REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE YAHIA FARES DE MEDEA

Faculté de **T**echnologie

Département du **G**énie **E**lectrique

Mémoire de fin d'études de Master

Filière : Télécommunications
Spécialité : Système de Télécommunications

Estimation de mouvement dans les images vidéo par la technique de "Block Matching" avec test de bloc de différence

Présentée par :

GHERNAOUT Safa

Proposés et Dirigés par :

Dr. LAIDI Kamel

Promotion 2022



Remerciements

Au terme de ce mémoire, je tiens remercier d'abord et avant tout ALLAH qui nous a donné la capacité, la force, le courage et la patience d'accomplir ce travail.

Je tiens également à adresser mes sincères remerciements au promoteur M. Laidi Kamel, pour son encadrement surtout pour sa confiance, et ses précieux conseils qui m'ont beaucoup aidé.

Je remercie beaucoup M. Bailiche et M. Mimen d'avoir accepté l'évaluation de ce travail.

Enfin, je tiens à exprimer ma profonde gratitude à ma famille et mes amis qui m'ont toujours soutenu. Et à tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail.

Dédicaces

A tout étudiant en sciences qui cherche à acquérir des connaissances et à apporter son équilibre scientifique et culturel

Aux deux plus grandes personnes qui ont illuminé mon chemin et ont été mon soutien après ALLAH. A ceux qui m'ont appris que le monde est un combat et que son arme est la science et la connaissance, mes chères parents, que Dieu les protège

A toute la généreuse famille qui m'a soutenu et qui sont toujours, mes chers frères et chères sœurs

A mes amis qui ont partagé mes moments

A tous mes collègues dans le domaine des télécommunications



À tous les cadres du Département de génie éclectique et à tous les travailleurs de la Faculté de technologie de l'Université Dr. Yahya Fares à Médéa

A tous ceux qui ont eu un impact dans ma vie et à tous ceux que mon cœur a aimés et qui ont été oubliés par ma plume

Je dédie ce modeste travail

Ghernaout Safa

ملخص

هناك العديد من التقنيات لتقدير الحركة بين الصور في مقاطع الفيديو. من بين هذه التقنيات تحتل تقنية مطابقة الكتل مكانا كبيرا. في هذا السياق و من اجل تحسين أداء هذه التقنية من حيث وقت الحساب ونسبة الإشارة على الضوضاء، وهما عاملان أساسيان، نريد تعديل هذه التقنية حيث نضيف خطوة للتمايز بين الإطارات. ستتم مقارنة النتائج مع نتائج تقنية مطابقة الكتل الكلاسيكية.

الكلمات المفتاحية: تقدير الحركة – مطابقة الكتل – وقت الحساب – نسبة الإشارة على الضوضاء.

Résumé

Il existe beaucoup de techniques d'estimation de mouvement entre frames dans une séquence vidéo. Parmi ces techniques, la technique de mise en correspondance des blocs prend une grande place. Dans ce contexte et afin d'améliorer les performances de cette technique en terme de temps de calcul et de rapport signal sur bruit, qui sont deux facteurs primordiaux, nous voulons modifier cette technique où l'on ajoute une étape de différenciation entre frames. Les résultats seront comparés avec ceux de la technique classique de mise en correspondance des blocs.

Mots clé : estimation de mouvement – mise en correspondance des blocs – temps de calcul – le rapport signal sur bruit.

Abstract

There are many techniques for estimating motion between frames in a video sequence. Among these techniques, the block matching technique takes a large place. In this context and in order to improve the performance of this technique in terms of computation time and signal to noise ratio, which are two essential factors, we want to modify this technique where we add a step of differentiation between frames. The results will be compared with those of the classical block matching technique.

Keys words: motion estimation – block matching – computation time – signal to noise ratio.

Sommaire

Remerciement	
Dédicace	
Résumé	
Sommaire	
Liste des figures	
Liste des tableaux	
Introduction générale	1
CHAPITRE I : Notion générales sur l'image	
I.1. Introduction	3
I.2. Notion de base sur l'image	3
I.2.1. Définition de l'image	3
I.2.2. Image Analogique	3
I.2.3. Image numérique	4
I.2.4. Caractéristiques de l'image numérique	4
I.2.4.1. Pixel	4
I.2.4.2. Dimension	4
I.2.4.3. Taille d'une image	4
I.2.4.4. Résolution	5
I.2.4.5. Histogramme	5
I.2.4.6. Luminance	6
I.2.4.7. Contraste	6
I.2.4.8. Bruit	6
I.2.5. Les type des images et des graphes	6
I.2.5.1. Image binaire	6
I .2.5.2. Images à niveaux des gris	7
I.2.2.5. Image couleur 24 bits	7
I.2.6. Le codage de la couleur	8
I.2.7.Formats d'images	9
I.2.7.1. Format GIF (Graphics Interchange)	9
I.2.7.2. Format JPEG (Joint Photographic Experts Group)	9
I.2.7.3. Format PNG (Portable Network Graphics)	10
I.2.7.4. Format TIFF (Tagged Image File Format)	10
I.2.7.5. Format Windows BMP	10
I.2.7.6. Format EXIF (Exchangeable Image File)	10
I.3. Concepts sur la vidéo	11
I.3.1. Définition de la vidéo	11
I.3.2. Compression vidéo	11
I.3.2.1. Contraintes	12
I.3.2.2. La compression de données	12

I.3.2.2.1. La redondance spatial	12
I.3.2.2.2. La redondance temporelle	12
I.3.2.2.3. La redondance psycho visuelle	13
I.3.3. Décomposition de la séquence vidéo compressée	14
I.3.4. Les méthodes de compression vidéo	14
I.3.4.1. Compression avec perte	14
I.3.4.2. Compression sans perte	15
I.3.5. Codec vidéo	15
I.3.5.1. Principe	15
I.3.5.2. Standards de compression vidéo	15
I.3.7. Estimation de mouvement	17
I.4. Conclusion	18
CHAPITRE II : Estimation de mouvement dans l'image vidéo	
II.1. Introduction	19
II.2. Le principe de l'estimation de mouvement	19
II.3. Domaines d'applications	20
II.4. Le mouvement dans la séquence vidéo	20
II.5. Classification de mouvement	21
II.5.1. Le mouvement réel	22
II.5.1.1. Le mouvement d'objet rigide	22
II.5.1.2. Le mouvement d'objet déformables	22
II.5.2. Le mouvement apparent	22
II.5.3. Le mouvement estimé	23
II.6. Estimation de mouvement	23
II.7. Les difficultés de l'estimation de mouvement	25
II.8. Modèles du champ de vecteurs mouvement	26
II.8.1. Modèle de translation	27
II.8.2. Méthodes non paramétrique	28
II.8.3. Modèle paramétrique du mouvement	29
II.9. Méthode différentielles	30
II.9.1. Equation de contrainte du mouvement	30
II.9.2 : Méthode de Horn et Schunck :	32
II.9.3. Méthodes différentielles avancées	33
II.9.3.1. Méthodes différentielles directionnelles	35
II.9.3.2. Méthodes différentielles hiérarchiques	35
II.10. Méthodes de mise en correspondance	35
II.10.1. Méthodes de mise en correspondance dans le plan transformée	36
II.10.2. Méthodes de mise en correspondance dans le plan image	38
II.11. Conclusion	39
CHAPITRE III : Technique de block Matching	
III.1. Introduction	40
III.2. Principe de mise en correspondance	40
III.3. Critères de mise en correspondance	41
III.3.1. Différence absolue	42

III.3.2. Erreur quadratique MSD ou MSE	42
III.4. Dimension optimal de bloc et de la fenêtre de recherche	42
III.5. Stratégies de recherche	43
III.5.1. Recherche en *n* pas	44
III.5.2. Recherche 2D-logarithmique	45
III.5.3. Recherche orthogonale	45
III.6. Les récents algorithmes étudiés	46
III. 6. 1. L'algorithme de recherche complet FS	46
III.6.2. L'algorithme à trois étapes de recherche TSS	47
III.6.3. Le nouvel algorithme à trois étapes de recherche NTSS	48
III.6.4. L'algorithme à quatre étapes de recherche FSS	49
III.6.5. L'algorithme de recherche en diamant DS	51
III.6.6. L'algorithme de recherche minimum MIN	53
III.7. Conclusion	54
CHAPITRE IV : Technique de BM avec test de Bloc de différence	
IV.1. Introduction	55
IV.2. Définition de MATLAB	55
IV.3. Application les algorithmes de BM à MATLAB	55
IV.4. Résultats des algorithmes de BM par applications directes	56
IV.4.1 Résultats de l'algorithme FS	57
IV.4.2. Résultats de l'algorithme TSS	58
VI. 4.3. Résultats de l'algorithme NTSS	59
VI. 4.4. Résultats de l'algorithme FSS	60
VI. 4.5. Résultats de l'algorithme DS	61
VI. 4.6. Résultats de l'algorithme MIN	62
IV.5. Remarques	63
IV.6. Evaluation	63
IV.6.1. PSNR (Peak Signal-to-Noise Ratio)	63
IV.6.2. Evaluation du Temps de calcul (C.T)	64
IV.7. Résultats d'évaluation	64
IV.8. Discussion des résultats de BMA par application direct	65
IV.9. algorithmes de BM avec test du bloc de différence	65
IV.9.1. La moyenne (ou la différence absolue MAD)	65
IV.9.2. La variance	66
IV.9.3. Ecart type	66
IV.10. Résultats des algorithmes de BM avec test de bloc de différence	66
IV.10.1. Résultats de l'algorithme FS avec test de bloc de différence	66
IV.10.2. Résultats de l'algorithme TSS avec test de bloc de différence	68
IV.10.3. Résultats de l'algorithme NTSS avec test de bloc de différence	69
IV.10.4. Résultats de l'algorithme FSS avec test de bloc de différence	70
IV.10.5. Résultats de l'algorithme DS avec test de bloc de différence	71
IV.10.6. Résultats de l'algorithme MIN avec test de bloc de différence	72
IV.11. Comparaison entre les résultats de BM par application direct et les résultats de BM avec test de bloc de différence	73

IV.11.1. Comparaison entre les résultats de FS par application direct et les résultats de FS avec test de bloc de différence	73
IV.11.2. Comparaison entre les résultats de TSS par application direct et les résultats de TSS avec test de bloc de différence	74
IV.11.3. Comparaison entre les résultats de NTSS par application direct et les résultats de NTSS avec test de bloc de différence	75
IV.11.4. Comparaison entre les résultats de FSS par application direct et les résultats de FSS avec test de bloc de différence	76
IV.11.5. Comparaison entre les résultats de DS par application direct et les résultats de DS avec test de bloc de différence	77
IV.11.6. Comparaison entre les résultats de MIN par application direct et les résultats de MIN avec test de bloc de différence	78
IV.12. Discussion des résultats	79
IV.13. Conclusion	79
Conclusion générale	
Conclusion générale	80
Bibliographie	

Liste des figures

Figure I.1	*Le pixel*	4
Figure I.2.	*Histogramme des niveaux de gris*	5
Figure I.3	*Image binaire. (Lena)*	7
Figure I.4	*Image à niveaux de gris de résolution 512*512.*	7
Figure I.5	*La somme des valeurs des trois couleurs primaires (R, G .B).*	8
Figure I.6	*La redondance spatiale : la redondance de la couleur noire est très prononcée.*	12
Figure I.7	*La redondance temporelle : la scène avec un fond fixe seul le visage de la personne qui bouge*	12
Figure I.8	*Format YUV 4 :1 :1.*	13
Figure I.9	*Format YUV.*	14
Figure I.10	*Structure d'un groupe d'image (GOP).*	14
Figure I.11	*Codeur/décodeur.*	15
Figure I.12	*Synoptique d'un compresseur MPEG.*	17
Figure I.13	*Détection et estimation de mouvement en MPEG.*	18
Figure II.1	*Utilisation de la technologie d'estimation de mouvement*	19
Figure II.2	*La séquence vidéo*	20
Figure II.3	*Projection 3D en 2D*	21
Figure II.4	*Exemples de flot optique*	21
Figure 1I.5	*Illustration de mouvement réel et apparent.*	22
Figure II.6	*Estimation directe et inverse des vecteurs déplacements.*	24
Figure II.7	*Problème d'occlusion*	26
Figure II.8	*Problème d'ouverture*	26
Figure II.9	*Modèle de translation du mouvement*	27
Figure II.10	*Modèle de translation, avec des blocs: (a) disjoints; (b) chevauchants.*	27
Figure II.11	*Exemples de transformations des coordonnées spatiales*	29
Figure II.12	*Illustration géométrique de l'équation et de la droite de contrainte du mouvement*	32
Figure III.1	*Illustration de méthode Block Matching*	41
Figure III.2	*Principe de mise en correspondance par blocs.*	41
Figure III.3	*Stratégies de balayage de la fenêtre de recherche.*	43
Figure III.4	*Le principe de recherche en trois pas.*	44
Figure III.5	*Le principe de recherche 2D-logarithmique.*	45
Figure III.6	*Principe de recherche orthogonale.*	46
Figure III.7	*Modèle de recherche"Full Search" FS*	47
Figure III.8	*Modèle de recherche à troisétapes TSS.*	47
Figure III.9	*Diagramme bloc de l'algorithme NTSS.*	49
Figure III.10	*Région de recherche de l'algorithme NTSS.*	49

Figure III.11	*Les modèles de recherche :*	50
Figure III.12	*Modèle de recherche de l'algorithme FSS*	51
Figure III.13	*Le modèle approprié de recherche, un secteur circulaire et radium de 2 pixels. (Les 13 crois indiquent tous les points de contrôle possibles dans le cercle).*	52
Figure III.14	*Les modèles de recherche :*	52
Figure III.15	*Méthode de recherche de l'algorithme minimum*	53
Figure IV.1	*Variation du C.T pour l'algorithme FS par application direct et FS avec test de bloc de différence (FS indirect) pour* $\mu = 3$ et $\sigma = 3$	73
Figure IV.2	*Variation du PSNR pour l'algorithme FS par application direct et FS avec test de bloc de différence (FS indirect) pour* $\mu = 3$ et $\sigma = 3$	73
Figure IV.3	*Variation du C.T pour l'algorithme TSS par application direct et TSS avec test de bloc de différence (TSS indirect) pour* $\mu = 3$ et $\sigma = 3$	74
Figure IV.4	*Variation du PSNR pour l'algorithme TSS par application direct et TSS avec test de bloc de différence (TSS indirect) pour* $\mu = 3$ et $\sigma = 3$	74
Figure IV.5	*Variation du C.T pour l'algorithme NTSS par application direct et NTSS avec test de bloc de différence (NTSS indirect) pour* $\mu = 3$ et $\sigma = 3$	75
Figure IV.6	*Variation du PSNR pour l'algorithme NTSS par application direct et NTSS avec test de bloc de différence (NTSS indirect) pour* $\mu = 3$ et $\sigma = 3$	75
Figure IV.7	*Variation du C.T pour l'algorithme FSS par application direct et FSS avec test de bloc de différence (FSS indirect) pour* $\mu = 3$ et $\sigma = 3$	76
Figure IV.8	*Variation du PSNR pour l'algorithme FSS par application direct et FSS avec test de bloc de différence (FSS indirect) pour* $\mu = 3$ et $\sigma = 3$	76
Figure IV.9	*Variation du C.T pour l'algorithme DS par application direct et DS avec test de bloc de différence (DS indirect) pour* $\mu = 3$ et $\sigma = 3$	77
Figure IV.10	*Variation du PSNR pour l'algorithme DS par application direct et DS avec test de bloc de différence (DS indirect) pour* $\mu = 3$ et $\sigma = 3$	77
Figure IV.11	*Variation du C.T pour l'algorithme MIN par application direct et MIN avec test de bloc de différence (MIN indirect) pour* $\mu = 3$ et $\sigma = 3$	78
Figure IV.12	*Variation du PSNR pour l'algorithme MIN par application direct et MIN avec test de bloc de différence (MIN indirect) pour* $\mu = 3$ et $\sigma = 3$	78

Liste des tableaux

Tableau I.1	*Tableau comparatif entre tailles des images*	5
Tableau IV.1	*Résultats de l'algorithme FS sur la séquence 'Akiyo'*	57
Tableau IV.2	*Résultats de l'algorithme FS sur la séquence 'Flower Garden'*	57
Tableau IV.3	*Résultats de l'algorithme FS sur la séquence 'Foreman'*	57
Tableau IV.4	*Résultats de l'algorithme TSS sur la séquence 'Akiyo'*	58
Tableau IV.5	*Résultats de l'algorithme TSS sur la séquence 'Flower Garden'*	58
Tableau IV.6	*Résultats de l'algorithme TSS sur la séquence 'Foreman'*	58
Tableau IV.7	*Résultats de l'algorithme NTSS sur la séquence 'Akiyo'*	59
Tableau IV.8	*Résultats de l'algorithme NTSS sur la séquence 'Flower Garden'*	59
Tableau IV.9	*Résultats de l'algorithme NTSS sur la séquence 'Foreman'*	59
Tableau IV.10	*Résultats de l'algorithme FSS sur la séquence 'Akiyo'*	60
Tableau IV.11	*Résultats de l'algorithme FSS sur la séquence 'Flower Garden'*	60
Tableau IV.12	*Résultats de l'algorithme FSS sur la séquence 'Foreman'*	60
Tableau IV.13	*Résultats de l'algorithme DS sur la séquence 'Akiyo'*	61
Tableau IV.14	*Résultats de l'algorithme DS sur la séquence 'Flower Garden'*	61
Tableau IV.15	*Résultats de l'algorithme DS sur la séquence 'Foreman'*	61
Tableau IV.16	*Résultats de l'algorithme DS sur la séquence 'Akiyo'*	62
Tableau IV.17	*Résultats de l'algorithme DS sur la séquence 'Flower Garden'*	62
Tableau IV.18	*Résultats de l'algorithme DS sur la séquence 'Foreman'*	62
Tableau IV.19	*Résultats PSNR et C.T sur les algorithmes de BM*	64
Tableau IV.20	*PSNR et C.T pour l'algorithme FS avec test de bloc de différence*	66
Tableau IV.21	*Résultats de l'algorithme FS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$	67
Tableau IV.22	*PSNR et C.T pour l'algorithme TSS avec test de bloc de différence*	68
Tableau IV.23	*Résultats de l'algorithme TSS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$	68
Tableau IV.24	*PSNR et C.T pour l'algorithme NTSS avec test de bloc de différence*	69
Tableau IV.25	*Résultats de l'algorithme NTSS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$	69
Tableau IV.26	*PSNR et C.T pour l'algorithme FSS avec test de bloc de différence*	70
Tableau IV.27	*Résultats de l'algorithme FSS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$	70
Tableau IV.28	*PSNR et C.T pour l'algorithme DS avec test de bloc de différence*	71
Tableau IV.29	*Résultats de l'algorithme DS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$	71
Tableau IV.30	*PSNR et C.T pour l'algorithme MIN avec test de bloc de différence*	72
Tableau IV.31	*Résultats de l'algorithme MIN avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$	72

Introduction générale

Introduction générale

Des milliards de vidéos sont visionnées chaque jour sur différentes plateformes. Les séquences vidéo sont des sources d'information et un moyen de communication. Elles sont même parmi les données les plus vues, partagées et consommées dans le monde, les séquences vidéo peuvent contenir des informations très importantes, il est donc intéressant de développer des techniques d'analyse vidéo pour extraire ces informations. Parmi ces dernières nous trouvons des informations liées au mouvement, qui joue un rôle essentiel dans de nombreuses applications telles que la compression vidéo, la robotique, la surveillance, la météorologie, l'océanographie, les applications médicales, le militaire...

Le mouvement des objets dans une séquence vidéo est étudié, en recherchant la corrélation entre deux images consécutives afin de prédire l'évolution de la position du contenu. Ce processus, appelé estimation de mouvement, est présent dans de nombreuses normes vidéo, notamment MPEG-1, MPEG-2, MPEG-4, H.261, H.263,.., ainsi que dans de nombreux codecs vidéo.

Il existe de nombreuses méthodes d'estimation de mouvement, les plus connues étant les méthodes de mise en correspondance par bloc ‘’ Block-Matching Algorithme ‘’ grâce à sa mise en œuvre simple et efficace. Le principe général de ces méthodes est que chaque deux images successives dans la séquence vidéo (courante et référence) sont divisées en blocs. Ensuite, pour chaque bloc dans l’image cible, l’algorithme cherche le bloc le plus similaire dans l’image de référence en minimisant une erreur de dis-similarité entre les illuminations correspondantes aux pixels respectifs des deux blocs. Le vecteur de mouvement est la différence entre les positions du bloc courant et du bloc le plus similaire dans l’image de référence. [BET19]

Mais dans cette méthode (BMA) il existe deux problèmes essentiel doivent être pris en considération [BAI06] :

a) Un problème de la recherche du minimum global (problème d’optimisation).
b) Un problème de temps de calcul

Dans les applications pratiques d’estimation du mouvement, nous cherchons un compromis entre précision et efficacité : la méthode doit fournir un résultat suffisamment précis dans un temps suffisamment court pour être utilisable. [BZR94], ([NAG83]

L’algorithme "Full Searche" aboutit à la solution optimale par une recherche exhaustive dans la zone de recherche du frame précédent [BAI06]

Nous présentons dans ce mémoire, les algorithmes de ‘Block Matching’ avec test de bloc de différence. Pour élaborer ce travail, nous avons organisé notre mémoire comme suite :

Dans le premier chapitre, nous présentons des notions concernant l'image numérique, la compression vidéo, et le rôle de l'estimation de mouvement dans les normes de compression vidéo

Dans le deuxième chapitre, nous présentons les différents concepts, les différentes méthodes de l'estimation de mouvement

Dans la troisième chapitre, nous allons discuter la méthode de compensation de mouvement la plus utilisée est la compensation de mouvement par correspondance de bloc (Block Matching), les algorithmes les plus utilisée dans cette méthode (FS, TSS, NTSS, FSS, DS, et MIN)

Dans le quatrième chapitre, s'intéresse plus particulièrement aux résultats expérimentaux pour les algorithmes de 'Block Matching' avec test de bloc de différence.

Et à la fin une conclusion générale de notre thème.

CHAPITRE I :
Notion générales sur l'image

I.1. Introduction

Avec la parole, l'image constitue l'un des moyens les plus importants qu'utilise l'homme pour communiquer avec autrui. C'est un moyen de communication universel dont la richesse contenu permet aux être humains de tout âge et de toute culture de se comprendre.

C'est aussi le moyen le plus efficace pour communiquer, chacun peut analyser l'image à sa manière, pour en dégager une impression et d'un extraire des informations précises.

De ce fait, le traitement d'image est l'ensemble des méthodes et techniques opérant sur celles-ci, dans le but de rendre cette opération possible, plus simple, plus efficace et plus agréable, d'améliorer l'aspect visuel de l'image et d'en extraire des informations jugées pertinentes. [ALB13]

I.2. Notion de base sur l'image

I.2.1. Définition de l'image

Une image est une représentation planaire d'une scène ou d'un objet situé en général dans un espace tridimensionnel. Son élaboration résulte de la volonté de proposer une entité observable par l'œil humain. Ceci explique d'une part son aspect planaire et d'autre part le fait que l'information élémentaire associé à chaque point de l'image soit transcrite en niveau de gris ou en couleur.

Cette information élémentaire est, en général, une grandeur physique liée à la nature du capteur. Une caméra CCD mesure la quantité de lumière émise par l'objet, un récepteur de rayons x mesure la quantité de rayonnement transmise par l'organe (en imagerie médicale).

L'image est donc une collection d'informations qui, dans un premier temps, se présentait sur un support photographique qui permettait le traitement en temps différé d'un phénomène fugace.

I.2.2. Image Analogique

Les images analogiques sont obtenues en utilisant divers instruments qui les projettent telles qu'elles sont dans la nature sur des supports différents (papier, film,...). Parmi ces instruments, on peut citer : l'appareil photo, la caméra,...

I.2.3. Image numérique

Le terme d'image numérique désigne, toute image sauvegardée sous une forme codée représentable par des valeurs numériques. La numérisation est le processus qui permet de passer de l'état d'image analogique, qui est caractérisée par l'aspect continu du signal qu'elle représente, à l'état d'image numérique qui est caractérisée par l'aspect discret. Le mot discret signifie que l'intensité lumineuse ne peut prendre que des valeurs quantifiées en un nombre fini de points distincts. Après numérisation, l'image numérique représentée par un ensembled'échantillons à deux dimensions appelé pixels. L'ensemble de l'image peut être considéré comme un tableau

bidimensionnel ou une matrice de valeurs de pixels. Nous appelons ce tableau (matrice) un bitmap.

I.2.4. Caractéristiques de l'image numérique

I.2.4.1. Pixel

C'est l'élément d'image ou «picture-element », le pixel est le plus petit point de l'image numérique (figure.I.1), c'est une entité calculable qui peut être quantifié.

Un pixel est aussi le plus petit élément que peuvent manipuler les matériels et logiciels d'affichage ou d'impression. L'ensemble des pixels est contenu dans un tableau à deux dimensions constituant l'image.

- Dans le cas d'une image monochrome chaque pixel est codé sur un octet.
- Dans une image en couleur ou à base de RVB (rouge, vert, bleu), un pixel est représenté sur trois octets (un octet pour chaque couleur)

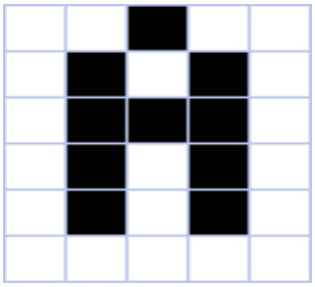


Figure I.1. *Le pixel*

I.2.4.2. Dimension

C'est la taille de l'image, qui n'est rien qu'une matrice de pixels. La taille est obtenue en multipliant le nombre de ligne par le nombre colonne. Une image possédant 640 pixels en largeur et 480 en hauteur aura une définition (dimension) de 640 pixels par 480, notée 640x480.

I.2.4.3. Taille d'une image [BAI06]

Pour connaître la taille (en octets) d'une image, il est nécessaire de compter le nombre de pixels que contient l'image, cela revient à calculer le nombre de cases du tableau, soit la hauteur de celui-ci que multiplie sa largeur. La taille (ou poids) de l'image est alors le nombre de pixels que multiplie la taille (en octets) de chacun de ces éléments ;
Voici le calcul pour une image 640x480 en True color :
Nombre de pixels : 640 x 480 = 307200

24 bits / 8 = 3 octets

Le poids de l'image est ainsi égal à : 307200 x 3 = 921600 octets

921600 / 1024 = 900 Ko

Voici quelques exemples (en considérant que l'image n'est pas compressée) :

Tableau I.1. *Tableau comparatif entre tailles des images*

Définition de l'image	**Noir et blanc (1bit)**	**256 couleurs (8 bits)**	**65000 couleurs (16 bits)**	**True color (24 bits)**
320*200	7.8 Ko	62.5Ko	125 Ko	187.5 Ko
640*480	37.5 Ko	300 Ko	600 Ko	900 Ko
800*600	58.6 Ko	468.7 Ko	937.5 Ko	1.4 Mo
1024*768	*96 Ko*	768 Ko	1.5 Mo	2.3 Mo

Cet met en évidence la qualité de la mémoire vidéo que nécessite votre carte graphique en fonction de la résolution de l'écran (nombre de points affichés) et du nombre de couleurs. L'exemple qu'il faut une carte ayant 4Mo de mémoire vidéo afin d'afficher une résolution de 1024x768 en true color.

I.2.4.4. Résolution

C'est la clarté ou la finesse de détails atteinte par un moniteur ou une imprimante dans la production d'une image. La résolution est exprimée en unité de mesure (pouce ou centimètre) par nombre de pixels. [MIC97]

I.2.4.5. Histogramme

C'est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (Couleur) dans l'image (figure.I.2). Il peut être utile dans les cas suivant :

- Pour diminuer l'erreur de quantification ;
- Pour comparer deux images obtenues sous des éclairages différents ;
- Pour amélioré certaine propriété d'une image (rehaussement) pour en tiré des informations utile.



Figure I.2. *Histogramme des niveaux de gris*

I.2.4.6. Luminance

C'est la variation d'intensité lumineuse ou le degré de luminosité des points de l'image, une bonne luminance se caractérise par :

- Des images brillantes
- Un bon contraste
- L'absence de parasites

I.2.4.7. Contraste

C'est l'opposition marqué entre deux régions juxtaposées d'une image, plus précisément les régions claires et les régions sombres de cette image, il et défini en fonction des luminances $\mathbf{L_i}$ et $\mathbf{L_{i+1}}$ de deux zone de l'image.

$$C = \frac{L_i - L_{i+1}}{L_i + L_{i+1}}$$

I.2.4.8. Bruit

Un bruit (parasite) dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par rapport à ses voisins, il provient de l'éclairage des dispositifs électroniques et optique du capteur [RCG77].

I.2.5. Les type des images et des graphes

En fonction du nombre de bits utilisés pour la représentation en pixels, une image numérique peut être caractérisée par :

- Image binaire : chaque pixel est représenté en utilisant un bit.
- Infographie : quatre bits par pixel sont utilisés.
- Image en niveaux de gris: huit bits par pixel sont utilisés.
- Image couleur : chaque pixel est représenté en utilisant 24 ou 32 bits.

I.2.5.1. Image binaire

Une image à un bit se compose uniquement de bits on et off. Il s'agit donc du type d'image le plus simple. Chaque pixel est stocké sous forme de bit unique (0 ou 1). Par conséquent, une telle image est également appelée image binaire. On l'appelle aussi parfois une image monochrome à 1 bit, car elle ne contient aucune couleur. La figure I.3 montre une image monochrome à 1 bit (appelée «Lena» par des scientifiques en multimédia). Il s'agit d'une image standard utilisée pour illustrer de nombreux algorithmes.

Une image monochrome 640 × 480 nécessite 38.4 kilo-octets (Ko) de stockage (= 640 × 480/8). Les images monochromes à 1 bit peuvent être satisfaisantes pour des images contenantuniquement des graphiques et du texte simples. De plus, les télécopieurs utilisent des données 1 bit. Par conséquent, les images 1 bit sont toujours importantes même si les capacités de stockage ont suffisamment augmenté pour permettre l'utilisation d'une imagerie contenant davantage d'informations.



Figure I.3. *Image binaire. (Lena)*

I .2.5.2. Images à niveaux des gris

Une image à niveau des gris n bits, est une image pour laquelle chaque pixel a une valeur de gris comprise entre 0 et $N=2^n-1$. Couramment pour les images à niveaux de gris, chaque pixel est représenté par un seul octet (n=8 N=255). L'un des paramètres de l'image est le rapport entre le nombre de ligne et le nombre de colonnes.Par exemple, pour une image 640 × 480 le rapport est 4:3.

