الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Mohamed Khider - Biskra
Faculté des sciences et de la technologie
Département de génie électrique
Ref: .....

جامعة محمد خيضر بسكرة
كلية العلوم و التكنولوجيا
قسم الهندسة الكهربائية
المرجع : .....

Thèse présentée en vue de l'obtention
du diplôme de

Doctorat en Sciences

Spécialité : Electronique

Option : Communications

Conception d'un système d'optimisation pour le positionnement de caméras pour la motion capture MOCAP (Utilisation des métaheuristiques OEP et RFS)

Présentée par:

Salim ABDESSELAM

Soutenue publiquement le : 04 juillet 2018

Devant le jury composé de :

Dr. OUAFI Abdelkarim	**Professeur**	**Président**	**Université de Biskra**
Dr. BAARIR Zine-Eddine	**Professeur**	**Rapporteur**	**Université de Biskra**
Dr. MOUSSAOUI Abdelkrim	**Professeur**	**Examinateur**	**Université de Guelma**
Dr. SEGHIR Rachid	**Maître de Conférence 'A'**	**Examinateur**	**Université de Batna**

Dédicaces

À Mes Parents.
À Ma Femme.
À Mes enfants Khalil, Akram et Joumana.
À Mes Frères et Sœurs.

Remerciements

Au terme de ce travail, je tiens à remercier en tout premier lieu notre Dieu le tout puissant et miséricordieux, de m'avoir donné le courage, la volonté, la santé et la patience pour accomplir cette modeste thèse de doctorat.

En second lieu, je remercie et j'exprime ma profonde gratitude à mon directeur de thèse, le Professeur BAARIR Zine-Eddine pour m'avoir encadré, guidé et encouragé pour réaliser mon travail. Je le remercie, aussi, pour les conseils qu'il ma prodigués, sa patience et sa confiance qu'il m'a témoignée, ont été déterminantes pour la réalisation de ce travail.

Je remercie le président du jurys Pr. OUAFI Abdelkarim ainsi que les membres Pr. MOUSSAOUI Abdelkrim et Dr. SEGHIR Rachid, qui ont acceptés de juger mon travail.

J'adresse mes remerciements particuliers au Dr. TOUMI Abida, Pr. Djedi Nour-Eddine et Pr. BITAM Salim, pour leurs avis précieux et fructueux, qui m'ont beaucoup aidés.

Mes sincères et vifs remerciements s'adressent à mes parents pour leurs encouragements et leur soutient moral.

Une belle et grande gratitude à ma femme pour sa patience et son sacrifice durant les années de préparation de ce travail.

Mes remerciements s'étendent également à tout le personnel du laboratoire LESIA, mes collègues, mes étudiants, ainsi que toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

"Merci ..."

Abstract

Motion capture is a technique used in many fields (such as medicine, bio-mechanics, ...). Its purpose is to record the movement of a real object to translate it into a mathematical model exploitable by graphic methods. Several criteria (such as actors number, movement complexity, ...) specify the choice of the motion capture system, whether optical, magnetic or mechanical. The optical system uses markers fixed to specific locations on the object and infrared cameras that capture the light reflected by these markers. The recording makes it possible to reconstruct a three-dimensional movement. This system has advantages compared to others, but it suffers from marker occlusion problems. A good placement of cameras can solve these problems.

The goal of this thesis is to design an optimization algorithm for camera network placement, dedicated to an optical motion capture system. For a good motion capture, the camera network placement consists in finding a combination of parameters called extrinsic parameters (positions and poses) and a combination of intrinsic parameters (depths, aperture angles). The combinatorial optimization method is chosen to solve this type of problems. Thus, we can solve it by recent methods called "meta-heuristics". We choose these methods for their algorithmic simplicity and their ability to find an optimal solution in an acceptable computing time compared to exact methods. In our work, an optimal solution is reached, when each marker is captured (or covered) by at least two cameras during each frame of its movement. This condition is necessary for three-dimensions motion reconstruction.

To reach the first goal of this thesis, the camera network placement problem is solved as a two dimensional placement. As first application, the variant Weight Particle Swarm Optimization (WPSO) is used to optimize the placement of a four-camera network for three scenarios: (1) a single fixed point, (2) a single fixed point with obstacle presence and (3) a single point in movement. In the second application, a variant called Canonical Particle Swarm Optimization (CPSO) is used to optimize a cameras network placement (from 4 to 10 cameras) to covered the movement of 8 markers (fixed on the cube extremities) in a defined scene. The objective of this optimization is achieved for three different scenarios (the cube move in straight line between two obstacles or to a big wall or the cube move between two pillars).

