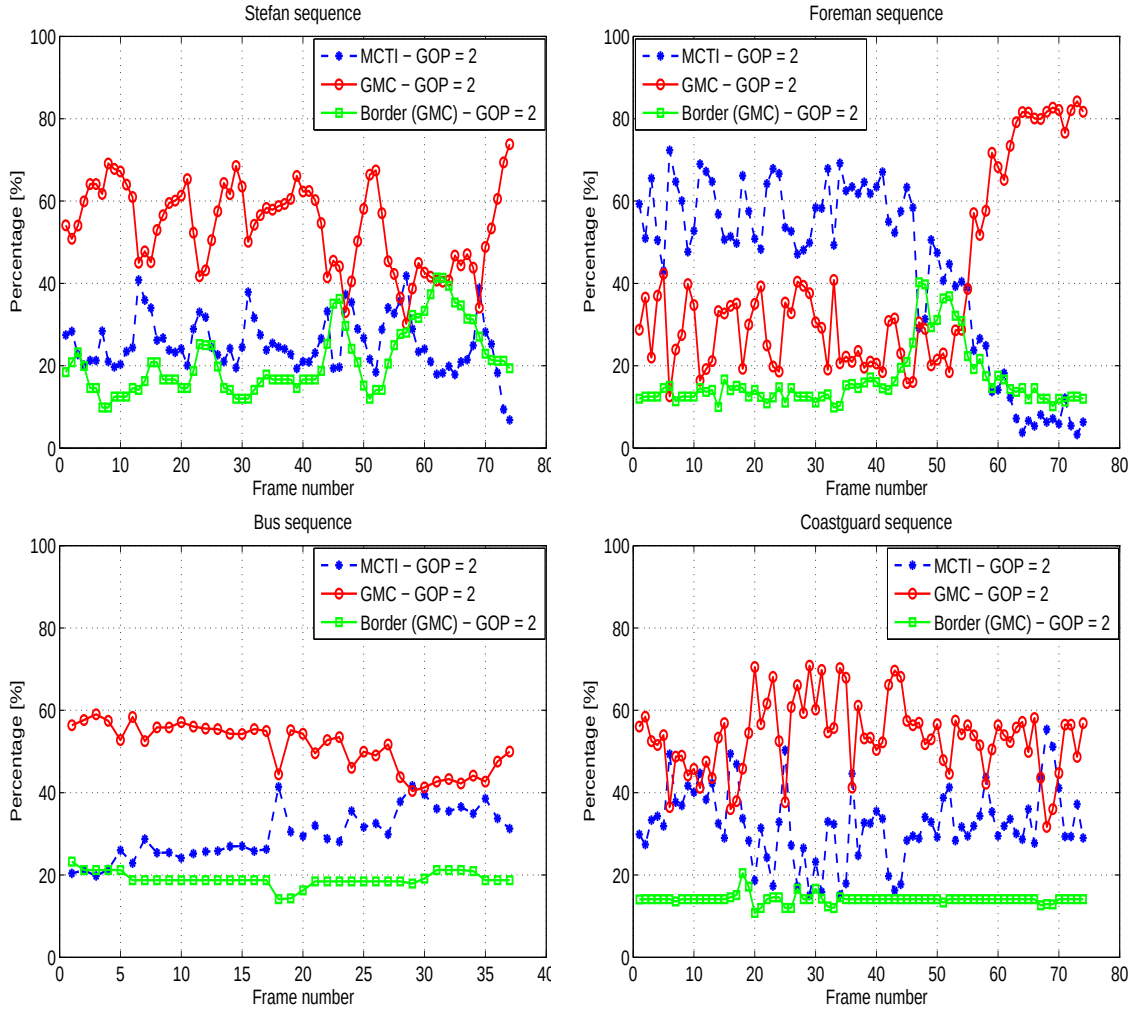

Figure 5.16: Percentage of blocks in the combination from MCTI SI, GMC SI, and the border for Stefan, Foreman, Bus and Coastguard sequences.

is clear that the percentage of the border increases with the amount of camera motion in the sequence.

Figs 5.17 and 5.18 show the original frame and the SI regions that are taken from the MCTI SI (white) and from GMC SI (black), for the frame number 16 of Stefan, Foreman, Bus, and Coastguard sequences. The gray color represents the blocks taken from GMC SI due to the camera motion. It is clear that most of the background blocks are taken from GMC SI (global motion), and that object blocks are taken from the MCTI SI (local motion).

Fig. 5.19 shows the PSNR of the SI estimated by MCTI and by the proposed methods SADbin and SVMbin, for Foreman sequence, for GOP sizes of 2 and 4. The proposed method SVMbin allows a significant gain compared to MCTI SI and a consistent improvement compared to the proposed fusion SADbin, where the gain reaches 4.4 dB for some frames.

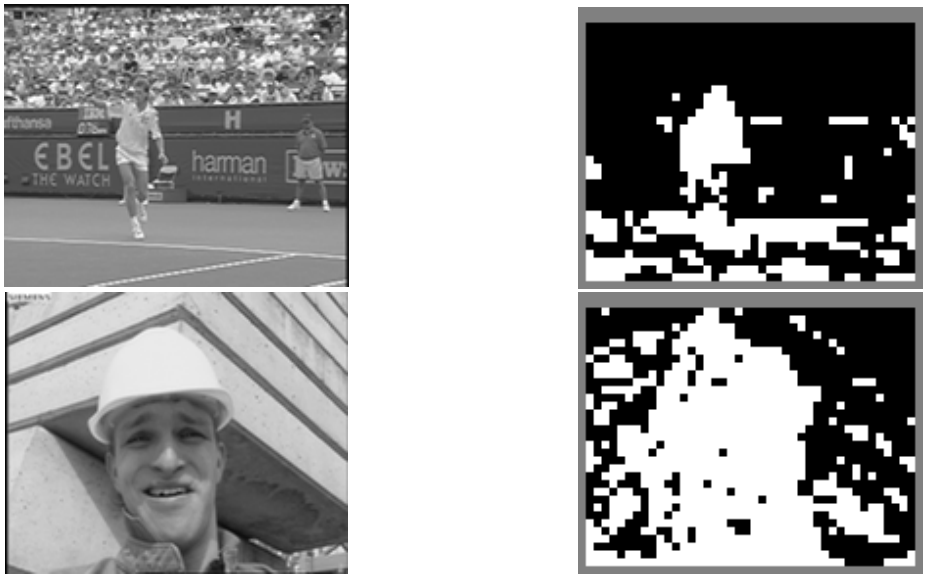

Figure 5.17: Frame number 16 of Stefan and Foreman sequences and the corresponding different regions in the fusion of MCTI SI and GMC SI. The white regions represent the blocks that are taken from MCTI SI, the black regions represent the blocks taken from GMC SI, and the gray regions represent the blocks corresponding to the border (camera motion).

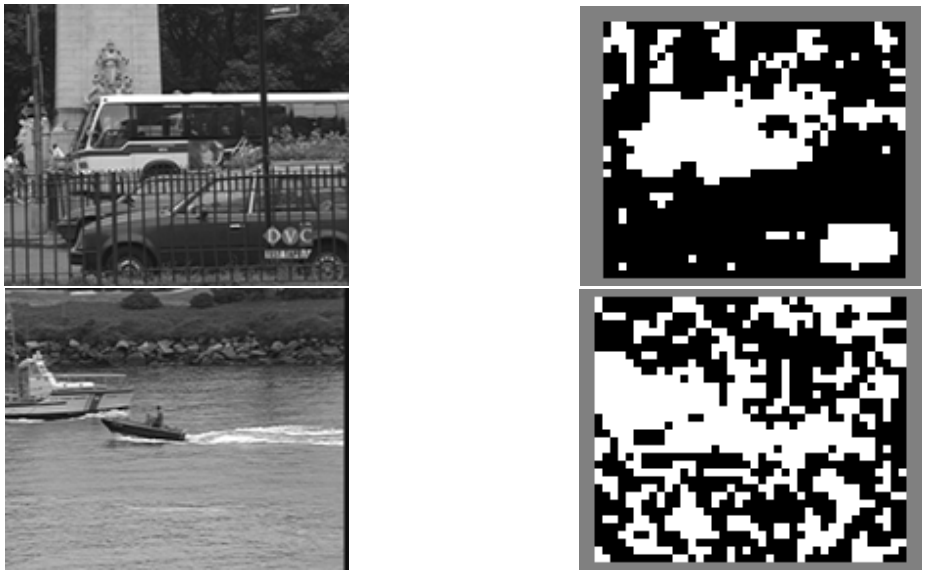

Figure 5.18: Frame number 16 of Bus and Coastguard sequences and the corresponding different regions in the fusion of MCTI SI and GMC SI. The white, black, and gray regions represent the blocks that are respectively taken from MCTI SI, GMC SI, and the border (camera motion).

Fig. 5.20 shows the visual difference of the SI for frame number of 125 of Foreman sequence, for a GOP size of 8. The SI obtained by MCTI technique has a poor quality, as shown in this figure (top-right - 18.33 dB). The proposed method SADbin can improve the quality of this SI, with a gain of 5.26 dB compared to MCTI SI. In addition, the SI

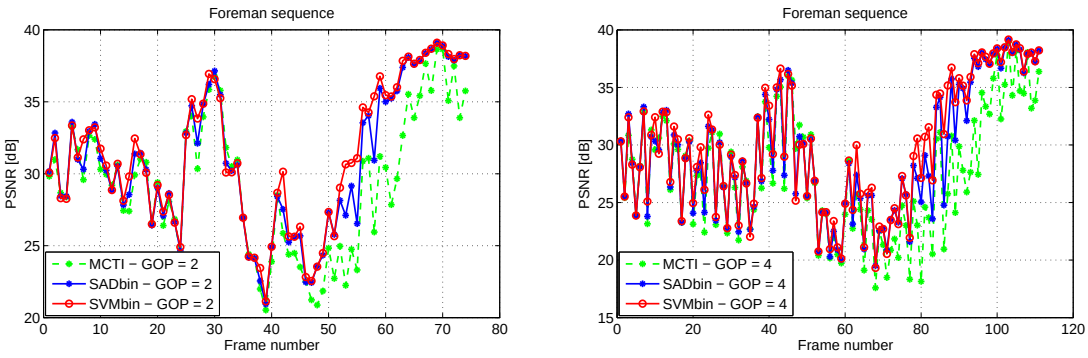

Figure 5.19: PSNR of MCTI SI and the proposed methods SADbin and SVMbin, for Foreman sequence, with GOP sizes of 2 and 4.

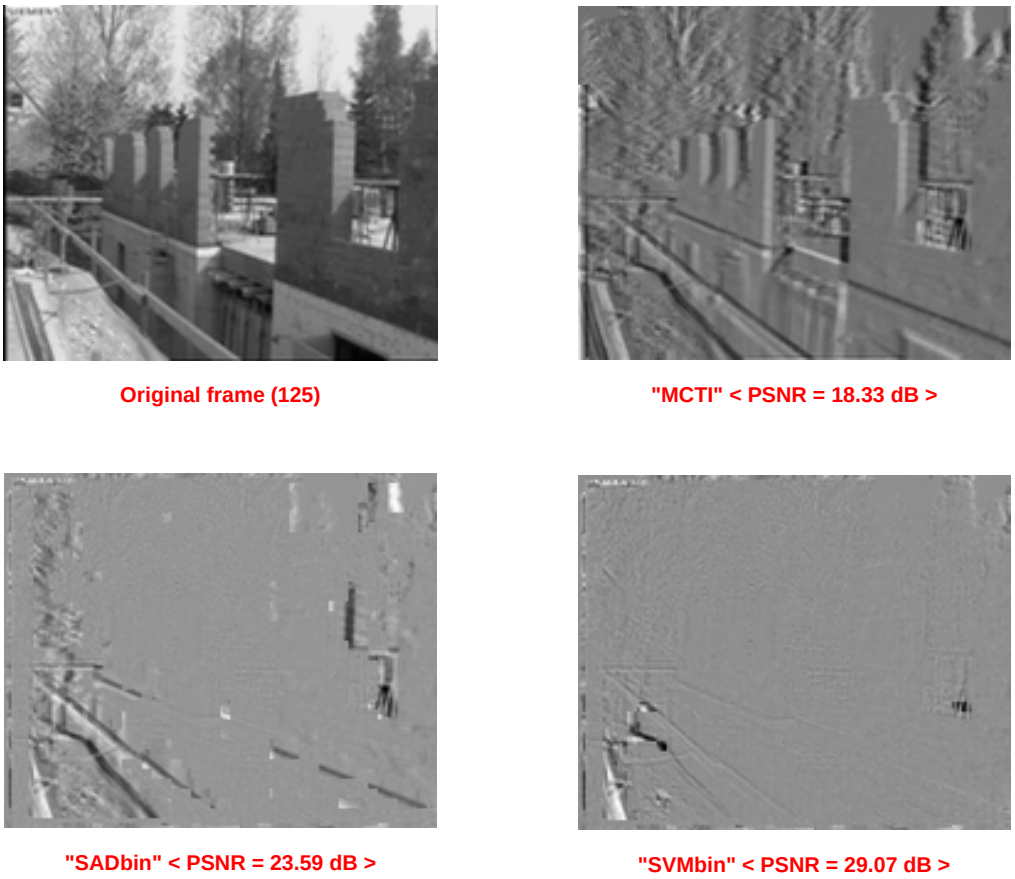

Figure 5.20: Visual difference of the SI estimated by MCTI, SAD-fusion, and the proposed SVM methods, for frame number 125 of Foreman sequence, for a GOP size of 8 (QI = 8).

obtained by the proposed method SVMlin is significantly better than the ones estimated by both MCTI and SADbin. The gain is 5.5 dB compared to the proposed method SADbin, for this frame.

Table 5.3 shows the average PSNR of the SI obtained with MCTI, GMC, SADbin,

Table 5.3: SI average PSNR for a GOP size equal to 2, 4, and 8 (QI = 8).

SI Average PSNR [dB]							
Method	**MCTI**	**GMC**	**SADbin**	**SADlin**	**SVMbin**	**SVMlin**	**Oracle**
GOP = 2							
Stefan	22.78	25.88	26.27	26.22	26.47	**26.55**	27.23
Foreman	29.38	30.70	30.82	31.02	31.24	**31.33**	31.94
Bus	25.37	23.10	27.30	27.19	27.25	**27.52**	28.34
Coastguard	31.47	29.28	32.00	31.92	32.08	**32.18**	32.57
GOP = 4							
Stefan	21.44	25.27	25.31	25.24	25.59	**25.66**	26.48
Foreman	27.64	29.62	29.27	29.51	29.81	**29.91**	30.75
Bus	24.00	22.53	26.27	26.23	26.23	**26.49**	27.30
Coastguard	29.91	28.19	30.76	30.74	30.88	**31.00**	31.43
GOP = 8							
Stefan	20.78	24.85	24.76	24.70	25.05	**25.13**	26.00
Foreman	26.29	28.62	28.09	28.33	28.71	**28.81**	29.71
Bus	22.95	21.95	25.26	25.26	25.22	**25.47**	26.24
Coastguard	28.82	27.50	29.85	29.85	29.98	**30.09**	30.58

SADlin, SVMbin, and SVMlin, for Stefan, Foreman, Bus, and Coastguard sequences for GOP sizes of 2, 4, and 8. It is clear that the proposed fusion methods SADbin, SADlin, SVMbin, and SVMlin can improve the quality of the SI compared to MCTI for all test sequences and all GOP sizes. The proposed technique SVMlin leads to the best SI quality for all test sequences.

Rate-Distortion performance

The RD performance of the proposed methods GMC, SADbin, SADlin, SVMbin, and SVMlin is shown with the Oracle fusion for the Stefan, Bus, Foreman, and Coastguard sequences in Table 5.4, in comparison to the DISCOVER codec, using the Bjontegaard metric [11], for GOP sizes of 2, 4, and 8. The first column represents the performance of the GMC scheme, *i.e.*, the SI is only generated using the global motion estimation, the second and third columns represent the performance of the binary and linear combinations of the global and local estimations respectively, and the fourth and fifth columns represent the performance of the binary and linear fusions of global and local SI using SVM. The last column represents the performance of the Oracle fusion which consists in combining the global and local motion estimations based on the original WZF. The Oracle performance is shown as an upper bound limit in order to assess the efficiency of the proposed fusion method.

The GMC method can achieve a significant gain compared to DISCOVER codec for Stefan and Foreman sequences, for a GOP size of 2, and for Stefan, Foreman, and Bus sequences, for GOP sizes of 4 and 8. For Stefan sequence, the gain compared to DISCOVER codec reaches 1.64, 3.21, and 3.96 dB for GOP sizes of 2, 4, and 8 respectively.

The proposed fusion methods SADbin, SADlin, SVMbin, and SVMlin can achieve an important gain compared to DISCOVER codec for all test sequences and all GOP sizes. The proposed method SVMlin always performs better than the other fusion methods for

Table 5.4: Rate-distortion performance gain for *Stefan*, *Foreman*, *Bus*, and *Coastguard* sequences towards DISCOVER codec, using Bjontegaard metric, for a GOP size of 2, 4, and 8.

Method	**GMC**	**SADbin**	**SADlin**	**SVMbin**	**SVMlin**	**Oracle**
GOP = 2						
Stefan						
Δ_R (%)	-23.67	-22.39	-19.69	**-23.63**	-23.26	-25.46
Δ_{PSNR} [dB]	1.64	1.54	1.33	**1.65**	1.62	1.80
Foreman						
Δ_R (%)	-8.51	-7.51	-8.71	-10.90	**-11.47**	-13.57
Δ_{PSNR} [dB]	0.52	0.46	0.53	0.68	**0.72**	0.86
Bus						
Δ_R (%)	6.14	-12.10	-9.28	-12.17	**-12.75**	-15.71
Δ_{PSNR} [dB]	-0.34	0.76	0.57	0.76	**0.80**	1.00
Coastguard						
Δ_R (%)	10.02	-4.40	-3.01	-4.90	**-5.24**	-7.43
Δ_{PSNR} [dB]	-0.47	0.22	0.15	0.25	**0.26**	0.38
GOP = 4						
Stefan						
Δ_R (%)	**-42.38**	-39.59	-34.48	-41.59	-40.80	-44.52
Δ_{PSNR} [dB]	**3.21**	2.94	2.45	3.16	3.07	3.44
Foreman						
Δ_R (%)	-21.89	-15.14	-17.62	-22.59	**-23.44**	-28.50
Δ_{PSNR} [dB]	1.35	0.90	1.06	1.41	**1.47**	1.84
Bus						
Δ_R (%)	-1.09	-23.60	-20.05	-23.83	**-24.59**	-29.14
Δ_{PSNR} [dB]	0.07	1.55	1.29	1.58	**1.63**	1.97
Coastguard						
Δ_R (%)	8.53	-13.26	-11.18	-14.89	**-15.49**	-20.08
Δ_{PSNR} [dB]	-0.35	0.58	0.48	0.66	**0.69**	0.91
GOP = 8						
Stefan						
Δ_R (%)	**-49.34**	-46.05	-40.30	-48.27	-47.60	-51.89
Δ_{PSNR} [dB]	**3.96**	3.61	3.00	3.89	3.80	4.26
Foreman						
Δ_R (%)	-30.51	-20.88	-23.41	-30.36	**-31.25**	-36.90
Δ_{PSNR} [dB]	1.99	1.28	1.45	1.98	**2.05**	2.51
Bus						
Δ_R (%)	-8.60	-28.23	-25.10	-28.64	**-29.67**	-34.63
Δ_{PSNR} [dB]	0.54	1.97	1.71	2.01	**2.09**	2.49
Coastguard						
Δ_R (%)	-2.37	-22.47	-20.19	-25.02	**-25.69**	-31.72
Δ_{PSNR} [dB]	0.10	1.04	0.92	1.18	**1.21**	1.55

Foreman, Bus, and Coastguard sequences, for all GOP sizes. For Stefan sequence, SVMbin achieves the best performance among all fusion methods, for all GOP sizes.

It is clear that the performance of the proposed fusion techniques SVMbin and SVMlin becomes closer to that of 'Oracle' fusion, compared to DISCOVER, for all test sequences. Their gap towards the 'Oracle' is smaller than 0.5 dB for all GOP sizes.

The gains obtained with our fusion techniques become even more significant for a GOP size equal to 8. In fact, for SVMbin, we obtain a bit rate reduction of up to -48.27%, which corresponds to an improvement of 3.89 dB on the decoded frames w.r.t. DISCOVER codec, for Stefan sequence. For Foreman sequence, SVMlin allows a gain of up to 2.05 dB, with a rate reduction of 31.25%, compared to the DISCOVER codec.