La figure I.4 donne un exemple d'une image en niveaux de gris appelée « Baboon » d'une résolution 512*512.

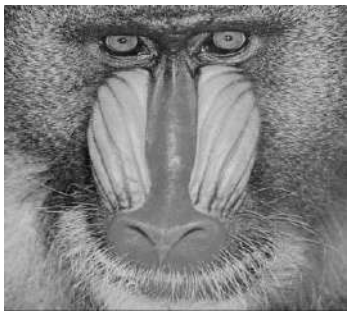


Figure I.4. *Image à niveaux de gris de résolution 512*512.*

I.2.2.5. Image couleur 24 bits

La couleur est l'une des caractéristiques les plus importantes de l'image. L'image couleur peut être modélisée à l'aide de divers systèmes de couleur. Le RVB « RGB » (rouge, vert bleu) est l'un des systèmes de couleurs couramment utilisés. Le modèle RVB est basé sur le fait que la couleur peut être vue comme une fonction vectorielle de trois coordonnées pour chaque position dans l'image. Parfois, ce modèle est appelé modèle additif, car l'image est obtenue en ajoutant les composants en couleurs primaires. Chaque point de l'image peut être représenté par la somme des valeurs des trois couleurs primaires (R, G, B) selon la représentation de la figure I.5. Le modèle

RVB soit basé sur la perception humaine des couleurs, il est utilisé donc pour afficher des images (moniteurs, téléviseurs, etc.).

Dans le modèle RVB, la valeur 0 (coordonnée 0) signifie l'absence de couleur, tandis que la valeur 255 (coordonnée 1) indique la couleur avec une intensité maximale. Ainsi, nous concluons que (0, 0, 0) représente le noir et (1, 1, 1) représente le blanc.

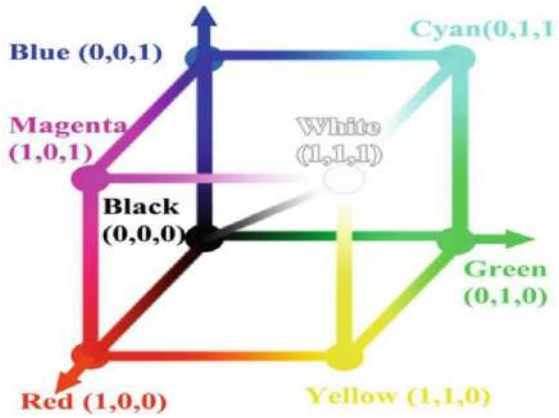


Figure I.5. *La somme des valeurs des trois couleurs primaires (R, G .B).*

En combinant deux des trois couleurs primaires (R, G, B), nous obtenons les couleursutilisées dans le modèle de couleur CMY

Le niveau de gris est défini par la ligne R=G=B.

$$G + B = C(cyan);$$

$$R + B = M(magenta);$$
$$R + G = Y(yellow);$$
$$R + G + B = W(white)$$

Dans une image couleur de 24 bits, chaque pixel représentant généralement le format RVB, comme chaque valeur est comprise entre 0 et 255, ce format prend en charge, soit un totale de 16777216 couleurs combinées possibles. Cependant, une telle flexibilité entraine une pénalité de stockage : une image de couleur 24 bits 640*480 nécessiterait 921,6 Ko de stockage sans compression.

I.2.6. Le codage de la couleur

Une image est donc représentée par un tableau à deux dimensions dont chaque case est un pixel. Pour représenter informatiquement une image, il suffit donc de créer un tableau de pixels dont chaque case contient une valeur. La valeur stockée dans une case est codée sur un certain nombre de bits déterminant la couleur ou l'intensité du pixel, on l'appelle profondeur de codage. Il existe plusieurs standards de codage de la profondeur :

- **bitmap noir et blanc**: en stockant un bit dans chaque case, il est possible de définir deux

couleurs (noir ou blanc).

- **bitmap 16 couleurs ou 16 niveaux de gris**: en stockant 4 bits dans chaque case, il est possible de définir 2^4 possibilités d'intensités pour chaque pixel, c'est-à-dire 16 dégradés de gris allant du noir au blanc ou bien 16 couleurs différentes.
- **bitmap 256 couleurs ou 256 niveaux de gris**: en stockant un octet dans chaque case, il est possible de définir 2^8 intensités de pixels, c'est-à-dire 256 dégradés de gris allant du noir au blanc ou bien 256 couleurs différentes.
- **palette de couleurs (colormap)**: grâce à cette méthode, on peut définir une palette contenant l'ensemble des couleurs pouvant être contenues dans l'image. A chacune des ces couleurs est associée un indice. Ainsi en codant ces indices sur 8 bits il est possible de définir 256 couleurs utilisables, c'est-à-dire que chaque case du tableau à deux dimensions représentant l'image va contenir un nombre indiquant l'indice de la couleurà utiliser.
- **true color**: cette représentation permet de représenter une image en définissant chacune des composantes (RGB: rouge, vert, bleu). Chaque pixel est représenté par unentier comportant les trois composantes, chacune codée sur un octet, c'est-à-dire au total 24 bits (16 millions de couleurs). Il est possible de rajouter une quatrièmecomposante permettant d'ajouter une information de transparence ou de texture, chaque pixel est alors codé sur 32 bits.

I.2.7. Formats d'images

I.2.7.1. Format GIF (Graphics Interchange)

Le format GIF a été conçu initialement pour la transmission d'images graphiques sur des lignes téléphoniques via des modems. La norme GIF utilise l'algorithme de Lempel-Ziv-Welch (LZW)avec de légère modification. La norme GIF utilse 8 bits de couleur (256 couleurs). Cette normeconvient mieux aux images contenant peu de couleurs distinctives (graphiques ou dessins, par exemple). Le format GIF est disponible en deux versions. La spécification d'origine est GIF87a. La version ultérieure, GIF89a, prend en charge l'animation simple.

I.2.7.2. Format JPEG (Joint Photographic Experts Group)

Le standard actuel le plus important pour la compression d'images est le format JPEG. Le système de vision humaine a certaines limitations spécifiques, dont JPEG tire parti pour atteindre des taux de compression élevés. Le système œil-cerveau ne peut pas voir les détails extrêmement fins. Si de nombreux changements se produisent dans quelques pixels, nous nous référons à ce segment d'image comme ayant une fréquence spatiale élevée, c'est-à-direbeaucoup de changements dans l'espace (x, y). Cette limitation est encore plus visible pour la vision des couleurs que pour les niveaux de gris (noir et blanc). Par conséquent, les informationsde couleur dans JPEG sont décimées (partiellement supprimées ou moyennées), puis de petits blocs d'une image sont représentés dans le domaine de fréquence spatiale (u, v), plutôt que dans (x, y). C'est-à-dire que la vitesse des changements de x et de y est évaluée, de bas en haut, et une nouvelle "image" est formée en groupant les coefficients ou les poids de ces vitesses. Les poids qui correspondent aux changements lents sont alors privilégiés.

Étant donné que nous éliminons efficacement beaucoup d'informations par les étapes de

division et de troncature, la compression est "avec perte" (bien qu'un mode sans perte existe). De plus, JPEG permet à l'utilisateur de définir le niveau de qualité souhaité, ou le taux de compression.

I.2.7.3. Format PNG (Portable Network Graphics)

Un développement intéressant découlant de la popularité d'Internet concerne les efforts visant à créer des formats d'image plus indépendants du système. Un tel format est appelé Portable Network Graphics (PNG). Les caractéristiques spéciales des fichiers PNG incluent la prise en charge d'un maximum de 16 bits par pixel dans chaque canal de couleur, c'est-à-dire une augmentation importante de la couleur sur 48 bits. Les fichiers peuvent également contenir des informations de correction gamma pour un affichage correct des images en couleur et des informations sur le canal α (jusqu'à 16 bits) pour des utilisations telles que le contrôle de la transparence. Le format PNG prend en charge la compression sans perte et avec perte avec uneperformance supérieure au GIF. Le format PNG est largement pris en charge par divers navigateurs Web et logiciels d'imagerie.

I.2.7.4. Format TIFF (Tagged Image File Format)

Le format TIFF est un autre format de fichier d'image populaire pris en charge par Microsoft. Sa prise en charge de la saisie d'informations supplémentaires (appelées «balises») offre une grande souplesse. La balise la plus importante est un signifiant de format: quel type de compression, etc., est utilisé dans l'image stockée. Par exemple, le format TIFF peut stocker de nombreux types d'images différents: 1 bit, niveaux de gris, 8 bits, RVB 24 bits, etc. Le format TIFF était à l'origine un format sans perte, mais une balise supplémentaire permet d'opter pour des compressions JPEG, JBIG et même JPEG-2000. Il est assez courant d'utiliser des fichiers TIFF pour stocker des données non compressées. Les fichiers TIFF sont divisés en sections, chacune d'entre elles pouvant stocker une image bitmap, une image vectorielle ou d'autres types de données. Le type de données de chaque section est spécifié dans sa balise.

I.2.7.5. Format Windows BMP

BitMap (BMP) est l'un des principaux formats de fichier image standard du système pour Microsoft Windows. Le format BMP prend en charge de nombreux formats de pixels notamment les couleurs indexées (jusqu'à 8 bits par pixel) et les images couleur de 16, 24 et 32bits. Il utilise la compression RLE (Run-Length Encoding) et peut compresser assez efficacement des images couleur 24 bits grâce à son algorithme RLE 24 bits. Les images BMPpeuvent également être stockées non compressées. En particulier, les images couleur 16 bits et32 bits (avec les informations du canal α) sont toujours non compressées.

I.2.7.6. Format EXIF (Exchangeable Image File)

L'EXIF est un format d'image destiné aux appareils photo numériques. Il permet l'enregistrement de métadonnées d'image (exposition, source de lumière / flash, balance des blancs, type de scène, etc.) pour la normalisation de l'échange d'images. Une grande variété d'étiquettes (beaucoup plus que dans le format TIFF) est disponible pour faciliter une impression de meilleure

qualité, car des informations sur l'appareil photo et les conditions de prise de vue peuvent être stockées et utilisées, par exemple par des imprimantes, en vue d'éventuels algorithmes de correction des couleurs. Le format EXIF est intégré au logiciel JPEG dans la plupart des appareils photo numériques. Il inclut également la spécification du format de fichier audio qui accompagne les images numériques.

I.3. Concepts sur la vidéo

I.3.1. Définition de la vidéo [ALB13]

La vidéo est une succession d'images animées. Le principe fondamental de la vidéo est que l'œil humain a la possibilité de retenir pendant un certain temps (de l'ordre du deuxième de seconde) toute image imprimée sur la rétine. Il suffit donc de faire défiler un nombre suffisant d'image par seconde, pour que l'œil ne se rende pas compte qu'il s'agit d'images distinctes.

I.3.2. Compression vidéo [KER15]

Nous allons maintenant donner un aperçu général du codage vidéo qui permet d'exploiter ces redondances. Nous décrirons pour cela d'abord les différents types de redondance, la manière de décomposer la vidéo en groupe d'images, nous rappellerons un des critères d'évaluations utilisés pour comparer ces techniques, puis nous donnerons un bref rappel historique de normes de compression et nous finissons par le schéma général d'un codeur vidéo. Le système de compression utilise deux modules complémentaires : le module de compression dit codeur qui consiste à convertir la source de données avant transmission ou stockage en une forme compressée-occupant un nombre de bits moindre, et un décompresseur (décodeur) qui convertit la forme compressée en une représentation de la source originale. La paire codeur/décodeur est appelée codec. La compression vidéo consiste à réduire la quantité de données nécessaires pour la représenter en exploitant les redondances d'informations, tout en cherchant que cela soit invisible par le système de vision humain. Cela produit à un compromis entre le taux de compression et la qualité de l'image compressée. L'image devient plus médiocre avec l'augmentation du taux de compression.

I.3.2.1. Contraintes

Une vidéo représente une quantité de données importante, et malgré l'augmentation de la puissance des processeurs et des capacités des périphériques de stockage, nous avons besoin de la représenter dans un format plus concis. La qualité d'une vidéo compressée varie très grandement en fonction de la fréquence d'entrée, de la résolution d'entrée, du débit imposé en sortie et de la qualité exigée. La compression des données vidéo numériques sans dégradation significative de la qualité est possible lorsque les séquences vidéo affichent un degré élevé de:

- Redondance spatiale : corrélation entre les pixels voisins.
- Redondance temporelle : corrélation entre les images vidéo.
- Redondance psycho-visuelle : propriétés de la vue humaine.

I.3.2.2. La compression de données

I.3.2.2.1. La redondance spatial

Lorsqu'on observe une seule image d'une vidéo, nous pouvons constater qu'elle constitue des zones homogènes plus ou moins grandes dans lesquelles les pixels ont des valeurs très proches, voire identiques : c'est la redondance spatiale (figure I.6).

En l'éliminant par codage, il est possible de réduire la quantité d'information à transmettre.

La compression des images fixes (norme JPEG Joint Photographic Experts Groupe), utilise ce type de redondance.



Figure I.6. *La redondance spatiale : la redondance de la couleur noire est très prononcée.*

I.3.2.2.2. La redondance temporelle

Dans une séquence vidéo, la différence entre une image et la suivante est relativement faible, sauf lors d'un changement de plan. Autrement dit, la position d'un bloc de pixels varie généralement peu d'une image à l'autre.

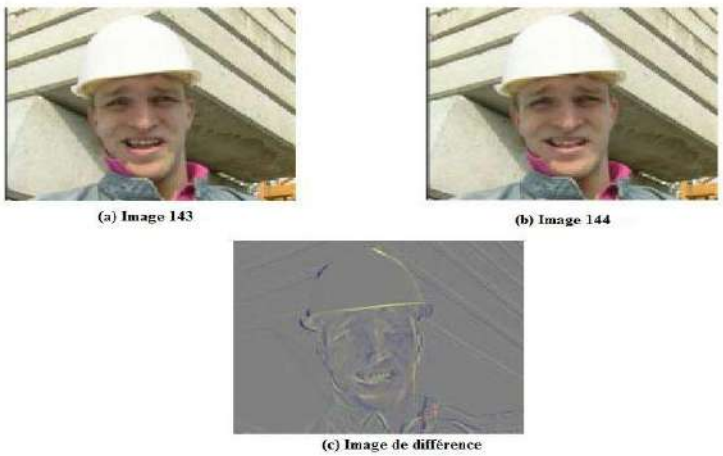


Figure I.7. *La redondance temporelle : la scène avec un fond fixe seul le visage de la personne qui bouge.*

La seule différence se trouve dans les parties qui ont subi un mouvement(voir figure 1.7). Ceci peut être démontré par la figure1.7(c) qui représente la différence entre les deux images successives (figure 1.7(a) et figure 1.7 (b)). La partie grise de cette image correspond à la similitude des deux images (a) et (b). La compression consiste à éliminer les redondances qu'elle soit spatiale ou temporelle.

I.3.2.2.3. La redondance psycho visuelle

La redondance psycho-visuelle est généralement réduite. La mesure la plus significative est une résolution réduite du détail des couleurs par rapport au détail de la luminance qui permet un meilleur rapprochement avec les caractéristiques de la perception humaine. Les images vidéo comprennent trois matrices rectangulaires de données de pixels, lesquelles représentent le signal de luminance (luma Y) et deux signaux de chrominance (chroma Cb et Cr aussi appelé U et V) comme le montre la figure I.8

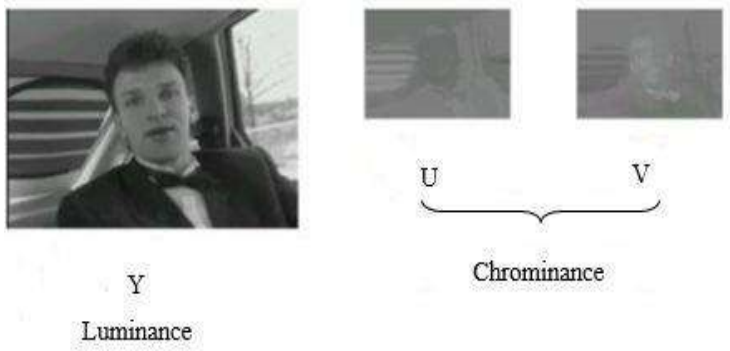


Figure I.8. *Format YUV 4 :1 :1.*

Ces matrices correspondent à une représentation décomposée des trois couleurs primaires (RVB) associées à chaque élément d'image. Pour la représentation des pixels vidéo numérique, il existe plusieurs formats comme le montre la figure I.9.

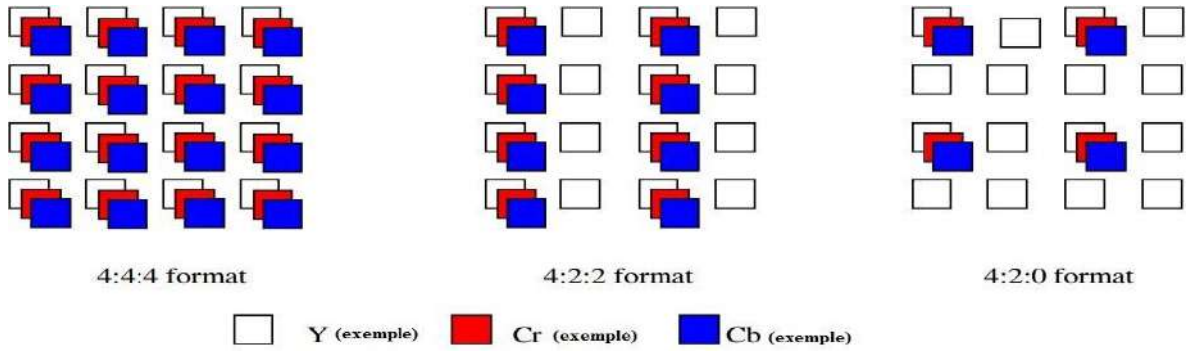


Figure I.9. *Format YUV.*

I.3.3. Décomposition de la séquence vidéo compressée [KER15]

Une séquence vidéo compressée est généralement structurée en groupe d'images GOP (Groupe Of Pictures). Un GOP est une suite d'images codées suivant trois modes de codage (voir figure I.10) :

- Les images Intra, dites de type I : sont des images codées intégralement, sans aucune référence aux images voisines de la séquence vidéo. Les images I sont les plus coûteuses en débit mais servent de point de référence dans une séquence vidéo. Chaque changement de plan dans une séquence vidéo commence obligatoirement par une image de type I.
- Les images Inter prédites, dites de type P : Les images « Prédictives » exploitent à la fois la redondance spatiale et la redondance temporelle des images d'une séquence vidéo. Elles sont codées à partir de l'image « I » ou « P » précédente à l'aide de vecteurs de mouvement obtenu par estimation du mouvement.
- Les images Inter prédites bidirectionnelles, dites de type B : Elles sont codées avec une estimation du mouvement par rapport à une image précédente et une image suivante. Elles ne servent jamais de référence. Les images de type I et P servent de référence aux images P et B, les images de type B ne servent jamais de référence.

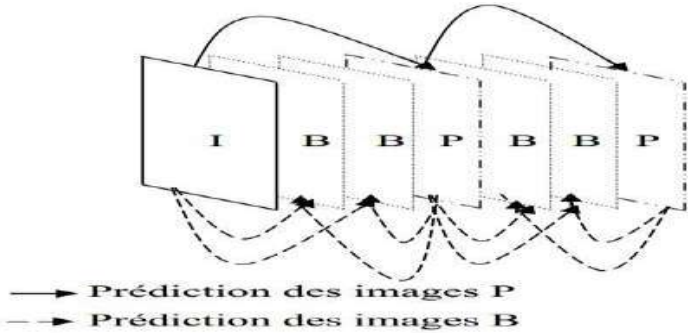


Figure I.10. *Structure d'un groupe d'image (GOP).*

I.3.4. Les méthodes de compression vidéo [KER15]

Le principe fondamental de la compression vidéo est de réduire autant que possible les redondances d'informations dans les données, sans bien sûr que cela ne soit visible de manière flagrante pour l'œil humain. Toute la difficulté est là, dans le dosage entre un taux de compression qui s'améliore en même temps que la qualité d'image devient médiocre.

I.3.4.1. Compression avec perte

Dans le cas de la compression avec perte, les données sont différentes à la sortie du décodeur par rapport à l'entrée du codeur. C'est ce type de compression qui est utilisé en vidéo, car on peut accepter des pertes d'informations qui ne sont pas toujours visibles à l'œil et qui se traduisent par de nets gains de compression. En compression avec pertes, on peut atteindre des taux de compression allant jusqu'à 300 :1 (c'est-à-dire qu'un vidéo de taille 300 Mo, peut attend 1 Mo après la compression).

I.3.4.2. Compression sans perte

Dans le cas de la compression sans perte, les données décodées à l'arrivée par le récepteur sont strictement identiques aux données codées au départ par l'émetteur. Ce type de compression, permettant au mieux un taux de compression de 2 :1, est évidemment insuffisant pour la compression vidéo et ne sera pas l'objet d'étude de cette thèse.

I.3.5. Codec vidéo

I.3.5.1. Principe

Un codec vidéo code une source en une forme compressée et la décode pour produire une copie ou une approximation de la source. La séquence originale est représentée par une modélisation, c'est-à-dire une représentation codée le plus efficacement possible et utilisable pour la reconstruction (figure I.11). Idéalement, la modélisation représente la séquence avec aussi peu de bits que possible et une qualité aussi bonne que possible. Ces deux buts à atteindre sont évidemment conflictuels. En effet, un débit faible entraîne une image de qualité réduite. Tous les codecs fonctionnent selon le même principe [KER15].

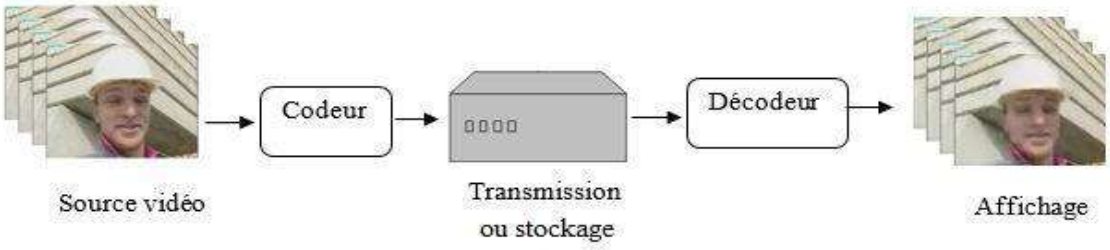


Figure I.11. *Codeur/décodeur.*

I.3.5.2. Standards de compression vidéo

Deux grandes normes sont dans le marché du codage vidéo : la norme H26x et la norme MPEGx (Moving Picture Experts Group). La première développée par ITU−T (International Telecommunication Union−Telecommunication) et la deuxième par ISO/CEI(International Organization for Standardization/International Electrotechnical Commission) pour le traitement et le codage audio/vidéo, notamment les familles MPEG et H.26x. Ce sont les objectifs du codage qui déterminent la structure et les principes de l'encodeur. La norme H26x est destinée initialement à la vidéo conférence qui est caractérisée par des mouvements lents et des arrières plans statiques, tandis que la norme MPEGx est destinée aux applications de la vidéo numérique et de l'informatique ainsi qu'aux applications de la télévision numérique [KER15].

Ci-après un extrait historique de ces normes :

- MPEG-1 : La norme de compression de l'audio/vidéo numérique MPEG-1 a été approuvée

par le groupe MPEG en novembre 1991 pour le codage des images animées et du son associé pour les supports de stockage numérique pour un débit allant jusqu'à environ 1,5 Mbit/s. Les supports de stockage numériques courants comprennent les disques compacts (CD) et les disques VCD (video compact discs). La norme MPEG-1 est également appelée ISO/IEC 11172

- MPEG-2 : Le développement de la norme MPEG-2 a débuté en 1990. Contrairement à la norme MPEG-1, qui est essentiellement une norme permettant de stocker et de lire des vidéos sur le CD d'un seul ordinateur à faible débit (1,5 Mbps), MPEG-2 est une norme pour les vidéos de meilleure qualité avec un débit supérieur à 4 Mbps. La norme MPEG-2 a été initialement développée pour la télévision numérique. La norme MPEG-2 est également appelée ISO/CEI 13818, elle a été approuvée par le Groupe d'experts en image animée ISO/CEI en novembre 1994. Semblable à la norme MPEG-1
- MPEG-3 : a été conçu pour la prise en charge des signaux (HDTV) (La télévision à haute définition) à des débits de 20 à 40 Mbits/s. Il a vite été observé que des résultats semblables pouvaient être obtenus par de légères modifications de MPEG−2. Perdant de son intérêt, MPEG-3 a été abandonné [KER15]
- MPEG-4 : La première version de MPEG-4 a été finalisée en octobre 1998 et est devenue une norme internationale au début de 1999. La norme MPEG-4 est aussi nommée ISO / IEC 14496.
- MPEG-7 : La norme MPEG-7 est devenue une norme internationale en septembre 2001. Son nom officiel est « Multimedia Content Description Interface » (interface de description de contenu multimédia), connue sous le nom ISO/IEC 15938.
- H.261 : est une encienne norme de compression vidéo numérique. Mais sont principe de compréssion qui est basée sur le compensation de mouvement est très largement retenu dans les normes de compression vidéo ultérieures récentes. Le Comité consultatif international télégraphique et téléphonique (CCITT : International Telegraph and Telephone Consultative Committee) a entamé le développement de H.261 en 1988. La recommandation finale a été adoptée par l'UIT-T (International Telecommunication Union— Telecommunication standardization sector) en 1990. La norme a été conçue pour le visiophone, la vidéoconférence et d'autres services audiovisuels sur les lignes téléphoniques RNIS : Réseau numérique à intégration de services (ISDN : Integrated Services Digital Network). Initialement, la norme H261 était prévu de prendre en charge des multiples (de 1 à 5) de canaux de 384 kbps (kilobits par seconde). Au final, le codec vidéo prend en charge les débits binaires de p × 64 kbps, où p est compris entre 1 à 30. Par conséquent, la norme était connue sous le nom de p * 64, notée «p star 64».
- H.263 : est une norme de codage vidéo améliorée pour la vidéoconférence et les autres services audiovisuels transmis sur les réseaux téléphoniques commutés publics RTCP (PSTN Public Switched Telephone Network). Il vise des communications à faible débit binaire à des débits inférieurs à 64 kbps. Il a été adopté par les 15 ième commissions d'étude de l'UIT-T en 1995. La deuxième version de H.263, également connue sous le nom de H.263 +, a été approuvée en janvier 1998 par la 16 ième Commission d'études de l'UIT-T. Elle est entièrement rétrocompatible avec la conception de la version H.263. Le but de H.263+ est d'élargir les applications potentielles et d'offrir une flexibilité supplémentaire. H.263 + comprend de nombreuses recommandations pour améliorer l'efficacité du code et la résilience aux erreurs. Puisque le développement de H263+ était après la standardisation de

MPEG-1 et 2, il n'est pas surprenant qu'elle adopte également de nombreux aspects des normes MPEG. La troisième version de la norme H263, connue sous le nom de H.263 ++, a été approuvée pour la première fois en 2000. Une version plus élaborée a été approuvée en 2005. La norme H.263 ++ inclut les méthodes de codage de base de H.263 en plus de recommandations supplémentaires.

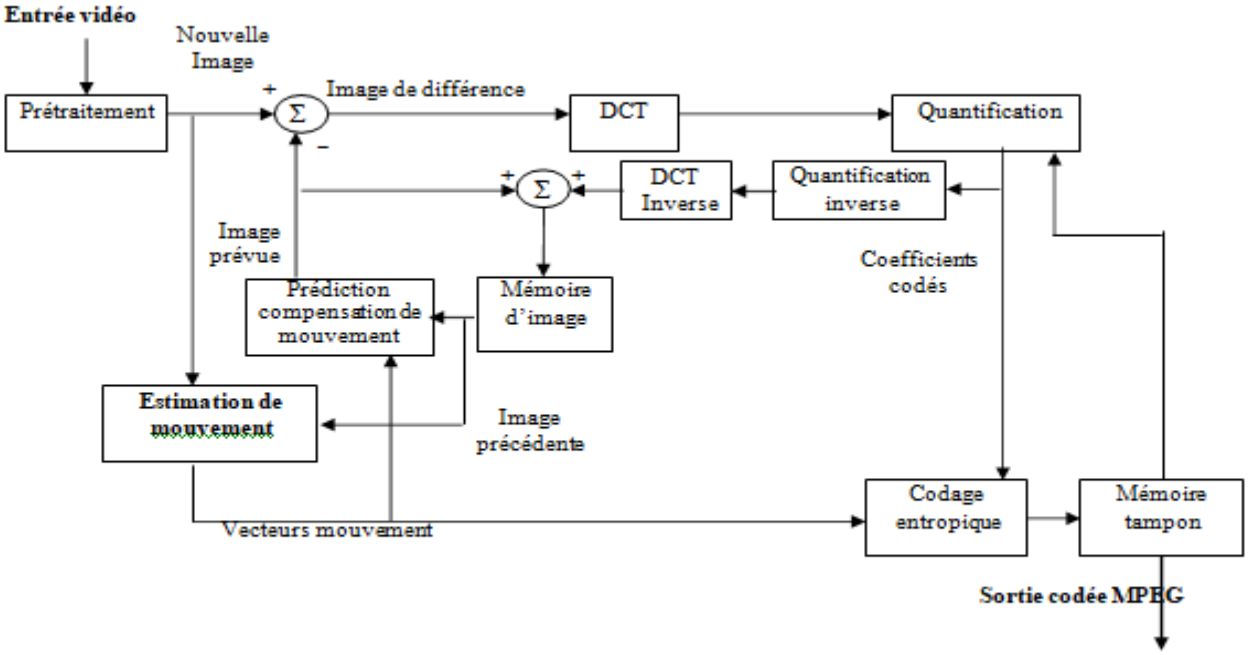


Figure I.12. *Synoptique d'un compresseur MPEG.*

I.3.7. Estimation de mouvement

L'estimation du mouvement est l'un des domaines du traitement d'image qui joue le rôle dans l'analyse des séquences d'images. Aussi c'est un procédé qui consiste à étudier le déplacement des objets dans la séquence vidéo. Le principe de l'estimation du mouvement consiste à construire une image de prédiction à partir d'une image précédente. L'estimation du mouvement n'est pas effectué sur chaque point de l'image mais sur des macros blocs, afin de réduire au minimum la richesse d'information à transmettre [ALL21].

