REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université Mohamed Khider de Biskra
Faculté des Sciences Exactes et Sciences Naturelle et de la Vie

Nº d'ordre : IVA21/M2/2021

Mémoire

Présenté pour l'obtention du Diplôme de Master en Informatique

Spécialité : Image et Vie Artificielle

Théme

L'amélioration de la méthode de block-matching par l'utilisation de l'algorithme d'optimisation par l'essaim de particulaires

---

Présenté par :
ALLALI ILIES

Proposé et dirigé par :

Pr. Djerou Leila

Année universitaire 2020/2021

Remerciements

Merci ALLAH de nous avoir donné la capacité d'écrire, de réfléchir, la force d'y croire, la patience et la volonté de mener à terme le présent travail. Je tiens à remercier vivement ma promotrice Madame Leila Djerou. Professeur à l'université de Biskra. Pour son encadrement et pour sa contribution à travers sa disponibilité, pour sa patience et surtout pour sa confiance, ses remarques et ses conseils qui m'ont beaucoup éclairés et à travers lesquels j'ai apprécié l'utilité de ce mémoire. Je voudrais également remercier les membres du jury pour avoir accepté d'évaluer ce travail et pour toutes leurs remarques et critiques. Enfin je tiens à exprimer ma profonde gratitude à ma famille et mes amis qui m'ont toujours soutenues. Et à tous ceux qui ont contribué de près et de loin à la réalisation de ce modeste travail.

♡

❖ *ILIES*

Dédicaces

Je dédie ce mode

famille qui ma toujours conseillée et Orientée :

Ma mère : qui e

Mon père : qui e

Me

Mon cher frère ainsi que sa femme et se

Me

se

Me

durant tout le parcours de me

Selma

Je le dédie aussi aux personne

Résumé

Les méthodes de mise en correspondance de blocs ou Block-Matching (BM) sont les méthodes d'EM les plus utilisées grâce à leur implémentation simple et efficace. La première méthode de BM adopte une stratégie de recherche exhaustive (FSA : full search algorithm) dans l'image de référence en évaluant tous les blocs existants dans la fenêtre de recherche. Avec la stratégie de recherche exhaustive, le bloc le plus similaire est trouvé, et par conséquent, la précision d'estimation de mouvement est élevée. Cependant, elle est très lourde en temps de calcul, ce qui rebute certains utilisateurs. Pour remédier à ce problème, nous avons traité le problème de BM comme étant un problème d'optimisation, en adoptant l'algorithme d'optimisation par essaim de particules (PSO : Particle Swarm Optimization) pour le résoudre. Des expérimentations, sur une séquence vidéo, et des comparaisons de résultats avec FSA ont montré la supériorité de l'algorithme proposé BM-PSO par rapport au FSA en termes de précision d'estimation de mouvement et en complexité de calcul.

Mots-Clés : Méthode de mise en correspondance de blocs,Algorithme de recherche exhaustive,Optimisation par essaim de particulaires.

ملخص

تعد طرق مطابقة الكتل أكثر طرق تقدير الحركة استخداما بفضل تنفيذها البسيط والفعال. تعتمد الطريقة الأولى لـمطابقة الكتل على خوارزمية البحث الكامل في الصورة المرجعية من خلال تقييم جميع الكتل الموجودة في نافذة البحث. مع إستراتيجية البحث الشاملة ، يتم العثور على الكتلة الأكثر تشابها ، ونتيجة لذلك ، تكون دقة تقدير الحركة عالية. ومع ذلك ، تستغرق وقتا طويلا للمعالجة ، مما يؤدي إلى تجنب تطبيقها من قبل المستخدمين. لمعالجة هذه المشكلة ، سوف نتعامل مع مشكلة مطابقة الكتل على أنها مشكلة استمثال، وذلك باستخدام خوارزمية استمثال عناصر السرب لحلها. أظهرت التجارب على تسلسل الفيديو ومقارنات النتائج مع خوارزمية البحث الكامل تفوق الخوارزمية المقترحة على خوارزمية البحث الكامل من حيث دقة تقدير الحركة والتعقيد الحسابي.

الكلمات المفتاحية: طريقة مطابقة الكتل، خوارزمية البحث الكامل ، استمثال عناصر السرب .

Abstract

Block Matching (BM) are the most used EM methods thanks to their simple and efficient implementation. BM's first method adopts a full search algorithm (FSA) in the reference image, by evaluating all existing blocks in the search window. With the exhaustive search strategy, the most similar block is found, and as a result, the accuracy of the motion estimate is high. However, it is very heavy in computing time, which puts off some users. To remedy this problem, we have treated the BM problem as an optimization problem by adopting the Particle Swarm Optimization algorithm (PSO) to solve it. Experiments, on a video sequence, and comparisons of results with FSA have shown the superiority of the proposed BM-PSO algorithm over FSA in terms of motion estimation precision and computational complexity.

Key words : (BM : Block-Matching), (FSA : full search algorithm), (PSO : Particle Swarm Optimization).

Table des matières

Table des figures

Annexes

EM : Estimation de Mouvement.

VM : Vecteurs de Mouvement.

BM : block-macthing algorithme.

PSO : Particle Swarm Optimization.

MAD : The Mean Absolute Difference.

MSE : the Mean Squared Error.

FS : Full Search.

TSS : Three Step Search.

FSS ou 4SS : Four Step Search.

GS : Gradient Search.

ED : Etoile Diamant.

TDL : Two Dimension Logarithm.

Introduction générale

Les séquences vidéo sont des sources d'informations riches et très importantes et représentes les types de données les plus vues, partagées et consommées dans le monde. Les informations liées au mouvement jouent un rôle essentiel pour des applications diverses comme la compression vidéo, la robotique, la surveillance, la météorologie, les applications médicales, militaires...
L'estimation de mouvement ou (ME : Motion estimation est un procédé qui 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.

Nombreuses techniques d'EM ont été proposées, dans la littérature qui peuvent regroupées en deux grandes catégories. Les méthodes de flot optique (à base de pixel) qui calculent un vecteur de mouvement pour chaque pixel dans l'image[38]-[39], et les méthodes de mise en correspondance de blocs (BM : block-matching) qui divisent l'image en blocs de pixels et calculent ensuite un vecteur de mouvement pour chaque bloc [40], [41]-[43].

Les méthodes de flot optique procurent un flot optique dense, en calculant un vecteur de mouvement pour chaque pixel. Cependant, ces méthodes demandent un temps de calcul excessif, et elles donnent des résultats satisfaisants seulement si le mouvement dans la séquence d'images est faible [38]. Les méthodes de mise en correspondance de blocs ou block-matching (BM) sont les méthodes d'EM les plus utilisées grâce à leur implémentation simple et efficace.

Le principe général de ces méthodes est que chaque deux images successives dans la séquence vidéo (courante et référence) sont divisées en blocs. Ensuite, pour chaque bloc dans l'image courante, l'algorithme cherche le bloc le plus similaire dans l'image de référence en minimisant une erreur de dissimilarité entre les illuminations correspondantes aux pixels respectifs des deux blocs. Le vecteur de mouvement est la différence entre les positions du bloc courant et du bloc le plus similaire dans l'image de référence.
La première méthode de BM adopte une stratégie de recherche exhaustive (FSA : full search algorithm) dans l'image de référence en évaluant tous les blocs existants dans la fenêtre de recherche .Avec la stratégie de recherche exhaustive, le bloc le plus similaire est trouvé, et par conséquent, la précision d'estimation de mouvement est élevée. Cependant, elle est très lourde en temps de calcul. Afin de réduire cette complexité plusieurs méthodes ont été proposées. Parmi elles, nous citons : Three Step Search, Diamond Search, Hexagon Search, Octogon, etc.

Malgré que ces méthodes réduisent la complexité de calcul, elles peuvent tomber sur des solutions sous-optimales (minima locaux).
Le problème de BM peut être reformulé comme étant un problème d'optimisation. L'utilisation des méta-heuristiques, pour le résoudre, est un moyen permettant de réduire la complexité de méthode FSA et éviter le problème des optimaux de minima locaux.
Les métaheuristiques sont des techniques d'optimisation approchées, permettant l'obtention de résultats d'optimisation satisfaisants dans un temps raisonnable. Elles se basent sur la minimisation (ou maximisation) d'une fonction objectif, en utilisant des mécanismes aléatoires avec des outils d'intensification et diversification pour explorer l'espace de recherche, éviter les minimaux locaux et déterminer ou d'approcher un optimum global.

Dans ce projet, nous avons adopté la méta-heuristique, l'optimisation d'essaim de particules (PSO : Particle Swarm Optimization) qui est une méthode simple à implémenter et son efficacité, de résolution de problème d'optimisation, a été démontré à travers plus domaines d'application.

Pour assurer les objectifs de notre travail, nous organisons ce mémoire en quatre chapitres :

Chapitre 1 : donne le principe général du problème de l'estimation de mouvement, ainsi que les difficultés soulevées lors de la résolution de ce problème. Nous présentons ensuite les trois grandes catégories des techniques d'estimation de mouvement.

Chapitre 2 : présent le principe général de block-matching et la fonction de coût utilisée, dans cette méthode, ainsi que les différentes méthodes du Block Matching nous insistons de façon générale sur l'algorithme de recherche exhaustive (FSA).

Chapitre 3 : décrit le principe de la méthode PSO en présentant son fonctionnement.

Chapitre 4 : présente notre adaptation de l'algorithme PSO pour améliorer l'amélioration de la méthode de BM, il comporte la conception et l'implémentation de notre système BM-PSO ainsi que les expérimentations et les résultats obtenus. Nous terminerons ce mémoire par une conclusion générale et quelques perspectives.

Chapitre 1

L'estimation de mouvement

1.1 Introduction

l'estimation du mouvement des objets contenus dans une scène est une primitive importante dans l'analyse d'une séquence vidéo, Cette phase d'estimation est dépendante de la modélisation sélectionnée pour rendre compte de l'évolution spatiale des objets. L'obtention de cette primitive s'appuie sur l'observation de plusieurs trames consécutives, Cette observation contient des objets ayant changé de position relative. L'évolution temporelle engendrée est appelée de façon générique : mouvement.

Ce chapitre est consacré à la description de différentes méthodes d'estimation du mouvement. Dans la suite de ce chapitre nous définirons le mouvement et différents types de mouvement. Ensuite, nous présenterons un bref état de l'art de l'estimation de mouvement suivi des techniques d'estimation de mouvement utilisées.

Nous expliquerons toutes les méthodes trouvées. Nous ne donnerons qu'un bref aperçu des méthodes les plus utilisées.

1.2 Le mouvement dans la séquence vidéo

1.2.1 Séquence vidéo

La séquence vidéo est une suite d'images fixes qui défilent à une certaine cadence. Cette cadence est de l'ordre de 25 images par seconde que l'œil humain peut la considérer comme une image animée (voir figure1.1). Généralement une image donnée de la séquence vidéo et les images qui l'entourent (précédent ou succédant) présentent une forte similarité, elles sont composées d'un même ensemble d'objets qui changent de position en chaque image [1].

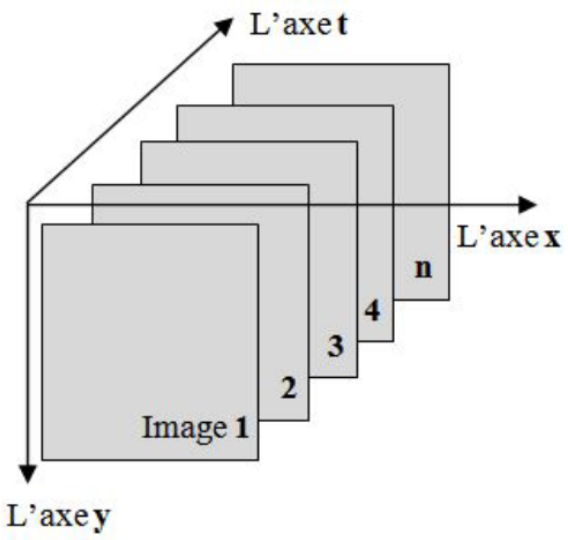


FIGURE 1.1 – La séquence vidéo [8].

1.2.2 Le mouvement

Le mouvement dans une séquence d'image bidimensionnelles ($2D$) est perceptible grâce aux changements de la distribution spatiale, des intensités lumineuses. Le mouvement ainsi perçu est appelé (apparent), car il ne correspond pas nécessairement à la projection dans le plan de l'image du mouvement ayant lieu dans l'espace tridimensionnel (3D)(voir figure 1.2) [9].

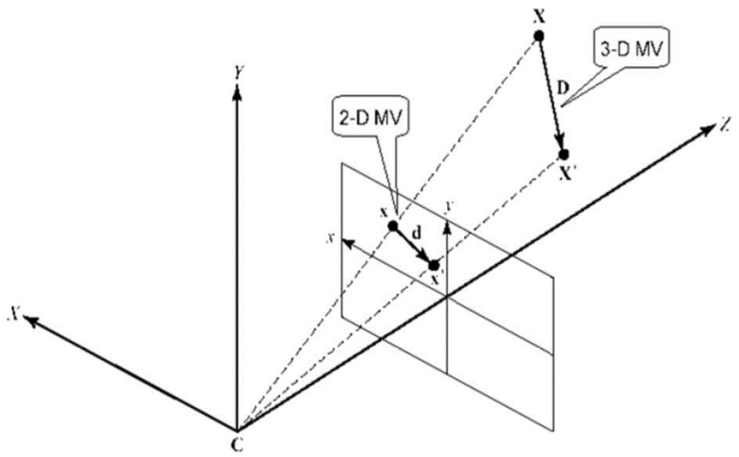


FIGURE 1.2 – Illustration de projection mouvement $3D$ en $2D$

1.3 Classification du mouvement

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

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

- Le mouvement réel.
- Le mouvement apparent ou le mouvement observé.
- Le mouvement estimé.

1.3.1 Le mouvement réel

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

Le mouvement d'objet rigide

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

Le mouvement d'objet déformables

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

1.3.2 Le mouvement apparent

Le mouvement apparent est une combinaison de deux sources de mouvement : le mouvement propre 3D des objets dans la scène et celui de la caméra. Le mouvement de la caméra correspond au mouvement apparent "global". Alors que le mouvement apparent des objets correspond au mouvement apparent " local" relatif au mouvement global [2].

Le mouvement apparent est le mouvement qui est observé sur l'image. Il fournit une représentation du mouvement observé sous la forme d'une séquence temporelle d'image.

Le champ de mouvement apparent représente la projection du mouvement $3D$ dans le plan image. C'est pourquoi, le champ de mouvement apparent s'appelle aussi mouvement projeté Il représente une approximation du mouvement réel(voir figure 1.3).

Bien que ce mouvement soit toujours observé, il n'est pas toujours dû à une réalité dans le monde réel. En effet, l'estimation d'un mouvement qui n'existe pas réellement peut avoir lieu.

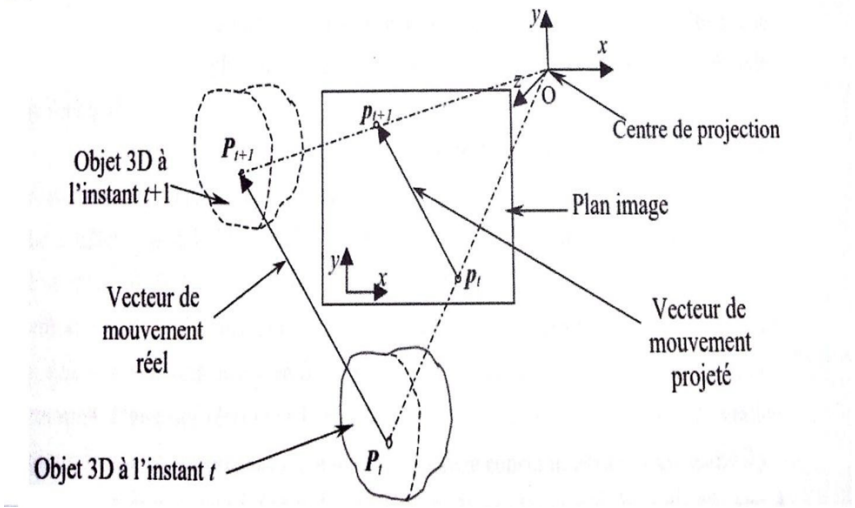


FIGURE 1.3 – Illustration de mouvement réel et apparent.

