République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Dr. YAHIA Fares de Médéa



Faculté des Sciences et de la Technologie

Département de Génie Electrique et de l'Informatique

Projet de Fin d'Etude Pour l'Obtention de Diplôme

De Master en Systèmes de Télécommunications

L'estimation du mouvement d'image vidéo par les algorithmes génétiques

Proposé et dirigé par:

M. BAILICHE Mohamed Amine.

Présenté par :

-ALBANE Sara

-BLAIFI Yasmine.

Année Universitaire 2012-2013

Remerciements

Nous tenons tout d'abord à remercier ***ALLAH*** *le tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce Modeste travail.*

En second lieu, nous tenons à remercier tout le corps enseignant qui a contribué à notre formation, plus particulièrement notre encadreur ***M. BAILICHE Mohamed Amine*** *pour son aide et les précieux conseils qu'il a bien voulu nous donner.*

Nos remerciements s'étendent également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail.

On tient à remercier ***M. LEBOUKH Aboubakr (dhaif allah)*** *pour son contribution à l'arrangement de ce mémoire, et M .****Baba Ali Riad*** *pour leur soutien moral et leur information.*

Dédicaces

Tout d'abord je tiens à remercier ***ALLAH*** *le tout puissant de m'avoir donné la santé, la volonté et le courage.*
Ensuite, à mes ***chers parents*** *pour leur soutien et leurs encouragements et qu'aucune dédicace ne pourra compenser leur sacrifice pour mon bien être et mon bonheur.*

A mes chères sœurs ***Fella, Kenza*** *et la chouchoute* ***Kawther*** *pour lequel j'ai beaucoup d'estimes.*

A mon frère ***Monsor*** *que je lui souhaite une longue vie pleine de joie et de réussite.*

A mes oncles, mes tantes et toute ma famille.

A mes amis, en particulier ma collègue du présent projet ***ALBANE Sara*** *qui n'a jamais manqué de travailler et ceux de la promotion de la 2ème année master systèmes de télécommunications.*

A mes amies ***Meriem, Habiba, Amina, Zineb, Rachida.***

A notre encadreur ***M. BALICHE Mohamed Amine.***

Et à tous ceux qui nous ont soutenus pour l'achèvement de ce projet avec réussite.
Je dédie ce modeste travail.

BLAIFI Yasmine



Dédicaces

*Merci **ALLAH** de m'avoir donné la capacité d'écrire et de réfléchir, la force d'y croire, la patience d'aller jusqu'au bout du rêve et le bonheur de lever mes mains vers le ciel et de dire "**Ya Kayoum**"*

Je dédie ce modeste travail à celle qui m'a donné la vie, le symbole de tendresse, qui s'est sacrifiée pour mon bonheur et ma réussite,

***A ma très chère mère** …*

***A mon très cher père**, école de mon enfance, qui a veillé tout au long de ma vie à m'encourager, à me donner l'aide.*

*Que **ALLAH** les gardes et les protège.*

*A mon adorable sœur **Amira.***

*A mes frères **Hassen ,Abdenour,Yassine** et **Billel** .*

*A mes amis, en particulier mon binôme de ce projet **M. BLAIFI Yasmine**.*

*A notre encadreur **M. BAILICHE Mohamed Amine**.*

A tous ceux qui me sont chères.

A tous ceux qui m'aiment et ceux que j'aime.

Je dédie ce modeste travail.

Albane Sara

الملخص:

في هذه الدراسة نحن مهتمون بتقدير حركة صور الفيديو التي هي جزء أساسي و مهم من مرحلة ضغط صور الفيديو، ولهذا السبب أردنا التوسع في استخدام الخوارزميات الجينية, و نحن بصدد تطوير خوارزمية على أساس التغيير الذي هو ظاهرة طبيعية والتي تقوم على تطور عدد السكان من الأفراد في بيئة معينة.

الكلمات المفتاحية: تقدير الحركة - ضغط الفيديو- مطابقة الكتل – الخوارزميات الجينية.

Résumé :

Dans cette étude, nous nous sommes intéressés à l'estimation de movement d'une image vidéo, qui est une partie essentielle et importante en compression des images vidéo, et pour cette raison on a voulu de la developer par l'utilisation des algorithms génétiques. Nous avons développé un algorithme basé sur l'évolution qui est un phénomène naturel qui s'inspire de l'évolution d'une population d'individus dans un milieu donné.

Mots clé: Estimation de mouvement - Compression video – Mise en correspondance des blocs -Les algorithms génétiques-

Abstract:

In this study, we are interested in the motion estimation of a video, which is an important part of video compression, and for this reason we want to develop it by the use of the genetic algorithms. We developed an algorithm based on the evolution which is a natural phenomenon which takes as a starting point the evolution of a population of individuals in a given medium.

Key words : motion estimation- video compression-block matching-genetic algorithm

TABLE DES MATIÈRES

Remerciement
Dédicaces
Liste des tableaux
Liste des figures
Table des matières
Introduction générale…….…...01

Chapitre 1 : Généralité sur le traitement d'image

1.1. Introduction……..….02
1.2. Notion de base……...02
1.2.1. Définition de l'image……..…....02
1.2.2. L'image numérique …….……..02
1.2.3. Caractéristique d'une image numérique…….. ……..….03
1.2.4. Qualité de l'image numérique ……...06
1.2.5. les formes de l'image ……07
1.3. Etapes de Traitement d'image …….....08
1.3.1. Définition du traitement d'image ……..…....08
1.3.2. Les systèmes de traitement d'images ……..…..….08
1.3.3. Avantages et inconvénients du traitement d'images …….....10
1.4. Compression vidéo ……..…..…...11
1.4.1. Définition de la vidéo …….……...11
1.4.2.Définition de la compression vidéo ……...11
1.4.3. Principes de base ……..….11
1.4.4. Classification des méthodes de compression…….....13
1.4.5. Les types de compression (avec et sans perte) …….…14
1.4.6.Les normes de compression vidéo (historique)…….…..14
1.7. Conclusion ……..…..18

Table des matières

Chapitre 2 : les algorithmes d'estimation d'image vidéo

2.1.Introduction ……..…..19

2.2.Mouvement réel, mouvement apparent et mouvement estimé ……..……...19

2.3. Estimation de mouvement ……...…....21

2.4. Techniques d'estimation de mouvement ……..…..…..24

2.4.1. Block-matching: mise en correspondance de blocs……...24

2.4.2. Les algorithmes d'estimation de mouvement …….....27

2.4.3.1. L'algorithme "Full search" FS …….……...…..27

2.4.3.2. L'algorithme à trois étapes de recherche TSS : …….…….…...28

2.4.3.3. Le nouvel algorithme à trois étapes de recherche NTSS ……..….....30

2.4.3.4. L'algorithme à quatre étapes de recherche FSS …….…….....31

2.4.3.5. L'algorithme de recherche en diamant DS …….....34

2.4.3.6. L'algorithme de recherche minimum MIN…….…....35

2.6. Conclusion …….....36

Chapitre 3 : L'estimation d'image vidéo par les algorithmes génétiques

3.1. Introduction : …….……..37

3.2. Les algorithmes génétiques …….…….....37

3.2.1. Principe …….…….....37

3.2.2. Le codage …….…….....38

3.2.3. La fonction d'adaptation ……..…..…...39

3.2.4. Les opérateurs de reproduction …….....39

3.3. Application des algorithmes génétiques en estimation de mouvement ……....42

3.3.1. Mise en œuvre les algorithmes ……..42

3.3.2. L'organigramme de l'algorithme génétique …….43

3.4. Resultats pratiques ……..……...44

Table des matières

3.5. L’analyse des résultats……..59

3.6. Quelques résultats des algorithmes précédents (FS, TSS, NTSS, FSS, DS , MinS) …….……63

3.7. Comparaison entre les résultats des algorithmes classiques et celle de l’algorithme génétique ……..…68

3.9. Conclusion ……....69

Conclusion générale…….71

Annexe

Bibliographie

LISTE DES FIGURES

Figure I.1. Pixel…….....page03

FigureI.2. Comparaison entre la forme d'image vectorielle et l'mage bitmap…....page08

Figure I.3. Architecture d'un système de traitement d'image……...page09

Figure I.4. Codage spatial ……..……page12

Figure I.5. Schéma d'un codeur source. ……...……page13

Figure I.6 : principe des compressions JPEG et MPEG……page18

FigureII.1. Illustration des mouvements réel et apparent, dans un système optique de prise de vues. …….....page20

Figure II.2. Estimation directe et inverse des vecteurs déplacements. ….....page23

Figure II.3. Illustration d'un bloc de taille 8×8 pixels. …….…...page25

Figure II.4. Stratégies de balayage de la fenêtre de recherche. ……page26

Figure II.5. Modèle de recherche "Full Search" FS. ……..…..page28

Figure II.6. Modèle de recherche à trois étapes TSS. ……..page29

Figure II.7. Région de recherche de l'algorithme NTSS. ……..……..page31

Figure II.8. Diagramme bloc de l'algorithme NTSS. ……....…..page31

FigureII.9. (a). Modèle de recherche 'première étape'. ……....page32

FigureII.9.(b). Modèle de recherche 'deuxième/troisième étape'…….. ….….page32

FigureII.9.(c). Modèle de recherche 'deuxième/troisième étape'…… …....page32

FigureII.9.(d). Modèle de recherche 'quatrième étape'. …….…....page32

FigureII.10. Modèle de recherche de l'algorithme FSS. ……..…...page33

FigureII.11. Diamond Search Algorithme……....page34

FigureII.12. a. Large DiamondSearch Pattern…….....page35

FigureII.12. b. Small DiamondSearch Pattern. ……...…….page35

Figure II.13. Méthode de recherche de l'algorithme minimum……..…..page36

FigureIII.1. Schéma du principe des algorithmes génétiques……..page38

Figure III.2.croisement avec un point de crossover…….....page40

FigureIII.3. croisement avec 2 points de crossover……..page41

FigureIII.4. Croisement uniforme…….page41

Figure III.5. La mutation……...page41

Figure III.6. L'organigramme de l'algorithme génétique …….…..page43

Tableau I.1 : différents standards de compression.....page17

Tableau III.1. application des algorithmes genetique en estimation de mouvement…….page42

Tableau III.2 : résultat sur la séquence ‘Akiyo’pour le recherche entre 50 blocs..... page44

TableauIII.3 : résultat sur la séquence ‘flower’pour le recherche entre 50 blocs..... page45

Tableau III.4: résultat sur la séquence ‘formen’pour le recherche entre 50 blocs..... page46

Tableau III.5 : résultat sur la séquence ‘Akiyo’pour le recherche entre 75blocs…..... page47

Tableau III.6 : résultat sur la séquence ‘flower’pour le recherche entre 75blocs…..... page48

Tableau III.7 : résultat sur la séquence ‘Formen’pour le recherche entre 75blocs…..... page49

Tableau III.8 : résultat sur la séquence ‘Akiyo’pour le recherche entre 100blocs..... page50

Tableau III.9 : résultat sur la séquence ‘flower’pour le recherche entre 100blocs.....… page51

Tableau III.10 : résultat sur la séquence ‘Formen’pour le recherche entre 100bloc.....… page52

Tableau III.11 : résultat sur la séquence ‘Akiyo’pour le recherche entre 125blocs…..... page53

Tableau III.12 : résultat sur la séquence ‘flower’pour le recherche entre 125blocs…..... page54

Tableau III.13 : résultat sur la séquence ‘Formen’pour le recherche entre 125blocs…..... page55

Tableau III.14 : résultat sur la séquence ‘Akiyo’pour le recherche entre 150blocs…..... page56

Tableau III.15 : résultat sur la séquence ‘flower’pour le recherche entre 150blocs..... page57

Tableau III.16 : résultat sur la séquence ‘Formen’pour le recherche entre 150blocs..... page58

Tableau III.17: temps de calcule et PSNR pour la sequence ‘Akiyo’..... page60

Tableau III.18: temps de calcule et PSNR pour la sequence ‘flower’..... page61

Tableau III.19 : temps de calcule et PSNR pour la séquence ‘Formen’..... .page62

Tableau III.20: résultats de l’algorithme FS sur toutes les séquences……..…..page64

Tableau III.21 : résultats de l’algorithme TSS sur toutes les séquences…….. .page64

Tableau III.22 : résultats de l’algorithme NTSS sur toutes les séquences…….…page65

Tableau III.23: résultats de l’algorithme FSS sur toutes les séquences……..page66

Tableau III.24: résultats de l’algorithme DS sur toutes les séquences……page66

Tableau III.25: : résultats de l’algorithme MinS sur toutes les séquences……..page67

TableauIII.26 : Résultats des algorithmes sur la séquence ‘Akiyo’ frame 25…….……....page67

Tableau III.27: Résultats de algorithmes sur la séquence ‘flower’ frame 6……....page68

Tableau III.28: Résultats des algorithmes sur la séquence ‘Formen’ frame 35…….….page68

Tableau III.29 : performance des algorithmes……..……..…page 69

Introduction générale

INTRODUCTION GÉNÉRALE :

Le traitement d'image, inventé au début des années soixante-dix, est de nos jours utilisé dans divers domaines avec des applications de plus en plus nombreuses.

La compression MPEG (Motion Picture Experts Group) est utilisée partout dans la vie courante, surtout le domaine vidéo qui est répandue à grande échelle que depuis la moitié des années 1990, est une norme créé par des collaborateurs qui travaille dans les technologies de l'information. Ils développent des normes internationales pour les codecs, le traitement et le codage de la vidéo et de l'audio, il existe plusieurs standards (le MPEG1, MPEG2, MPEG4……).

Tel que, La compression vidéo est une méthode de compression de données, qui consiste à réduire la quantité de données, en minimisant l'impact sur la qualité visuelle de la vidéo. L'intérêt de la compression vidéo est de réduire les coûts de stockage et de transmission des données vidéo

L'estimation du mouvement est une étape ressente et nécessaire dans la compression des images vidéo, et l'un des domaines du traitement d'image, elle connait une progression importante liée à l'évolution des technologies de l'information ainsi sur les nouvelles technologies dans de nombreux domaine. L'estimation du mouvement consiste à étudier le déplacement des objets dans une séquence vidéo, en cherchant la corrélation entre deux images successives afin de prédire le changement de position du contenu.

Le développement des algorithmes génétiques s'inscrit dans une démarche visant à l'amélioration de qualité d'image, et pour la résolution de problème d'optimisation ils permettent de trouver une solution acceptable dans un temps raisonnable.

Nous présentons dans ce mémoire, une nouvelle méthode d'estimation de mouvement par algorithme génétique.

Dans le premier chapitre nous faisons une rapide généralité sur le traitement d'image ainsi que les différents types d'images.

Le deuxième chapitre contient les différents algorithmes d'estimation de mouvement par la méthode de mise en correspondance (Block Matching).

Le dernier chapitre consiste à montrer le principe des algorithmes génétiques, après on les applique en estimation de mouvement, nous allons analyser tous les résultats obtenus, finalement nous allons faire une comparaison entre les résultats obtenus par l'algorithme génétique développé et les résultats obtenus par les algorithmes classique (block matching).

Chapitre 1 :

Généralité sur le traitement d'image vidéo

1.1. 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 du contenu permet aux êtres 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'en extraire des informations précises.

De ce fait, le traitement d'images 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.

1.2. Notion de base

1.2.1. Définition de l'image :

L'image est une représentation d'une personne ou d'un objet par la peinture, la sculpture, le dessin, la photographie, le film, etc. C'est aussi un ensemble structuré d'informations qui, après affichage sur l'écran, ont une signification pour l'œil humain.

Elle peut être décrite sous la forme d'une fonction I(x ,y) de brillance analogique continue, définie dans un domaine borné, tel que x et y sont les coordonnées spatiales d'un point de l'image et I est une fonction d'intensité lumineuse et de couleur. Sous cet aspect, l'image est inexploitable par la machine, ce qui nécessite sa numérisation, **[1].**

1.2.2. L'image numérique :

Une image numérique est composée d'unités élémentaires (appelées pixels) qui représentent chacun une portion de l'image. Une image est définie par:

- ✓ le nombre de pixels qui la compose en largeur et en hauteur (qui peut varier presque à l'infini).
- ✓ l'étendue des teintes de gris ou des couleurs que peut prendre chaque pixel (on parle de dynamique de l'image). **[2]**

On distingue alors :

1.2.2.1. Les images binaires (noir ou blanc) :

Exemple, images les plus simples, un pixel peut prendre uniquement les valeurs noir ou blanc. C'est typiquement le type d'image que l'on utilise pour scanner du texte quand

Celui-ci est compose d'une seule couleur.

1.2.2.2. Les images en teintes de gris :

En général, les images en niveaux de gris renferment 256 teintes de gris. Image a 256 couleurs, simplement chacune de ces 256 couleurs est definie dans la gamme des gris.

Par convention la valeur zéro représente le noir (intensité lumineuse nulle) et la valeur 255 le blanc (intensité lumineuse maximale).

1.2.2.3. Les images couleurs :

Il existe plusieurs modes de représentation de la couleur, le plus utilise pour le maniement des images numériques est l'espace couleur Rouge, Vert, Bleu(R,V, B).Cet espace couleur est base sur la synthèse additive des couleurs ,c'est-à-dire quel mélange des trois composantes (R, V, B) donne une couleur.

1.2.3. Caractéristique d'une image numérique :

L'image est un ensemble structuré d'informations caractérisé par les paramètres suivants:

1.2.3.1.pixel :

Une image est composée d'un ensemble de points, appelés pixel (abréviation de Picture Elément). Ces pixels sont regroupés en lignes et en colonnes afin de former un espace à deux dimensions. Chaque point sera représenté par ses coordonnées (X, Y), avec X l'axe orienté de gauche à droite, et Y l'axe orienté de haut en bas.

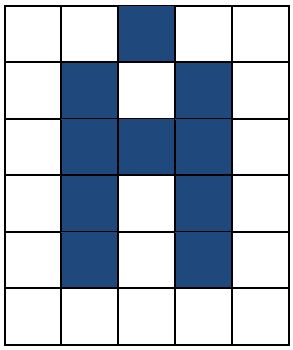


Figure I.1: pixel

1.2.3.2. Dimension :

C'est la taille de l'image. Cette dernière se présente sous forme de matrice dont les éléments sont des valeurs numériques représentatives des intensités lumineuses (pixels). Le nombre de lignes de cette matrice multiplié par le nombre de colonnes nous donne le nombre total de pixels dans une image.

1.2.3.3. Résolution :

C'est la clarté ou la finesse de détails atteinte par un moniteur ou une imprimante dans la production d'images. Sur les moniteurs d'ordinateurs, la résolution est exprimée en nombre de pixels par unité de mesure (pouce ou centimètre). On utilise aussi le mot

résolution pour désigner le nombre total de pixels affichables horizontalement ou verticalement sur un moniteur; plus grand est ce nombre, meilleure est la résolution. **[3]**

1.2.3.4. 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 optiques et électroniques du capteur **[2].**

1.2.3.5. Histogramme :

L'histogramme des niveaux de gris ou des couleurs d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (couleur) dans l'image. Pour diminuer l'erreur de quantification, pour comparer deux images obtenues sous des éclairages différents, ou encore pour mesurer certaines propriétés sur une image, on modifie souvent l'histogramme correspondant.

Il permet de donner un grand nombre d'information sur la distribution des niveaux de gris (couleur) et de voir entre quelles bornes est repartie la majorité des niveaux de gris (couleur) dans les cas d'une image trop claire ou d'une image trop foncée.

Il peut être utilisé pour améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci.

1.2.3.6. Contours et textures :

Les contours représentent la frontière entre les objets de l'image, ou la limite entre deux pixels dont les niveaux de gris représentent une différence significative **[4].** Les textures décrivent la structure de ceux-ci. L'extraction de contour consiste à identifier dans l'image les points qui séparent deux textures différentes.

1.2.3.7. Luminance :

C'est le degré de luminosité des points de l'image. Elle est définie aussi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface, pour un observateur lointain, le mot luminance est substitué au mot brillance, qui correspond à l'éclat d'un objet. Une bonne luminance se caractérise par :

- Des images lumineuses (brillantes);
- Un bon contraste : il faut éviter les images où la gamme de contraste tend vers le blanc ou le noir; ces images entraînent des pertes de détails dans les zones sombres ou lumineuses.
- L'absence de parasites.

1.2.3.8. Contraste :

C'est l'opposition marquée entre deux régions d'une image, plus précisément entre les régions sombres et les régions claires de cette image. Le contraste est défini en fonction des luminances de deux zones d'images.

Si L1 et L2 sont les degrés de luminosité respectivement de deux zones voisines A1 et A2 d'une image, le contraste C est défini par le rapport :

$$\mathbf{c} = \frac{\mathbf{L_1} - \mathbf{L_2}}{\mathbf{L_1} + \mathbf{L_2}}$$

1.2.3.9. Transparence :

La transparence est une caractéristique permettant de définir le niveau d'opacité des éléments d'une image, c'est-à-dire la possibilité de voir à travers l'image des éléments graphiques située derrière celle-ci.

Il existe deux modes de transparence :

- La transparence simple s'applique pour une image indexée et consiste à définir parmi la palette de couleurs une des couleurs comme transparence.
- La transparence par couche alpha (ou canal alpha, en anglais alpha channel) consiste à rajouter pour chaque pixel de l'image un octet définissant le niveau de transparence (de 0 à 255). Le processus consistant à ajouter une couche transparente à une image est généralement appelé alpha blinding.

1.2.3.10. 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 (parfois profondeur de couleur). Il existe plusieurs standards de codage de la profondeur :

- **Codage d'une image en noir et blanc :**

Pour ce type de codage, chaque pixel est soit noir, soit blanc. Il faut un bit pour coder un pixel (0 pour noir, 1 pour blanc). L'image de 10000 pixels codée occupe donc 10000 bits en mémoire. Ce type de codage peut convenir pour un plan ou un texte mais on voit ses limites lorsqu'il s'agit d'une photographie.

- **Codage d'une image en niveaux de gris :**

Si on code chaque pixel sur 2 bits on aura 4 possibilités (noir, gris foncé, gris clair, blanc).L'image codée sera très peu nuancée. En général on code chaque pixel sur 8 bits =

1 octet. On a alors 256 possibilités (on dit 256 niveaux de gris). L'image de 10 000 pixels codée occupe alors 10 000 octets en mémoire. Exemple d'image en 72 pixels par pouce (environ 30 pixels par cm), codée en 256 niveaux de gris.

Une image de 303 x 303 pixels occupe 303 x 303 = 91809 octets puisque chaque pixel occupe 1 octet en mémoire.

- **Codage d'une image en couleurs 8 bits :**

Dans ce cas on attache une palette de 256 couleurs à l'image.

Ces 256 couleurs sont choisies parmi les 16 millions de couleurs de la palette RVB. Pour chaque image le programme recherche les 256 couleurs les plus pertinentes.

Chaque code (de 0 à 255) désigne une couleur.

L'image occupe 3 fois moins de place en mémoire qu'avec un codage 24bits. L'image est moins nuancée : sa qualité est bonne mais moindre.

- **Codage d'une image en couleurs 24 bits**

Il existe plusieurs modes de codage de la couleur. Le plus utilisé est le codage Rouge, Vert, bleu (RVB). Chaque couleur est codée sur 1 octet = 8 bits. Chaque pixel sur 3 octets c'est à dire 24 bits : le rouge de 0 à 255, le vert de 0 à 255, le Bleu de 0 à 255.

Le principe repose sur la synthèse additive des couleurs : on peut obtenir une couleur quelconque par addition de ces 3 couleurs primaires en proportions convenables.