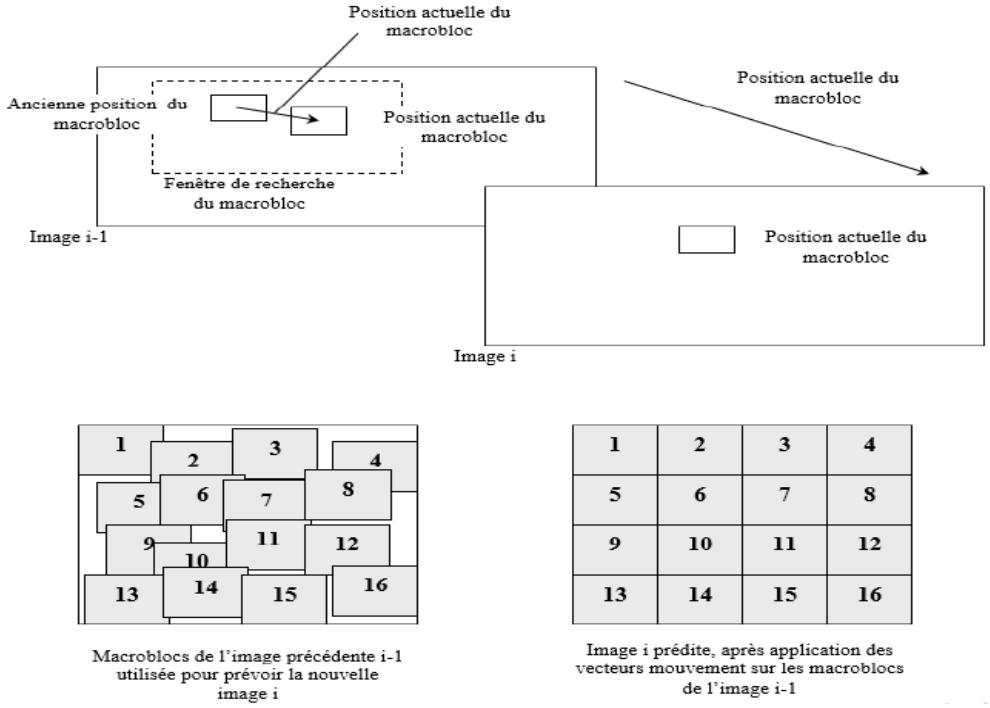


Figure I.13. *Détection et estimation de mouvement en MPEG.*

I.4. Conclusion

Dans tous les standards de la compression vidéo numérique, En remarque que l'estimation de mouvement joue un rôle très important, Il existe plusieurs techniques pour estimer le mouvement.

Dans le chapitre suivant, nous expliquerons comment estimer le mouvement.

CHAPITRE II :
Estimation de mouvement dans l'image vidéo

II.1. Introduction

Dans une scène l'estimation du mouvement des objets contenus est une primitive importante dans l'analyse d'une séquence vidéo, Cette étape d'estimation dépend du modèle choisi pour calculer l'évolution spatiale des objets. L'obtention de cette primalité repose sur l'observation de plusieurs trames consécutives, et cette observation contient des objets qui ont changé de position relative. L'évolution temporelle générée est appelée génériquement : mouvement. Dans ce chapitre, nous expliquerons le principe de l'EM, ensuite, nous exposerons les grandes catégories des techniques d'EM.

II.2. Le principe de l'estimation de mouvement

Le principe de l'estimation de mouvement consiste à construire une image de prédiction à partir d'une image précédente et d'informations relatives aux déplacements subis par ses composantes. L'estimation de mouvement est effectuée sur des macros blocs, afin de réduire au minimum la richesse d'informations à transmettre. Si, entre une image actuelle et celle qui la précède, deux macro blocs semblables sont repérés, mais à des emplacements légèrement différents; il suffit de transmettre une seule fois ce macro bloc et d'indiquer quelle est sa nouvelle position sur l'image actuelle. Cette information de position - amplitude et direction - est donnée, par un vecteur mouvement

L'estimation de mouvement se décompose en cinq étapes comme suit :

- Recherche des macros blocs semblables entre la nouvelle image et l'image précédente.
- Calculer des vecteurs mouvements caractérisant le déplacement des macros blocs.
- Construction d'une image prédite en utilisant ces vecteurs du mouvement.
- Comparaison de cette image prédite avec la vraie nouvelle image pour générer des données d'erreurs de prédiction.
- Codage et transmission des vecteurs et des données d'erreurs de prédiction.

Dans la suite on s'intéressera à l'estimation du champ de vecteur vitesse et vecteur déplacement.

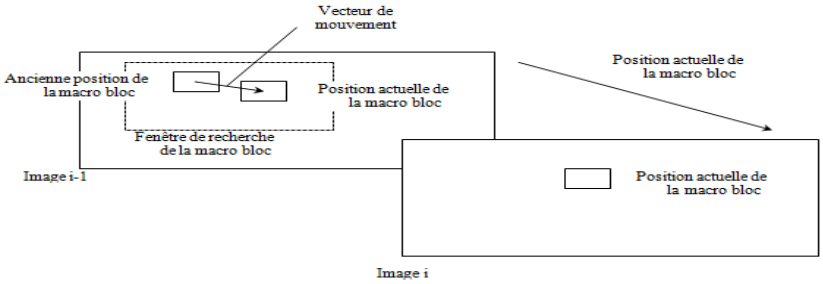


Figure II.1. *Utilisation de technique d'estimation de mouvement*

II.3. Domaines d'applications [BET19]

L'estimation de mouvement est une étape essentielle pour plusieurs applications dans le domaine de la vision par ordinateur. En compression de vidéo, l'EM exploite la redondance temporelle entre les images pour un gain de compression significatif ; cela est possible en représentant l'image *t* à partir de l'image *t*-1 de la séquence en transmettant uniquement les vecteurs de mouvement des blocs et l'erreur résiduelle entre les deux images. L'EM revêt ainsi une importance primordiale en robotique pour le guidage automatique des engins, la conduite autonome des voitures, le contrôle des actions des robots, etc. La surveillance automatique est une application intéressante de l'EM ; en particulier pour l'identification des gestes et expressions faciales, l'analyse de mouvement de foule et comporte- ment de piétons. Dans le domaine médical, l'EM peut être utilisé pour l'analyse de mouvement du cœur, l'étude des écoulements bio-fluides et les déformations des organes. En traitement d'image, les vecteurs de mouvement obtenus peuvent être utilisés à des fins diverses comme le filtrage, la segmentation, la restauration d'image, la reconstruction 3D, etc. Dans les dernières années, on trouve aussi l'EM dans des applications militaires pour la poursuite des cibles, dans la météorologie pour l'étude de champs de vents, et dans l'océanographie pour l'étude des courants marins, etc.

II.4. Le mouvement dans la séquence vidéo [BET19]

Généralement, l'image est le résultat de la projection d'une scène réelle tridimensionnelle sur le plan de l'image bidimensionnelle (voir figure II.3). Le processus d'EM cherche donc à recouvrir la projection des différents mouvements tridimensionnels de la scène sur le plan image. De façon formelle, il consiste à estimer à partir d'une séquence d'images le mouvement apparent des objets composant une scène tridimensionnelle. La caméra ou les objets peuvent être selon les cas mobiles ou immobiles.

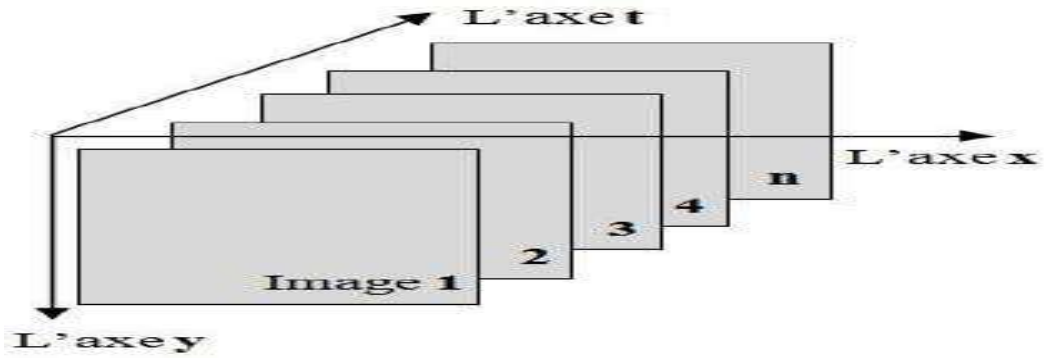


Figure II.2. *La séquence vidéo*

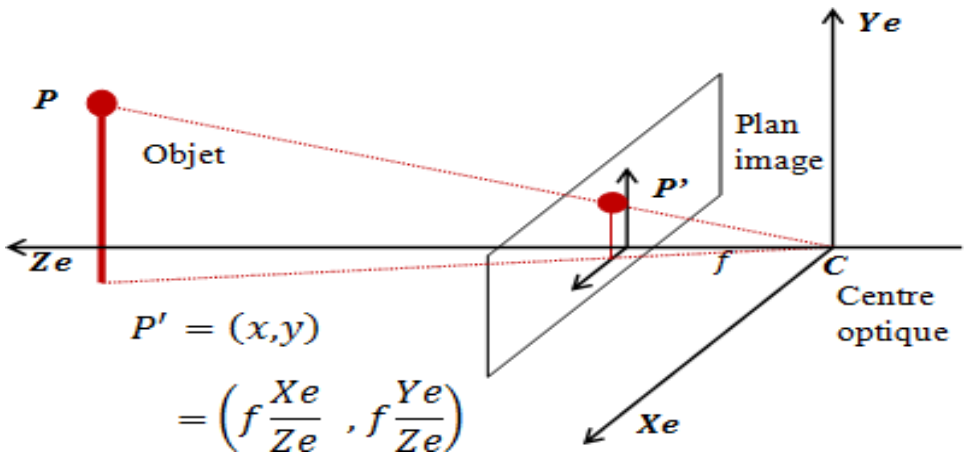


Figure II.3. *Projection 3D en 2D*

L'EM est un problème extrêmement difficile à résoudre. En effet, d'une part la projection 3D en 2D cause une perte d'information, et d'autre part, les variations d'intensité lumineuse sont les seules données disponibles. Donc, à partir des images 2D et en utilisant les variations d'intensité lumineuse, nous pouvons accéder au mouvement apparent, appelé aussi flot optique, traduisant de manière plus ou moins fidèle le mouvement réel (voir figure II.4).

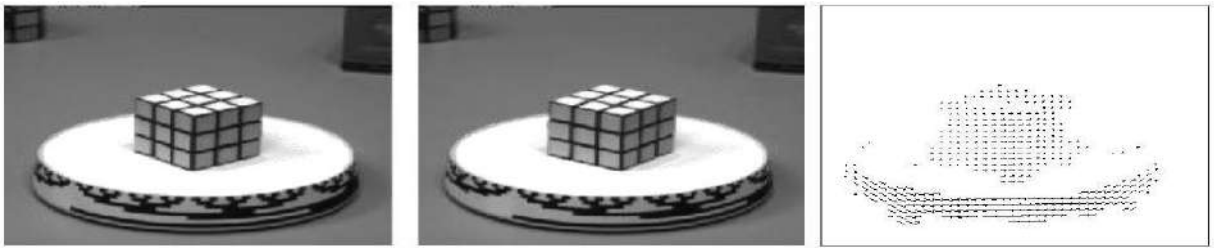


Figure II.4. *Exemples de flot optique*

II.5. Classification de mouvement

Les images représentent souvent la projection des scènes réelles 3D. C'est le mouvement observé (ou le mouvement apparent) dans une séquence temporelle d'images représente le plus souvent la projection du mouvement 3D, dans le plan image. Dans ce cas, le mouvement observé correspond à la fois au déplacement dans le plan image et également à la projection sur ce plan d'un mouvement 3D.

Le but de l'estimation de mouvement est donc d'estimer le champ de mouvement à partir d'une séquence d'image. Ce champ estimé est appelé mouvement estimé. On doit donc différencier entre :

Le mouvement réel.
Le mouvement apparent ou le mouvement observé.
Le mouvement estimé. [ALL21]

II.5.1. Le mouvement réel

Le mouvement réel est le mouvement qui se produit dans le repère. Ce mouvement est observé soit par l'œil humain soit par un système d'acquisition. L'inconvénient de ce mouvement est qu'il n'est pas toujours observé lors du passage à l'image [ALL21]. Dans le cas du mouvement réel il y a principalement deux types de mouvement :

II.5.1.1. Le mouvement d'objet rigide

Par définition un objet rigide est un objet solide qui garde toujours sa forme. Par conséquent, lorsqu'un tel objet se déplace, tous les points qui le composant se déplacent de façon identique. Les paramètres du mouvement sont les mêmes pour tous les points de l'objet. [ALL21]

II.5.1.2. Le mouvement d'objet déformables

Un objet déformable est un objet non rigide. La déformation se traduit toujours par un agrandissement ou une réduction de l'objet. Par conséquent, lorsqu'il est en mouvement, chacun de ses points peut subir un mouvement différent. [ALL21]

II.5.2. Le mouvement apparent [ALL21]

Le mouvement apparent est une combinaison de deux sources de mouvement : le mouvement propre 3D des objets dans la scène et celui de la caméra. Le mouvement de la caméra correspond au mouvement apparent 'globale'. Alors que le mouvement apparent des objets correspond au mouvement apparent 'local' relatif au mouvement global.

Le mouvement apparent est le mouvement qui est observé sur l'image. Il fournit une représentation du mouvement observé sous la forme d'une séquence temporelle d'image. Le champ de mouvement apparent représente la projection du mouvement 3D dans le plan image. C'est pourquoi, le champ de mouvement apparent s'appelle aussi mouvement projeté Il représente une approximation du mouvement réel (voir figure II.5). Bien que ce mouvement soit toujours observé, il n'est pas toujours dû à une réalité dans le monde réel. En effet, l'estimation d'un mouvement qui n'existe pas réellement peut avoir lieu.

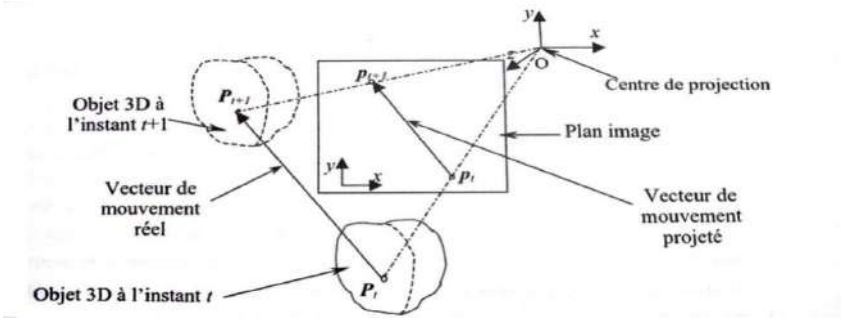


Figure 1I.5. *Illustration de mouvement réel et apparent.*

II.5.3. Le mouvement estimé

L'estimation du mouvement apparent porte uniquement sur ce qui est observé. Ce mouvement estimé est caractérise par le champ de vecteur vitesse ou par le champ de vecteurs déplacement. Ces deux champs de vecteurs vitesse et déplacement sont identique quand l'échantillonnage temporel de la séquence est constant. [ALL21]

II.6. Estimation de mouvement [BAI06]

Une des hypothèses qui doit être faite pour estimer le mouvement à partir du champ de mouvement apparent est que l'intensité image reste constante au cours du mouvement ou qu'elle varie d'une manière prédictible d'une image à l'autre.

L'hypothèse de conservation de l'intensité lumineuse en chaque point le long de la trajectoire du mouvement peut s'exprimer par l'équation DFD des différences entre les images déplacée (en anglais DFP= Displaced Frame Difference), c'est-à-dire entres les images aux instants t et $t + \Delta t$, avec $\Delta t = \pm 1$:

$$\boldsymbol{DFD = I(x + d_x y + d_y t + \Delta t) - I(x, y, t) = 0} \quad \text{(II.1)}$$

Où *I(x,y,t)* est l'intensité du point image $p = (x, y)$à l'instant t et $d(p) = (d_x(p), d_y(p))$ est le vecteur déplacement correspondant entre les instants t et $t + \Delta t$. Nous observons que :

a) Si les composant de *d* ne sont pas entières, le calcul de la DFD en chaque pixel nécessite une étape d'interpolation ;
b) Si les composantes de *d* correspondent à la valeur réelle du déplacement, alors l'erreur d'estimation, donc la DFD, est nulle en chaque pixel.

L'estimation du mouvement réel à partir du mouvement apparent peut être abordée de deux manières différentes :

a) L'estimation des vecteurs déplacements dans le plan d'image :

$$d(x, y, t) = (d_x(x, y, t), d_y(x, y, t)) \quad \text{(II.2)}$$

Estimés entre les images t et t+1 *;*

b) L'estimation des vecteurs vitesses :

$$v(x, y, t) = (v_x(x, y, t), v_y(x, y, t)) \quad \text{(II.3)}$$

Les vecteurs déplacements (respectivement vitesse) estimées peuvent varier en espace et en temps.

a) Estimation des vecteurs déplacements

L'estimation peut être vue comme un problème d'estimation de mouvement direct ou inverse (''avant'' ou ''arrière''), selon que l'estimation est réalisée entre l'instant *t* et *t* +1 ou entre l'instant *t* et *t*+1 (figure II.6)

Dans le cas de l'estimation directe (avant) du mouvement, le problème se pose de la manière suivant : si on connait les échantillons spatio-temporels $I_t(x,y)$ et $I_{t+1}(x,y)$, qui sont liés par la relation (hypothèse de conservation de l'intensité) :

$$I(x,y) = I_{t+1}\Big(x + d_x(x,y,t), y, d_y(x,y,t)\Big); \qquad \text{(II, 4)}$$

On doit trouver le vecteur de déplacement :

$$d_t(x,y) = \Big(d_{xd}(x,y,t), d_{yd}(x,y,t)\Big); \qquad \text{(II, 5)}$$

Dans le cas de l'estimation inverse (arrière) du mouvement où les vecteurs de déplacement sont définis entre les instants *t* et *t*-1, on a la relation :

$$I_t(x,y) = I_{t-1}\Big(x - d_{xi}(x,y,t), y - d_{yi}(x,y,t)\Big); \qquad \text{(II, 6)}$$

En estimation de mouvement, on utilise généralement l'estimation inverse (arrière). L'estimation avec compensation directe du mouvement est classiquement utilisée dans la compression prédictive de séquence d'images. Les valeurs prises par les déplacements sont souvent réelles, ce qui nécessite une étape d'interpolation pour l'estimation du mouvement.

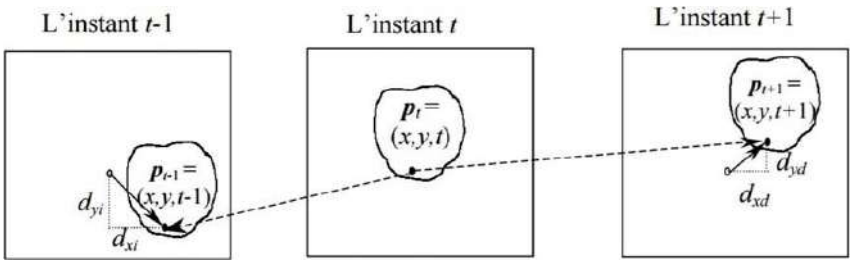


Figure II.6. *Estimation directe et inverse des vecteurs déplacements.*

Dans le cas de l'estimation directe : $d_d(x,y,t) = (d_{xd}(x,y,t), d_{yd}(x,y,t))$.
Dans le cas de l'estimation inverse : $d_i(x,y,t) = (d_{xi}(x,y,t), d_{yi}(x,y,t))$.

b) Estimation des vecteurs vitesses

étant donnés les échantillons $I_t(x, y)$, on doit déterminer les vecteurs vitesse *v(x, y, t)*. On observe que si la vitesse reste constante dans l'intervalle Δt entre deux images et si Δt est petit, alors

la vitesse estimée peut être assimilée au déplacement :

$$V_t(x, y) = d_t(x, y)/\Delta t \quad \text{(II.7)}$$

Pour un mouvement accéléré, on doit prendre en compte plus de deux images afin d'estimer correctement le champ de vitesse.

En conclusion, le champ de mouvement estimé peut être caractérisé par le champde vecteurs vitesse ou par le champ de vecteurs déplacement (ou de correspondance). Le champ de vitesse estimé et le champ de déplacement sont identiques quand l'échantillonnage temporel de la séquence est constant. L'estimation du mouvement par la perspective de l'estimation du champ de vecteurs vitesse ou par l'estimation de vecteurs déplacement (ou de correspondance) représente donc deux approches équivalentes, quand on connaît le pas d'échantillonnage temporel de la séquence. Dans la suite on s'intéresseraà l'estimation du champ de vecteurs déplacements.

L'estimation de mouvement est très sensible au bruit présent dans les images quipeut être interprété comme étant le résultat d'un mouvement dans la scène réelle.

L'estimation du mouvement est un problème mal-posé qui nécessite l'ajout decontraintes.

Un problème est dit mal-posé s'il n'a pas une solution unique et/ou la solutionn'est pas unique:

1. le problème d'existence de la solution peut être illustré dans le cas "d'occultation" ou de mouvement 3D multiples projetés;
2. le problème d'unicité de la solution est lié au problème d'ouverture.

II.7. Les difficultés de l'estimation de mouvement [BET19]

Malgré les progrès significatifs connus dans le domaine d'EM, mais une estimation de mouvement très précise reste difficile à obtenir à cause de nombreuses raisons théoriques et pratiques:

- Le passage du 3D au 2D provoque une perte de certaines informations importantes de la scène. Des points différents de l'espace peuvent être projetés sur le même point de l'image, une droite peut être représentée par un point et un plan peut devenir une droite, ce qui pose entre autres des problèmes de mise en correspondance.
- L'hypothèse de la conservation de la luminance d'un pixel au cours de déplacement combine effectivement plusieurs d'autres hypothèses concernant la réflectivité de la scène (la scène est lambertienne), l'illumination de la scène (illumination uniforme) et la formation de l'image au niveau de la caméra (pas de vignettage).
- Le problème d'occlusion se manifeste lors du recouvrement d'une surface par une autre, dû au mouvement de la caméra ou des objets dans la scène, certains points sont déplacés à des positions où ils sont devenus invisibles (voir figure II.7).

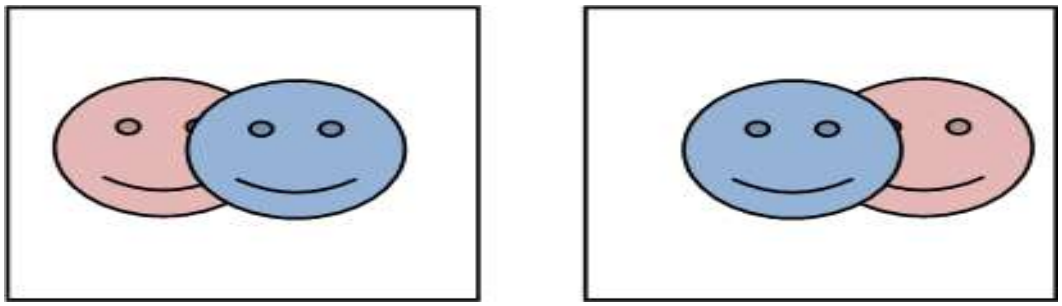


Figure II.7. *Problème d'occlusion*

- Le problème de région transparente : lorsque l'objet mobile est transparent, l'hypothèsede la conservation d'intensité n'est pas valide.
- Le problème de région homogène : un exemple sur ce problème est celui d'une sphère homogène tournant sur elle-même. Malgré qu'un mouvement réel est existant, mais aucune variation d'intensité n'est induite, donc le flot optique est nul.
- Le problème d'ouverture est une reformulation du fait que la solution du problème d'EM n'est pas unique. Dans la figure. II.6, un objet rectangulaire qui se déplace avec un mouvement uniforme, trois cas pratiques s'apparaissent lors de l'EM.

1. Dans la zone 1 : la zone est homogène donc le gradient d'intensité est nul. Il n'est pas possible d'en déduire le véritable mouvement.
2. Dans la zone 2 : le gradient d'intensité est orienté vers une seule direction. Le mouvement estimé est normal au contour.
3. Dans la zone 3 : plusieurs gradients d'intensité sont non nuls. Le mouvement réelest estimé correctement.

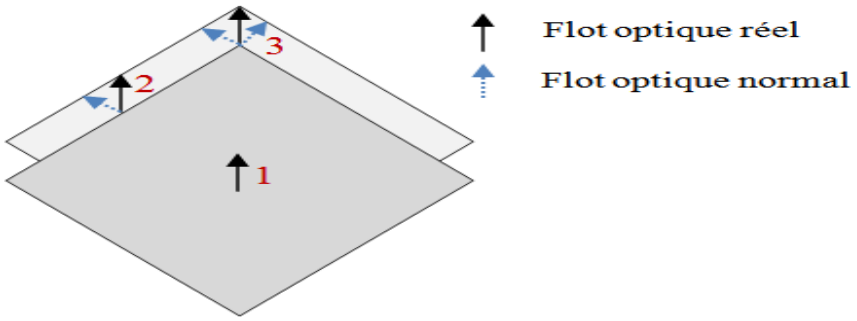


Figure II.8. *Problème d'ouverture*

- La présence du bruit : plusieurs sources contribuent à la présence du bruit dans la séquence vidéo tel que le bruit du capteur, le bruit dû aux erreurs de transmission de données, le bruit de quantification, etc.

II.8. Modèles du champ de vecteurs mouvement [BAI06]

A cause du caractère sous-déterminé de l'estimation de mouvement (problème d'ouverture), il est nécessaire d'introduire des modèles des contraintes supplémentaires concernant le champ de

mouvement ou directement des modèles (paramétriques ou non paramétriques) de ce champ.

II.8.1. Modèle de translation

Le modèle de mouvement le plus utilisé dans les méthodes d'estimation du mouvement est le modèle de mouvement constant ou modèle de translation, qui suppose que tous les pixels du bloc effectuent le même déplacement ***d*** (figure II.9).

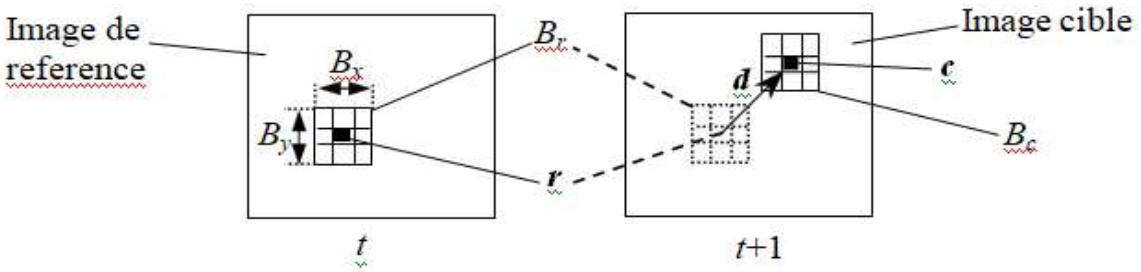


Figure II.9. *Modèle de translation du mouvement*

Ainsi, un bloc B_r de pixels de l'image de référence, de dimensions $B_x \times B_y$, centré sur le pixel (***r***) de coordonnées (*x,y*) dans l'image de référence à l'instant *t*, sera mis en correspondance avec le bloc B_c centré en pixel ***c***, dans l'image cible à l'instant $t+1$, ce qui s'écrit :

$$B_r(x, y, t) = B_c(x + d_x, y + d_y, t + 1) \qquad \text{(II.8)}$$

où $\boldsymbol{d}=(d_x, d_y)$ est le vecteur déplacement. Dans ce modèle, les blocs peuvent être disjoints (le cas le plus fréquent en pratique) ou chevauchants (figure 1.3.5). Dans le cas du modèle de translation avec des blocs disjoints, on peut attribuer un seul vecteur au bloc entier. Ainsi, la compensation de mouvement peut être réalisée en copiant l'information de gris ou de couleur, pixel par pixel, de l'image de référence dans l'image cible. Dans le cas du modèle de translation avec blocs chevauchants, on doit estimer le vecteur moyen de mouvement dans la région de superposition.

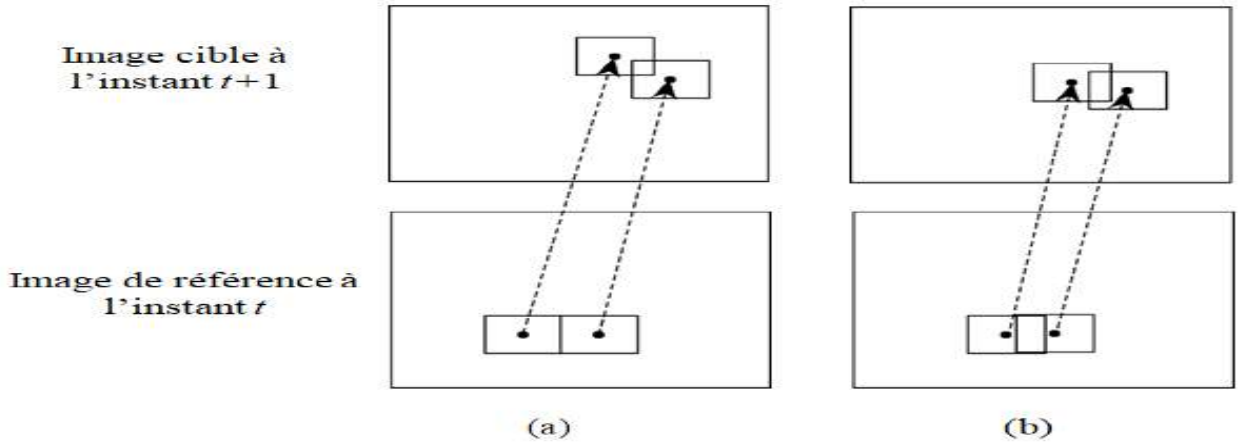


Figure II.10. *Modèle de translation, avec des blocs: (a) disjoints; (b) chevauchants*

Les avantages de l'estimation de mouvement utilisant un modèle de translation du mouvement sont:

- ✓ la compression obtenue en attribuant un seul vecteur de mouvement par bloc;
- ✓ la facilité d'implantation hardware.

Un des inconvénients de l'estimation de mouvement utilisant un modèle de translation du mouvement est constitué par les artefacts de type bloc, dus à un sous- échantillonnage du champ de mouvement.

II.8.2. Méthodes non paramétrique

L'avantage des méthodes non paramétriques est qu'elles peuvent être utilisées pour régulariser des mouvements complexes.

Les principales approches non paramétriques sont:

A. **Les méthodes différentielles :** Les méthodes différentielles d'estimation du mouvement s'appuient sur l'estimation des gradients spatio-temporels de l'intensité en chacun pixel. Dans le cas des images monochromatiques, les méthodes différentielles avec une contrainte de lissage spatio-temporel font l'hypothèse que le vecteur déplacement varie lentement au voisinage du pixel. Dans le cas des images couleurs, cette contrainte peut être imposée séparément pour chaque couleur, ce qui peut contraindre le vecteur déplacement dans 3 directions différentes. En général, une contrainte supplémentaire est nécessaire pour obtenir des résultats satisfaisants. Dans le cas de superposition des objets (occultations), une contrainte globale appliquée à l'image conduit à une estimation inexacte du mouvement.