1.3.3 Le mouvement estimé

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

1.4 L'estimation du mouvement

L'estimation du mouvement est l'un des domaines du traitement d'image qui joue le rôle dans l'analyse des séquences d'images. Aussi c'est un procédé qui consiste à étudier le déplacement des objets dans la séquence vidéo.

Le principe de l'estimation du mouvement consiste à construire une image de prédiction à partir d'une image précédente. L'estimation du mouvement n'est pas effectuée sur chaque point de l'image mais sur des macros blocs, afin de réduire au minimum la richesse d'information à transmettre [3][4] .

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

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

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

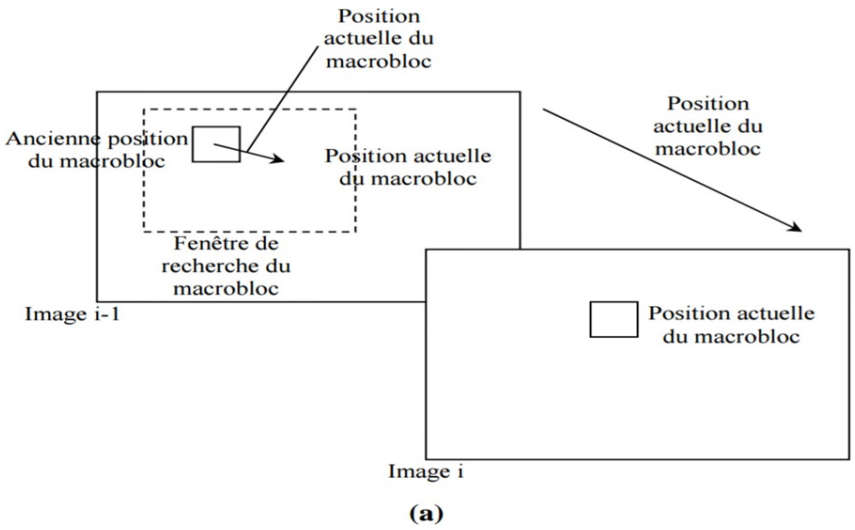


FIGURE 1.4 – Utilisation de la technologie d'estimation de mouvement

1.4.1 Estimation des vecteurs de déplacement

L'estimation peut être vue comme un problème d'estimation de mouvement direct ou inverse, selon que l'estimation est réalisée entre l'instant (t) et $(t+1)$ ou entre l'instant (t) et $(t-1)$.

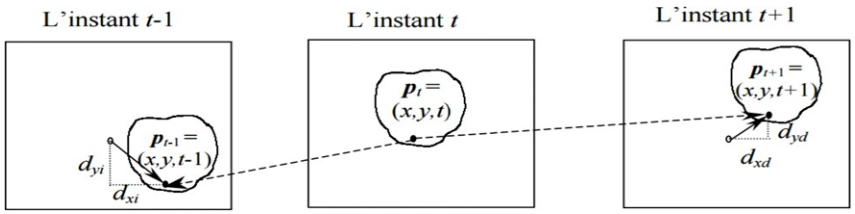


FIGURE 1.5 – Estimation directe et inverse des vecteurs déplacements.

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

1.4.2 Estimation des vecteurs vitesse

Le champ de mouvement estimé peut être caractérisé par le champ de vecteur vitesse ou par le champ de vecteurs déplacement. Le champ de vitesse estimé et le champ de déplacement sont identiques quand l'échantillonnage temporel de la même séquence est constant. Pour un mouvement accéléré, on doit prendre en compte plus de deux images afin d'estimer correctement le champ de vitesse. Etant donné les échantillons $It(x, y)$, on doit déterminer la vecteurs vitesse $V(x, y, t)$.
On observe que si la vitesse reste constante dans l'intervalle Δt entre deux images. Si Δt est petit, alors la vitesse estimée peut être assimilée au déplacement.

$$V\boldsymbol{t}(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{d}\boldsymbol{t}(\boldsymbol{x}, \boldsymbol{y})/\Delta\mathbf{t}$$

1.5 Les problème de l'estimation de mouvement

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

1.5.1 problème d'occlusion

Le problème de l'occlusion est un problème non résolu dans le traitement de séquence vidéo. Le phénomène d'occlusion dans une séquence vidéo est dû à l'apparition et à la disparition, ou aux croisements d'objets, dans la vidéo. C'est le résultat du mouvement de la caméra et des objets dans la scène.

Les occlusions rendent difficiles l'estimation de mouvement apparent dans la séquence vidéo. En effet, les techniques d'estimation de mouvement entre deux images présentes à deux instants (t) et $(t+1)$ d'une séquence vidéo sont basées sur les ressemblances existantes entre ces deux images.

Si la zone apparaît ou disparaît entre les deux images. La ressemblance diminue et l'estimation échoue dans cette zone.

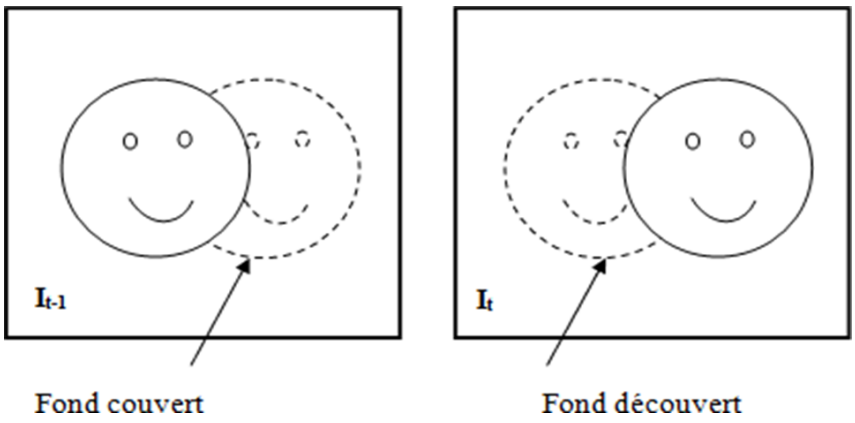


FIGURE 1.6 – Illustration du problème d'occlusion.

La région représentée en pointillés dans l'imageI_t représente la région du fond qui a été découverte par l'objet en mouvement dans I_{t-1}. Par conséquent, pour les pixels de cette région n'auront aucun correspondant dans l'image I_{t-1}. Les algorithmes d'estimation de mouvement fournissent malgré tout une estimation de mouvement sur cette zone, mais dénuée de tout sens physique.

1.5.2 Problème d'ouverture

Le problème d'ouverture représente une formulation du fait que la solution du problème d'estimation de mouvement n'est pas unique. Si on suppose les vecteurs de mouvement en chacun des pixels comme des variables indépendantes, le nombre d'inconnues sera deux fois plus grand que le nombre d'équation disponible.
Le problème d'ouverture est illustré sur la figure (1.7). Dans cet exemple, le coin d'un objet se déplace dans la direction de l'axe des y. Si nous estimons le mouvement à partir d'une fenêtre locale, il est alors impossible de déterminer si l'objet se déplace vers le haut ou perpendiculairement à son contour.

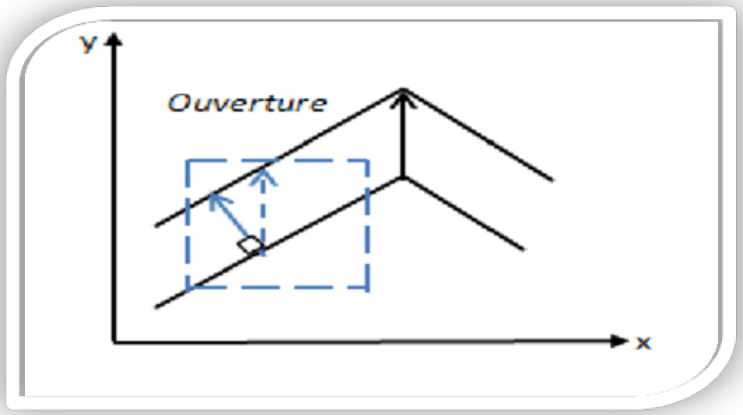


FIGURE 1.7 – Illustration du problème d'occlusion.

1.5.3 problème du modèle de translation

Le modèle de mouvement le plus utilisé dans les méthodes d'estimation du mouvement est le modèle de mouvement constant ou modèle de translation, qui suppose que tous les pixels du bloc effectuent le même déplacement (cas d'objets rigides).
Les avantages de l'estimation de mouvement utilisant un modèle de translation du mouvement :

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

Par contre, un des inconvénients majeurs de l'estimation de mouvement utilisant un modèle de translation du mouvement se traduit par les artefacts de type bloc, dus à un sous- échantillonnage du champ de mouvement.

1.6 Les méthodes d'estimation de mouvement

L'estimation du mouvement dans les séquences vidéo est un vaste domaine de recherche qui a suscité de nombreuses études. Il existe différentes techniques d'estimation de mouvement et plusieurs façons d'exprimer les résultats :

1.6.1 Méthodes de mise en correspondance de blocs

Les méthodes de mise en correspondance connues sous le nom Block-Matching, sont basées sur l'hypothèse selon laquelle l'intensité lumineuse des pixels est constante ou faiblement variée le long de la trajectoire du mouvement.Ces méthodes comparent entre deux blocs de deux images successives afin de déterminer la position du bloc le plus proche en utilisant un critère de similarité, ces techniques sont simples et relativement moins sensibles aux bruits ce qui fait que ces méthodes sont plus utilisées dans la pratique que ce soit dans les standards de compression ou l'estimation de mouvement.
Elles sont parmi les plus utilisées en pratique grâce à la simplicité de l'implantation physique du modèle de translation utilisé le plus souvent dans ces méthodes.

1.6.2 Méthodes différentielles

Les méthodes différentielles calculent la vitesse à partir de dérivées spatiales et temporelles d'intensité lumineuse des pixels. Sont des méthodes basées sur une hypothèse forte de conservation de l'intensité des pixels au cours du mouvement dans la séquence d'images.
Parmi les méthodes différentielles existant, on peut citer les méthodes de Horn et Schunck [5], de Lucas et Kanade, de Nagel [6], d'Uras et al, le principe de ces méthodes est le même quelle que soit la technique employée, ils sont basées toujours sur l'équation générale du flux optique pour estimer les mouvements dans les images.

1.6.3 Méthodes fréquentielles

Les méthodes fréquentielles d'estimation de vitesse dans les séquences d'images sont fondées sur une caractérisation du mouvement dans le domaine des fréquences. Elles ont pour origine des recherches concernant la vision des mammifères, et la mise en évidence de la présence de cellules simples dans l'aire corticale, qui se comportent comme des filtres passe- bande spatio-temporels. Ces méthodes fournissent des flux optiques de qualité, mais sont réputées gourmandes en calculs.

Dans les approches fréquentielles, les fréquences spatio-temporelles sont mises en relation avec la vitesse du stimulus de mouvement, et le flot optique devient l'identification d'un plan d'énergie dans l'espace de la fréquence spatio-temporelle. En général, les mécanismes sensibles au mouvement fondés sur l'énergie orientée spatio-temporelle dans l'espace de Fourier peuvent estimer le mouvement dans les endroits où les autres approches échouent. Ainsi, la détection du mouvement dans l'image revient ici à extraire l'orientation spatio-temporelle [7]. On distingue deux sous-approches, Les méthodes fréquentielles basées sur l'énergie et Les méthodes exploitant la phase du signal.

1.7 Conclusion

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

Dans le chapitre suivant, nous allons détailler les méthodes de Block-Matching qui sont des méthodes d'estimation du mouvement les plus utilisées dans la pratique.

Chapitre 2

Block-Matching

2.1 Introduction

L'estimation du mouvement est l'un des domaines du traitement d'image. Elle consiste principalement dans l'analyse des séquences d'images et à exploiter les redondances temporelles qui existent entre les images consécutives de la séquence, afin de représenter des vecteurs caractérisant les mouvements existant dans cette séquence.
Les méthodes du Block-Matching sont parmi les méthodes d'estimation du mouvement les plus utilisées dans la pratique. Elles peuvent être utilisées pour découvrir des redondances temporelles dans une séquence vidéo tel qu'un objet sans (avec) mouvement dans une vidéo, ce qui accroît l'efficacité de compression d'une image. On les retrouve presque dans tous les standards de compression vidéo (H.261, MPEG-1, MPEG-2 ou MPEG-4). L'estimation du mouvement n'est pas utilisée uniquement pour la compression vidéo, mais elle présente aussi un élément principal de plusieurs applications vidéo tels que l'analyse des séquences, le suivi des objets en mouvement (Applications militaires), ... etc.
Dans ce chapitre, nous nous s'intéresserons aux différentes méthodes d'algorithme de block-matching et les plus utilisées.

2.2 Principe de block-matching

Dans une séquence d'images, l'image actuelle est prédite à partir d'une image précédente appelée image de référence. La trame actuelle est divisée en macroblocs sans chevauchement, généralement de 16 pixels x 16 pixels. Ce choix de taille est un bon compromis entre précision et coût de calcul. Cependant, les techniques d'estimation de mouvement peuvent choisir différentes tailles de bloc et peuvent faire varier la taille des blocs dans une trame donnée [10].

Chaque macrobloc dans la trame courante est comparé à un macrobloc dans la trame de référence [11], en utilisant une certaine fonction de coût, et le meilleur macrobloc correspondant est sélectionné. La recherche est effectuée dans une fenêtre de recherche prédéterminée définie par le paramètre de recherche p.

En général, p est réglé sur ± 7 pixels. Déterminer un vecteur indiquant le déplacement du macrobloc dans la trame de référence par rapport au macrobloc dans la trame courante. Ce vecteur est appelé vecteur de mouvement.

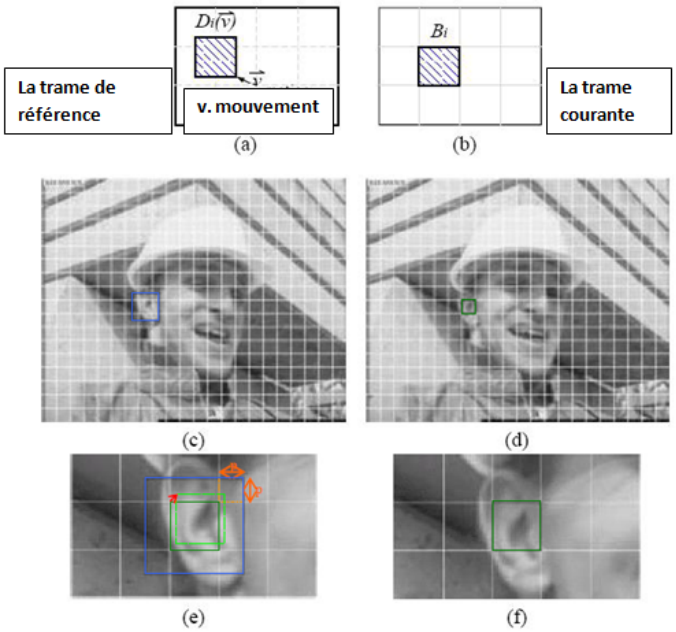


FIGURE 2.1 – Principe de la méthode block matching.