On obtient ainsi 256 x 256 x 256 = 16777216 (plus de 16 millions de couleurs différentes).

Dans le cas d'une image de 10 cm x 10 cm avec une résolution convenable de 100 pixels par cm (un pixel mesure 0,1 mm), elle est codée sur 1000 x 1000 = 1 000 000 pixels

Elle occupe : en noir et blanc : 1 000 000 bits = 125 000 octets en 256 niveaux de gris : 1 million d'octets en couleurs (24bits) : 3 millions d'octets.

C'est ce codage de la couleur qui est utilisé par la plupart des écrans d'ordinateurs actuellement.

On constate qu'il est très gourmand en mémoire. Pour faciliter le stockage des images en mémoire on utilise d'autres formes de codage.

1.2.4. Qualité de l'image numérique :

Elle dépend, d'une part, de la qualité des images d'origine et, d'autre part, des moyens mis en œuvre pour convertir un signal analogique en signal numérique. Elle dépend aussi de :

- La qualité des périphériques de numérisation de l'image, du nombre de niveaux de gris ou de couleurs enregistrées.
- La qualité de l'affichage à l'écran : définition de l'écran, nombre de teintes disponibles simultanément, calibrage de l'écran, etc.

Les critères d'appréciation de la qualité d'une image, tels que cités succinctement ci-dessus, dépendent largement de la structure même de l'image réaliste ou conceptuelle et de son mode de représentation (bitmap ou vectorielle). **[5]**

1.2.5. les formes de l'image :

1.2.5.1.Les images bitmap :

Les images sont constituées d'une matrice de points élémentaires (les pixels), tous porteurs d'un attribut couleur qui leur est spécifique.

Ces matrices des points colorés combinés permettent d'afficher des images plus riches en détails et en complexité. Pour cela le résultat des numérisations est toujours de type bitmap.

Chaque pixel dans la grille de l'image est unique. Si vous souhaitez agrandir votre image, vous demandez à votre programme d'ajouter un certain nombre de pixels à ceux qui existent déjà, vous introduisez donc une distorsion qui dégrade la qualité de l'image de départ.

1.2.5.2.Les images vectorielles :

Les dessins dans le mode vectoriel s'affranchissent totalement de ces contraintes, puisque leur représentation va être stockée, dans le fichier d'enregistrement comme dans l'espace mémoire de travail, sous la forme d'une formule mathématique.

Les données vectorielles peuvent décrire non seulement les formes et les attributs (tel que couleurs ou surface) des objets dessinés, mais aussi leur localisation spatiale.

Prenons l'exemple d'un rectangle. C'est une forme géométrique à quatre angles droits, dont les côtés sont égaux et parallèles deux à deux. Le format vectoriel va stocker la position (dans la largeur et la hauteur de la feuille de travail) de chaque sommet ainsi que les attributs de la ligne les reliant.

L'opération consistant à agrandir le rectangle se résume en fait à déplacer ses sommets (les coordonnées des points dans l'espace) vers les nouvelles positions, puis de redessiner avec la même qualité que dans le rectangle d'origine.

1.2.5.3. Comparaison entre les formes d'image :

Nous avons vu auparavant qu'une image était constituée d'un ensemble de points, nommés pixels, de position et de couleur différente. C'est ce qu'on appelle une image du « bitmap», ou en français « tableau de données binaires ».

Cependant, il existe une autre forme d'image, appelée image vectorielle. Les images vectorielles sont des représentations géométriques (cercle, rectangle, droite, etc.…). Cette propriété spécifique aux images vectorielles, nous permet de coder une image entière en formules mathématiques (exemple : un cercle est défini par son centre et son rayon, le rectangle lui peut être déterminé par ses deux points extremums…). C'est le processeur qui sera ensuite chargé de "traduire" ces formes en informations interprétables par la carte graphique.

Le grand avantage des images vectorielles est le suivant : puisque ces images sont constituées d'entités mathématiques, il est possible de leur appliquer des transformations géométriques.

Lorsqu'on applique un grossissement sur une image du type bitmap, on s'aperçoit que l'image n'est plus nette, et selon le degré de grossissement, on peut observer les pixels, surtout sur les contours des objets (Figure I.2). Avec les images vectorielles, la précision ne dépend pas du facteur d'agrandissement.

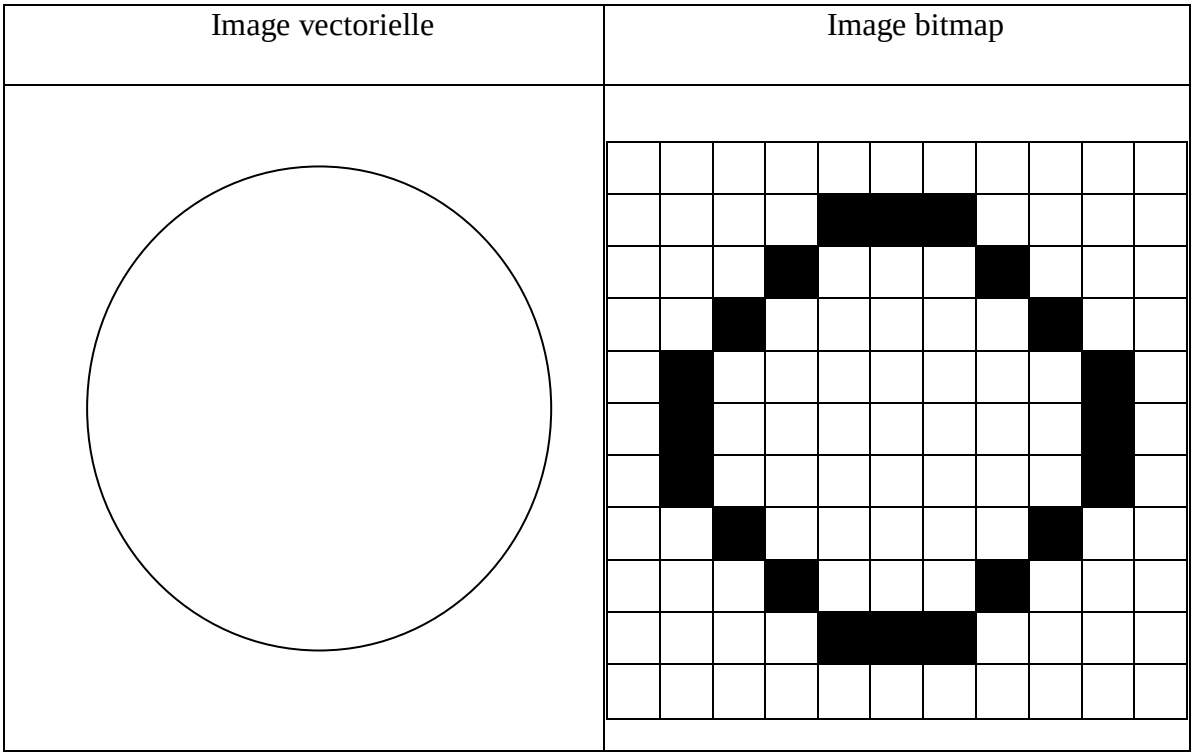


Figure I.2 : comparaison entre la forme d'image vectorielle et l'mage bitmap

1.3. Etapes de Traitement d'image :

1.3.1. définition du traitement d'image :

Le traitement d'image est défini comme l'ensemble des techniques qui permettent de calculer à partir d'une image d'entrée une nouvelle image de sortie. Pour faire le traitement, il faut analyser l'image (classification, description des différentes régions qu'elle contient, mesure des paramètres divers…) ainsi que son interprétation (relation entre les régions et l'objet de façon à donner un sens). La sortie n'est plus une image mais une description de cette image.

1.3.2. Les systèmes de traitement d'images :

Un système de traitement numérique d'images est composé de :



Figure I.3 : Architecture d'un système de traitement d'image

Le processus de système de traitement d'image peut se décompose en étapes suivantes :

1.3.2.1.Acquisition des images:

L'acquisition d'images constitue un des maillons essentiels de toute chaîne de conception et de production d'images. Pour pouvoir manipuler une image sur un système informatique, il est avant tout nécessaire de lui faire subir une transformation qui la rendra lisible et manipulable par ce système. Le passage de cet objet externe (l'image d'origine) à sa représentation interne (dans l'unité de traitement) se fait grâce à une procédure de numérisation.

Ces systèmes d'acquisition sont le plus souvent des camera dotée d'un capteur CCD (Charge Coupled Device).

L'image analogique captée par une caméra sera convertie grâce à une carte graphique d'un ordinateur sous une forme numérique pour un traitement ultérieur.

1.3.2.2. Prétraitement :

Le premier objectif du prétraitement est l'élimination des informations non pertinentes pour faciliter l'extraction des informations utiles aux autres étapes de traitement d'image. Toute acquisition est bruitée, le bruit dépendant de la technologie (capteur, échantillonnage) ; mais aussi de la scène, du contexte (plein soleil, ombres), ou de la nature des objets eux-mêmes (sécularités, transparence).

1.3.2.3. Le traitement représente des fois une étape incontournable.

A. Segmentation :

Le cœur d'un système de traitement d'images est l'étape de la segmentation ou d'extraction des caractéristiques. La segmentation des images consiste à regrouper les pixels de ces images qui partagent une même propriété pour former des régions connexes. Il existe deux familles d'approches que l'on peut faire coopérer : les approches « contours » et les approches «régions ». Dans le premier cas, les régions sont délimitées par les contours des objets qu'elles représentent (séparation). Dans le second cas, les régions sont déterminées en fonction de leurs propriétés intrinsèques (agrégation de pixels fonction d'un critère d'homogénéité).Les deux approches sont duales. Une image est donc une mosaïque de régions homogènes séparées par des zones de transitions.

B. Interprétation :

Suivant la nature de la tâche, cette phase exploite un ou plusieurs types d'informations estimés à partir des données pixels (Niveau de gris, forme, etc.) .Donc l'objectif de l'analyse est de permettre l'exploitation du contenu de l'image à des fins d'interprétation (aide au diagnostic en imagerie aérienne, satellitaire, médicale par exemple), de localisation et de reconnaissance (vidéosurveillance, contrôle robotique) ou de mesure des évolutions (contrôle qualité, suivi thérapeutique, etc.)

1.3.3. Avantages et inconvénients du traitement d'images :

Quelques avantages et inconvénients du traitement d'images sont indiqués ci-dessous :

- **Avantages :**

1. **Rapidité :** Le traitement d'images accélère beaucoup les travaux traditionnellement réalisés par photo-interprétation, mais qui sont automatisables et donc reproductible (e. g. vérification de la bonne disposition d'étiquettes sur les bouteilles produites par une usine d'eau minérale). Dans des cas défavorables, l'interprétation visuelle d'image peut être plus rapide et précise que l'interprétation numérique. En effet, l'opérateur humain peut s'adapter à des conditions hors des limites de bon fonctionnement du système d'analyse numérique (e.g. image déformée du fait d'un éclairage défectueux non modélisable). Ainsi, un opérateur peut plus ou moins faire abstraction de la présence de "bruit" dans une partie de l'image lors de l'analyse de celle-ci. De manière intuitive, il arrive à séparer le bruit et l'information recherchée dans l'image. Ceci est le cas d'une image qui contient une route en partie au soleil et à l'ombre. Une photo interprète peut immédiatement réaliser qu'il s'agit d'un même objet alors qu'une interprétation purement numérique réalisée par un logiciel non adapté peut mener à la conclusion qu'il s'agit de deux objets différents. L'extraction numérique d'information dans une image "bruitée" nécessite des algorithmes complexes capables d'utiliser des informations annexes qui renseignent sur les caractéristiques du bruit présent.
2. **Travaux répétés et fastidieux** : Ces travaux (correction géométrique, étalonnage, gestion de base de données,…) sont souvent difficiles à réaliser, sinon impossibles, de manière non numérique. Ceci explique pourquoi les images analogiques sont souvent numérisées pour être "interprétées".
3. **Reproductibilité de l'analyse et de l'interprétation :** L'interprétation humaine (i.e., photo interprétation) est souvent fastidieuse, longue, coûteuse, difficilement reproductible et très dépendante de l'opérateur. Ainsi, l'interprétation d'une même image par plusieurs photos interprètes donne souvent des résultats très différents, car les êtres humains n'ont pas la même sensibilité aux couleurs, au contexte des objets analysés, etc. Par exemple, une photo interprète peut confondre 2 niveaux de gris alors qu'un ordinateur ne peut confondre 2 valeurs numériques.
4. **Quantification** : Un intérêt majeur du numérique est de donner une information quantifiée.

- **Inconvénients** :

1. Dimensions (aire, périmètre, etc.) de tout ou partie d'un matériau (*i.e.*, nombre de pixels), à partir d'une délimitation numérique ou manuelle.
2. Concentrations chimiques par le biais des caractéristiques spectrales d'objets de l'image.
3. Grandeurs physiques (rugosité, etc.) dérivées de propriétés optiques comme la réflectance.

1.4. Compression vidéo :

1.4.1. Définition de la vidéo :

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 dixième de seconde) toute image imprimée sur la rétine. Il suffit donc de faire défiler un nombre suffisant d'images par seconde, pour que l'œil ne se rende pas compte qu'il s'agit d'images distinctes.

1.4.2. Définition de la compression vidéo :

La compression vidéo est une méthode de compression de données, qui consiste à réduire la quantité de données, en minimisant l'impact sur la qualité visuelle de la vidéo. L'intérêt de la compression vidéo est de réduire les coûts de stockage et de transmission des fichiers vidéo.

1.4.3. Principes de base :

Les techniques principales qui permettent de diminuer les redondances dans une séquence vidéo sont l'espace de couleurs, le codage spatial, le codage temporel différentiel et la compensation de mouvement.

1.4.3.1. Espaces de couleurs :

La représentation classique d'une image numérique utilise un espace de couleur dit RGB. Cela correspond à une représentation discrète de l'image où l'on quantifie la couleur en chaque point par trois valeurs : Rouge, Vert et Bleu (RVB ou RGB en anglais). Cette représentation a cependant plusieurs défauts : elle utilise une grande quantité d'information et ne tient pas compte du fait que l'œil humain ne perçoit pas les couleurs en RVB mais au travers de deux types de cellules : les une perçoivent la luminosité (en noir et blanc), et les autres la coloration. Pour cela on utilise un espace de couleurs plus approprié, de façon à mieux profiter du format de la vision humaine. Le format adopté est un format dit YUV. (Y représente la luminosité U représente la première valeur de chrominance (ou chroma) V représente la deuxième valeur de chrominance.)

L'utilisation du format YUV permet de profiter d'une caractéristique de l'œil humain : sa capacité de distinction des couleurs est plus faible que celle de distinction de la luminosité. Cela permet de réduire la quantité d'information des espaces de chrominance par rapport à celle de luminosité, on fait c'est ce qu'on appelle du sous-

échantillonnage sur les composantes U et V. Cela correspond en général à les réduire de moitié de résolution, pour une perte de qualité assez faible.

1.4.3.2.Codage spatial :

Le codage spatial consiste à prendre l'image, la découper en blocs de 8x8 pixels, et à coder chaque bloc via DCT, puis quantifier et compresser comme l'indique la figure (1.5)

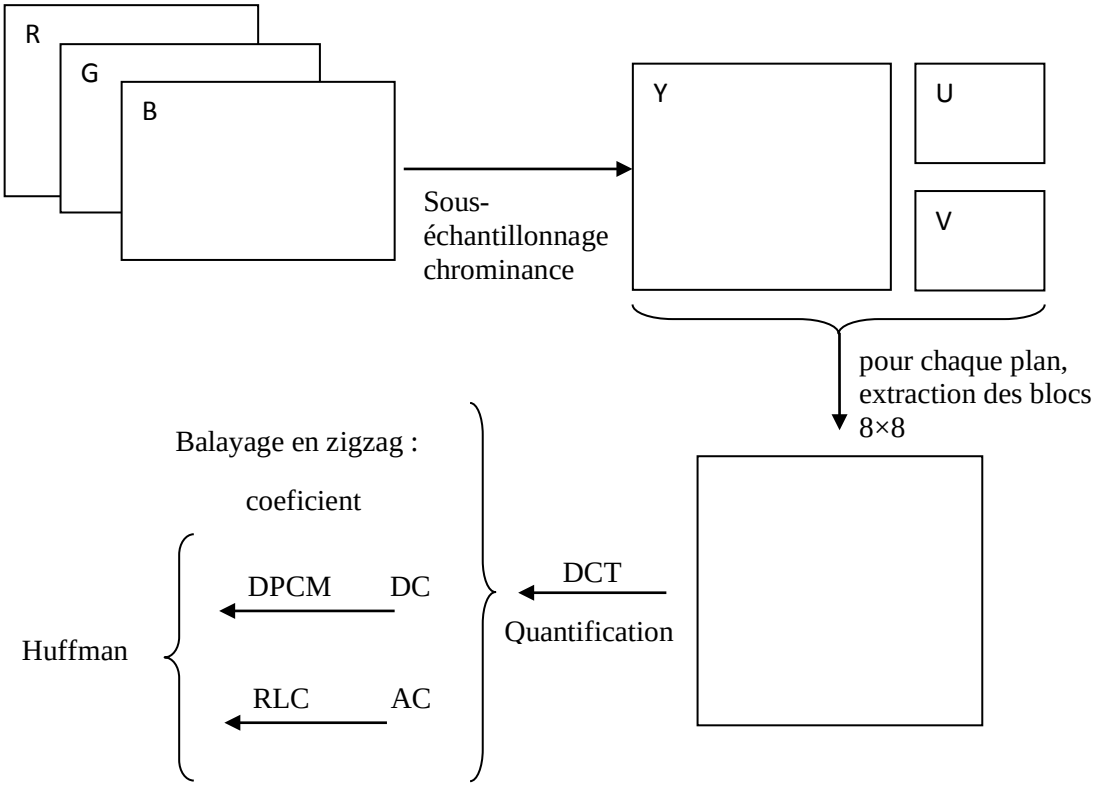


Figure I.4: Codage spatial

1.4.3.3. Codage temporel différentiel :

Le codage temporel différentiel est un moyen de codage assez naturel. Il part du constat que deux images I_n et I_{n+1} successives sont fortement semblables et qu'il est plus avantageux de coder I_n puisI_{n+1}-I_n que les images de manière indépendante.

1.4.3.4. Compensation de mouvement :

La technique de compensation de mouvement part du constat, qu'une séquence est souvent composée de déplacement, ou d'objets et personnages qui traverse le champ de la caméra. De tels évènements peuvent être facilement modélisés par la translation d'une portion de l'image In de la séquence vers une autre portion d'une image In+1 de la séquence. Le gain de code est réel car il suffira de transmettre le vecteur de mouvement plutôt que la totalité du bloc (ou de la différence entre les deux blocs dans le cas d'un codage différentiel).

1.4.4. Classification des méthodes de compression :

La plupart des méthodes de compression visent à enlever la redondance présente dans l'image de manière à diminuer le nombre de bits nécessaires à sa représentation.
Plusieurs types de redondance en termes de corrélation peuvent être considérés :
– La redondance spatiale entre pixels ou blocs voisins dans l'image ;
– La redondance temporelle entre images successives dans une séquence vidéo ;

Les méthodes de compression peuvent se regrouper, en deux classes :
– Les méthodes sans perte d'information (sans distorsion ou réversible) ;
– Les méthodes avec perte d'information (avec distorsion ou irréversible).

Les expérimentations menées montrent que généralement les méthodes qui atteignent des taux de compression très élevés sont les méthodes avec distorsion. Par contre, les méthodes sans distorsion engendrent des taux de compression très faibles et ne sont utilisées que dans des applications sensibles telles que les images médicales et les images satellites **[6]**

Les principaux critères d'évaluation de toute méthode de compression sont :
– La qualité de reconstitution de l'image ;
– Le taux de compression ;
– La rapidité du codeur et décodeur (codec).

❖ Le schéma fonctionnel de la compression est présenté dans la figure ci-dessous :

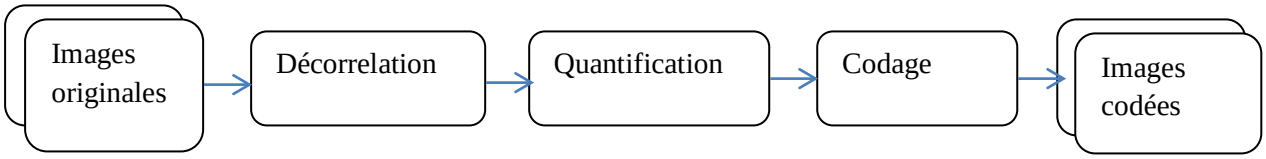


Figure I.5 : Schéma d'un codeur source.

➢ **Décorrélation :**

La dépendance existante entre chacun des pixels et ses voisins (la luminosité varie très peu d'un pixel à un pixel voisin) traduisent une corrélation très forte sur l'image **[7]**. On essaie donc de tirer partie de cette corrélation, pour réduire le volume d'information en effectuant une opération de décorrélation des pixels. La décorrélation consiste à transformer les pixels initiaux en un ensemble de coefficients moins corrélés, c'est une opération réversible **[8].**

➢ **Quantification :**

La quantification des coefficients a pour but de réduire le nombre de bits nécessaires pour leurs représentations. Elle représente une étape clé de la compression **[9]**. Elle

approxime chaque valeur d'un signal par un multiple entier d'une quantité q, appelée quantum élémentaire ou pas de quantification. Elle peut être scalaire ou vectorielle. Un des résultats fondamentaux des travaux de Shannon concernant la relation : Débit/Distorsion, montre que l'on obtient de meilleures performances en utilisant la quantification vectorielle **[10].**

- **Codage :**

Une fois les coefficients quantifiés, ils sont codés. Un codeur doit satisfaire à priori les conditions suivantes **[11]** :

- ✓ Unicité : deux messages différents ne doivent pas être codés de la même façon,
- ✓ Déchiffrabilité : deux mots de codes successifs doivent être distingués sans ambiguïté.

1.4.5. Les types de compression (avec et sans perte) :

Il est important de comprendre que toute compression apportée à une image n'implique pas forcément une diminution de la qualité d'une image. En effet, dans le cas d'une compression sans pertes, aucune information n'a subi de détérioration mais présente simplement une réorganisation des données informatiques composant une image. L'exemple le plus parlant est le principe des poupées russes qui prendront moins de place dans une armoire si celles-ci sont toutes ramenées l'une dans l'autre, sans qu'il n'en manque pourtant une seule.

A la différence d'une compression d'image sans pertes, la compression avec pertes ou destructive, réglera le sort de bon nombre d'informations jugées inutiles car non perceptible par un œil humain, c'est le cas des images compressée en JPEG par exemple.

1.4.6. Les normes de compression vidéo (historique) :

En un peu plus de dix ans, quatre standards de compression numérique des images ont été définis par des groupes de travail spécialisé, formés sous les auspices de l'ISO (*International Standard Organization*): JPEG (*Joint Photographics Experts Group*), MPEG-l (*Motion Pictures Experts Group*), MPEG-2 et MPEG-4. **[5]**

1.4.6.1. JPEG et M-JPEG :

Apparue en 1989, JPEG est une norme de compression des images fixes, conçue à l'origine pour le monde de l'impression et de la photocomposition. JPEG accepte n'importe quelle définition d'image et exploite les uniformités présentes à l'intérieur de chacune d'elle; le codage est dit « intra-image ». Or, comme les circuits intégrés de codage JPEG étaient disponibles bien avant tout autre circuit de compression, les fabricants d'équipements vidéo se sont rapidement intéressés à ce standard, même s'il ne leur était pas destiné au départ. Partant du principe qu'une séquence vidéo n'est qu'une succession rapide d'images fixes, ils ont développé des systèmes JPEG capables dé compresser/décompresser en temps réel 25 ou 30 images par seconde.' C+ solutions,

baptisées M-JPEG (Motion-JPEG), se sont vite répandues à partir du début des années 90, notamment dans les stations de montage non-linéaire et dans le domaine de l'enregistrement haut de gamme. Cependant, les modifications ayant permis de passer du JPEG au M-JPEG, ainsi que la synchronisation du son, n'ont jamais fait l'objet d'une normalisation. Les fabricants ont donc développé sans concertation des solutions propriétaires, si bien que les fichiers générés par des équipements de marques différentes sont très souvent incompatibles entre eux.