.

B. **Les méthodes de mise en correspondance** : Dans ces méthodes on suppose quel'image est divisée en plusieurs régions de mouvement. Il existe des méthodes de mise en correspondance dans le plan image ou dans le plan transformé.

C. **Les méthodes récursives :** Les méthodes récursives sont basées sur la correction d'une prédiction (ou d'un estimé) du vecteur déplacement. Ainsi, l'estimée ou la prédiction peut être considérée comme étant le vecteur déplacement d'un pixelvoisin ou une combinaison linéaire des déplacements des pixels dans un voisinagedu pixel courant. La correction de la prédiction est faite en minimisant le gradient de l'image de différence entre les images déplacées DFD. Le pas de la prédiction est considéré, en général, comme une contrainte de lissage. Une extension de cette méthode dans le cas des méthodes de mise en correspondance de blocs, a comme résultat les méthodes d'estimation de type Wiener.

D. **Les méthodes statistiques :** Parmi les méthodes statistiques, les méthodes Markoviennes ou Bayesiennes sont les plus répandues. Elles utilisent pour estimer lechamp de déplacement des contraintes probabilistes de lissage, en général, sous la forme d'un champ aléatoire (éventuellement de Gibbs). Le principal désavantage de ces méthodes consiste en la grande quantité de calculs nécessaires pour estimer le champ de mouvement.

II.8.3. Modèle paramétrique du mouvement

Le modèle de translation du mouvement, décrit par la relation (II.8) peut être réécrit sous la forme d'une transformation linéaire de coordonnées spatiales :

$$\begin{cases} x_c = x_r + d_x \\ y_c = y_r + d_y \end{cases} \quad \text{(II.9)}$$

Où (x_c,y_c) représentent les coordonnées du point de l'image cible, à l'instant t+1 et (x_r,y_r) représentent les coordonnées du point de l'image de référence, à l'instant t.

La transformation spatiale (II.10) peut être généralisée pour inclure des transformations affines des coordonnées :

$$\begin{cases} x_c = a_1.x_r + a_2.y_r + a_3.d_x \\ y_c = a_4.x_r + a_5.y_r + a_6.d_y \end{cases} \quad \text{(II.10)}$$

Où $a_i \in \mathcal{R}, i \in \{1,\dots,6\}$

La transformation affine (II.10) peut traiter des mouvements de rotation des blocs ou des déformations relatives rigides (par exemple, la transformation d'un carré en parallélogramme). Parmi les transformations spatiales utilisées dans l'estimation du mouvement basées sur la mise en correspondance de blocs, on peut rappeler les transformations bilinéaires et projectives (figure II.11).

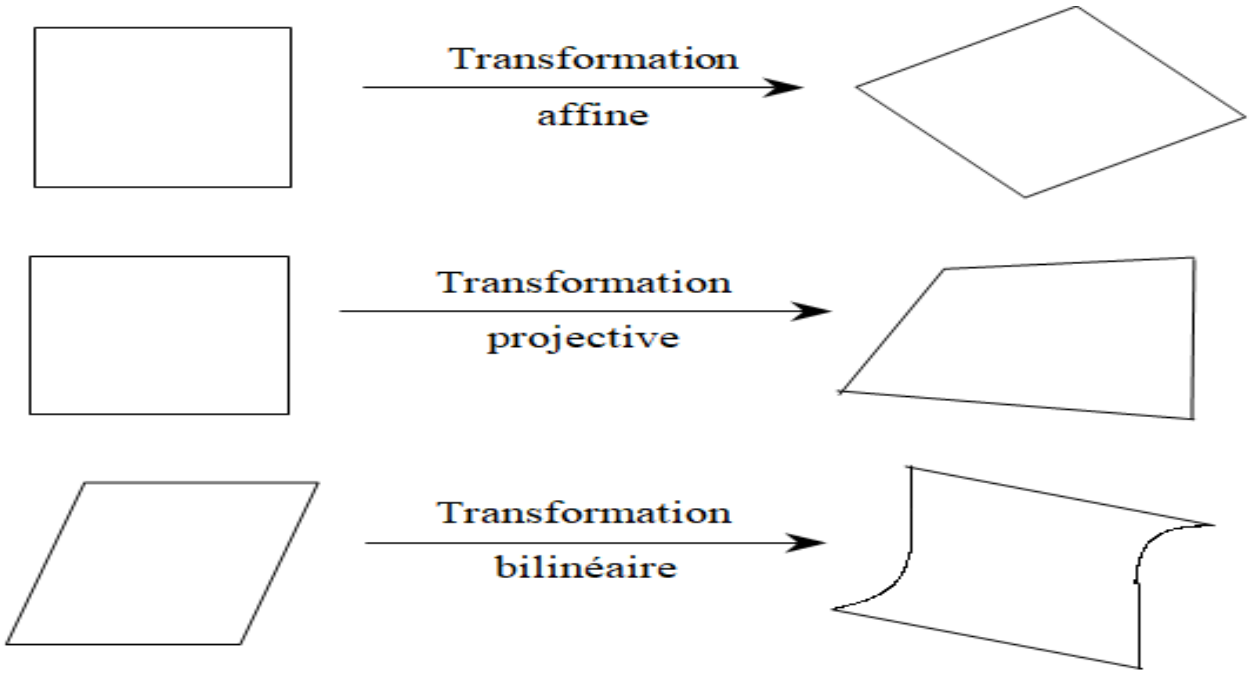


Figure II.11. *Exemples de transformations des coordonnées spatiales*

Une transformation projective peut être décrite par la relation:

$$\begin{cases} x_c = \frac{b_1.x_r+b_2.y_r+b_3}{b_7.x_r+b_8.y_r+1} \\ y_c = \frac{b_4.x_r+b_5.y_r+b_6}{b_7.x_r+b_8.y_r+1} \end{cases} \quad \text{(II.11)}$$

Où $b_i \in \mathcal{R}, i \in \{1, \dots, 8\}$.

Une transformation bilinéaire peut être décrite par la relation:

$$\begin{cases} x_c = c_1.x_r + c_2.y_r + c_3.x_r.y_r \\ y_c = c_5.x_r + c_6.y_r + c_7.x_r.y_r + c_8 \end{cases} \quad \text{(II .12)}$$

Où $c_i \in \mathcal{R}, i \in \{1, \dots, 8\}$.

Une transformation affine (resp. projective) correspond à la projection orthographique (resp. projective) du mouvement rigide 3D, dans le plan image 2D. La transformation bilinéaire n'a aucune correspondance physique particulière, mais elle présente des propriétés algébriques et géométriques intéressantes.

Une sous-classe de modèles paramétriques est représentée par les modèles appelés quasi-paramétriques, qui traitent la profondeur de chaque point comme une inconnue indépendante.

Le principal désavantage des méthodes paramétriques est qu'elles sont applicables seulement dans le cas de mouvements rigides. Les différents modèles de mouvement présentés dans ce paragraphe, peuvent être vus comme des contraintes possibles de régularisation locale du champ de déplacement.

II.9. Méthode différentielles [BAI06]

Les méthodes différentielles sont les méthodes basées sur les gradients spatiaux et temporaux d'intensité lumineuse. Ces gradients sont approchés, dans le cas discret, par des différences finies.

II.9.1. Equation de contrainte du mouvement

L'hypothèse d'invariance en temps de l'intensité lumineuse le long de la trajectoire du mouvement est exprimée par l'équation DFD (II.1). En réécrivant cette équation, on obtient:

$$\frac{dI(x,y,t)}{dt} = 0 \quad \text{(II.13)}$$

Où x et y varient en temps, le long de la trajectoire du mouvement. Sous l'hypothèse de différentiabilité spatio-temporelle de l'intensité lumineuse et en utilisant les règles de différentiation connues, on obtient:

$$I(x + d_x, y + d_y, t + \Delta t) = I(x, y, t) + \frac{\partial I(x,y,t)}{\partial x} d_x + \frac{\partial I(x,y,t)}{\partial y} d_y + \frac{\partial I(x,y,t)}{\partial t} \Delta t + tos \qquad \text{(II.14)}$$

Où *tos* sont des termes d'ordre supérieur. En remplaçant ce développement dansl'équation de DFD est en négligeant les termes d'ordre supérieur, on obtient:

$$DFD(x, y, t) = \frac{\partial I(x,y,t)}{\partial x} d_x + \frac{\partial I(x,y,t)}{\partial y} d_y + \frac{\partial I(x,y,t)}{\partial t} \Delta t = 0 \qquad \text{(II.15)}$$

Si nous divisons l'équation (II.15) par rapport à la distance en temps inter-images Δt, on obtient l'équation de contrainte du mouvement (ECM) nommée aussi l'équation duflux optique (EFO) :

$$\frac{\partial I(x,y,t)}{\partial x} v_x(x, y, t) + \frac{\partial I(x,y,t)}{\partial y} v_y(x, y, t) + \frac{\partial I(x,y,t)}{\partial t} = 0$$

(II.16)

où: $v_x(x,y,t)=d_x/\Delta t$, $v_y(x,y,t)=d_y/\Delta t$ sont les deux composantes de la vitesse.L'équation ECM peut être réécrite, en chaque point (x,y,t):

$$I_x . v_x + I_y . v_y + I_t = 0 \qquad \text{où} \qquad < \nabla I, v > + \frac{dI}{dt} = 0$$

(II.17)

Où : $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) = (I^x, I^y)$ est : Le gradient spatial de l(image avec les deux composantes $I^x = \frac{\partial I}{\partial x}$ et $I^y = \frac{\partial I}{\partial y}$, représentent respectivement. $I^t = \frac{\partial I}{\partial t}$ est le gradient temporel de l'intensité, et < .,.> représente le produit scalaire entre deux variables. Dans le cas discret, les gradients spatiaux et temporaux de l'image sont approches par des différences finies.

Comme nous pouvons l'observer, l'équation de contrainte du mouvement n'est pas suffisante pour déterminer d'une manière unique le champ de vitesse, parce que l'ECM représente une équation à deux inconnues (v_x et v_y), en chacun pixel. La composante normale, $v_\perp(x, y, t)$ du vecteur vitesse, orientée dans la direction du gradient spatial d'image $\frac{\nabla I}{\|\nabla I\|}$, est accessible. En effet, la composante tangentielle, tangente aux frontières photométriques des objets, disparaît suite au produit scalaire. La résolution du problème d'estimation du mouvement nécessite d'introduire une contrainte supplémentaire.

L'équation de contrainte du mouvement (ECM) décrit une droite dans l'espace de vitesses (v_x,v_y), comme sur la figure II.12 :

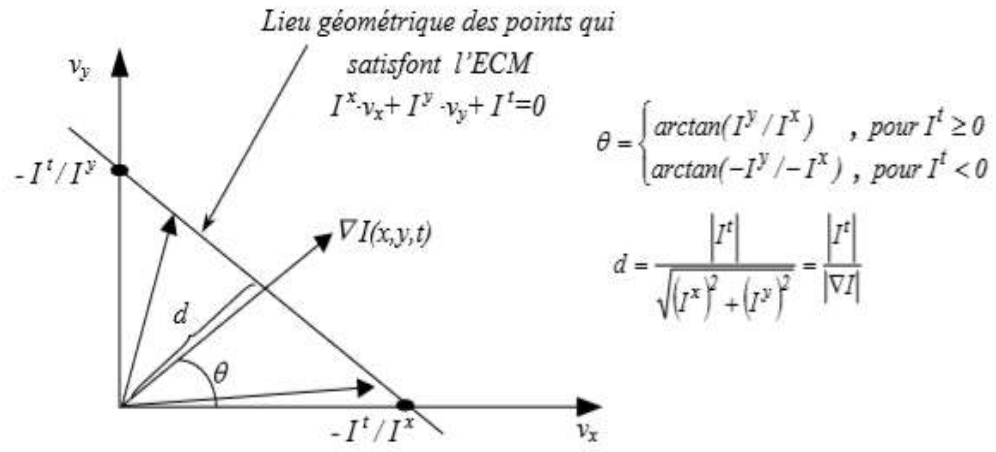


Figure II.12. *Illustration géométrique de l'équation et de la droite de contrainte du mouvement*

La distance qui sépare la droite de contrainte du mouvement d'origine dépend du rapport des modules des gradients spatiaux et temporaux, sachant que l'orientation de cette droite est déterminée par l'orientation du gradient spatial au point d'intérêt.

II.9.2 : Méthode de Horn et Schunck :

La méthode de Horn et Schunck n'est pas limitée aux translations. C'est une méthode différentielle itérative adaptée à l'estimation des petits déplacements et basée sur le développement en série de Taylor du DFD, des gradients spatiaux et temporaux. Le but est de trouver le champ de vecteurs qui satisfait l'équation de contrainte du mouvement en chacun pixel. Soit:

$$\xi_{flux} = I^x.v_x + I^y.v_y + I^t \qquad \text{(II.18)}$$

l'erreur par rapport à l'équation de contrainte du mouvement (ECM), en chaque point (*x,y,t*). On observe que l'ECM est satisfaite quand l'erreur ξ_{flux} est nulle. Dans la méthode de Horn et Schunck on minimise le carré de l'erreur ξ_{flux}. Une contrainte supplémentaire de lissage qui suppose que tous les points voisins ont un mouvement semblable est utilisée:

$$\xi_{flux}^2 = \|\nabla v_x\|^2 + \|\nabla v_y\|^2 = \left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2 \qquad \text{(II.19)}$$
$$= (v_x^x)^2 + (v_x^y)^2 + (v_y^x)^2 + (v_y^y)^2$$

Nous pouvons montrer que l'erreur $\xi_{uniformité}^2$ est faible si le champ de vecteurs vitesse est lisse. La méthode de Horn et Schunck consiste donc à minimiser une énergie de la forme :

$$\xi^2 = (1-\gamma).\xi_{flux}^2 + \gamma.\xi_{uniformité}^2 \qquad \text{(II.20)}$$

Où γ est un coefficient de pondération entre les termes d'attache aux données et de lissage. Pour minimiser ξ^2 (II.20) on utilise les équations d'Euler-Lagrange qui conduisent aux relations suivantes :

$$\begin{cases}(I^x).v_x + I^x.I^y.v_y = \gamma.(\bar{v}_x - v_x) - I^x.I^t \\ (I^y).v_y + I^x.I^y.v_x = \gamma.(\bar{v}_y - v_y) - I^y.I^t\end{cases} \quad \text{(II.21)}$$

Où $\bar{v}_x$ (resp. $\bar{v}_y$) est la valeur moyenne locale de v_x (resp. de v_y).
A partir de ce système on obtient v_x et v_y.

$V_x = \Delta v_x/\Delta$ Et $V_y = \Delta v_y/\Delta$

Avec : $\Delta v_x = \begin{vmatrix}\gamma.\bar{v}_x - I^x.I^t & I^x.I^y \\ \gamma.\bar{v}_y - I^y.I^t & \gamma + (I^y)^2\end{vmatrix}$

$\Leftrightarrow \Delta v_x = \gamma.(\gamma.\bar{v}_x + (I^y)^2.\bar{v}_x - I^x.I^t - \bar{v}_y.\ I^x.I^y)$

Et $\Delta = \gamma.(\gamma + (I^x)^2 + (I^y)^2\ ).$

$$\Rightarrow \begin{cases}\bar{v}_x = \frac{\Delta\bar{v}_x}{\Delta} = \frac{\bar{v}_x.(\gamma+(I^y)^2) - \bar{v}_y.\ I^x.I^y - I^x.I^t}{\gamma+(I^x)^2+(I^y)^2} \\ \bar{v}_y = \frac{\Delta\bar{v}_y}{\Delta} = \frac{\bar{v}_y.(\gamma+(I^x)^2) - \bar{v}_x.\ I^x.I^y - I^y.I^t}{\gamma+(I^x)^2+(I^y)^2}\end{cases} \quad \text{(II.22)}$$

Ainsi, en parcourant l'image entière, on obtient pour chaque pixel le couple de valeurs (v_x, v_y). Si on effectue plusieurs itération (le nombre des itérations étant le deuxième paramètre de la méthode de Horn et Schunck), ces valeurs vont diminuer, $\bar{v}_x$ et $\bar{v}_y$ étant des fractions des v_x etv_y. La solution est obtenue en utilisant la formule récurrente :

$$\begin{cases} v_{x,n+1} = \overline{v}_{x,n} - \frac{I^x.\lfloor I^x.\overline{V}_{x,n} + I^y.\overline{V}_{y,n} + I^t\rfloor}{\gamma+(I^x)^2+(I^y)^2} \\ v_{y,n+1} = \overline{v}_{y,n} - \frac{I^y.\lfloor I^x.\overline{V}_{x,n} + I^y.\overline{V}_{y,n} + I^t\rfloor}{\gamma+(I^x)^2+(I^y)^2}\end{cases} \quad \text{(II.23)}$$

La convergence est obtenue quand l'erreur (c'est-à-dire la différence entre deux itérations successives de v_x et v_y) est inférieure à un seuil choisi ou pour un nombre (n) d'itérations choisi.

II.9.3. Méthodes différentielles avancées

La méthode de Horn et Schunck impose l'équation de contrainte du mouvement (II.17) et la contrainte de lissage du champ de vitesse au niveau global (II.19), dans l'image entière ou dans la fenêtre d'estimation du mouvement. Cette globalité spécifique aux méthodes non adaptatives a au moins deux effets non souhaités:

- l'effet de "flou" sur les frontières du mouvement estimé, dû à la contrainte de lissage (II.19) dans la direction perpendiculaire à frontière d'un objet en mouvement. Les frontières du mouvement peuvent être préservées en imposant la contrainte de lissage seulement dans la direction dans laquelle le champ de vitesse varie peu ;

- les méthodes non adaptatives imposent l'équation de contrainte du mouvement dans les zones d'occultation, où l'effet de cette équation devrait être annulé. Ce résultatpeut être obtenu en modifiant localement le coefficient de pondération γ. Par exemple, dans les zones d'occultation on doit conserver seulement la contrainte de lissage.

En général, les méthodes différentielles adaptatives supposent une contrainte de lissage directionnel, en particulier le long des contours du mouvement et pas dans la direction perpendiculaire. Ainsi, dans on suppose la minimisation du même critère (II.20) seulement le long des contours.

II.9.3.1. Méthodes différentielles directionnelles

La contrainte de lissage directionnel peut être exprimée par la relation :

$$\xi_{ud}^2(v) = (\nabla v_x)^T . W . (\nabla v_x) + \left(\nabla v_y\right)^T . W . \left(\nabla v_y\right) \qquad \text{(II.24)}$$

Où $\boldsymbol{W}$ est une matrice de pondération qui pénalise les variations du champ de mouvement en fonction de variations spatiales de niveaux de gris par la séquence d'images. Un exemple de définition de la matrice $\boldsymbol{W}$ est :

$$W = \frac{F+\delta I}{trace(F+\delta I)} \qquad \text{(II.25)}$$

Où I est la matrice identité qui représente le terme de lissage global nécessaire pour assurer une matrice de pondération non nulle dans les zones d'uniformité spatiale, δ est un scalaire et :

$$F = \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 + b^2\left(\left(\frac{\partial^2 I}{\partial x^2}\right)^2 + \left(\frac{\partial^2 I}{\partial x.\partial y}\right)^2\right) & \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} + b^2\frac{\partial^2 I}{\partial x.\partial\partial y}\left(\left(\frac{\partial^2 I}{\partial x^2}\right)^2 + \left(\frac{\partial^2 I}{\partial x.\partial y}\right)^2\right) \\ \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} + b^2\frac{\partial^2 I}{\partial x.\partial\partial y}\left(\left(\frac{\partial^2 I}{\partial x^2}\right)^2 + \left(\frac{\partial^2 I}{\partial x.\partial y}\right)^2\right) & \left(\frac{\partial I}{\partial x}\right)^2 + b^2\left(\left(\frac{\partial^2 I}{\partial y^2}\right)^2 + \left(\frac{\partial^2 I}{\partial x.\partial y}\right)^2\right) \end{bmatrix}^{-1} \qquad \text{(II.26)}$$

Où b^2 est une constante.

Ainsi, dans le cadre de méthodes d'estimation du mouvement avec un lissage directionnel on minimise la fonction suivante, qui est similaire à la fonction utilisée dansle cas de la méthode de Horn et Schunck (II.20) :

$$\xi^2 = (1-\gamma).\xi_{flux}^2 + \gamma.\xi_{ud}^2 \qquad \text{(II.27)}$$

Où γ est un coefficient de lissage.

On observe que la méthode de Horn et Schunck (II.20) représente un cas particulier de cette formulation et correspond à l'équation (II.27), pour: δ=1 et $\boldsymbol{F}$=0.Pour minimiser la fonction (II.27) on peut utiliser une minimisation de type Gauss-Seidel, dans laquelle le terme actualisé à chaque itération peut être calculé par un algorithme linéaire. Les performances de la

méthode avec lissage directionnel dépendent de la précision avec laquelle on peut estimer les dérivées partielles d'ordre 1 et surtout celles d'ordre 2 de l'intensité de l'image, qui interviennent dans l'équation (II.27).

II.9.3.2. Méthodes différentielles hiérarchiques

Le concept de lissage directionnel et d'adaptabilité pondérée est décrit au travers une formulation hiérarchique. Une représentation multi résolution d'une séquence d'images peut être définie de la manière suivante:

$$I_a(x,y,t) = \int_S I(x,y,t).h\left(\frac{m-x}{a},\frac{n-y}{a}\right)dmdn \qquad \text{(II.28)}$$

Où a est le paramètre de résolution, S représente le support d'image et h est un noyau de filtrage. Un exemple de filtre passe-bas de paramètres A, B, C et D est donné par:

$$h(x,y,t) = \begin{cases} \{A\ exp\left[\frac{\left(-(C^2.x^2+D^2.y^2)\right)}{(B^2-(C^2.x^2+D^2.y^2))}\right], \text{pour } (C^2.x^2+D^2.y^2) < B^2 \\ 0 \end{cases} \qquad \text{(II.29)}$$

Nous pouvons observer que la résolution spatiale diminue quand α augmente. On peut observer aussi que les dérivées spatiales d'intensité $I_\alpha(x,y,t)$ peuvent être calculées par la convolution de $I(x,y,t)$ avec les dérivées partielles de $h(x,y)$, qui peuvent être calculées analytiquement. A chaque niveau de résolution on peut utiliser une méthode d'optimisation non-linéaire, comme par exemple la méthode de Quasi-Newton. L'estimation obtenue à un niveau de résolution est utilisée comme estimation initiale au niveau supérieur de résolution.

Pour augmenter l'efficacité des méthodes différentielles, on peut combiner différentes approches: l'approche hiérarchique avec l'approche directionnelle ou la prise en compte de discontinuités dans le champ de mouvement.

II.10. Méthodes de mise en correspondance

Les méthodes de mise en correspondance sont parmi les plus utilisées en pratique grâce à la simplicité de l'implantation physique du modèle de translation utilisé le plus souvent dans ces méthodes. Elles sont utilisées dans les standards de compression vidéo, comme H.261, MPEG-1, MPEG-2, MPEG-4 et MPEG-7.

Parmi les méthodes de mise en correspondance, on distingue deux classes:

- les méthodes de mise en correspondance dans le plan transformé;
- les méthodes de mise en correspondance dans le plan image.

II.10.1. Méthodes de mise en correspondance dans le plan transformée

Dans les méthodes de mise en correspondance dans le plan transformé, le champ de déplacement ***d*** est estimé par la mise en correspondance de blocs définis dans un plan transformé. La méthode la plus répandue est la méthode de corrélation de phase obtenue par la transformée de Fourier. Dans cette méthode, on calcule la corrélation de phase entreles transformées de Fourier de deux blocs correspondants dans les deux images successives.

On va noter : $\Im\{I(x,y,t)\} = F_t(f_x,f_y) = F_t(f)$ la transformée de Fourier de l'image *I(x,y,t)* à l'instant *t*, où $f = (f_x,f_y)$ représente les fréquences spatiales. La transformée de Fourier de l'image à l'instant $t+\Delta t$:$J(x,y,t+\Delta t) = I(x+d_x,y+d_y,t)$ est :

$$\Im_{t+\Delta t}\{J(x,y,t+\Delta t)\} = F_{t+\Delta t}(f) = F_t(f).exp\{-j.2\pi.(f_x.d_x + f_y.d_y)\} \quad \text{(II.30)}$$

Où ***I(p,t)*** est l'intensité d'image au point ***p***=*(x,y)*, à l'instant *t*, *x* et *y* représentent les coordonnées spatiales. Δt=±1 représente l'intervalle de temps entre les deux images et ***d(p)***=*(*d_x*(**x**),*d_y*(**p**))* est le vecteur déplacement correspondant au point ***p***=*(x,y)*.

Dans le cas d'un modèle de translation du mouvement, équivalent à l'équation (II.8) ou (II.9), la différence de phase entre les transformées de Fourier des deux blocs est donnée par:

$$arg\{F_t(f)/F_{t+\Delta t}(f)\} = \Delta\varphi(f_x,f_y) = -2\pi.(f_x.d_x + f_y.d_y) \quad \text{(II.31)}$$

L'équation (II.31) détermine un plan dans l'espace des variables f_x et f_y .Le mouvement entre les deux images consécutives peut être exprime par la différance de phase entre deux pairs de fréquences spatiales. Il est donne par un système linéaire des deux équations, correspondant ainsi à l'orientation du plan (équation (II.31)).

Les méthodes de corrélation de phase estiment le déplacement relatif entre deux images consécutives en utilisant une fonction normalisée d'inter-corrélation calculée dans l'espace de Fourier. Le résultat est un Dirac dont la position correspond au vecteurdéplacement. Un exemple d'une telle fonction ou coefficient d'inter-corrélation entre les images *t* et *t*+Δt est:

$$C_{t,t+\Delta t}(x,y) = I_{t,t+\Delta t}(x+d_x,y+d_y) * I_t(-x,-y) \quad \text{(II.32)}$$

Où * représente l'opération de convolution. La transformée de Fourier de l'équation (II.32) fournit l'expression complexe du spectre inter-images, *F** étant la conjuguée de *F*:

$$C_{t,t+\Delta t}(f_x,f_y) = F_{t+\Delta t}(f_x,f_y).F_t(f_x,f_y) \quad \text{(II.33)}$$

En normalisant l'amplitude du spectre inter-images (II.33), on obtient la phase du spectre :

$$arg\{C_{t,t+\Delta t}(f_x, f_y)\} = \frac{F_{t+\Delta t}(f_x,f_y).F_t(f_x,f_y)}{|F_{t+\Delta t}(f_x,f_y).F_t^*(f_x,f_y)|} \quad \text{(II.34)}$$

En supposant le modèle de translation (II.3) et en remplaçant (II.31) en (II.34), on obtient :

$$arg\{C_{t,t+\Delta t}(f_x, f_y)\} = exp\{-j.2\pi.(f_x.d_x + f_y.d_y)\} \quad \text{(II.35)}$$

La transformation de Fourier inverse (II.35), donne la fonction de corrélation de phase :

$$arg\{C_{t,t+\Delta t}(f_x, f_y)\} = \delta(x - d_x, y - d_y) \quad \text{(II.36)}$$

$$\delta(q) = \begin{cases} 1, \ pour \ \ q = 0 \\ 0, \ ailleurs \end{cases}$$

En pratique, la transformée de Fourier est remplacée par la Transformée de Fourier (TF) discrète, pour laquelle il y a des algorithmes rapides de calcul. Un exemple d'un tel algorithme utilisant la corrélation de phase est décrite comme suit :

1. Calcul de TF discrète 2D des blocs correspondants dans les images *t* et *t+Δt*;
2. Calcul de la phase du spectre inter-images (II.35);
3. Calcul de la TF inverse de la relation (II.35), afin d'obtenir la fonction de corrélation de phase (II.36).
4. Détection de la position du pic du maximum de corrélation de phase, donnant la valeur estimée du déplacement $\hat{\boldsymbol{d}} = (\hat{d}_X, \hat{d}_Y)$.

Dans le cas idéal, la fonction de corrélation de phase doit présenter un seul maximum (pic) qui indique le déplacement relatif entre les deux blocs. En pratique, la fonction de corrélation de phase peut présenter plusieurs pics à cause de:

- L'utilisation de la transformée de Fourier discrète (TFD) au lieu de la transformée de Fourier continue;
- la présence de bruit;
- la présence à l'intérieur du bloc de référence de plusieurs objets en mouvement.

Les méthodes de corrélation de phase ont plusieurs avantages qui sont:

- Leur insensibilité aux variations d'illumination de la scène. Les méthodes de corrélation de phase sont relativement insensibles aux modifications de l'illumination parce qu'un déplacement ou une multiplication par une constante de la valeur moyenne de l'intensité n'affecte pas la phase de la transformée de Fourier. Les méthodes de corrélation de la phase sont insensibles aux variations d'amplitude de la transformée de Fourier.
- Leur capacité à traiter plusieurs objets en mouvement. La possibilité de traiter plusieurs objets en mouvement ayant des vitesses différentes dans un même bloc de calcul est une propriété très importante. Celle-ci se traduit par la présence de plusieurs pics dans la

fonction de corrélation de phase, chacun d'entre eux correspondant au déplacement d'un objet. Dans ce cas, une étape supplémentaire est nécessaire pour déterminer quel vecteur déplacement correspond à quel pixel du bloc de calcul. Cette étape peut être réalisée en comparant l'amplitude de la différence entre les images déplacées (DFD) avec chaque vecteur déplacement.

En pratique, les méthodes de corrélation de phase sont rarement utilisées à cause de plusieurs inconvénients, parmi lesquels on retiendra:

- Les effets de bord. Pour obtenir un impulsion parfaite, le déplacement doit être cyclique (à cause du calcul par transformée de Fourier), ce qui n'est pas le cas en pratique car les objets qui disparaissent à un bord du bloc de calcul ne réapparaissent pas sur l'autre bord. Ceci entraînera donc la présence de pics non souhaités, conduisant à une estimation erronée du mouvement.
- Les déformations du spectre, dues aux valeurs non entières correspondant aux vecteurs de mouvement. Pour obtenir un pic idéal, les composantes du vecteur déplacement doivent correspondre à un multiple de la fréquence fondamentale. Dans le cas contraire, de faux pics peuvent apparaître, déformant le spectre.
- L'amplitude maximale du déplacement qui peut être estimée. Parce que la TFD est périodique, de période égale à la dimension du bloc ($B_x \times B_y$), le déplacement estimé doit être réécrit :

$$\hat{d}_i \begin{cases} d_i, \text{si} |d_i| \leq d_{max}/2, \text{quand } d_{max} \text{est pair, ou si } |d_i| \leq (d_{max}-1)/2, \text{quand } d_{max} \text{est impair} \\ d_i - d_{max}, \text{en rest} \end{cases}$$

- pour permettre des déplacements négatifs. Ainsi, quand le déplacement maximal(d_{max}) qui peut être estimé est pair, la gamme maximale du déplacement qui peut être estimé est [-d_{max}/2+1, d_{max}/2]. Par exemple, pour estimer un déplacement maximal de8 pixels, c'est-à-dire une amplitude maximale du déplacement comprise entre [-7, +8] pixels, la dimension du bloc de calcul doit être de [16×16] pixels.