Les algorithmes d'estimation de mouvement par correspondance de blocs trouvent le vecteur de mouvement du bloc actuel à travers en trouvant le bloc déplacé et le mieux adapté dans le cadre de référence. Par exemple, (c) et (d) montrent la 137e image et la 138e image. La trame courante (138) est divisée en blocs ne se chevauchent pas, comme illustré en (d). Le vecteur de mouvement pour le bloc actuel de l'image actuelle - indiqué en (f) en vert - est trouvé en localisant le bloc déplacé le mieux adapté dans la fenêtre de recherche correspondante dans la plage $[-p, p]$ - indiquée en (e) en bleu - dans l'image précédente (137). Le vecteur de déplacement qui produit l'erreur d'appariement minimale via une fonction de coût est le vecteur de mouvement - représenté ci-dessus par une flèche rouge.

2.3 Fonction de coût

La mise en correspondance d'un macrobloc à l'autre est basé sur la sortie d'une fonction de coût également appelé une mesure de distorsion de bloc (BDM : The Mean Absolute Difference).
Macrobloc que les résultats du moindre coût est celui qui correspond au bloc courant le plus proche. Il existe différentes fonctions de coût [12], dont la plus populaire et la moins coûteuse en calcul est :

La différence absolue moyenne (MAD) donnée par l'équation (2.1).

$$\mathbf{MAD} = \frac{1}{\mathrm{N}^2} \sum_{\mathrm{i}=0}^{\mathrm{N}-1} \sum_{\mathrm{j}=0}^{\mathrm{N}-1} |\mathrm{Cij} - \mathrm{Rij}| \tag{2.1}$$

Une autre fonction de coût est l'erreur quadratique moyenne (MSE : the Mean Squared Error) donnée par l'équation (2.2).

$$MSE = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (C_{ij} - R_{ij})^2 \tag{2.2}$$

où N est la taille du macrobock, C_ij et R_ij sont les pixels qui sont comparés dans le macrobloc actuel et le macrobloc de référence, respectivement.

Ghanbari (1990) déclare que l'utilisation de l'erreur absolue moyenne plutôt que de l'erreur quadratique moyenne plus complexe comme mesure de distorsion, se traduit par une performance meilleure (presque $0,8\%$ d'erreur de prédiction en moins).

Pour chaque fonction de coût, une comparaison est faite pixel par pixel en utilisant uniquement la valeur de luminance. Ces erreurs sont additionnées sur le macrobloc et si cette erreur est inférieure à l'erreur précédente, l'emplacement du macrobloc dans l'image de référence est enregistré. Une fois que tous les macroblocs de la fenêtre de recherche ont été examinés, le vecteur de mouvement est déterminé sur la base du macrobloc avec la mesure d'erreur la plus faible.

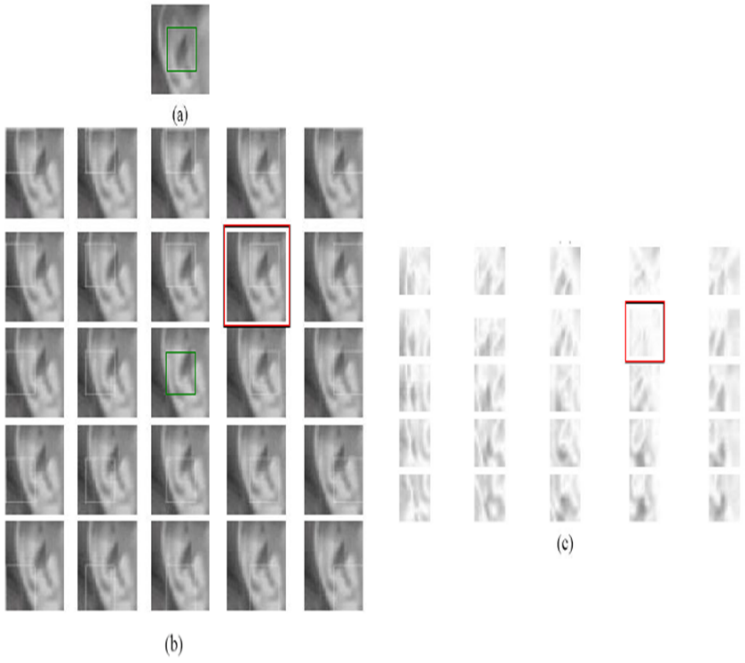


FIGURE 2.2 – Une fonction de coût appliquée à un certain nombre d'emplacements dans la fenêtre de recherche.

a) montre le bloc courant de la trame courante en vert.

b) montre différents blocs déplacés dans la trame de référence dont l'emplacement correspondant du bloc courant en vert.

c) montre les résidus correspondants (erreurs de La mise en correspondance).

Le déplacement (en haut à droite) qui trouve le bloc qui correspond le mieux (marqué en rouge) est le vecteur de mouvement.

2.4 Prédiction avant (forward) et arrière (backward)

L'algorithme du Bloc Matching permet d'estimer le mouvement des blocs entre deux images à des temps consécutifs. Le mouvement ainsi calculé va permettre de prédire l'image $t + 1$ en combinant l'information contenue dans les blocs dans l'image t et les vecteurs mouvements.

Ainsi. Deux types de prédictions sont possibles : la prédiction « avant » dite « forward » et la prédiction « arrière »dite « backward ». La prédiction forward consiste à diviser

l'image n en blocs et à trouver leur position dans l'image $t+1$. En revanche, la prédiction backward, divise l'image $t+1$en blocs et cherche leur position dans l'image t.

FIGURE 2.3 – Prédiction avant (forward)

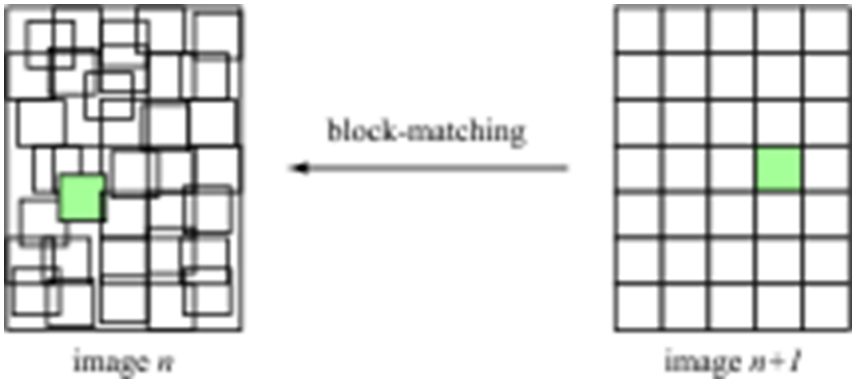


FIGURE 2.4 – Prédiction arrière (backward)

FIGURE 2.5 – Prédiction avant et arrière [13].

Les résultats sont très différents. Il apparaît par construction que l'image prédite par prédiction forward présente des « trous ». En effet, de manière générale les blocs n'ont pas un mouvement unique pour toute l'image. Lors de la prédiction certains blocs se recouvrent et laissent apparaître de zones n'ont prédites et par conséquent noires (voir figure 2.3) .

Ce problème est entièrement résolu par l'utilisation de la prédiction backward qui ne laisse entrevoir aucun « trou » dans la prédiction (voir figure 2.4) .

2.5 Critères d'évaluation

Les blocs sont assignés sur la base d'un critère d'évaluation, ce critère n'est autre qu'une mesure de ressemblance (similarité) ou de dissemblance (disparité) entre les blocs testés. Dans le cas d'un critère de ressemblance, ce dernier doit être maximisé, sinon, minimisé dans le cas d'une dissemblance. De manière générale, la plupart des articles traitant de l'estimation du mouvement, avec les algorithmes du Block Matching, considèrent que les blocs sont caractérisés par la luminance (intensité lumineuse). Ceci est dû au fait que l'œil humain est davantage sensible à l'intensité lumineuse qu'à la couleur [14].
Pour définir le meilleur bloc cible, l'algorithme se base donc sur des critères d'évaluation métrique qui mesurent la différence de contenu entre les deux blocs et les compare avec tous les autres blocs cibles potentiels. Voici des exemples de critères :

- Sum of Squared Differences (SSD) : somme des différences des pixels de chaque bloc au carré, non adapté à la couleur (critère à rajouter).
- Sum of Absolute differences (SAD) : Somme des différences absolues calcule la somme des valeurs absolues des différences de luminance des pixels de l'image originale et celle de référence.
- Zero-mean Normalized Sum of Absolute Differences (ZNSAD) : prend en compte les moyennes des blocs et normalise le résultat (ZNSSD pour SAD)
- Erreur absolue moyenne EAM : traite les erreurs de manière uniforme.
- Coefficient d'inter-corrélation (CC) : le critère de ressemblance statistique le plus utilisé.
- Coefficient d'inter-corrélation normalisé et centré (NCC) : c'est le CC normé et centré à la variance.
- Erreur quadratique moyenne (EQM) : Ce critère est dérivé de l'erreur moyenne absolue.

Les expressions des critères sont données pour une fenêtre de recherche de taille $L \times L$, centrée sur le pixel de coordonnées $p(px, py)$ (position initiale) dans l'image de référence. Le déplacement d'un bloc candidat à l'intérieur de la fenêtre est caractérisé par un vecteur $V = (dx, dy)$et est borné par la taille de la zone de recherche. La position du meilleur bloc candidat est notée $(d\hat{x}, d\hat{y})$.

2.6 Stratégies de recherche du meilleur bloc

Il existe plusieurs méthodes du Block Matching pour le parcours du voisinage, afin de trouver le bloc qui correspond au mieux au bloc de référence, et qui optimise le critère d'évaluation. La méthode la plus fondamentale pour trouver le meilleur bloc candidat est de visiter et de tester tous les blocs de la zone de recherche [15].
Cette stratégie, appelée recherche exhaustive, est optimale mais pourrait s'avérer très coûteuse en temps de calcul, surtout pour des amplitudes de déplacement grandes. Cela implique des zones de recherche de taille importante et donc un nombre considérable de blocs candidat à visiter. D'autres balayages non exhaustifs de la zone de recherche ont été proposés dans la littérature. L'objectif de ces techniques est de converger plus rapidement vers le minimum de la fonction de similarité.

2.6.1 Algorithme de recherche exhaustive (Full Search (FS))

Cette méthode consiste à faire la recherche de bloc sur l'ensemble des blocs de l'image référence [16]. Cet algorithme est le plus simple, mais aussi le plus coûteux 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.

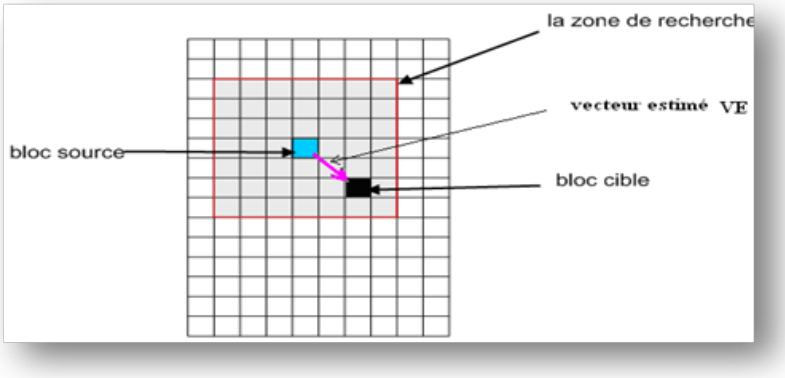


FIGURE 2.6 – Algorithme de recherche exhaustive (FS).

2.6.2 Recherche en trois pas (Three Step Search (TSS))

L'algorithme Three Step Search a été proposé en 1981 par Koga et al [17]. Contrairement à l'algorithme FS, cette méthode (TSS) se limite à trois pas pour trouver le meilleur bloc. Au lieu de parcourir tous les blocs de la zone de recherche, le but de ces algorithmes est de parcourir la fenêtre en se rapprochant pas à pas du bloc candidat.

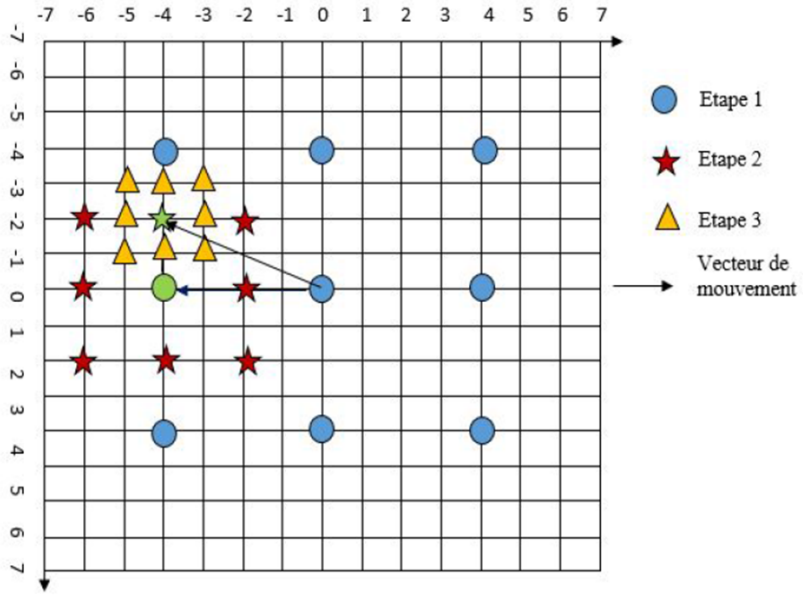


FIGURE 2.7 – Algorithme TSS.

La Figure 2.7 : illustre les trois étapes nécessaires à l'algorithme pour retourner un résultat. Dans la première étape le bloc centré se place au vecteur de mouvement nul. Ensuite tester le critère aux alentours du centre (8-connexité) avec un premier pas, et on repère le point qui vérifie le critère. On prend le point trouvé précédemment comme nouveau centre et on recherche aux alentours de ce point avec un pas inférieur au premier pas. On reprend le minimum comme centre de la prochaine étape. Dans cette dernière étape, on répète une troisième fois l'opération avec un pas inférieur aux deux premiers. (Recherche en multi résolution).

L'algorithme peut être décrit comme suit : [18]

Début

Étape 1 **:** Un premier pas est choisi. Huit blocs à une distance de la taille du pas du centre (autour du bloc central) sont sélectionnés pour comparaison.

Étape 2 **:** La taille du pas est réduite de moitié. Le centre est déplacé au point avec la distorsion minimale. Fin.
Les étapes 1 et 2 sont répétées jusqu'à ce que la taille de pas soit inférieure à 1.

Fin.

2.6.3 Recherche en quatre pas (four step search(FSS))

Cet algorithme (Four Step Search) a été proposé en 1996 par Lai-Man Po et Wing-Chung Ma [19]. L'algorithme commence par une comparaison de neuf points ensuite les autres points de comparaison sont sélectionnés en fonction de l'algorithme suivant :

Début :

Étape 1 **:** commence par un pas de 2. Choisissez neuf points autour du centre de la fenêtre de recherche. Calculez la distorsion et trouvez le point avec la plus petite distorsion. Si ce point est le centre de la zone de recherche, passez à l'étape 4, si non passez à l'étape 2.

Étape 2 : Déplacez le centre vers le point avec la plus petite distorsion. La taille de pas est maintenue à 2. Le motif de recherche, cependant, dépend de la position de la distorsion minimale précédente

A) Si le point minimum précédent est situé au coin de la zone de recherche précédente, cinq points sont sélectionnés (comme indiqué sur la figure).

B) Si le point de distorsion minimum précédent est situé au milieu de l'axe horizontal ou vertical de la fenêtre de recherche précédente, trois points de contrôle supplémentaires sont sélectionnés. (Comme le montre la figure). Localisez le point avec la distorsion minimale. Si c'est au centre, passez à l'étape 4 sinon passez à l'étape 3.