For Soccer sequence, the fusion of MCTI SI and GMC SI does not allow a gain compared to MCTI SI, due to the fact that global motion estimation does not improve the

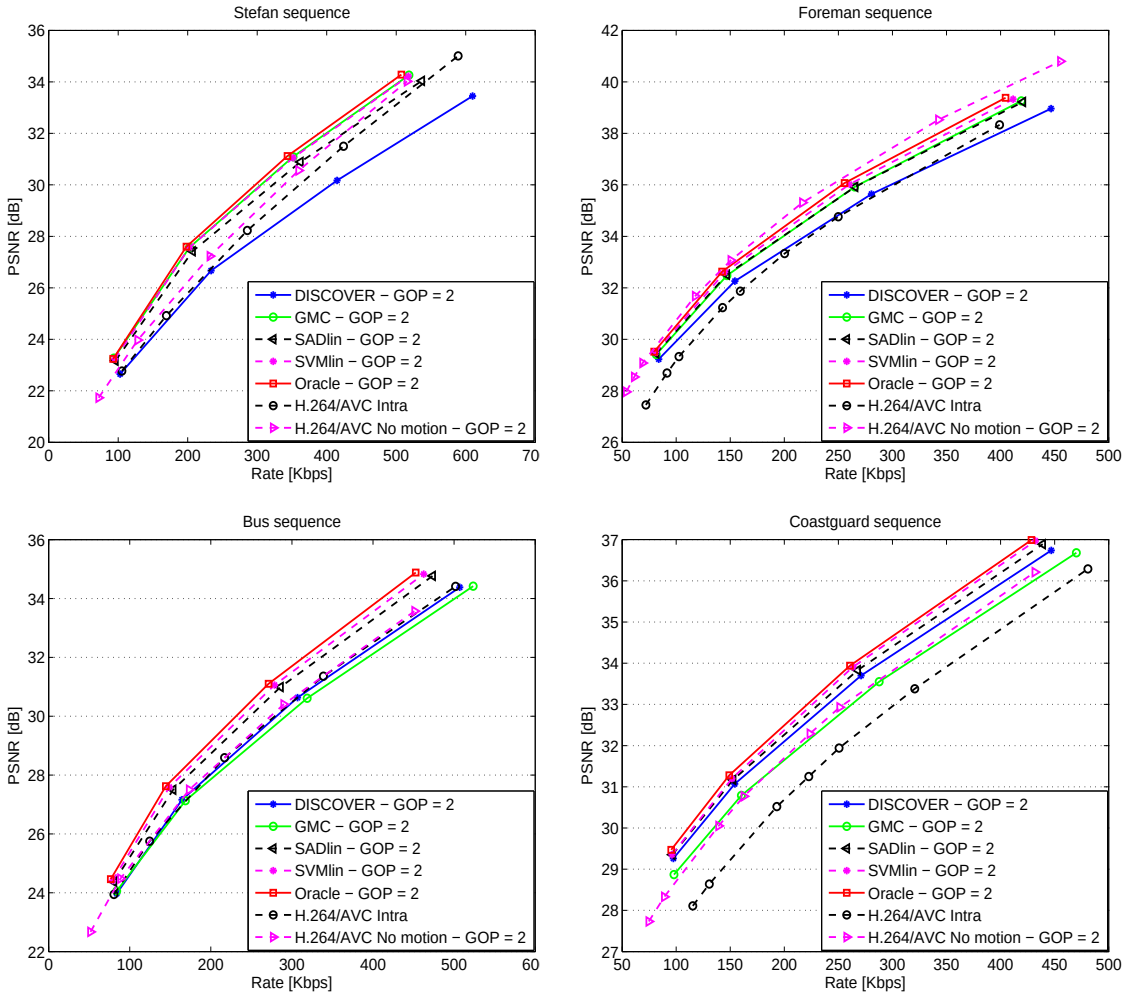

Figure 5.21: RD performance comparison among DISCOVER, GMC, SADlin, SVMlin, H.264/AVC Intra and H.264/AVC No motion for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 2.

prediction quality.

Figs. 5.21 and 5.22 show the RD performance performance of the DISCOVER codec, GMC, SADlin, SVMlin, Oracle, H.264/AVC Intra, and H.264/AVC No motion for Stefan, Foreman, Bus and Coastguard sequences for GOP sizes of 2 and 8 respectively. The proposed method SVMlin can achieve a gain compared to H.264/AVC No motion for Stefan, Bus, and Coastguard sequences. In addition, it can reduce the gap with H.264/AVC No motion for Foreman sequence.

Complexity assessment

The complexity of the SIFT algorithm and the matching process increases with the number of feature points and therefore depends on the video content. However, given that original frames are used for global motion estimation, the complexity of the SIFT algorithm is

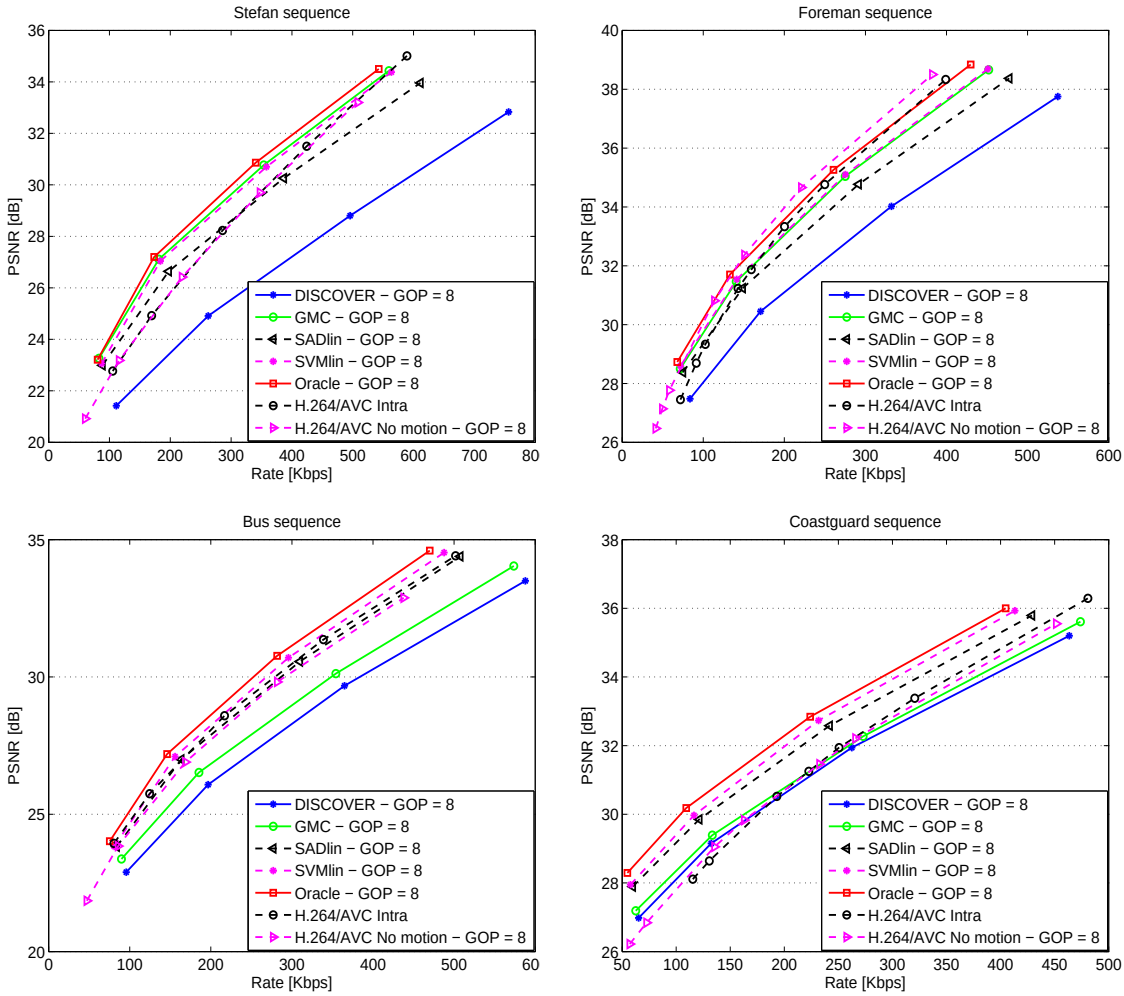

Figure 5.22: RD performance comparison among DISCOVER, GMC, SADlin, SVMlin, H.264/AVC Intra and H.264/AVC No motion for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 8.

independent of the RD operating point.

For 60 to 120 feature points, typical for sequences such as Foreman or Coastguard, the encoding complexity is increased by 15 to 30 % compared to the DISCOVER codec. In [76], it is shown that the encoding complexity of WZFs is about 1/6 the average encoding complexity of H.264/AVC Intra or H.264/AVC No motion. Therefore, despite the complexity increase due to SIFT, the encoding time for the proposed scheme remains lower than the one for H.264/AVC Intra or H.264/AVC No motion.

With the proposed DVC scheme or DISCOVER, the encoding complexity saving compared to conventional H.264/AVC Intra or No motion coding increases with the GOP size, as fewer KFs are used. However, DISCOVER tends to perform very poorly at a large GOP size, making such operating points less attractive. In contrast, the proposed scheme performs almost equally well at GOP sizes of 2, 4, and 8. Hence, our system makes the

use of large GOP sizes more appealing, since it allows for an important reduction in the encoding complexity compared to conventional coding techniques.

In order to further reduce the encoding complexity, Speeded Up Robust Features (SURF) [77] could be used instead of SIFT to extract feature points. Indeed, it has been shown that SURF achieves similar performance to SIFT with a greatly reduced complexity. Therefore, SURF could be effectively used at the encoder to extract feature points, allowing for a marginal increase in complexity compared to DISCOVER.

Finally, it should be noted that the execution time of the decoding process is significantly reduced due to the enhancement of the SI, which results in fewer requests through the feedback channel, despite the additional processing for global motion compensation.

5.3.4 Conclusion

First, a new technique for the fusion of global and local motion estimations is proposed in this chapter. This fusion is based on the SADs between the corresponding blocks in the global and local SI. Based on these differences, binary and linear combinations are performed.

Second, a new technique based on SVM for the fusion of global and local SI is investigated. Three features are defined and the SVM classifier gives a predicted value for each block in WZF. Based on the predicted values, we propose a binary and a linear fusion of the two SI frames.

Experimental results show that our proposed method can achieve a gain in RD performance up to 1.65 dB for a GOP size of 2 and 3.89 dB for longer GOP sizes, compared to DISCOVER codec, especially when the video sequence contains high global motion. The improvement becomes even more important as the GOP size increases.

5.4 Fusion enhancement during the decoding process

In this section, two different approaches are introduced to improve the fusion of GMC SI and MCTI SI (SADbin) during the decoding process. The first one consists in improving the fusion using the decoded DC coefficients and the PDWZF, after decoding the first DCT band. The second one consists in refining the MCTI SI during the decoding of the DCT bands, and, at the same time, successively improving the fusion between the two SI frames using the PDWZF.

5.4.1 Fusion enhancement after the decoding of the DC band

Once the decoded DC coefficients are obtained after decoding the first DCT band (*i.e.* DC band), a PDWZF (called PDWZF_{DC}) is reconstructed. The PDWZF_{DC} and the decoded DC coefficients are used to refine the fusion of the two SI frames (MCTI SI and GMC SI). Then, the improved SI is used to decode the remaining DCT coefficients, *i.e.*, the AC

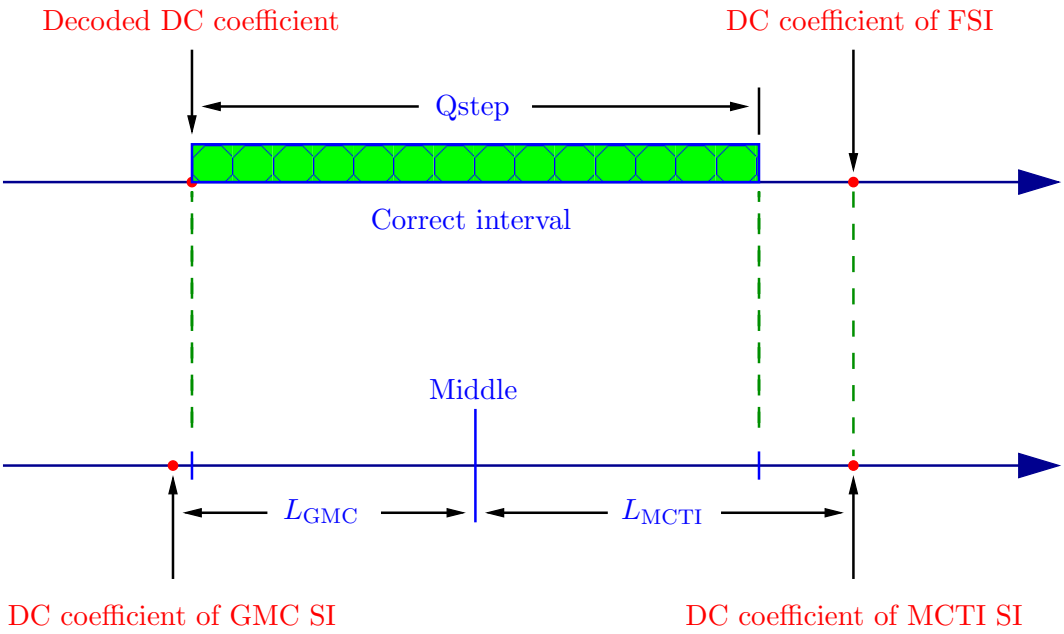

Figure 5.23: Fusion improvement using the decoded DC coefficients.

coefficients. This improved technique is motivated by the fact that the enhancement of the SI significantly reduces the amount of requested parity bits through the feedback channel, as well as the decoder processing time.

Here, two approaches are proposed to improve the combination of the global and local SI using the PDWZF_{DC} and the decoded DC coefficients. The first approach consists in exploiting the PDWZF_{DC} to enhance the fusion of the two estimations and it is referred to as 'FsPF'. The second approach aims at using the PDWZF_{DC} with the decoded DC coefficients to improve the combination of global and local SI frames. This approach is referred to as 'DCFsPF'.

FsPF method - For each 4×4 block, the SADs between the PDWZF_{DC} and the GMC SI and MCTI SI are computed as follows:

$$\begin{aligned} \text{SGMC}_1 &= \sum_{i=0}^{3}\sum_{j=0}^{3} |\text{GMC SI}(i+x_0, j+y_0) - \text{PDWZF}_{\text{DC}}(i+x_0, j+y_0)| \\ \text{SMCTI}_1 &= \sum_{i=0}^{3}\sum_{j=0}^{3} |\text{MCTI SI}(i+x_0, j+y_0) - \text{PDWZF}_{\text{DC}}(i+x_0, j+y_0)| \end{aligned} \tag{5.18}$$

where (x_0, y_0) is the coordinate of the origin of the current block. The fusion consists in choosing the most similar block in MCTI SI or GMC SI to the current block in PDWZF_{DC}.

In other words, the block that gives the smallest SAD is chosen for the next SI as follows:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } \mathrm{SGMC}_1 < \mathrm{SMCTI}_1 \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{5.19}$$

Then, the improved SI is used to decode the remaining DCT bands.

DCFsPF method - Recall that the WZF is transformed using a 4×4 block-based integer DCT. The DC coefficients are quantized using a quantization step Qstep. In order to improve the fusion, for each block in the current WZF, the decoded DC coefficient is compared to the DC coefficients of MCTI SI and GMC SI.

For the current block in the FSI (the first fusion of GMC SI and MCTI SI), let DD_{DC} be the decoded quantization DC coefficient. We refer to the quantization interval that corresponds to DD_{DC} by the term 'correct interval', as shown in Fig. 5.23. Let 'Middle' be the center of the correct interval:

$$\mathrm{Middle} = \mathrm{DD}_{DC} + \frac{\mathrm{Qstep}}{2} \tag{5.20}$$

Let GMC_{DC} and MCTI_{DC} be the DC coefficients of the GMC SI and MCTI SI transformed using a 4×4 block-based integer DCT respectively. The distances L_{GMC} and L_{MCTI} between the Middle and GMC_{DC} and MCTI_{DC} are computed using:

$$\begin{aligned} L_{\mathrm{GMC}} &= |Middle - \mathrm{GMC}_{DC}| \\ L_{\mathrm{MCTI}} &= |Middle - \mathrm{MCTI}_{DC}| \end{aligned} \tag{5.21}$$

The FSI enhancement technique is described by several steps as follows:

if $L_{\mathrm{GMC}} < L_{\mathrm{MCTI}}$ and $\mathrm{SGMC}_1 < \mathrm{SMCTI}_1$
- The fusion for this block is selected from GMC SI

otherwise

if $L_{\mathrm{MCTI}} < L_{\mathrm{GMC}}$ and $\mathrm{SMCTI}_1 < \mathrm{SGMC}_1$
- The fusion for this block is selected from MCTI SI

otherwise
- The fusion for this block is the average of GMC SI and MCTI SI

In both FsPF and DCFsPF techniques, the improved SI is used to decode the remaining DCT bands b_k ($k > 1$). In these approaches, the first fusion of GMC SI and MCTI SI is improved once after decoding the first band b_1 using the decoded coefficients and the $\mathrm{PDWZF}_{\mathrm{DC}}$. In addition, the correlation noise model is updated according to the improvement of the fusion of MCTI and GMC SI.

5.4.2 Fusion enhancement after each decoded DCT band

First, a simple approach is proposed for improving the fusion of GMC SI and MCTI SI, after decoding each DCT band, using the PDWZF. For this purpose, let PDWZF_k be the PDWZF obtained after the decoding of the kth band b_k. In this approach, the SADs between PDWZF_k and GMC SI and MCTI are computed for each 4×4 block:

$$\begin{aligned} \text{S}_{\text{GMC}} &= \sum_{i=0}^{3}\sum_{j=0}^{3} |\text{GMC SI}(i+x_0, j+y_0) - \text{PDWZF}_k(i+x_0, j+y_0)| \\ \text{S}_{\text{MCTI}} &= \sum_{i=0}^{3}\sum_{j=0}^{3} |\text{MCTI SI}(i+x_0, j+y_0) - \text{PDWZF}_k(i+x_0, j+y_0)| \end{aligned} \tag{5.22}$$

Based on S_{GMC} and S_{MCTI}, the fusion is improved according to the block which gives the smallest SAD as follows:

$$\text{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } \text{S}_{\text{GMC}} < \text{S}_{\text{MCTI}} \\ \text{MCTI SI} & \text{otherwise} \end{cases} \tag{5.23}$$

The enhancement of the fusion is carried out after each decoded DCT band without changing the initial estimations GMC SI and MCTI SI during the decoding process. This approach is referred to as **'FsPFAll'**.

On the contrary, the second approach consists in improving MCTI SI during the decoding process while the GMC SI remains unchanged. Afterwards, the fusion of GMC SI with the improved MCTI SI is performed after decoding each DCT band. MCTI enhancement consists in re-estimating the suspect vectors by integrating the algorithm that we formerly proposed in Chapter 3, due to its high performance. This algorithm is applied after the decoding of each DCT band. Furthermore, the fusion between the global and local motion estimations is carried out after each improvement of the local motion estimation using the PDWZF_k. This method will be referred to as **'FsIter'**.

Let MCTI SI_k be the MCTI SI used at the decoding of the band b_k (MCTI SI_1 is the initial MCTI SI). The algorithm consists in re-estimating the vectors suspected of being false. For a given block (8×8 pixels), the Mean of Absolute Differences (MAD) between the PDWZF_k and the MCTI SI_k is calculated and compared to a threshold T_1 as follows:

$$\text{MAD}(\text{MCTI SI}_k, \text{PDWZF}_k(\mathbf{MV})) < T_1, \tag{5.24}$$

where $\mathbf{MV} = (MV_x, MV_y)$ is the candidate motion vector. An extended block of 12×12 pixels is considered for calculating the MAD.

If Eq. (5.24) is not satisfied, the motion vector is identified as a suspicious vector and will be re-estimated. Otherwise, the motion vector **MV** for this block is only refined twice

within a small search area; the first time, after the decoding of the first DCT band and the second time after the decoding of all DCT bands. This step consists in relaxing the symmetric bidirectional motion vectors constrained in MCTI and allows a small refinement of those estimated motion vectors. In the simulations, we have set $T_1 = 4$ after preliminary tests, in such a way to achieve high performance with a low computational load.

The refinement of MCTI SI_k is applied during the decoding process by using this algorithm after decoding each DCT band. It starts by a first decoding of the FSI frame (*i.e.* the SI obtained after the first fusion of MCTI SI_1 and GMC SI) using the parity bits of the first DCT band. The reconstructed PDWZF_k is then used for refinement, together with the backward and forward reference frames. After each refinement step, the fusion of MCTI SI and GMC SI is applied using the PDWZF_k: For each block in the actual SI (4×4 pixels), the SADs between the PDWZF_k and MCTI SI_k and GMC SI are computed using a window of 8×8 pixels as follows:

$$\begin{aligned} \mathrm{S}_{\mathrm{GMC}_k} &= \sum_{i=-4}^{+3} \sum_{j=-4}^{+3} |\text{GMC SI}(i + x_0, j + y_0) - \mathrm{PDWZF}_k(i + x_0, j + y_0)| \\ \mathrm{S}_{\mathrm{MCTI}_k} &= \sum_{i=-4}^{+3} \sum_{j=-4}^{+3} |\text{MCTI SI}_k(i + x_0, j + y_0) - \mathrm{PDWZF}_k(i + x_0, j + y_0)| \end{aligned} \tag{5.25}$$

The fusion consists in choosing the block in MCTI SI_k or GMC SI that gives the smallest SAD to the current block in PDWZF_k:

$$\mathrm{SI}(b) = \begin{cases} \text{GMC SI} & \text{if } \mathrm{S}_{\mathrm{GMC}_k} < \mathrm{S}_{\mathrm{MCTI}_k} \\ \text{MCTI SI}_k & \text{otherwise} \end{cases} \tag{5.26}$$

The enhancement of MCTI SI is carried out after each decoded DCT band using this approach and the fusion is performed after each improvement.

5.4.3 Experimental results

To assess the performance of the proposed methods through simulations, we adopted, once again, the same test conditions as in DISCOVER [4, 5],. The obtained results are compared to the DISCOVER codec, the Alg. II (Chapter 3), the H.264/AVC Intra (Main profile), H.264/AVC No motion (*i.e.* all motion vectors are zero), and H.264/AVC with Inter prediction and motion estimation in Main profile exploiting temporal redundancy in an IB...IB... structure.