L'échange des programmes M-JPEG n'est possible qu'à travers un cycle de compression/décompression du signal. Depuis, est apparue le format d'enregistrement DV, qui utilise une compression de type M-JPEG, mais dont l'algorithme a été totalement normalisé à l'échelle mondiale. Initialement développé pour le grand public avec un débit de 25.Mbits/s, le DV a progressivement évolué vers des débits supérieurs (50 et 100 Mbits/s), donnant naissance à une famille de systèmes de compression adaptés à tous les besoins de la chaîne de production et postproduction broadcast, en définition standard comme en haute définition: DV, DVCAM, DVCPRO25, DVCPRO50 et DVCPRO-HD.

1.4.6.1. MPEG-1:

Apparue en 1992, MPEG-l est une norme de compression des images animées à faible résolution, destinée aux applications multimédia. Fruit de quatre années de travaux au sein du comité MPEG, le MPEG-1 a été conçu pour coder un programme audio/vidéo dans une qualité tout juste comparable à du VHS. Son débit maximal a en effet volontairement été bridé à 1,5 Mbits/s (dont 1,15 Mbits/s pour la vidéo), afin d'être compatible avec le stockage sur un Compact Disc (CD-Rom, CD-Vidéo), qui peut ainsi contenir 74 mn de programme. Pour parvenir à un débit aussi bas, MPEG-l travaille sur une résolution d'image réduite à 1/4 seulement de celle d'une image standard (360 x 288), et ne prend en compte qu'une trame sur deux (l'autre est simplement répétée). Il effectue ensuite un codage intra-image de type JPEG à l'intérieur de chaque image, qu'il complète par un redoutable codage inter-image. C'est ce dernier qui, en exploitant les redondances temporelles entre plusieurs images successives (rendues de ce fait indissociables) permet d'obtenir des taux de compression aussi élevés. L'une des applications phare du MPEG-l est le CD-Vidéo, qui n'a pas véritablement percé en Europe, mais qui a rencontré un grand succès dans d'autres pays comme la Chine ou plusieurs millions de lecteurs ont été vendus. MPEG-l est aujourd'hui notamment utilisé comme format de consultation dans plusieurs applications de diffusion, mais il tend à être remplacé par MPEG4.

1.4.6.2. MPEG-2 :

Normalisé en 1994, MPEG-2 est devenu le standard de compression de référence pour tous les secteurs de diffusion/distribution audiovisuelles, et a permis à l'industrie grand public de migrer avec succès de la technologie analogique à la technologie digitale. Il n'est pas exagéré de dire que toute image affichée sut un écran de télévision aujourd'hui a été au moins une fois codée en MPEG2 au cours de son cheminement. MPEG- 2 reprend la philosophie et l'ensemble des techniques de base de MPEG-l (compressions

spatiale de type JPEG et temporelle), avec des outils optimisés pour le faire sérieusement monter en puissance en termes de débit, donc de qualité d'image. Tout d'abord, la résolution de l'image d'entrée s'étend de la définition standard (720 x 576J à la haute définition (1920 x 1080), avec un balayage entrelacé ou progressif. Le débit vidéo s'échelonne typiquement de 4 à 50.Mbits/s en SDTV, et peut atteindre 300 Mbits/s en HDTV. Du coup, MPEG3, qui devait se consacrer à part entière à la haute définition, a été totalement absorbé par MPEG-2, qui possède tous les outils nécessaires. MPEG-2 a d'abord été étudié pour la diffusion numérique ; il est aujourd'hui mondialement utilisé par tous les opérateurs de télévision numérique et constitue également le standard des DVD- Vidéo. Par la suite, MPEG-2 a été décliné en une version « studio », baptisée MPEG-2 422P@ML (ou MPEG-2 422), répondant aux exigences plus sévères de la production/postproduction b1roadcast et permettant de travailler en mode intra-image, comme en M-JPEG, pour permettre des manipulations et traitements complexes et précis des images.

1.4.6.3. MJPEG-4 :

Normalisée en 2000, MPEG-4 se distingue fondamentalement de MPEG-l et MPEG-2 par son approche orientée objet et l'intégration, dans une norme unique, de toutes les composantes liées à l'interactivité. Ce standard est capable de coder individuellement lès différents éléments média d'une image (les « objets» audio, Vidéo, graphiques...) en les faisant bénéficier du traitement le mieux adapté à leur nature. Ces objets sont alors transmis séparément sur tout type de réseau, afin de permettre à l'utilisateur final d'interagir sur la composition de la scène audiovisuelle qu'il reçoit. En particulier, le traitement de la vidéo, qui a jusqu'à présent concentré l'essentiel des développements de la norme, est le plus performant de tous les standards actuels. Il reprend les grands principes déployés par MPEG-2; mais avec des outils plus sophistiqués et des techniques inédites, offrant un codage plus raffiné et aboutissant à une efficacité de compression très sensiblement supérieure. MPEG-4 est un standard extrêmement riche et complexe, pouvant fournir des débits allant de quelques Kbits/s à plusieurs centaines de Mbits/s.

Son champ d'application est, on l'imagine, très vaste, allant du streaming sur Internet à l'enregistrement de la vidéo à haute définition, en passant par le multimédia mobile.

1.4.6.4. JPEG-2000 :

JPEG 2000 est une récente norme de compression des images, utilise notamment dans le domaine du cinéma numérique. Elle met en œuvre un processus d'analyse innovant, radicalement différent de celui de toutes les normes précédentes, plus puissant mais impliquant des ressources d'encodage plus gourmandes. l'un des atouts majeurs du JPEG-2000 est de permettre le décodage progressif .comme nous le verrons plus en détail par la suite, l'image est en effet compressée résolution par résolution, ce qui permet d'afficher instantanément, à la réception, des images de résolution et de qualité différentes à partir d'un unique flux, sans aucun calcul additionnel. Généralement, l'efficacité de compression de JPEJ-2000est de 20à 40% supérieure à celle de JPEG

JPEG	Initialement codage d'images fixes. A été décliné pour la vidéo, considéré comme une succession d'images fixes, en plusieurs versions dites M-JPEG (non normalisé) et DV (normalisé). Débit types de 25 à 100 Mbits/s.
MPEG-1	Codage vidéo en qualité VHS pour le CD-Rom. Débits de 1.15 Mbits/s.
MPEG-2	Codage vidéo en qualité de diffusion standard et haute définition. Débits de 4 à 300 Mbits/s.
MPEG-4	Codage orienté objet appliqué individuellement à chaque élément média d'une scène audiovisuelle (audio, vidéo, 2D, 3D). Débits de quelques Kbits/s à plusieurs centaines de Mbits/s.
JPEG 2000	Probable successeur de JPEG, devenu le standard de compression de cinéma numérique. Débit de type 450Mbit/s en projection 4K (taux de compression12 :1) et 2,2Gbit en production (compression 5 :1)

Tableau I.1 : différents standards de compression.

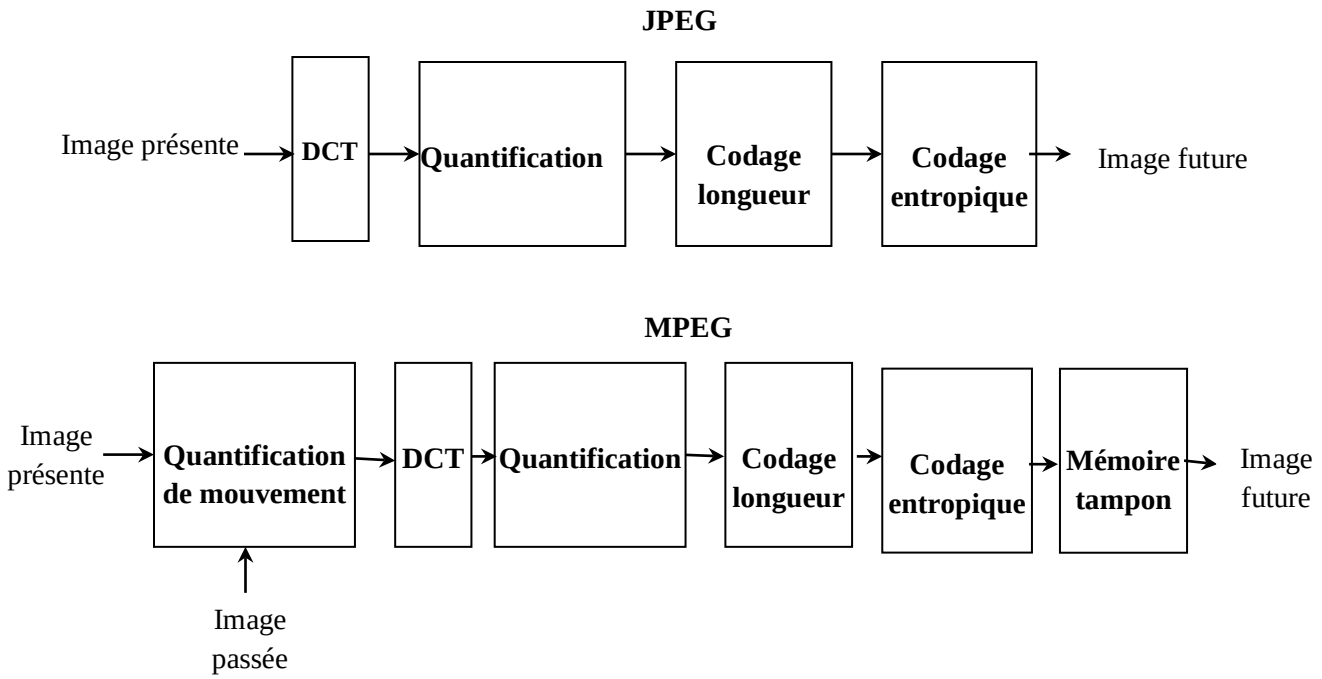


Figure I.6 : principe des compressions JPEG et MPEG.

La compression JPEG (et sa déclinaison vidéo M-JPEG) traite indépendamment chaque image en éliminant ses redondances spatiales : le codage est dit « intra-image ».

Les compressions MPEG-1 et MPEG-2 traitent les images par groupes, en exploitant leurs redondances temporelles, c'est-à-dire les données communes à plusieurs images successives : le codage est dit « inter-image ». MPEG-2 peut cependant travailler uniquement en mode intra-image pour les applications de studio. MPEG-4 reprend les principes de MPEG-1 et 2, mais avec des outils plus performants adaptés à tous les débits, et avec une approches orientée objet.

1.5. Conclusion :

Dans la compression numérique MPEG, l'estimation de mouvement joue un rôle très important, plutôt un rôle principal, ainsi que pour toutes les normes de compression, MPEG-2, MPEG-4. Il existe de nombreuses techniques pour estimer le mouvement, la technique utilisée par les normes MPEG est celle d'appariement de blocs, ou "Block Matching".

On va étudier l'estimation de mouvement dans le chapitre suivant et les techniques (ou bien les méthodes) les plus utilisées pour cette étape inclus dans la compression vidéo.

Chapitre 2 :

Les algorithmes d'estimation de mouvement

2.1. Introduction :

L'estimation de mouvement est un problème couramment rencontre en traitement d'images et de vidéos. De nombreuses techniques existent. Dans ce chapitre, on présentera l'estimation du mouvement de manière générale en illustrant son fonctionnement. Puis on réalisera un inventaire des principales techniques de mise en correspondance de blocs

2.2. Mouvement réel, mouvement apparent et mouvement estimé: [13]

Les images représentent souvent la projection de scènes réelles 3D. C'est pourquoi le mouvement observé (ou mouvement apparent) dans une séquence temporelle d'images représente le plus souvent la projection du mouvement 3D dans le plan image. Une image peut également représenter une coupe d'un objet 3D. Dans ce cas, le champ de mouvement observe à l'intérieur de cette coupe pourra représenter à la fois un déplacement dans le plan image et également la projection sur ce plan d'un mouvement 3D. Le but de l'estimation de mouvement est d'estimer le champ de mouvement 2D ou 3D à partir d'une séquence temporelle d'images 2D ou 3D dont le contenu varie en fonction du temps. On doit différencier:

- Le mouvement réel;
- Le mouvement apparent (ou observé);
- Le mouvement estimé.

2.2.1. Champ de mouvement réel et champ de mouvement apparent :

Le mouvement (ou déplacement) réel anime la scène réelle, dans l'espace réel 3D. Ce mouvement réel est observé soit par l'œil humain soit par un système de prise de vue. Le second fournit un enregistrement du mouvement observé ou apparent sous la forme d'une séquence temporelle d'images. Souvent, Le champ de mouvement réel n'est pas directement mesurable dans une séquence d'images 2D ou 3D si l'on s'appuie seulement sur l'intensité des pixels. Dans ce cas, on observe en fait des changements de la distribution spatiale d'intensité lumineuse. Le mouvement ainsi perçu est appelé champ de "mouvement apparent" ou "flux optique" qui, en général, est différent du champ réel de mouvement.

Le champ de mouvement ou de déplacement apparent représente, en général, la projection (orthographique ou projective) du mouvement réel 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 (figure II.1).

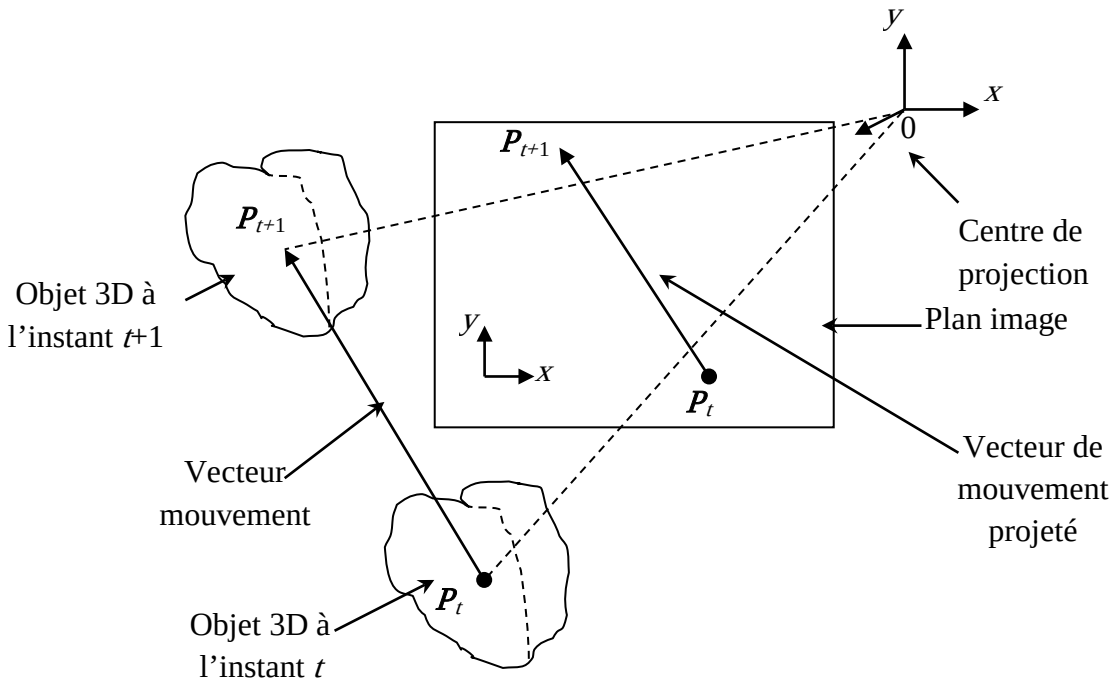


Figure II.1 : Illustration des mouvements réel et apparent, dans un système optique de prise de vues.

On suppose que le point P_t d'un objet réel 3D, à l'instant t, devient Le point P_{t+1}à l'instant t+1. La projection perspective des points P_t et P_{t+1}dans le plan image est notée P_t et P_{t+1}, respectivement avec $I_t(x,y)$ et $I_{t+1}(x,y)$ leur intensité. Sur la figure II.1 on note que le déplacement apparent $\overline{p_t\, p_{t+1}}$correspond à la projection perspective du mouvement réel $\overline{P_t\, P_{t+1}}$. Tous les vecteurs déplacements dont les extrémités appartiennent aux droites pointillées correspondant à l'opération de projection ont le même vecteur 2D déplacement apparent (ou projeté). L'illustration présentée sur la figure II.1 est valable dans le cas optique, c'est-à-dire pour une séquence d'images acquise avec une caméra vidéo.

2.2.2. Champ de mouvement apparent et champ de mouvement estimé :

Le vecteur déplacement estimé $d(\boldsymbol{p})=(d_x(\boldsymbol{p}),d_y(\boldsymbol{p}))$ correspondant au déplacement du point $\boldsymbol{p}=(x,y)$ du plan image, est déterminé à partir du champ de mouvement apparent, c'est-à-dire des variations locales d'intensité lumineuse $I_t(x,y)$ entre les instants t et $t+\Delta t$, où Δt est la distance en temps inter-images, dans le cas continue le vecteur vitesse estimé $v=(v_x(\boldsymbol{p}),\ v_y(\boldsymbol{p}))$ est défini comme la variation temporelle du déplacement par unité de temps, $(v_x,v_y)=(d_x/dt,d_y/dt)$. Ceci explique pourquoi une séquence temporelle d'images ne permet que d'estimer le champ de mouvement (déplacement ou vitesse) apparent observable dans la séquence, et non le champ de vitesse réel déplacement (respectivement de vitesse) estimé avec une méthode quelconque.

En optique par exemple, la différence entre le champ de vitesse estimé et le mouvement apparent, peut exister si:

Le gradient spatial d'intensité est trop faible. Pour que le mouvement réel soit observable, il faut que la variation de niveaux de gris (ou des couleurs) soit suffisamment grande dans les régions où il y a mouvement. L'exemple classique qui met en évidence l'importance de ce facteur, est celui d'une sphère ayant une distribution uniforme d'intensité (éclairage homogène), qui effectue un mouvement de rotation autour de son axe propre, dans une scène à éclairage constant. Même si en réalité il y a mouvement, celui-ci ne peut être observé à partir d'une séquence temporelle d'images à niveaux de gris, parce qu'aucune variation apparente de niveaux de gris n'intervient pendant le mouvement.

L'illumination de la scène varie. Un mouvement observable dans une séquence d'images d'intensité ne correspond pas toujours à un mouvement réel. La même sphère immobile soumise à une illumination variable au cours de la séquence génère un mouvement apparent (c'est-à-dire une variation d'intensité) artificiel.

2.3. Estimation de mouvement : [13]

L'estimation du mouvement est un problème incontournable dans le domaine du traitement de séquences d'images. Le but est d'estimer le mouvement des objets contenus dans une vidéo sous forme d'un champ dense de vecteurs. Les applications d'une telle estimation sont nombreuses et variées et nous pouvons noter par exemple la segmentation d'objets en mouvement, la compensation du mouvement utilisée en compression vidéo (normes MPEG2, MPEG4), etc.

L'estimation de mouvement est excessivement couteuse en temps de calcul. Cela peut prendre jusqu'à 80%des ressources matérielles de la machine. Une estimation de faible qualité est souvent source de problèmes importants comme par exemple des effets de blocs visuellement gênants. Ainsi, depuis plus de vingt ans, son étude reste un sujet de recherche important.

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 déplacées (en anglais DFD=Displaced Frame Difference), c'est-à-dire entre les images aux instants t et $t+\Delta t$, avec $\Delta t = \pm 1$:

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

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 $t+\Delta t$. On observe que:

- si les composantes de d ne sont pas entières, le calcul de la DFD en chaque

pixel, nécessite une étape d'interpolation;

- 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:

1) l'estimation des vecteurs déplacement dans Le plan image:

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

estimés entre les images à t et $t+1$;

2) l'estimation des vecteurs vitesse:

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

Les vecteurs déplacement (respectivement vitesse) estimés peuvent varier en espace et en temps.

2.3.1. 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 les instants t et $t+1$ ou entre les instants t et t-1 (figure II.2).

Dans le cas de l'estimation directe (avant) du mouvement, le problème se pose de la manière suivante: si on connaît 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):

$$I(x,y)=I_{t+1}(x+d_x(x,y,t),y+d_y(x,y,t)),$$
(II.4)

on doit trouver le vecteur de déplacement:

$$d_t(x,y)=(d_{xd}(x,y,t),d_{yd}(x,y,t))$$
(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:

$$I_t(x,y)=I_{t-1}(x-d_{xi}(x,y,t),y-d_{yi}(x,y,t)),$$
(II.6)

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

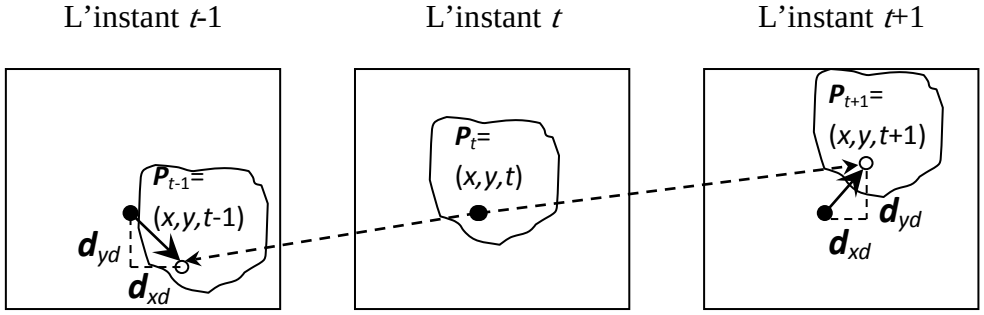


Figure II.2 : Estimation directe et inverse des vecteurs déplacement.

Dans Le cas de l'estimation directe on a note: $d_d(x,y,t)=(d_{xd}(x,y,t), d_{yd}(x,y,t))$.

Dans Le cas de I 'estimation inverse on a note: $d_i(x,y,t)=(d_{xi}(x,y,t), d_{yi}(x,y,t))$.

2.3.2. Estimation des vecteurs vitesses:

étant donnes 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:

$$\boldsymbol{v}_t(x,y)=\boldsymbol{d}_t(x,y)/\Delta t \qquad \text{(II.7)}$$

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 champ de 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'intéressera à l'estimation du champ de vecteurs déplacement.

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

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

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

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

2.4.Techniques d'estimation de mouvement :

Dans le cas des techniques standard de compression vidéo (MPEG-2, MPEG-4), chaque image est encodée bloc par bloc. Les vecteurs de mouvement sont exploites afin de réduire les redondances temporelles et ainsi réduire la taille nécessaire au codage de la séquence. H.264 prévoit de plus une compensation de mouvement avec des blocs de taille variable afin d'adapter l'information de mouvement aux données.

L'opération d'estimation de mouvement s'avère cruciale dans un encodeur vidéo. En effet, d'une part la qualité de l'estimation influe sensiblement sur les performances de compression, et d'autre part c'est l'opération qui nécessite le plus de puissance de calcul. Notons que nous ne considérons ici que des mouvements de translation.