Étape 3 : La stratégie de recherche est la même, mais elle passera finalement à l'étape 4.

Étape 4 **:** La taille du pas est réduite à 1 et les neuf points autour du centre de la recherche sont examinés.

Fin.

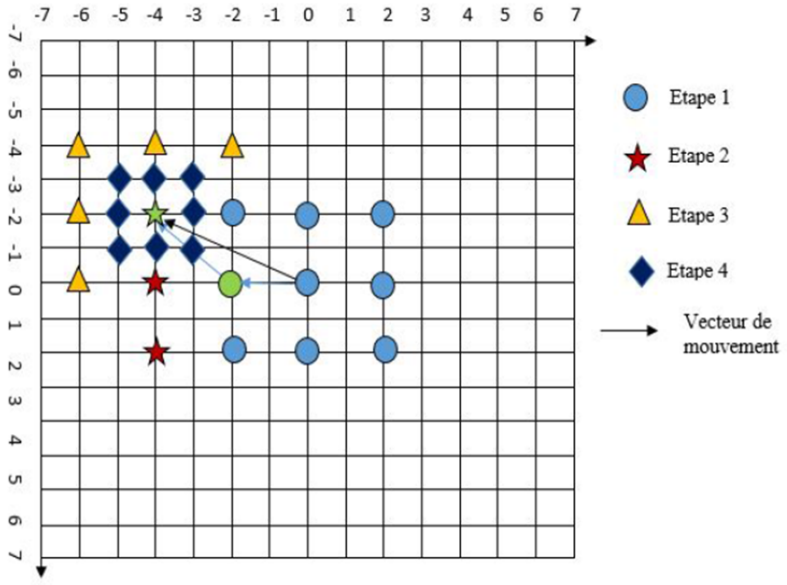


FIGURE 2.8 – Algorithme 4SS.

La complexité computationnelle de la recherche en quatre étapes est inférieure à celle de la recherche en trois étapes, alors que la performance en termes de qualité est aussi bonne. Il est également plus robuste que la recherche en trois étapes et il maintient sa performance pour les séquences d'images avec des mouvements complexes comme le

zoom de caméra et le mouvement rapide. Il s'agit donc d'une stratégie très attractive pour l'estimation du mouvement.

2.6.4 Recherche selon le gradient (Gradient Search GS)

Cet algorithme est proposée par Liu et Feig, en , 1996. Il est basé sur l'utilisation d'un pas de recherche fixe et le passage d'un bloc candidat à l'autre se fait en suivant la direction indiquée par la descente du gradient de la fonction de coût utilisée. C'est un algorithme qui fonctionne efficacement dans le cas des séquences stationnaires.

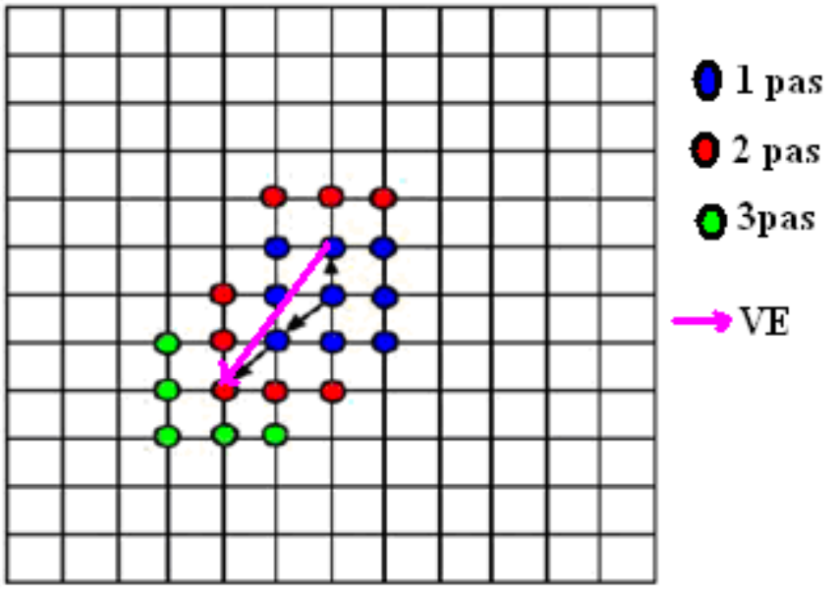


FIGURE 2.9 – Recherche selon le gradient.

2.6.5 Algorithme de recherche en diamant (diamond search DS)

L'algorithme de recherche en diamant est proposée par A.M. Tourapis et al, en 2000 [20]. C'est un algorithme rapide d'estimation de mouvement de correspondance de bloc, il est base sur l'étude de la distribution des vecteurs de mouvement de plusieurs séquence vidéo. L'algorithme DS utilise deux modèles de recherche. Un modèle de recherche en grand diamant (Large diamond search pattern) de 09 points de recherche comme montré sur la figure (a), et l'autre modèle de recherche en petit diamant (Small diamond search pattern) de 05 points comme montré sur la figure (b).

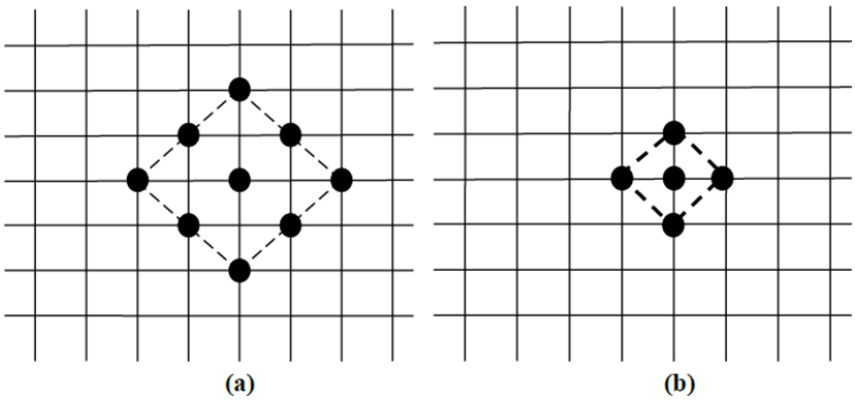


FIGURE 2.10 – (a) Modèle de recherche En grand diamant LDSP.
(b) Modèle de Petit diamant SDSP

Le modèle de recherche en grand diamant est toujours employé dans la procédure de recherche jusqu'à l'étape dans laquelle le point minimum de la déformation de bloc se produit au centre du diamant. Le modèle de recherche est alors passé à la recherche en petit diamant. Le point rapportant le bloc, parmi les 05 points de contrôle dans le SDSP fournit le vecteur de mouvement du bloc le mieux correspondant.
Le principe de l'algorithme DS est le suivant :

- 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 du bloc calculé est situé à la position centrale, passer à l'étape 3 ; autrement, passer à l'étape 2.
- Le point minimum du bloc trouvé dans l'étape de recherche précédente devient le point central du nouveau LDSP. Si le nouveau point minimum du bloc obtenu est situé à la position centrale, passer à l'étape 3 ; autrement, répéter périodiquement cette étape.
- Changer le modèle de recherche de LDSP à SDSP. Le point minimum du bloc trouvé dans cette étape est la solution finale du vecteur de mouvement, donc c'est le bloc le plus correspondant.

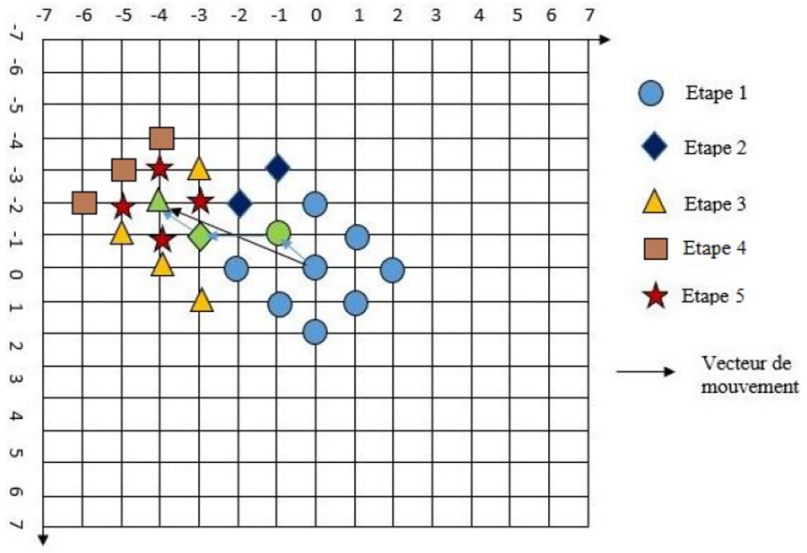


FIGURE 2.11 – Algorithme de recherche en diamant.

2.6.6 Algorithmes $2D$-logarithmiques (TDL : Two Dimension Logarithm)

Les méthodes 2D-logarithmiques sont présentées par Jain et al en 1981. Dans ce type de méthodes, une fois fixé la taille prototype on procède par itération et on réduit la recherche à certains points seulement.

On part du vecteur de mouvement nul, et on cherche le bloc en cinq points : le centre et dans les quatre directions Nord Sud Est Ouest à une distance D. Le point où on trouve le critère vérifié est gardé comme centre de la prochaine recherche. On réduit la distance de recherche D à log2 (D) dans le cas où le point est trouvé au centre du prototype ou bien le centre du prototype touche le bord de l'image.

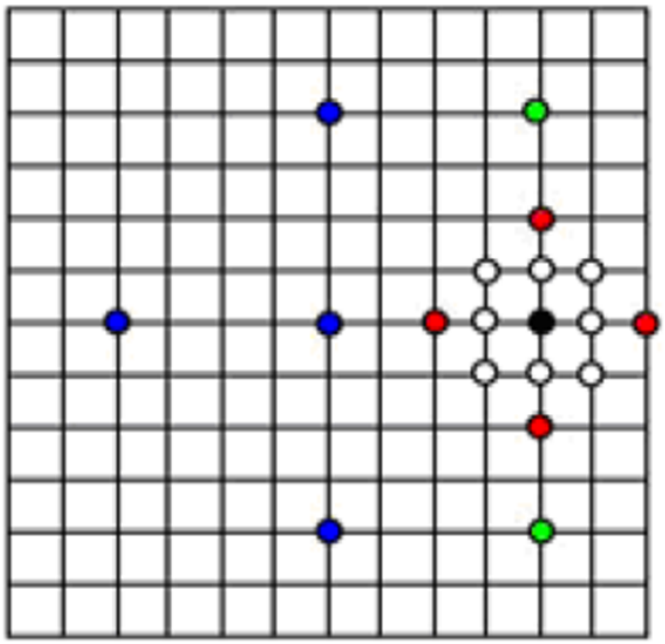


FIGURE 2.12 – Algorithme $2D$-logarithmique.

2.7 Conclusion

Dans ce chapitre, nous avons expliqué les principales techniques de la mise en correspondance de blocs (block-matching) ainsi que le principe général de leur fonctionnement. Dans la suite de notre travail nous nous intéressons à l'amélioration de cette méthode avec l'utilisation de l'algorithme d'optimisation par l'essaim de particulaires.

Chapitre 3

Optimisation par essaim de Particules (PSO)

3.1 Introduction

Trouver un algorithme qui résout un problème, c'est trouver une série d'opérations qui permettent d'apporter une solution au problème. Ce n'est pas toujours facile, mais ce qui peut être plus difficile et plus intéressant en particulier est de trouver rapidement un algorithme qui fournit cette solution. Cela signifie que le temps d'exécution joue un rôle important dans les algorithmes. Comme nous l'avons dit dans le chapitre précédent, la plupart des techniques de mise en correspondance du bloc prennent du temps.
Dans ce mémoire nous avons opté d'utiliser l'algorithme d'optimisation par essaim de particules (PSO) pour tenter de résoudre cette problématique.
Le présent chapitre va être consacré à la présentation de l'algorithme PSO et son fonctionnement .

3.2 Notion fondamentale

D'après Louis Gacôgne [21], les méthodes de résolution de problèmes peuvent être subdivisées en quatre grandes classes. Tous les problèmes rencontrés ne sont pas forcément en rapport avec l'optimisation mais s'y rapportent pour la plupart, en ce sens qu'il est généralement possible de trouver une fonction à minimiser ou à maximiser. Fréquemment, la difficulté première liée à ce genre de problème est la formalisation (définition de la fonction, puis définition d'un codage de solutions).
Il est admis que les quatre procédures sont : les méthodes combinatoires, les méthodes constructives, les méthodes locales, enfin les méthodes évolutionnaires.

Les méthodes évolutionnaires sont des méthodes stochastiques et globales faisant intervenir une population de points en s'inspirant de l'évolution des espèces vivantes.
Parmi ces méthodes, nous pouvons distinguer celles utilisant des opérateurs explicites (mutation, croisement, etc.) telles que (la programmation évolutionnaire, les algorithmes génétiques, etc.), et celles ayant implicitement des règles de transitions telles que "the Space Partition Algorithm" (SPA), "l'algorithme macro évolutionnaire" (MGA), "l'optimisation par essaim de particules" (PSO), etc.

3.3 Présentation de l'algorithme PSO

L'Optimisation par Essaim de Particules (OEP), connu sous le nom anglophone de Particle Swarm Optimization (PSO), est un algorithme inscrit dans la famille des algorithmes évolutionnaires.
Il a été proposé par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio psychologue) en 1995 [22].

Cette méthode trouve sa source dans les observations faites lors des simulations informatiques de vols groupés d'oiseaux et de bancs de poissons de Reynold [24], Heppner et Grenander [23].
Autrement-dit, elle s'inspire fortement de l'observation des relations grégaires d'oiseaux migrateurs, qui pour parcourir de « longues distances » (migration, quête de nourriture, parades aériennes, etc.), doivent optimiser leurs déplacements en termes d'énergie dépensée, de temps, (etc.), comme par exemple la formation en forme de V présentée dans la Figure 3.1.

FIGURE 3.1 – Volée d'Anser en formation en V [25].

Le déplacement de ses animaux en essaim est complexe, sa dynamique obéit à des règles et des facteurs bien spécifiques qu'il s'agit de cerner :

- Chaque individu dispose d'une certaine intelligence « limitée » (qui lui permet de prendre une décision).
- Chaque individu doit connaître sa position locale et disposer d'informations locales de chaque individu se trouvant dans son voisinage.
- Obéir à ces trois règles simples, « rester proche des autres individus », « aller dans une même direction » ou « voler à la même vitesse ».

Tous ses facteurs et règles sont indispensables pour le maintien de la cohésion dans l'essaim, ceci par l'adoption d'un comportement collectif complexe et adaptatif.

3.4 Mode de fonctionnement de l'algorithme PSO

La population dans l'algorithme PSO est nommée essaim, chaque individu du groupe est dit particule. Le déplacement de toute particule (comme indiqué ci-dessus) est régi par des règles et conditions bien spécifiques, influencé par le mouvement des autres particules du voisinage.

Dans un tel contexte, ce déplacement à une signification et doit parallèlement répondre à une logique, fondement même du PSO. Il consiste à chercher un optimum dans un voisinage donné, ce déplacement est influé par les trois composantes suivantes :

- Une composante d'inertie : la particule s'efforce de suivre instinctivement son cap de déplacement.
- Une composante cognitive : la particule fait tout pour se diriger vers la meilleure position rencontrée jusqu'à présent.
- Une composante sociale : la particule s'inspire également de l'expérience, du parcours des autres particules, pour se diriger vers la meilleure position rencontrée par ses voisins.

Ce déplacement est illustré dans la Figure 3.2.