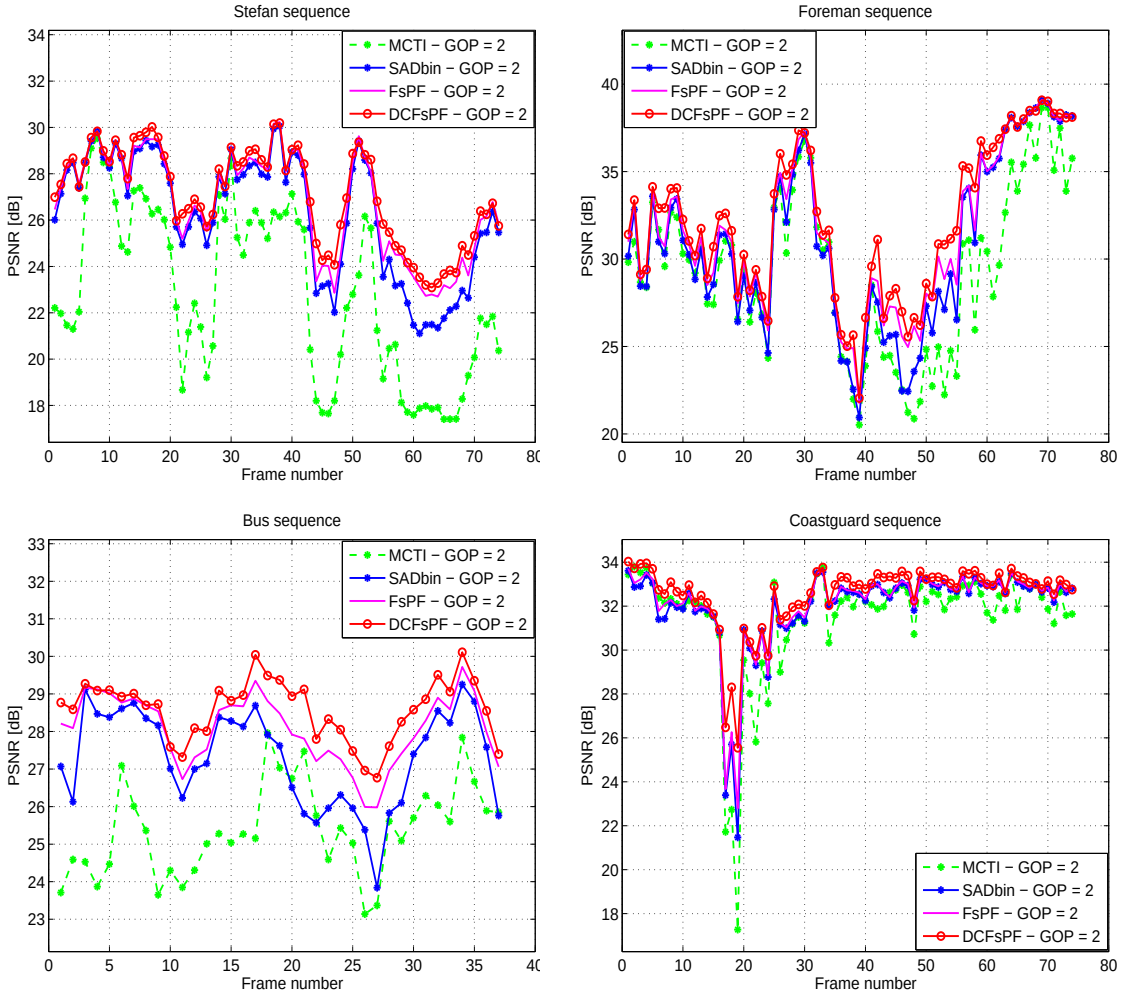

Figure 5.24: PSNR of MCTI SI, the fusion of MCTI SI and GMC SI (SADbin), FsPF, and DCFsPF, for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 2.

Table 5.5: SI average PSNR for a GOP size equal to 2, 4, and 8 (QI = 8).

SI Average PSNR [dB]								
Method	**MCTI**	**GMC**	**SADbin**	**FsPF**	**DCFsPF**	**FsPFAll**	**Alg. II**	**FsIter**
GOP = 2								
Stefan	22.78	25.88	26.27	26.76	**27.09**	27.52	26.61	**28.85**
Foreman	29.38	30.70	30.82	31.55	**32.18**	32.65	34.45	**35.56**
Bus	25.37	23.10	27.30	28.07	**28.59**	29.20	27.78	**29.53**
Coastguard	31.47	29.28	32.00	32.11	**32.53**	33.16	32.35	**33.46**
GOP = 4								
Stefan	21.44	25.27	25.31	25.92	**26.32**	26.81	25.97	**28.33**
Foreman	27.64	29.62	29.27	30.22	**30.94**	31.46	33.63	**34.76**
Bus	24.00	22.53	26.27	27.21	**27.77**	28.36	27.33	**29.04**
Coastguard	29.91	28.19	30.76	30.93	**31.41**	31.95	31.49	**32.53**
GOP = 8								
Stefan	20.78	24.85	24.76	25.47	**25.91**	26.39	25.63	**28.00**
Foreman	26.29	28.62	28.09	29.14	**29.85**	30.38	32.93	**34.04**
Bus	22.95	21.95	25.26	26.36	**26.94**	27.51	26.69	**28.57**
Coastguard	28.82	27.50	29.85	30.07	**30.55**	31.05	30.88	**31.84**

SI performance assessment

Fig. 5.24 shows the SI PSNR of the SI estimated by MCTI, the fusion of MCTI and GMC SI (SADbin), the improvement of the fusion after decoding the first band using PDWZF (FsPF), and the enhancement of the fusion after decoding the first band using PDWZF with the decoded DC coefficients (DCFsPF), for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 2. It is clear that the methods FsPF and DCFsPF can improve the quality of the SI for all test sequences. As we can see, the exploitation of the decoded DC coefficients with the PDWZF (DCFsPF) allows a gain compared to FsPF.

Table 5.5 shows the average PSNR of the SI obtained with MCTI, GMC, SADbin, FsPF, DCFsPF, FsPFAll, Alg. II, and FsIter, for Stefan, Foreman, Bus, and Coastguard sequences, for GOP sizes of 2, 4, and 8. It is clear that the proposed methods can significantly improve the quality of the SI compared to the first fusion SADbin, for all test sequences and all GOP sizes. The proposed technique FsIter leads to the best SI quality for all test sequences.

Rate-Distortion performance

The RD performance of the different methods is shown in Table 5.6 for Stefan, Bus, Foreman and Coastguard sequences, with GOP sizes of 2, 4, and 8. All the proposed methods can achieve a gain compared to the first fusion SADbin. DCFsPF allows a gain up to 2.21 dB, compared to DISCOVER, with a rate reduction up to 33.21 %, for Foreman sequence, with a GOP size of 8. On the other side, the first fusion achieves a gain of 1.27 dB with a rate reduction of 20.88 %. Thus, the DCFsPF method can improve the fusion by using the decoded DC coefficients and the PDWZF_1, especially when the gap between the first fusion and the Oracle fusion is high (refer to the results of Foreman and Coastguard sequences in Table 5.4). Moreover, the DCFsPF method is very light in terms of computational load.

The proposed method FsIter can achieve a significant gain compared to DISCOVER codec and Alg. II, for all sequences and all GOP sizes. The gain reaches 4.59 dB with a rate reduction of 53.98 %, when Alg. II achieves a gain of 2.51 *dB* with a rate reduction of 34.13 %, for Stefan sequence, with a GOP size 8. For Foreman sequence, the gain obtained with FsIter becomes 3.65 dB and the rate reduction 47.86 %, facing 3.02 *dB* and 41.88 % for Alg. II. It can be seen that the FsIter method allows an important performance improvement compared to the first fusion of global and local motion estimation, especially for Foreman and Stefan sequences.

The RD of Alg. II, the proposed methods DCFsPF and FsIter, H.264/AVC Intra, H.264/AVC No motion and H.264/AVC Inter is shown in Table 5.7. Figs. 5.25 and 5.26 show the RD performance curves for Stefan, Bus, Foreman, and Coastguard sequences. The performance of the proposed method FsIter is always better than both H.264/AVC Intra and H.264/AVC No motion for all sequences and for all GOP sizes.

Table 5.6: Rate-distortion performance gain of Alg. II and the proposed methods for *Stefan*, *Bus*, *Foreman*, and *Coastguard* sequences w.r.t. DISCOVER codec, using Bjontegaard metric

Method	**Alg. II**	**SADbin**	**FsPF**	**DCFsPF**	**FsPFAll**	**FsIter**
GOP = 2						
Stefan						
Δ_R (%)	-14.06	-22.39	-22.92	**-23.45**	-23.87	**-26.51**
Δ_{PSNR} [dB]	0.93	1.54	1.59	**1.63**	1.67	**1.89**
Foreman						
Δ_R (%)	-16.29	-7.51	-8.99	**-12.24**	-11.39	**-19.68**
Δ_{PSNR} [dB]	1.05	0.46	0.55	**0.77**	0.71	**1.30**
Bus						
Δ_R (%)	-4.50	-12.10	-13.19	**-14.23**	-14.46	**-15.32**
Δ_{PSNR} [dB]	0.27	0.76	0.83	**0.90**	0.92	**0.98**
Coastguard						
Δ_R (%)	-2.24	-4.40	-4.69	**-6.07**	-6.03	**-7.68**
Δ_{PSNR} [dB]	0.11	0.22	0.24	**0.31**	0.31	**0.40**
GOP = 4						
Stefan						
Δ_R (%)	-27.84	-39.59	-40.55	**-41.68**	-42.25	**-46.11**
Δ_{PSNR} [dB]	1.93	2.94	3.03	**3.14**	3.22	**3.65**
Foreman						
Δ_R (%)	-32.65	-15.14	-19.28	**-25.28**	-23.99	**-38.12**
Δ_{PSNR} [dB]	2.19	0.90	1.17	**1.60**	1.52	**2.69**
Bus						
Δ_R (%)	-15.82	-23.60	-25.83	**-27.73**	-27.74	**-30.89**
Δ_{PSNR} [dB]	0.99	1.55	1.72	**1.87**	1.88	**2.13**
Coastguard						
Δ_R (%)	-11.94	-13.26	-14.30	**-17.59**	-16.57	**-21.60**
Δ_{PSNR} [dB]	0.52	0.58	0.63	**0.79**	0.74	**1.00**
GOP = 8						
Stefan						
Δ_R (%)	-34.13	-46.05	-47.44	**-48.99**	-49.58	**-53.98**
Δ_{PSNR} [dB]	2.51	3.61	3.76	**3.93**	4.03	**4.59**
Foreman						
Δ_R (%)	-41.88	-20.88	-26.20	**-33.21**	-31.52	**-47.86**
Δ_{PSNR} [dB]	3.02	1.28	1.66	**2.21**	2.10	**3.65**
Bus						
Δ_R (%)	-22.83	-28.23	-31.73	**-34.31**	-34.18	**-40.40**
Δ_{PSNR} [dB]	1.53	1.97	2.25	**2.47**	2.48	**3.04**
Coastguard						
Δ_R (%)	-24.21	-22.47	-23.93	**-28.53**	-26.98	**-34.51**
Δ_{PSNR} [dB]	1.14	1.04	1.12	**1.37**	1.29	**1.74**

Fig. 5.27 shows the performance of the proposed method FsIter in comparison to that of H.264/AVC Inter prediction with motion, for Stefan, Bus, Foreman, and Coastguard sequences. The gap between the performance of H.264/AVC Inter prediction with motion and the proposed method is reduced to a large extent, compared to previous techniques.

The performance of our proposed method is significantly better than the performance of [67], where the global motion is estimated at the decoder without any fusion with the local motion information. However, it should be noted that [67] uses a pixel-domain DVC. The proposed method in [74], where the global and local motion estimations are combined at the encoder, allows a gain up to 1 dB in the high bitrate range, and up to 0.5 dB in the low bitrate range, for Foreman sequence and a GOP size of 2. In comparison, our proposed method achieves an average gain of 1.37 dB for this sequence. In [72][73], where the local motion estimation task is shared between the encoder and the decoder, the RD

Table 5.7: Rate-distortion performance gain (w.r.t. DISCOVER codec) of Alg. II, H264, and the proposed methods, for *Stefan*, *Bus*, *Foreman*, and *Coastguard* sequences, using Bjontegaard metric.

Method	**Alg. II**	**SADbin**	**DCFsPF**	**FsIter**	**H.264/AVC Intra**	**H.264/AVC No motion**	**H.264/AVC Inter**
GOP = 2							
Stefan							
Δ_R (%)	-14.06	-22.39	-23.45	**-26.51**	-6.01	-12.10	-38.97
Δ_{PSNR} [dB]	0.93	1.54	1.63	**1.89**	0.42	0.72	3.18
Foreman							
Δ_R (%)	-16.29	-7.51	-12.24	**-19.68**	6.17	-16.77	-35.90
Δ_{PSNR} [dB]	1.05	0.46	0.77	**1.30**	-0.41	1.13	2.73
Bus							
Δ_R (%)	-4.50	-12.10	-14.23	**-15.32**	2.33	0.02	-31.23
Δ_{PSNR} [dB]	0.27	0.76	0.90	**0.98**	-0.13	-0.02	2.20
Coastguard							
Δ_R (%)	-2.24	-4.40	-6.07	**-7.68**	30.18	9.92	-17.15
Δ_{PSNR} [dB]	0.11	0.22	0.31	**0.40**	-1.44	-0.49	1.04
GOP = 4							
Stefan							
Δ_R (%)	-27.84	-39.59	-41.68	**-46.11**	-25.40	-27.20	-58.30
Δ_{PSNR} [dB]	1.93	2.94	3.14	**3.65**	1.78	1.77	5.22
Foreman							
Δ_R (%)	-32.65	-15.14	-25.28	**-38.12**	-12.28	-30.39	-58.93
Δ_{PSNR} [dB]	2.19	0.90	1.60	**2.69**	0.68	2.08	5.07
Bus							
Δ_R (%)	-15.82	-23.60	-27.73	**-30.89**	-12.18	-10.33	-49.87
Δ_{PSNR} [dB]	0.99	1.55	1.87	**2.13**	0.75	0.57	3.87
Coastguard							
Δ_R (%)	-11.94	-13.26	-17.59	**-21.60**	26.01	14.42	-35.73
Δ_{PSNR} [dB]	0.52	0.58	0.79	**1.00**	-1.04	-0.64	2.06
GOP = 8							
Stefan							
Δ_R (%)	-34.13	-46.05	-48.99	**-53.98**	-36.34	-36.13	-61.58
Δ_{PSNR} [dB]	2.51	3.61	3.93	**4.59**	2.78	2.56	5.64
Foreman							
Δ_R (%)	-41.88	-20.88	-33.21	**-47.86**	-28.48	-37.93	-67.93
Δ_{PSNR} [dB]	3.02	1.28	2.21	**3.65**	1.93	2.66	6.40
Bus							
Δ_R (%)	-22.83	-28.23	-34.31	**-40.40**	-27.19	-23.96	-49.98
Δ_{PSNR} [dB]	1.53	1.97	2.47	**3.04**	1.86	1.50	3.94
Coastguard							
Δ_R (%)	-24.21	-22.47	-28.53	**-34.51**	0.86	2.00	-50.31
Δ_{PSNR} [dB]	1.14	1.04	1.37	**1.74**	0.04	-0.11	3.01

performance is unfortunately not shown.

5.4.4 Conclusion

In this section, two different methods for improving the fusion during the decoding process are presented. The first method consists in exploiting the decoded DC coefficients and the PDWZF after the decoding of the first DCT band to improve the first fusion of MCTI SI and GMC SI. The second method aims at exploiting the PDWZF after the decoding of each DCT band to improve the MCTI SI, and enhance the fusion of GMC SI and MCTI SI after each improvement.

Experimental results show that our second method can achieve a gain in RD perform-

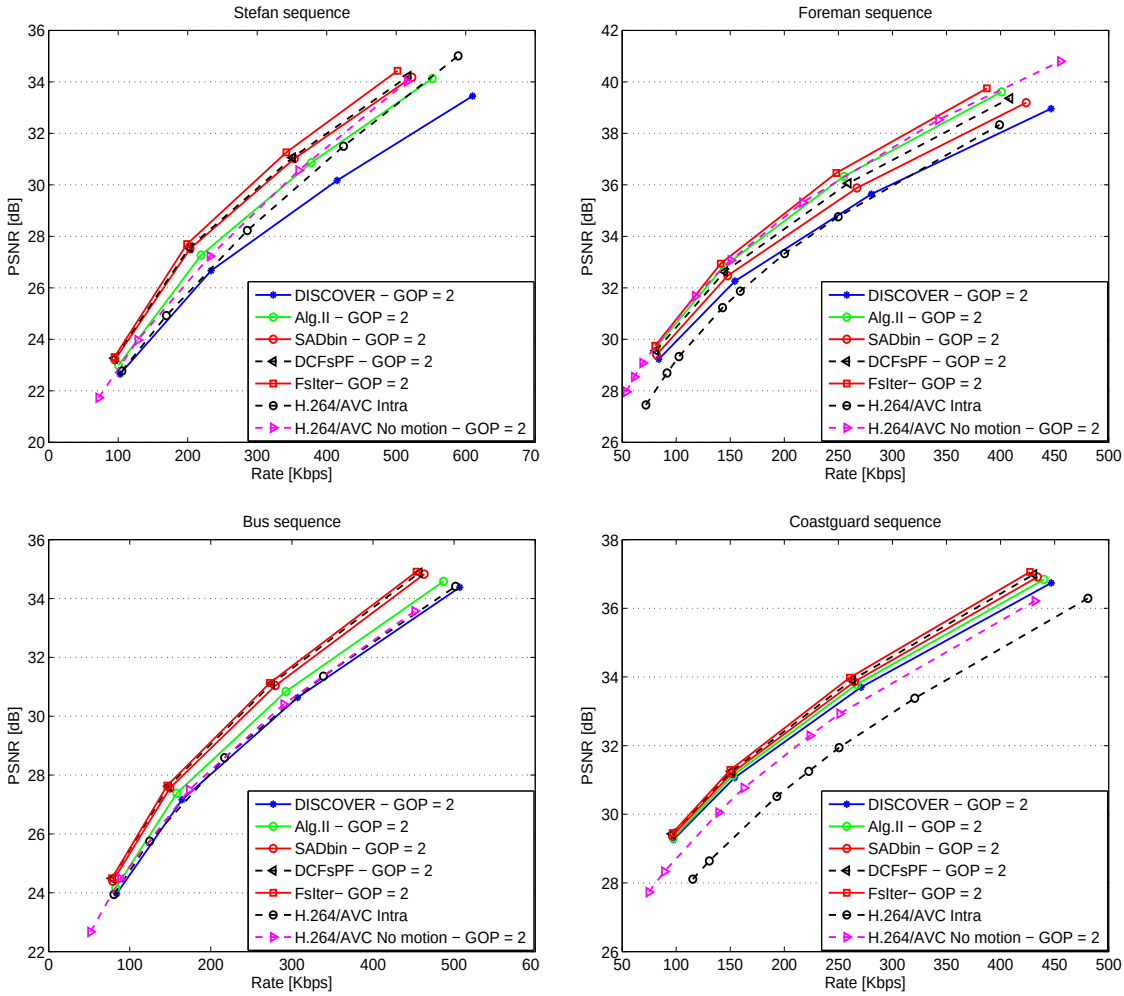

Figure 5.25: RD performance comparison - DISCOVER, Alg.II, SADbin, DCFsPF, FsIter, H.264/AVC Intra, and H.264/AVC No motion for Stefan, Bus, Foreman and Coastguard sequences, for a GOP size of 2.

ance up to 1.89 dB for a GOP size of 2, and 4.59 dB for longer GOP sizes, compared to DISCOVER codec, especially when the video sequence contains high global motion. The improvement becomes even more important as the GOP size increases.

With the second proposed method, DVC now outperforms H.264/AVC Intra and H.264/AVC No motion in all reported test conditions. Moreover, the performance gap between the proposed DVC scheme and H.264/AVC Inter prediction with motion is significantly reduced.

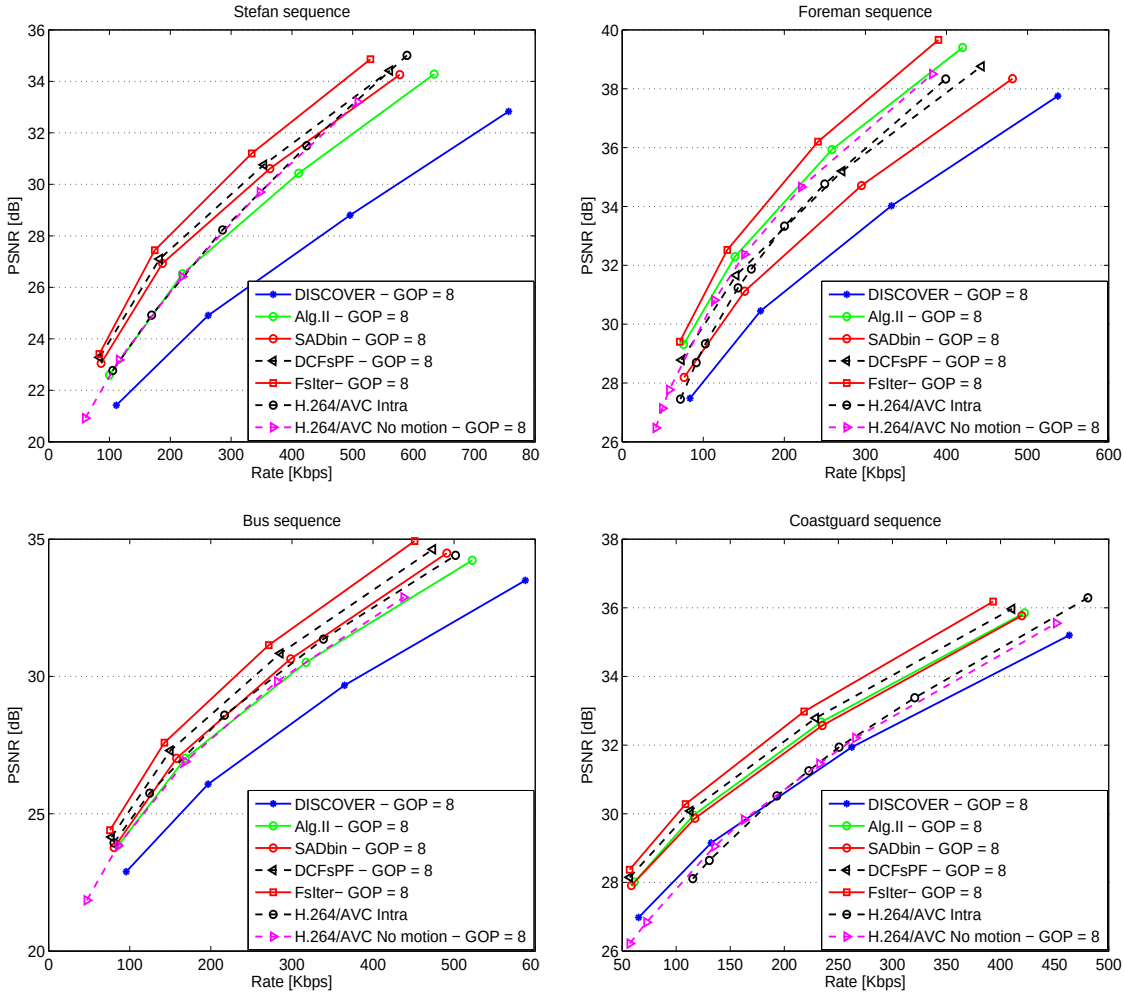

Figure 5.26: RD performance comparison - DISCOVER, Alg.II, SADbin, DCFsPF, FsIter, H.264/AVC Intra, and H.264/AVC No motion for Stefan, Bus, Foreman and Coastguard sequences, for a GOP size of 8.