To reach the second goal, the problem of a camera network placement is solved as three dimensional placement. The Stochastic Fractal Search (SFS) technique and CPSO, Modified Canonical Particle Swarm Optimization (MCPSO) variants are used to optimize a cameras network placement (for 4 to 10) to cover a diagonal movement of the cube inside the scene. Although the MCPSO technique present a slight improvement in the results compared to the CPSO technique, the two variants did not achieve their objective except by using the technique SFS; either, that each marker is covered by two cameras ($C_{seuil} = 2$) or by three cameras ($C_{seuil} = 3$) during each frame of its movement.

Keywords

motion capture, MoCap, optical system, camera network, placement, markers, 2D, 3D, meta-heuristics, WPSO, CPSO, MCPSO, SFS.

Résumé

La capture de mouvement est une technique utilisée dans des domaines très diversifiés (telles que la médecine, la biomécanique, etc.). Son objectif est d'enregistrer le mouvement d'un objet réel pour le traduire en un modèle mathématique exploitable par des méthodes graphiques. Plusieurs critères (tels que le nombre d'acteurs, la complexité du mouvement, ...) précisent le choix du système de capture de mouvement, qu'il soit optique, magnétique ou mécanique. Le système optique utilise des marqueurs fixés à des emplacements spécifiques sur l'objet et des caméras infrarouges qui capturent la lumière réfléchie par ces marqueurs. L'enregistrement permet de reconstruire un mouvement tridimensionnel. Ce système a des avantages par rapport aux autres, mais il souffre de problèmes d'occultation des marqueurs. Un bon placement des caméras peut résoudre ces problèmes.

L'objectif de cette thèse consiste en la conception d'un algorithme d'optimisation pour le placement d'un réseau de caméras, dédié à un système de capture de mouvement optique. Pour une bonne capture de mouvement, le placement d'un réseau de caméras consiste à trouver une combinaison de paramètres appelés paramètres extrinsèques (positions et poses) et une combinaison de paramètres intrinsèques (profondeurs, angles d'ouverture). La méthode d'optimisation combinatoire est choisie pour résoudre ce type de problème. Ainsi, nous pouvons le résoudre par des méthodes récentes appelées "méta-heuristiques". Nous avons choisi ces méthodes pour leur simplicité algorithmique et leurs capacités à trouver une solution optimale dans un temps acceptable par rapport aux méthodes exactes. Dans notre travail, une solution optimale est atteinte, lorsque chaque marqueur est capturé (ou couvert) par au moins deux caméras au cours de chaque trame de son mouvement. Cette condition est nécessaire pour la reconstruction du mouvement en trois dimensions.

Pour atteindre le premier objectif de cette thèse, le problème de placement du réseau de caméras est résolu comme un placement bidimensionnel. Comme première application, la variante WPSO est utilisée pour optimiser le placement d'un réseau à quatre caméras pour trois scénarios: (1) un seul point fixe, un seul point fixe avec présence d'obstacles et (3) un seul point en mouvement. Dans la seconde application, une variante appelée CPSO est utilisée pour optimiser le placement d'un réseau de caméras (de 4 à 10 caméras) pour couvrir le mouvement de 8 marqueurs (fixés sur les extrémités d'un cube) dans une scène bien définie. L'objectif de cette optimisation est atteint pour trois différents scénarios (le cube se déplace en ligne droite entre deux obstacles ou vers un grand mur ou le cube se déplace entre deux piliers).

Pour atteindre le second objectif, le problème du placement d'un réseau de caméras est résolu en trois dimensions. La technique Recherche Fractale Stchastique (RFS) et les variantes CPSO, MCPSO sont utilisées pour optimiser le placement d'un réseau de caméras (de 4 à 10) afin de couvrir un mouvement diagonal d'un cube à l'intérieur de la scène. Bien que la technique MCPSO présente une légère amélioration des résultats par rapport à la technique CPSO, les deux variantes n'ont pas atteint leur objectif sauf en utilisant la technique RFS; soit, que chaque marqueur est couvert par deux caméras ($C_{seuil} = 2$) ou par trois caméras ($C_{seuil} = 3$) au cours de chaque trame de son mouvement.