2.4.1. Block-matching: mise en correspondance de blocs :

2.4.1.1.Principe :

Le principe général du bloc-matching est d'exploiter les redondances temporelles existant entre des images consécutives. Pour cela, considérons une séquence vidéo dans laquelle nous voulons estimer le mouvement des différents objets qui y sont contenus. 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 8*8 ou 16*16 pixels, (la figure II.3) et chaque bloc est considéré comme étant un objet indépendant. 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 (passée ou future) et à calculer un critère comparaison entre ces deux blocs. 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 testes ou jusqu'à un critère d'arrêt arbitraire. Le bloc le plus semblable est ainsi identifie dans l'image de référence.

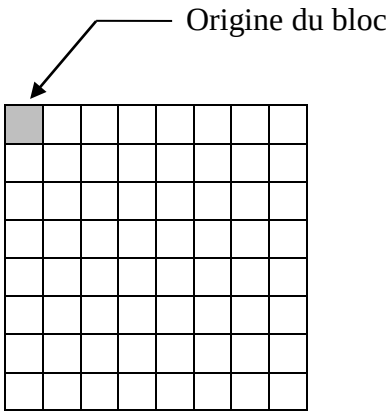


Figure II.3 : illustration d'un bloc de taille 8×8 pixels

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 autorise fixe par l'utilisateur.

Initialement, on peut vouloir placer l'origine d'un bloc en son centre. Cependant, il est beaucoup plus astucieux de le situer dans le coin supérieur gauche pour deux raisons: premièrement, les blocs sont en général des côtes de taille paire ce qui implique qu'il n'y a pas de pixel central; deuxièmement, les tests d'appartenance à la fenêtre sont moins couteux en temps de calcul avec cette convention. Le nombre et l'ordre dans lequel les blocs que les conditions d'arrêt ont des influences majeures sur l'efficacité de l'algorithme de recherche.

2.4.1.2. Stratégies de recherche :

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 de comparaison étant alors déplace pixel par pixel, à l'intérieur de la fenêtre de recherche. Le balayage exhaustif pressente 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 II.4):

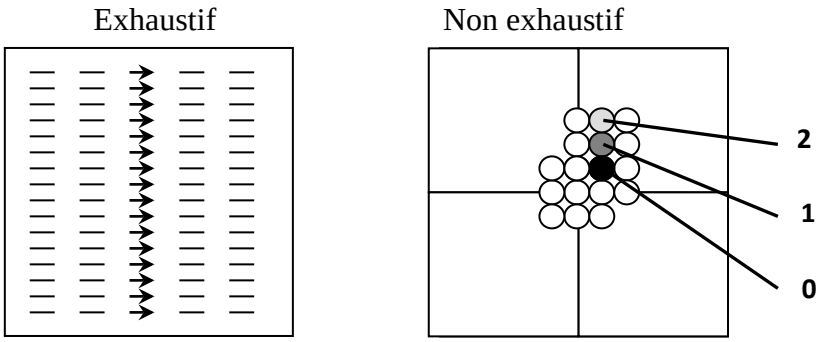


Figure II.4 : Stratégies de balayage de la fenêtre de recherche.

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 II.4). Les stratégies de recherche orientée les plus utilisées sont:

- la recherche en *n* pas;
- la recherche 2D-Iogarithmique (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.

2.4.1.3. Critères de comparaison :

Le critère de comparaison précédemment mentionne pour trouver le bloc correspondant est une mesure objective de la dissemblance entre les valeurs des pixels contenus dans chaque bloc. Il en existe plusieurs qui prennent en compte ou non la couleur. De manière générale, la plupart des articles traitant de l'estimation de mouvement considèrent que les blocs sont caractérisés par la luminance. Ceci est dû au fait que l'œil humain est davantage sensible à l'intensité lumineuse qu'a la couleur. Cependant, ignorer les chrominances peut alors poser quelques problèmes dont un exemple sera donne dans la section évaluant les critères de comparaison. Je vous présente les différents critères de comparaison couramment utilisés en traitement d'image et la manière dont la couleur a été inclue. Mais avant, introduisons les notations nécessaires a la bonne compréhension de la suite.

2.4.1.2.1. Différence absolue :[13]

Le critère le plus utilisé en pratique dans l'implantation hardware est minimisation de l'erreur absolue MAD, définie comme:

$$MAD(l,k) = \frac{1}{B_x.B_y} \sum_{i=1}^{B_x} \sum_{j=1}^{B_y} \left| B_r(i,j) - B_c(i,j) \right| \qquad (II.8)$$

$$= \frac{1}{B_x.B_y} \sum_{i=1}^{B_x} \sum_{j=1}^{B_y} \left| I(i, j, t_k) - I(i + d_x, j + d_y, t_k + \delta t) \right| \quad \text{(II.9)}$$

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

2.4.1.2.2. Erreur quadratique moyenne :[13]

L'erreur quadratique moyenne (MSE) peut être définie comme:

$$MSE(l,k) = \frac{1}{B_x.B_y} \sum_{i=1}^{B_x} \sum_{j=1}^{B_y} \left[B_r(i, j) - B_c(i, j) \right]^2 \quad \text{(II.10)}$$

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

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

$$= \frac{1}{B_x.B_y} \sum_{i=1}^{B_x} \sum_{j=1}^{B_y} \left[I(i, j, t_k) - I(i + d_x, j + d_y, t_k + \delta t) \right]^2 \quad \text{(II.11)}$$

$$\Rightarrow \hat{d} = (\hat{d}_x, \hat{d}_y) = \arg_{(d_x, d_y)} \min MSE(d_x, d_y) \quad \text{(II.12)}$$

2.4.2. Les algorithmes d'estimation de mouvement :

Il existe de nombreuses méthodes de block-matching dans l'ensemble des articles cherchant à optimiser l'efficacité et la rapidité de l'algorithme. Rappelons qu'étant donne un bloc dans l'image courante, l'objectif est de trouver le bloc dans l'image précédente ou suivante (selon le type de prédiction utilise) qui lui correspond au mieux selon des critères précédemment définis. Nous vous présentons dans cette partie diverses méthodes largement utilisées, de la plus simple et la plus ancienne, à savoir la recherche exhaustive, a des méthodes très récentes. Je vous expose l'évolution des techniques au cours du temps, les intérêts et les inconvénients qu'elles représentent.

2.4.2.1. L'algorithme "Full search" 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 **[14]**, **[15]**. 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 $P \times Q$ avec une fenêtre de recherche de taille $(2.W+1) \times (2.W+1)$, un taux d'images de T frames/seconde et des blocs de taille $N \times N$, 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) **[16].**

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.

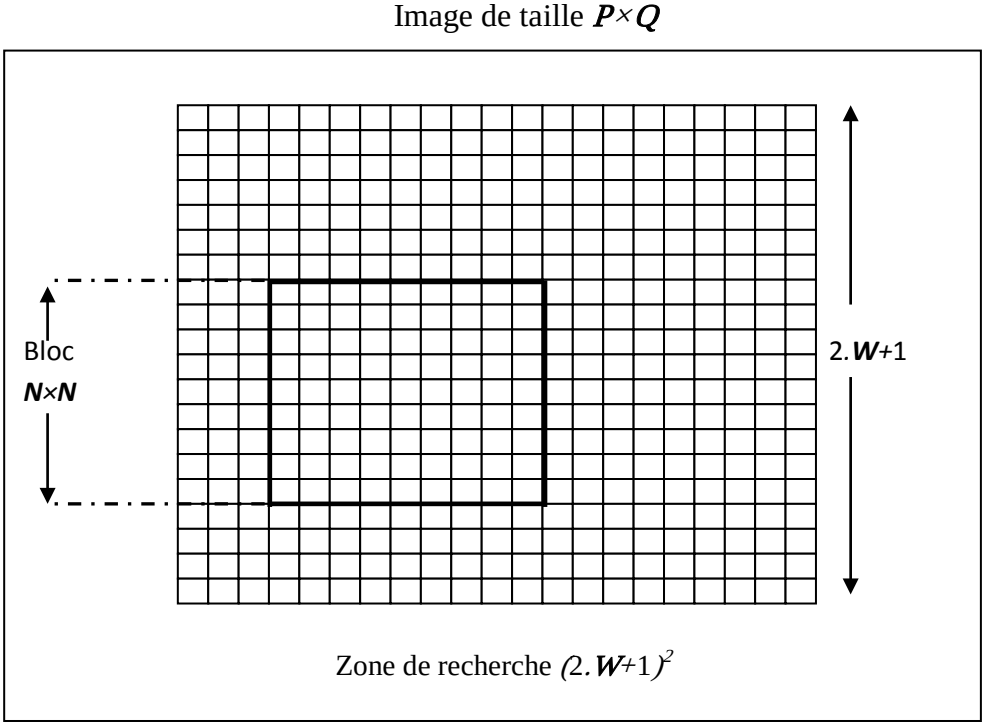


Figure II.5: Modèle de recherche "Full Search" FS

2.4.2.2. L'algorithme à trois étapes de recherche TSS :

L'algorithme à trios étapes de recherche (***ThreeStepSearch*** 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.

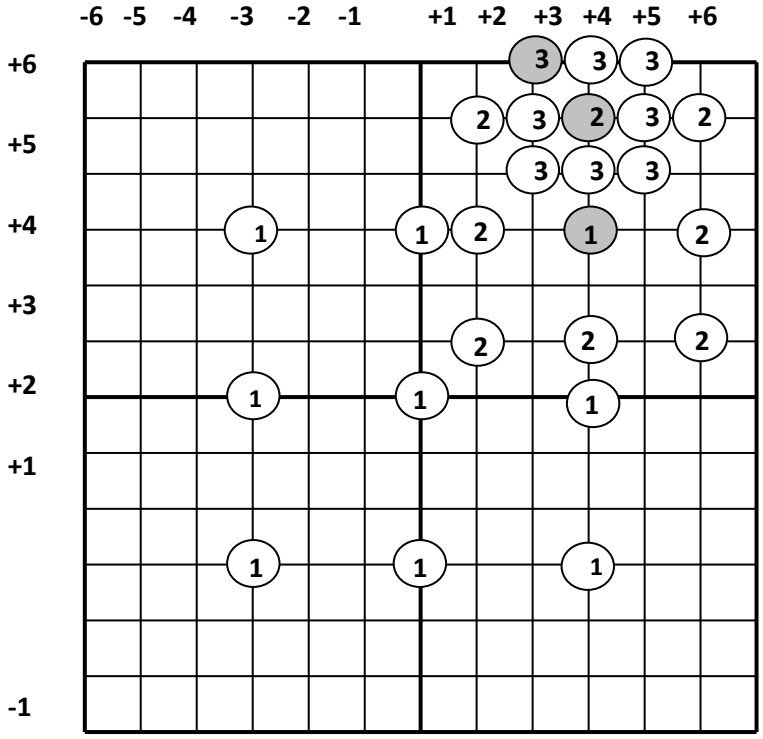


Figure II.6. Modèle de recherche à trois étapes TSS.

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 **[17]**:

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 rechercher. (Les neuf points sont signés par le numéro (1) dans la figure II.6)

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.

2.4.2.3.Le nouveau 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 a bas débit binaire. Cependant, comme mentionné dans le 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. Ceci est réalisé par l'emploi d'un modèle au 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.

L'algorithme NTSS est comme suit. Comme mentionné dessus, NTSS diffère de TSS par l'utilisation d'un modèle a point de contrôle à centre compensé dans la première étape et par l'incorporation d'une technique d'arrêt à mi-chemin pour les blocs stationnaires ou quasi-stationnaires.

Les détails sont comme suit **[18]** :

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 (centre-polarisé) (les cercles remplis en noir et les carrés dans la figure(II.8).

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 (II.8).

Le diagramme bloc montré dans la figure (II.7) montre les étapes impliquées dans l'algorithme NTSS.

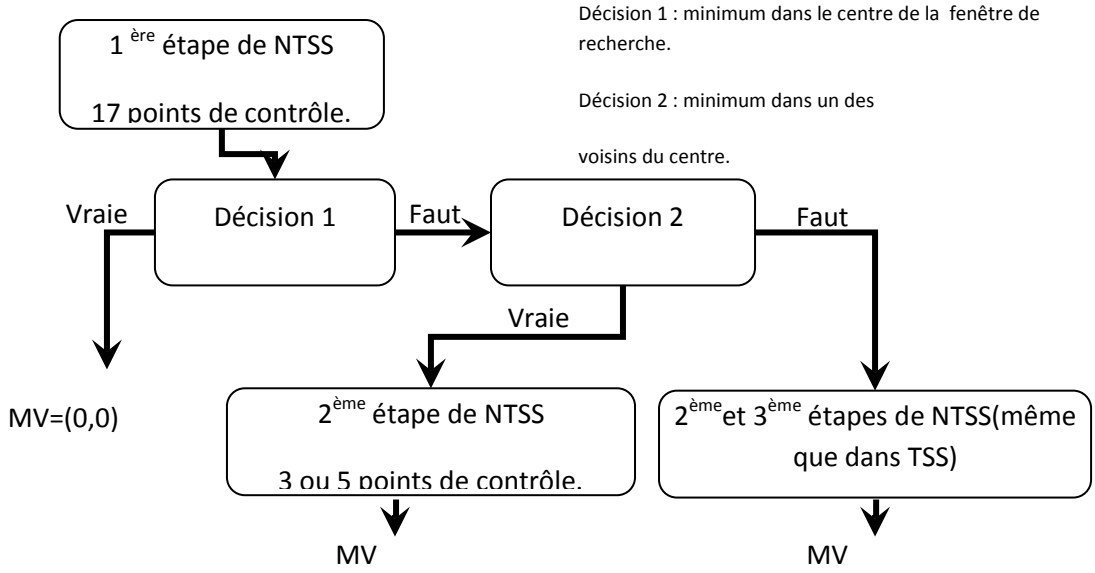


Figure II.7. Diagramme bloc de l'algorithme NTSS.

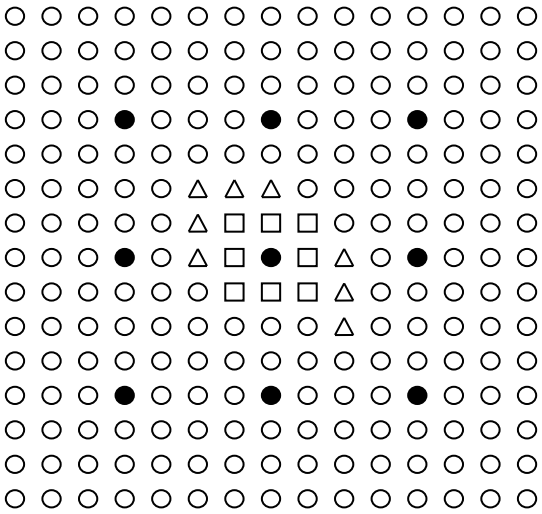


Figure II.8. Région de recherche de l'algorithme NTSS.

2.4.2.4. L'algorithme à quatre étapes de recherche FSS:

L'algorithme à quatre étapes de recherche (***F**our **S**tep**S**earch* 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 **[19].**

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 à centre compensé 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.

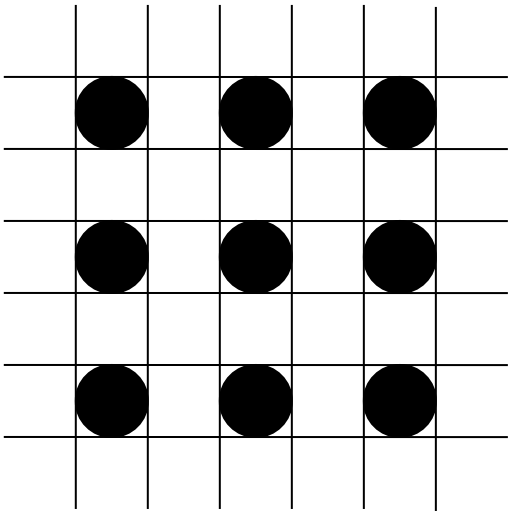


Figure II.9.(a) : Modèle de recherche 'première étape'.

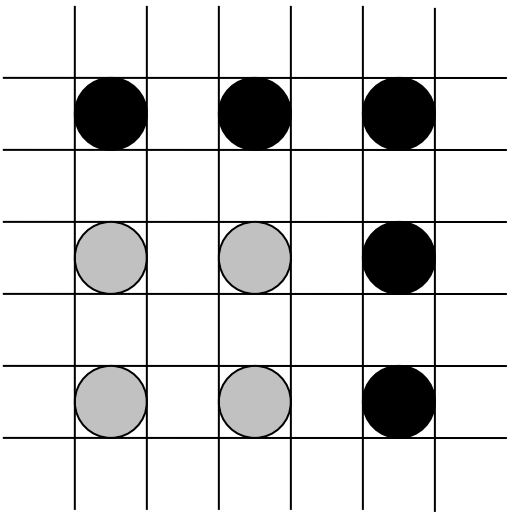


Figure II.9.(b): Modèle de recherche 'deuxième/troisième étape'.

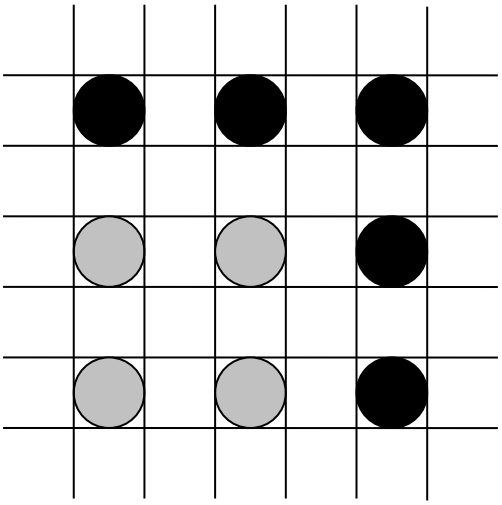


Figure II.9.(c). Modèle de recherche 'deuxième/troisième étape'.

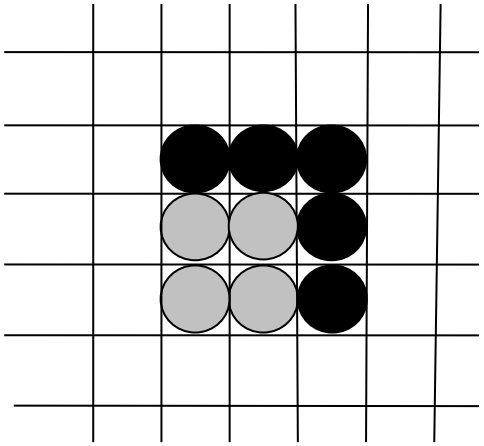


Figure II.9.(d). Modèle de recherche 'quatrième étape'.

L'algorithme FSS peut être résumé comme suit **[20]**:

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 II.9.(a). Si le minimum BDM est trouvé 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 dépendra de 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 II.9.(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 II.9.(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 FigureII.9.(d) et la direction du vecteur global de mouvement est considéré comme le point du minimum *BDM* parmi ces neuf points de recherche.

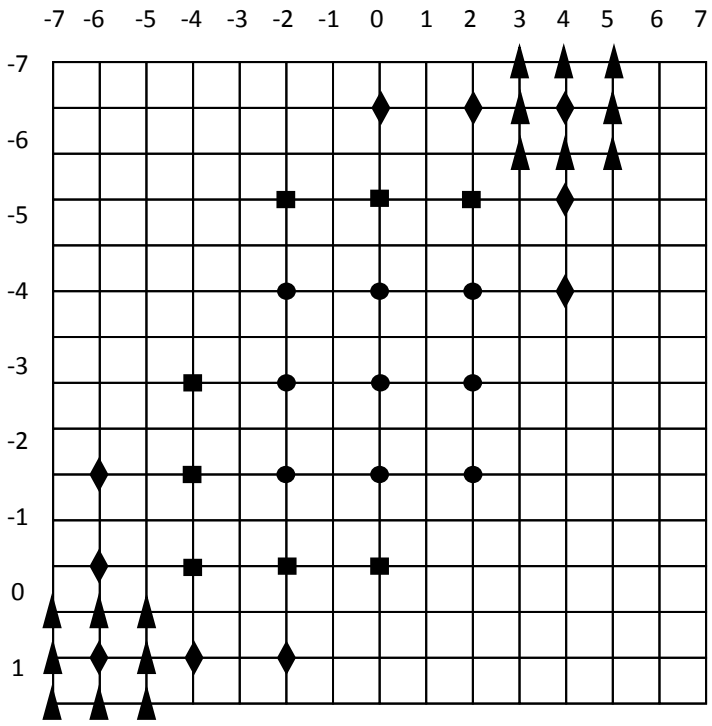


Figure II.10. Modèle de recherche de l'algorithme FSS.

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.

2.4.2.5. L'algorithme de recherche en diamant DS:

L'algorithme de recherche en diamant (***D**iamond **S**earch* 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 dans un 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 compensé.

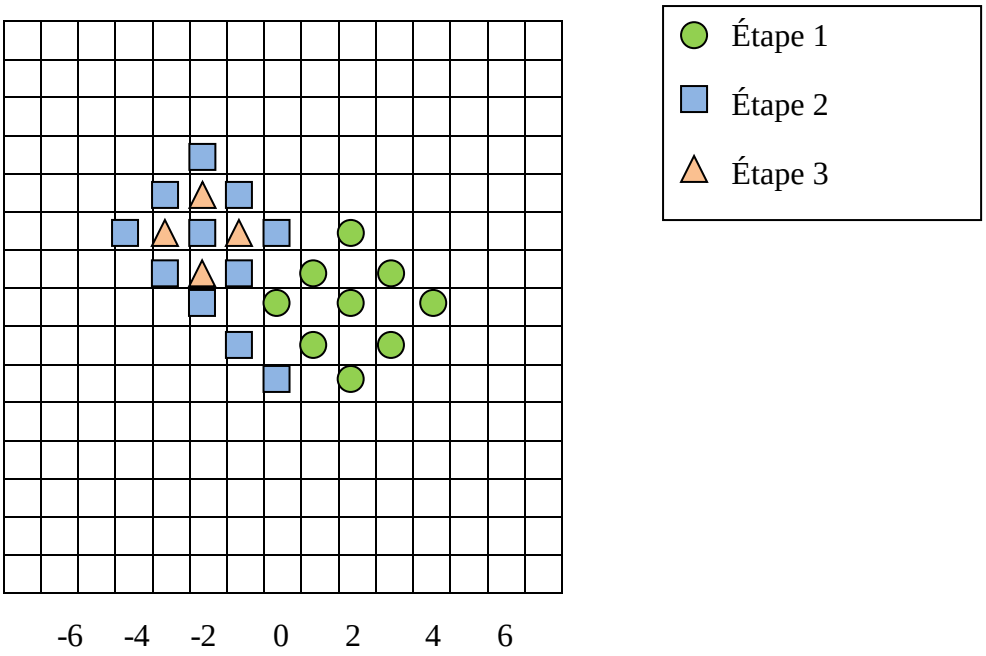
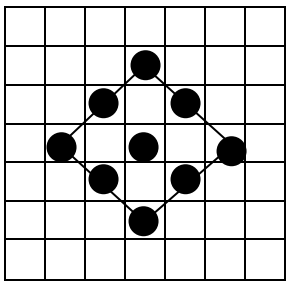


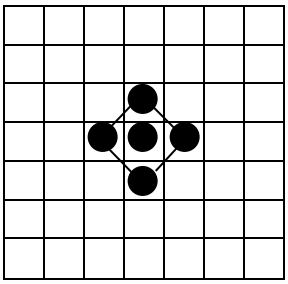
Figure II.11. Diamond Search Algorithme

L'algorithme *DS* utilise deux modèles de recherche. Un modèle de recherche en grand diamant (***L**arge **D**iamond**S**earch**P**attern*) de 9 points de recherche comme montré sur la figureII.12.(a), et un modèle de recherche en petit diamant (***S**mall **D**iamond**S**earch**P**attern*) de 5 points comme montré sur la Figure II.12.(b)



FigureII.12.a:

Large Diamond Search Pattern



FigureII.12.b:

Small Diamond Search Pattern

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 **[21]** :

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.