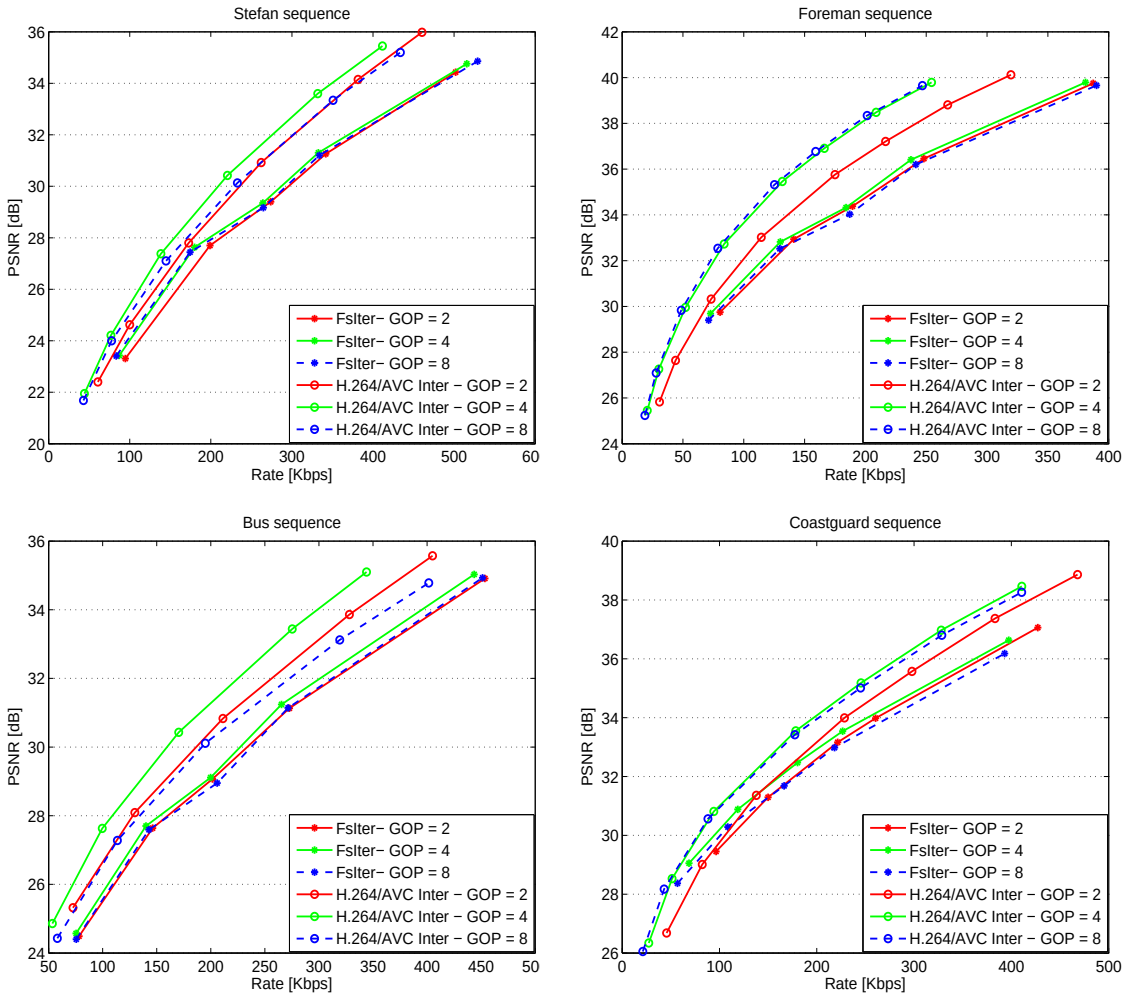

Figure 5.27: RD performance of the proposed method FsIter and H.264/AVC Inter prediction with motion for Stefan, Bus, Foreman, and Coastguard sequences, for all GOP sizes.

5.5 Conclusions

In this chapter, we first proposed an approach to generate a new SI based on global motion estimation and compensation (GMC). First, the feature points of the original reference frames and the original WZF are extracted at the encoder using the SIFT algorithm. Then, the matching between the feature points of the WZF and the backward (and forward) reference frames is carried out. In addition, we proposed an efficient algorithm to eliminate the feature matches that belong to the objects. This algorithm allows a robust estimation of the parameters of the global model characterized by the two transforms T_b and T_f between the WZF and the BRF and FRF respectively. These global parameters are encoded (each parameter on 15 bits) and sent to the decoder. Then, the transforms T_b and T_f are applied to the decoded backward and forward reference frames respectively, to generate the backward and forward GMC SI. Finally, an average of the two obtained frames is applied to obtain the GMC SI.

Second, we proposed different approaches to combine the global and local estimations at the decoder, in order to generate an improved SI. The local estimation is obtained using MCTI as in DISCOVER codec, and the global estimation is the GMC SI. To this aim, the SADs between the corresponding blocks in the global and local estimations are computed. Based on the obtained SADs, a binary or a linear combination of the two SI frames (GMC SI and MCTI SI) is performed. Afterwards, we proposed an approach for combining MCTI SI and GMC SI using a Support Vector Machine (SVM). Based on a constructed model, the SVM classifier provides a predicted value for each block in the SI. Then, a binary or a linear fusion of the GMC SI and MCTI SI is performed based on the predicted value. These fusion methods achieve a significant gain compared to DISCOVER codec, for all test sequences and all GOP sizes.

Finally, we proposed to improve the fusion of GMC SI and MCTI SI during the decoding process. In this context, a first approach consists in improving the fusion after decoding the first DCT band, using the PDWZF and the decoded DC coefficients. A second approach aims at refining the MCTI SI based on the PDWZF and the reference frames, after each decoded DCT band. Then, the fusion of GMC SI and the improved MCTI SI is performed using the PDWZF, after each enhancement of MCTI SI. These proposed approaches allow an important gain compared to DISCOVER codec, and a consistent gain compared to the first fusion of GMC SI and MCTI SI.

The material in this chapter has been published in:

1 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu “Fusion of global and local motion estimation for Distributed Video Coding ”, *IEEE transactions on circuits and systems for Video Technology*, (in press).

2 A. Abou-Elailah, F. Dufaux, J. Farah, and M. Cagnazzo “Fusion of Global and Local Side Information using Support Vector Machine in transform-domain DVC

", *European Signal Processing Conference (EUSIPCO)*, August 2012, Bucharest, Romania.

Chapter 6

Fusion based on foreground objects estimation

Contents

In this chapter, we propose new methods to combine the global and local motion estimations. The two estimations MCTI SI and GMC SI are generated at the decoder, using the reference frames and the global parameters estimated and transmitted by the encoder. Normally, the background pixels are assigned to the global motion and the foreground objects are assigned to the local motion. For this reason, the foreground objects are segmented from the backward and forward reference frames at the decoder side. Then, an estimation of the foreground objects in the SI is computed using the backward and forward foreground objects, while the background pixels are directly taken from GMC SI. However, when the backward and forward global estimation are averaged to generate the GMC SI, an artifact effect around the foreground objects can occur in GMC SI, due to the different motion of the foreground objects. This artifact effect appears in GMC SI when the average between the background and foreground pixels is carried out. Here, an approach is proposed to remove the artifact effect around the foreground objects in GMC SI using the backward and forward segmented foreground objects.

First, we propose a new method based on elastic curves [14, 18] in order to estimate the foreground objects masks in the estimated SI. Then, the pixels in the estimated masks

are selected from MCTI SI, while GMC SI is used to cover all the remaining pixels in the estimated SI. More specifically, the foreground objects masks are generated using the segmented foreground objects in the reference frames. Then, the foreground objects contours are constructed from the generated masks. Furthermore, the contours are considered as closed curves and the algorithm in [14] is used to generate the curves in the estimated SI using the curves of the reference frames. Finally, the masks are generated using the obtained curves.

Second, two different approaches are proposed for generating the foreground objects in the SI, based on local motion compensation. In the first approach, the MCTI technique is directly applied on the backward and forward foreground objects to generate the foreground objects in the SI. In the second approach, a local motion estimation method is proposed to generate the foreground objects in the SI exploiting the backward and forward foreground objects. Contrary to MCTI technique, in this approach, an extended window is used in estimating backward motion vectors, bi-directional motion refinement and spatial smoothing of the motion vectors are not applied.

Then, a mask is generated using the estimated foreground objects in the SI. Based on the mask, two approaches are proposed to combine global and local motion estimations. The first one aims at directly using the estimated foreground objects and GMC SI. The second one consists in using MCTI SI for the pixels in the object mask and GMC SI for the remaining pixels.

This chapter is structured as follows. The proposed methods for the fusion of global and local motion estimations are described in Section 6.1. More specifically, the removal of artifacts affecting the GMC SI is described in Section 6.1.1, fusion using elastic curves is described in Section 6.1.2, fusion using local motion compensation is described in Section 6.1.3, and the oracle fusion is described in Section 6.1.4. Experimental results are shown in Section 6.2 in order to evaluate and compare the RD performance of the proposed approaches. Finally, conclusions are presented in Section 6.3.

6.1 Proposed methods

For the segmentation of the foreground objects, the authors in [78, 79] propose a coarse-to-fine segmentation method for extracting moving regions from compressed video. In the proposed methods, we consider that the foreground objects in the Backward Reference Frame (BRF) and Forward Reference Frame (FRF) are already segmented. Here, we are interested in the combination of global and local motion estimations.

Let R_B and R_F be the backward and forward reference frames respectively. The foreground objects F_B^i and F_F^i ($i = 1, 2, ..., N_o$, N_o is the number of foreground objects) are already segmented from the backward and forward reference frames, respectively. Furthermore, the foreground objects masks M_B^i and M_F^i are generated from the foreground objects

Figure 6.1: Original frame number 1 of Stefan sequence.

Figure 6.2: Foreground object (F) of frame number 1 of Stefan sequence.

Figure 6.3: Foreground object mask (M) of frame number 1 of Stefan sequence.

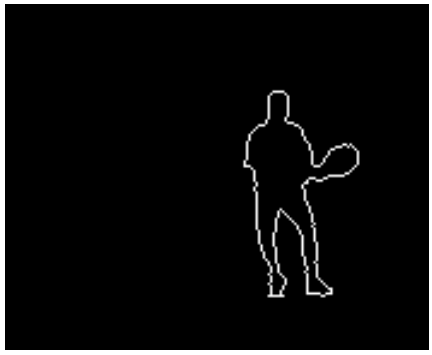

Figure 6.4: Foreground object contour (β) of frame number 1 of Stefan sequence.

according to:

$$\begin{cases} M_B^i(x,y) = \begin{cases} 0 & \text{if } F_B^i(x,y) = 0 \\ 1 & \text{otherwise} \end{cases} \\ M_F^i(x,y) = \begin{cases} 0 & \text{if } F_F^i(x,y) = 0 \\ 1 & \text{otherwise} \end{cases} \end{cases} \tag{6.1}$$

Then, the foreground objects contours are extracted from the foreground objects masks. The contours can be considered as closed curves. Here, β_B^i and β_F^i are the representations of the backward and forward foreground objects contours. Figs. 6.1, 6.2, 6.3 and 6.4 show, respectively, the original frame, the foreground object, the foreground object mask generated from the foreground object, and the generated foreground object contour, for frame number 1 of Stefan sequence.

6.1.1 Artifact removal in GMC SI using foreground objects masks

The two transforms T_B and T_F are defined as the affine transforms between the WZF and the backward and forward reference frames respectively. Let $\widehat{R}_B$ and $\widehat{R}_F$ be the results of the GMC transforms T_B and T_F applied to R_B and R_F respectively. The GMC SI is simply defined as the average of the frames $\widehat{R}_B$ and $\widehat{R}_F$. Fig. 6.5 shows an example of a

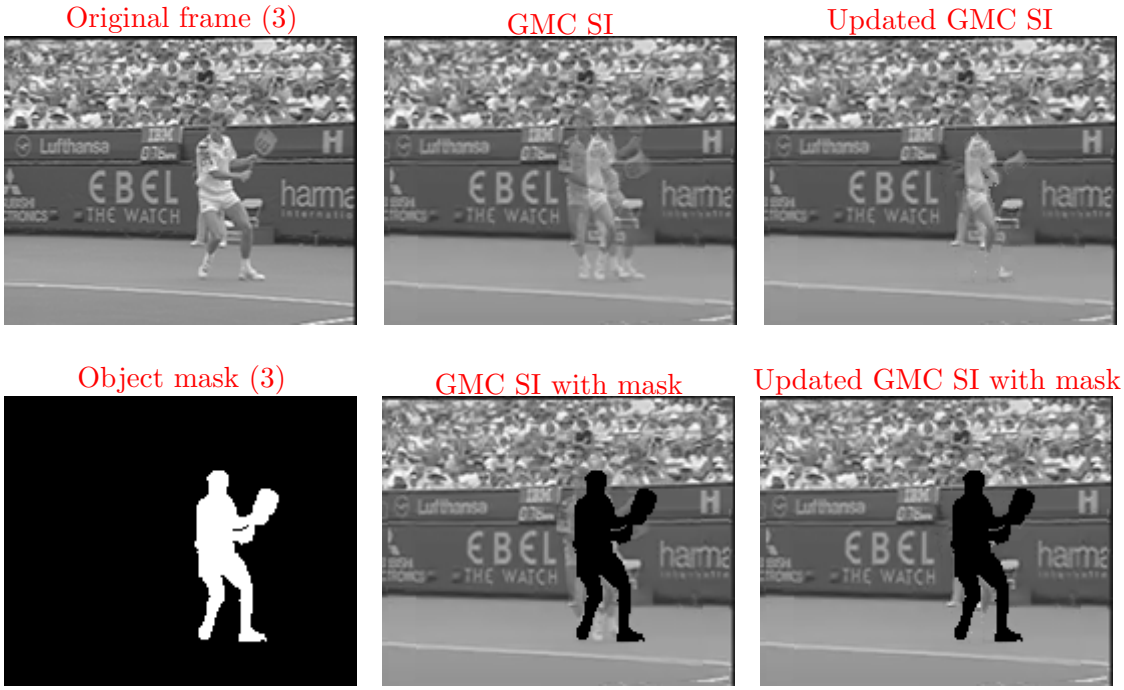

Figure 6.5: Original frame, GMC SI, updated GMC SI, Object mask, GMC SI with mask and updated GMC SI with mask for frame number 3 of Stefan sequence.

GMC SI (top center) and the GMC SI with the object mask (bottom center), for frame number 3 of Stefan sequence. As we can see, the background around the foreground object in GMC SI is affected by the shifted foreground objects due to global motion. In this case, the background in one of the reference frames is averaged with the foreground objects of the other reference frame. We propose to remove this artifact effect around the foreground objects using the obtained segmented foreground objects of the reference frames.

The masks M_B and M_F are defined as the union of all foreground objects masks M_B^i and M_F^i respectively:

$$\begin{cases} M_B = \bigcup_{i=1}^{N_o} M_B^i \\ M_F = \bigcup_{i=1}^{N_o} M_F^i \end{cases} \tag{6.2}$$

Let $\widehat{M}_B$ and $\widehat{M}_F$ be the results of the GMC transforms T_B and T_F applied to the masks M_B and M_F respectively. $\widehat{M}_B$ and $\widehat{M}_F$ are used in order to remove the artifact of the pixels in the background around the foreground objects. First, each pixel in the transformed frames $\widehat{R}_B$ and $\widehat{R}_F$ is assigned to either the background or the foreground objects, using $\widehat{M}_B$ and $\widehat{M}_F$. Then, in order to avoid the averaging between the background

and the foreground objects, the GMC SI can be updated as follows:

$$\begin{cases} \text{if } \widehat{M}_B(x,y) = 1 \text{ and } \widehat{M}_F(x,y) = 0 \\ \quad \text{GMC SI}(x,y) = \widehat{R}_F(x,y) \\ \text{otherwise} \\ \quad \text{if } \widehat{M}_B(x,y) = 0 \text{ and } \widehat{M}_F(x,y) = 1 \\ \quad\quad \text{GMC SI}(x,y) = \widehat{R}_B(x,y) \end{cases}$$

In such situations, only the background is taken for GMC SI. Fig. 6.5 shows the updated GMC SI (top right) and the updated GMC SI with the object mask, for frame number 3 of Stefan sequence. It is clear that the artifact effect is removed around the foreground object compared to the GMC SI.

6.1.2 Fusion using elastic curves

The function β is defined as follows:

$$\begin{aligned} \beta : \mathbb{D} &\longmapsto \mathbb{R}^2 \\ t &\longmapsto (x,y) \end{aligned} \tag{6.3}$$

where $t \in \mathbb{D} = [0,1]$ and (x,y) represent the coordinates of each point in the contour. For the purpose of studying the shape of β, it is represented using the Square Root Velocity (SRV) function defined as $q : \mathbb{D} \longmapsto \mathbb{R}^2$ [14]:

$$q(t) = \frac{\dot{\beta}}{\sqrt{||\dot{\beta}||}} \tag{6.4}$$

where $||.||$ is the Euclidean norm in $\mathbb{R}^2$ and $\dot{\beta} = \frac{d\beta}{dt}$. The curve β can be obtained using q as follows:

$$\beta(t) = \int_0^t q(s)||q(s)||ds \tag{6.5}$$

The backward and forward curves β_b^i and β_f^i are considered as closed curves in our application, but they are considered as open curves when finding the best matching between them. Here, we aim at finding the estimated curve β_e^i between the backward and forward curves. The algorithm used to estimate β_e^i is described as follows (we refer the reader to [14] for the theory behind this estimation): First, the SRV representation of the curve β_b^i is computed as follows:

$$q_b^i(t) = \frac{\dot{\beta}_b^i(t)}{\sqrt{||\dot{\beta}_b^i(t)||}} \tag{6.6}$$

At the beginning of this algorithm, the parameters θ_{min}, δt, and k are respectively set to 2π, $\frac{1}{n}$, and zero.

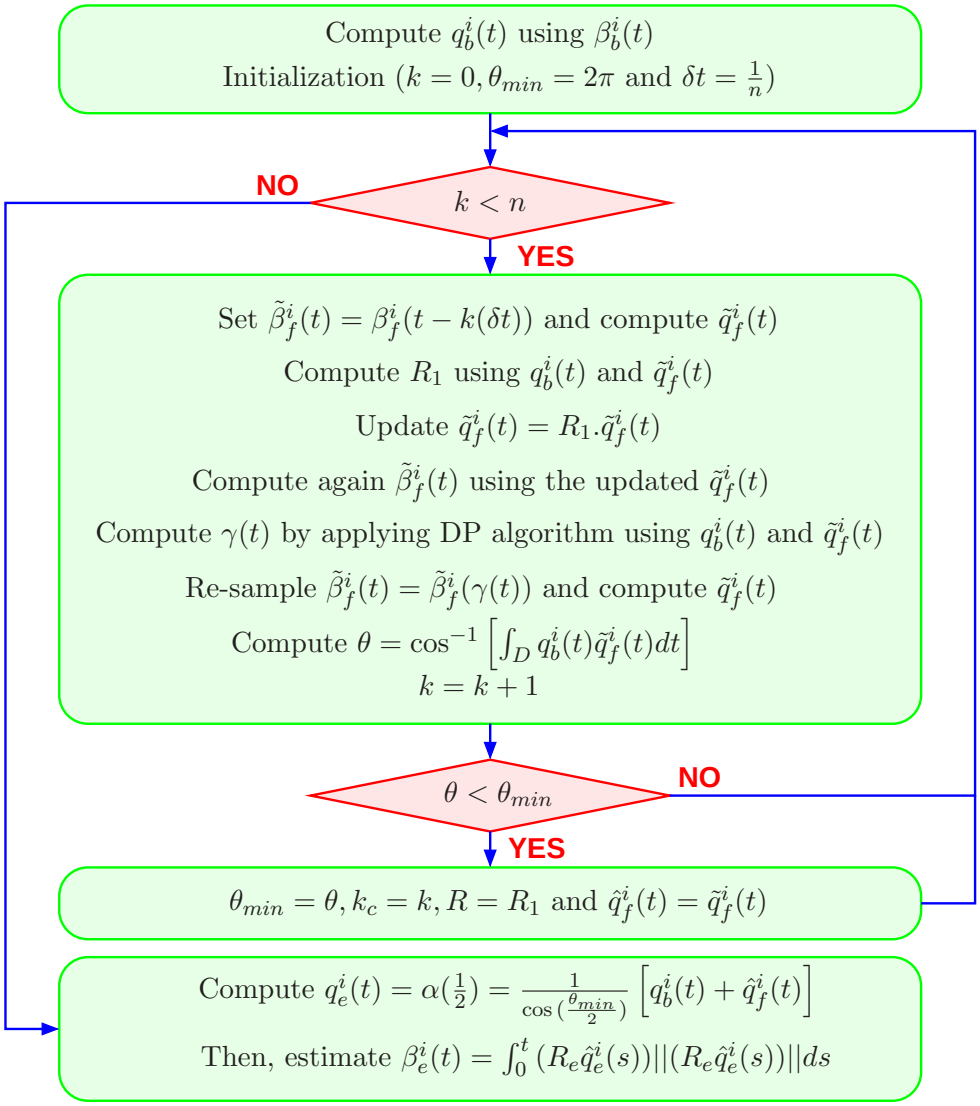

Figure 6.6: Algorithm proposed in [14] for estimating $\beta_e^i(t)$.