Mots-Clés

capture de mouvement, système optique, réseau de caméras, placement, marqueurs, 2D, 3D, méta-heuristiques, WPSO, CPSO, MCPSO, RFS.

الملخص:

التقاط الحركة هي تقنية تستخدم في مجالات متنوعة للغاية (مثل الطب ، والميكانيكا الحيوية ، ..إلخ). والغرض منه هو تسجيل حركة كائن حقيقي لترجمته إلى مصطلح رياضي يمكن استغلاله بواسطة طرق بيانية. تحدد عدة معايير (مثل عدد الممثلين ، وتعقيد الحركة ، ...) اختيار نظام التقاط الحركة ، سواء أن يكون بصري، مغناطيسي أو ميكانيكي. يستخدم النظام البصري علامات مثبة في أماكن محددة على الجسم و كذلك كاميرات تعمل بالأشعة تحت الحمراء والتي تلتقط الضوء المنعكس من هذه العلامات. يمكن التسجيل من اعادة بناء حركة ثلاثية الأبعاد. هذا النظام له ميزات حسنة مقارنة بالنظم الأخرى، لكنه يعاني من مشاكل احتجاب العلامات. يمكن التموضع الجيد للكاميرات من حل هذه المشاكل.

الهدف من أطروحتنا هو تصميم خوارزمية تحسين تموضع شبكة الكاميرات ، خاصة بنظام التقاط الحركة الضوئي. للحصول على التقاط جيد للحركة ، فإن تموضع شبكة الكاميرات يتحدد من مجموعة من المعلمات تسمى المعلمات الخارجية (المواضع والوجهات) ومجموعة من المعلمات الداخلية (الأعماق وزوايا الفتحة). يتم اختيار طريقة التحسين التوافقي لحل هذا النوع من المشاكل. وهكذا ، يمكننا حلها بالطرق الحديثة المسماة الاستدلال الفوقي. لقد اخترنا هذه الطرق لبساطتها الخوارزمية وقدرتها على إيجاد الحل الأمثل في وقت مقبول مقارنة بالطرق الدقيقة. في عملنا هذا، يتم التوصل إلى الحل الأمثل ، عندما يتم التقاط كل علامة (أو تغطيتها) بواسطة كاميرتين على الأقل خلال كل إطار من حركتها. هذا الشرط ضروري لإعادة بناء حركة ثلاثية الأبعاد.

للوصول إلى الهدف الأول من هذه الأطروحة ، يتم حل مشكلة تموضع شبكة الكاميرا كتموضع ثنائي الأبعاد. كأول تطبيق ، يتم استخدام تقنية التحسين بسرب الجسيمات الموزونة (WPSO) لتحسين وضع شبكة من أربع كاميرات من أجل ثلاثة سيناريوهات: (١) لتغطية نقطة واحدة ، (٢) نقطة واحدة مع وجود عقبة و (٣) نقطة واحدة متحركة . في التطبيق الثاني ، يتم استخدام تقنية التحسين بسرب الجسيمات الكانونيكية (CPSO) لتحسين تموضع شبكة الكاميرات (من ٤ إلى ١٠ كاميرات) لتغطية حركة ٨ علامات (ثابتة على أطراف المكعب) في مشهد محدد. يتم تحقيق الهدف من هذا التحسين لثلاثة سيناريوهات مختلفة (تحرك المكعب في خط مستقيم بين عائقين أو جدار كبير أو تحرك المكعب بين عمودين).

للوصول إلى الهدف الثاني ، يتم حل مشكلة تموضع شبكة الكاميرات كتموضع ثلاثي الأبعاد. يتم استخدام تقنية التحسين بسرب الجسيمات الكانونيكية و تقنية التحسين بسرب الجسيمات

الكانونيكية المعدلة (MCPSO) و تقنية البحث بالكسور العشوائية (SFS) لتحسين تموضع شبكة الكاميرات (من ٤ إلى ١٠) لتغطية حركة قطرية للمكعب داخل القاعة. على الرغم من أن تقنية التحسين بسرب الجسيمات الكانونيكية المعدلة تقدم تحسنًا طفيفًا في النتائج مقارنةً بتقنية التحسين بسرب الجسيمات الكانونيكية ؛ لكن التقنيتين لم تحققا هدفهما إلا باستخدام تقنية البحث بالكسور العشوائية ؛ و ذلك إما أن كل علامة تغطى بكاميرتين فقط أو بثلاث كاميرات خلال كل إطار من حركتها.