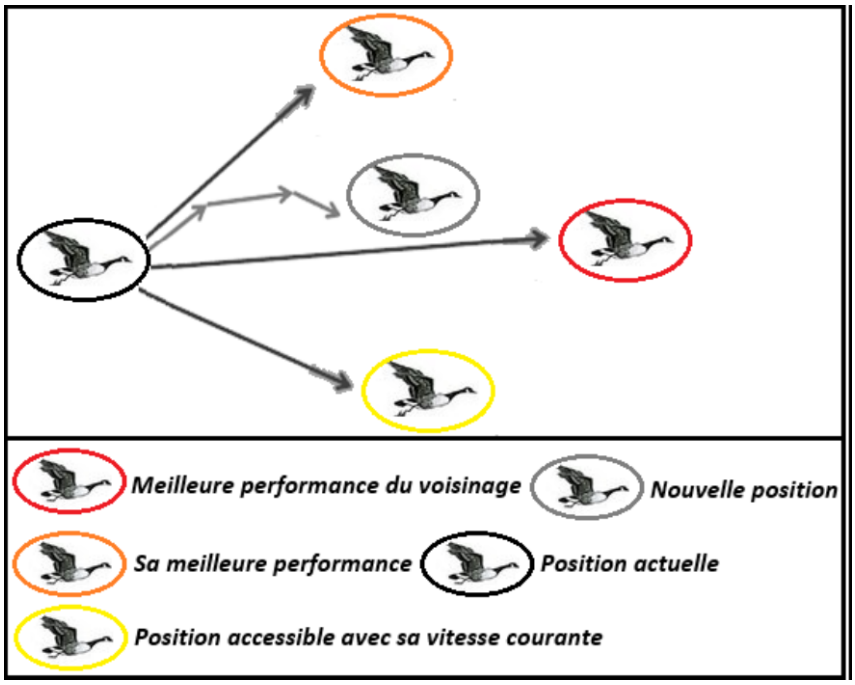


FIGURE 3.2 – Déplacement d'une particule.

A terme, on se rend compte que toutes les particules, après un certain nombre d'itérations, convergent vers une même position, somme toute la meilleure rencontrée par l'ensemble des particules. Cela ne signifie pas pour autant, dans l'absolu, que c'est la meilleure position du voisinage, juste que c'est la meilleure rencontrée !

3.4.1 Composantes de la PSO

Pour être en mesure d'utiliser le PSO, il est indispensable de définir un espace de recherche (composé de particules) et une fonction "objectif" à optimiser. La méthode de l'algorithme consiste alors à déplacer ces particules de telle sorte qu'elles trouvent l'optimum (comme explicité précédemment), elles doivent disposer :

- De données relatives à leurs positions, connaître leurs coordonnées avec comme condition qu'elles soient comprises dans l'espace de définition.
- De la meilleure position qu'elles ont rencontrées.
- De la meilleure position rencontrée par leur voisinage et le résultat de leur fonction «objectif".
- De leur vitesse qui leur permet de se déplacer et de changer de position au fil des itérations.
- D'un voisinage, c'est le sous-ensemble de particules qui interagit directement avec la particule (surtout celle possédant la meilleure position).

D'après Maurice Clerc et Patrick Siarry [26] , l'évolution d'une particule n'est finalement qu'une fusion de trois types de comportements :

- Egoïste (se déplacer suivant sa vitesse actuelle).
- Conservateur (revenir en arrière en prenant en compte sa meilleure performance).
- Panurgien (suivre aveuglément le meilleur de tous en considérant sa performance).

Finalement on remarque un compromis psycho-social entre d'une part la confiance en soi et d'autre part l'influence des relations sociales.

3.5 Notion de voisinage

Chaque particule dispose d'un sous-ensemble d'autres particules avec lequel elle est en interaction, c'est le voisinage de la particule. Cet entrelacement de rapports entre toutes les particules est assimilé à la sociométrie ou à la topologie de l'essaim. On dénombre deux types de voisinage :

3.5.1 Voisinage géographique

C'est un voisinage dynamique où les voisins sont les particules les plus proches. A chaque itération, les nouveaux voisins ou groupes doivent être réajustés en se référant à une distance prédéfinie dans l'espace de recherche. C'est donc bien un voisinage dynamique tel qu'illustré sur la Figure 3.3.

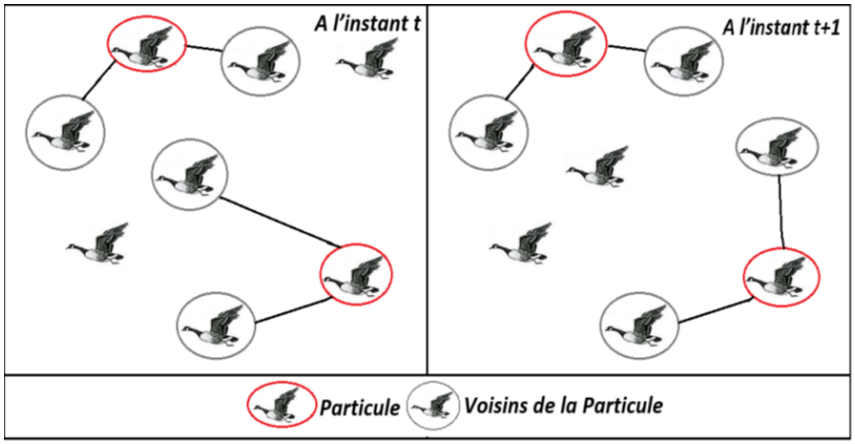


FIGURE 3.3 – Voisinage géographique à l' instant t et $t+1$.

Dans cet exemple, on va supposer que le voisinage d'une particule est un groupe formé des deux particules les plus proches.Dans la figure 3.3 , La notion de voisinage dynamique est mise en évidence puisque pour un même essaim à l'instant « t » et à l'instant « $t+1$ » le voisinage n'est plus le même.

3.5.2 Voisinages sociaux

Ce type de voisinage est considéré comme statique, les voisins restent figés, autrement-dit, ils demeurent inchangés. C'est le voisinage auquel on a le plus souvent recours, en raison :

— De sa simplicité de programmation.

— Parce qu'il offre un meilleur rapport temps/coût, en termes de calcul.

— Dans un scénario de convergence, un voisinage social s'oriente forcément vers un voisinage géographique.

Le voisinage d'une particule constitue la structure du réseau social, à l'intérieur duquel les particules communiquent entre elles.Dans ce type, différents topologies de voisinages

ont été proposées [27]-[28] :

- **Topologie en étoile (a) :** le réseau social est complet, chaque particule est attirée vers la meilleure particule notée L_{best} et communique avec les autres.
- **Topologie en anneau (b) :** chaque particule communique avec n voisines immédiates ($n = 3$). Chaque particule tend à se déplacer vers la meilleure particule dans son voisinage local notée L_{best}.
- **Topologie en rayon (c) :** une particule "centrale" est connectée à tous les autres. Seule cette particule centrale ajuste sa position vers la meilleure, si cela provoque une amélioration l'information est propagée aux autres.
- **Topologie Von Neumann (d) :** chaque particule est connectée à ses quatre particules voisines. Chaque particule tend à se déplacer vers la meilleure dans son voisinage local notée L_{best}.

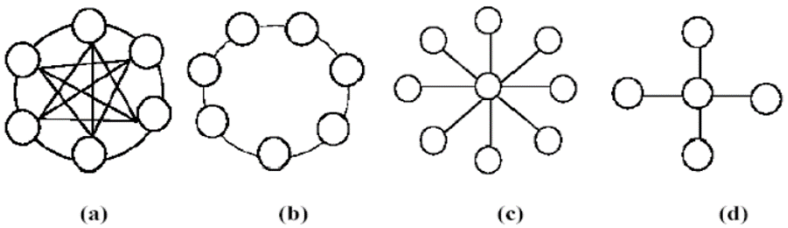


FIGURE 3.4 – Différentes topologies de Voisinages sociaux.

3.6 Algorithme PSO

L'algorithme PSO est initialisé par une population de N solutions potentielles Aléatoires, interprétées comme des particules se déplaçant dans l'espace de recherche. Chaque particule i est modélisée par sa position $\overrightarrow{x_l}$ et un vecteur de changement de position $\overrightarrow{v_l}$ (appelé vélocité ou bien vitesse).

La position d'une particule est influencée par sa meilleure position visitée par elle-même $\overrightarrow{y_l}$ (la meilleure position personnelle désignant sa propre expérience) et la meilleure position parmi ses particules voisines $\overrightarrow{\hat{y}_l}$ (la meilleure position globale, désignant l'expérience de particules voisines).

Selon la taille du voisinage, on peut identifier deux modèles de PSO : le modèle GL_{best}, si le voisinage d'une particule est l'ensemble de l'essaim, et le modèle L_{best} ,dans lequel le voisinage d'une particule est un sous-ensemble de l'essaim.

La performance de chaque particule (comment les particules se rapprochent de l'optimum global) est mesurée par une fonction fitness qui dépend du problème d'optimisation à traiter.

La meilleure position personnelle de la particule i est la meilleure position (c'est- à-dire, celle qui résulte de la meilleure valeur de la fitness) visitée par particule i. Soit f la fonction objectif (nous supposons qu'il s'agit en fait de minimisation), alors la meilleure position personnelle d'une particule est mise à jour comme suit :

$$y_i(t+1) = \begin{cases} y_i(\boldsymbol{t}) \text{ si } f\left(\boldsymbol{x}_i(\boldsymbol{t}+\mathbf{1})\right) \geq f(\boldsymbol{y}_i(\boldsymbol{t})) \\ \boldsymbol{x}_i(\boldsymbol{t}+\mathbf{1}) \text{ si } f\left(\boldsymbol{x}_i(\boldsymbol{t}+\mathbf{1})\right) < f(\boldsymbol{y}_i(\boldsymbol{t})) \end{cases} \tag{3.1}$$

La meilleure position globale est déterminée selon le modèle G_{best} comme suit :

$$\hat{\mathbf{y}}(\mathbf{t}) \in \{\mathbf{y}_0(\mathbf{t}), \mathbf{y}_1(\mathbf{t}), \ldots\ldots, \mathbf{y}_s(\mathbf{t})\} \text{ et } f(\hat{\mathbf{y}}(\mathbf{t})) = \min\left\{f\left(\mathbf{y}_0(\mathbf{t})\right), f\left(\mathbf{y}_1(\mathbf{t})\right), \ldots., f\left(\mathbf{y_s}(\mathbf{t})\right)\right\} \tag{3.2}$$

où s dénote la dimension de l'essaim. Pour chaque itération de l'algorithme PSO, l'étape de la mise à jour de la vitesse v_i est spécifiée pour chaque dimension $j \in 1 \ldots . N_d$, où N_d est la dimension du problème (espace de recherche).

D'où, V_{ij} représente le $J^{\text{iéme}}$ élément du vecteur de la vitesse de la particule i. La mise à jour de la vitesse de particule est :

$$v_{ij}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{\omega} \boldsymbol{v}_{ij}(\boldsymbol{t}) + \boldsymbol{\rho}_{1j}\left(\boldsymbol{y}_{ij}(\boldsymbol{t}) - \boldsymbol{x}_{ij}(\boldsymbol{t})\right) + \boldsymbol{\rho}_{2j}\left(\widehat{\boldsymbol{y}}_i(\boldsymbol{t}) - \boldsymbol{x}_{ij}(\boldsymbol{t})\right) \tag{3.3}$$

Tels que : $\boldsymbol{\rho_1}$ et $\boldsymbol{\rho}_2$ sont les coefficients de confiance, ils pondèrent les tendances de la particule à vouloir suivre son instinct de conservation ou son panurgisme. Les variables aléatoires $\boldsymbol{\rho_1}$ et $\boldsymbol{\rho}_2$ peuvent être définis de la façon suivante :

$$\begin{cases} \rho_1 = r_1 c_1 \\ \rho_2 = r_2 c_2 \end{cases} \tag{3.4}$$

où $\boldsymbol{r_1}$ et $\boldsymbol{r}_2$ suivent une loi uniforme sur $[\mathrm{O}, 1]$ et $\boldsymbol{c_1}$ et $\boldsymbol{c_2}$ sont des constantes positives déterminées de façon empirique [29], et suivant la relation $c_1 + c_1 \leq 4$ $-\omega$ est le facteur

d'inertie qui a été introduit, pour la première fois, par Shi et Eberhart [30], il sert comme une mémoire de vitesses antérieure, pour contrôler l'influence de la vitesse obtenue au pas précédent. Un grand facteur d'inertie provoque une grande exploration de l'espace de recherche alors qu'un petit facteur d'inertie concentre la recherche sur un petit espace.

— La composante cognitive, $\boldsymbol{y}_i(\boldsymbol{t}) - \boldsymbol{x}_i(\boldsymbol{t})$ représente l'expérience individuelle de la particule.

— La composante sociale $\widehat{y}_i(\boldsymbol{t}) - \boldsymbol{x}_i(\boldsymbol{t})$, représente la communication sociale (la croyance de l'essaim).

Selon Van den Bergh [29], la relation entre le coefficient de l'inertie et les constantes d'accélération devraient satisfaire l'équation suivante pour garantir la convergence.

$$\frac{c_1 + c_2}{2} - 1 < \omega \leq 1. \tag{3.5}$$

Autrement, les particules peuvent exposer une divergence de comportement cyclique. Les détails sur la relation entre le coefficient de l'inertie et les constantes d'accélération se trouvent dans [31]-[29]-[32].

La modification de vitesse peut aussi être limitée par une vitesse maximale $\boldsymbol{V_{max}}$ définie par l'utilisateur, pour éviter que les particules se déplacent trop rapidement d'une région à une autre dans l'espace de recherche, et provoquer une convergence prématurée [29].

La position x_i de la particule i, est modifiée en utilisant l'équation suivante :

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{3.6}$$

Les équations de mise à jour de position et de vitesse sont exécutées plusieurs fois, jusqu'à atteindre un nombre maximal d'itérations, donné par l'utilisateur, ou un manque de progrès dans l'amélioration de la meilleure solution trouvée, pour un nombre d'itérations consécutif spécifié par l'utilisateur.

La qualité de particules est mesurée en utilisant une fonction fitness f, qui reflète l'optimalité d'une solution particulière. L'algorithme de base de PSO se décrit comme suit (voir figure 3.5) [33] [34] :

Algorithme *PSO*

Pour chaque particule $i \in 1, \ldots, s$ **Faire**
Initialiser aléatoirement x_i ;
Initialiser aléatoirement v_i ;
Mettre $y_i = x_i$;
Fin pour
Répéter
Pour chaque particule $i \in 1, \ldots, s$ **Faire**
Calculer la fitness $f(x_i)$ de la particule i ;
Mettre à jour y_i en utlisant l'equation **(3.1)** ;
Mettre à jour $\widehat{y}$ en utlisant l'equation **(3.2)** ;
Pour chaque dimension $j \in 1 \ldots, N_d$ **Faire**
Mettre à jour la vitesse $v_{i,j}$ en utilisant l'equation **(3.3)** ;
FinPour
Mettre à jour la position x_i **(3.6)** ;
FinPour
Jusqu'au critère d'arrêt soit satisfait

FIGURE 3.5 – Algorithme d'optimisation par essaim particulaire.

L'algorithme PSO peut être schématisé à l'aide d'un organigramme se trouvant dans la Figure 3.6 [33]

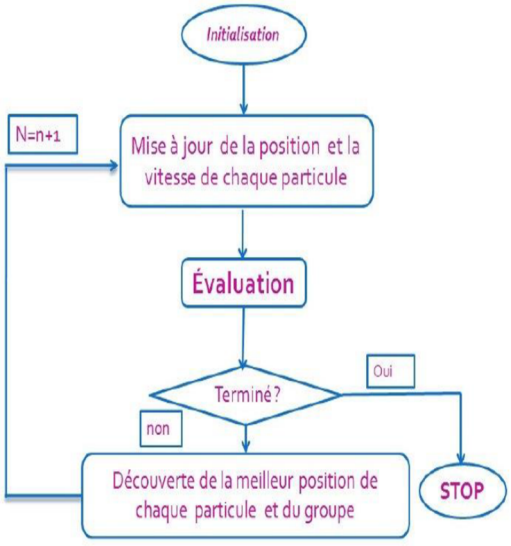


FIGURE 3.6 – Organigramme « méthode des essaims particulaires ».