2.4.2.6. 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 pixels, donc pour les petits déplacements, il converge rapidement.

La stratégie de recherche est comme suit **[22] :**

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 II.13. Le minimum BDM sera le centre de la fenêtre de recherche suivante.

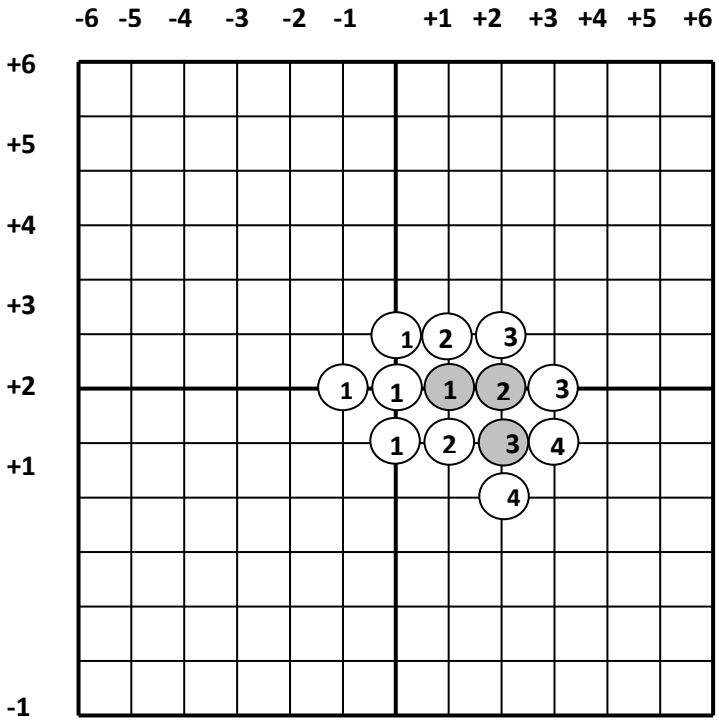


Figure II.13. Méthode de recherche de l'algorithme minimum.

Dans la deuxième étape (minimum de l'étape précédente), il y a quatre points de recherche, le point central 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 de bloc 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.

2.5. Conclusion :

L'estimation de mouvement par la méthode de mise en correspondance est l'une des méthodes les plus utilisées dans ce domaine vue sa simplicité et son efficacité, mais pouvant facilement se faire piéger dans un minimum local, c'est pour cette raison on a vu qu'on utilise une deuxième méthode.

Les algorithmes génétiques sont de plus en plus employés dans des domaines très différents, c'est pourquoi nous avons voulu les tester dans le cadre de l'estimation de mouvement.

Chapitre 3 :

L'estimation d'image vidéo par Les algorithmes génétiques et résultats

3.1. Introduction :

Dans le dernier chapitre, une présentation générale des algorithmes génétiques sera faite et leur principe sera énoncé. Pour mettre en place leur utilisation en tant que nouvelle méthode de résolution pour le problème posé. Les résultats obtenus par l'application des algorithmes génétique sur l'estimation de mouvement sont montré dans ce chapitre.

Finalement, nous allons discuter ces résultats et nous allons faire une comparaison entre eux et les résultats obtenus par les algorithmes classiques (FS, TSS, …….).

3.2. Les algorithmes génétiques :

Les algorithmes génétiques, initiés dans les années 1970 par John Holland, sont des algorithmes d'optimisation stochastique s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature : croisement, mutation, sélection. **[23]**

Plus précisément, ils s'inspirent de l'évolution d'une population d'individus dans un milieu donné, dans le but d'obtenir une population acceptable dans un temp convenable.

3.2.1. Principe :

Les algorithmes génétiques sont basés sur trois éléments principaux : la sélection, le croisement et la mutation. Dans la littérature on parle alors d'opérateurs de reproduction.

Leur principe est simple. Il est basé sur trois phases :

1. la genèse = l'initialisation aléatoire d'individus pour la population de la première génération.
2. la reproduction = l'évolution des individus de la génération courante vers la suivante.

(a) la sélection des individus reproducteurs.
(b) le croisement génétique de ces individus pour la création de nouveaux individus.
(c) la mutation de certains individus pour que le pool génétique ne s'affaiblisse pas.
(d) l'évaluation des individus par le calcul de leur fitness.

3. recueil du meilleur individu = recherche de l'individu le plus adapt selon les critères souhaités.

Les sous étapes 2 et 3 se répètent autant de fois qu'il y a besoin de générations (itérations de l'algorithme) pour satisfaire un critère d'arrêt. Celui-ci est défini avant que le processus ne commence. La solution est alors représente par le meilleur individu de la dernière génération. **[24]**

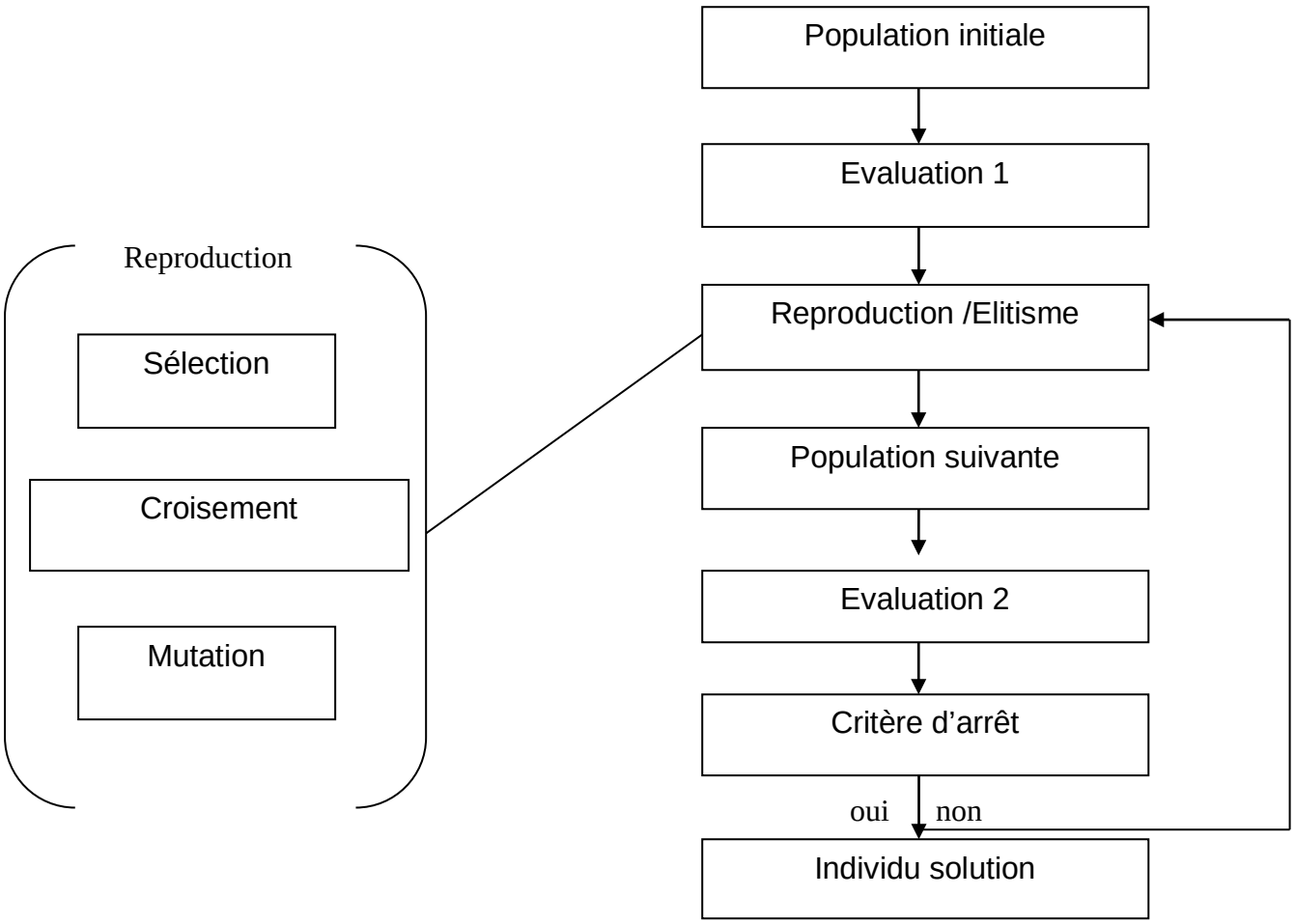


Figure III.1 : Schéma du principe des algorithmes génétiques

3.2.2. Le codage :

Chaque paramètre d'une solution est assimilé à un gène, toutes les valeurs qu'il peut prendre sont les allèles de ce gène, on doit trouver une manière de coder chaque allèle différent de façon unique (établir une bijection entre l'allèle "réel" et sa représentation codée).

Un chromosome est une suite de gêne, on peut par exemple choisir de regrouper les paramètres similaires dans un même chromosome (chromosome à un seul brin) et chaque gène sera repérable par sa position : son locus sur le chromosome en question. Chaque individu est représenté par un ensemble de chromosomes, et une population est un ensemble d'individus.

Plusieurs codages sont employés. Voici quelques exemples.

3.2.2.1. Le codage binaire :

Le gène est code par un caractère binaire, 0 ou 1. C'est le plus courant et celui qui a et employé lors de la première application des algorithmes génétiques.

Exemples :

Chromosome A

0	0	1	0	1	1	1	0

Chromosome B

1	0	1	1	0	1	0	0

3.2.2.2. Le codage par permutations de valeurs entières :

Le gène est coder par une valeur entière dans un ensemble de cardinalité égale au nombre de gènes. Ce codage est bien adapté aux problèmes d'ordonnancement.

Exemples :

Chromosome A

3	6	7	9	8	5	1	2

Chromosome B

8	5	1	2	6	7	4	3

3.2.2.3. Le codage par valeurs :

Le gène est code par une valeur prise dans un ensemble fini ou infini. Ce codage est généralement utilise pour des valeurs qu'on ne peut pas mettre sous la forme d'un des deux codages précédents. Ces valeurs sont bien entendu liées au problème à résoudre.

Exemples :

Chromosome A

E	R	Z	G	N	K	J	M

Chromosome B

2.361	9.506	0.321	8.458	6.753	3.015	6.148	4.587

Chromosome C

Rouge	Noir	Bleu	Vert	Gris	Blanc	Rouge	Bleu

3.2.3. La fonction d'adaptation :

La fonction d'adaptation, ou fitness, associe une valeur pour chaque individu. Cette valeur a pour but d'évaluer si un individu est mieux adapté qu'un autre a son environnement. Ce qui signifie qu'elle quantifie la réponse fournit au problème pour une solution potentielle donnée. Ainsi les individus peuvent être comparés entre eux.

Cette fonction, propre au problème, est souvent simple à formuler lorsqu'il y a peu de paramètres. Au contraire, lorsqu'il y a beaucoup de paramètres ou lorsqu'ils sont corrélés, elle est plus difficile à définir. Dans ce cas, la fonction devient une somme pondérée de plusieurs fonctions. Un ajustement des coefficients est alors nécessaire.

3.2.4. Les opérateurs de reproduction :

Les algorithmes génétiques sont basés sur un phénomène naturel : l'évolution. Plus précisément, ils supposent, qu'a priori, deux individus adaptés à leur milieu donnent, par recombinaison de leurs gènes, des individus mieux adaptés. Pour ce faire, trois opérateurs sont à disposition : la sélection, le croisement et la mutation, plus un opérateur optionnel, l'élitisme.

3.2.4.1. L'opération de sélection :

La sélection sert à choisir dans l'ensemble de la population les individus qui participeront à la reproduction. Plusieurs méthodes existent et sont, généralement, basées sur la théorie de Darwin. Ainsi les meilleurs individus ont plus de chance de survivre et de se reproduire. Nous présenterons les deux plus connus.

- **La roulette**

Cette méthode exploite la métaphore d'une roulette de casino. La roue est divisee en autant de secteurs que d'individus dans la population. La taille de ces secteurs est proportionnelle à l'adaptation de chaque individu. En faisant tourner la roue, l'individu pointé à l'arrêt de la boule est sélectionné. Les individus les mieux adaptés ont donc plus de chance d'être tirés au sort lors du déroulement du jeu.

- **Le tournoi**

Cette méthode ressemble plus à ce qui se passe dans la réalité. Comme son nom l'indique, elle fait s'affronter deux ou plusieurs individus afin que le meilleur gagne. Plusieurs variantes existent. On peut par exemple faire varier le nombre d'individu qui doivent s'affronter au départ, ou encore permettre ou non que le même individu soit éligible plusieurs fois lors d'un même tournoi.

3.2.4.2. L'opération de croisement ou crossover :

Le croisement combine les gènes des deux individus parents pour donner deux nouveaux chromosomes d'individus enfants. La zone de croisement est généralement choisie aléatoirement dans les chromosomes. Les méthodes de croisement sont liées au codage. Le croisement de chromosomes codé en binaire ne sera pas le même que celui d'un chromosome codé par valeur entière mais leur principe est identique.

Ci-dessous, un schéma illustrant un croisement en un point, un autre pour un croisement en deux points, et enfin un schéma représentant un croisement avec *lg - 1* points de croisements (on notera d'ailleurs sur ce schéma que l'échange d'un segment avec son homologue ne se fait pas toujours) :

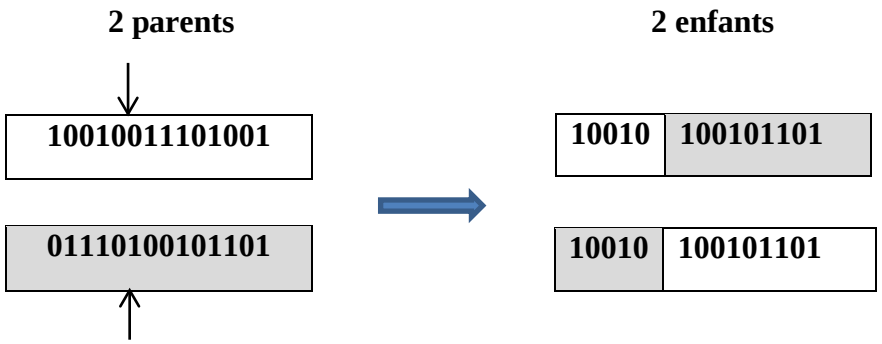


Figure III.2: croisement avec un point de crossover

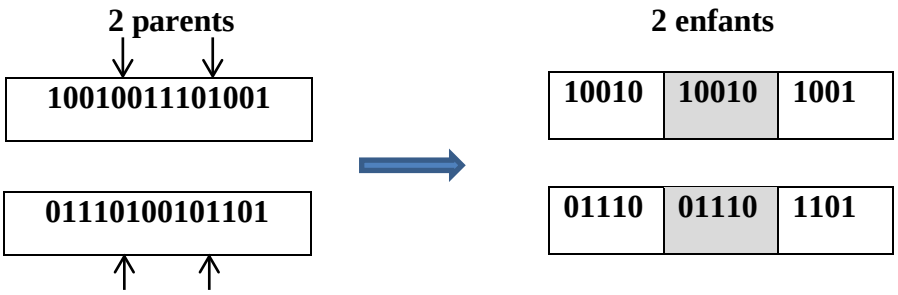


Figure III.3: croisement avec 2 points de crossove



Figure III.4: croisement uniforme

3.2.4.3. La mutation :

Comme les individus les mieux adaptés sont les plus susceptibles d'être choisis lors de la sélection, la perte de certains gènes est inévitable avec le temps. La mutation est l'opérateur qui permet d'éviter la d´dégénérescence de la population et d'enrichir le pool de gènes. Cette dégénérescence peut se traduire par une convergence des individus vers un optimum local,

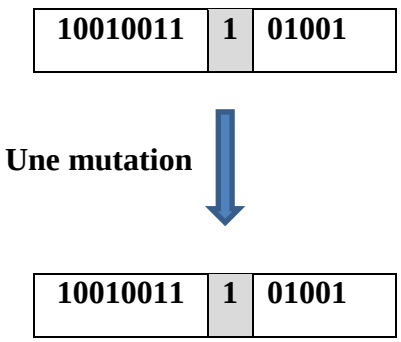


Figure III.5 : la mutation

D'où l'importance de la mutation.
-Classiquement, la mutation modifie aléatoirement, un petit nombre de gènes, avec un faible taux de probabilité.
-Bien entendu, comme pour le croisement, la mutation d´dépend du problème.

2.2.4.4. L'élitisme :

L'élitisme est un opérateur optionnel particulier. Il permet de garder l'individu le mieux adapté d'une génération `a la suivante. En effet, l'opérateur de sélection peut ne pas le sélectionner, le croisement avec un autre individu peut donner des individus moins adaptés si les gènes ne sont pas bien recombiner ou encore sa mutation peut également le rendre moins adapté.

3.3. Application des algorithmes génétiques en estimation de mouvement :

3.3.1. Mise en œuvre les algorithmes (analogie):

Les algorithmes	**un algorithme génétique**	**Algorithme génétique appliqué à l'estimation du mouvement**
les différents paramètres appliqués.	**La population**	**Les blocs d'images (16x16 pixels)**
	L'individu	**Les pixels de l'image**
	La fonction d'évaluation	**Le minimum entre les deux MSE de deux blocs, le premier bloc est de l'image actuelle et le deuxième de l'image précédente.**
	Chromosome	**MSE**
	Gène	**La valeur radiométrique du chaque pixel.**

Tableau III.1 : application des algorithmes genetique en estimation de mouvement

3.3.2. L'organigramme de l'algorithme génétique :

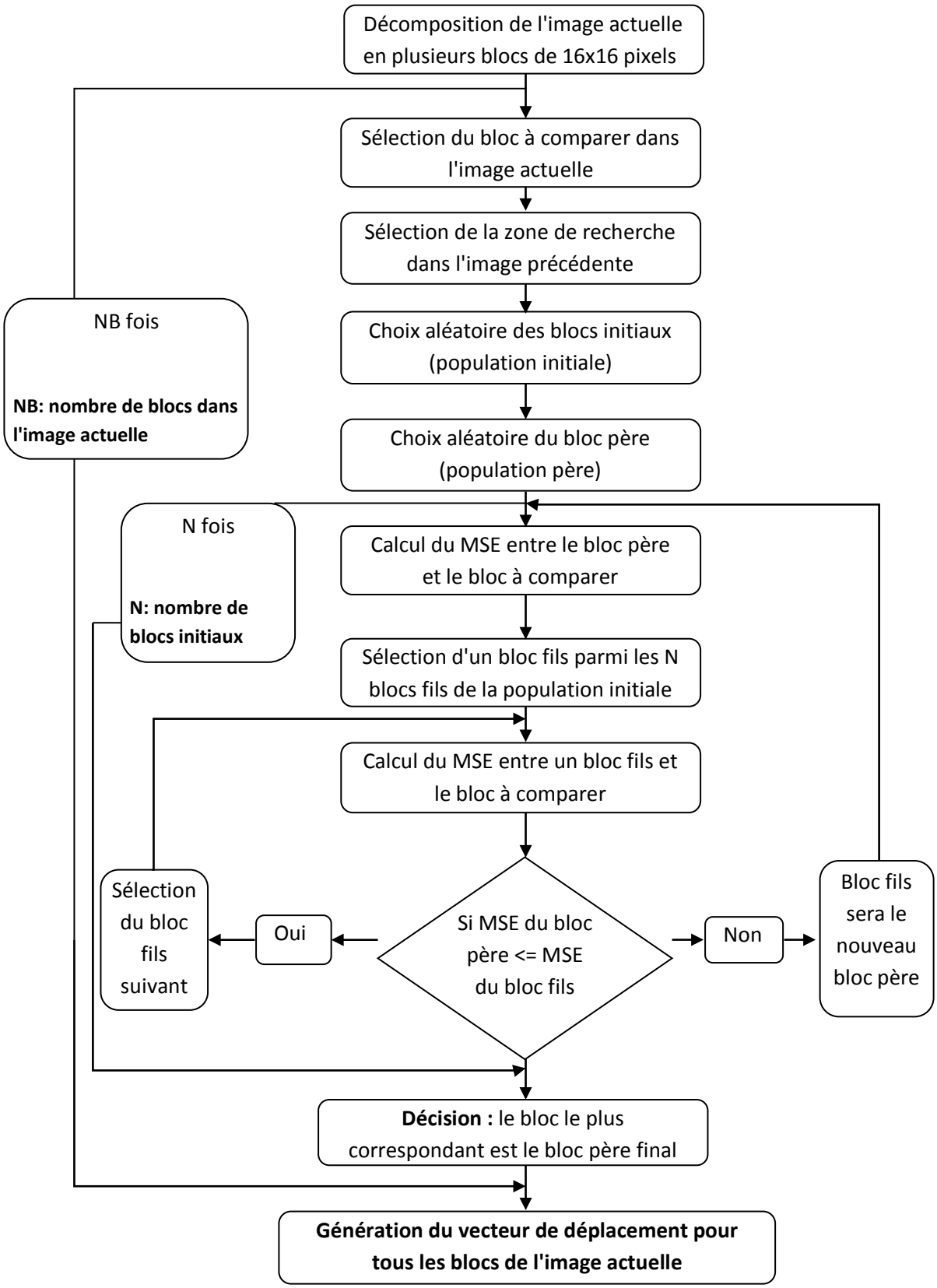


Figure III.6 : L'organigramme de l'algorithme génétique :

3.4. Résultats pratiques :

3.4.1. Résultats de recherche entre 50 blocs aléatoires :

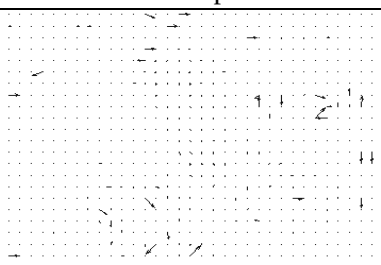
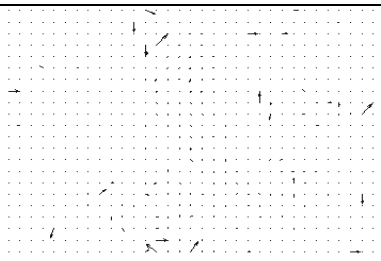
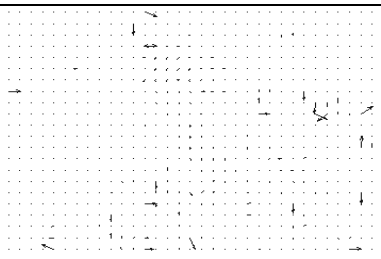
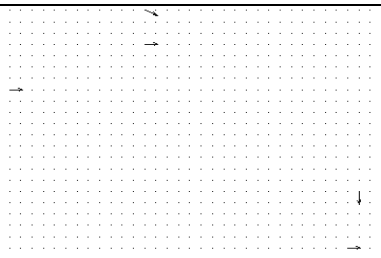
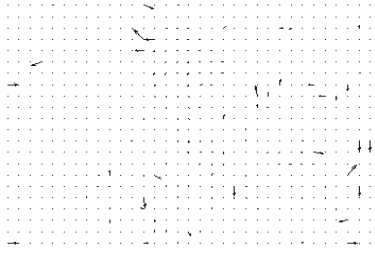
Images originaux	Image reconstruites	Vecteurs de déplacement
 Frame 21	 Frame 21 reconstruite à partir du frame 20 originale	 Entre les frames n° 20 et 21
 Frame 22	 Frame 22 reconstruite à partir du frame 21 reconstruite	 Entre les frames n° 21 et 22
 Frame 23	 Frame 23 reconstruite à partir du frame 22 reconstruite	 Entre les frames n° 22 et 23
 Frame 24	 Frame 24 reconstruite à partir du frame 23 reconstruite	 Entre les frames n° 23 et 24
 Frame 25	 Frame 25 reconstruite à partir du frame 24 reconstruite	 Entre les frames n° 24 et 25