La dimension du bloc est un paramètre important dans tous les algorithmes d'estimation du mouvement utilisant la mise en correspondance. Le choix de la dimension du bloc suppose un compromis entre deux contraintes contradictoires. Dans le cas des méthodes de corrélation de la phase, le bloc doit être suffisamment grand pour estimer de grands déplacements. Mais, celui ci doit rester suffisamment petit pour que le déplacement soit constant à l'intérieur du bloc. Une manière de résoudre le problème consiste à utiliser des méthodes hiérarchiques multi résolutions.

II.10.2. Méthodes de mise en correspondance dans le plan image

La méthode de mise en correspondance par bloc dans le plan image Block Matching en Anglais est une méthode largement utilisé dans l'estimation de mouvement, pour les grandes familles de codecs MPEG (MPEG, MPEG-1, MPEG-2, et MPEG-4) ainsi pour les codecs H.261 et H263.

II.11. Conclusion

Dans ce chapitre, nous avons présenté les types de mouvement représentant l'image, en définissant l'estimation du mouvement ainsi que ses différents types. Ensuite, nous avons décrit les difficultés rencontrés et posés lors de l'estimation de mouvement et les méthodes d'estimation de mouvement.

Dans le chapitre suivant, nous allons détailler les différentes techniques de recherches à méthode de mise en correspondance ou Block-Matching en Anglais qui sont des méthodes d'estimation du mouvement les plus utilisées dans la pratique

CHAPITRE III :
Technique de block Matching

III.1. Introduction

La méthode de compensation de mouvement la plus utilisée est la compensation de mouvement par correspondance de bloc (Block Matching).

Les méthodes du Block-Matching sont parmi les méthodes d'estimation du mouvement les plus utilisées dans la pratique. Elles peuvent être utilisées pour découvrir des redondances temporelles dans une séquence vidéo tel qu'un objet sans (avec) mouvement dans une vidéo, ce qui accroît l'efficacité de compression d'une image. On les retrouve presque dans tous les standards de compression vidéo (H.261, MPEG-1, MPEG-2 ou MPEG-4) [DAL13].

Dans ce chapitre, nous nous s'intéresserons aux déférentes stratégies de recherche, déférentes méthodes d'algorithme de block Matching et les plus utilisées.

III.2. Principe de mise en correspondance [DAL13]

Pour simplifier l'estimation, nous ne considérons que le mouvement présent entre deux images successives : l'image courante et l'image de référence. Chaque image est subdivisée en blocs de taille égale (généralement de 8x8 ou 16 x16 pixels), Nous faisons l'hypothèse que le mouvement des pixels est uniforme à l'intérieur d'un bloc. De manière informelle, l'algorithme consiste, pour un bloc de l'image courante, à choisir un bloc dans l'image de référence et à calculer un critère comparaison entre ces deux blocs, tout en supposant que les pixels du même bloc ont le même mouvement translationnel auront ainsi le même vecteur de mouvement.

L'opération est réitérée en choisissant un autre bloc jusqu'à ce que tous les blocs d'une zone déterminée de l'image de référence (appelée « fenêtre de recherche ») aient été testés ou jusqu'à un critère d'arrêt arbitraire. Le bloc le plus semblable est ainsi identifié dans l'image de référence pour chaque bloc de l'image courante. Nous obtenons de cette manière pour tout bloc un vecteur déplacement caractéristique du mouvement de ce dernier.

L'utilisation d'une fenêtre de recherche permet de limiter le nombre de blocs de référence que l'algorithme doit tester. La taille de la fenêtre dépend d'un déplacement maximal autorisé fixé par l'utilisateur.

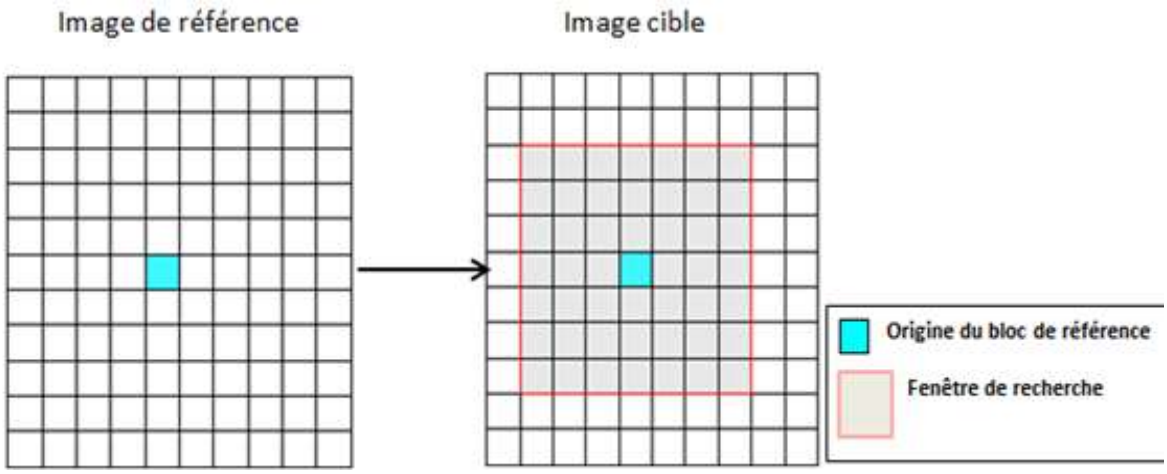


Figure III.1. *Illustration de méthode Block Matching*

III.3. Critères de mise en correspondance

Pour chaque point de l'image courante à l'instant t on cherche le point d'intensité la plus proche dans l'image cible (instant t-1 ou t+1). Pour éviter les correspondances incohérentes dans l'image cible, on limite la recherche à une fenêtre centrée sur le point courant (x,y). Ainsi, le déplacement d'un pixel de coordonnées (x,y), centre du bloc B de dimension $B_x \times B_y$, de l'image courante, est déterminé par la position du bloc de même dimension, le plus proche au sens d'un critère de similarité, dans l'image cible (instants t-1 ou t+1), figure 1.5.1. La recherche est limitée à une fenêtre de dimensions: $F = (2 \cdot d_{x_max}+1) \times (2 \cdot d_{y_max}+1)$, où d_{x_max} (resp. d_{y_max}) est la valeur maximale du déplacement à estimer dans la direction x (resp. y). [BAI06]

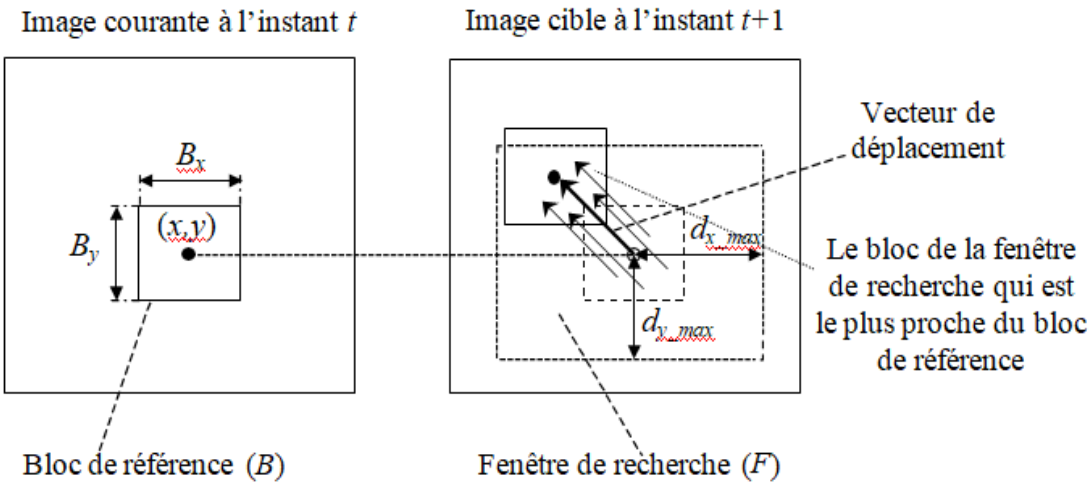


Figure III.2. *Principe de mise en correspondance par blocs.*

III.3.1. Différence absolue

La différence entre deux macroblocs peut être mesurée par leur différence absolue moyenne (MAD : Mean Absolute Difference), définie comme suit : [ALB13], [BAI06]

$$MAD(l,k) = \frac{1}{B_x.B_y}\sum_{i=1}^{B_x}\sum_{j=1}^{B_y}|B_r(i,j) - B_c(i,j)|$$
$$= \frac{1}{B_x.B_y}\sum_{i=1}^{B_x}\sum_{j=1}^{B_y}\left|\sum_{i=1}^{B_x}\sum_{j=1}^{B_y} I(i,j,t_k) - I(i+d_x, j+d_y, t_k+\delta t)\right| \quad \text{(III.1)}$$

Où (l,k) sont les coordonnées du centre du bloc et (i,j) sont les coordonnées des pixels du bloc.

III.3.2. Erreur quadratique MSD ou MSE [BAI06]

Elle peut être aussi mesurée par la différence quadratique moyenne (MSD Mean Square Difference) qui est donnée par :

$$MSE(l,k) = \frac{1}{B_x.B_y}\sum_{i=1}^{B_x}\sum_{j=1}^{B_y}[B_r(i,j) - B_c(i,j)]^2 \quad \text{(III.2)}$$

L'estimé du vecteur déplacement est $\hat{d} = (\hat{d}_x, \hat{d}_y)$ qui minimise MSE :

$$MSE(l,k) = \frac{1}{B_x.B_y}\sum_{i=1}^{B_x}\sum_{j=1}^{B_y}[B_r(i,j) - B_c(i,j)]^2$$
$$= \frac{1}{B_x.B_y}\sum_{i=1}^{B_x}\sum_{j=1}^{B_y}[I(i,j,t_k) - I(i,j,t_k+\delta t)]^2 \quad \text{(III.3)}$$

$$\Rightarrow \hat{d} = (\hat{d}_x, \hat{d}_y) = arg_{(d_x,d_y)} \min MSE\ (d_x, d_y) \quad \text{(III.4)}$$

III.4. Dimension optimal de bloc et de la fenêtre de recherche [DAL13]

La stratégie de recherche et les dimensions de la fenêtre de recherche et du bloc influencent le coût de calcul. La détermination de leurs dimensions optimales respectives nécessite toujours un compromis. Une grande fenêtre de recherche implique des calculs longs et un risque élevé de confusion du bloc recherché avec un bloc semblable. Mais plus la taille de la fenêtre de recherche décroit, plus le déplacement maximal estimé diminue. De la même manière, quand les blocs sont grands, le coût de calcul est grand, la résolution spéciale est faible et un bloc peut contenir les pixels appartenant à des objets différents. Inversement, un bloc trop petit peut ne pas contenir suffisamment d'information discriminante. Ce dernier problème apparait, en particulier, dans les zones homogènes ou la mise en correspondance n'est pas fiable, pouvant devenir aléatoire. Le compromis peut être résumé en quatre cas suivants :

- Fenêtre de recherche grande ⇒ coût de calcul élevé et un risque élevé de confusion du bloc recherché
- Fenêtre de recherche petite ⇒ petit déplacements.
- Bloc grand ⇒ coût de calcul élevé et le bloc peut contenir les pixels appartenant à des

objets différents.

- Bloc petit ⇒ pas de discriminant.

III.5. Stratégies de recherche [BAI06]

La méthode de mise en correspondance de blocs suppose une stratégie de recherche exhaustive dans l'image et dans la fenêtre de recherche. Un balayage exhaustif peut s'envisager pour l'image et également pour la fenêtre de recherche, le bloc decomparaison étant alors déplacé pixel par pixel, à l'intérieur de la fenêtre de recherche.

Le balayage exhaustif présente le désavantage d'un temps de calcul prohibitif, car il impose d'estimer le critère de ressemblance en chacun pixel et pour toutes les positions possibles du bloc dans la fenêtre de recherche.

Le balayage non exhaustif nécessite le choix d'une stratégie. On envisagera les 2 cas classiques suivants (figure III.3):

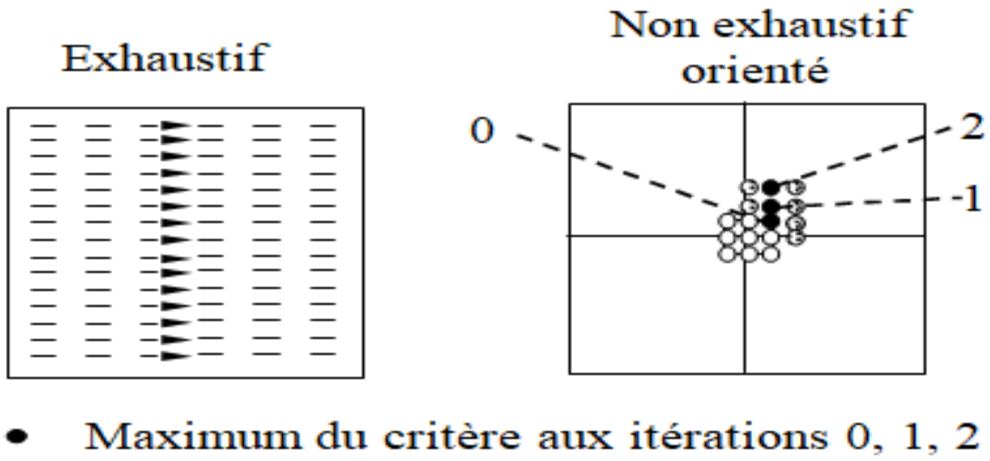


Figure III.3. *Stratégies de balayage de la fenêtre de recherche.*

Le balayage non exhaustif orienté consiste en une exploration initiale du voisinage direct de la fenêtre de recherche puis sélective dans la direction de la ressemblance croissante (points noirs sur la figure III.3). Les stratégies de recherche orientée les plus utilisées sont:

- la recherche en *n* pas;
- la recherche 2D-logarithmique (2DLog);
- la recherche orthogonale.

Ces stratégies de balayage sont plus rapides, mais peuvent échouer dans le cas d'existence de plusieurs optima (maxima ou minima) locaux.

III.5.1. Recherche en *n* pas

L'algorithme de recherche en 3 pas est basé sur un raffinement successif, avec décroissance logarithmique du pas (figure III.4). La flèche indique la direction et le sens de la recherche pour le pas suivant.

Le principe de recherche en trois pas, comme les principes des autres méthodes de balayage qui seront présentées par la suite, sera illustré en considérant un déplacement maximal de 7 pixels. Le pixel noté "0" sur la figure III.3 représente le pixel courant. A la première itération le critère de ressemblance est évalué, en 9 points: le point noté "0" etles points notés "1".

Si le critère de ressemblance optimal correspondant est le point "0", le déplacement estimé est nul. Le pas initial (1) est choisi égal à la moitié du déplacement maximal admis d_{max} (on va considérer $d_{x_max}= d_{y_max}=d_{max}$), c'est-à-dire $\lceil d_{max}/2 \rceil$ où $\lceil . \rceil$ est la fonction d'arrondi à l'entier supérieur. A chaque itération, on évalue le critère de ressemblance et l'optimum devient le centre de recherche pour l'itération suivante. L'amplitude du pas, à chaque itération, décroît selon une loi logarithmique.

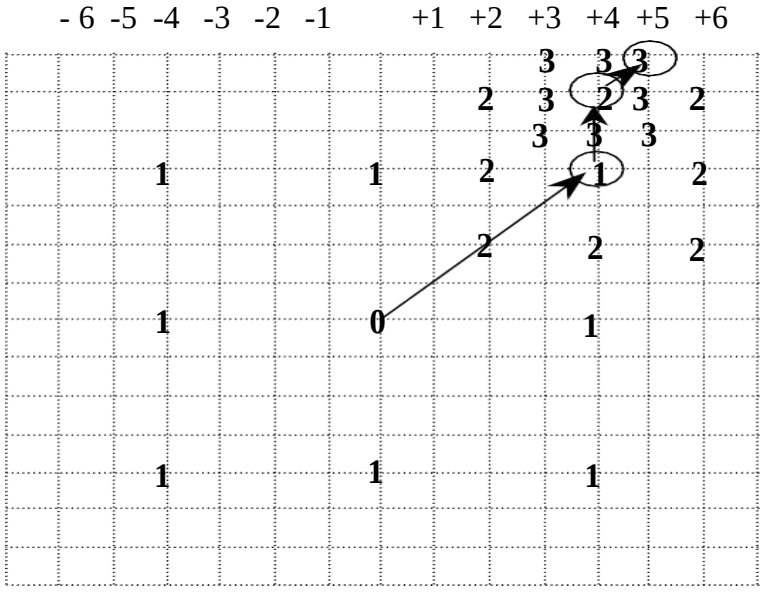


Figure III.4. *Le principe de recherche en trois pas.*

L'estimation du déplacement sera obtenue à la troisième itération, le pas vaut 1 pixel. Pour d_{max}=7, le nombre total de points de comparaison nécessaire est : (9+8+8)=25. L'algorithme peut être étendu à des fenêtres de recherche plus larges. En utilisant la même stratégie, le nombre n de points de comparaison nécessaires sera n=1+8·$\lceil \log2\ (d_{max}+1) \rceil$. L'algorithme est dit "recherche en n-pas" ou "recherche en log-n".

III.5.2. Recherche 2D-logarithmique

La recherche 2D-logarithmique met en œuvre une stratégie de recherche en croix (+) à chaque itération. Le pas initial est égal à $\lceil d_{max}/4 \rceil$. Il est réduit de moitié quand le point correspondant à l'optimum trouvé est le centre ou situé au bord de lafenêtre de recherche. Dans le cas contraire, le pas ne change pas. Quand le pas atteint la valeur 1, les 8 points immédiatement voisins du pixel central sont testés. Sur la figure III.5 on présente deux situations différentes: la situation en haut nécessite: n=5+3+3+8=19 points de calcul et la situation en bas et à droite nécessite: n=5+3+2+3+2+8=23 points de calc

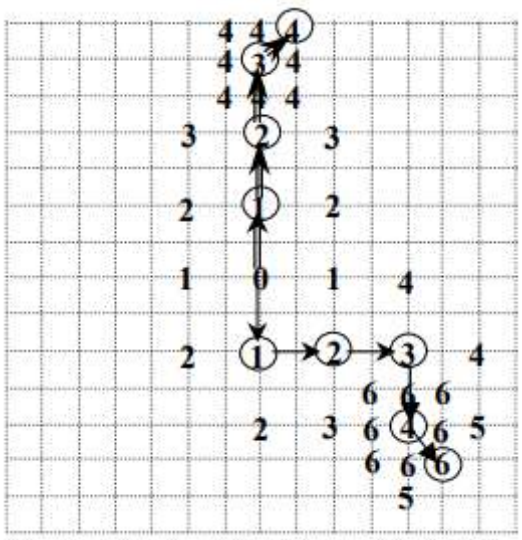


Figure III.5. *Le principe de recherche 2D-logarithmique.*

III.5.3. Recherche orthogonale

L'algorithme de recherche orthogonale consiste en paires de comparaisons horizontales et verticales avec une décroissance logarithmique du pas. La dimension initiale du pas est $\lfloor d_{max}/2 \rfloor$, où $\lfloor . \rfloor$ est la fonction d'arrondi à l'entier inférieur.Le principe de recherche orthogonale est montré sur la figure III.6.

Une itération a deux étapes. Dans la première étape, trois calculs du critère de ressemblance sont faits en trois pixels horizontaux, 0 et 1. L'optimum dans la direction horizontale devient le centre de la deuxième étape (recherche dans la direction verticale), le pas restant inchangé. Le pas est alors diminué de moitié en répétant la même stratégie de recherche à l'itération suivante. L'algorithme s'arrête quand le pas devient égal à 1. Pour d_{max}=7, dans le cas présenté sur la figure III.6 en haut-à droite, l'algorithme de recherche orthogonale nécessite d'estimer le critère en: n=3+2+2+2+2+2=13 points de calcul. Dans le cas général: $n=1+4\cdot\lfloor \log2(d_{max}+1) \rfloor$ points de calcul sont nécessaires.

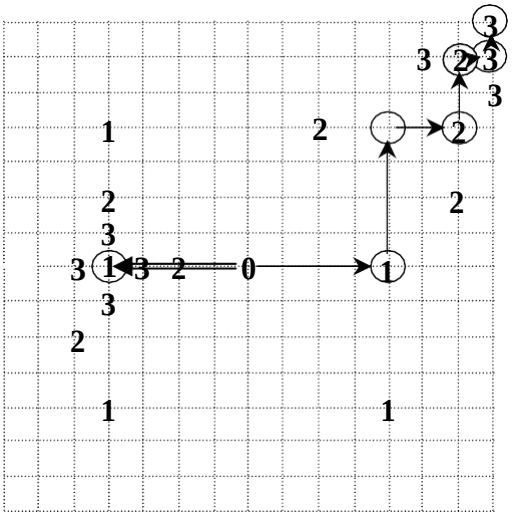


Figure III.6. *Principe de recherche orthogonale.*

III.6. Les récents algorithmes étudiés

Il existe de nombreuses méthodes de block-matching cherchant à optimiser l'efficacité et la rapidité de l'algorithme.

III. 6. 1. L'algorithme de recherche complet FS

La technique de recherche complète FS (Full Search en Anglais) consiste à faire la recherche du bloc le plus correspondant de l'image actuelle dans l'image précédente [CHE94, RED04]. Cet algorithme est le plus simple. Il est aussi le moins performant car il n'optimise pas la recherche : il n'y a pas de seuil pour la fenêtre de recherche. Il sert néanmoins de référence pour les autres algorithmes.

Pour une image de taille $\boldsymbol{P} \times \boldsymbol{Q}$ avec une fenêtre de recherche de taille$(2.\boldsymbol{W} + 1) \times (2.\boldsymbol{W} + 1)$, un taux d'image de $\boldsymbol{T}$ frames/seconde et des blocs de taille $\boldsymbol{N} \times \boldsymbol{N}$, le nombre d'instruction processeur par seconde est :

$$T \times (P \times Q/N^2) \times (2.W + 1)^2 \times (2.N^2 - 1)$$

Cette méthode est optimal car elle teste tous les blocs possibles, mais nécessite énormément de calculs (c'est l'inconvénient majeur de cet algorithme) [LIN97].

Il est possible de réduire les calculs de cette méthode en rejetant la plupart des blocs : l'inégalité triangulaire montre que si la différence entre les luminances moyenne des deux blocs est moins bonne que le minimum actuel, il n'est pas nécessaire de calculer le critère.
L'erreur sera de toute façon. Supérieure. On peut donc ne pas traiter la plupart des blocs.

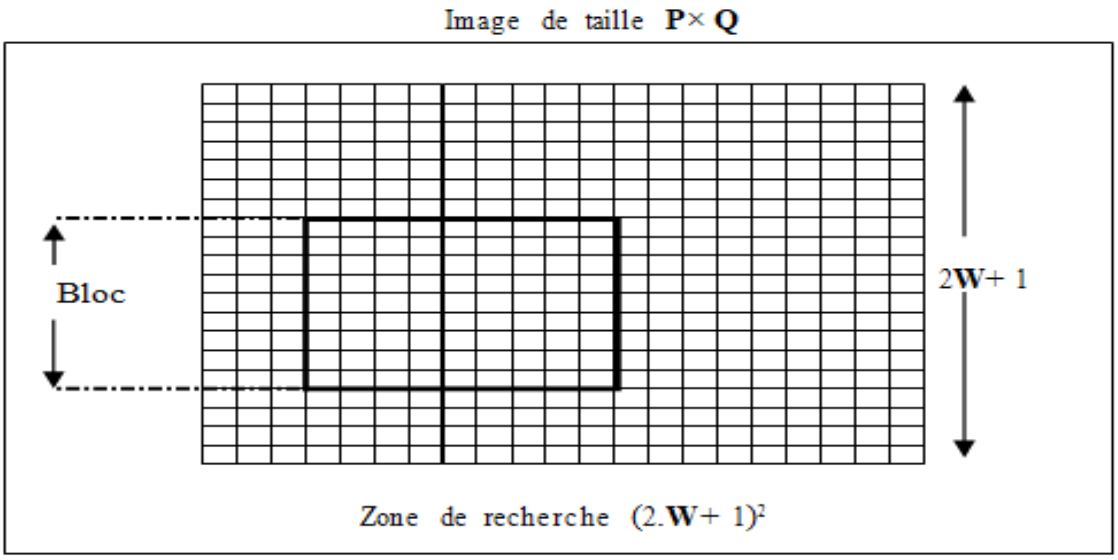


Figure III.7. *Modèle de recherche"Full Search" FS*

III.6.2. L'algorithme à trois étapes de recherche TSS

L'algorithme à trios étapes de recherche (**T**hree **S**tep **S**earch en Anglais) est l'algorithme le plus connu. C'est une technique de mise en correspondance de blocs dans le plan image et il utilise au maximum (03) étapes. Il est très utilisé pour l'estimation de mouvement dans les applications de la compression vidéo à bas débit binaire comme la téléphonie-visuelle et la vidéo-conférence. Le TSS est un modèle simple et effectif utilisé pour l'estimation de mouvement. Comparé avec EBMA il a un facteur d'économie plus grand que 100. Il à un nombre fixe de trois étapes de recherche et un nombre maximum et minimum de 25 points de recherche.

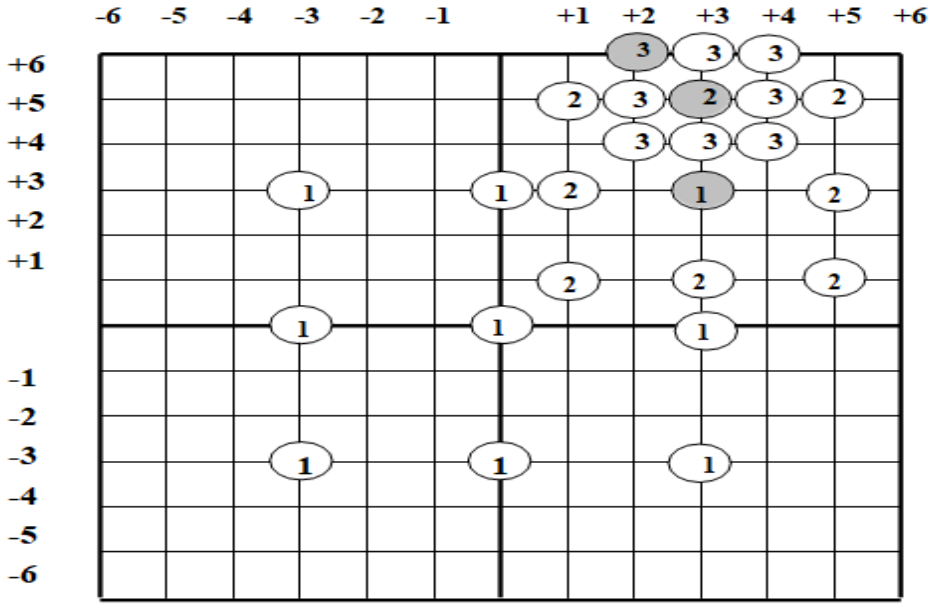


Figure III.8. *Modèle de recherche à troisétapes TSS.*

La technique TSS commence avec une taille de l'étape plus large ou égale à la moitié de la gamme maximale. A la fin de recherche la taille de l'étape devient 1 pixel. Comme suit [JIN04]:

1. neuf points de recherche (le centre et huit points au frontière de la zone de recherche) sont comparés (son critère de minimum) dans la première étape, et après huit points seulement vont recherchés.(Les neuf points sont signés par le numéro (1) dans la figure 5)
2. Au début de la nouvelle étape le point central de recherche va se déplacer vers le point le plus correspondant de l'étape précédente (le (1) en gris). Et la taille du carré de recherche vas se diminue en moitié après chaque étape.
3. A la fin de la recherche la taille de la recherche devient (01) un pixel.

Donc la technique NTSS est un simple algorithme pour l'estimation de mouvement spécialement utilisé pour l'implémentation dû à sa structure régulière. Mais il n'est pas efficace dans l'estimation du mouvement des objets stationnaires ou quasi-stationnaires qui est requis dans la compression vidéo.

III.6.3. Le nouvel algorithme à trois étapes de recherche NTSS

L'algorithme à quatre étapes de recherche (**N**ew **T**hree **S**tep **S**earch en Anglais) est une modification de l'algorithme le bien connu TSS, qui est largement utilisé dans quelques algorithmes de compression vidéo à bas débit binaire. Cependant, comme mentionné dans la section précédente, l'algorithme TSS assignes des points de recherche uniformes dans la première étape. Par conséquent, il n'est efficace dans la détection des petits mouvements apparents dans les blocs stationnaires ou quasi stationnaires. Puisque la plupart des séquences sont lisses et doux, la détection des petits mouvements devient très importante. Une solution à ceci a pu être de rendre la recherche adaptable à l'incertitude et la variation du mouvement. Ceciest réalisé par l'emploi d'un modèle a point de contrôle à centre compensé dans la première étape. L'augmentation de la complexité due à l'addition des points dans la première étape est compensée en ayant une technique d'arrêt à mi-chemin. Ceci fait la complexité de l'algorithme NTSS comparable à celle de l'algorithme TSS.

Les détails sont comme suit [REO94]:

1. en plus des points de contrôle utilisés en TSS, huit points supplémentaires sont ajoutés, qui sont les huit voisins supplémentaires de la fenêtre de recherche (les cercles remplis en noir et les carrés dans la figure III.10).
2. une technique d'arrêt à mi-chemin est employée pour les blocs stationnaires et quasi-stationnaires afin d'identifier rapidement et aussi estimer les mouvements pour ces blocs.

    a. Si le minimum BDM dans la première étape se trouve au centre de fenêtre de recherche, arrêter la recherche (arrêt à la première étape).
    b. Si le point du minimum BDM dans la première étape est l'un des huit voisins du centre de fenêtre, la recherche dans la deuxième étape sera exécutée seulement pour les huit points voisins du minimum et arrêter alors la recherche (arrêt à la deuxième étape). (les

triangles dans la Figure III.10).

Le diagramme bloc montré dans la figure III.9 montre les étapes impliquées dans l'algorithme NTSS.