3.7 Conclusion

Dans ce chapitre, nous avons présenté en détail l'algorithme d'optimisation par essaim particulaire (PSO), ses paramètres et son principe de fonctionnement.Vu que cette méthode, qui est inspirée du monde du vivant, a rencontré un succès remarquable depuis sa création, grâce à sa simplicité, nous allons l'adopter pour l'améliorer la méthode de block-matching qui sera l'objectif du chapitre suivant.

Chapitre 4

Conception et implémentation

4.1 Introduction

Dans ce chapitre, nous allons détailler l'adaptation de l'algorithme PSO pour améliorer l'amélioration de la méthode de BM. Au début, nous présentons la conception du notre système BM-PSO, ensuite nous décrivons son implémentation et enfin nous exposons les expérimentations et les résultats obtenus.

4.2 Motivation et positionnement du problème

L'estimation de Mouvement (EM) est un processus qui consiste à estimer, à partir d'une séquence d'images, le mouvement apparent des objets composant une scène. Les méthodes de mise en correspondance de blocs ou Block Matching (BM) sont les méthodes d'EM les plus utilisées grâce à leur implémentation simple et efficace.
La première méthode de BM adopte une stratégie de recherche exhaustive (FSA : full search algorithm) dans l'image de référence en évaluant tous les blocs existants dans la fenêtre de recherche. Avec la stratégie de recherche exhaustive, le bloc le plus similaire est trouvé, et par conséquent, la précision d'estimation de mouvement est élevée. Cependant, elle est très lourde en temps de calcul, ce qui rebute certains utilisateurs. Pour remédier à ce problème, nous allons traiter le problème de BM comme étant un problème d'optimisation, en adoptant l'algorithme d'optimisation par essaim de particules (PSO : Particle Swarm Optimization) pour le résoudre. L'objectif de thème :

- Implémentation de l'algorithme de recherche exhaustive de BM.
- Adaptation de l'algorithme PSO pour résoudre le problème de BM

4.2.1 Architecture générale du système

L'architecture générale de notre système BM-PSO qui concerne l'amélioration de la méthode de BM par l'utilisation de l'algorithme PSO est représentée comme suit :

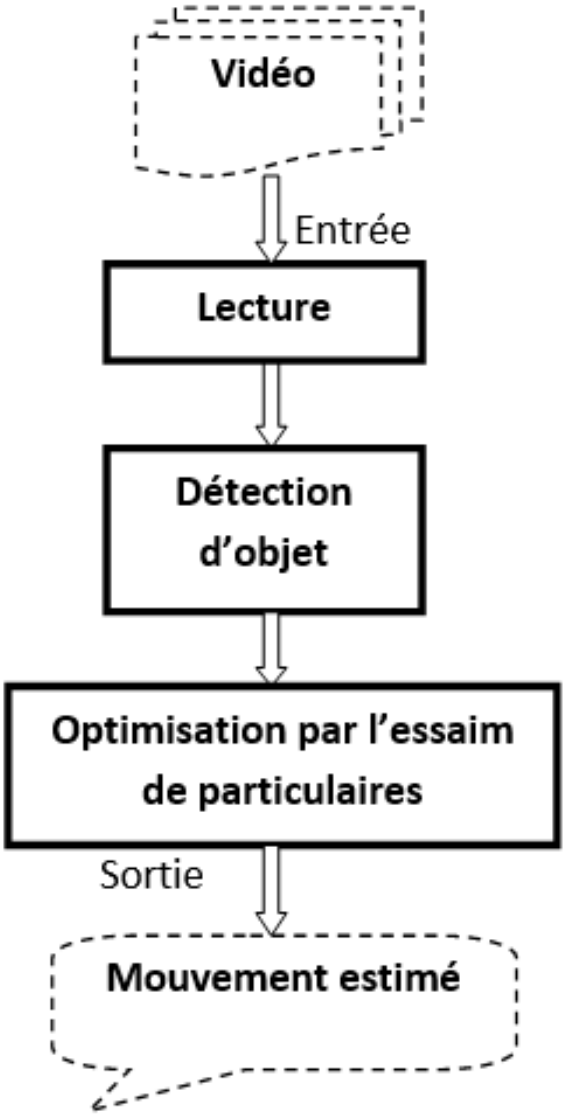


FIGURE 4.1 – L'architecture générale du système BM-PSO.

Comme il est montré dans l'architecture précédente, on peut diviser notre système en trois grands modules.

Lecture vidéo

Dans ce module, vous pouvez lire des vidéos à partir de fichiers ou directement depuis la caméra, via plusieurs commandes dans Matlab, que nous verrons dans la partie d'implémentation. Cette étape est nécessaire pour obtenir des données vidéo, telles que l'extraction de la longueur et de la largeur des images. Ainsi que l'extraction de la série d'images qui composent cette vidéo pour l'application de notre système.

Détection d'objet

Dans ce module, on va détecter les objets en mouvement dans la séquence d'images, en extrayant la taille et la position de chaque objet. Deux méthodes peuvent être utilisées pour détecter les objets en mouvement : Soustraction de l'arrière-plan et Segmentation par contour.

Optimisation par essaim de particules

Dans ce module, nous appliquons l'algorithme d'optimisation de l'essaim de particules, qui forme une approche d'intelligence de groupe pour résoudre des problèmes d'optimisation.
L'algorithme de cette approche, tel que développé, est basé sur un concept simple et peut être implémenté en quelques lignes de code. De plus, il n'utilise que des opérateurs mathématiques.
- Le système doit fournir en sortie :

- Un mouvement estimer (vecteur de mouvement représenté le déplacement de chaque objet en mouvement).

4.2.2 Architecture détaillée

Dans cette partie, on va présenter séparément chaque partie du système proposé en détaillant le principe du travail de chaque partie.

Détection d'objet

La détection du mouvement constitue la deuxième étape d'un notre système après la lecture de vidéo, elle permet de détecter les objets mobiles sur la scène. C'est une étape critique et difficile car, elle doit être robuste aux variations de la luminosité de la scène et la présence des ombres. Il existe différentes méthodes de détection du mouvement, dans notre travail, nous avons utilisé deux méthodes, qui sont comme suit :

- Soustraction de l'arrière-plan.
- Segmentation par contour.

Voire le diagramme suivant :

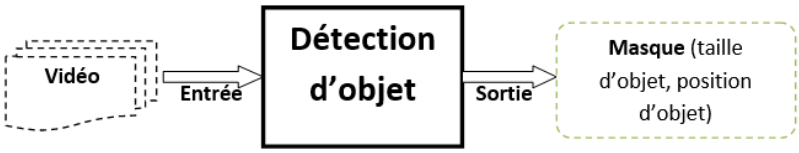


FIGURE 4.2 – L'architecture détection d'objet

a) Soustraction de l'arrière-plan :

L'image de l'arrière-plan d'une scène contient les éléments statiques de cette scène, c'est la représentation de l'environnement dans lequel les objets évoluent. Cette image est considérée comme une image de référence qui se soustrait de chaque image de la séquence afin d'extraire les objets en mouvement [36].

Cette méthode dans sa version simple, consiste à faire la différence des intensités des pixels $I(x, y)$ d'une image I prise à l'instant t, par rapport aux intensités des pixels $B(x, y)$ d'une image de référence d'arrière-plan B.

Elle permet de détecter les pixels (x, y) ayant subi un changement d'intensité significatif. Ces pixels forment les objets en mouvement. Cela est modélisé par :

$$val = abs|I(x, y)B(x, y)| \quad \text{Si } val > M$$

Alors pixel (x, y) de l à l'instant t est en mouvement.

Sinon Le pixel (x, y) est statique.

Avec M un seuil fixé manuellement.

b) Segmentation par contour :

La deuxième approche que nous présentons consiste à définir un contour fermé Manuellement (Il y a un processus prêt dans MATLAB qui met en œuvre cette approche(imfreehand)). [37].

Ce contour est une courbe qui détermine la taille de l'objet d'intérêt (voir figure 4.3).

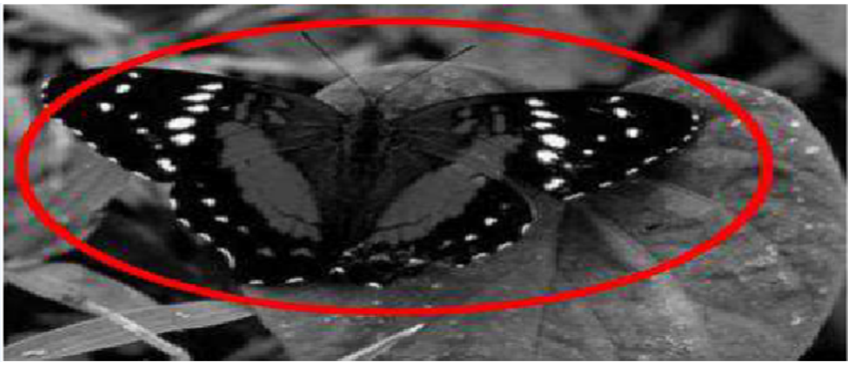


FIGURE 4.3 – Segmentation par contour.

Optimisation par essaim de particules

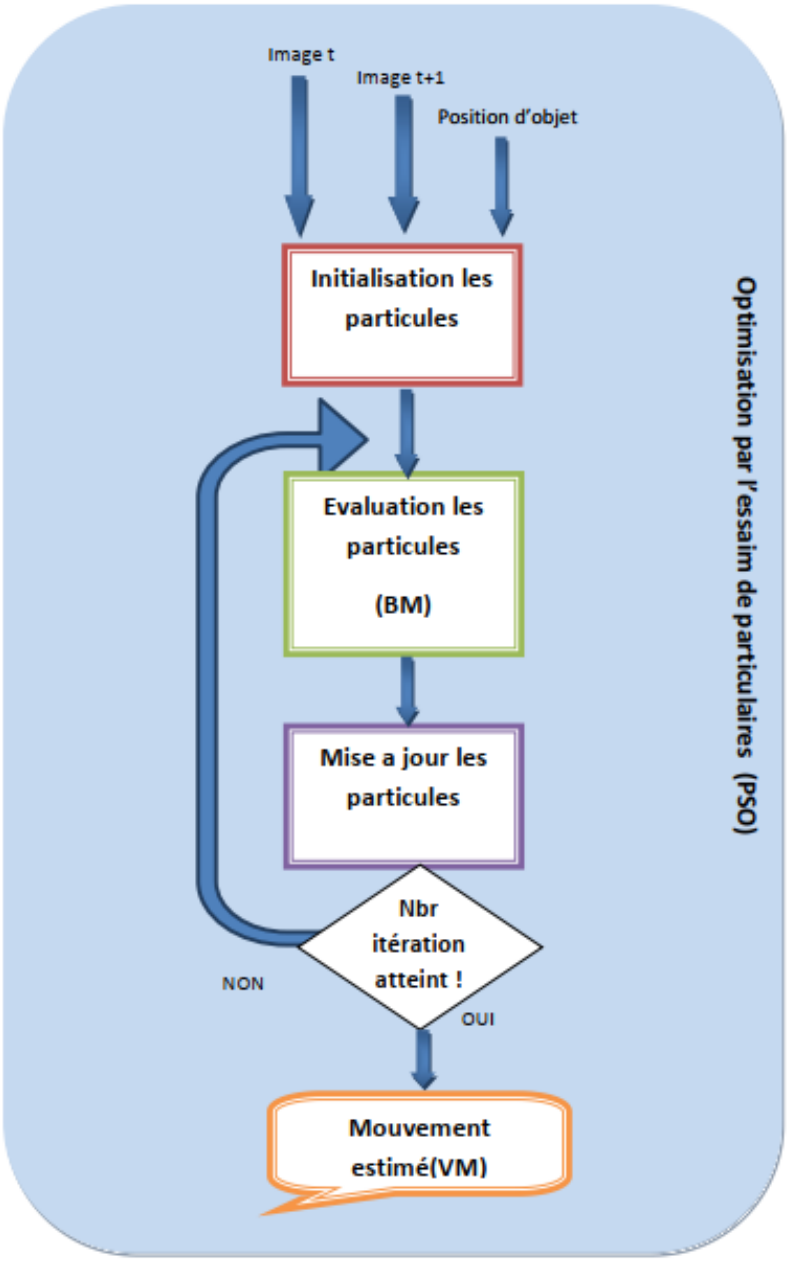


FIGURE 4.4 – L'architecture d'optimisation par essaim de particulaires (PSO)

Initialisation les particules

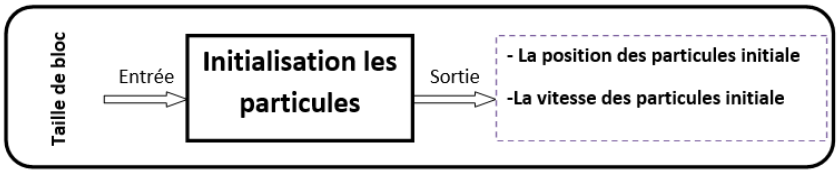


FIGURE 4.5 – L'architecture des Initialisation les particules

a) Les vitesses initiales de particules doivent être initialisées aléatoirement selon une loi uniforme sur $[0, 1]$.

b) Pour déterminer la position des particules, il faut passer par deux étapes :
   — Diviser l'image $(t+1)$ en des blocs de même taille de bloc de l'objet de l'image t.
   — Affectation du centre de chaque bloc à une particule (position initiale de particule = centre d'un bloc)$\boldsymbol{X}_i(CX_i, CY_i)$. Initialement, la meilleure position personnelle d'une particule :$\boldsymbol{Y}_i = X_i(CX_i, CY_i)$.

Evaluation les particules

L'évaluation de la position de chaque particule $\boldsymbol{x}_{ij}$ se fait par le calcul de la fonction d'objectif qui est une mesure de distorsion du bloc affecté à la particule (BDM : The Mean Absolute Difference) qui doit être minimisée , elle est donnée par l'équation :

$$\mathrm{MAD} = \frac{1}{\mathrm{N}^2} \sum_{\mathrm{i}=0}^{\mathrm{N}-1} \sum_{\mathrm{j}=0}^{\mathrm{N}-1} |\mathrm{Cij} - \mathrm{Rij}|$$

où N est la taille du macrobock, $\boldsymbol{C}_{ij}$ et $\boldsymbol{R}_{ij}$ sont les pixels qui sont comparés dans le macrobloc actuel et le macrobloc de référence, respectivement.

Macrobloc que les résultats du moindre coût est celui qui correspond au bloc courant le plus proche.

Après avoir calculé la fonction d'objectif, nous trouvons deux positions :

a) La meilleure position personnelle d'une particule $\overrightarrow{y_l}$ (c'est- à-dire, celle qui résulte de la meilleure valeur de la fitness) est mise à jour comme suit :

$$y_i(t+1) = \begin{cases} y_i(t) \text{ si } f\left(x_i(t+1)\right) \geq f(y_i(t)) \\ x_i(t+1) \text{ si } f\left(x_i(t+1)\right) < f(y_i(t)) \end{cases}$$

b) La meilleure position globale $\overrightarrow{\hat{y}_l}$ est déterminée comme suit :
$\hat{\mathbf{y}}(\mathbf{t}) \in \{\mathbf{y}_0(\mathbf{t}), \mathbf{y}_1(\mathbf{t}), \ldots\ldots, \mathbf{y_s}(\mathbf{t})\}$ et $f(\hat{\mathbf{y}}(\mathbf{t})) = \min\{f(\mathbf{y}_0(\mathbf{t})), f(\mathbf{y}_1(\mathbf{t})), \ldots., f(\mathbf{y_s}(\mathbf{t}))\}$.