Step 1 - A circular shift of $k(\delta t)$ is applied on the forward curve $\beta_f^i(t)$ as follows:

$$\tilde{\beta}_f^i(t) = \beta_f^i(t - k(\delta t)) \tag{6.7}$$

Then, the SRV representation of $\tilde{\beta}_f^i(t)$, denoted by $\tilde{q}_f^i(t)$, is computed using Eq. 6.4.

Step 2 - Rotation: The optimal rotation between q_b^i and $\tilde{q}_f^i$ is given by R_1 as follows:

$$R_1 = UIV^T \tag{6.8}$$

where

$$[U, S, V] = \text{SVD}(B), \;\; B = \int_D q_b^i(t)\tilde{q}_f^i(t)^T dt \;\text{ and }\; I = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right) \tag{6.9}$$

SVD stands for the Singular Value Decomposition of matrix B. If $\det(B) < 0$, the last column of I changes sign before multiplication in Eq. 6.8. Then, $\tilde{q}_f^i$ is multiplied by R_1 as follows:

$$\tilde{q}_f^i(t) = R_1.\tilde{q}_f^i(t) \tag{6.10}$$

Following that, $\tilde{q}_f^i(t)$ is used to reconstruct $\tilde{\beta}_f^i(t)$ as follows:

$$\tilde{\beta}_f^i(t) = \int_0^t \tilde{q}_f^i(s)||\tilde{q}_f^i(s)||ds \tag{6.11}$$

Step 3 - Reparameterization: This step consists in using q_b^i and $\tilde{q}_f^i$ to find a function $\gamma(t)$, by applying the Dynamic Programming (DP) algorithm. The obtained function $\gamma(t)$ is used to re-sample $\tilde{\beta}_f^i(t)$ as follows:

$$\tilde{\beta}_f^i(t) = \tilde{\beta}_f^i(\gamma(t)) \tag{6.12}$$

Furthermore, $\tilde{q}_f^i(t)$ is again computed using the updated $\tilde{\beta}_f^i(t)$ (Eq. 6.4).

Step 4 - Compute the length of the geodesic θ as follows:

$$\theta = \cos^{-1}\left[\int_D q_b^i(t)\tilde{q}_f^i(t)dt\right] \tag{6.13}$$

If $\theta < \theta_{min}$, the parameters θ_{min}, k_c, R and $\hat{q}_f^i(t)$ are updated as follows:

$$\begin{cases} \theta_{min} = \theta \\ k_c = k \\ R = R_1 \\ \hat{q}_f^i(t) = \tilde{q}_f^i(t) \end{cases} \tag{6.14}$$

Then, k is set to $k+1$. If k is smaller than n, go to **Step 1**. Otherwise, go to **Step 5**

Step 5 - The geodesic $\alpha(\tau), \tau \in [0,1]$ that connects $q_b^i(t)$ and $\hat{q}_f^i(t)$, is defined as follows:

$$\alpha(\tau) = \frac{1}{\sin(\theta_{min})}\left[\sin(\theta_{min}(1-\tau))q_b^i(t) + \sin(\theta_{min}\tau)\hat{q}_f^i(t)\right] \tag{6.15}$$

It is clear that $\alpha(0) = q_b^i(t)$ and $\alpha(1) = \hat{q}_f^i(t)$. This equation allows predicting the curves between the backward curve β_b^i and the forward curve β_f^i at any time $\tau \in [0,1]$. Here, we aim at estimating the curve at the middle between the backward

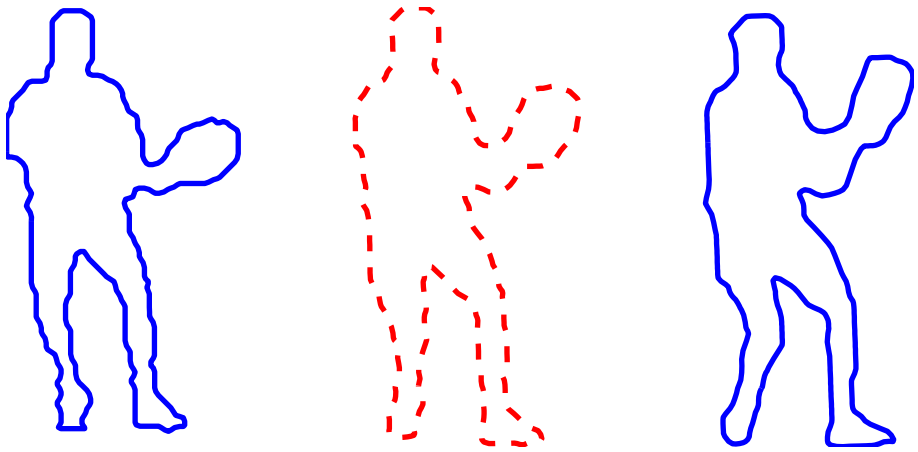

Figure 6.7: The backward curve $\beta_b^i(t)$ (left, frame number 1), the forward curve $\beta_f^i(t)$ (right, frame number 3) and the estimated curve $\beta_e^i(t)$ (center, $\tau = \frac{1}{2}$) between the backward and forward curves.

and forward curves. For this reason, we compute $\alpha(\frac{1}{2})$ to obtain $q_e^i(t)$ as follows:

$$\begin{aligned} q_e^i(t) = \alpha(\frac{1}{2}) &= \frac{1}{\sin(\theta_{min})}\left[\sin(\frac{\theta_{min}}{2})q_b^i(t) + \sin(\frac{\theta_{min}}{2})\hat{q}_f^i(t)\right] \\ &= \frac{1}{\cos(\frac{\theta_{min}}{2})}\left[q_b^i(t) + \hat{q}_f^i(t)\right] \end{aligned} \tag{6.16}$$

Then, $q_e^i(t)$ is projected [14] in $\mathbf{C}^c$ to obtain $\hat{q}_e^i(t)$ ($\mathbf{C}^c$ represents the closed curves).

Step 6 - The objective of this step is to obtain the curve $\beta_e^i(t)$ using $\hat{q}_e^i(t)$ with the rotation matrix R. The rotation matrix can be written as follow:

$$R = \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix}$$

where φ is the angle of rotation. The rotation matrix R_e for the estimated curve can be written as follows:

$$R_e = \begin{pmatrix} \cos(\phi_e) & -\sin(\phi_e) \\ \sin(\phi_e) & \cos(\phi_e) \end{pmatrix}$$

where $\phi_e = \frac{\varphi}{2}$. The curve $\beta_e^i(t)$ can be estimated as follows:

$$\beta_e^i(t) = \int_0^t (R_e\hat{q}_e^i(s))||(R_e\hat{q}_e^i(s))||ds \tag{6.17}$$

Figure 6.8: The backward curve $\beta_b^i(t)$ (left, frame number 1 of Stefan sequence), the forward curve $\beta_f^i(t)$ (right, frame number 5) and the three estimated curves $\beta_e^i(t)$ for $\tau = \frac{1}{4}, \frac{2}{4}$ and $\frac{3}{4}$ (center curves).

Fig. 6.7 shows an application example of this algorithm, where we show the backward curve $\beta_b^i(t)$ (left curve) of frame number 1 of Stefan sequence, the forward curve $\beta_f^i(t)$ (right curve) of frame number 3 of this sequence, and the estimated curve $\beta_e^i(t)$ (center curve) between the backward and forward curves using this algorithm. Moreover, Fig. 6.8 shows the backward curve $\beta_b^i(t)$ (left) of frame number 1 of Stefan sequence, the forward curve $\beta_f^i(t)$ (right) of frame number 5 of Stefan sequence and the estimated curves $\beta_e^i(t)$ for $\tau = \frac{1}{4}, \frac{2}{4}$ and $\frac{3}{4}$(center curves).

The obtained curves $\beta_e^i(t)$ are then used to obtain the foreground objects masks M_e^i by covering all the pixels lying inside the curves. The mask M_e is defined as the union of all masks M_e^i:

$$M_e = \bigcup_{i=1}^{N_o} M_e^i \tag{6.18}$$

Then, to generate the SI, the pixels inside the mask M_e are selected from MCTI SI and the background pixels from GMC SI:

$$\text{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_e(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{6.19}$$

This fusion method is referred to as 'FusElastic'.

6.1.3 Fusion using local motion compensation

In this section, we propose to apply the MCTI technique [7] to the foreground objects in order to estimate the local motion. Then, a new scheme for local motion estimation is

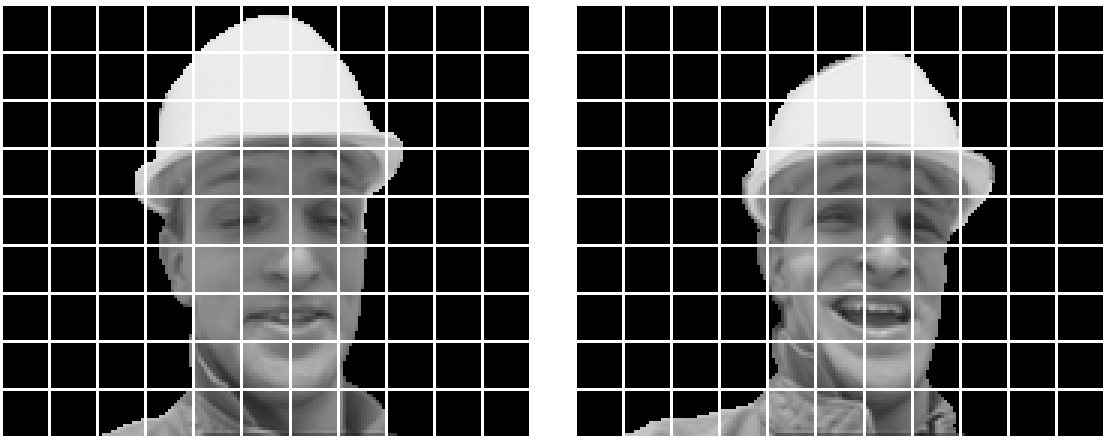

Figure 6.9: Foreground objects number 1 and 9 of Foreman sequence, split into 16×16 blocks.

proposed.

Applying MCTI on the foreground objects

In this approach, the MCTI technique is applied to the backward foreground object F_B^i and the forward foreground object F_F^i, in order to estimate the foreground object F_{MCTI}^i in the SI. In this case, there are blocks entirely black, partly black or entirely white. Fig. 6.9 shows foreground objects for frame number 1 and 9 of Foreman sequence, split into 16×16 blocks. In contrast, MCTI SI is estimated by applying the MCTI technique to the whole (Background and Foreground) reference frames. Let F_{MCTI} be the union of all foreground objects in the SI, which are estimated using the MCTI technique:

$$F_{\text{MCTI}} = \bigcup_{i=1}^{N_o} F_{\text{MCTI}}^i \tag{6.20}$$

The mask M_{MCTI} is generated from the estimated foreground objects F_{MCTI} as follows:

$$M_{\text{MCTI}}(x,y) = \begin{cases} 0 & \text{if } F_{\text{MCTI}}(x,y) = 0 \\ 1 & \text{otherwise} \end{cases} \tag{6.21}$$

Here, we propose two approaches for the combination of global and local motion estimations, based on the generated mask M_{MCTI}. The first approach consists in fusing GMC SI with the estimated foreground objects F_{MCTI} using:

$$\text{SI}(x,y) = \begin{cases} F_{\text{MCTI}}(x,y) & \text{if } M_{\text{MCTI}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{6.22}$$

This method is referred to as 'FoMCTI'.

The second approach fuses GMC SI and MCTI SI (taken within the masks) and is

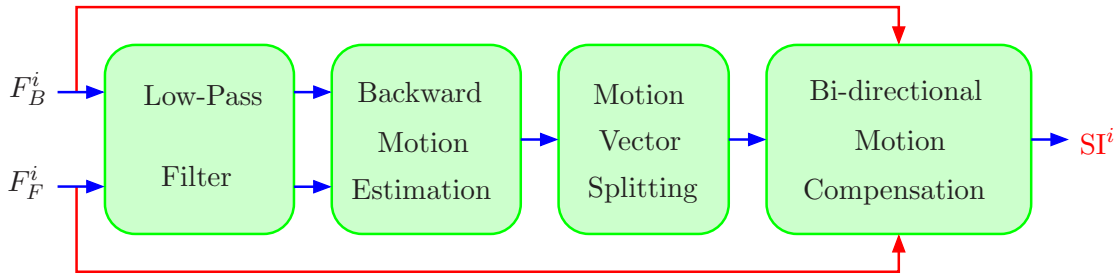

Figure 6.10: Proposed method for foreground objects estimation.

defined as follows:

$$\mathrm{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_{\text{MCTI}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{6.23}$$

This method is referred to as 'FoMCTI2'.

Proposed local motion estimation

In this section, we propose a new method for estimating the foreground objects in the SI, using the backward and forward foreground objects. The proposed scheme is illustrated in Fig. 6.10. This technique is referred to as Foreground Object Motion Compensation (FOMC).

- **Low-Pass Filtering:** The backward F_B^i and foreground F_F^i foreground objects are low-pass filtered in order to improve the motion vectors reliability.
- **Backward Motion Estimation:** A Block Matching Algorithm (BMA) is applied to estimate the backward motion vector field. This estimation is done using a block size 16×16, a search area ($\mathbf{S}$) of ± 32 pixels, and a step size of 2 pixels. First, if all the pixels in the current block b in F_F^i and the co-located block in F_B^i are black (corresponding to non-object pixels), the motion vector is set to $\mathbf{0}$ for this block (see Fig. 6.9). In the case when the block b is partly black, the BMA is used to find the corresponding block (*i.e.*, BMA can find the most similar shape).

  In the BMA, the Weighted Mean Absolute Difference (WMAD) criterion is used to compute the similarity between the target block b in the forward foreground object frame F_F^i and the shifted block in the backward foreground object frame F_B^i by the motion vector $\mathbf{v} \equiv (v_x, v_y) \in \mathbf{S}$, as follows:

$$\mathrm{WMAD}(b,\mathbf{v}) = \frac{1}{16^2} \sum_{x=-e}^{16+e} \sum_{y=-e}^{16+e} |F_F^i(x,y) - F_B^i(x+v_x, y+v_y)| \left(1 + \lambda \sqrt{v_x^2 + v_y^2}\right) \tag{6.24}$$

  with λ a penalty factor used to penalize the MAD by the length of the motion vector $\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2}$ (it is empirically set to 0.05). An extended block of $(16+2e, 16+2e)$

(e being empirically set to 8) is used in the WMAD. The best backward motion vector $\mathbf{V_b}$ for the block b is obtained by minimizing the WMAD as follows:

$$\mathbf{V_b} = \arg\min_{\mathbf{v_i}\in\mathbf{S}} \mathrm{WMAD}(b, \mathbf{v}_i). \tag{6.25}$$

- **Motion Vector Splitting:** Here, the obtained motion vectors are divided in such a way to obtain bi-directional motion vectors for the blocks in the estimated foreground object F^i_{FOMC}. For each block b in F^i_{FOMC}, the distances between the center of the block b and the center of each obtained motion vector are computed. The closest motion vector to the block b is selected. Then, the selected motion vector is associated to the center of the block b, and divided by symmetry to obtain the bidirectional motion field.

- **Bi-directional Motion Compensation:** Once the final bidirectional motion vectors are estimated, the F^i_{FOMC} can be interpolated using bidirectional motion compensation as follows:

$$F^i_{\mathrm{FOMC}}(\mathbf{p}) = \frac{1}{2}(F^i_B(\mathbf{p}+\mathbf{s}_b) + F^i_F(\mathbf{p}-\mathbf{s}_b)), \tag{6.26}$$

  where $\mathbf{s}_b$ and $-\mathbf{s}_b$ are the bidirectional motion vectors, associated to the position $\mathbf{p} = (x, y)$, toward the F^i_B and F^i_F respectively.

The F^i_{FOMC} is estimated for each foreground object i ($i = 1, 2, ..., N_o$). Then, all F^i_{FOMC} are combined to form F_{FOMC} as follows:

$$\mathrm{F_{FOMC}} = \bigcup_{i=1}^{N_o} F^i_{\mathrm{FOMC}} \tag{6.27}$$

Furthermore, the mask M_{FOMC} is generated using F_{FOMC} as follows:

$$M_{\mathrm{FOMC}}(x, y) = \begin{cases} 0 & \text{if } F_{\mathrm{FOMC}}(x, y) = 0 \\ 1 & \text{otherwise} \end{cases} \tag{6.28}$$

Here, two approaches are proposed to combine the global and local motion estimations using M_{FOMC}. The first one aims at combining GMC SI and F_{FOMC} as follows:

$$\mathrm{SI}(x, y) = \begin{cases} F_{\mathrm{FOMC}}(x, y) & \text{if } M_{\mathrm{FOMC}}(x, y) = 1 \\ \text{GMC SI}(x, y) & \text{otherwise} \end{cases} \tag{6.29}$$

This method is referred to as 'BmEst'.

The second approach consists in combining GMC SI and MCTI SI as follows:

$$\mathrm{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_{\mathrm{FOMC}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{6.30}$$

This method is referred to as 'BmMCTI'.

6.1.4 Oracle method

In this section, we describe the oracle method which consists in fusing GMC SI and MCTI SI using the foreground objects masks of the original WZFs. Let M_{WZF} be the union of all foreground objects masks in the original WZF :

$$M_{\mathrm{WZF}} = \bigcup_{i=1}^{N_o} M^i_{\mathrm{WZF}} \tag{6.31}$$

M^i_{WZF} is the i^{th} foreground object mask in the WZF. The oracle method combines GMC SI and MCTI SI as follows:

$$\mathrm{SI}(x,y) = \begin{cases} \text{MCTI SI}(x,y) & \text{if } M_{\mathrm{WZF}}(x,y) = 1 \\ \text{GMC SI}(x,y) & \text{otherwise} \end{cases} \tag{6.32}$$

This method is of course impractical, but it allows us to estimate the ideal upper bound limit that can be achieved by combining GMC SI and MCTI SI, using the foreground objects masks of the original WZF.

6.2 Experimental results

The performance of the proposed methods are assessed using extensive simulations under the same test conditions as in DISCOVER [4, 5]. An example is illustrated in Fig 6.11 for several test sequences with the corresponding foreground objects: Stefan (one object, 45 frames), Foreman (one object, 150 frames), Bus (three objects, 75 frames), and Coast-guard (two objects, 150 frames). Here, the segmentation masks for the reference frames are assumed to be known. The obtained results of the proposed methods are compared to the DISCOVER codec and to our previous fusion technique SADbin (presented in Chapter 5).

SI performance assessment

Fig. 6.12 shows the original curve and the estimated curve using the elastic curve algorithm [14], for frame number 2 of Stefan sequence, for a GOP size of 2. It is clear that the difference between the two curves is small.

Figure 6.11: The foreground objects in the test sequences: Stefan (one object), Foreman (one object), Bus (three objects), and Coastguard (two objects).

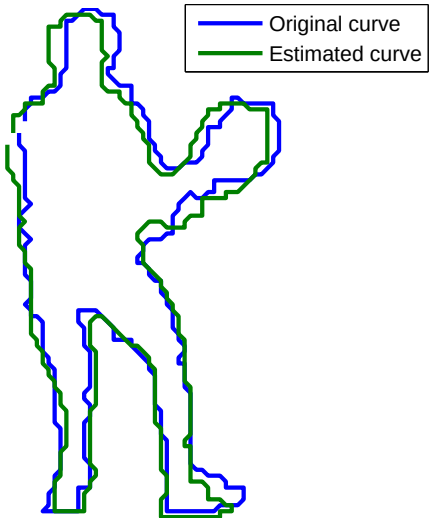

Figure 6.12: Comparison between the original curve and the estimated curve using the elastic curve [14] for frame number 2 of Stefan sequence.

Table 6.1 shows the average PSNR of the SI obtained with MCTI, SADbin, FusElastic, BmEst, BmMCTI, FoMCTI, FoMCTI2, and Oracle for Stefan, Foreman, Bus, and Coastguard sequences, for GOP sizes of 2, 4, and 8. It is clear that the proposed fusion methods

Table 6.1: SI average PSNR for a GOP size equal to 2, 4, and 8 (QI = 8).

SI Average PSNR [dB]								
Method	**MCTI**	**SADbin**	**FusElastic**	**BmEst**	**BmMCTI**	**FoMCTI**	**FoMCTI2**	**Oracle**
GOP = 2								
Stefan	25.17	28.16	28.43	**28.72**	28.53	28.69	28.49	28.71
Foreman	29.38	30.82	31.09	30.97	31.11	30.99	**31.13**	31.15
Bus	25.37	27.30	27.30	26.92	**27.56**	27.30	27.48	27.90
Coastguard	31.47	32.00	31.80	31.91	31.91	**32.03**	31.89	32.07
GOP = 4								
Stefan	23.49	27.18	27.72	**27.95**	27.86	27.87	27.79	28.14
Foreman	27.64	29.27	29.79	29.71	29.82	29.71	**29.83**	29.88
Bus	24.00	26.27	26.29	26.02	**26.54**	26.28	26.39	26.91
Coastguard	29.91	30.76	30.68	30.77	30.73	**30.88**	30.72	30.88
GOP = 8								
Stefan	22.84	26.91	27.35	**27.67**	27.55	27.55	27.46	27.80
Foreman	26.29	28.09	28.74	28.64	28.75	28.65	**28.77**	28.83
Bus	22.95	25.26	25.33	25.13	**25.55**	25.36	25.45	25.94
Coastguard	28.82	29.85	29.77	29.88	29.83	**29.96**	29.82	30.00

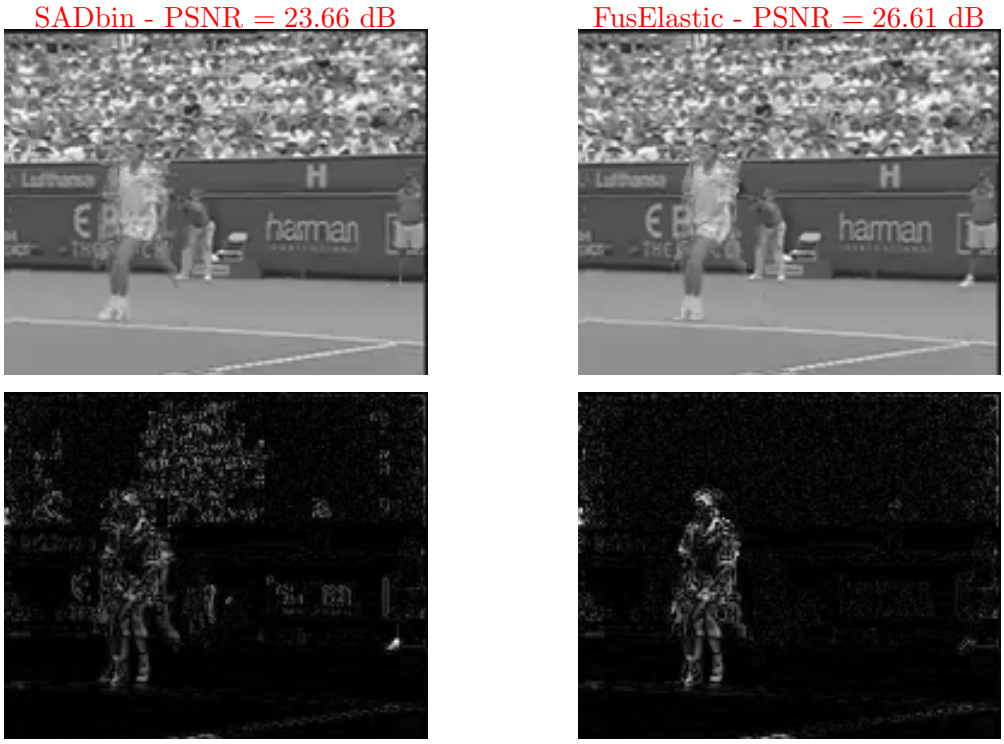

Figure 6.13: Visual result of the SI estimated by SADbin (PSNR = 23.66 dB) and FusElastic (PSNR = 26.61 dB), for frame number 27 of Stefan sequence, for a GOP size of 4 (QI = 8). The bottom images represents the visual differences of these SI frames.

can improve the quality of the SI compared to MCTI for all test sequences and all GOP sizes. The proposed method FusElastic can achieve a gain compared to the previous fusion SADbin for Stefan and Foreman sequences. For Bus sequence, the PSNR average of the two approaches SADbin and FusElastic is almost the same. For Coastguard sequence, the SADbin can achieve a slight gain compared to FusElastic.