Tableau III.2 : résultat sur la séquence 'Akiyo' pour la recherche par 50 blocs

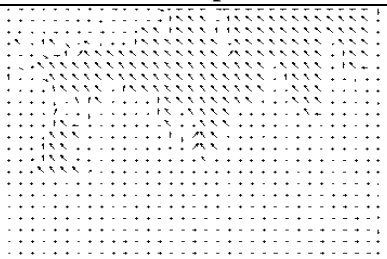
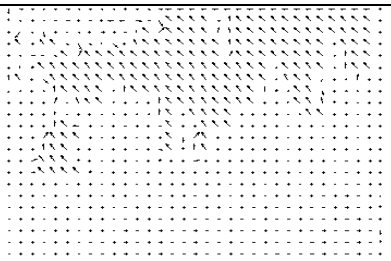
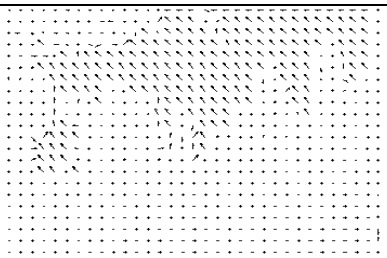
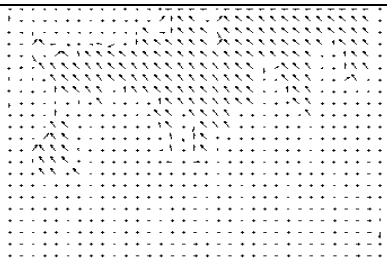
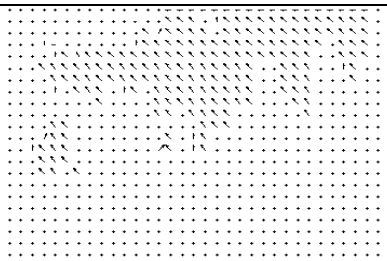
Images originaux	Images reconstruites	**Vecteurs de déplacement :**
 Frame 2	 Frame 2 reconstruite à partir du frame1 originale	 Entre les frames n° 1 et n° 2
 Frame 3	 Frame 3 reconstruite à partir du frame 2 reconstruite	 Entre les frames n° 2 et n° 3
 Frame 4	 Frame 4 reconstruite à partir du frame 3 reconstruite	 Entre les frames n° 3 et n° 4
 Frame 5	 Frame 5 reconstruite à partir du frame 4 reconstruite	 Entre les frames n° 4 et n° 5
 Frame 6	 Frame 6 reconstruite à partir du frame 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau III.3 : résultat sur la séquence 'flower' pour la recherche entre 50 blocs

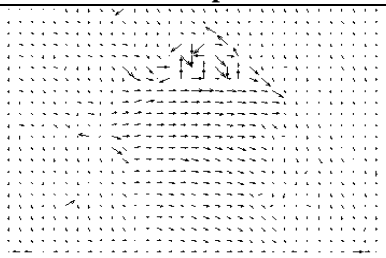
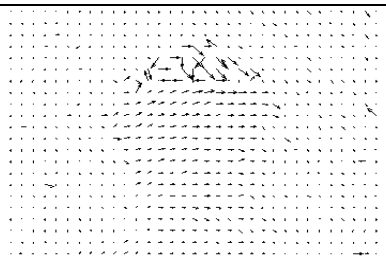
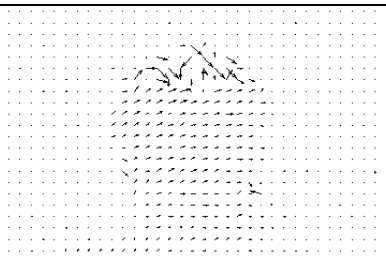
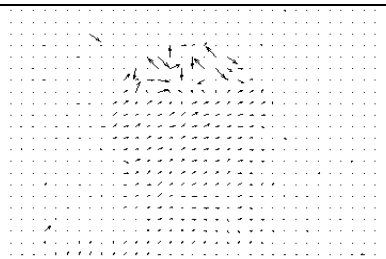
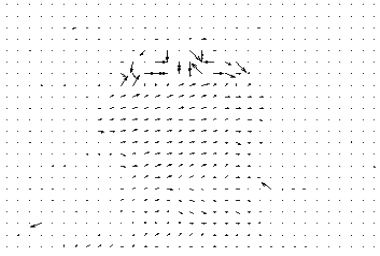
Images originaux	Image reconstruites	**Vecteurs de déplacement :**
 Frame 31	 Frame 31 reconstruite à partir du frame 30 originale	 Entre les frames n° 30 et 31
 Frame 32	 Frame 32 reconstruite à partir du frame 31 reconstruite	 Entre les frames n° 31 et 32
 Frame 33	 Frame 33 reconstruite à partir du frame 32 reconstruite	 Entre les frames n° 32 et 33
 Frame 34	 Frame 34 reconstruite à partir du frame 33 reconstruite	 Entre les frames n° 33 et 34
 Frame 35	 Frame 35 reconstruite à partir du frame 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.4: résultat sur la séquence 'formen' pour la recherche entre 50 blocs

3.4.2. Résultats de recherche entre 75 blocs aléatoires :

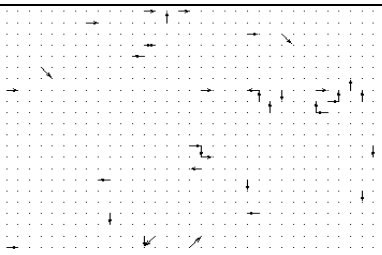
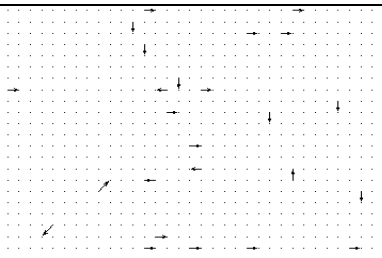
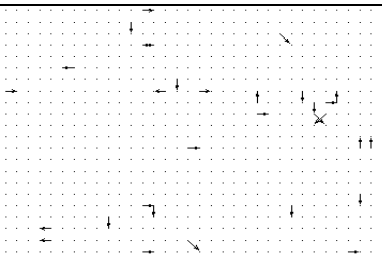
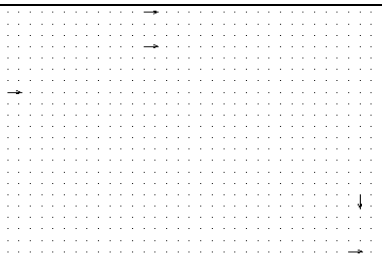
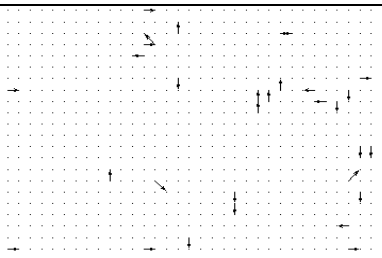
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 21	 Frame 21 reconstruite à partir du frame 20 originale	 Entre les frames n° 20 et 21
 Frame 22	 Frame 22 reconstruite à partir du frame 21 reconstruite	 Entre les frames n° 21 et 22
 Frame 23	 Frame 23 reconstruite à partir du frame 22 reconstruite	 Entre les frames n° 22 et 23
 Frame 24	 Frame 24 reconstruite à partir du frame 23 reconstruite	 Entre les frames n° 23 et 24
 Frame 25	 Frame 25 reconstruite à partir du frame 24 reconstruite	 Entre les frames n° 24 et 25

Tableau III.5 : résultat sur la séquence 'Akiyo' pour la recherche entre 75 blocs

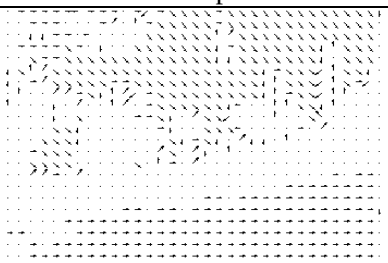
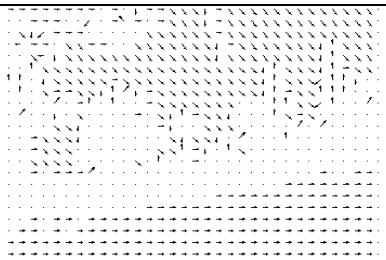
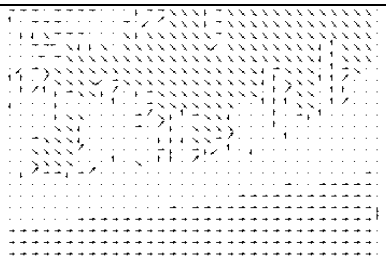
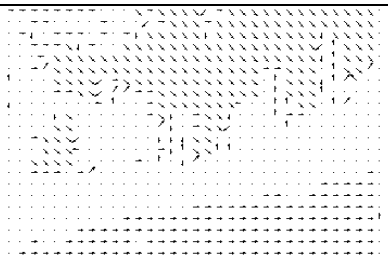
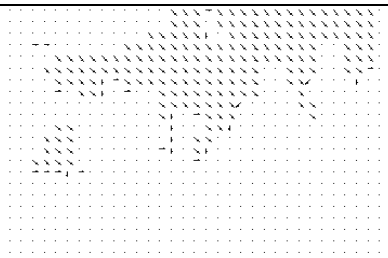
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 2	 Frame 2 reconstruite à partir du frame 1 originale	 Entre les frames n° 1 et n° 2
 Frame 3	 Frame 3 reconstruite à partir du frame 2 reconstruite	 Entre les frames n° 2 et n° 3
 Frame 4	 Frame 4 reconstruite à partir du frame 3 reconstruite	 Entre les frames n°3 et n° 4
 Frame 5	 Frame 5 reconstruite à partir du frame 4 reconstruite	 Entre les frames n° 4 et n° 5
 Frame 6	 Frame 6 reconstruite à partir du frame 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau III.6 : résultat sur la séquence 'flower' pour la recherche entre 75 blocs

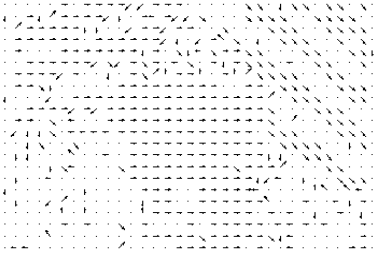
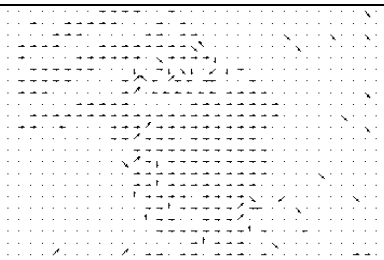
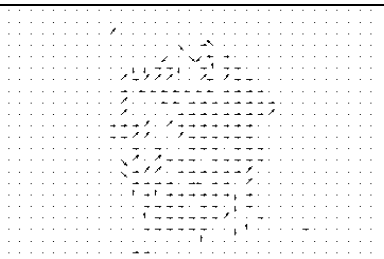
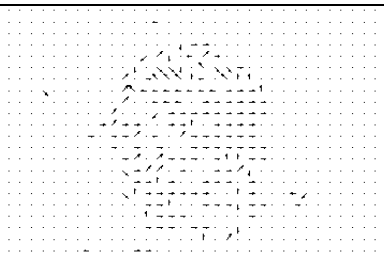
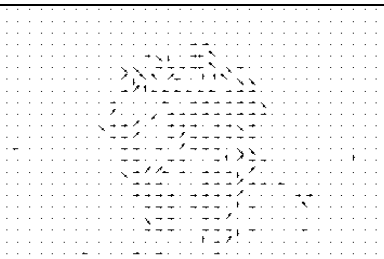
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 31	 Frame 31 reconstruite à partir du frame 30 originale	 Entre les frames n° 30 et 31
 Frame 32	 Frame 32 reconstruite à partir du frame 31 reconstruite	 Entre les frames n° 31 et 32
 Frame 33	 Frame 33 reconstruite à partir du frame 32 reconstruite	 Entre les frames n° 32 et 33
 Frame 34	 Frame 34 reconstruite à partir du frame 33 reconstruite	 Entre les frames n° 33 et 34
 Frame 35	 Frame 35 reconstruite à partir du frame 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.7 : résultat sur la séquence 'Formen' pour la recherche entre 75 blocs

3.4.3. Résultats de recherche entre 100 blocs aléatoires :

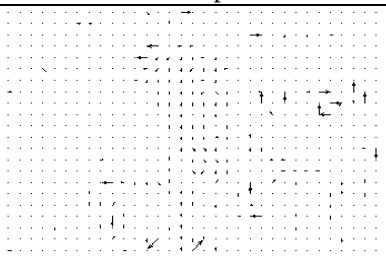
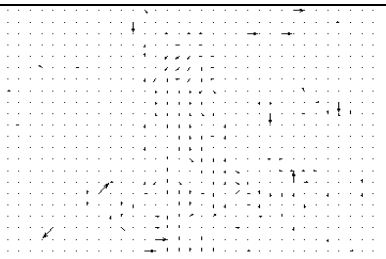
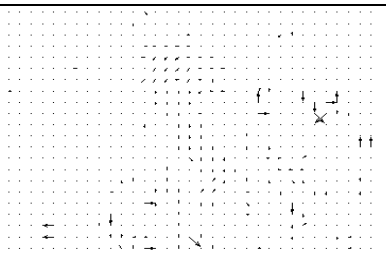
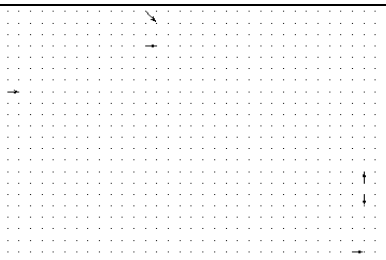
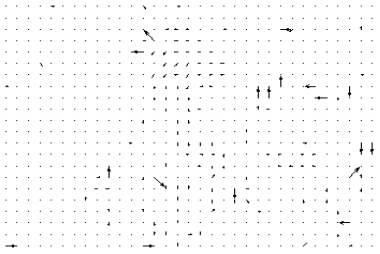
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 21	 Frame 21 reconstruite à partir du frame 20 originale	 Entre les frames n° 20 et 21
 Frame 22	 Frame 22 reconstruite à partir du frame 21 reconstruite	 Entre les frames n° 21 et 22
 Frame 23	 Frame 23 reconstruite à partir du frame 22 reconstruite	 Entre les frames n° 22 et 23
 Frame 24	 Frame 24 reconstruite à partir du frame 23 reconstruite	 Entre les frames n° 23 et 24
 Frame 25	 Frame 25 reconstruite à partir du frame 24 reconstruite	 Entre les frames n° 24 et 25

Tableau III.8 : résultat sur la séquence 'Akiyo' pour la recherche entre 100 blocs

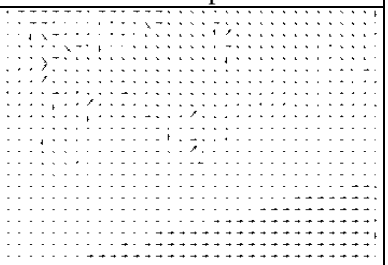
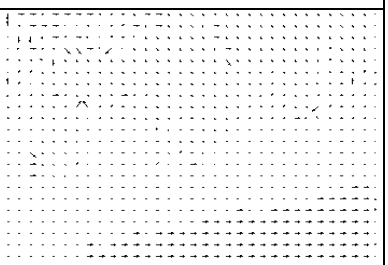
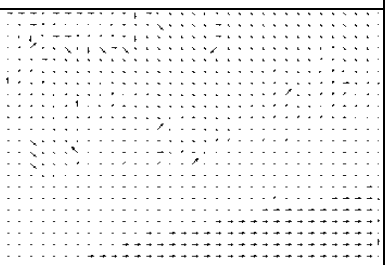
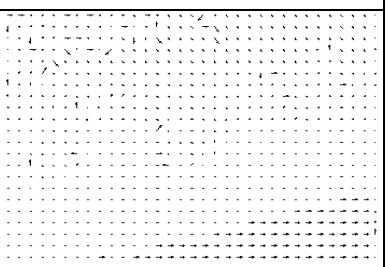
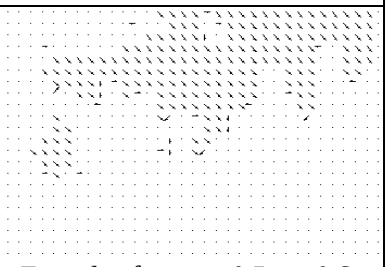
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 2	 Frame 2 reconstruite à partir du frame 1 originale	 Entre les frames n° 1 et n° 2
 Frame 3	 Frame 3 reconstruite à partir du frame 2 reconstruite	 Entre les frames n° 2 et n° 3
 Frame 4	 Frame 4 reconstruite à partir du frame 3 reconstruite	 Entre les frames n°3 et n° 4
 Frame 5	 Frame 5 reconstruite à partir du frame 4 reconstruite	 Entre les frames n° 4 et n° 5
 Frame 6	 Frame 6 reconstruite à partir du frame 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau III.9 : résultat sur la séquence 'flower' pour la recherche entre 100 blocs

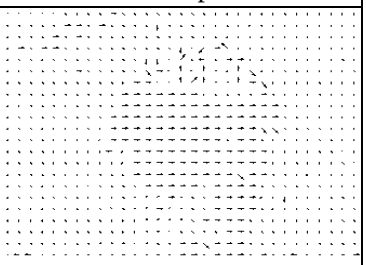
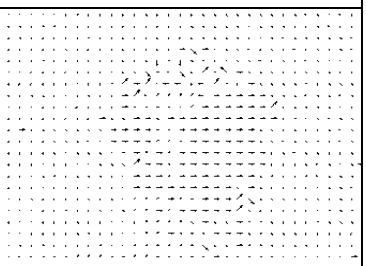
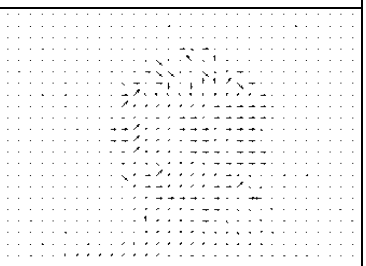
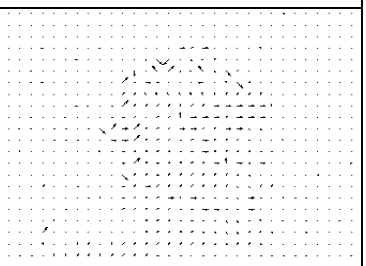
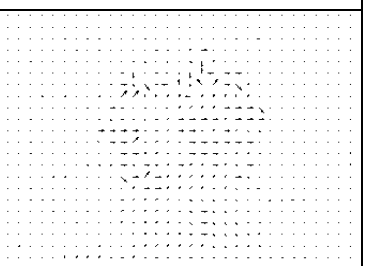
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 31	 Frame 31 reconstruite à partir du frame 30 originale	 Entre les frames n° 30 et 31
 Frame 32	 Frame 32 reconstruite à partir du frame 31 reconstruite	 Entre les frames n° 31 et 32
 Frame 33	 Frame 33 reconstruite à partir du frame 32 reconstruite	 Entre les frames n° 32 et 33
 Frame 34	 Frame 34 reconstruite à partir du frame 33 reconstruite	 Entre les frames n° 33 et 34
 Frame 35	 Frame 35 reconstruite à partir du frame 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.10 : résultat sur la séquence 'Formen' pour la recherche entre 100 blocs

3.4.4. Résultats de recherche entre 125 blocs aléatoires :

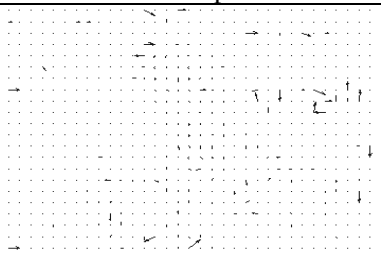
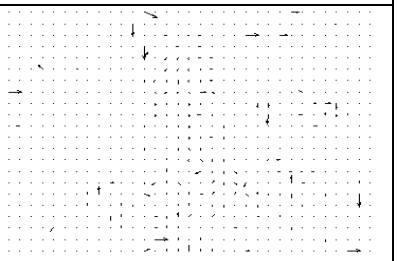
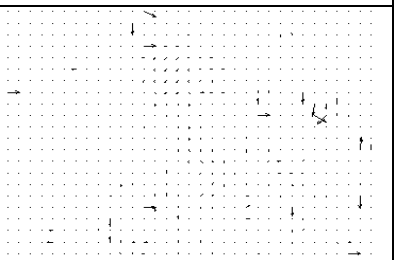
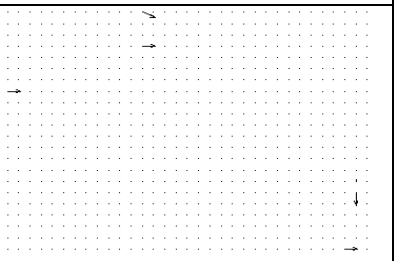
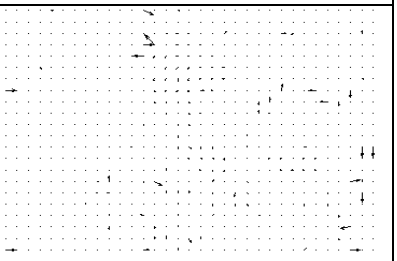
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 21	 Frame 21 reconstruite à partir du frame 20 originale	 Entre les frames n° 20 et 21
 Frame 22	 Frame 22 reconstruite à partir du frame 21 reconstruite	 Entre les frames n° 21 et 22
 Frame 23	 Frame 23 reconstruite à partir du frame 22 reconstruite	 Entre les frames n° 22 et 23
 Frame 24	 Frame 24 reconstruite à partir du frame 23 reconstruite	 Entre les frames n° 23 et 24
 Frame 25	 Frame 25 reconstruite à partir du frame 24 reconstruite	 Entre les frames n° 24 et 25

Tableau III.11 : résultat sur la séquence 'Akiyo' pour la recherche entre 125 blocs