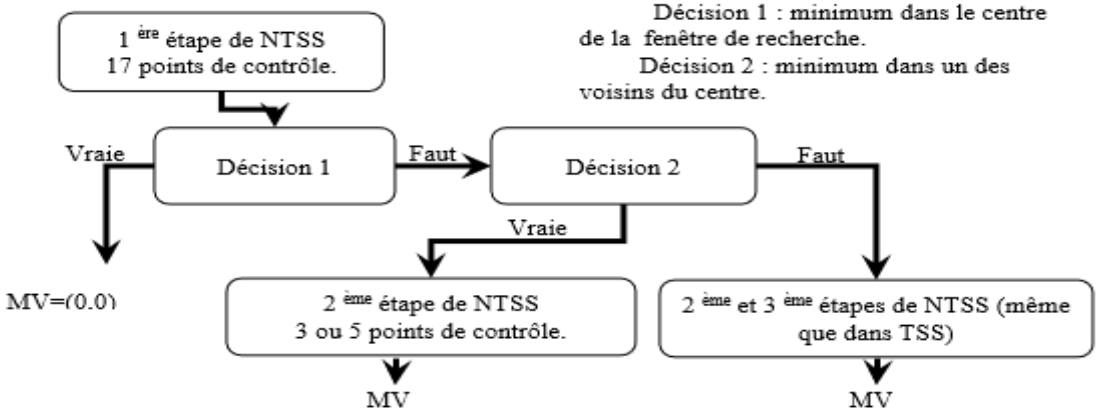


Figure III.9. *Diagramme bloc de l'algorithme NTSS.*

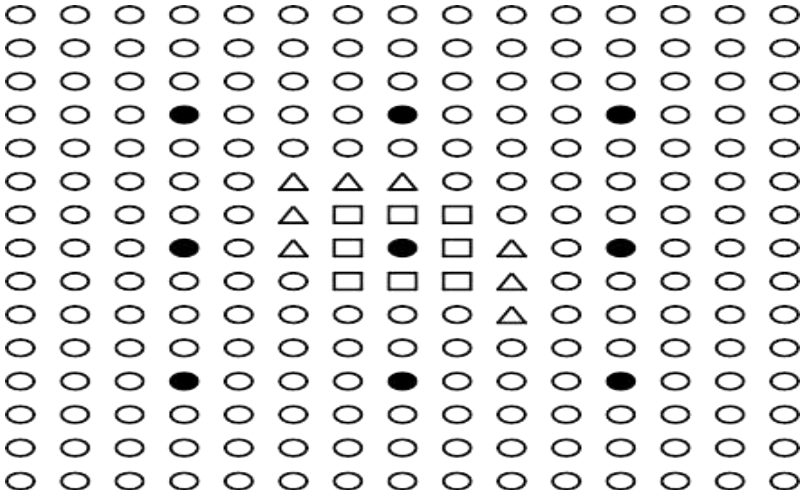


Figure III.10. *Région de recherche de l'algorithme NTSS.*

III.6.4. L'algorithme à quatre étapes de recherche FSS

L'algorithme à quatre étapes de recherche (Four Step Search en Anglais) à une meilleure performance par rapport à celle de TSS et une performance similaire par rapport à celle de NTSS. En plus, FSS réduit le nombre des points de recherche de 33 à 27 points [LAI96].

Comme le NTSS, FSS a également un centre compensé et la technique d'arrêt à mi-chemin. Avec un modèle de recherche de +-7, l'algorithme FSS utilise un modèle de recherche avec neuf points de contrôle sur une fenêtre 5×5 dans la première étape, au lieu d'une fenêtre 9×9 comme dans le TSS. Le centre de la fenêtre est aussi décalé vers le point minimum BDM.

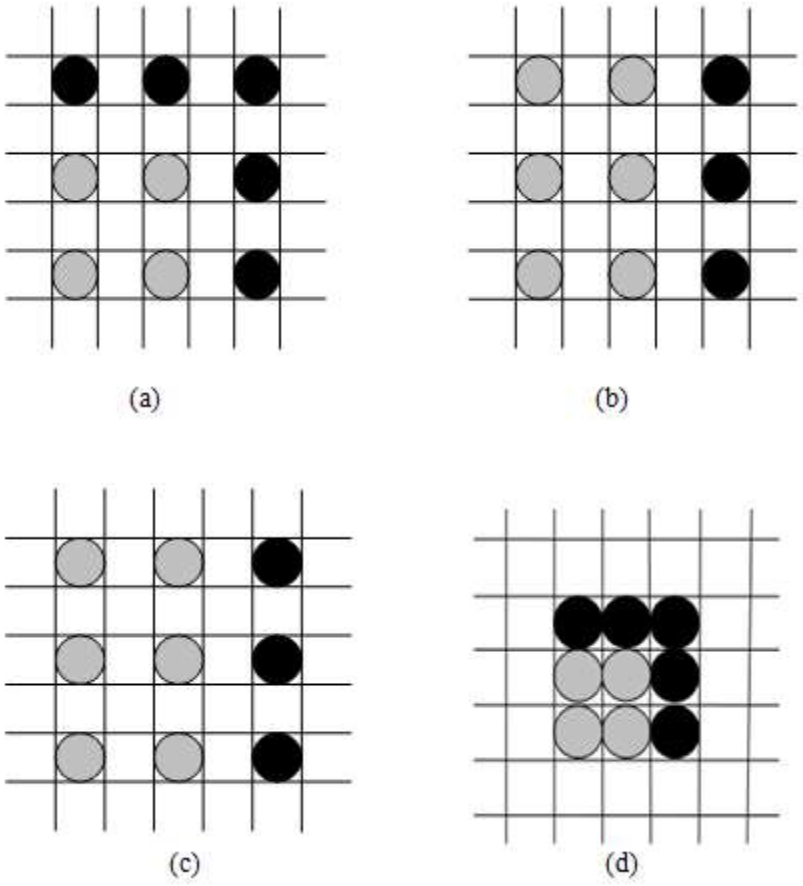


Figure III.11. *Les modèles de recherche :*
(a) Premier étape
(b) Deuxième étape
(c) Troisième étape
(d) *Quatrième étape*

L'algorithme FSS peut être résumé comme suit [WAN98] :

1. Recherche de minimum BDM parmi les neuf points de contrôle sur une fenêtre de 5×5 situé au centre de la fenêtre de recherche, comme mentionné dans la figure III.11 (a). Si le minimum BDM est troé au centre de la fenêtre de recherche, alors aller a l'étape 4 sinon passer à l'étape 2.
2. La taille de fenêtre de recherche est maintenue à 5×5. Cependant le modèle de recherche dépendrade la position du minimum BDM précédent.
   a. Si le point minimum BDM précédent est situé au coin de la fenêtre de recherche précédente, cinq points de contrôle additionnels montrés dans la figure III.11 (b) sont employés.
   b. Si le point minimum BDM précédent est situé au milieu de l'axe horizontal ou vertical de la fenêtre de recherche précédente, trois points de contrôle additionnels comme montrés dans la figure III.11 (c) sont employés.
   c. Si le minimum BDM est trouvé au centre de la fenêtre de recherche, passer à l'étape 4 autrement, passer à l'étape 3.

3. La stratégie du modèle de recherche est identique à celle de l'étape 2, mais passera finalement à l'étape 4.
4. La fenêtre de recherche est réduite à 3×3 comme montré dans Figure 10, et la direction du vecteur global de mouvement est considérée comme le point du minimum BDM parmi ces neuf points de recherche.

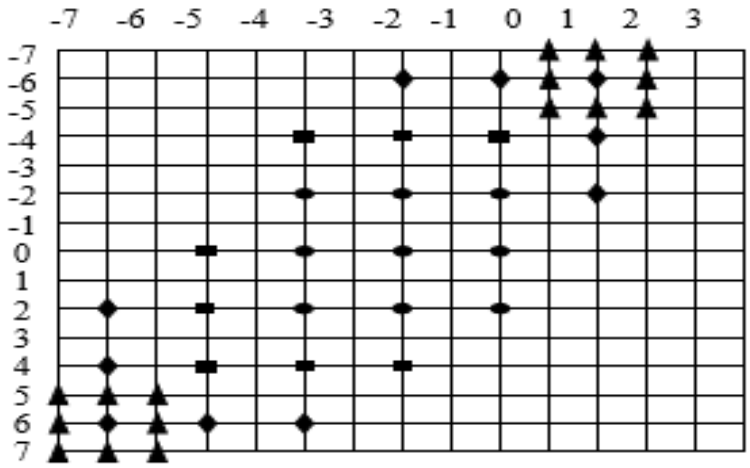


Figure III.12. *Modèle de recherche de l'algorithme FSS*

L'algorithme FSS est plus robuste par rapport aux algorithmes TSS et NTSS. C'est parce que l'utilisation de FSS est maintenue pour des séquences d'image qui contiennent le mouvement complexe de la caméra telle que le zoom et le mouvement rapide. En même temps, la régularité et la simplicité des dispositifs orientés par matériel sont également maintenues.

III.6.5. L'algorithme de recherche en diamant DS

L'algorithme de recherche en diamant (Diamond Search en Anglais) est un algorithme rapide d'estimation de mouvement de correspondance de blocs, basé sur l'étude de la distribution des vecteurs de mouvement de plusieurs séquences vidéo généralement utilisées.

Les résultats de simulation prouvent que l'algorithme DS surpasse l'algorithme de TSS. Comparé avec l'algorithme NTSS, le DS exige 22% moins de calcul par rapport NTSS. La performance de cet algorithme est mieux que FSS en termes de nombre de points de recherche exigés.

Les résultats expérimentaux prouvent que 53% à 98% des vecteurs de mouvement sont joints dansun appui circulaire avec un radium de 2 pixels et portés sur la position du mouvement nul. En outre, le déplacement de bloc des séquences vidéo peut être dans n'importe quelle direction, mais principalement dans des directions horizontales et verticales. Basé sur ces deux observations, les points de recherche encourus dans le cercle avec du radium de 2 pixels sont les plus appropriés à choisir de composer le modèle de recherche. L'algorithme DS est fondé sur l'hypothèse que les vecteurs de mouvement sont en général de centre polarisé.

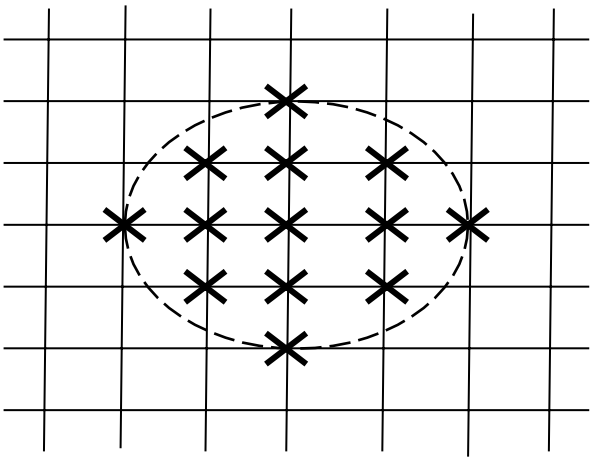


Figure III.13. *Le modèle approprié de recherche, un secteur circulaire et radium de 2 pixels. (Les 13 crois indiquent tous les points de contrôle possibles dans le cercle).*

L'algorithme DS utilise deux modèles de recherche. Un modèle de recherche en grand diamant (**L**arge **D**iamond **S**earch **P**attern en Anglais) de 9 points de recherche comme montré sur la figureIII.14 (a), et un modèle de recherche en petit diamant (**S**mall **D**iamond **S**earch **P**attern en Anglais) de 5 points comme montré sur la Figure III.14 (b)

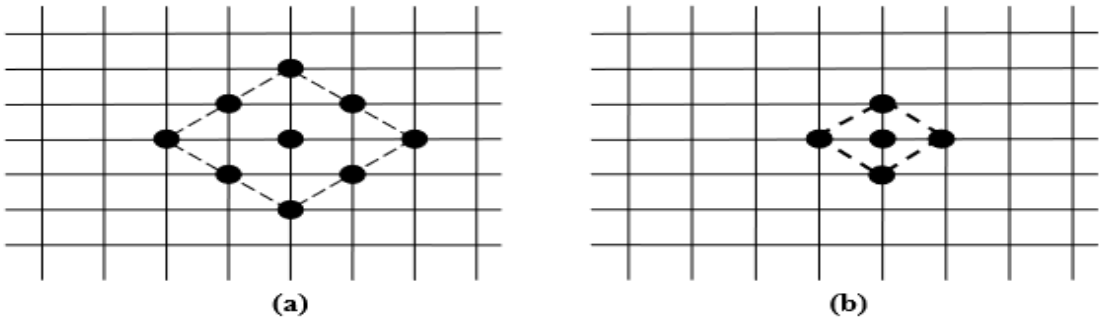


Figure III.14. *Les modèles de recherche :*
(a) *En grand diamant LDSP*
(b) *En petit diamant SDSP*

Le LDSP est toujours employé dans la procédure de recherche jusqu'à l'étape dans laquelle le point minimum de la déformation de bloc (MBD) se produit au centre du diamant. Le modèle de recherche est alors passer à SDSP. Le point rapportant le MBD parmi les 5 points de contrôle dans le SDSP fournit le vecteur de mouvement du bloc le mieux correspondant.

L'algorithme DS peut se récapitulé comme suit [ZHU00]:

1. Le LDSP initial est centré à l'origine de la fenêtre de recherche, et les 9 points de contrôle de LDSP sont examinés. Si le point minimum MBD calculé est situé à la position centrale, passer à l'étape 3; autrement, passer à l'étape 2.
2. Le point minimum MBD trouvé dans l'étape de recherche précédente devient le point central du

nouveau LDSP. Si le nouveau point minimum MBD obtenu est situé à la position centrale, passer à l'étape 3; autrement, répéter périodiquement cette étape.

3. Changer le modèle de recherche de LDSP à SDSP. Le point minimum MBD trouvé dans cette étape est la solution finale du vecteur de mouvement, donc c'est le bloc le plus correspondant.

III.6.6. L'algorithme de recherche minimum MIN

L'algorithme de recherche minimum est un algorithme efficace pour la détection des petits mouvements, car il fait toujours un pas de (1) un pixel, donc pour les petits déplacements, il converge rapidement.

La stratégie de recherche est comme suit [WAT95]:

Il y a cinq points de recherche dans la première étape, un point du centre de la fenêtre de recherche et quatre points voisins (avec une distance de 01 pixel entre chacun des quatre points et le point central) comme indiqué sur la figure 13. Le minimum BDM sera le centre de la fenêtre de recherche suivante.

Dans la deuxième étape, il y a quatre points de recherche, le point central (minimum de l'étape précédente) et trois points voisins, avec l'élimination de point minimum de l'étape précédente.

Les autres étapes sont identiques à l'étape 2, est l'algorithme s'arrête et donne la position debloc le plus correspondant lorsque la position des minimum des deux recherches successives sont les mêmes ou lorsqu'il touche la frontière de la zone de recherche.

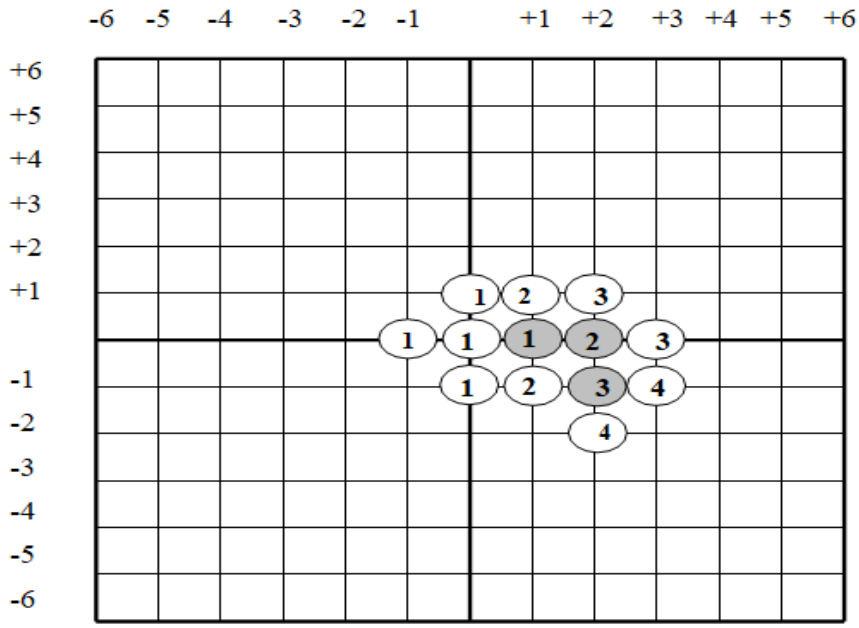


Figure III.15. *Méthode de recherche de l'algorithme minimum.*

III.7. Conclusion

Dans ce chapitre, nous avons expliqué le principe général de la mise en correspondances ‘Block-Matching’ ainsi que les récents algorithmes étudiés. Dans la suite de notre travail nous nous étudions ces algorithmes (FS, FSS, DS, TSS, NTSS et MIN) avec test de bloc de différence.

CHAPITRE IV :
Technique de BM avec test de Bloc de différence

IV.1. Introduction

Dans les applications pratiques d'estimation du mouvement, nous cherchons un compromis entre précision et efficacité : la méthode doit fournir un résultat suffisamment précis dans un temps suffisamment court pour être utilisable. Ce compromis peut être caractérisé par une courbe efficacité-précision, respectivement temps de calcul-erreur obtenue [BAR94], [NAG83].

Dans l'étude qui suit, nous allons présenter les résultats obtenus avec les algorithmes de BM sur des séquences d'images avec test de bloc de différence

Les algorithmes sont mis en œuvre sur la même plate-forme de calcul. Le logiciel choisi pour effectuer les différentes algorithmes est Matlab, principalement pour la facilite de manipulation des matrices (images 2D) et la toolbox très développée Image Processing.

IV.2. Définition de MATLAB [SEL17]

MATLAB est une langue haute performance pour l'informatique technique. Il intègre le calcul, la visualisation et la programmation dans un environnement facile à utiliser où les solutions et les problèmes sont exprimés dans une notation mathématique familière. Les utilisations typiques comprennent:

- Mathématiques et calcul.
- Développement d'algorithme.
- Modélisation, simulation et prototypage.
- Analyse, exploration et visualisation des données.
- Graphiques scientifiques et techniques.
- Développement d'applications, y compris le bâtiment d'interface utilisateur graphique.

IV.3. Application les algorithmes de BM à MATLAB

Après avoir extrait deux frames consécutives d'une séquence vidéo, nous suivons les étapes suivantes afin d'obtenir les vecteurs de mouvement entre ces deux images :

- **Convertir chaque image en valeurs (matrices)**

  Il faut convertir les deux images aux deux matrices

- **Sélection des zones de recherche**

  Nous déterminons l'emplacement où chaque pixel peut être en mouvement. En suivant ces étapes :

  - Diviser la deuxième matrice en blocs de 16×16 (mat16de2)

- Pour chaque bloc, ou pour chaque pixel de ce bloc, nous définissions son propre matrice de recherche à partir de la première matrice (matderecherche)

❖ **Calculer des vecteurs mouvements caractérisant le déplacement des macros bloc**

Les vecteurs de mouvement sont la plus petite différence entre les pixels de la deuxième matrice et la matrice de recherche (pixel par pixel) :

- Calcul la différence entre la deuxième matrice et la matrice de recherche extraite de la première matrice, afin de rechercher les valeurs minimales qui représentent le plus petit changement entre les deux images, en suivant les étapes suivantes :

1. Extraire des matrices (c) de taille 16×16 de chaque matrice de recherche, cette matrice représente un bloc de la matrice de recherche, il transmet pixel par pixel
2. Calculer la valeur absolue (s) de la différence entre la dernière matrice (c) et la matrice (mat16de2). Calculer la somme (sum) qui est représente l'addition entre sum et la valeur s, pour chaque matrice (c)
3. Mettre les valeurs de la somme précédente de chaque bloc dans des vecteurs (vecmin)
4. Nous prenons les valeurs minimales de chaque vecteur (minimum)
5. Chargement des minimums dans un vecteur (vpmin)
6. chargement des numéros des lignes et colonnes des minimums dans un vecteur (vectpositionfinal)

❖ **construction d'une image prédite en utilisant ces vecteurs de mouvements**

Extraire une nouvelle image en utilisant l'image source (image1) et en utilisant les vecteurs de mouvement extraits (vectpositionfinal)

❖ **Comparaison de cette image prédite avec la vraie nouvelle image pour générer des données d'erreurs de prédiction**.

Nous faisons une comparaison entre l'image et sa image reconstruite en utilisant plusieurs critères (calcul la différence entre eux, calcul le PSNR,..,etc)

❖ **Codage et transmission des vecteurs et des données d'erreurs de prédiction.**

Afficher les vecteurs de mouvement entre l'image source (image1) et l'image cible (image2)

IV.4. Résultats des algorithmes de BM par applications directes

Nous avons choisi comme séquence vidéo pour le test; trois séquences connues dans ce domaine de recherche, la première est 'Akiyo', la deuxième est 'Flower garden', la troisième est

'Foreman', la dimesion de ces séquences est 528 pixels × 352 pixels, et voila des tableaux qui nous montre les frames originaux de ces trois séquences et prés d'eux les frames résultantes reconstruites à partir des frames originaux en utilisant les algorithmes étudiées.

Les blocs utilisés dans notre algorithme sont de taille 16 × 16 pixels.

IV.4.1 Résultats de l'algorithme FS

Tableau IV.1. *Résultats de l'algorithme FS sur la séquence 'Akiyo'*

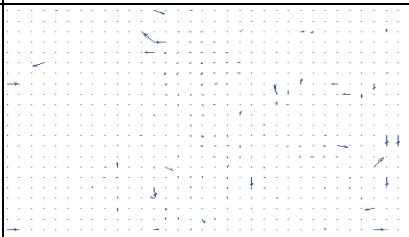
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 25	 Frame n° 25 reconstruite à partirde frame n° 24 reconstruite	 Entre les frames n° 24 et n° 25

Tableau IV.2. *Résultats de l'algorithme FS sur la séquence 'Flower Garden'*

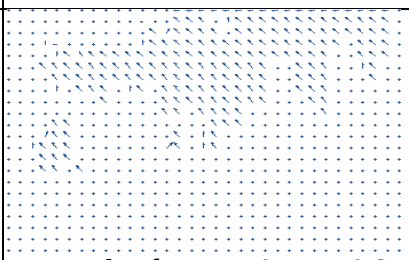
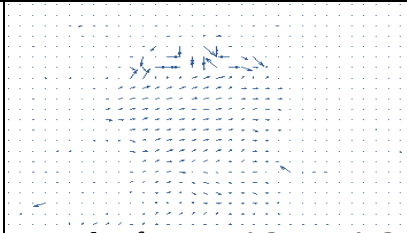
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 6	 Frame n° 6 reconstruite à partir deframe n° 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau IV.3. *Résultats de l'algorithme FS sur la séquence 'Foreman'*

Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 35	 Frame n° 35 reconstruite à partir de frame n° 34 reconstruite	 Entre les frames n° 34 et n° 35

IV.4.2. Résultats de l'algorithme TSS

Tableau IV.4. *Résultats de l'algorithme TSS sur la séquence 'Akiyo'*

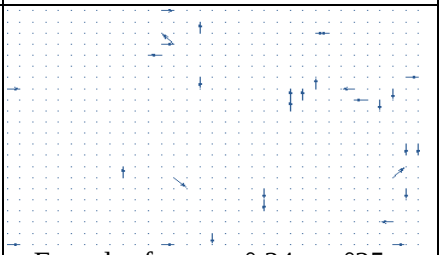
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 25	 Frame n° 25 reconstruite à partir de frame n° 24 reconstruite	 Entre les frames n° 24 et n°25

Tableau IV.5. *Résultats de l'algorithme TSS sur la séquence 'Flower Garden'*

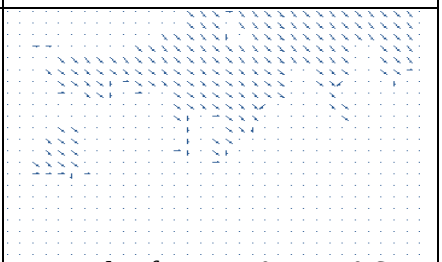
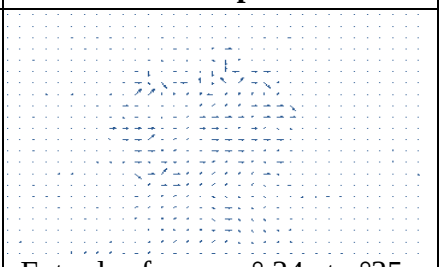
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 6	 Frame n° 6 reconstruite à partir deframe n° 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau IV.6. *Résultats de l'algorithme TSS sur la séquence 'Foreman'*

Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 35	 Frame n° 35 reconstruite à partir de frame n° 34 reconstruite	 Entre les frames n° 34 et n°35

VI. 4.3. Résultats de l'algorithme NTSS

Tableau IV.7. *Résultats de l'algorithme NTSS sur la séquence 'Akiyo'*

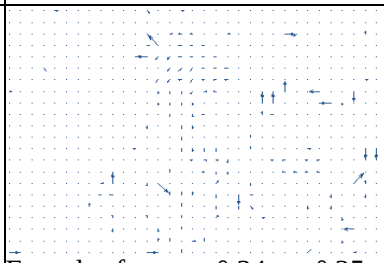
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 25	 Frame n° 25 reconstruite à partir de frame n° 24 reconstruite	 Entre les frames n° 24 et n° 25

Tableau IV.8. *Résultats de l'algorithme NTSS sur la séquence 'Flower Garden'*

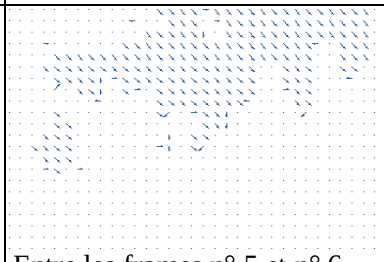
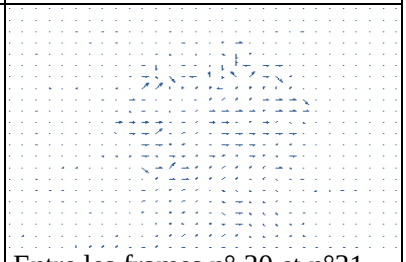
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 6	 Frame n° 6 reconstruite à partir de frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau IV.9. *Résultats de l'algorithme NTSS sur la séquence 'Foreman'*

Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 35	 Frame n° 35 reconstruite à partir de frame n° 34 reconstruite	 Entre les frames n° 30 et n°31

VI. 4.4. Résultats de l'algorithme FSS

Tableau IV.10. *Résultats de l'algorithme FSS sur la séquence 'Akiyo'*

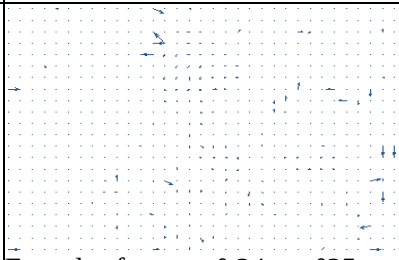
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 25	 Frame n° 25 reconstruite à partir deframe n° 24 reconstruite	 Entre les frames n° 24 et n°25

Tableau IV.11. *Résultats de l'algorithme FSS sur la séquence 'Flower Garden'*

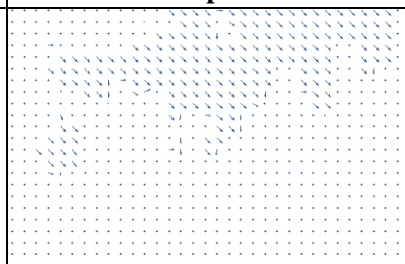
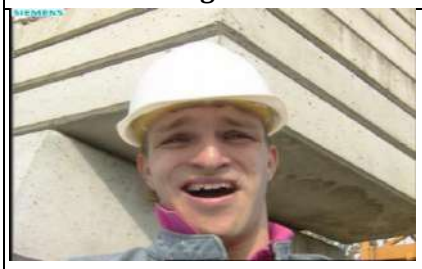
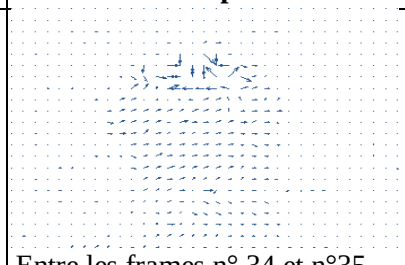
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 6	 Frame n° 6 reconstruite à partir de frame n° 5 reconstruite	 Entre frames n° 5 et n° 6

Tableau IV.12. *Résultats de l'algorithme FSS sur la séquence 'Foreman'*

Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 35	 Frame n° 35 reconstruite à partir deframe n° 34 reconstruite	 Entre les frames n° 34 et n°35

VI. 4.5. Résultats de l'algorithme DS

Tableau IV.13. *Résultats de l'algorithme DS sur la séquence 'Akiyo'*

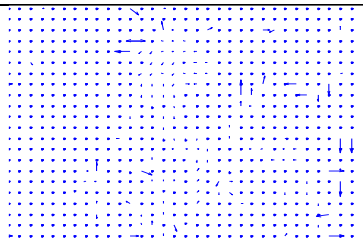
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 25	 Frame n° 25 reconstruite à partir deframe n° 24 reconstruite	 Entre les frames n° 24 et n°25

Tableau IV.14. *Résultats de l'algorithme DS sur la séquence 'Flower Garden'*

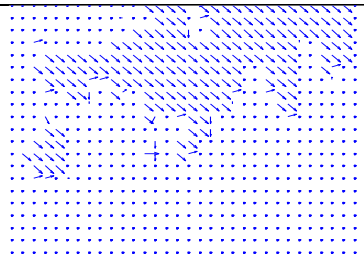
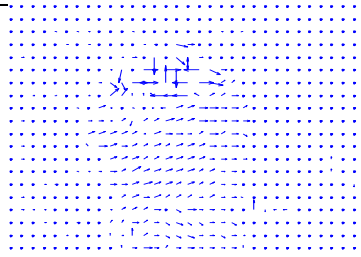
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 6	 Frame n° 6 reconstruite à partir de frame n° 5 reconstruite	 Entre les frames n° 5 et n°6

Tableau IV.15. *Résultats de l'algorithme DS sur la séquence 'Foreman'*

Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 35	 Frame n°34 reconstruite à partir deframe n°35 reconstruite	 Entre les frames n° 34 et n°35

VI. 4.6. Résultats de l'algorithme MIN

Tableau IV.16. *Résultats de l'algorithme DS sur la séquence 'Akiyo'*

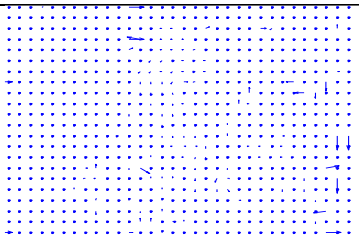
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 25	 Frame n°24 reconstruite à partir deframe n°25 reconstruite	 Entre les frames n° 24 et n°25

Tableau IV.17. *Résultats de l'algorithme DS sur la séquence 'Flower Garden'*

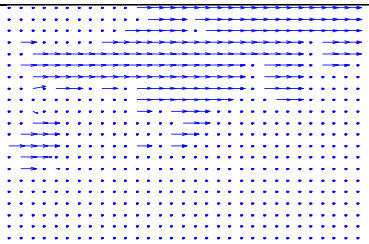
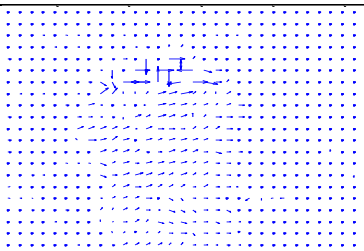
Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 6	 Frame n°5 reconstruite à partir de frame n°6 reconstruite	 Entre les frames n° 5 et n°6

Tableau IV.18. *Résultats de l'algorithme DS sur la séquence 'Foreman'*

Frames originaux	Frames reconstruites	Vecteurs de déplacement
 Frame n° 35	 Frame n°35 reconstruite à partir deframe n°36 reconstruite	 Entre les frames n° 35 et n°36

IV.5. Remarques

Dans ces résultats, nous remarquons que les vecteurs de déplacement ou les soi-disant le flot optique sont différentes entre les séquences vidéo étudiées. Cette différence est due à la compression des données. Dans la séquence 'Akiyo', nous voyons qu'ils forment des régions homogènes assez grandes où les pixels ont des valeurs très proches voire identiques : c'est la redondance spatiale ; Dans la séquence 'Flower Garden', il devient clair pour nous à travers les vecteurs du mouvement qu'il y a un mouvement qui remonte à la légèreté et à la sensibilité des fleurs en raison de la présence de vents. Ce que nous avons remarqué à travers les images représentées dans le mouvement ou la rotation du moulin à air .Ces vecteurs se sont identifiés grâce à la précision réduite des détails des couleurs par rapport aux détails des objectifs qui permettent plus près des caractéristiques de la perception humaine : c'est la redondance psycho visuelle ; Dans la séquence 'Foreman' nous remarquons que la différence entre une image et la suivante est relativement faible, sauf lors d'un changement de plan. Autrement dit, la position d'un bloc de pixels varie généralement peu d'une image à l'autre. Il y a la seule différence dans les parties qui ont été soumises à un mouvement : c'est la redondance temporelle.