Concerning BmEst and BmMCTI fusion methods, BmEst can achieve a gain compared

Table 6.2: Rate-distortion performance gain for *Stefan*, *Foreman*, *Bus*, and *Coastguard* sequences towards DISCOVER codec, using Bjontegaard metric, for a GOP size of 2, 4, and 8.

Method	**SADbin**	**FusElastic**	**BmEst**	**BmMCTI**	**FoMCTI**	**FoMCTI2**	**Oracle**
GOP = 2							
Stefan							
Δ_R (%)	-17.97	-19.72	**-20.06**	-19.98	**-20.05**	-19.79	-20.38
Δ_{PSNR} [dB]	1.23	1.36	**1.39**	1.38	**1.39**	1.37	1.41
Foreman							
Δ_R (%)	-7.58	**-9.65**	-8.51	**-9.67**	-8.37	**-9.70**	-10.07
Δ_{PSNR} [dB]	0.45	**0.59**	0.52	**0.59**	0.49	**0.59**	0.61
Bus							
Δ_R (%)	-12.94	-12.51	-10.25	**-13.34**	-10.75	-11.25	-14.51
Δ_{PSNR} [dB]	0.79	0.75	0.61	**0.80**	0.64	0.68	0.87
Coastguard							
Δ_R (%)	-4.60	-4.32	-4.34	**-4.74**	-4.40	-4.33	-5.36
Δ_{PSNR} [dB]	0.23	0.22	0.22	**0.24**	0.22	0.21	0.27
GOP = 4							
Stefan							
Δ_R (%)	-40.66	-45.18	-45.73	-45.74	-45.80	**-45.71**	-46.42
Δ_{PSNR} [dB]	2.93	3.38	3.42	3.44	3.44	**3.45**	3.51
Foreman							
Δ_R (%)	-15.54	**-21.72**	-20.91	-21.81	-20.34	**-21.93**	-22.41
Δ_{PSNR} [dB]	0.90	**1.33**	1.25	1.32	1.19	**1.33**	1.36
Bus							
Δ_R (%)	-25.95	-25.97	-24.10	**-27.45**	-22.19	-23.67	-28.60
Δ_{PSNR} [dB]	1.60	1.57	1.41	**1.67**	1.34	1.40	1.78
Coastguard							
Δ_R (%)	-14.91	-16.48	-16.37	**-16.59**	-16.24	-15.70	-17.94
Δ_{PSNR} [dB]	0.61	0.68	0.68	**0.69**	0.67	0.65	0.75
GOP = 8							
Stefan							
Δ_R (%)	-51.56	-55.95	**-57.12**	-57.04	**-57.10**	-56.94	-57.84
Δ_{PSNR} [dB]	4.05	4.60	**4.72**	4.72	**4.73**	4.72	4.83
Foreman							
Δ_R (%)	-22.29	**-31.24**	-30.09	-31.01	-29.12	-30.78	-31.80
Δ_{PSNR} [dB]	1.29	**1.93**	1.84	1.92	1.76	1.91	1.97
Bus							
Δ_R (%)	-32.07	-32.82	-31.58	**-34.16**	-27.87	-28.53	-35.50
Δ_{PSNR} [dB]	2.04	2.07	1.97	**2.19**	1.72	1.74	2.31
Coastguard							
Δ_R (%)	-26.32	-29.50	**-30.37**	-29.73	-29.48	-28.19	-31.32
Δ_{PSNR} [dB]	1.10	1.24	**1.27**	1.26	1.23	1.18	1.35

to BmMCTI for Stefan and Coastguard sequences, while BmMCTI allows a gain compared to BmEst for Foreman and Bus sequences. According to this comparison, we can say that the estimation of the foreground objects in MCTI SI is better than the estimation of the foreground objects using our FOMC method for Foreman and Bus sequences. However, FOMC is better than MCTI in the estimation of the foreground objects for Stefan and Coastguard sequences.

For FoMCTI and FoMCTI2, we can see the same comparison as between BmEst and BmMCTI. Therefore, when the MCTI technique is only applied on the foreground objects, the quality of the estimated foreground objects is better than the quality of MCTI SI, for Stefan and Coastguard sequences. For Foreman and Bus sequences, the estimation of the foreground objects in MCTI SI is better than the quality of the generated foreground objects by applying MCTI only on the foreground objects.

It is important to note that the oracle method represents the fusion of GMC SI and MCTI SI using the foreground objects of the original WZF. However, BmEst and FoMCTI methods represent the fusion of GMC SI and the estimated foreground objects. Thus, the oracle fusion represents the upper bound limit that can be achieved by the proposed fusion methods excluding BmEst and FoMCTI. For this reason, the average PSNR obtained by BmEst (28.72 dB) is better than that the average PSNR of the oracle fusion (28.71 dB), for Stefan sequence, for a GOP size of 2.

Fig. 6.13 shows the visual results and the visual differences of the SI for frame number of 27 of Stefan sequence, for a GOP size of 4. The SI obtained by SADbin fusion may contain a block artifact (top-left - 23.66 dB). The proposed fusion FusElastic can improve the quality of the SI for this frame (top-right - 26.61 dB), with a gain of 2.95 dB compared to SADbin.

Rate-Distortion performance

The RD performance of the proposed methods SADbin, FusElastic, BmEst, BmMCTI, FoMCTI, and FoMCTI2 is shown along with the Oracle fusion, for Stefan, Bus, Foreman, and Coastguard sequences in Table 6.2, in comparison to the DISCOVER codec, using the Bjontegaard metric [11], for GOP sizes of 2, 4, and 8.

All the fusion methods can achieve a gain compared to DISCOVER codec. The proposed method FusElastic allows a gain compared to SADbin for Stefan and Foreman sequences for a GOP size of 2, and for all test sequences for a GOP size of 8. The gain is up to 4.6 dB compared to DISCOVER codec and 0.55 dB compared to SADbin, for a GOP size of 8. The loss is up to 0.04 dB compared to SADbin for Bus sequence with a GOP size of 2.

The remaining fusion methods almost achieve the same gains compared to DISCOVER. The gain is up to 4.73 dB compared to DISCOVER codec for Stefan sequence, for a GOP size of 8.

Figs. 6.14, 6.15, and 6.16 show the RD performance of the DISCOVER codec, SADbin, FusElastic, and the Oracle, for Stefan, Foreman, Bus, and Coastguard sequences, for GOP sizes of 2, 4, and 8 respectively. The proposed fusion methods SADbin and FusElastic always achieve a gain compared to DISCOVER codec for all test sequences. The proposed fusion FusElastic can achieve a gain up to 0.13 dB, 0.45 dB, and 0.55 dB compared to SADbin fusion for a GOP size of 2, 4, and 8 respectively, for Stefan sequence. For Foreman sequence, FusElastic fusion allows a gain up to 0.14 dB, 0.43 dB, and 0.64 dB respectively for a GOP size of 2, 4, and 8. For Bus and Coastguard sequences, the two methods SADbin and FusElastic almost achieve the same RD performance.

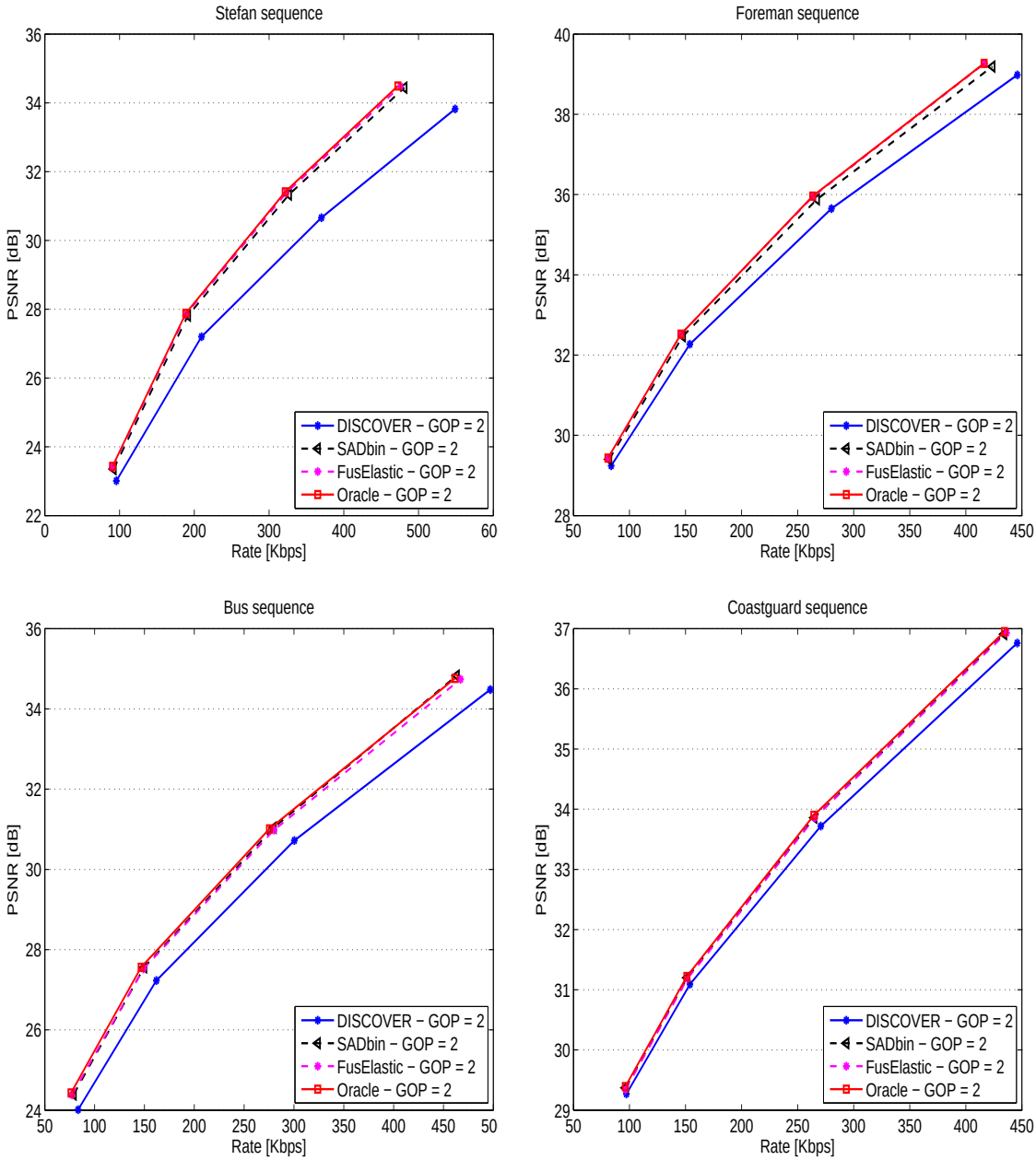

Figure 6.14: RD performance comparison among DISCOVER, SADlin, FusElastic, and Oracle for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 2.

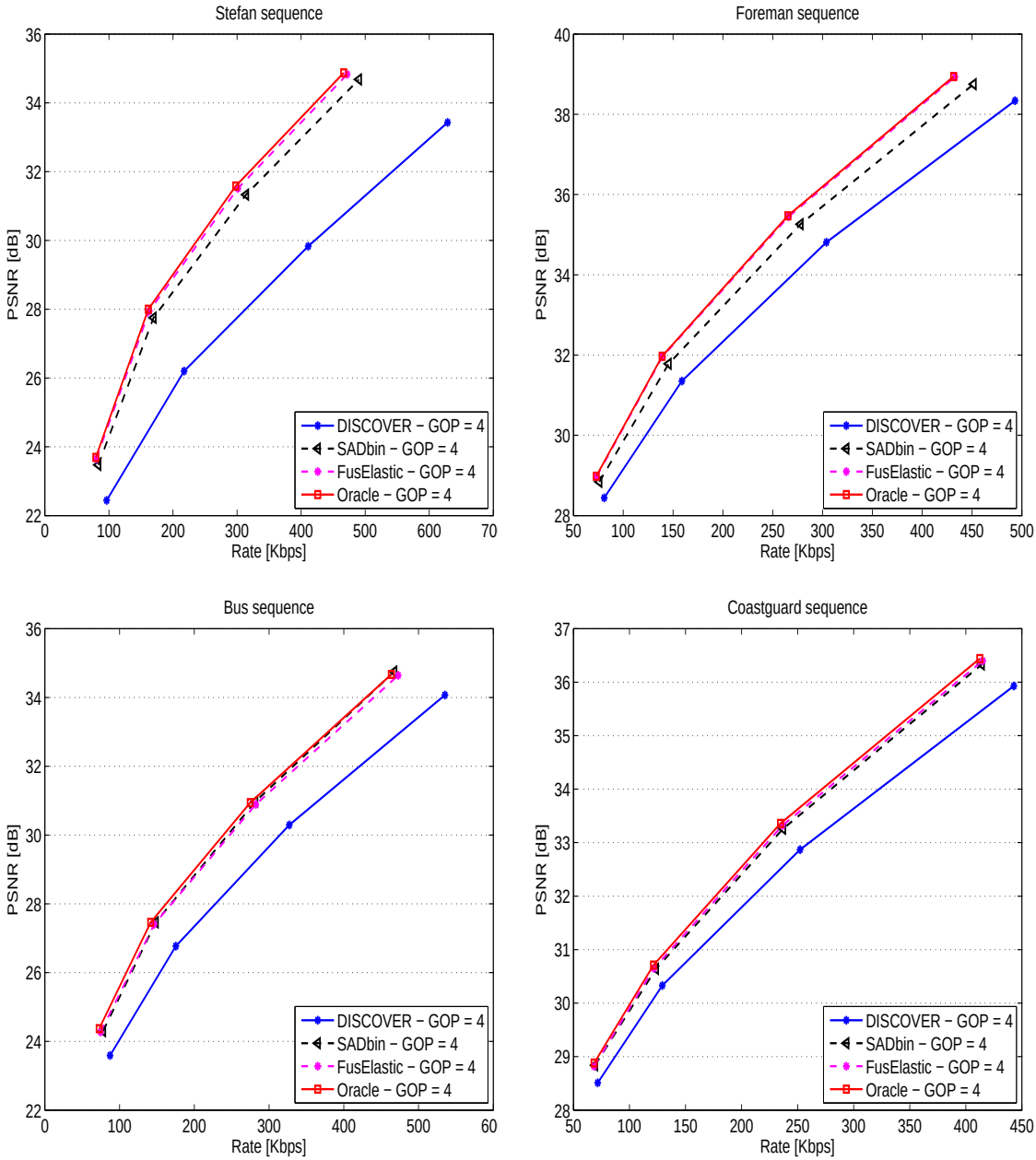

Figure 6.15: RD performance comparison among DISCOVER, SADlin, FusElastic, and Oracle for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 4.

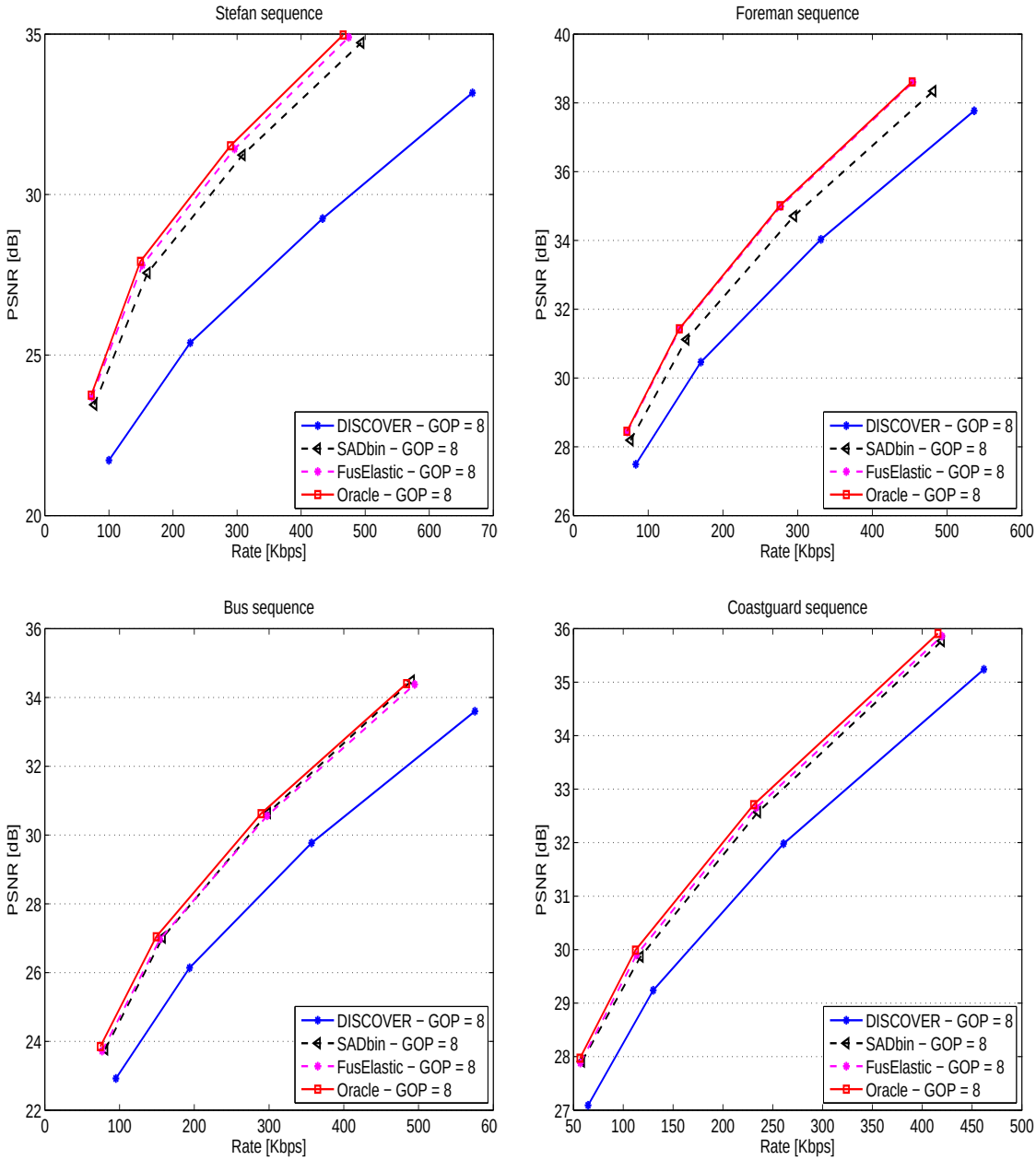

Figure 6.16: RD performance comparison among DISCOVER, SADlin, FusElastic, and Oracle for Stefan, Foreman, Bus, and Coastguard sequences, for a GOP size of 8.

6.3 Conclusion and Future work

In this chapter, new approaches have been proposed to combine the global and local motion estimations, based on the foreground objects. In the first one, elastic curves [14] are used to estimate the contour of the foreground objects. Based on the estimated contour, the fusion of GMC SI and MCTI SI is performed.

Second, the foreground objects are estimated using MCTI and FOMC techniques. In this case, for the local motion, MCTI SI and the estimated foreground objects are available. Thus, two approaches for the fusion are proposed. The first one aims at fusing GMC SI with the estimated foreground objects. The second one combines GMC SI and MCTI SI.

The proposed fusion methods allow consistent performance gains compared to DISCOVER codec and to our SADbin fusion method. The gain is up to 4.73 dB compared to DISCOVER codec, and up to 0.68 dB compared to SADbin, for a GOP size equal to 8.

Future work will be focusing on further improvement of the fusion in order to achieve a better RD performance. We will investigate the use of the estimated contours by elastic curves in the estimation of the foreground objects.

We are preparing and plan to submit a journal paper presenting the work in this chapter:

1 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Fusion of global and local motion estimation using foreground objects for Distributed Video Coding", *(in preparation)*.

Chapter 7

Conclusions and Future work

In this chapter, we give a summary of the contributions presented in this manuscript, conclusions and propose some directions for future work.