الكلمات الدلائلية:

التقاط الحركة، النظام البصري، علامات، تموضع، شبكة الكاميرات، ثنائي الأبعاد، ثلاثي الأبعاد، التحسين التوافقي، التحسين بسرب الجسيمات الكانونيكية، التحسين بسرب الجسيمات الكانونيكية المعدلة ، البحث بالكسور العشوائية.

Productions Scientifiques

Publications dans un journal

- ABDESSELAM, Salim, BETKA, Abir, TOUMI, Abida, et BAARIR, Zine-Eddine. Application of Stand-PSO Technique for Optimization Cameras' 2D Dispositions in a MoCap system. Journal of Applied Computer Science & Mathematics, 2016, no 21.
- Abdesselam, Salim, and Zine-Eddine Baarir. "Three variants Particle Swarm Optimization technique for optimal cameras network two dimensions placement." Journal of Applied Engineering Science & Technology 4.1 (2018): 37-45.

Workshop

- Salim, ABDESSELAM, Zine-Eddine, BAARIR. Une etude sur les parametres extrinseques et intrinseques de la camera. In : IGVA'13 Workshop, Poster (No.1).

Table des matières

Liste des Figures

Liste des Tableaux

Abréviations

2D	deux dimensions
3D	trois dimensions
ACO	Ant Colony Optimization
AG	Algorithmes Génétiques
AE	Algorithmes Evolutionnaires
BP	Best Point
CPSO	Canonical Particle Swarm Optimization
CCD	Charged Coupled Device
CMOS	Complementary Meatl Oxide Semiconductor
DEL	Diode Electro Luminescente
Gen	maximal Generation
GRW	Gaussian Random Walks
IMU	Inertial Measurement Unit
IR	Infra Red
FoV	Field of View
FS	Fractal Search
LAMIH	Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines
LB	Lower Boundary
LED	Light Emitting Diode
MoCap	Motion Capture
MCPSO	Modified Canonical Particle Swarm Optimization
MMC	Markerless Motion Capture

NDM	Nombre Diffusif Maximal
OC	Optimisation Combinatoire
OCF	Optimisation par Colonie de Fourmis
OEP	Optimisation par Essaim de Particules
PSO	Particle Swarm Optimization
PR	Pourcentage de Recouvrement
RFq	Radio Fréquence
RGB	Red Green Bleu
RF	Recherche Fractale
RFS	Recherche Fractale Stchastique
SFS	Stochastic Fractal Search
SPSO	Standard Particle Swarm Optimization
UB	Uper Boundary
UMI	Unité de Mesure Inertielle
WPSO	Weight Particle Swarm Optimization

1

Introduction

1.1 Vue d'ensemble

La capture de mouvement est une technique utilisée comme un processus d'enregistrement d'un mouvement complet ou partiel d'un corps humain, animal ou objet réel, afin de le traduire en un modèle mathématique exploitable. Cette technique utilise des points clés appelés "Marqueurs" qui représentent les parties intéressantes du sujet (articulations, extrémités ...etc.) pour une modélisation numérique en trois dimensions (3D) de son comportement par ordinateur. Cette méthode est utilisée dans des domaines très diversifiés, tels que : *le sport* (pour augmenter les performances des athlètes), *le diagnostique médical et réadaptation* (analyse du système d'équilibre ou connaissance de la physiopathologie de l'appareil squelettique et locomoteur), *la biomécanique* (possibilité de créer des dispositifs avec une conception plus confortable et utile), *les films, animation et les jeux vidéo* (pour créer des films d'animation ou des jeux vidéo avec des mouvements et des actions plus naturelles), *la justice* (reconstitutions d'une scène de crime), ...etc.

Pour un système de capture de mouvement (ou Motion Capture (MoCap), en anglais), la caméra représente l'élément essentiel pour l'acquisition des données (images des marqueurs), issues du déplacement d'un objet ou d'un humain dans une scène de surface bien déterminée. Les images capturées par la caméra présentent une projection du monde (système de coordonnées 3D) sur une image (système de coordonnées à deux dimensions (2D)). Pour que le système soit opérationnel et se rapproche au comportement physique (projection perspective), il faut définir un modèle géométrique pour la caméra basée sur un ensemble de paramètres décrivant le comportement physique du capteur. L'extraction de ces paramètres est dite "phase de calibration".