IV.6. Evaluation

Afin de connaître la qualité de l'image dans les algorithmes de BM et de connaître la rapidité de ces algorithmes, nous nous basons sur deux critères essentiels :

IV.6.1. PSNR (Peak Signal-to-Noise Ratio)

Le PSNR est le rapport signal sur bruit crête, ce bruit est définit dans l'estimation du mouvement comme MSE (la différence entre les éléments du frame originale et le frame reconstruite)

Ce paramètre de mesure est défini comme suit : [MAN13]

$$PSNR = 10\ \log_{10}\left(\frac{R^2}{\sqrt{MSE}}\right) \quad \text{(IV.1)}$$

Avec :

$$MSE = \left(\frac{1}{N \times M}\right) \times \sum_{N,M}(I_1 - I_2)^2 \quad \text{(IV.2)}$$

Où :

MSE : est l'erreur quadratique moyenne.
I_1 : Sont les éléments du frame original.
I_2 : Sont les éléments du frame reconstruite.
N : La hauteur du frame.
M : La largeur du frame.
R : Valeur maximale de chaque élément du frame ou le seuil est défini à $2^n - 1$ tel que n le

nombre de bit.

Dans l'équation (IV.1), nous remarquons qu'il y a une proportion inverse entre PSNR et MSE, Alors plus MSE est minime, plus le PSNR est important et plus les résultats sont meilleurs.

IV.6.2. Evaluation du Temps de calcul (C.T)

Pour évaluer l'efficacité d'une méthode, nous pouvons utiliser comme critère quantitatif, le temps d'exécution, sur une plate-forme de calcul donnée.

Matlab R2021b de 64-bit (win64) la plate-forme de logiciel est utilisée afin d'exécuter nos algorithmes BM.

Spécification de notre appareil (PC) :

- Processeur : Intel * Core^{TM} i5 4460S CPU 2.9 GHz
- Mémoire RAM installé : 6 GB
- Type du système : système d'exploitation 64 bits, processeur x64
- 1000 Gb HDD

IV.7. Résultats d'évaluation

Nous avons calculé PSNR et C.T de chaque algorithme (FS, FSS, DS, TSS, NTSS et MIN). Le meilleur algorithme est celui qui a le PSNR le plus élevé, est celui qui a le temps de calcul le plus court.

Tableau IV.19. *Résultats PSNR et C.T sur les algorithmes de BM*

algorithmes de recherche	**Séquence 'Akiyo' (frames n°20 et n°21)**		**Séquence 'Flower Garden' (frames n°1 et n°2)**		**Séquence 'Foreman' (frames n°30 et n°31)**	
	C.T	**PSNR**	**C.T**	**PSNR**	**C.T**	**PSNR**
FS	2.37	38.95	2.38	28.83	2.41	33.76
FSS	0.72	39.36	0.84	28.75	0.93	33.02
DS	0.79	39.43	0.83	28.80	0.86	33.06
TSS	0.95	36.98	0.94	22.63	0.91	28.94
NTSS	0.87	39.51	1.00	28.78	0.98	33.29
MIN	0.20	39.51	0.41	28.78	0.35	33.29

IV.8. Discussion des résultats de BMA par application direct

D'après les résultats dans les tableaux (de tableau. IV.1 à tableau IV.19) nous remarquons que l'algorithme DS est le plus performant car il présente le meilleur compromis entre le C.T et le PSNR, ainsi que l'absence des déformations au niveau des frames reconstruites.

Donc, nous pouvons classer les algorithmes selon ses performances:

- ✓ L'algorithme DS est plus performant, PSNR élevé et C.T presque le plus court, et pas se déformation sur les trois séquences.
- ✓ L'algorithme MIN est meilleur selon ses PSNR élevé et son temps de calcul le plus court, mais son inconvénient est les déformations des frames.
- ✓ L'algorithme FS est meilleur selon son PSNR élevé et son C.T est très élevé.
- ✓ L'algorithme FSS a un PSNR élevé et un C.T moyen, et peut de déformations des frames.
- ✓ L'algorithme NTSS a un PSNR moyen et un C.T moyen aussi, et des déformations aux niveaux des frames.
- ✓ L'algorithme TSS a un PSNR le moins important et un C.T le plus long, et il y a trop de déformations aux niveaux de ces des frames résultante.

IV.9. algorithmes de BM avec test du bloc de différence

Plus haut le PSNR, mieux la qualité de l'image reconstruite [MAN13]. Selon notre évaluation, l'algorithme FS est à la première place en termes de PSNR, mais il est à la dernière place en termes de temps de calcul. C'est pourquoi nous avons adopté une condition utilisant trois critères (moyenne, variance et écart type) pour gagner du temps (pour minimiser le temps), et cette condition vise à utiliser un algorithme dans le cas où il y a une grande (spécifique) différence entre les blocs et le contournement l'utilisation de l'algorithme si cette différence est faible (nulle ou quasi inexistante).

IV.9.1. La moyenne (ou la différence absolue MAD)

$$\mu = \left(\frac{1}{B_x \times B_y}\right) \times BD \qquad \text{(IV.3)}$$

Avec :

$$BD = \sum_{i=1}^{B_x} \sum_{j=1}^{B_y} |B_r(i,j) - B_c(i,j)| \qquad \text{(IV.4)}$$

Où BD est le bloc de différence, B_r est bloc de référence, B_c est bloc cible et $B_x \times B_y$ est la taille de bloc, dans notre cas nous avons 16×16.

IV.9.2. La variance

La variance est une valeur numérique qui décrit la variance des notes par rapport à leur moyenne arithmétique.

$$\sigma^2 = \left(\frac{1}{B_x \times B_y}\right) \times \sum_{i=1}^{B_x} \sum_{j=1}^{B_y} (bloc\ de\ différence(i,j) - \mu)^2 \quad \text{(IV.5)}$$

IV.9.3. Ecart type

L'écart type est une mesure de la dispersion des observations au sein d'un ensemble de données.

$$\sigma = \sqrt{\sigma^2} \quad \text{(IV.6)}$$

IV.10. Résultats des algorithmes de BM avec test de bloc de différence

nous utilisons les deux critères (la moyenne et l'écart type) ensemble car la moyenne est une condition nécessaire et non suffisante car il peut y avoir un écart (une différence notable) entre les valeurs de chaque bloc, par exemple deux étudiants obtiennent la même moyenne, par exemple dix, mais leurs scores diffèrent, il y a donc la probabilité qu'une personne dont les scores sont tous proches de dix alors que la seconde a des notes bon et a des notes faibles, et pourtant ils obtiennent la même moyenne, et cette différence entre les notes dans ce cas est appelé variance.

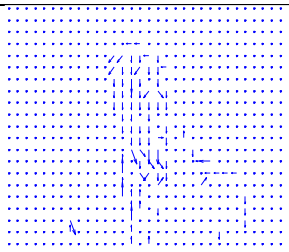
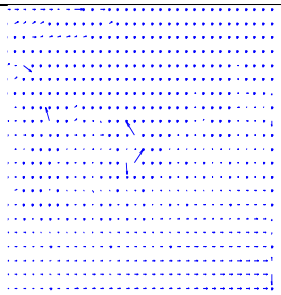
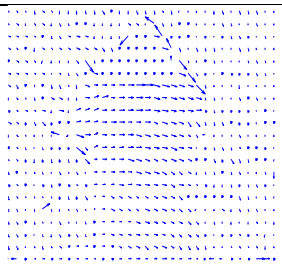
IV.10.1. Résultats de l'algorithme FS avec test de bloc de différence

Tableau IV.20. *PSNR et C.T pour l'algorithme FS avec test de bloc de différence*

(μ, σ)	**Séquence 'Akiyo' (frames n°20 et n°21)**		**Séquence 'Flower Garden' (frames n°1 et n°2)**		**Séquence 'Foreman' (frames n°30 et n°31)**	
	C .T	**PSNR**	**C.T**	**PSNR**	**C.T**	**PSNR**
(0,0)	2.39	38.95	2.40	28.83	2.44	33.76
(0,3)	2.07	38.95	2.07	28.83	2.06	33.76
(3,0)	2.10	38.95	2.08	28.83	2.19	33.76
(3,3)	0.76	39.56	1.86	28.83	2.12	33.74
(6,0)	2.13	38.94	2.14	28.83	2.05	33.76
(6,3)	0.54	39.56	1.57	28.83	1.86	33.73
(9,0)	2.07	38.94	2.09	28.83	2.08	33.76
(9,3)	0.56	39.56	1.25	28.83	1.84	33.73
(12,0)	2.05	38.94	2.06	28.83	2.04	33.76
(12,3)	0.54	39.56	1.25	28.83	1.84	33.73
(12,12)	0.08	38.65	1.06	28.70	0.92	32.10

Pour $\mu = 3$ et $\sigma = 3$

Tableau IV.21. *Résultats de l'algorithme FS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$

	Frames originaux	**Frames reconstruites**	**Vecteur de mouvement**
Séquence 'Akiyo' (frames n°20 et n°21)			
Séquence 'Flower Garden' (frames n°1 et n°2)			
Séquence 'Foreman' (frames n°30 et n°31)			

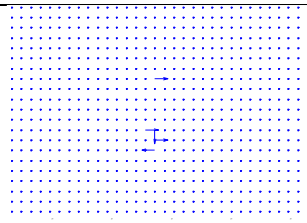
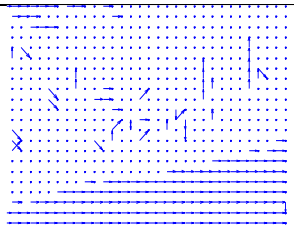
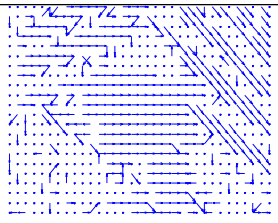
IV.10.2. Résultats de l'algorithme TSS avec test de bloc de différence

Tableau IV.22. *PSNR et C.T pour l'algorithme TSS avec test de bloc de différence*

(μ, σ)	Séquence 'Akiyo' (frames n°20 et n°21)		Séquence 'Flower Garden' (frames n°1 et n°2)		Séquence 'Foreman' (frames n°30 et n°31)	
	C .T	PSNR	C.T	PSNR	C.T	PSNR
(0,0)	0.96	36.95	0.95	22.63	0.93	28.94
(0,3)	0.53	36.95	0.55	22.63	0.50	28.94
(3,0)	0.15	36.95	0.15	22.63	0.15	28.94
(3,3)	0.21	37.02	0.50	22.63	0.64	28.94
(6,0)	0.54	36.95	0.51	22.63	0.48	28.94
(6,3)	0.16	37.02	0.31	22.63	0.42	28.93
(9,0)	0.45	36.95	0.45	22,63	0.46	28.94
(9,3)	0.15	37.02	0.30	22.63	0.44	28.93
(12,0)	0.45	36.95	0.43	22.63	0.45	28.94
(12,3)	0.16	37.02	0.29	22.63	0.33	28.93
(12,12)	0.05	37.01	0.08	22.62	0.21	28.63

Pour $\mu = 3$ et $\sigma = 3$

Tableau IV.23. *Résultats de l'algorithme TSS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$

	Frames originaux	Frames reconstruites	Vecteur de mouvement
Séquence 'Akiyo' (frames n°20 et n°21)			
Séquence 'Flower Garden' (frames n°1 et n°2)			
Séquence 'Foreman' (frames n°30 et n°31)			

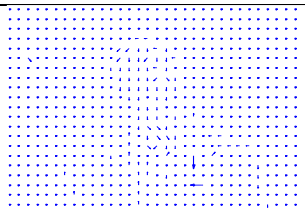
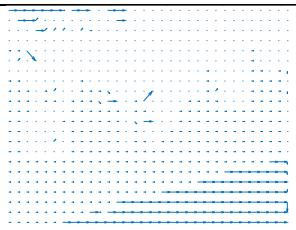
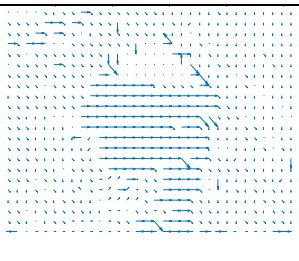
IV.10.3. Résultats de l'algorithme NTSS avec test de bloc de différence

Tableau IV.24. *PSNR et C.T pour l'algorithme NTSS avec test de bloc de différence*

(μ, σ)	Séquence 'Akiyo' (frames n°20 et n°21)		Séquence 'Flower Garden' (frames n°1 et n°2)		Séquence 'Foreman' (frames n°30 et n°31)	
	C .T	PSNR	C.T	PSNR	C.T	PSNR
(0,0)	0.88	39.51	1.02	28.78	1.00	33.29
(0,3)	0.41	39.51	0.51	28.78	0.52	33.29
(3,0)	0.36	39.51	0.48	28.78	0.48	33.29
(3,3)	0.21	39.52	0.54	28.78	0.72	33.28
(6,0)	0.45	39.51	0.51	28.78	0.51	33.29
(6,3)	0.13	39.52	0.33	28.78	0.46	33.28
(9,0)	0.36	39.51	0.47	28.78	0.49	33.29
(9,3)	0.14	39.51	0.31	28.78	0.44	33.28
(12,0)	0.34	39.51	0.45	28.78	0.46	33.29
(12,3)	0.14	39.52	0.30	28.78	0.32	33.28
(12,12)	0.07	38.65	0.25	28.66	0.23	31.65

Pour $\mu = 3$ et $\sigma = 3$

Tableau IV.25. *Résultats de l'algorithme NTSS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$

	Frames originaux	Frames reconstruites	Vecteur de mouvement
Séquence 'Akiyo' (frames n°20 et n°21)			
Séquence 'Flower Garden' (frames n°1 et n°2)			
Séquence 'Foreman' (frames n°30 et n°31)			

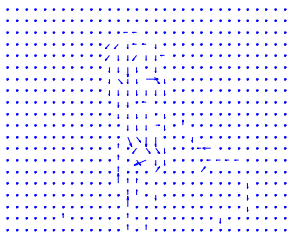
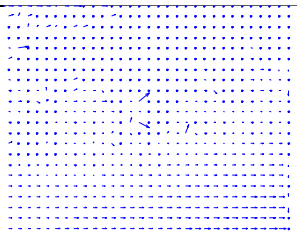
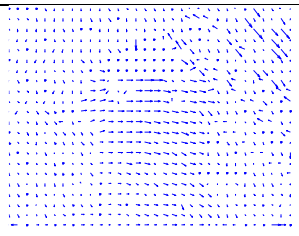
IV.10.4. Résultats de l'algorithme FSS avec test de bloc de différence

Tableau IV.26. *PSNR et C.T pour l'algorithme FSS avec test de bloc de différence*

(μ, σ)	Séquence 'Akiyo' (frames n°20 et n°21)		Séquence 'Flower Garden' (frames n°1 et n°2)		Séquence 'Foreman' (frames n°30 et n°31)	
	C .T	PSNR	C.T	PSNR	C.T	PSNR
(0,0)	0.73	39.36	0.86	28.75	0.95	33.02
(0,3)	0.59	39.36	0.47	28.75	0.48	33.02
(3,0)	0.52	39.36	0.42	28.75	0.42	33.02
(3,3)	0.20	39.43	0.43	28.74	0.65	33.01
(6,0)	0.55	39.36	0.48	28.75	0.46	33.02
(6,3)	0.18	39.43	0.27	28.74	0.41	33.00
(9,0)	0.52	39.36	0.43	28.75	0.42	33.02
(9,3)	0.16	39.43	0.23	28.74	0.62	33.00
(12,0)	0.51	39.36	0.43	28.75	0.37	33.02
(12,3)	0.17	39.43	0.23	28.74	0.40	33.00
(12,12)	0.06	38.56	0.07	28.62	0.37	31.63

Pour $\mu = 3$ et $\sigma = 3$

Tableau IV.27. *Résultats de l'algorithme FSS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$

	Frames originaux	Frames reconstruites	Vecteur de mouvement
Séquence 'Akiyo' (frames n°20 et n°21)			
Séquence 'Flower Garden' (frames n°1 et n°2)			
Séquence 'Foreman' (frames n°30 et n°31)			

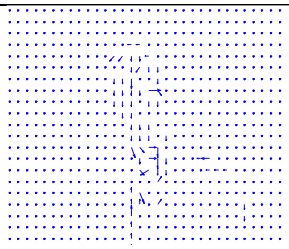
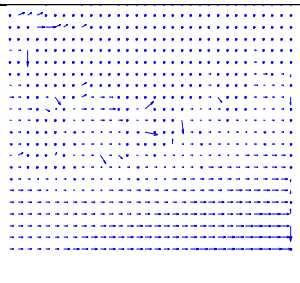
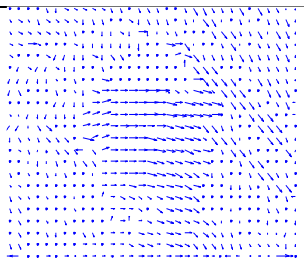
IV.10.5. Résultats de l'algorithme DS avec test de bloc de différence

Tableau IV.28. *PSNR et C.T pour l'algorithme DS avec test de bloc de différence*

(μ, σ)	Séquence 'Akiyo' (frames n°20 et n°21)		Séquence 'Flower Garden' (frames n°1 et n°2)		Séquence 'Foreman' (frames n°30 et n°31)	
	C .T	PSNR	C.T	PSNR	C.T	PSNR
(0,0)	0.80	39.43	0.84	28.80	0.87	33.06
(0,3)	0.45	39.43	0.47	28.80	0.45	33.06
(3,0)	0.14	39.43	0.13	28.80	0.14	33.06
(3,3)	0.20	39.46	0.59	28.80	0.59	33.05
(6,0)	0.46	39.43	0.43	28.80	0.44	33.06
(6,3)	0.14	39.46	0.27	28.80	0.39	33.05
(9,0)	0.42	39.43	0.39	28.80	0.41	33.06
(9,3)	0.14	39.46	0.27	28.80	0.37	33.05
(12,0)	0.40	39.43	0.36	28.80	0.26	33.06
(12,3)	0.13	39.46	0.08	28.80	0.21	33.05
(12,12)	0.05	38.63	0.07	28.62	0.19	31.48

Pour $\mu = 3$ et $\sigma = 3$

Tableau IV.29. *Résultats de l'algorithme DS avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$

	Frames originaux	Frames reconstruites	Vecteur de mouvement
Séquence 'Akiyo' (frames n°20 et n°21)			
Séquence 'Flower Garden' (frames n°1 et n°2)			
Séquence 'Foreman' (frames n°30 et n°31)			

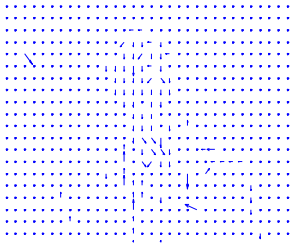
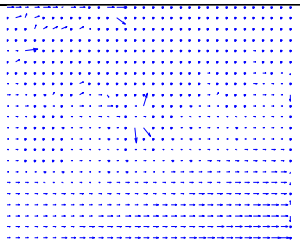
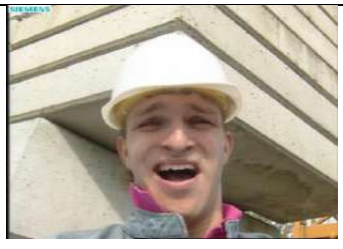
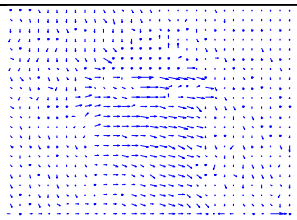
IV.10.6. Résultats de l'algorithme MIN avec test de bloc de différence

Tableau IV.30. *PSNR et C.T pour l'algorithme MIN avec test de bloc de différence*

(μ, σ)	Séquence 'Akiyo' (frames n°20 et n°21)		Séquence 'Flower Garden' (frames n°1 et n°2)		Séquence 'Foreman' (frames n°30 et n°31)	
	C .T	PSNR	C.T	PSNR	C.T	PSNR
(0,0)	0.21	39.51	0.44	28.78	0.39	33.29
(0,3)	0.17	39.51	0.37	28.78	0.27	33.29
(3,0)	0.08	39.51	0.15	28.78	0.09	33.29
(3,3)	0.09	39.52	0.24	28.78	0.31	33.28
(6,0)	0.16	39.51	0.51	28.78	0.09	33.29
(6,6)	0.08	39.52	0.05	28.78	0.12	33.28
(9,0)	0.05	39.51	0.11	28.78	0.10	33.29
(9,9)	0.02	39.52	0.05	28.78	0.08	33.28
(12,0)	0.25	39.51	0.12	28.78	0.09	33.29
(12,3)	0.05	39.52	0.05	28.78	0.09	33.28
(12,12)	0.02	38.58	0.05	28.66	0.05	31.65

Pour $\mu = 3$ et $\sigma = 3$

Tableau IV.31. *Résultats de l'algorithme MIN avec test de bloc de différence pour* $\mu = 3$ et $\sigma = 3$

	Frames originaux	Frames reconstruites	Vecteur de mouvement
Séquence 'Akiyo' (frames n°20 et n°21)			
Séquence 'Flower Garden' (frames n°1 et n°2)			
Séquence 'Foreman' (frames n°30 et n°31)			

IV.11. Comparaison entre les résultats de BM par application direct et les résultats de BM avec test de bloc de différence

Nous représentons les résultats de chaque algorithme BM par application direct et les résultats de chaque algorithme BM avec test de bloc de différence sur même histogramme, pour faire la comparaison entre eux

IV.11.1. Comparaison entre les résultats de FS par application direct et les résultats de FS avec test de bloc de différence

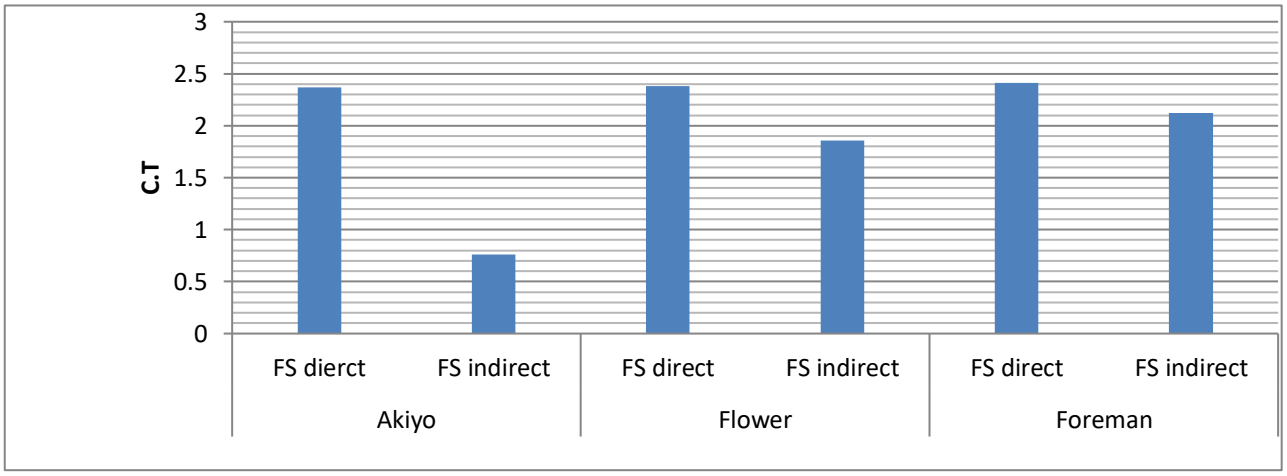


Figure IV.1. *Variation du C.T pour l'algorithme FS par application direct et FS avec test de bloc de différence (FS indirect) pour* $\mu = 3$ et $\sigma = 3$

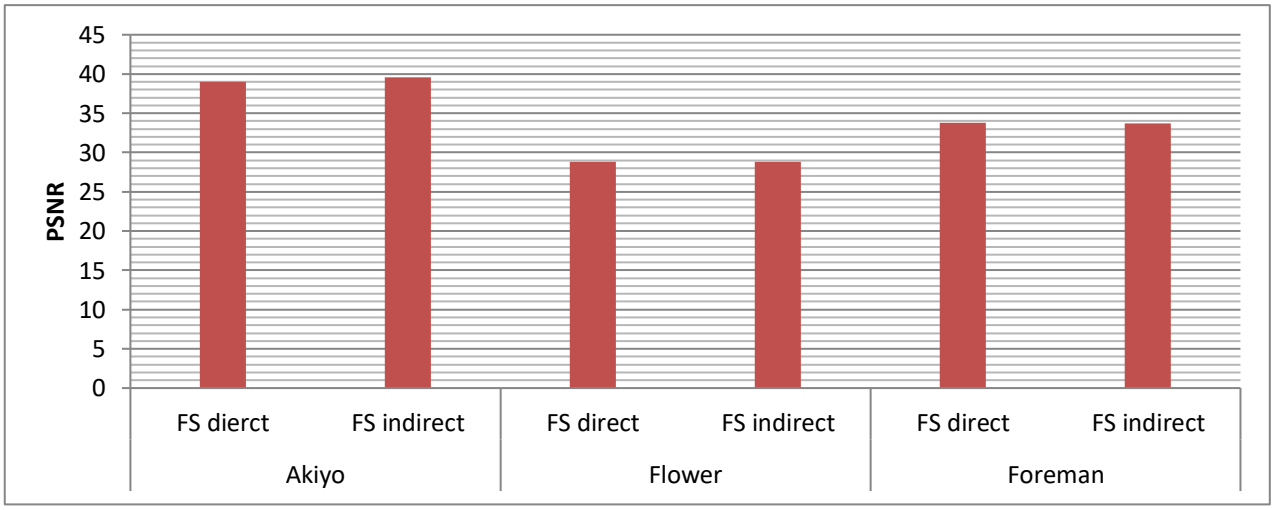


Figure IV.2. *Variation du PSNR pour l'algorithme FS par application direct et FS avec test de bloc de différence (FS indirect) pour* $\mu = 3$ et $\sigma = 3$

IV.11.2. Comparaison entre les résultats de TSS par application direct et les résultats de TSS avec test de bloc de différence

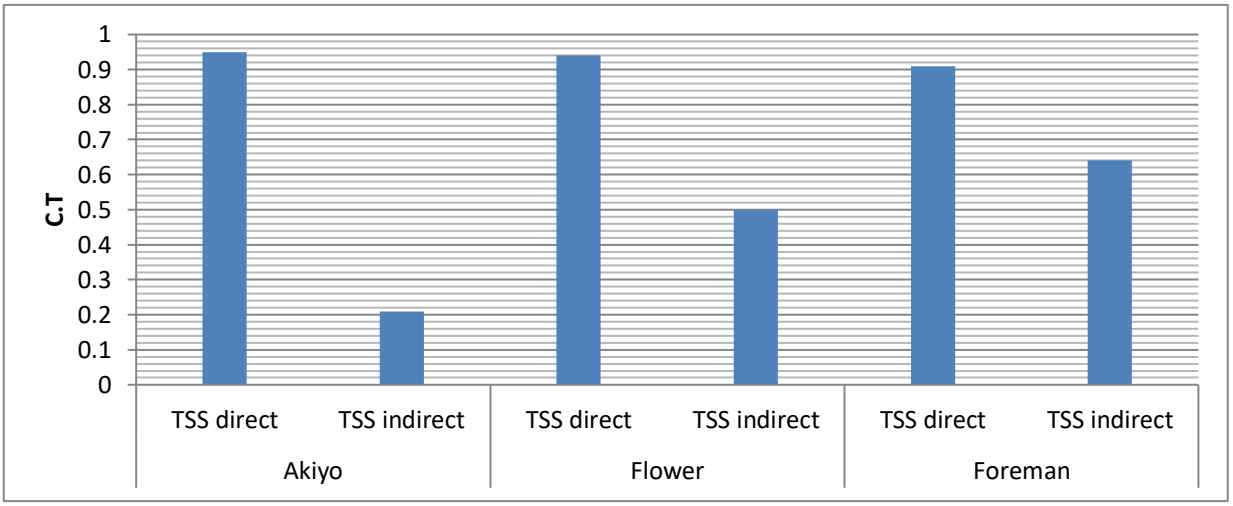


Figure IV.3. *Variation du C.T pour l'algorithme TSS par application direct et TSS avec test de bloc de différence (TSS indirect) pour* $\mu = 3$ et $\sigma = 3$