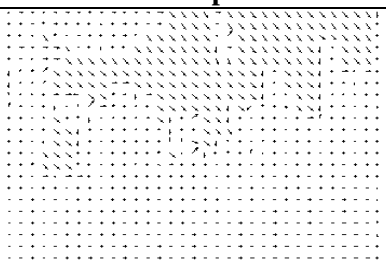
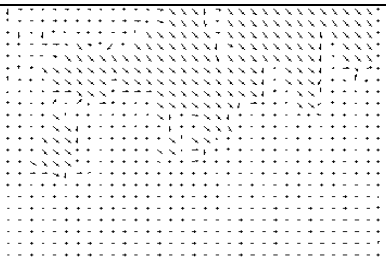
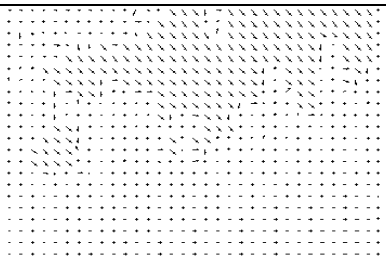
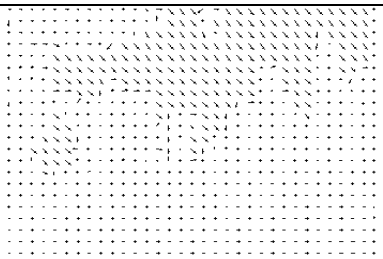
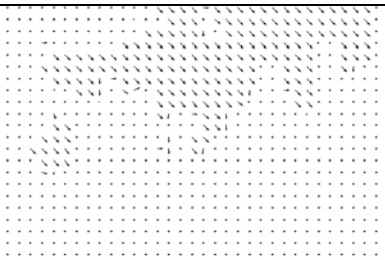
Images originaux	Images reconstruites	**Vecteurs de déplacement :**
 Frame 2	 Frame 2 reconstruite à partir du frame 1 originale	 Entre les frames n° 1 et n° 2
 Frame 3	 Frame 3 reconstruite à partir du frame 2 reconstruite	 Entre les frames n° 2 et n° 3
 Frame 4	 Frame 4 reconstruite à partir du frame 3 reconstruite	 Entre les frames n° 3 et n° 4
 Frame 5	 Frame 5 reconstruite à partir du frame 4 reconstruite	 Entre les frames n° 4 et n° 5
 Frame 6	 Frame 6 reconstruite à partir du frame 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau III.12 : résultat sur la séquence 'flower' pour la recherche entre 125 blocs

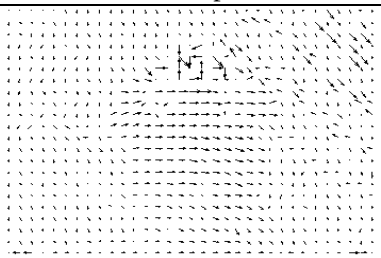
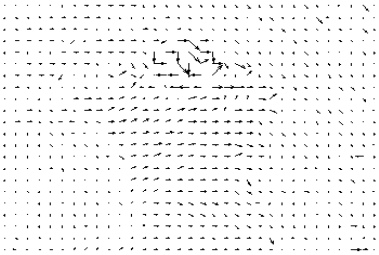
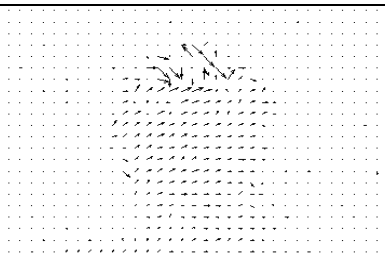
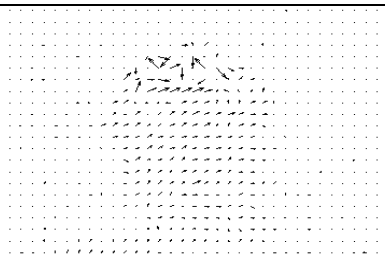
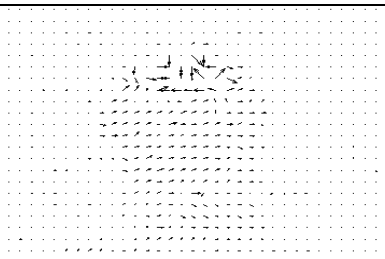
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 31	 Frame 31 reconstruite à partir du frame 30 originale	 Entre les frames n° 30 et 31
 Frame 32	 Frame 32 reconstruite à partir du frame 31 reconstruite	 Entre les frames n° 31 et 32
 Frame 33	 Frame 33 reconstruite à partir du frame 32 reconstruite	 Entre les frames n° 32 et 33
 Frame 34	 Frame 34 reconstruite à partir du frame 33 reconstruite	 Entre les frames n° 33 et 34
 Frame 35	 Frame 35 reconstruite à partir du frame 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.13 : résultat sur la séquence 'Formen' pour la recherche entre 125 blocs

3.4.5. Résultats de recherche entre 150 blocs aléatoires :

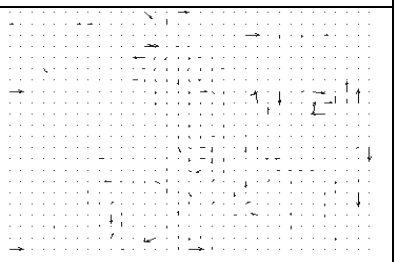
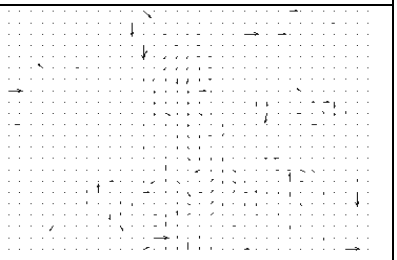
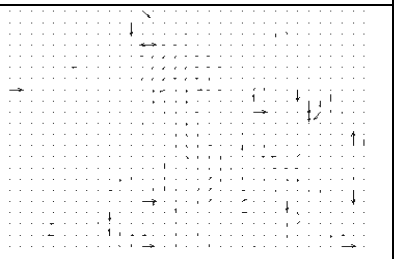
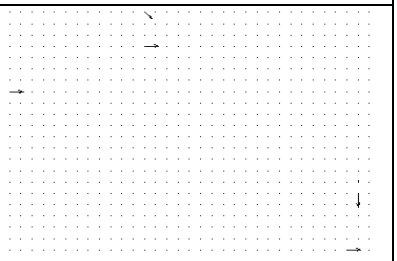
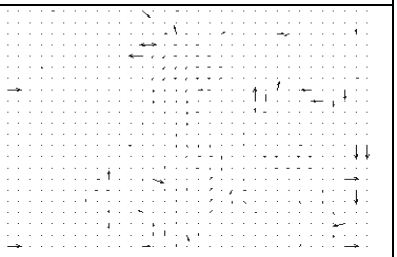
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 21	 Frame 21 reconstruite à partir du frame 20 originale	 Entre les frames n° 20 et 21
 Frame 22	 Frame 22 reconstruite à partir du frame 21 reconstruite	 Entre les frames n° 21 et 22
 Frame 23	 Frame 23 reconstruite à partir du frame 22 reconstruite	 Entre les frames n° 22 et 23
 Frame 24	 Frame 24 reconstruite à partir du frame 23 reconstruite	 Entre les frames n° 23 et 24
 Frame 25	 Frame 25 reconstruite à partir du frame 24 reconstruite	 Entre les frames n° 24 et 25

Tableau III.14: résultat sur la séquence 'Akiyo' pour la recherche entre 150 blocs

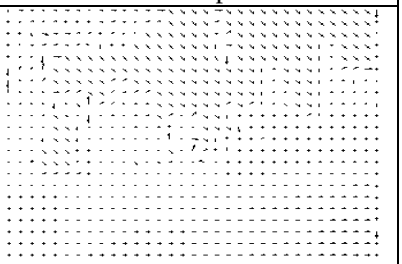
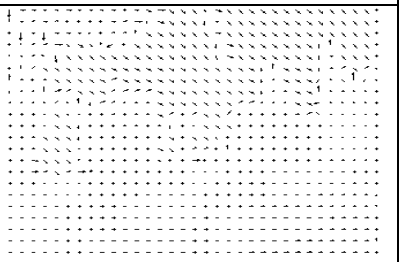
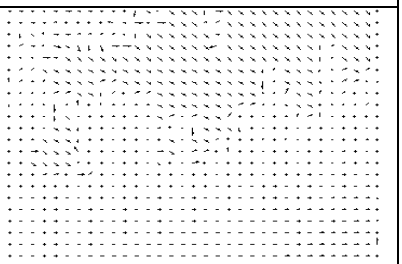
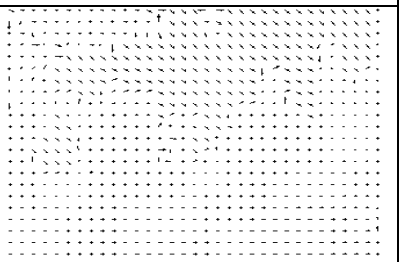
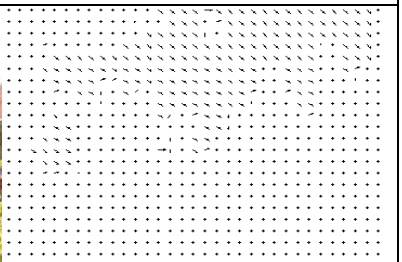
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 2	 Frame 2 reconstruite à partir du frame 1 originale	 Entre les frames n° 1 et n° 2
 Frame 3	 Frame 3 reconstruite à partir du frame 2 reconstruite	 Entre les frames n° 2 et n° 3
 Frame 4	 Frame 4 reconstruite à partir du frame 3 reconstruite	 Entre les frames n° 3 et n° 4
 Frame 5	 Frame 5 reconstruite à partir du frame 4 reconstruite	 Entre les frames n° 4 et n° 5
 Frame 6	 Frame 6 reconstruite à partir du frame 5 reconstruite	 Entre les frames n° 5 et n° 6

Tableau III.15 : résultat sur la séquence 'flower' pour la recherche entre 150 blocs

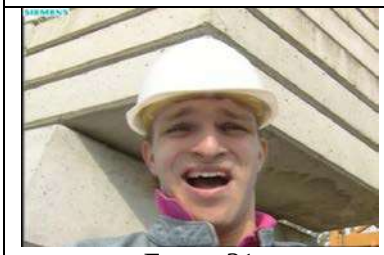
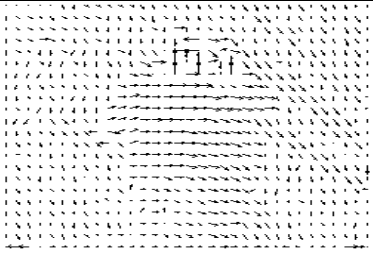
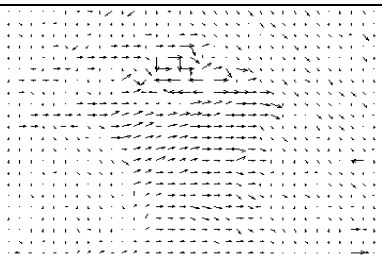
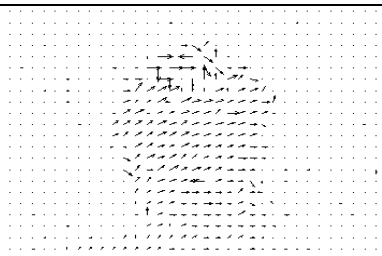
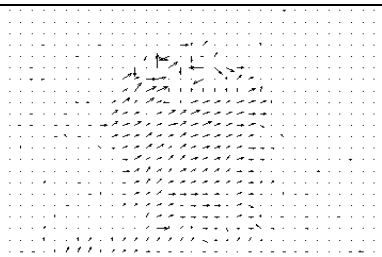
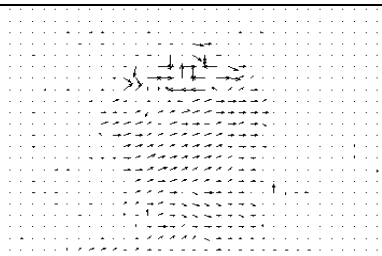
Images originaux	Images reconstruites	Vecteurs de déplacement
 Frame 31	 Frame 31 reconstruite à partir du frame 30 originale	 Entre les frames n° 30 et 31
 Frame 32	 Frame 32 reconstruite à partir du frame 31 reconstruite	 Entre les frames n° 31 et 32
 Frame 33	 Frame 33 reconstruite à partir du frame 32 reconstruite	 Entre les frames n° 32 et 33
 Frame 34	 Frame 34 reconstruite à partir du frame 33 reconstruite	 Entre les frames n° 33 et 34
 Frame 35	 Frame 35 reconstruite à partir du frame 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.16 : résultat sur la séquence 'Formen' pour la recherche entre 150 blocs

3.5. L'analyse des résultats :

Afin d'améliorer les observations et de clarifier les résultats obtenues; en plus des frames reconstruites on a fait des tableaux qui nous indique le temps de calculs (en secondes) et le rapport signal sur bruit **PSNR** (*Pic Signal Noise Rate*) pour chaque algorithme et pour les trois séquences. Le rapport Signal/Bruit pour chaque image est obtenu par les relations suivantes **[25]** :

$$PSNR = 20\log_{10}(P^2 / \sqrt{MSE}).$$
$$MSE = (1/N) \times \sum_N X_i - X_i^{'})^2.$$

Où :

MSE : est l'erreur quadratique moyenne.

$\mathbf{X_i}$ **:** les éléments de la frame originale.

$\mathbf{X_i^{'}}$ **:** les éléments de la frame reconstruite.

N : la hauteur de la frame × la largeur de la frame (en pixels),

dans notre cas 352 × 528=185 856.

P : valeur maximale de chaque élément de la frame**,**

dans notre cas 255, car on a des valeurs sur 8 bits.

3.5.1. Pour la séquence 'Akiyo' :

Les points utilisés dans l'algorithme génétique		Frame 21	Frame 22	Frame 23	Frame 24	Frame 25
Recherche de 50 points	Temps de calcule(s)	19 ,99	20,07	20,06	20,10	20,04
	PSNR	31,64	30,96	29,51	29 ,05	28,12
Recherche de 75 points	Temps de calcule(s)	30,01	30 ,07	30,04	29 ,95	29,98
	PSNR	32,80	31 ,85	30,12	29,99	29,07
Recherche de 100 points	Temps de calcule(s)	40,15	40,17	40,15	40,13	40 ,20
	PSNR	33,58	32,60	31,28	30,52	29,70

Recherche de 125 points	Temps de calcule(s)	49,99	49,96	49,98	50,40	50,48
	PSNR	34,64	33,98	32,38	31,87	30,57
Recherche de 150 points	Temps de calcule(s)	60 ,27	60,37	60 ,32	60,32	60,20
	PSNR	35,02	35,09	33,02	32,23	30,69

Tableau III.17 : temps de calcule et PSNR pour la sequence 'Akiyo'

- **Pour une recherche par 50 points :**

L'image se reconstruit dans un temps court avec un PSNR moyen et une image a une déformation très élevée pour toutes les frames.

- . **Pour une recherche par 75 points :**

L'image se reconstruit dans un temps moyen avec un PSNR moyen et une image trop déformée pour toutes les frames.

- **Pour une recherche par 100 points :**

Un temps de calcule plus ou moins élevé, un PSNR élevé et l'image reconstruite est déformée.

- **Pour une recherche par 125 points :**

Un temps de calcule élevé, un PSNR élevé et l'image déformée pour toutes les frames.

- **Pour une recherche par 150 points :**

Le temps de calcule le plus élevé ainsi que le PSNR et une image moins déformée.

On constate :
Pour cette séquence, la recherche dans 150 a donné un résultat acceptable partiellement avec un temps de calcule élevé et une deformation au niveau des fremes de l'image.

3.5.2. Pour la séquence 'flower' :

Les points utilises dans l'algorithme génétique		Frame 2	Frame 3	Frame 4	Frame 5	Frame 6
Recherche de 50 points	Temps de calcule(s)	20,09	20 ,03	20,09	20,21	20,18
	PSNR	19,30	20,56	18 ,74	19,43	19,07
Recherche de 75 points	Temps de calcule(s)	30.18	30.23	30.21	30.23	30.27
	PSNR	19,75	21,09	19,42	20,20	19,98
Recherche de 100 points	Temps de calcule(s)	40,52	40,40	40,49	40,24	40,32
	PSNR	19,72	22,87	19,37	21,09	20,73
Recherche de 125 points	Temps de calcule(s)	50,37	50,32	50,54	50,45	50,41
	PSNR	19,71	22,99	19,54	21,44	21,20
Recherche de 150 points	Temps de calcule(s)	61,51	60,88	60,76	59,87	60,02
	PSNR	19,91	23,82	19,78	21 ,94	21,66

Tableau III.18 : temps de calcule et PSNR pour la sequence 'flower'

- **Pour une recherche par 50 points :**

L'image se reconstruit dans un temps le plus court avec une mauvaise valeur de PSNR et une image a une déformation pour toutes les frames.

- **Pour une recherche par 75 points :**

Le temps de calcule court, une mauvaise valeur de PSNR et une image déformée.

- **Pour une recherche par 100 points :**

Le temps de calcule court, une mauvaise valeur de PSNR et une image moins déformée.

- **Pour une recherche par 125 points :**

Un temps de calcule un peu long, une mauvaise valeur de PSNR avec une déformation lingère au niveau de l'image.

- **Pour une recherche par 150 points :**

Un temps de calcule le plus élevé, une mauvaise valeur de PSNR avec une image un peu déformée.

On constate :
Pour cette séquence, la recherche dans 125 points a donné un résultat pareil que la recherche dans 150 points pour le critère de déformation mais ce dernier a eu un temps de calcule élevé que le premier, alors la recherche dans 125 points la plus performante.

3.5.3. Pour la séquence 'Formen' :

Les points utilises dans l'algorithme génétique		Frame 21	Frame 22	Frame 23	Frame 24	Frame 25
Recherche de 50 points	Temps de calcule(s)	20,68	20,71	20,73	20,70	20,74
	PSNR	24,48	27,60	26 ,19	26,44	26, 34
Recherche de 75 points	Temps de calcule(s)	31,09	31,04	31 ,05	31,05	31,05
	PSNR	24,54	27,50	26,25	26,79	25,70
Recherche de 100 points	Temps de calcule(s)	40,37	40,23	40,40	40,29	40,27
	PSNR	24,72	28,46	26,67	27,00	25,89
Recherche de 125 points	Temps de calcule(s)	50 ,41	50,43	50,38	50,57	50,37
	PSNR	24,68	28 ,64	26,87	25,43	27,25
Recherche de 150 points	Temps de calcule(s)	60,10	60,18	59,96	59,82	59,98
	PSNR	24,77	28,61	26,79	25,35	27,29

Tableau III.19 : temps de calcule et PSNR pour la séquence 'Formen'

- **Pour une recherche par 50 points :**

Un temps de calcule le plus court, un PSNR moyen et un peu de déformation.

- **Pour une recherche par 75 points :**

Un temps de calcule court, un PSNR moyen et un peu de déformation.

- **Pour une recherche par 100 points :**

Un temps de calcule moyen, un PSNR moyen et une image moins déformée.

- **Pour une recherche par 125 points :**

Un temps de calcule un peu élevé, un PSNR moyen et une image avec une déformation légère.

- **Pour une recherche par 150 points :**

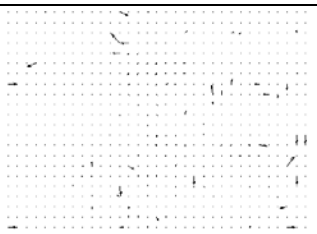
Un temps de calcule le plus élevé, un PSNR moyen et une image avec une déformation légère

On constate :

Pour cette séquence, la recherche dans 125 points est la plus performante car elle a donné une image avec une déformation légère qui ressemble l'image donnée par la recherche dans 150 points mais dans un temps convenable.

3.6. Quelques résultats des algorithmes précédents (FS, TSS, NTSS, FSS, DS , MinS):

3.6.1. Résultats de l'algorithme FS :

Les sequences	Frame originale	Frames reconstruites	Vecteurs de déplacement
Pour la séquence 'Akiyo'	 Frame n° 25	 Frame n° 25 reconstruite à partir de frame n° 24 reconstruite	 Entre les frames n° 24 et 25

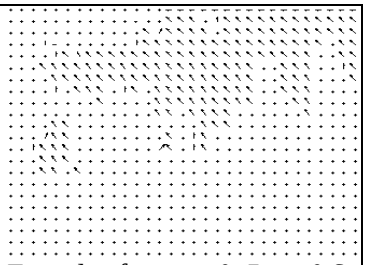
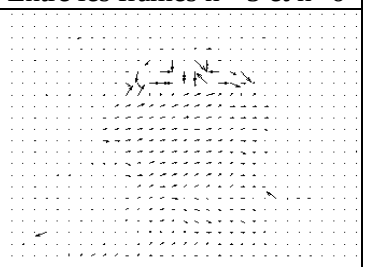
Pour la séquence 'flower'	 Frame n° 6	 Frame n° 6 reconstruite à partir de la frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6
Pour la séquence 'Formen'	 Frame n° 35	 Frame n° 35 reconstruite à partir de la frame n° 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.20 : résultats de l'algorithme FS sur toutes les séquences

3.6.2 Résultats de l'algorithme TSS :

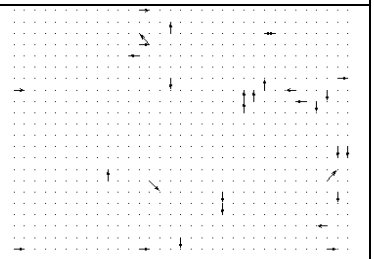
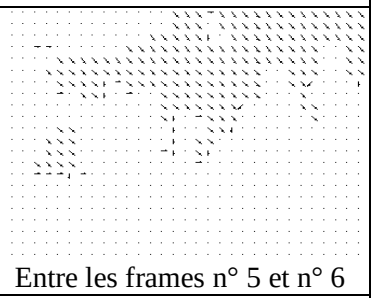
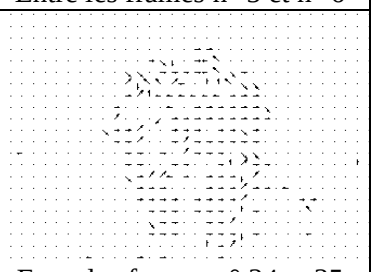
Les sequences	Frame originale	Frames reconstruites	Vecteurs de déplacement
Pour la séquence 'Akiyo'	 Frame n° 25	 Frame n° 25 reconstruite à partir de la frame n° 24 reconstruite	 Entre les frames n° 24 et 25
Pour la séquence 'flower'	 Frame n° 6	 Frame n° 6 reconstruite à partir de la frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6
Pour la séquence 'Formen'	 Frame n° 35	 Frame n° 35 reconstruite à partir de la frame n° 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.21 : résultats de l'algorithme TSS sur toutes les séquences

3.6.3 : Résultats de l'algorithme NTSS :

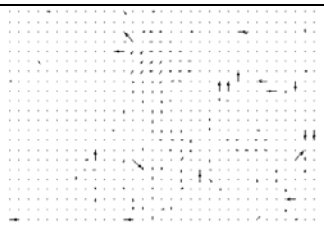
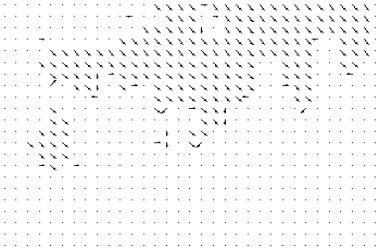
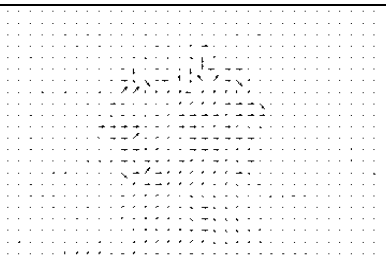
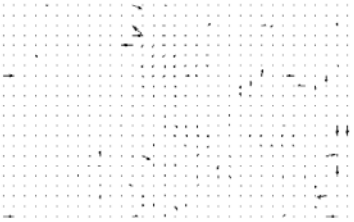
Les sequences	Frame originale	Frames reconstruites	Vecteurs de déplacement
Pour la séquence 'Akiyo'	 Frame n° 25	 Frame n° 25 reconstruite à partir de la frame n° 24 reconstruite	 Entre les frames n° 24 et 25
Pour la séquence 'flower'	 Frame n° 6	 Frame n° 6 reconstruite à partir de la frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6
Pour la séquence 'Formen'	 Frame n° 35	 Frame n° 35 reconstruite à partir de la frame n° 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.22 : résultats de l'algorithme NTSS sur toutes les séquences