Mise à jour de particules

a) La mise à jour de la vitesse de particule est déterminée par l'équation suivante :

$$v_{ij}(t+1) = \omega v_{ij}(t) + \rho_{1j}(y_{ij}(t) - x_{ij}(t)) + \rho_{2j}(\widehat{y}_i(t) - x_{ij}(t))$$

Tels que :

- ω est le facteur d'inertie, il sert comme une mémoire de vitesses antérieures, pour contrôler l'influence de la vitesse obtenue au pas précédent.
- ρ_1 et ρ_2 sont les coefficients de confiance, peuvent être définis de la :

$$\begin{cases} \boldsymbol{\rho_1} = \boldsymbol{r_1 c_1} \\ \boldsymbol{\rho}_2 = \boldsymbol{r_2 c_2} \end{cases}$$

  Où $\boldsymbol{r_1}$ et $\boldsymbol{r}_2$ suivent une loi uniforme sur $[\mathrm{O}, 1]$ et $\boldsymbol{c_1}$ et $\boldsymbol{c_2}$ sont des constantes positives déterminées de façon empirique, et suivant la relation $c_1 + c_1 \leq 4$.
- $\boldsymbol{y}_i(\boldsymbol{t}) - \boldsymbol{x}_i(\boldsymbol{t})$ représente l'expérience individuelle de la particule.
- $\widehat{\boldsymbol{y}}_i(\boldsymbol{t}) - \boldsymbol{x}_i(\boldsymbol{t})$, représente la communication sociale.

La modification de vitesse peut aussi être limitée par une vitesse maximale $\boldsymbol{V_{max}}$ définie par l'utilisateur, pour éviter que les particules se déplacent trop rapidement d'une région à une autre dans l'espace de recherche, et provoquer une convergence prématurée.

b) La mise à jour de la position de particule est déterminée par l'équation suivante :

$$x_i(t+1) = x_i(t) + v_i(t+1).$$

Les équations de mise à jour de position et de vitesse sont appliquées sur chaque composant du vecteur de vitesse et de position.

4.3 Implémentation

4.3.1 L'environnement matériel de système

Notre système est développé sous l'environnement :

— Micro-Ordinateur portable Dell (Inspiron 3558) : Intel(R) Core (TM) i3-3217U CPU @ 2.20 GHz (4 CPUs) RAM 2 G SSD 120 GB .

— Système d'exploitation Windows 764 bits.

4.3.2 L'environnement software de système

MATLAB

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

- Maths et calcul.
- Développement d'algorithmes.
- Modélisation, simulation et prototypage.
- Analyse, exploration et visualisation des données.
- Graphiques scientifiques et techniques.
- Développement d'applications, y compris la création d'interface utilisateur graphique.

Les outils utilisés

Computer Vision

Computer Vision Toolbox™ contient des algorithmes, des fonctions et des applications pour concevoir et tester des systèmes de Computer Vision, de vision $3D$ et de traitement vidéo. Cet outil peut être utilisé pour la détection et le pistage d'objets ainsi que pour la détection, l'extraction et la correspondance de caractéristiques.

Image Processing

Image Processing Toolbox™ propose un ensemble complet d'algorithmes standard de référence et d'applications pour le traitement d'images, l'analyse, la visualisation et le développement d'algorithmes. Les opérations que vous pouvez effectuer sont la segmentation des images, l'amélioration des images, la réduction du bruit, les transformations géométriques, le recalage des images et le traitement des images $3D$.

Les applications d'Image Processing Toolbox vous permettent d'automatiser les processus courants de traitement d'images. Vous pouvez, de manière interactive, segmenter des données d'images, comparer des techniques de recalage d'images et traiter par lots de grandes quantités de données. Les applications et les fonctions de visualisation vous permettent d'explorer des images, des volumes $3D$ et des vidéos, d'ajuster le contraste, de créer des histogrammes et de manipuler des régions d'intérêt (ROI).

GUI

Les interfaces graphiques (ou interfaces homme-machine) sont appelées GUI (pour Graphical User Interface) sous MATLAB. Elles permettent à l'utilisateur d'interagir avec un programme informatique, grâce à différents objets graphiques (boutons, menus, cases à cocher...). Ces objets sont généralement actionnés à l'aide de la souris ou du clavier (voir figure 4.6).

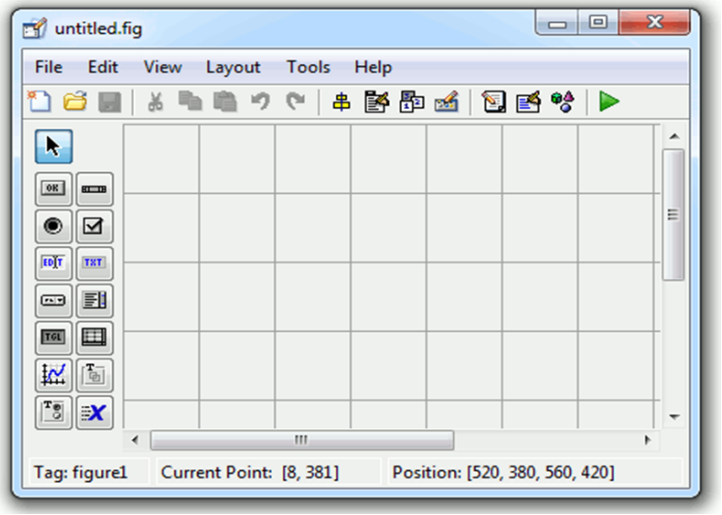


FIGURE 4.6 – Fenêtre principale du GUIDE.

4.3.3 Présentation des interfaces

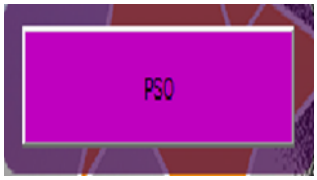
Dans cette section, nous avons présenté les interfaces graphiques de notre système, ou ces interfaces graphiques sont créées à l'aide de l'outil GUI qui est écrit en Matlab.

Écran de l'interface de système

FIGURE 4.7 – L'interface de notre système.

Cette interface offre à l'utilisateur d'effectuer les opérations suivantes :

Cliquez sur ce bouton pour exécuter la méthode de bloc-matching.



Cliquez sur ce bouton pour exécuter l'algorithme de PSO.

L'interface de la méthode de block-matching

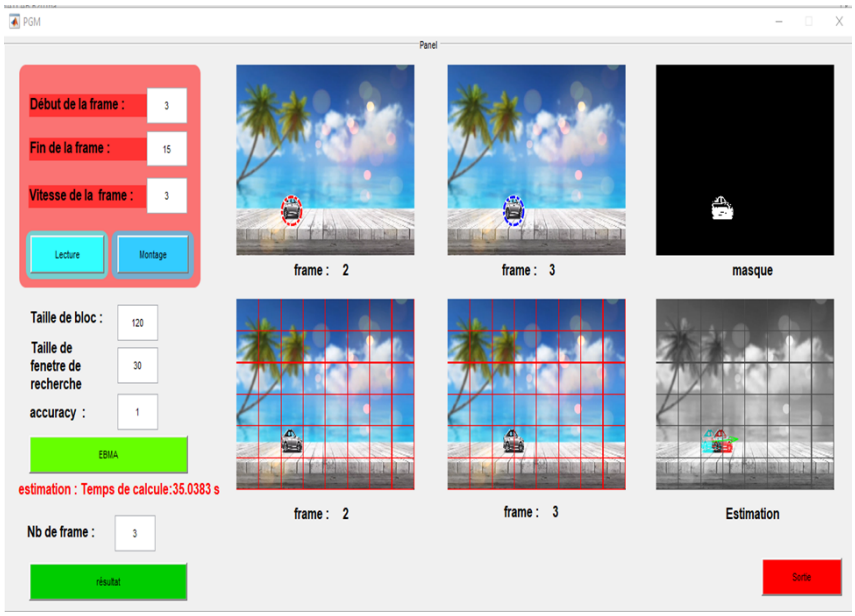


FIGURE 4.8 – L'interface de la méthode de bloc-matching.

Cette interface permet de :

- Lecture et montage vidéo.
- Sélectionné la taille de bloc et fenêtre de recherche.
- Ensuite sélectionner les deux images (image à l'instant t, image $t-1$) pour appliquer la méthode de BM et afficher le résultat (qui sont les vecteurs de mouvement des blocs).
- Enfin, afficher le temps de calcul de cette méthode.

L'interface d'optimisation par essaim de particules

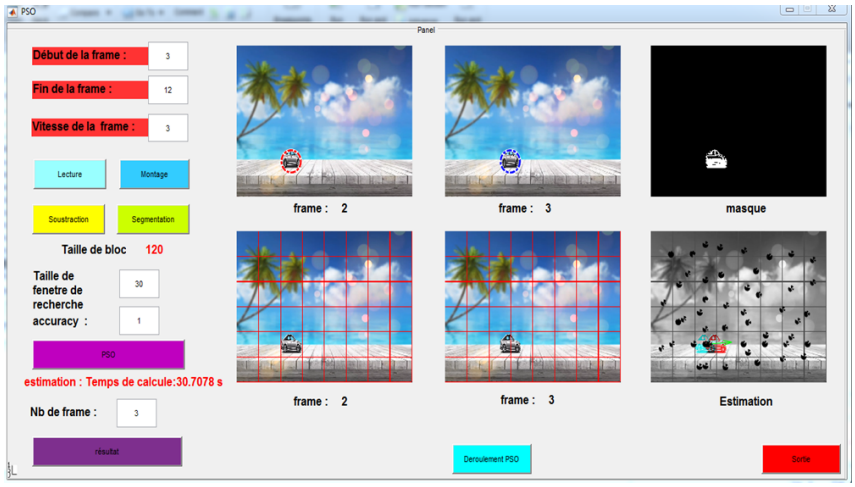


FIGURE 4.9 – l'interface d'optimisation par essaim de particules

Cette interface permet de :

- Lecture et montage vidéo.
- Détection d'objet et puisque c'est l'utilisateur qui choisit la méthode (il y a deux méthodes :
  1. Soustraction de l'arrière-plan.
  2. Segmentation par contour) puis extrait la taille du bloc pour initialiser les particules.
- Ensuite sélectionner les deux images (image à l' instant t, image $t-1$) pour appliquer l'algorithme PSO et afficher le résultat (qui sont les vecteurs de mouvement des blocs).
- Enfin, afficher le temps de calcule de cette méthode.
- 

Cliquez sur ce bouton pour afficher le courbe déroulement de PSO (voir figure 4.10) .

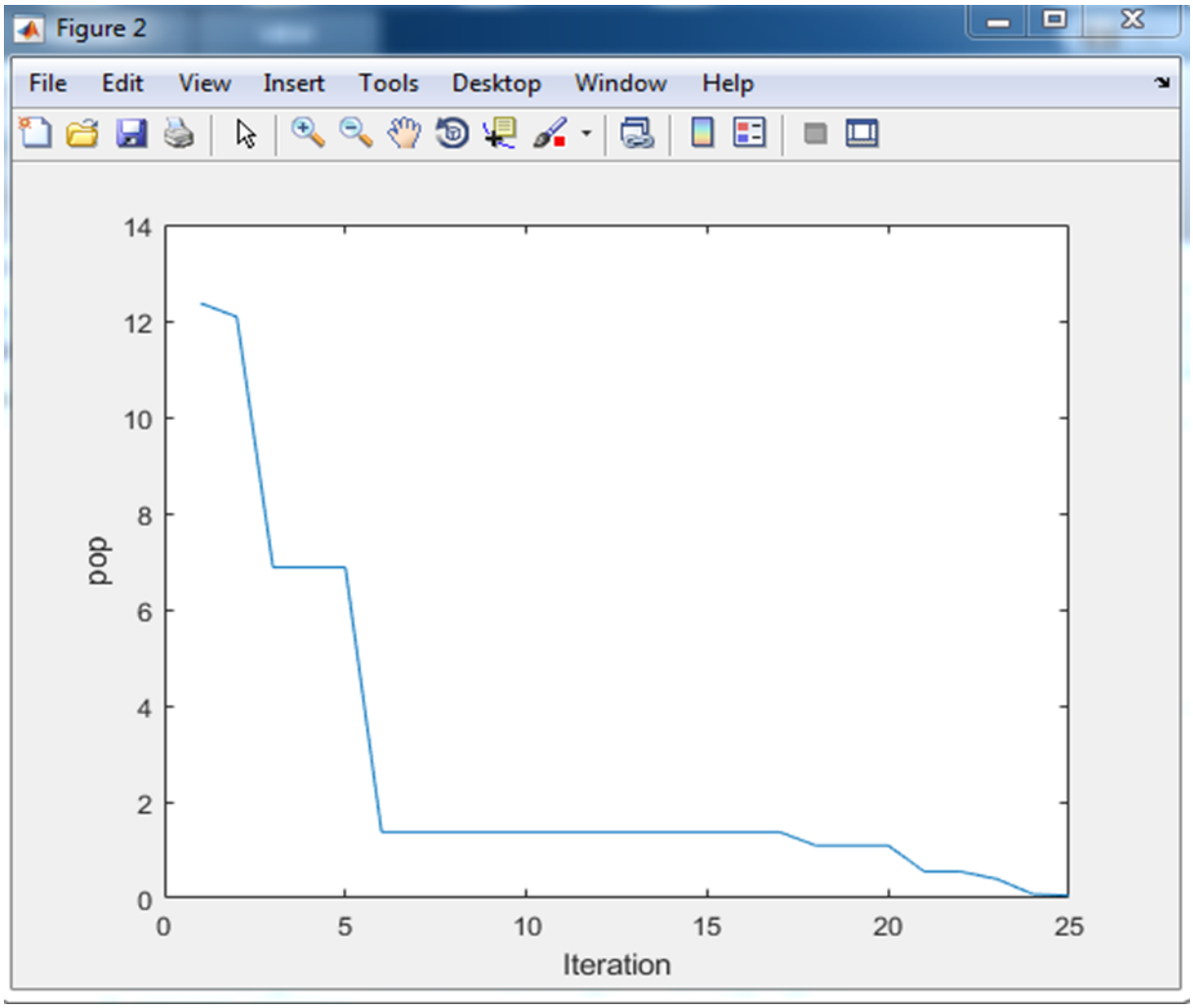


FIGURE 4.10 – Déroulement de PSO.

4.4 Expérimentations et résultats obtenues

Les paramètres du notre système sont les coefficients de confiance ($C1$ et $c2$) et Le facteur d'inertie ($\boldsymbol{\omega}$) et Une vitesse maximale $\boldsymbol{V_{max}}$.Pour trouver les meilleurs valeurs pour ces paramètres nous avons fait plusieurs expérimentations.

4.4.1 Première expérimentation

Dans la première expérimentation, nous allons entrer les variables suivantes dans notre algorithme :

$$\begin{cases} (C1 = 3 \text{ et } C2 = 1) \\ V_{\max} = 200 \\ w = w_{\max} - t \times ((w_{\max} - w_{\min}) / \max\limits_{\text{iteration}}). \end{cases}$$

$\boldsymbol{w}$	$\boldsymbol{C1}$	$\boldsymbol{C2}$	$\boldsymbol{V_{max}}$	résultat
$\boldsymbol{W_{max} = 0.9}$	3	1	200	divergence de particules.
				vitesse élevée dans le mouvement des
$\boldsymbol{W_{min} = 0}.2$				particules.