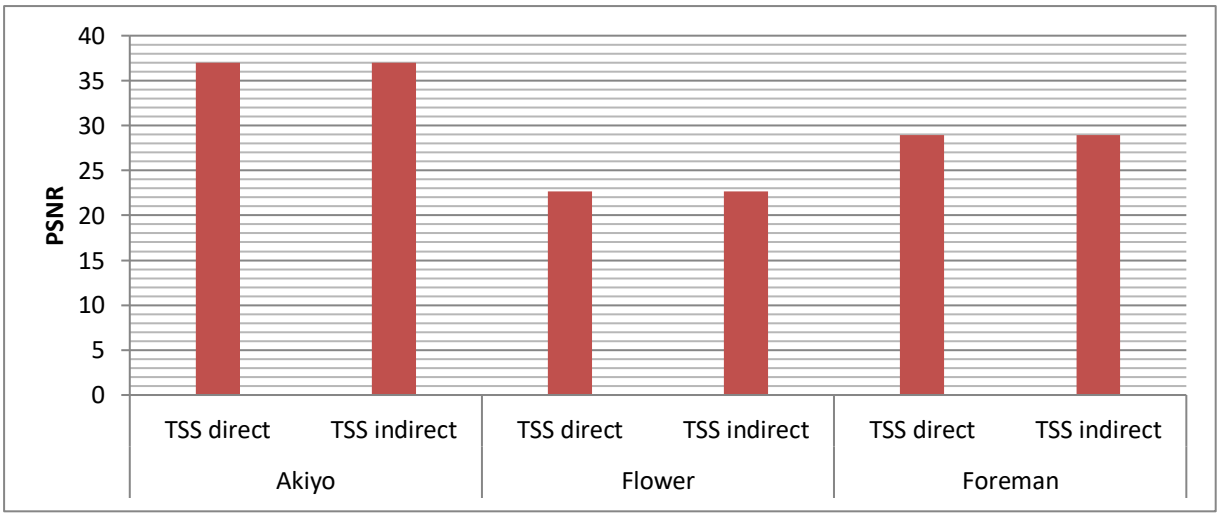


Figure IV.4. *Variation du PSNR pour l'algorithme TSS par application direct et TSS avec test de bloc de différence (TSS indirect) pour* $\mu = 3$ et $\sigma = 3$

IV.11.3. Comparaison entre les résultats de NTSS par application direct et les résultats de NTSS avec test de bloc de différence

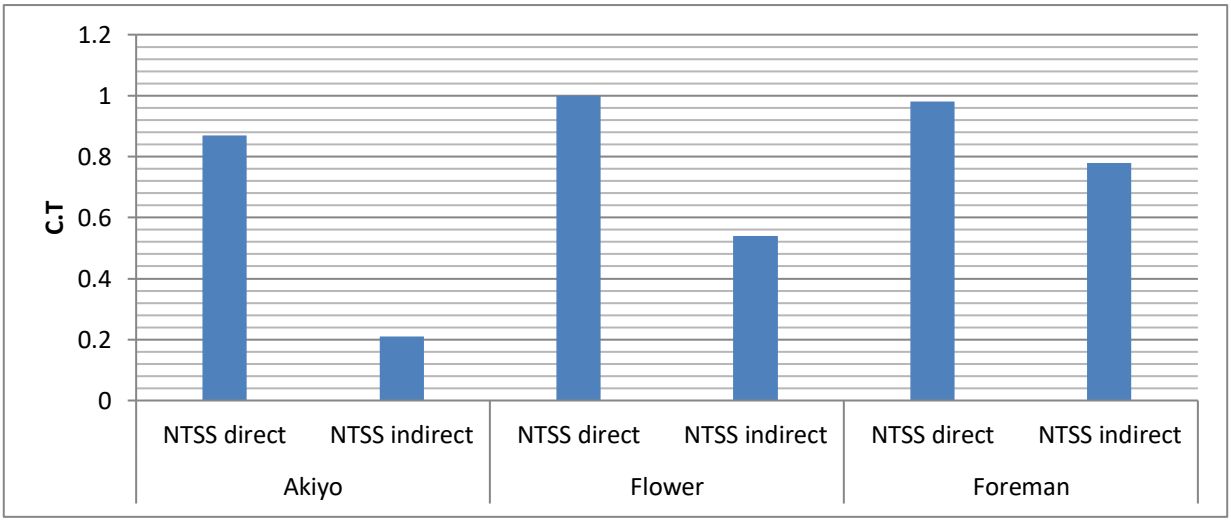


Figure IV.5. *Variation du C.T pour l'algorithme NTSS par application direct et NTSS avec test de bloc de différence (NTSS indirect) pour* $\mu = 3$ et $\sigma = 3$

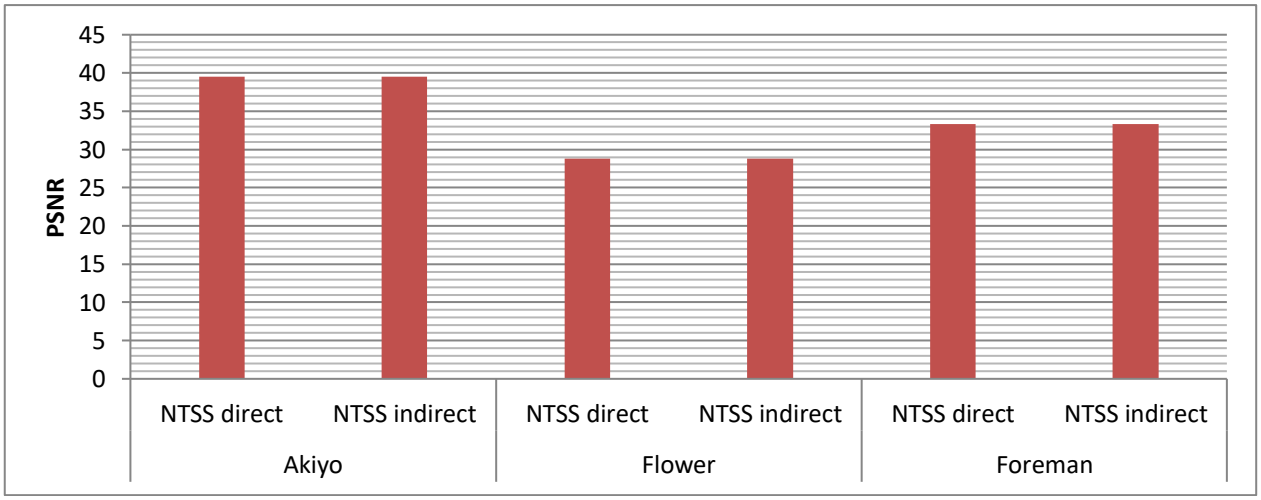


Figure IV.6. *Variation du PSNR pour l'algorithme NTSS par application direct et NTSS avec test de bloc de différence (NTSS indirect) pour* $\mu = 3$ et $\sigma = 3$

IV.11.4. Comparaison entre les résultats de FSS par application direct et les résultats de FSS avec test de bloc de différence

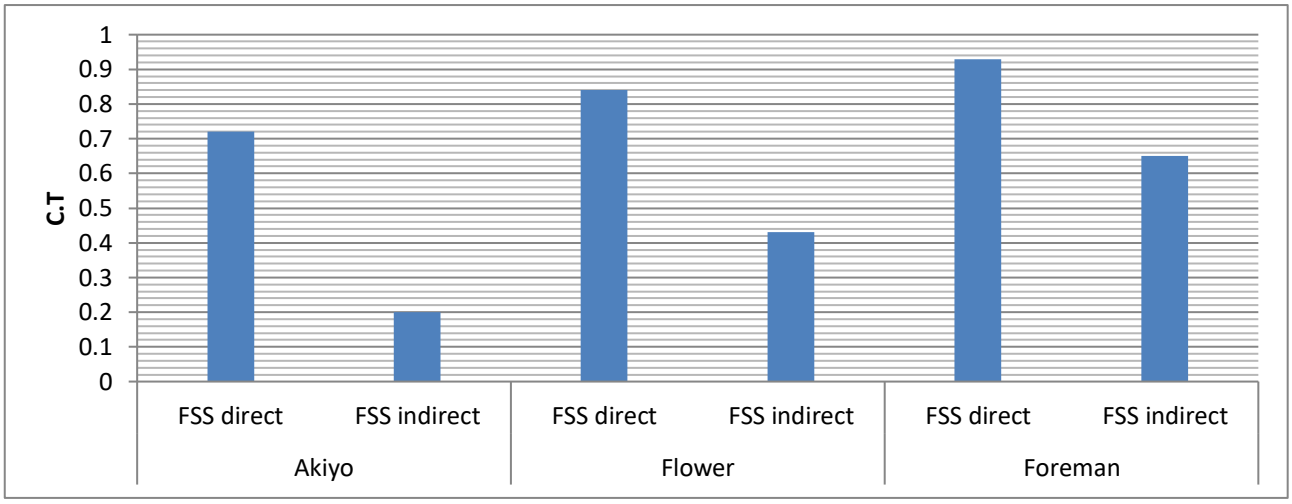


Figure IV.7. *Variation du C.T pour l'algorithme FSS par application direct et FSS avec test de bloc de différence (FSS indirect) pour* $\mu = 3$ et $\sigma = 3$

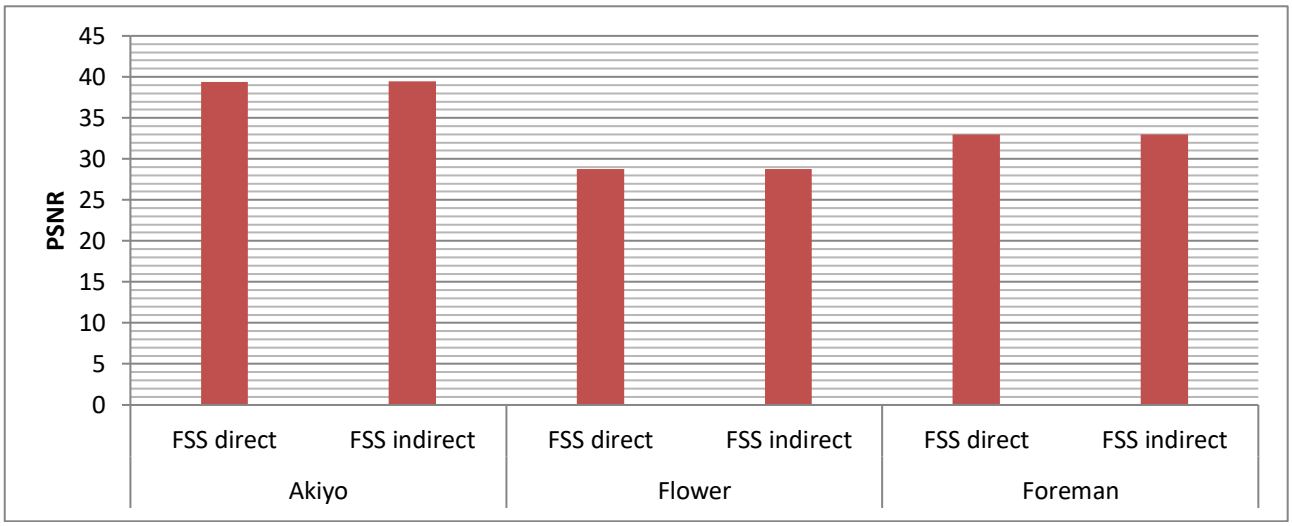


Figure IV.8. *Variation du PSNR pour l'algorithme FSS par application direct et FSS avec test de bloc de différence (FSS indirect) pour* $\mu = 3$ et $\sigma = 3$

IV.11.5. Comparaison entre les résultats de DS par application direct et les résultats de DS avec test de bloc de différence

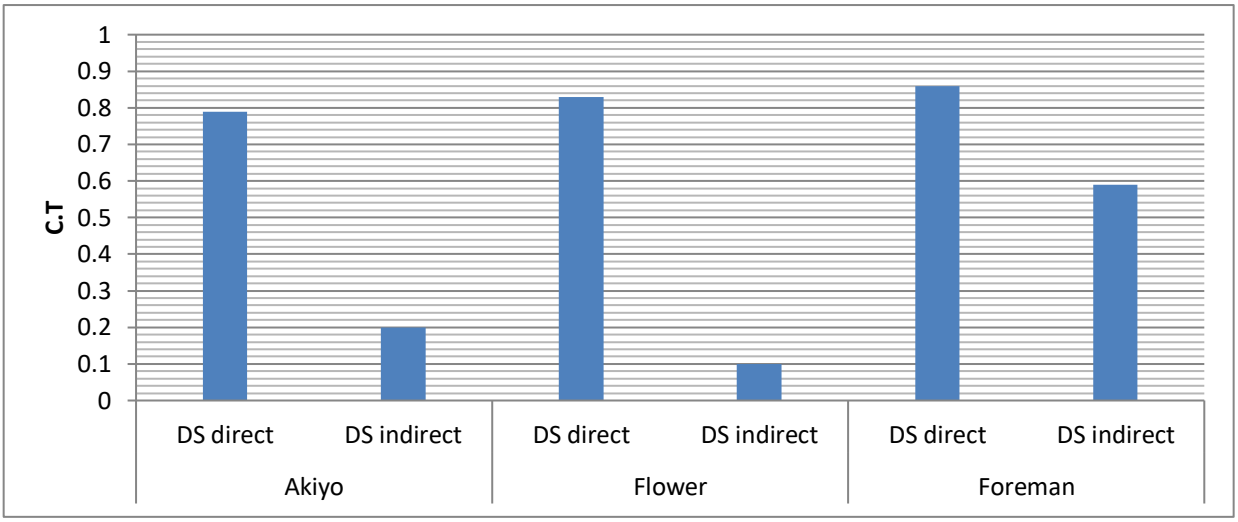


Figure IV.9. *Variation du C.T pour l'algorithme DS par application direct et DS avec test de bloc de différence (DS indirect) pour* $\mu = 3$ et $\sigma = 3$

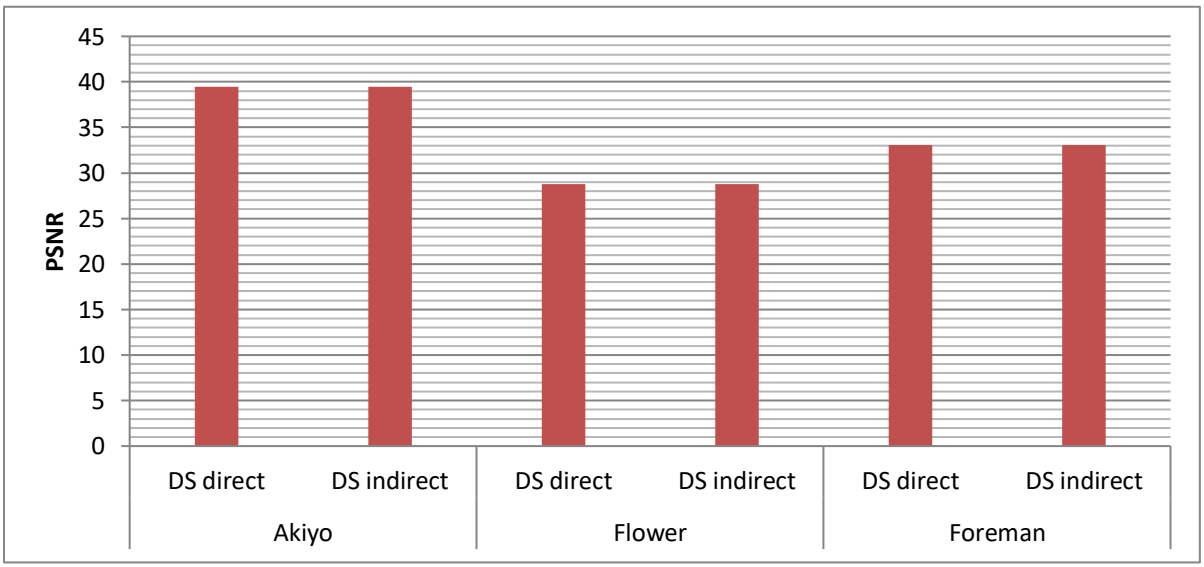


Figure IV.10. *Variation du PSNR pour l'algorithme DS par application direct et DS avec test de bloc de différence (DS indirect) pour* $\mu = 3$ et $\sigma = 3$

IV.11.6. Comparaison entre les résultats de MIN par application direct et les résultats de MIN avec test de bloc de différence

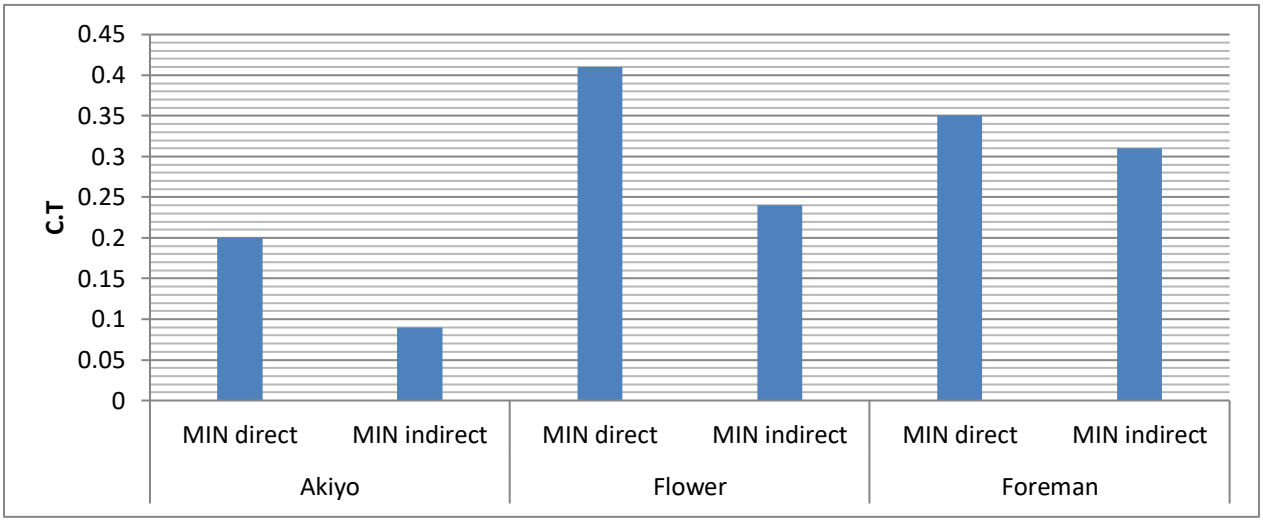


Figure IV.11. *Variation du C.T pour l'algorithme MIN par application direct et MIN avec test de bloc de différence (MIN indirect) pour* $\mu = 3$ et $\sigma = 3$

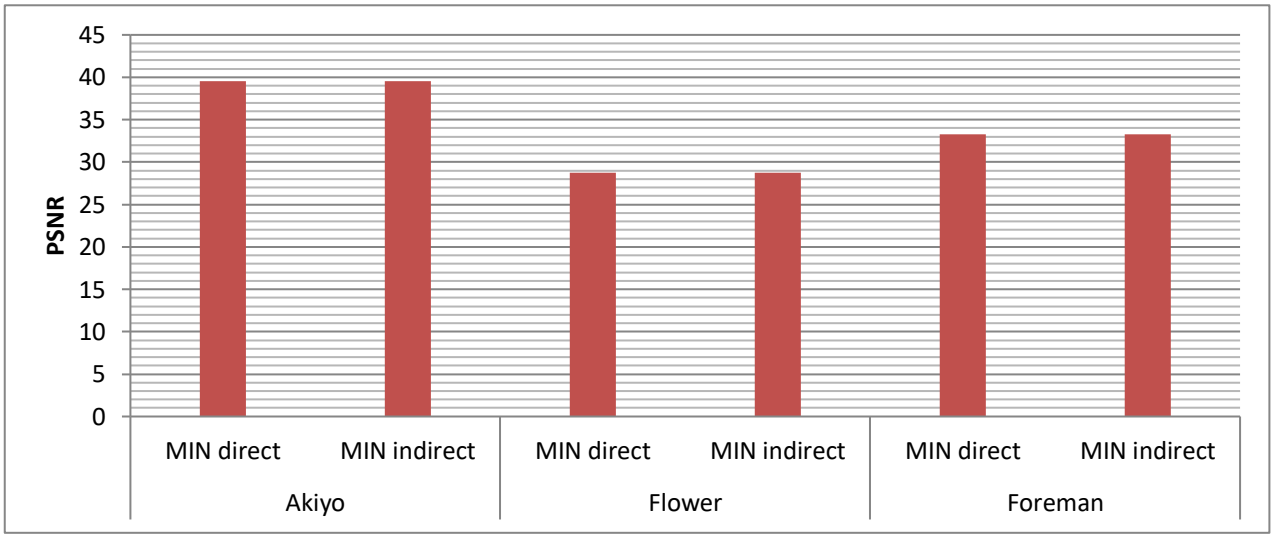


Figure IV.12. *Variation du PSNR pour l'algorithme MIN par application direct et MIN avec test de bloc de différence (MIN indirect) pour* $\mu = 3$ et $\sigma = 3$

IV.12. Discussion des résultats

D'après les résultats dans les histogrammes (de figure IV.1 à figure IV.12) nous remarquons que le C.T diminué dans tous les algorithmes de BM avec test de bloc de différence, et PSNR est resté presque identique avec PSNR de l'algorithme de BM par application direct. Donc nous pouvons dire que notre test est efficace pour tous les algorithmes de BM.

IV.13. Conclusion

Selon l'application d'algorithmes BM avec test de bloc de différence dans l'estimation de mouvement et en utilisant les critères de moyenne et d'écart type, nous avons obtenu des résultats acceptables compte tenu du temps de calcul, du PSNR ainsi que des déformations de l'image. Mais il n'y a pas de valeurs spécifiques pour chacune de la moyenne et de l'écart type pour on fait dire qu'elles nous donnent de bons résultats car chaque image a ses propres valeurs. Par exemple, dans l'algorithme FSS la valeur de temps pour la séquence 'Akiyo' dans (0,3) égale 0.59 et dans (3,3) égale 0.20, pour même des paires dans la séquence 'Flower Garden' le C.T égale 0.47 et 0.43 successivement ; dans la séquence 'Foreman' le C.T égale 0.48 et 0.45. C'est-à-dire que nous remarquons dans la séquence 'Akiyo' et la séquence 'Flower Garden' que le C.T diminue lorsque les valeurs de moyenne et de variance arithmétiques augmentent, alors que nous remarquons le contraire dans la séquence 'Foreman', c'est-à-dire les valeurs qui ont la moyenne et la variance moins que (0.3) sont inférieures aux valeurs qui ont la moyenne et la variance inférieures à (3.3) dans la séquence 'Akiyo' et dans la séquence 'Flower Garden' et vice versa dans la séquence 'Foreman', ce qui signifie que ce changement dépend sur l'image elle-même.

Conclusion générale

Conclusion générale

Ce projet, nous avons permis d'approfondir notre connaissance dans le domaine du traitement d'image et plus précisément dans le domaine estimation du mouvement par méthode de mise en correspondance avec test de bloc de différence.

La rapidité et la qualité de l'estimation de mouvement dans les séquences d'images sont fondamentales dans beaucoup d'applications par exemple la robotique mobile pour construire un représentation de l'environnement d'un robot qui se déplace, la météorologie avec le suivi de masses nuageuses, la médecine avec l'estimation du mouvement d'organes mobiles (comme le cœur ou les poumons), et dans la reconstruction de scènes 3D si des informations supplémentaires sont disponibles sur la nature du mouvement réel.et dans la compression d'images utilisant l'information du mouvement, cette dernière permet de réduire significativement permit de réduire énormément l'espace mémoire nécessaire au stockage d'une image, le but est d'estimer le mouvement des objets contenue dans une vidéo sous forme d'un champ de vecteurs qui s'appelle 'flot optique', c'est-à-dire au lieu de transmettre six frames successives dans une séquence vidéo, il faut transmettre une seule frame et cinq matrice de (33×22) éléments qui représentent les vecteurs de déplacements entre eux, c'est-à-dire $33 \times 22 \times 5 = 3630$ éléments, lorsqu'une frame $528 \times 352 \times 3 = 557568$ éléments on a un taux de compression de $(55768 + 3630)/557568 \times 6 = 16.77\%$.

Dans ce cadre plusieurs algorithmes de recherche pour la mise en correspondance ont été appliqué, pour obtenue des résultats acceptables considérant le temps de calcul, le PSNR ainsi que la déformation des images. Selon des études et notre évaluation, nous avons remarqué que l'algorithme FS par une application directe donne de bons résultats dans la proportion de bruit (PSNR).mais nécessite un temps de calcul énorme, Cela a du sens car cet algorithme est un algorithme de recherche complet en soi.

Nous avons travaillé de développer cet algorithme BM pour modifier (minimiser) le temps de calcul en tenant compte du PSNR, ce qui a donné des résultats acceptables, concernant PSNR, C.T, ainsi que l'absence des déformations au niveau des images reconstruites. Nous avons constaté une amélioration en termes de temps de calcul dans différents algorithmes de BM. Nous notons le pourcentage d'amélioration, dans toutes les séquences vidéo étudiées de 'Akiyo', 'Flower Garden' et 'Foreman' successivement, dans l'algorithme 'FS' comme suit (67.93%, 21.84% et 12.03%) , dans l'algorithme 'TSS' (77.89%, 46.80% et 29.67%), dans l'algorithme 'NTSS' (24.13%, 46% et 26.53%), dans l'algorithme 'FSS' (27.77%, 48.80% et 30.10%), dans l'algorithme 'DS' (74.68%, 28.91%, 31.39%) et dans l'algorithme 'MIN' (55%, 41.46% et 11.42%). Concernant les valeurs du PSNR, elles sont des résultats acceptables et similaire avec tous les algorithmes de BM par application directe, pour des petits valeurs pour chaque de la moyenne et de l'écart type. Plus nous prends des grandes valeurs pour 'μ' et 'σ', plus 'PSNR' diminue. Ce qui est normal car nous allons dépasser quelques valeurs (la différence entre l'image originale et l'image reconstruite) dans cette technique et le PSNR est inversement proportionnelle à cette différence.

Ces résultats restent une application d'algorithmes qui peuvent changer avec le type de séquence vidéo. Et qui peut être améliorés par l'association avec d'autres techniques d'optimisation.

Bibliographie

[ALB13] ALBANE, Sara and BLAIFI, Yasmine. L'estimation du mouvement d'image vidéo par les algorithmes génétiques.70 p. Projet de Fin d'Etude Pour l'Obtention de Diplôme De Master : Systèmes de Télécommunications : Médéa, Université Dr.YAHIA Fares : 2013.

[ALL21] ALLALI, Ilies. L'amélioration de la méthode de block-matching par l'utilisation de l'algorithme d'optimisation par l'essaim de particulaires.76 p. Mémoire Présenté pour l'obtention du Diplôme de Master en Informatique : Image et Vie Artificielle : Biskra, Université Mohamed Khider : 2021.

[BAI06] BAILICH, Mohamed Amine. Etude comparative des algorithmes d'estimation de mouvements dans l'image vidéo par la technique de 'Block Matching'. 91 p. Mémoire de Magister : Télécommunications: Alger, Ecole Nationale Polytechnique : 2006.

[BAR94] Barron, John L., David J. Fleet, and Steven S. Beauchemin. "Performance of optical flow techniques." International journal of computer vision 12.1 (1994): 43-77

[BET19] BETKA, Abir. Estimation de mouvement par les techniques métaheuristiques.149 p. Thèse de Doctorat : Signaux et Communications: Biskra, Université Mohamed Khider : 2019.

[CHE94] Chen, M-J., L-G. Chen, and T-D. Chiueh. "One-dimensional full search motion estimation algorithm for video coding." IEEE transactions on circuits and systems for video technology 4.5 (1994): 504-509.

[DAL13] DALI, Amel. Estimation de mouvement par les méthodes fréquentielles de feux optiques. 73 p. Mémoire de fin d'étude en vue de l'obtention du diplôme Master En Informatique : Imagerie : Oum EL Bouaghi, Université Larbi Ben Mhidi : 2013.

[JIN04] Jing, Xuan, and Lap-Pui Chau. "An efficient three-step search algorithm for block motion estimation." IEEE transactions on multimedia 6.3 (2004): 435-438.

[KER15] KERFA, Djoudi. Estimation de mouvement d'objets dans une séquence d'images. 109 p. Thèse de Doctorat : Signaux et Systèmes Intelligents : Oran, Université Mohamed Boudiaf'des sciences et de la technologie : 2015.

[LAI96] Po, Lai-Man, and Chok-Kwan Cheung. "A new center-biased orthogonal search algorithm for fast block motion estimation." Proceedings of Digital Processing Applications (TENCON'96). Vol. 2. IEEE, 1996.

[LIN97] Lin, Yih-Chuan, and Shen-Chuan Tai. "Fast full-search block-matching algorithm for motion-compensated video compression." IEEE Transactions on communications 45.5 (1997): 527-531.

[MAN13] Manuel de Matlab 2013 - Wavelet Toolbox

[MIC97] MICROSOFT,'Encyclopédie ENCARTA', 1997.

[NAG83] Nagel, Hans-Hellmut. "Displacement vectors derived from second-order intensity variations in image sequences." Computer Vision, Graphics, and Image Processing 21.1 (1983): 85-117

[RCG77] R.C.GONZALES P.WINTZ, Digital Image processing'-Addison Wessley-1977

[RED04] Reddy, V. S. K., and Somnath Sengupta. "A new predictive full-search block motion estimation." Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.. Vol. 4. IEEE, 2004.

[REO94] Li, Reoxiang, Bing Zeng, and Ming L. Liou. "A new three-step search algorithm for block motion estimation." IEEE transactions on circuits and systems for video technology 4.4 (1994): 438-442.

[SEL17] SELMI, Mohammed El-Amin and YALA, Mohammed. Synthèse et étude comparative sur les méthodes de compression vidéo. 44 p. Projet de Fin d'Etude Pour l'Obtention de Diplôme De Master : Informatique: Telmcen, Université Abou Baker Belkaid : 2017.

[WAN98] Wang, Kuan-Tsang, and OT-C. Chen. "Motion estimation using an efficient four-step search method." 1998 IEEE International Symposium on Circuits and Systems (ISCAS). Vol. 4. IEEE, 1998.

[WAT95] Watkinson, John. La réduction de débit en audio et vidéo: JPEG, MPEG1, MPEG2, redondance, entropie, inter, intra. Éditions Eyrolles, 1995

[ZHU00] Zhu, Shan, and Kai-Kuang Ma. "A new diamond search algorithm for fast block-matching motion estimation." IEEE transactions on Image Processing 9.2 (2000): 287-290.