3.6.4 : Résultats de l'algorithme FSS :

Les sequences	Frame originale	Frames reconstruites	Vecteurs de déplacement
Pour la séquence 'Akiyo'	 Frame n° 25	 Frame n° 25 reconstruite à partir de frame n° 24 reconstruite	 Entre les frames n° 24 et 25

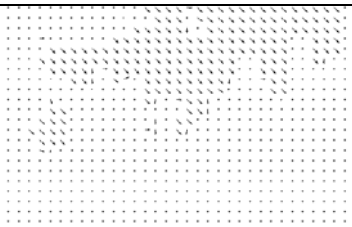
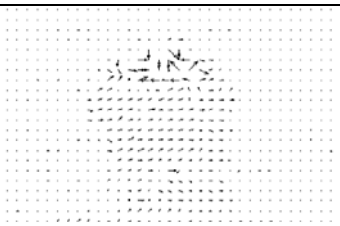
Pour la séquence 'flower'	 Frame n° 6	 Frame n° 6 reconstruite à partir de la frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6
Pour la séquence 'Formen'	 Frame n° 35	 Frame n° 35 reconstruite à partir de la frame n° 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.23 : résultats de l'algorithme FSS sur toutes les séquences

3.6.5 : résultats de l'algorithme DS :

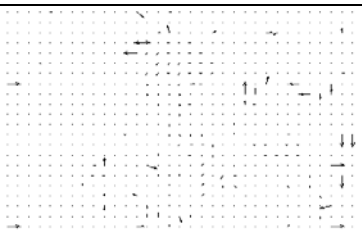
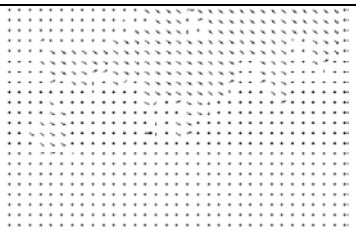
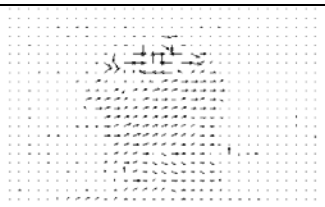
Les sequences	Frame originale	Frames reconstruites	Vecteurs de déplacement
Pour la séquence 'Akiyo'	 Frame n° 25	 Frame n° 25 reconstruite à partir de frame n° 24 reconstruite	 Entre les frames n° 24 et 25
Pour la séquence 'flower'	 Frame n° 6	 Frame n° 6 reconstruite à partir de la frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6
Pour la séquence formation' Formen'	 Frame n° 35	 Frame n° 35 reconstruite à partir de la frame n° 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.24 : résultats de l'algorithme DS sur toutes les séquences

3.6.6 : résultats de l'algorithme MinS :

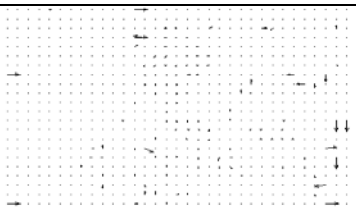
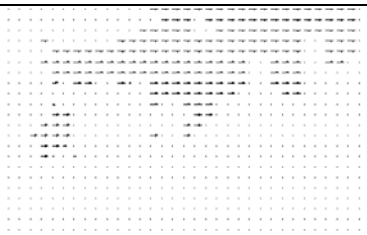
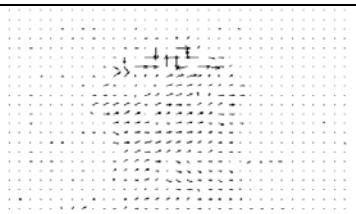
Les séquences	Frame originale	Frames reconstruites	Vecteurs de déplacement
Pour la séquence 'Akiyo'	 Frame n° 25	 Frame n° 25 reconstruite à partir de frame n° 24 reconstruite	 Entre les frames n° 24 et 25
Pour la séquence 'flower'	 Frame n° 6	 Frame n° 6 reconstruite à partir de la frame n° 5 reconstruite	 Entre les frames n° 5 et n° 6
Pour la séquence 'Formen'	 Frame n° 35	 Frame n° 35 reconstruite à partir de la frame n° 34 reconstruite	 Entre les frames n° 34 et 35

Tableau III.25 : résultats de l'algorithme MinS sur toutes les séquences

3.6.7. Résultats sur la séquence 'Akiyo' frame 25 :

Algorithmes de recherche	**temps de calcule**	**PSNR**
FS	114.97	32,22
TSS	10.90	32,20
NTSS	7.59	32,33
FSS	7.02	28,88
DS	9.34	32,34
MinS	2.87	32,30

Tableau III.26: Résultats des algorithmes sur la séquence 'Akiyo' frame 25

4.6.8. Résultats sur la séquence 'flower' frame 6:

Algorithmes de recherche	**temps de calcule**	**PSNR**
FS	49.88	22,19
TSS	4.89	22,15
NTSS	4.68	22,12
FSS	3.79	17,43
DS	4.27	18,85
MinS	3.55	22,17

Tableau III.27 : Résultats de algorithmes sur la séquence 'flower' frame 6

3.6.9. Résultats sur la séquence 'Formen' frame 35:

Algorithmes de recherche	**temps de calcule**	**PSNR**
FS	113.25	22,19
TSS	11.04	22,15
NTSS	9.39	22,12
FSS	7.58	17,43
DS	9.60	18,85
MinS	4.33	22,17

Tableau III.28: Résultats de algorithmes sur la séquence 'Formen' frame 35

3.7. Comparaison entre les résultats des algorithmes classiques et celle de l'algorithme génétique :

Les résultats des algorithmes génétique se sont des résultats, qui différent de séquence à l'autre tel que :

- Les résultats obtenus par l'application des algorithmes génétiques sur la séquence 'flower' se sont les bons résultats selon les critères de la déformation des frames de la séquence tel que on a des images acceptables qui ont n'influés après l'information dans un temps moyen .

- Les séquences 'formen 'a un plus de déformation que la séquence 'flower' durant un temps acceptable.

- Les résultats de la séquence 'Akio 'sont des mauvaise résultats car on a des images déformé énormément au niveau des frames même si on augmentait le temps de calcule (augmenter les nombres de block à comparer) .

Pour les algorithmes classiques les résultats sont comme suite **: [13]**

Les algorithmes	Temps de calcul	PSNR	Déformations au niveau des frames reconstruites
FS	n° 04	n° 01	n° 02
TSS	n° 03	n° 04	n° 06
NTSS	n° 03	n° 03	n° 05
FSS	n° 02	n° 03	n° 03
DS	n° 02	n° 02	n° 01
MIN	n° 01	n° 01	n° 04

Tableau III.29 : performance des algorithmes.

D'après le tableau précédent il est clair et net que l'algorithme DS de recherche en diamant est le plus performant car il présente le meilleur compromis entre le temps de calcul et le PSNR, ainsi que l'absence des déformations au niveau des frames reconstruites.

Donc, on peut classer les algorithmes selon ses performances :

- L'algorithme DS est le plus performant, PSNR élevé et temps de calcul presque le plus court, et pas de déformation sur les trois séquence.
- 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 mais son temps de calcul est très élevé.
- L'algorithme FSS a un PSNR élevé et un temps de calcul moyen, et peu de déformations des frames.
- L'algorithme NTSS a un PSNR moyen et un temps de calcul moyen aussi, et des déformations aux niveaux des frames.
- L'algorithme TSS a un PSNR le moins important et un temps de calcul le plus long, et il y a trop de déformations aux niveaux de ces des frames résultante.

3.9. Conclusion :

Après l'application des algorithmes génétiques en l'estimation de mouvement, on a eu des résultats acceptables considérant le temps de calcule, le PSNR ainsi que la déformation des images.

Considérant le critère de déformation, les bons résultats sont apparus notamment sur la séquence 'flower' tel qu'on n'a observé des petites déformations, des résultats moyen pour la séquence 'Formen' et des mauvais résultats sur la séquence 'Akiyo'.

Par rapport aux deux critères restant (temps de calcule et PSNR) on a eu des résultats pas tellement acceptables par ce que le temps de calcule a été un peu long par rapport aux algorithmes classiques et un PSNR imparfait.

Ces résultats restent une application directe des algorithmes génétiques qui peuvent être améliorés car les résultats de l'algorithme DS a donné des résultats mieux que les résultats des de l'algorithme utilisée dans cette étude.

Conclusion générale

Conclusion générale :

Ce projet nous a permis d'approfondir nos connaissances dans le domaine du traitement d'image et plus précisément dans le domaine estimation du mouvement par algorithme génétique.

Dans cette étude,La représentation des images vidéo est un des éléments essentiels des applications multimédias, comme dans la plupart des systèmes de communication. L'estimation de mouvement est une essentielle étape pour la compression d'image vidéo, qui permet de réduire énormément l'espace mémoire nécessaire au stockage d'une image, le but est d'estimer le mouvement des objets contenus dans une vidéo sous forme d'un champ dense de vecteurs, les applications d'estimation du mouvement sont nombreuses, nous pouvons noter par exemplela compensation du mouvement utilisée en compression vidéo (normes MPEG2, MPEG4, H264), etc. et la segmentation d'objets en mouvement.

Le travail que nous avons effectué est l'objectif qu'il faut réaliser un programme capable d'amélioré la qualité d'image, on a utilisé plusieurs algorithmes pour obtenue des résultats acceptables considérant le temps de calcule, le PSNR ainsi que la déformation des images.

De point de vus on a remarqué que les résultats de PSNR 'AKIO'Pour cette séquence, la recherche dans 150 a donné un résultat acceptable partiellement avec un temps de calcule élevé et une deformation au niveau des fremes de l'image. La séquence 'flower' la recherche dans 125 points a donné un résultat pareil que la recherche dans 150 points pour le critère de déformation mais ce dernier a eu un temps de calcule élevé que le premier, alorsla recherche dans 125 points la plus performante.et la séquence 'formen', la recherche dans 125 points est la plus performante car elle a donné une image avec une déformation légère qui ressemble l'image donnée par la recherche dans 150 points mais dans un temps convenable.

Nous avons travaillé de développer un algorithme de l'estimation de mouvement d'image vidéo dans un minimum de temps et un PSNR élevé, tout cela contribuer la bonne qualité d'image.

Mais l'algorithme de recherche en Dimant 'DS' demeure le meilleur algorithme pour parvenir à une image non déformée dans un temps bref et un PSNR élevé.

L'algorithme génétique

```
clc
%88888888888888888888888888888888888888888%
%8888      Genetic ALGORITHM Search    8888%
%8888   la lecture des deux matrices   8888%
%88888888888888888888888888888888888888888%
mat1=input('entrer le nom de  l image source.....  ');RETURN;
image1=imread(mat1,'jpg');
matrice1=double(image1);
sizemat1=size(matrice1);
longuermat1=sizemat1(1,1);
largeurmat1=sizemat1(1,2);
mat2=input('entrer le nom de la nouvelle image.....  ');RETURN;
image2=imread(mat2,'jpg');
matrice2=double(image2);
sizemat2=size(matrice2);
longeurmat2=sizemat2(1,1);
largeurmat2=sizemat2(1,2);
tempsdedebut=cputime;
%selection des zones de recherche%
M=longuermat1;
N=largeurmat1;
vectpositionfinal=[];
decalage=1;
for compteurligne2=1:16:longeurmat2-14
   for compteurcolonne2=1:16:largeurmat2-14

mat16de2(1:16,1:16)=matrice2(compteurligne2:compteurligne2+15,compteurcolonne2:c
ompteurcolonne2+15,2);
      matderecherche=[];
      if     (compteurligne2==1) & (compteurcolonne2==1)

matderecherche(1:24,1:24)=matrice1(compteurligne2:compteurligne2+23,compteurcolon
ne2:compteurcolonne2+23,2);
        di=0;dj=0;
      elseif (compteurligne2==1) & (compteurcolonne2>1) & (compteurcolonne2<N-15)

matderecherche(1:24,1:32)=matrice1(compteurligne2:compteurligne2+23,compteurcolon
ne2-8:compteurcolonne2+23,2);
        di=0;dj=8;
      elseif (compteurligne2==1) & (compteurcolonne2==N-15)

matderecherche(1:24,1:24)=matrice1(compteurligne2:compteurligne2+23,compteurcolon
ne2-8:N,2);
        di=0;dj=8;
```

```
      elseif (compteurligne2>1) & (compteurligne2<M-15) & (compteurcolonne2==1)
        matderecherche(1:32,1:24)=matrice1(compteurligne2-
8:compteurligne2+23,1:compteurcolonne2+23,2);
        di=8;dj=0;
      elseif (compteurligne2==M-15) & (compteurcolonne2==1)
        matderecherche(1:24,1:24)=matrice1(compteurligne2-
8:M,1:compteurcolonne2+23,2);
        di=8;dj=0;
      elseif (compteurligne2>1) & (compteurligne2<M-17) & (compteurcolonne2==N-15)
        matderecherche(1:32,1:24)=matrice1(compteurligne2-
8:compteurligne2+23,compteurcolonne2-8:N,2);
        di=8;dj=8;
      elseif (compteurligne2==M-15) & (compteurcolonne2>1) & (compteurcolonne2<N-
15)
        matderecherche(1:24,1:32)=matrice1(compteurligne2-8:M,compteurcolonne2-
8:compteurcolonne2+23,2);
        di=8;dj=8;
      elseif (compteurligne2==M-15) & (compteurcolonne2==N-15)
        matderecherche(1:24,1:24)=matrice1(compteurligne2-8:M,compteurcolonne2-
8:N,2);
        di=8;dj=8;
      elseif (compteurligne2>1) & (compteurligne2<M-15) & (compteurcolonne2>1) &
(compteurcolonne2<N-15)
        matderecherche(1:32,1:32)=matrice1(compteurligne2-
8:compteurligne2+23,compteurcolonne2-8:compteurcolonne2+23,2);
        di=8;dj=8;
      end;
        sizematderecherche   =size(matderecherche);
        lignematderecherche  =sizematderecherche(1,1);
        colonnematderecherche=sizematderecherche(1,2);
        sizemat16de2   =size(mat16de2);
        lignemat16de2  =sizemat16de2(1,1);
        colonnemat16de2=sizemat16de2(1,2);
        comsom=1;
        commin=1;
        vecsom=[];
        vecmin=[];
        vpmin=[];
        for h=1:125
        ii=randi([1 lignematderecherche-lignemat16de2+1],1);
           jj=randi([1 colonnematderecherche-colonnemat16de2+1],1);
           c(1:16,1:16) = matderecherche(ii:ii+15,jj:jj+15);
           sum=0;
             for r=1:lignemat16de2
               for d=1:colonnemat16de2
                 s=abs(mat16de2(r,d)-c(r,d));
```

```
                sum=sum+s;
              end;
            end;
           vecsom(comsom)=sum;
           comsom=comsom+1;
           vecsom(comsom)=ii-di-1;
           comsom=comsom+1;
           vecsom(comsom)=jj-dj-1;
           comsom=comsom+1;
           vecmin(commin)=sum;
           commin=commin+1;
       end
%recherche de minimum ds vecmin qui est le minimum%
minimum=min(vecmin);
%chargement des minimums ds un vecteur vpmin%
compmin=1;
            for g=1:(lignematderecherche-lignemat16de2+1)*(colonnematderecherche-
colonnemat16de2+1)
              if vecmin(g)==minimum
                vpmin(compmin)=g;
                compmin=compmin+1;
                break;
              end;
            end;
 %chargement des numeros des lignes et colonnes des minimums(i.e debut%
 %de la matrice recherchée) ds un vecteur vectpositionfinal%
               for h=1:compmin-1
                 i1=vecsom(vpmin(h).*3-1);
                 j1=vecsom(vpmin(h).*3);
              end;
              vectpositionfinal(decalage,1)=i1;
              vectpositionfinal(decalage,2)=j1;
              vecvec(decalage)=compmin;
              decalage=decalage+1;
           end;
  end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%
%%%%%%%%%%%        Temps De Calcul        %%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%
tempsdefin=cputime;
tempsdecalcul=tempsdefin-tempsdedebut
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%
%%%%%%%%%%%  Ecriture De La Nouvelle Image  %%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mat3=input('entrer le nom de  l image source..................  ');RETURN;
image3=imread(mat3,'jpg');
matrice3=double(image3);
mmm=1;
for comi=1:16:M-10
   for comj=1:16:N-10
new(comi:comi+15,comj:comj+15,1:3)=image3(comi+vectpositionfinal(mmm,1):comi+
vectpositionfinal(mmm,1)+15,comj+vectpositionfinal(mmm,2):comj+vectpositionfinal(m
mm,2)+15,1:3);
       mmm=mmm+1;
   end;
end;
imagesortie=uint8(new);
imwrite(imagesortie,[mat2 'GAS125.jpg'],'JPG');
imshow(imagesortie)
exe=input('executer shema des vecteurs de déplacement?...........1 si oui 0 sinon
');RETURN;
if exe==1
mm=1;
for hh=1:16:M-10
   for kk=1:16:N-10
       vectpositioninitial(mm,1)=hh;
       vectpositioninitial(mm,2)=kk;
       vetpositionfinal(mm,1)=vectpositionfinal(mm,1)+hh;
       vetpositionfinal(mm,2)=vectpositionfinal(mm,2)+kk;
       mm=mm+1;
   end;
end;
pp=vetpositionfinal-vectpositioninitial;
max1=(M/16)*(N/16);
for compt=1:max1
   py1(compt)=pp(compt,1);
   px1(compt)=pp(compt,2);
end;
im=M/16;
for ii=1:(M/16)
   px(im,1:(N/16))=px1((N/16)*(ii-1)+1:(N/16)*(ii-1)+(N/16));
   py(im,1:(N/16))=-py1((N/16)*(ii-1)+1:(N/16)*(ii-1)+(N/16));
   im=im-1;
end;
x=1:16:N-10;
y=1:16:M-10;
quiver(x,y,px,py,1);
else
end;
```

L’algorithme de calcule de PSNR

```
mat1=input('entrer le nom de l image originale.....');RETURN;
image1=imread(mat1,'jpg');
matrice1=double(image1);
sizemat1=size(matrice1);
lignemat1=sizemat1(1,1);
colonemat1=sizemat1(1,2);
mat2=input('entrer le nom de l image reconstruite.......');RETURN;
image2=imread(mat2,'jpg');
matrice2=double(image2);
sizemat2=size(matrice2);
lignemat2=sizemat2(1,1);
colonemat2=sizemat2(1,2);
MSE1=0;
MSE2=0;
MSE3=0;

matdiff1(1:352,1:528)=(matrice2(:,:,1)-matrice1(:,:,1));
matdiff2(1:352,1:528)=(matrice2(:,:,2)-matrice1(:,:,2));
matdiff3(1:352,1:528)=(matrice2(:,:,3)-matrice1(:,:,3));

for ii=1:352
    for jj=1:528
    MSE1=MSE1+(matdiff1(ii,jj)^2);
    end;
end;
MSE1=MSE1/(352*528);
for ii=1:352
    for jj=1:528
    MSE2=MSE2+(matdiff2(ii,jj)^2);
    end;
end;
MSE2=MSE2/(352*528);
for ii=1:352
    for jj=1:528
    MSE3=MSE3+(matdiff3(ii,jj)^2);
    end;
end;
MSE3=MSE3/(352*528);
MSE=(MSE1+MSE2+MSE3)/3;
PSNR1=20*log10(255/sqrt(MSE1));
PSNR2=20*log10(255/sqrt(MSE2));
PSNR3=20*log10(255/sqrt(MSE3));
PSNR=20*log10(255/sqrt(MSE))
```

Bibliographie

[1] M.ANDRE, 'Introduction aux techniques de traitement d'images.' Eyrolles 1987

[2] R.C.GONZALES P.WINTZ, Digital Image Processing'-Addison Wessley – 1977

[**3**] MICROSOFT, 'Encyclopédie ENCARTA', 1997.

[4] R.M.GRAY, 'Vector Quantization' -IEEE ASSP magazine -, Avril 1984

[5] S.DEKKAR**,** 'Compression des images fixes par la méthode des fractales.', PFE INI 1996.

[6] S.Dekkar, " Compression des images fixes par la méthode des fractales ", PFE INI, 1996.

[7] W. Pratt, " Digital Image Processing ", Willey Interscience Publications, 1991.

[8] N.Gamaz, " Etude et développement d'algorithmes de compression d'images fixes par TCD-2D ", Magister CDTA 1992.

[9] L.BOUAZIZ, " Etude et implémentation des techniques de compression et restitution d'images, application aux données spectrométriques ", PFE INI, 1992.

[10] O.SADOU, " Introduction de l'analyse en composantes principales pour la construction d'un dictionnaire de QV ", PFE USTHB 1993.

[11] M.Abdat, " Etudes des techniques de compression des images fixes et amélioration de la résistance aux erreurs de transmission ", Thèse PhD, Blida-Algerie, Dec.1995

[12] Bellaiche Philipe, "les secrets de l'image video: colorimétrie, éclairage optique, signal vidéo, compression numérique, et formats d'enregistrement", 5^{éme} édition, Paris, Eyrolles, 2005.

[13] M.BAILICHE, "Étude Comparative des Algorithmes d'Estimation de Mouvement dans l'Image Vidéo par la Technique de "Block Matching" ", Magister en électronique 2006

[14] Mei-Juan Chen, Liang-Gee Chen, Tzi-Dar Chiueh, "One dimentional Full Search motion estimation algorithm for video coding", IEEE transactions on circuits and systems for video technology. Vol 4, N° 5 , Octobre 1994.

[15] VSK Reddy and Somnath Sengupta, "A new predictive Full-Search Block Motion Estimation", Proceeding of the 17th international Conference on pattern Recognition 2004.

[16] Yih-Chuan and Shen-Chuan Tai, "Fast Full Search Block Matching Algorithm for Motion Compensated Video Compression", IEEE transactions on communications, Vol 45, N°5, Mai 1997

[17] Xuan Jing and Lap-Pui Chau, "An Efficient Three-Step Search Algorithm For Block Motion Estimation", IEEE transactions on multimedia, Vol 6, No 3, Juin 2004.

[18] Renxiang Li, 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. Vol 4 N° 4 Aout 1994.

[19] Lai-Man Po; Chok-Kwan Cheung; "A new center-biased orthogonal search algorithm for fast block motion estimation". Proceedings IEEE TENCON, digital signal processing applications. Volume 2, Page(s):874 – 877. Nov. 1996.
[20] Kuan-Tsang Wang , Oscal T.-C. Chen, "Motion estimation using an efficient four-step search method", IEEE transaction on circuits and systems for video technology, vol. IV page 217-220
[21] Shan Zhu , Kai-Kuang Ma, "A new diamond search algorithm for fast block matching motion estimation", IEEE transactions on image processing, Vol 9, N° 2, fevrier 2000, pp 287-290.

[22] John Watkinson, " la réduction de débit en audio et vidéo : JPEG, MPAG-1, MPEG-2, redondance ", Paris , Eyrolles, 1998.

[23] A. Dipanda, S. Woo, F. Marzani, J-M. Bilbault - *3-D Shape Reconstruction in an Active Stereo Vision System Using Genetic Algorithms.* Pattern Recognition 36 (2003)

[**24]** M. Zaim, A. El ouaazizi, R. Benslimane - *Genetic Algorithms Based Motion Estimation.* 2001.

[25] Jau-Ling Chen and Pei-Yin Chen, "A new search algorithm for block motion estimation", IEEE transactions on circuits and systems for video technology, 2000.pp 979-982.