7.1 Summary

The Side Information (SI) has a strong impact on the Rate-Distortion (RD) performance of Distributed Video Coding (DVC). The SI can be considered as a noisy version of the Wyner-Ziv frame (WZF) being decoded. This SI is estimated using the available decoded frames at the decoder. An accurate SI (*i.e.*, a high correlation between the SI and the original WZF) allows requesting less parity bits for error correction, and at the same time, improving the quality of the decoded WZF. In this manuscript, several approaches have been proposed to improve the quality of the SI. Due to this improvement, the RD performance of DVC is enhanced, and the decoding time is reduced. The proposed approaches allow significant gains, especially for sequences containing high motion and for large GOP sizes.

First, an approach based on successive refinement of the SI after each decoded DCT band was described in Chapter 3. More specifically, a Partially Decoded WZF (PDWZF) is reconstructed after decoding each DCT band. This PDWZF is exploited, along with the backward and forward reference frames, to refine the bi-directional motion vectors. Experimental results showed that the performance of DVC is significantly improved compared to DISCOVER codec and that it can reach the performance of H.264/AVC No motion in some cases. Furthermore, the decoder processing time is reduced due to the significant improvement of the SI.

Furthermore, a new method for SI generation is described in Chapter 4, based on backward/forward motion estimation and quad-tree refinement. In addition, the reliable motion vectors are selected from the backward and forward motion estimations. Then, a new method for successive refinement of the SI is described using an adaptive search area. In this method, the search area is adapted using the PDWZF after decoding the first DCT band and the adapted search area is used to refine the SI after decoding each DCT band.

For long duration GOPs, a new approach is proposed for enhancing the decoded WZFs by carrying out again the reconstruction on the improved SI. This approach consists in using the adjacent decoded frames for re-estimating the SI. In this re-estimation, a variable block size and an adaptive search are used.

Afterwards, an estimation of a global SI is presented in Chapter 5 using a matching of feature points between the WZF and the reference frames. The feature points are extracted using Scale-Invariant Feature Transform (SIFT) algorithm. Then, the fusion of the global and local SI frames is presented to improve the SI quality. Two different approaches for the combination of the two SI frames are described. Afterwards, the improvement of the fusion during the decoding process is presented using the PDWZF and the decoded DC coefficients. Experimental results showed that the proposed methods can achieve a significant improvement compared to DISCOVER codec,outperform H.264.AVC Intra and H.264/AVC No motion, and significantly reduce the gap with H.264/AVC (Inter IB...IB configuration).

Finally, in Chapter 6, new approaches are presented for the fusion of the global and local SI. These approaches are based on the segmented foreground objects. First, a new method for estimating the foreground object contours in the SI frame is presented using elastic curves. Based on the estimated contours, the fusion of the global and local SI is performed. Then, the foreground objects are estimated by applying MCTI on the backward and forward foreground objects. Fusion of the global and local motion estimations is then performed using the estimated foreground objects. Furthermore, a new method for estimating the foreground objects is presented, along with the corresponding fusion technique.

7.2 Conclusions

In this thesis, several different approaches are presented, that aim at enhancing the performance of DVC. These approaches can be divided into two major groups:

- **1.** In this group of methods, the SI is improved using the available decoded information at the decoder side, while the encoding process of WZF is not changed compared to DISCOVER codec (*i.e.* the proposed approaches in Chapter 3 and 4). In such schemes, the encoding process complexity is kept very low. Thus, these approaches can be used for applications that require a simple encoder (*i.e.* mobile devices, low-power sensors, etc.). At the decoder, the computational complexity (*i.e.* hardware implementation) is increased compared to DISCOVER codec due to the fact that some modules are added, while the decoding time is reduced and the RD performance is improved.

- **2.** This groups concerns the fusion techniques of global and local motion estimations (*i.e.* the proposed approaches in Chapter 5 and 6). In these schemes, the global

parameters are estimated at the encoder using SIFT algorithm, and sent to the decoder. Thus, the complexity of the encoder is increased compared to DISCOVER codec. However, the encoding complexity is still saved compared to conventional H.264/AVC Intra and H.264/AVC No motion encoding. Thus, the proposed schemes can be used for low encoding complexity applications. At the decoder, a global SI is computed using the global parameters and a combination of global and local SI is performed. Thus, an additional processing for global motion compensation and for the fusion is required. However, the execution time of the decoding process is reduced due to the enhancement of the SI, which yields fewer requests through the feedback channel.

7.3 Future work

In this section, we give some points that seem interesting to investigate, in order to further improve the performance of DVC.

- **HEVC:** At the beginning, H.263+ was used in DVC for the encoding and decoding of KFs. Then, most DVC research works have used H.264/AVC Intra for KFs encoding and for performance comparison. HEVC will soon become an international standard with stable features. Thus, HEVC can be used instead of H.264/AVC for KFs encoding, to further improve the performance of DVC. Furthermore, DVC coding can be considered for high/very high resolution video, *e.g.* HDTV, 4K and beyond.

- **Elastic curves :** As shown in Chapter 6, the elastic curves have been used in order to estimate the foreground object contours in the SI. An interesting idea would be to use these estimated foreground object contours, along with the foreground objects of the reference frames, in order to further improve the estimation of the foreground objects in the SI.

- **Foreground objects :** We presented, in Chapter 6, the fusion of global and local motion estimations using the foreground objects. It would be interesting to apply the successive refinement techniques of Chapter 3 in this context: during the decoding process, the PDWZF can be used in order to refine the estimation of the foreground objects in the SI, and to further enhance the combination of the global and local motion estimations.

- **Feedback channel :** As known, if the SI was available at the encoder, an accurate estimation of the necessary amount of parity bits, to be transmitted for the decoding of the WZF, could be made at the encoder side. This step would allow decreasing the delay which is affected by the feedback channel and reducing the decoding time (by avoiding the iterative decoding process). Some research works try to create a rough estimation of the SI at the encoder by a simple average or using Fast Motion

Compensated Interpolation (FMCI) of the reference frames, in order to keep a low complexity encoding.

However, in our scheme, the global parameters are available at the encoder. These global parameters can be used to control the compression rate at the encoder (*i.e.* remove the feedback channel). For example, a global SI can be constructed at the encoder by applying the global parameters to the reference frames. Also, a local SI can be estimated by a simple average of the reference frames. Then, an oracle fusion of the global and local SI can be performed at the encoder, using the original WZF. The result of this fusion can be seen as a good estimation of the SI and can be used to predict the necessary compression rate at the encoder side.

- **Iterative refinement :** We showed, in Chapter 3, the efficiency of the successive refinement of the SI, after each decoded DCT band, in enhancing the RD performance of DVC. An interesting perspective would be to apply the SI refinement technique after each decoded bit-plane, such that the remaining bit-planes can be improved by exploiting the already decoded ones.

- **Quality assessment :** Visual quality assessment such as subjective tests and perceptually-based objective metrics can be used to measure the quality of the SI and the decoded WZFs.

Publications

Journal papers

1 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Fusion of global and local motion estimation for Distributed Video Coding", *IEEE transactions on circuits and systems for Video Technology*, vol. 23, pages 158-172, Jan. 2013 (Chapter 5).

2 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Fusion of global and local motion estimation using foreground objects for Distributed Video Coding", (in preparation) (Chapter 6).

Conference papers

1 A. Abou-Elailah, J. Farah, M. Cagnazzo, B. Pesquet-Popescu, and F. Dufaux "Improved Side Information Generation for Distributed Video Coding", *European Workshop on Visual Information Processing (EUVIP)*, July 2011, Paris, France (Chapter 3).

2 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Successive Refinement of Motion Compensated Interpolation for Transform-Domain Distributed Video Coding", *European Signal Processing Conference (EUSIPCO)*, August 2011, Barcelona, Spain (Chapter 3).

3 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Amelioration Progressive de l'Information Adjacente Pour le Codage Video Distribue", *GRETSI*, September 2011, Bordeaux, France (Chapter 3).

4 A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Successive Refinement of Side Information using Adaptive Search Area for Long Duration GOPs in Distributed Video Coding", *19th International Conference on Telecommunications (ICT 2012)*, April, Jounieh, Lebanon (Chapter 4).

5 A. Abou-Elailah, F. Dufaux, J. Farah, and M. Cagnazzo "Fusion of Global and Local Side Information using Support Vector Machine in transform-domain DVC",

European Signal Processing Conference (EUSIPCO), August 2012, Bucharest, Romania (Chapter 5).

6 A. Abou-Elailah, G. Petrazzuoli, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu "Side Information Improvement in Transform-Domain Distributed Video Coding", *SPIE, Applications of Digital Image Processing XXXV Conference*, August 2012, San Diego, CA, United States (Chapter 4).

Annex A

Finite Rate of Innovation Signals

In the present section, we aim at using the Finite Rate of Innovation (FRI) signals in the context of DVC. First, we give a brief introduction of FRI signals. Second, we focus on the application of FRI signals in the case of DVC. This application consists in estimating the parameters of an affine transformation among successive frames, at the decoder side, using quantized transmitted samples of the non-key frames. Results show that the parameters of the transformation are well-estimated for video sequences containing a unique moving object. Unfortunately, these parameters are considerably affected when real video sequences are used.

A.1 Introduction

In 2002, the notion of FRI signals was introduced by Vetterli, Marziliano, and Blu [80]. They considered a set of 1-D signals that neither are bandlimited nor they belong to a given subspace. The Shannon's theory of classical sampling can provide perfect reconstruction strategies for bandlimited signals. Therefore, FRI signals cannot be perfectly reconstructed using Shannon's theory. The authors in [80] propose new sampling schemes for perfect reconstruction of the original continuous signal out of the discrete representation provided by samples. Examples of such signals are streams of Dirac or piecewise polynomial signals. The FRI signal can be completely characterized by a finite number of parameters. Moreover, the intuitive idea behind the sampling theory for FRI signals is that by recovering exactly those parameters using the available samples, the original FRI signal can be perfectly reconstructed.

Sampling theory for FRI signals has already found applications in several areas such as resolution enhancement [81], distributed compression [82], biomedical signals like ECG signals [83], image super-resolution algorithms [84], ... etc. Feature extraction refers to the problem of finding geometrical structures such as edges, corners and ellipses into an image. Current techniques for image feature extraction are based, for example, on statistical methods that do not cope well with very low-resolution images [85]. In [84], the authors

proposed two novel feature extraction techniques based on FRI signals, that allow the exact retrieval of global features like moments or local features like step edges in low-resolution images.

A.1.1 Definition

An FRI signal can be written as:

$$x(t) = \sum_{k\in\mathbb{Z}} \sum_{r=0}^{R-1} a_{k,r}\varphi_r(t - t_k), t \in \mathbb{R}. \tag{A.1}$$

where functions $\varphi_r(t)_{r=0,\dots,R-1}$ are known, R is the number of the functions $\varphi_r(t)$ and the unknown parameters are the coefficients $a_{k,r}$ and the time shifts t_k, in the signal $x(t)$.

The function that computes the number of unknown parameters in the signal $x(t)$ over a given interval $[t_a, t_b]$ is defined as the counting function $C_x(t_a, t_b)$. Then, the rate of innovation ρ is defined as the average number of free parameters present in $x(t)$:

$$\rho = \lim_{l\to\infty} \frac{1}{l} C_x(-\frac{l}{2}, \frac{l}{2}). \tag{A.2}$$

It is important to note that bandlimited signals can be considered as a particular case of FRI signals. Let F_{max} be the maximum non-zero frequency in a bandlimited real signal $x_b(t)$. The well-known formula for the reconstruction of bandlimited signals with a sampling period $T = \frac{1}{2F_{max}}$, using the samples $x_b(nT)$, is defined as:

$$x_b(t) = \sum_{n=-\infty}^{\infty} x_b(nT) sinc(\frac{t - nT}{T}). \tag{A.3}$$

For these signals, the rate of innovation is $\rho = \frac{1}{T}$ since it has a finite number of coefficients per unit of time. The rate of innovation of a signal can be finite but is not necessarily constant with time. For this reason, one can define the local rate of innovation at time t over a moving window of size l [80].

A.1.2 Sampling Setup

The sampling process of a signal $x(t)$ involves a sampling kernel $\phi(t)$ which represents the impulse response of the acquisition device. The samples are then expressed as:

$$y_k =< x(t), \phi(t/T - k) >= \int_{\mathbb{R}} x(t)\phi(t/T - n)dt. \tag{A.4}$$

where T is the sampling period. It has been shown that a FRI signal can be perfectly reconstructed from its samples when infinite-support kernels are used [80]. However, recently it was also found [81] that many FRI signals are perfectly reconstructed from their

samples, even with some particular finite-support and physically realizable kernels, such as B-splines, exponential splines, signals with rational Fourier transform, and scaling and wavelet functions. This result has the consequence that many interesting signals can in principle be completely reconstructed from their wavelet low-pass coefficients.

A.2 FRI signals for Distributed Video Coding

In this section, we show how FRI signals can be used in DVC. The material in this section is inspired from the work of V. Chaisinhop and L. Dragotti [86]. We re-implement the method proposed in [86] for mono-view DVC and present some of the obtained results. Then, we aim at extending the proposed method to the case of multi-view DVC.

This section is organized as follows. First, we introduce the Sampling of 2D FRI Signals and show how the continuous moments can be retrieved from the samples if a specific family of sampling kernels is used. Second, the estimation of affine parameters using higher order moments is explained. Third, applications of FRI signals in mono-view and multi-view DVC are investigated. Finally, we show the efficiency of those applications for synthetic video sequences that are generated using a real object scene and the reason behind the failing of the application of FRI signals in multi-view DVC.

A.2.1 Sampling of 2-D FRI Signals

In this section, we introduce the family of sampling kernels that reproduce polynomials and therefore satisfy the Strang-Fix conditions [81, 87]. Let $f(x, y)$ be a 2-D continuous signal and $\varphi(x, y)$ be the 2-D sampling kernel, with $x, y \in \mathbb{R}$. The obtained samples by convolving $f(x, y)$ with $\varphi(x, y)$ can be expressed as:

$$S_{m,n} =< f(x, y), \varphi(x/T - m, y/T - n) >, \tag{A.5}$$

where m and $n \in Z$.

The sampling kernel $\varphi(x, y)$ is considered to satisfy the polynomial reproduction property. Thus, the sampling kernel can be used to produce a polynomial function as follows:

$$\sum_{m \in Z} \sum_{n \in Z} c_{m,n}^{p,q} \varphi(x/T - m, y/T - n) = x^p y^q, \tag{A.6}$$

with p and $q \in Z$.

On the other side, the continuous geometric moment $m_{p,q}$ of order $(p+q)$ of the signal

$f(x,y)$ can be expressed as:

$$\begin{aligned} m_{p,q} &= \int\int f(x,y)x^p y^q dxdy \\ &= < f(x,y), x^p y^q > \end{aligned} \tag{A.7}$$

Using Eq. A.6, the continuous geometric moments can be written as:

$$\begin{aligned} m_{p,q} &= < f(x,y), \sum_{m\in Z}\sum_{n\in Z} c_{m,n}^{p,q}\varphi(x/T-m, y/T-n) > \\ &= \sum_{m\in Z}\sum_{n\in Z} c_{m,n}^{p,q} < f(x,y), \varphi(x/T-m, y/T-n) > \end{aligned} \tag{A.8}$$

Finally, the continuous geometric moments (using Eq. A.5) are expressed as:

$$m_{p,q} = \sum_{m\in Z}\sum_{n\in Z} c_{m,n}^{p,q} S_{m,n}. \tag{A.9}$$

Therefore, given a sampling kernel $\varphi(x,y)$ satisfying Eq. A.6, a set of coefficients $c_{p,q}^{m,n}$ can be computed. Using those coefficients, the continuous geometric moments can be retrieved from an arbitrarily low-resolution set of samples $S_{m,n}$ as shown in Eq. A.9.

A.2.2 Affine parameters estimation

In this section, we assume that the disparity between two adjacent frames f_k and f_{k+1} can be modeled by an affine transformation. In this case, the relationship between the positions of the pixels of f_k and f_{k+1} can be written as follows:

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} a_{xx} & a_{xy} \\ a_{yx} & a_{yy} \end{pmatrix} \begin{pmatrix} x_k \\ y_k \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{A.10}$$

where (x_{k+1}, y_{k+1}) and (x_k, y_k) are the pixels coordinates in f_{k+1} and f_k respectively, and $\{a_{xx}, a_{xy}, a_{yx}, a_{yy}, t_x, t_y\}$ are the affine transform parameters. In [88], the author proposed an approach that consists in retrieving the affine parameters using second and higher order moments of f_k and f_{k+1} (for more details, the reader can refer to [88]).

A.2.3 Application of FRI signals in mono-view DVC

Let I_k (k $= 0, 1, ..., N$) be $k+1^{\text{th}}$ frame of a video sequence, where the relationship among successive frames is considered to be an affine transformation. Fig. A.1 shows the scheme of the proposed method [86] for mono-view DVC. The first frame I_0 (let us call it key-frame) of the sequence is encoded and decoded using JPEG2000 codec. The non-key-frames I_k,

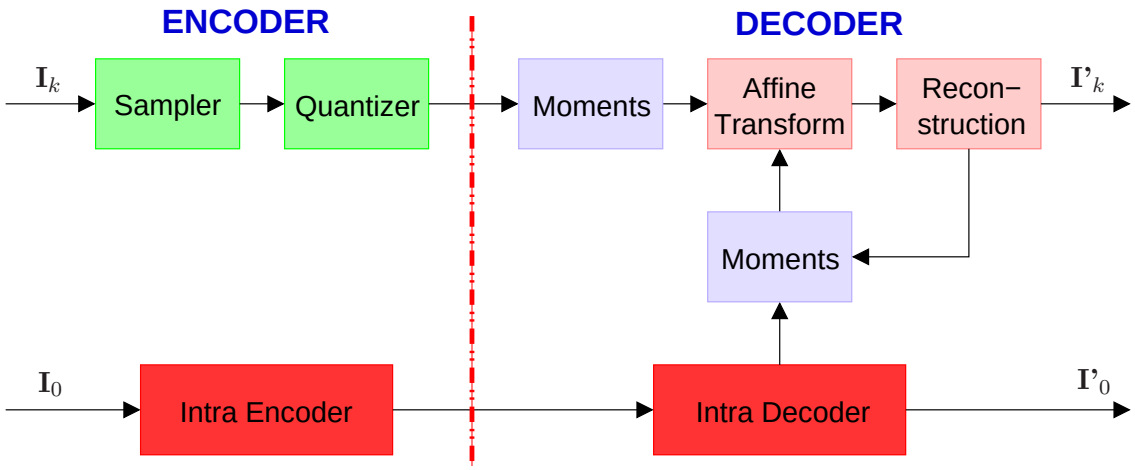

Figure A.1: Coding schema with intraframe encoding and interframe decoding based on the concept of sampling of signals with FRI.

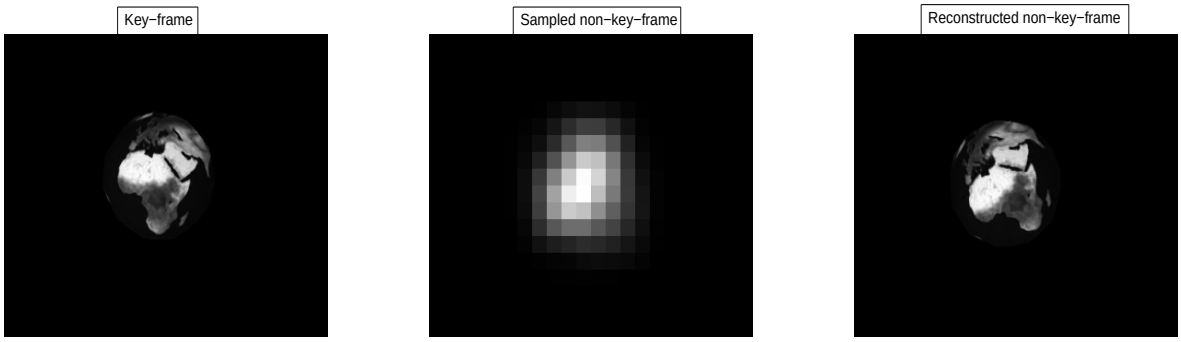

Figure A.2: Reconstructed non-key-frame using the key-frame and the sampled non-key-frame.

$k = 1, 2, ..., N$, are sampled using a sampling kernel $\varphi(x, y)$. This kernel satisfies Eq. A.6, *i.e.* the used kernel can produce a polynomial function $x^p y^q$. Afterwards, the obtained samples are quantized and transmitted to the decoder.

At the decoder side, the continuous geometric moments of the key-frame I_0 are directly computed as follows:

$$m_{p,q} = \int \int (I_0) x^p y^q dx dy \tag{A.11}$$

As shown in Eq. A.9, the continuous geometric moments of the non-key-frame I_k can be retrieved using the quantized samples, since those samples are generated by a sampling kernel that satisfies the condition in Eq. A.6. As previously mentioned, the affine parameters can be estimated between two frames using the second and higher order moments [88]. Thus, the affine parameters between the key-frame I_0 and the non-key-frame I_k are computed using the continuous geometric moments. Then, the estimated parameters are applied to the decoded key-frame I'_0 to obtain the reconstructed non-key-frame I'_k. Note that the received quantized samples from I'_k can also be used in order to

Figure A.3: Frames from MIT sequence.

improve the quality of the reconstructed non-key-frame I'_k.

In order to assess the performance of this scheme, a synthetic video is created using a real object (left image in Fig. A.2). The remaining frames are created by applying an affine transformation on the first frame, *i.e.*, the real object frame. Fig. A.2 shows the key-frame I_0, the samples of the non-key-frame I_k, and the reconstructed non-key-frame I'_k. I'_k is obtained using the moments of the key-frame and the retrieved moments from the samples of the non-key-frame. For a real video sequence with a fixed background, the proposed scheme [86] can be efficiently used. First, the approach consists in extracting the objects from the background. Second, the same scheme can be applied on each moving object (assuming that the objects movements along the video sequence can be modeled using affine transformations) and the background is encoded and decoded once using JPEG2000 codec. At the decoder, the reconstructed objects and the decoded background are used to obtain the decoded frame. For more details, we refer the reader to [86].