TABLE 4.1 – Résultat de première expérimentation.

4.4.2 Deuxième expérimentation

Dans la deuxième expérimentation, nous allons utiliser les variables suivantes dans notre algorithme :

$$\begin{cases} (C1 = 3 \text{ et } C2 = 1) \\ V_{\max} = 200 \\ w = w_{\max} - t \times ((w_{\max} - w_{\min}) / (\max\limits_{\text{iteration}})) \end{cases}$$

$\boldsymbol{w}$	$\boldsymbol{C1}$	$\boldsymbol{C2}$	$\boldsymbol{V_{max}}$	résultat
$\boldsymbol{W_{max} = 0.9}$	3	1	200	divergence de particules.
				vitesse élevée dans le mouvement des
$\boldsymbol{W_{min} = 0.2}$				particules.

TABLE 4.2 – Résultat de deuxième expérimentation.

4.4.3 Troisième expérimentation

Dans la troisième expérimentation, nous allons entrer les variables suivantes dans notre algorithme :

$$\begin{cases} (C1 = 2 \text{ et } C2 = 2) \\ \mathrm{V}_{\max} = 70 \\ w = w_{\max} - t \times ((w_{\max} - w_{\min})/(\max_{\text{iteration}})). \end{cases}$$

$\boldsymbol{w}$	$\boldsymbol{C1}$	$\boldsymbol{C2}$	$\boldsymbol{V_{max}}$	résultat
$\boldsymbol{W_{max}}$	3	1	100	divergence de particules.
$\boldsymbol{= 0.5}$				vitesse moyenne dans le mouvement
$\boldsymbol{W_{min} = 0.5}$				des particules.

TABLE 4.3 – Résultat de troisième expérimentation.

Après ces expériences, nous avons conclu que les meilleurs paramètres de l'algorithme d'essaim de particules sont :

$$\begin{cases} (C1 = 2 \text{ et } C2 = 2) \\ \mathrm{V}_{\max} = 70 \\ w_{\max} = 0.9 \text{ et } w_{\min} = 0.2. \end{cases}$$

4.5 Comparaison et discussion des résultats

Après avoir déterminé les meilleurs paramètres de l'algorithme de l'essaim de particules, nous avons effectué un test de performance de cet algorithme sur une vidéo, et comparé les résultats obtenus avec ceux de la méthode de bloc-matching (recherche exhaustive), en utilisant pour chaque méthode les mêmes paramètres, à savoir la taille du bloc et la zone de recherche. Le choix de la dimension du bloc120 et de la fenêtre de recherche 30.

4.5.1 Résultats obtenus avec la méthode de block matching

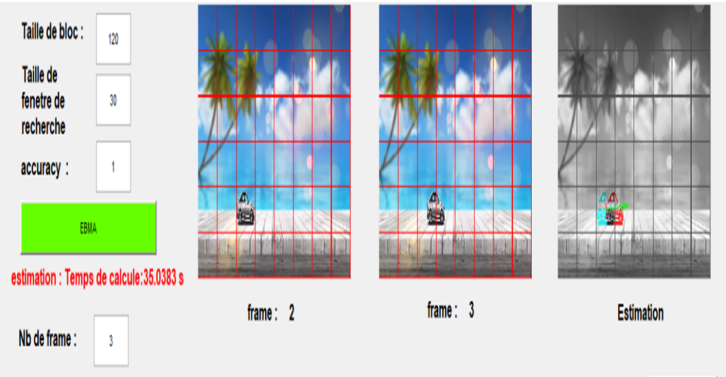


FIGURE 4.11 – Résultats obtenus par l'algorithme de block matching

4.5.2 Résultats obtenus avec l'algorithme de l'algorithme de l'essaim de particules

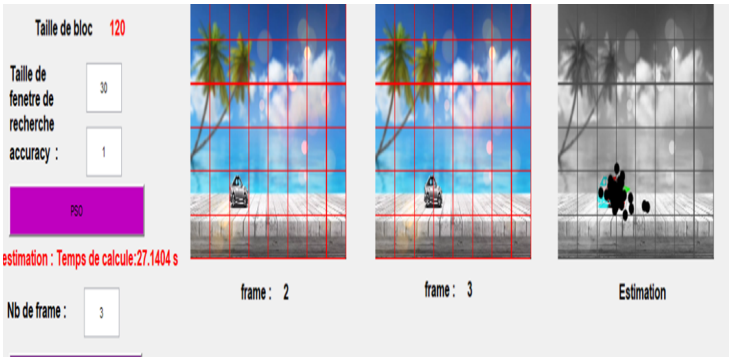


FIGURE 4.12 – Résultats obtenus avec l'algorithme de l'algorithme de l'essaim de particules.

méthode	Temps de calcul
BM	35 s
PSO	27 s

TABLE 4.4 – Le temps de calcul pour les deux méthodes.

4.6 Discussion des résultats

D'après les résultats par les deux méthodes, nous remarquons que l'algorithme de bloc-matching sont assez performants avec un gain en nombre de calculs (voir image 4.11) .Où nous trouvons que le temps de calcul est 34 secondes. Cela est dû au fait que cet algorithme effectue une recherche complète de l'image de manière séquentielle, ce qui lui permet de donner une bonne estimation du mouvement dans un grand temps.
Par conséquent, nous avons implémenté l'algorithme d'essaim de particules, qui a donné des résultats efficaces (voir image 4.12), Où l'on trouve que l'estimation du mouvement est efficace, c'est-à-dire que le vecteur de mouvement est dans la même direction d'objet. Et le temps de calcul est moindre par apport l'algorithme de bloc-matching 27 secondes. C'est pour la raison que cet algorithme utilise des particules qui sont localisées aléatoirement dans l'image et effectue une recherche intelligente de manière parallèle.

4.7 Conclusion

Dans ce chapitre, nous avons présenté en détail la conception et l'implémentation de notre système BM-PSO qui s'intéresse à l'amélioration de la méthode de BM par l'utilisation de l'algorithme PSO. Nous avons fait une étude expérimentale concernant l'ajustement des paramètres du système et l'évaluation des résultats obtenus en les comparant avec la méthode recherche exhaustive de BM qui aussi est implémentée.

Conclusion générale

Notre travail s'est situé dans le contexte d'estimation de mouvement qui est une opération permettant d'améliorer les techniques de compression et de codage vidéo. Cette opération est cruciale. En effet, d'une part la qualité de l'estimation influe sensiblement sur les performances de la compression et d'autre part c'est l'opération qui nécessite le plus de puissance de calcul.

L'algorithme de BM est la technique d'estimation de mouvement la plus populaire, en raison de sa simplicité de mise en œuvre et de son efficacité. FSA (Full search algorithm) est une technique de recherche exhaustive de BM qui donne une précision élevée d'estimation de mouvement cependant, elle est très lourde en temps de calcul.

Afin d'obtenir une estimation de mouvement plus rapide, nous avons reformulé le problème de BM comme étant un problème d'optimisation et nous avons adopté l'algorithme PSO pour le résoudre.

Apres avoir fait une étude sur les différents concepts liés au contexte de notre travail, nous avons présenté notre stratégie BM-PSO d'adaptation de l'algorithme PSO, pour améliorer la méthode de BM, en présentant sa conception et son implémentation.
Nous avons implémenté l'algorithme de recherche exhaustive FSA et notre algorithme proposé BM-PSO puis nous avons fait une étude expérimentale sur une séquence vidéo. Les résultats obtenus par ces deux algorithmes ont montrés la supériorité de l'algorithme

BM-PSO par rapport au FSA en termes de précision d'estimation de mouvement et en complexité de calcul.

En conclusion, l'algorithme proposé BM-PSO permet d'obtenir informations sur le mouvement des objets dans une scène réelle, il pourra donc être inséré dans diverses applications appartenant à des domaines variés, tels que la médecine, la compression vidéo, le suivi d'objet, la surveillance, la robotique, qui reste comme perspective.

Bibliographie

[1] Abir BETKA. "*Estimation de mouvement par les techniques méta-heuristiques* ". Thèse doctorat : Électronique, l'université Mohamed Khider de Biskra, 2019.

[2] M. Etienne. "*Estimation du flot-optique : contributions et panorama de différentes approches*». Habilitation à diriger des recherches, l'université Rennes 1 Institut de Formation Supérieure en Informatique et en Communication, 2003.

[3] Estimation de mouvement, (http://www.iptvdictionary.com/iptv_dictionary_MPEG_Motion_Estimation_Definition.html , Consulté le : 10/01/2021).

[4] estimation de mouvement, (http://en.wikipedia.org/wiki/Motion_estimation, Consulté le : 18/01/2021).

[5] B. K. P. Horn, B. G. Shunck, "*Determining optical flow*", Artificial intelligence, 1981, Vol. 17, p. 185-203.

[6] H. H.Nagel, "*Displacement vectors derived from second-order intensity variations in image sequences*". CVGIP, 1986. 21(1) : p. 85-117.

[7] D.J. Heeger, "*Optical Flow using Spatiotemporal Filters*", Int. Journal of Computer Vision, 1987, p. 279—302.

[8] La séquence vidéo, (https://www.researchgate.net/figure/A-video-is-a-sequence-of-images-called-frames-Each-frame-is-a-two-dimensional-grid-of_fig2_281719202, Consulté le : 08/01/2021).

[9] F. Cherif. "*Estimation hiérarchique du mouvement par ondelettes géométriques*". Thèse doctorat : Électronique, l'université Mohamed Khider de Biskra, 2015.

[10] Xavier Marichal, *Motion Estimation and Compensation for Very Low Bit Rate Coding*, Thèse PHD, Université Catholique de Louvain, Mai 1998.

[11] Fulvio Moschetti, *A Statistical Approach to Motion Estimation, Thèse PHD*, Ecolepolytechnique Fédérale de Lausanne 2001.

[12] I.Patrs, E.A.Hendriks, R.L.Lagendijk, *Confidence Measure for Block Matching Motion Estimation*, IEEE ICIP 2002.

[13] I. T. U. T. H. 261 REC. Rec, i. t. u. t. h. 261,video codec for audio visual service, 1993.

[14] block matching, (https://cagnazzo.wp.imt.fr/files/2013/05/02_motion_estimation_2012.pdf , Consulté le : 07/02/2021).

[15] block matching, (https://www.cari-info.org/actes2006/39.pdf, Consulté le : 14/02/2021).

[16] KERFA Djoudi. "*Estimation de mouvement d'objets dans une séquence d'images* ". Thèse doctorat : Électronique, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, 2015.

[17] Koga et al . *Block-matching criterion for efficient vlsi implementation of motion estimation*. Electronics Letters , 1981.

[18] DEEPAK TURAGA, MOHAMED ALKANNAL. IEEE : " *Search Algorithms for block- matching motion estimation* " 1998.

[19] Lai-Man Po and Wing-Chung Ma. *four-step search algorithm for fast block motion estimation*. Circuits and Systems for Video Technology, 1996.

[20] A.M. Tourapis et al. *diamond search algorithm for block-matching motion estimation*. Circuits and Systems for Video Technology, 2000.

[21] L. Gacôgne. "*Comparaison entre PSO et autres heuristiques d'optimisation avec opérateurs implicites.*" Séminaire Optimisation par Essaim Particulaire, Paris, France 2003.

[22] J. Kennedy and R. C. Eberhart. "*Particle Swarm Optimization*". In : Proceedings of the IEEE International Conference on Neural Networks IV, 1995.

[23] F. Heppner and U. Grenander. *A stochastic nonlinear odel for coordinated bird flocks*. AAAS Publication, Washington, DC, 1990.

[24] Craig W. Reynolds. "*Flocks, herds, and schools : A distributed behavioral model. Computer Graphics*", 1987.

[25] l'algorithme d'optimisation par essaim de particules (PSO), (http://www.journal.forces.gc.ca/vol13/no2/page26-fra.asp, Consulté le : 25/03/2021).

[26] CLERC et SIARRY, *Une nouvelle métaheuristique pour l'optimisation difficile : la méthode des essaims particulaires*, 2003

[27] J. Kennedy, Small Worlds and Mega-Minds : Effects of Neighborhood Topology on Particle Swarm Performance. In IEEE Congress on Evolutionary Computation , volume III, 1999.

[28] P.Suganthan, *Particle Swarm Optimizer with Neighborhood Operator*. In IEEE Congress on Evolutionary Computation, volume III, 1999.

[29] F.Van den Bergh, *An Analysis of Particle Swarm Optimizers. PhD thesis, Département of Computer Science*, University of Pretoria, 2002.

[30] Y.Shi, R.C.Eberhart, *A modied particle swarm optimizer, IEEE International Conference on Evolutionary Computation*, Piscataway, NJ : IEEE Press, 1998.

[31] M.Clerc, J.Kennedy, *The Particle Swarm : Explosion, Stability, and Convergence in a Multi-Dimensional Complex Space, In Proceedings of the IEEE Transactions on Evolutionary Computation*, volume VI, 2002.

[32] Y.Shi, R.C.Eberhart, *Empirical study of particle swarm optimization*, In : Proc.Congress on Evolutionnary Computation, 1999.

[33] PSO, (http://www.particleswarm.net/SéminaireOEP'2003, Consulté le : 03/03/2021).

[34] l'algorithme d'optimisation par essaim de particules (PSO), (http://www.particleswarm.info/, Consulté le : 12/03/2021).

[35] Méthode des essaims particulaires, (http://www.particleswarm.net/SéminaireOEP'2003, Consulté le : 16/03/2021).

[36] Sadoun abdelbaki & Ouellabi yacine. *Détection et suivi d'objets mobiles. Application dans un environnement de foule.* UNIVERSITÉ ECHAHID HAMMA LAKHDAR D'EL OUED.2015.

[37] Wassima Ait Fares. *Détection et suivi d'objets par vision fondés sur segmentation par contour actifbasé région*. Automatique / Robotique. Université Paul Sabatier - Toulouse *III*, 2013. Français

[38] B. KP. Horn & B. G. Schunck. "*Determining optical flo*w". Artificial intelligence, 1981, vol. 17, *n*. 1-3, *p*. 185-203.

[39] M. Khalid, L. Pénard & E. Mémin. "*Optical flow for image-based river velocity estimation*". Flow Measurement and Instrumentation, 2019, vol. 65, p. 110 - 121, issn. 0955-5986.

[40] C. H. Hsieh & T. P. Lin. "*VLSI architecture for block-matching motion estimation algorithm*". IEEE Transactions on Circuits and Systems for Video Technology, 1992, vol. 2, n. 2, p. 169-175.

[41] H. H. Jong, L. G. Chen & T. D. Chiueh. "*Accuracy improvement and cost reduction of 3-step search block matching algorithm for video coding*". IEEE Transactions on Circuits and Systems for Video Technology, 1994, vol. 4, n. 1, p. 88-90.

[42] J. Cai, & W. D. Pan. "*On fast and accurate block-based motion estimation algorithms using particle swarm optimization*". Information Sciences, 2012, vol. 197, p. 53-64.

[43] K. Bhattacharjee, S. Kumar, H. M. Pandey, et al. "*An improved block matching algorithm for motion estimation in video sequences and application in robotics*". Computers & Electrical Engineering, 2018, vol. 68, p. 92-106.

[44] S. Zhu. "*A new diamond search algorithm for fast block-matching motion estimation*". IEEE Transaction on Image Processing, 2000, vol. 9, n. 2, p. 287-290.

[45] S. I. A. Pandian, G. J. Bala & J. Anitha. "*A pattern based PSO approach for block matching in motion estimation*". Engineering Applications of Artificial Intelligence, 2013, vol. 26, n. 8, p. 1811-1817.

[46] K. Hussain, M. N. Salleh, S. Cheng, et al. "*Metaheuristic research : a comprehensive survey*". Artificial Intelligence Review, 2018, p. 1-43.