A.2.4 Application of FRI signals in multi-view DVC

In the present section, we extend the previous study to the case of multiple cameras capturing overlapped images from the same scene, with different viewing positions. It is important to note that correlation exploitation via motion search requires significant computing resources, and exploiting inter-view correlation at the encoder implies that a large amount of data must be communicated between cameras. Since most of the current video capturing devices have limited computing ability and power supply, we are interested in multi-view DVC since it avoids the motion search and inter-view communication at the encoder. In other words, the inter-view communication is shifted to the decoder to exploit the correlation among the views.

Inspired from [86], we try to apply the same scheme to the case of multi-view DVC. In the general case, the disparity among the different views is modeled by a Homography transform [62]. However, for the sake of simplicity, we assume that this disparity can be modeled by an affine transformation. At each moment, we have a frame from each one of the

cameras (N in this study). In our scheme, all frames originating from the central camera are encoded and decoded using JPEG2000 codec, and frames from the other cameras are sampled and quantized before being transmitted to the decoder (*i.e.* using the same encoding process of the non-key-frames in the previous scheme). At the decoder, non-key-frames can be reconstructed using the moments of the frames from the central camera and the retrieved moments from the samples of the frames from the other cameras.

Before studying the feasibility of the proposed scheme for multi-view DVC, simulations are performed on MIT sequence (mono-view sequence). Note that the relationship among frames is an affine transformation in MIT sequence. The quality of the reconstructed frames is compared to [89], in order to assess the accuracy of the estimated affine parameters. The obtained results in [89] are significantly better than that of the proposed method (more than 5 dB for the reconstructed frames). Fig. A.3 shows two frames from the MIT sequence, where we can observe that a part of the border (contour) in the first frame is clipped when a zoom is done in the second frame. We can conclude that the clipped border significantly disturbs the efficiency of the proposed method. Similarly, the lost pixels among views from different positions disturb the performance of the proposed scheme for multi-view DVC, therefore limiting its applicability to this context.

A.2.5 Conclusion

In this section, the state of the art of FRI signals was briefly shown. Moreover, the application of FRI signals [86] in DVC was explained. For video sequences that contain a fixed background or a moving object, the proposed method can give a significant gain compared to JPEG2000. However, the proposed method cannot be efficiently applied on real video sequences due to the motion complexity in such contexts, and because of the clipped pixels among frames, especially in multi-view DVC. In the remaining of this thesis, other new techniques will be proposed in order to improve the RD performance of DVC.

Bibliography

[1] "Cisco Visual Networking Index (VNI)," Tech. Rep., 2012. sections (document), 1

[2] J. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. IT-19, pp. 471–480, Jul. 1973. sections (document), 1, 2.4

[3] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, Jul. 1976. sections (document), 1, 2.4

[4] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M.Ouaret, "The DISCOVER codec: Architecture, techniques and evaluation," in *Proc. of Picture Coding Symposium*, Lisboa, Portugal, Oct. 2007. sections (document), 1, 2.5, 2.5.2, 2.5.4, 3.2, 3.3, 4.1.3, 4.2.1, 4.2.2, 5, 5.3.3, 5.4.3, 6.2

[5] "Discover project," http://www.discoverdvc.org/. sections (document), 1, 2.5, 2.5.2, 3.2, 3.3, 4.2.1, 4.2.2, 5, 5.3.3, 5.4.3, 6.2

[6] B. Girod, A. Aaron, S. Rane, and D. Rebello-Monedero, "Distributed video coding," *Proceedings of the IEEE*, vol. 93, pp. 71–83, Jan. 2005. sections (document), 1, 2.5, 2.5.2, 4.1.1

[7] J. Ascenso, C. Brites, and F. Pereira, "Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding," in *5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services*, Slovak, Jul. 2005. sections (document), 1, 2.5.2, 3.2, 4, 4.1.1, 6.1.3

[8] S. Ye, M. Ouaret, F. Dufaux, and T. Ebrahimi, "Improved side information generation for distributed video coding by exploiting spatial and temporal correlations," *EURASIP Journal on Image and Video Processing*, vol. 2009, p. 15 pages, 2009. sections (document), 2.5.3, 3.1, 3.2

[9] J. Ascenso, C. Brites, F. Dufaux, A. Fernando, T. Ebrahimi, F. Pereira, and S. Tubaro, "The VISNET II DVC Codec: Architecture, Tools and Performance," in *Proc. of the*

18th European Signal Processing Conference (EUSIPCO), 2010. sections (document), 1, 2.5, 2.5.3, 2.5.4, 3.1, 3.3.3, 3.7

[10] R. Martins, C. Brites, J. Ascenso, and F. Pereira, "Refining side information for improved transform domain Wyner-Ziv video coding," *IEEE Transactions on circuits and systems for video technology*, vol. 19, no. 9, pp. 1327 – 1341, Sep. 2009. sections (document), 1, 3.1, 3.3.3, 3.7

[11] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," in *VCEG Meeting*, Austin, USA, Apr. 2001. sections (document), 3.3.3, 4.1.3, 4.3, 4.2.2, 4.5, 4.3.3, 5.3.3, 6.2

[12] A. Abou-Elailah, J. Farah, M. Cagnazzo, B. Pesquet-Popescu, and F. Dufaux, "Improved side information for distributed video coding," in *3nd European Workshop on Visual Information Processing (EUVIP)*, Paris, France, Jul. 2011, pp. 42 – 49. sections 4, 4.2, 4.2.2, 4.5, 4.3.3

[13] G. Petrazzuoli, T. Maugey, M. Cagnazzo, and B. Pesquet-Popescu, "Side information refinement for long duration GOPs in DVC," in *IEEE Workshop on Multimedia Signal Processing (MMSP)*, Saint-Malo, France, 2010. sections (document), 5, 4.3.1, 4.3.2, 4.3.3, 4.7, 4.3.4

[14] A. Srivastava, E. Klassen, S. Joshi, and I. Jermyn, "Shape analysis of elastic curves in euclidean spaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 1415–1428, Jul. 2011. sections (document), 1, 6, 6.1.2, 6.6, 6.2, 6.12, 6.3

[15] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003. sections 1, 2.2, 2.5

[16] R. Puri and K. Ramchandran, "PRISM: A video coding architecture based on distributed compression principles," *EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M03/6*, 2003. sections 1, 2.5, 2.5.1

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91 – 110, 2004. sections 1, 5, 5.2.1

[18] S. Joshi, E. Klassen, A. Srivastava, and I. Jermyn, "A novel representation for Riemannian analysis of elastic curves in $\mathbb{R}^n$," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2007, pp. 1–7. sections 1, 6

[19] P. Read and M. Meyer, *Restoration of Motion Picture Film.* Publisher: Butterworth-Heinemann, 2000. sections 2.1

[20] D. Salomon, *Data Compression: The Complete Reference, Fourth Edition*, 2007, with contributions by Giovanni Motta and David Bryant. sections 2.1

[21] S. Rane and G. Sapiro, "Evaluation of JPEG-LS, the new lossless and controlled-lossy still image compression standard, for compression of high-resolution elevation data," *IEEE Transactions on Geoscience and Remote sensing*, vol. 39, pp. 2298–2306, Oct. 2001. sections 2.1

[22] ISO/IEC and ITU-T, "ISO/IEC 13818-2 | ITU-T Recommendation H.262: Information technology - generic coding of moving pictures and associated audio information - Part 2: Video," ISO/IEC JTC1/SC29/WG11 and ITU-T, 1995. sections 2.1

[23] ISO/IEC, "MPEG-4 visual, Information technology-coding of audio-visual objects-part 2: Visual," ISO/IEC JTC1/SC29/WG1 (MPEG), 14496-2, 1999. sections 2.1

[24] ITU-T, "ITU-T Recommendation H.261: Video codec for audiovisual services at p x 64 kbit/s," ITU-T, 1990. sections 2.1

[25] ——, "ITU-T Recommendation H.263: Video coding for low bit rate communication," ITU-T, 1995. sections 2.1

[26] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia*, 2003. sections 2.2

[27] B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, "High Efficiency Video Coding (HEVC) text specification draft 6," Feb. 2012. sections 2.3

[28] B. Li, G. Sullivan, and J. Xu., "Comparison of compression performance of HEVC working draft 5 with avc high profile," 2012. sections 2.3

[29] S. Pradhan, j. Chou, and K. Ramchandran, "Duality between source coding and channel coding and its extension to the side information case," *IEEE Transactions on Information Theory*, vol. 49, no. 5, pp. 1181–1203, May 2003. sections 2.4

[30] R. Zamir, "The rate loss in the Wyner-Ziv problem," *IEEE Transactions on Information Theory*, vol. 42, no. 11, pp. 2073–2084, Nov. 1996. sections 2.4

[31] F. Pereira, L. Torres, C. Guillemot, T. Ebrahimi, R. Leonardi, and S. Klomp, "Distributed video coding: selecting the most promising application scenarios," *Signal Processing: Image Communication*, vol. 23, no. 5, pp. 339–352, 2008. sections 2.5

[32] R. Puri, A. Majumdar, and K. Ramchandran, "PRISM: A video coding paradigm with motion estimation at the decoder," *IEEE Transactions on Image Processing*, vol. 16, pp. 2436–2448, Oct. 2007. sections 2.5, 2.5.1

[33] A. Aaron, R. Zhang, and B. Girod, "Wyner-Ziv coding for motion video," in *Proceedings of Asilomar Conference on Signals, Systems and Computers*, California, USA, 2002, pp. 240–244. sections 2.5, 4.1.1

[34] A. Aaron, S. Rane, E. Setton, and B. Girod, "Transform-domain Wyner-Ziv codec for video," in *Proceedings of SPIE Visual Communications and Image Processing Conference*, California, USA, Jan. 2004. sections 2.5, 4.1.1

[35] A. Aaron, S. Rane, and B. Girod, "Wyner-Ziv video coding with hash-based motion compensation at the receiver," in *Proceedings of IEEE International Conference on Image Processing*, vol. 05, Singapore, Oct. 2004, pp. 3097–3100. sections 2.5, 3.1, 5.1

[36] W.-J. Chien, L. Karam, and G. Abousleman, "Distributed video coding with 3d recursive search block matching," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2006. sections 2.5

[37] W.-J. Chien and L. Karam, "Transform-domain distributed video coding with rate-distortion-based adaptive quantisation," *IET Image Processing*, vol. 3, pp. 340–354, Dec. 2009. sections 2.5

[38] S. S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," in *IEEE Data Compression Conference*, Snowbird, USA, Mar. 1999. sections 2.5.1

[39] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," *IEEE Transactions on Information Theory*, vol. 49, pp. 626–643, 2003. sections 2.5.1

[40] P. Ishwar, V. Prabhakaran, and K. Ramchandran, "Towards a theory for video coding using distributed compression principles," in *IEEE International Conference on Image Processing*, Barcelona, Spain, Sep. 2003. sections 2.5.1

[41] F. J. Macwilliams and N. J. A. Sloane, *The theory of error correcting codes.* Amsterdam, The Netherlands: Elseiver, 1977. sections 2.5.1

[42] T. Maugey and B. Pesquet-Popescu, "Side information estimation and new schemes for multiview distributed video coding," *Journal of Visual Communication and Image Representation*, vol. 19, no. 8, pp. 589–599, 2008. sections 2.5.2

[43] C. Brites, J. Ascenso, and F. Pereira, "Studying temporal correlation noise modeling for pixel based Wyner-Ziv video coding," in *IEEE International Conference on Image Precessing*, Atlanta, USA, Oct. 2006. sections 2.5.2, 3.2.4

[44] C. Brites and F. Pereira, "Correlation noise modeling for efficient pixel and transform domain Wyner-Ziv video coding," *IEEE Trans. on Circuit ans System for Video Technology*, vol. 18(9), pp. 1177–1190, 2008. sections 2.5.2, 3.2.4, 5.3.1

[45] D. Kubasov, J. Nayak, and C. Guillemot, "Optimal reconstruction in wyner-ziv video coding with multiple side information," in *IEEE 9th Workshop on Multimedia Signal Processing (MMSP)*, 2007, pp. 183 –186. sections 2.5.2

[46] R. Martins, C. Brites, J. Ascenso, and F. Pereira, "Adaptive deblocking filter for transform domain Wyner-Ziv video coding," *IET Signal Processing*, vol. 3, no. 6, pp. 315 – 328, Dec. 2009. sections 2.5.3

[47] C. Brites, J. Ascenso, and F. Pereira, "Improving transform domain Wyner-Ziv video coding performance," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, May 2006, pp. 525–528. sections 2.5.4

[48] J. Alparone, M. Barni, F. Bartolini, and V. Cappellini, "Adaptively weighted vector-median filters for motion fields smoothing," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, Atlanta, GA, USA, May 1996, pp. 2267 –2270. sections 2.5.4, 4.1.2

[49] J. Ascenso and F. Pereira, "Adaptive hash-based side information exploitation for efficient Wyner-Ziv video coding," in *Proc. Int. Conf. on Image Processing*, vol. 03, San Antonio, Oct. 2007, pp. 29–32. sections 3.1, 5.1

[50] C. Yaacoub, J. Farah, and B. Pesquet-Popescu, "A genetic algorithm for side information enhancement in distributed video coding," in *16th IEEE International Conference on Image Processing (ICIP)*, Cairo, Egypt, Nov. 2009, pp. 2933 –2936. sections 3.1, 4.1.1

[51] T. Maugey, C. Yaacoub, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu, "Side information enhancement using an adaptive hash-based genetic algorithm in a Wyner-Ziv context," in *IEEE International Workshop on Multimedia Signal Processing*, Saint-Malo, France, Oct. 2010, pp. 298 –302. sections 3.1

[52] J. Ascenso, C. Brites, and F. Pereira, "Motion compensated refinement for low complexity pixel based distributed video coding," in *Proceedings of the IEEE international conference on Advanced Video and Signal-Based Surveillance*, Sep. 2005, pp. 593 – 598. sections 3.1

[53] X. Fan, O. Au, N. Cheung, Y. Chen, and J. Zhou, "Successive refinement based Wyner-Ziv video compression," *Signal Processing: Image Communication*, vol. 25, pp. 47–63, Jan. 2010. sections 3.1

[54] X. Artigas and L. Torres, "Iterative generation of motion-compensated side information for distribued video coding," in *Proc. IEEE Int. Conf. Image Processing*, Nov. 2005, pp. 833–836. sections 3.1

[55] A. B. B. Adikari, W. A. C. Fernando, H. K. Arachchi, and W. A. R. J. Weerakkody, "Sequential motion estimation using luminance and chrominance information for distributed video coding of Wyner-Ziv frames," *IEEE Electron. Lett.*, vol. 42, no. 7, pp. 398–399, Mar. 2006. sections 3.1

[56] W. A. R. J. Weerakkody, W. A. C. Fernando, J. L. Martinez, P. Cuenca, and F. Quiles, "An iterative refinement technique for side information generation in DVC," in *IEEE International Conference Multimedia Expo*, Beijing, China, Jul. 2007. sections 3.1

[57] D. Varodayan, D. Chen, M. Flierl, and B. Girod, "Wyner-Ziv coding of video with unsupervised motion vector learning," *Signal Processing: Image Communication*, vol. 23, no. 5, pp. 369–378, Jun. 2008. sections 3.1

[58] M. Badem, W. Fernando, J. Martinez, and P. Cuenca, "An iterative side information refinement technique for transform domain distributed video coding," in *IEEE International Conference on Multimedia and Expo, ICME*, 2009, pp. 177 – 180. sections 3.1

[59] G. Petrazzuoli, M. Cagnazzo, and B. Pesquet-Popescu, "High order motion interpolation for side information improvement in DVC," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Jun. 2010, pp. 2342 – 2345. sections 3.1

[60] A. Aaron, E. Setton, and B. Girod, "Towards practical Wyner-Ziv coding of video," in *Proceedings of IEEE International Conference on Image Processing*, Barcelona, Spain, 2003, pp. 869–872. sections 4.1.1

[61] X. Guo, Y. Lu, F. Wu, W. Gao, and S. Li, "Distributed multi-view video coding," in *SPIE Visual Commun and Image Processing*, vol. 6077, California, USA, 2006, pp. 15–19. sections

[62] M. Ouaret, F. Dufaux, and T. Ebrahimi, "Fusion-based multiview distributed video coding," in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, California, USA, 2006. sections A.2.4

[63] ——, "Multiview distributed video coding with encoder driven fusion," in *European Signal Processing Conference (EUSIPCO)*, Poznan, Poland, 2007. sections 4.1.1

[64] T. N. Dinh, G. Lee, J.-Y. Chang, and H.-J. Cho, "A novel motion compensated frame interpolation method for improving side information in distributed video coding," in *International Symposium on Information Technology Convergence*, Joenju, South Korea, 2007, pp. 179–183. sections 4.1.1

[65] S. Argyropoulos, N. Thomos, N. Boulgouris, and M. Strintzis, "Adaptive frame interpolation for Wyner-Ziv video coding," in *IEEE Workshop on Multimedia Signal Processing*, Chania, Crete, Greece, Oct. 2007. sections 4.1.1

[66] X. Huang and S. Forchhammer, "Improved side information generation for distributed video coding," in *IEEE International Workshop on Multimedia Signal Processing*, Cairns, Queensland, Australia, Oct. 2008. sections 4.1.1

[67] R. Hansel and E. Muller, "Global motion guided adaptive temporal inter-/extrapolation for side information generation in distributed video coding," in *IEEE International Conference on Image Processing*, Brussels, Belgium, Sep. 2011, pp. 2681 – 2684. sections 5.1, 5.4.3

[68] M. Guo, Z. Xiong, F. Wu, D. Zhao, X. Ji, and W. Gao, "Witsenhausen-Wyner video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, pp. 1049 – 1060, 2011. sections 5.1

[69] H. Xiong, H. Lv, Y. Zhang, L. Song, Z. He, and T. Chen, "Subgraphs matching-based side information generation for distributed multiview video coding," *EURASIP Journal on Advances in Signal Processing*, p. 17 pages, 2009. sections 5.1

[70] T. Maugey, W. Miled, M. Cagnazzo, and B. Pesquet-Popescu, "Fusion schemes for multiview distributed video coding," in *17th European Signal Processing Conference (EUSIPCO)*, Scotland, Aug. 2009. sections 5.1, 5.3.1

[71] F. Dufaux, "Support vector machine based fusion for multi-view distributed video coding," in *17th International Conference on Digital Signal Processing (DSP)*, Corfu, Aug. 2011, pp. 1 –7. sections 5.1

[72] T. Clercks, A. Munteanu, J. Cornelis, and P. Schelkens, "Distributed video coding with shared encoder/decoder complexity," in *Proc. IEEE International Conference on Image Processing*, San Antonio, TX, Sep. 2007. sections 5.1, 5.4.3

[73] H. Chen and E. Steinbach, "Flexible distribution of computational complexity between the encoder and the decoder in distributed video coding," in *Proc. IEEE International Conference on Multimedia and Expo*, Hannover, Germany, Jun. 2008. sections 5.1, 5.4.3

[74] F. Dufaux and T. Ebrahimi, "Encoder and decoder side global and local motion estimation for distributed video coding," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2010, pp. 339 – 344. sections 5.1, 5.4.3

[75] "SVM implementation," http://www.cs.cornell.edu/People/tj/svm_light/. sections 5.3.2

[76] C. Brites, J. Ascenso, J. Pedro, and F. Pereira, "Evaluating a feedback channel based transform domain Wyner-Ziv video codec," *Signal Processing: Image Communication*, vol. 23, no. 4, pp. 269–297, Apr. 2008. sections 5.3.3

[77] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," *Computer Vision and Image Understanding (CVIU)*, vol. 110, no. 3, pp. 346 – 359, 2008. sections 5.3.3

[78] Y.-M. Chen, I. Bajic, and P. Saeedi, "Coarse-to-fine moving region segmentation in compressed video," in *10th Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, May 2009, pp. 45–48. sections 6.1

[79] Y.-M. Chen and I. Bajic, "Compressed-domain moving region segmentation with pixel precision using motion integration," in *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PacRim)*, Aug. 2009, pp. 442–447. sections 6.1

[80] M. Vetterli, P. Marziliano, and T. Blu, "Sampling signals with finite rate of innovation," *IEEE Trans. Signal Processing*, vol. 50, no. 6, pp. 1417 – 1428, Jun. 2002. sections A.1, A.1.1, A.1.2

[81] P. L. Dragotti, M. Vetterli, and T. Blu, "Sampling moments and reconstructing signals of finite rate of innovation : Shannon meets Strng-Fix," *IEEE Trans. on Signal Processing*, vol. 55, pp. 1741 – 1757, May 2007. sections A.1, A.1.2, A.2.1

[82] N. Gehrig and P. L. Dragotti, "Distributed sampling and compression of scenes with finite rate of innovation in camera sensor networks," in *Data Communication Conference (DCC)*, Mar. 2006. sections A.1

[83] Y. Hao, P. Marziliano, M. Vitterli, and T. Blu, "Compression of a ECG as a signal with finite rate of innovation," in *IEEE Int. Conf. of Engineering in Medicine and Biology Society*, Sep. 2005, pp. 7564 – 7567. sections A.1

[84] L. Baboulaz and P. L. Dragotti, "Exact feature extraction using finite rate of innovation principles with an application to image super-resolution," *IEEE Transactions on Image Processing*, vol. 18, no. 2, pp. 281 – 298, Feb. 2009. sections A.1

[85] D. Capel and A. Zisserman, "Computer vision applied to superresolution," *IEEE Signal Proc. Mag.*, pp. 75 – 86, 2003. sections A.1

[86] V. Chaisinthop and P. L. Dragotti, "Distributed video coding based on sampling of signals with finite rate of innovation," *Society of Photo-Optical Instrumentation*, 2007. sections A.2, A.2.3, A.2.4, A.2.5

[87] G. Strang and G. Fix, "A Fourier analysis of the finite element variational method," in *Constructive Aspect of Functional Analysis*, Rome - Italy, pp. 796–830. sections A.2.1

[88] J. Heikkila, "Pattern matching with affine moment descriptors," *Pattern Recognition*, vol. 37, no. 9, pp. 1825 – 1834, 2004. sections A.2.2, A.2.3

[89] F. Dufaux and J. Konrad, "Efficient, robust, and fast global estimation for video coding," *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 497 – 501, Mar. 2000. sections A.2.4