Actuellement, plusieurs domaines sont en perpétuelle évolution, surtout dans les domaines de la science et de l'industrie (ingénieurs, décideurs, ...etc.), soit pour la conception d'un système mécanique, informatique, ou électronique. Cette évolution est due aux améliorations technologiques qui provoquent à leurs tour l'augmentation de la quantité des données ainsi que leurs types. Les gens sont confrontés à divers problèmes dont la complexité augmente de jour en jour. Par conséquent, il faudrait que les méthodes de traitement utilisées doivent suivre cette évolution. Au début, la tâche consistait à accumuler le maximum de données possibles pour

une exploitation directe; mais aujourd'hui, il faudrait utiliser des méthodes complexes pour l'extraction de l'information utile.

Heureusement, il y a une alternative à ces inconvénients, qui consiste à utiliser des méthodes dont la recherche de la solution du problème est plus rapide et moins encombrante. L'une de ces techniques formalise le problème sous forme d'optimisation. Les méthodes d'optimisation dites "approchées", peuvent trouver une solution satisfaisante avec un temps de calcul acceptable là où les méthodes exhaustives sont trop lentes. Alors, de nombreux problèmes difficiles, peuvent être formulés comme des problèmes d'optimisation [10].

Dans ce travail, nous cherchons à optimiser le placement d'un certain nombre de caméras (de quatre à dix caméras) dans un système MoCap pour couvrir un mouvement virtuel d'un objet. Le but du placement d'un réseau de caméras est que les caméras doivent couvrir les marqueurs fixés sur l'objet par deux caméras (au minimum) en même temps et durant chaque trame de la scène.

Puisqu'il s'agit d'un problème d'optimisation, il est intéressant de résoudre le problème avec une méta-heuristique. Parmi les techniques utilisées actuellement il y a la méta-heuristique d'Optimisation par Essaim de Particules (OEP) (ou Particle Swarm Optimization (PSO) en anglais). Le premier avantage, de l'OEP réside dans sa rapidité en raison de sa simplicité algorithmique. Le second, est que nous avons plusieurs paramètres à ajuster (la position (C_{x1},C_{x2}), la pose φ, la portée (d_{max} et d_{min}), l'angle d'ouverture α de quatre à dix caméras, c'est à dire de 24 jusqu'à 60 paramètres dans le domaine 2D et de 36 jusqu'à 90 paramètres dans le domaine 3D) et quelques contraintes à respecter (marqueur à l'intérieur du champ de vision de la caméra, marqueur vu par trois caméras dans la même trame, angle critique, obstacle).

Dans le cas réel, un système MoCap est un système tridimensionnel, c'est à dire que le champ de vision des caméras infrarouges et le mouvement d'un objet ou d'un humanoïde à l'intérieur de la scène doit être pris dans son cas naturel en 3D.

La technique OEP a abouti à des résultats acceptables dans le domaine 2D, mais elle a montrée ses limites dans le domaine 3D où le nombre de paramètres à optimiser devient élevé. Pour cette raison, nous avons utilisé une méthode plus récente dite "la Recherche Fractale Stochastique RFS" qui a permis d'atteindre l'objectif souhaité là où l'OEP a échoué.

1.2 Contexte et motivations

Lors de la mise en place d'un système MoCap optique, le positionnement du réseau de caméras est une étape qui doit répondre aux besoins de la scène considérée, comme le type de mouvement et la qualité des images qui influent directement sur la précision des résultats.

L'étape de positionnement des caméras autour de l'acteur à l'intérieur du volume de capture est une étape très importante et très coûteuse en termes de durée d'installation et de réglage. Il s'agit d'avoir le plus large recouvrement possible de la scène avec une bonne résolution possible pour les marqueurs (pour ne pas confondre deux marqueurs proches) avec un temps d'exécution minimal. Cette problématique fait l'objet de notre travail de recherche. Par exemple, pour capturer un mouvement de montée et de descente d'un escalier de cinq marches par l'acteur, il a fallu cinq heures de travail de positionnement et d'orientation de dix caméras et un réglage de résolution pour chaque marqueur et en plus, il a fallu l'intervention de l'ingénieur pour une acquisition d'un mouvement de quelques dizaines de secondes. Cette expérience a eu lieu au Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines (LAMIH) à l'université de Valenciennes, France.

1.3 Objectifs et contributions

L'objectif de cette thèse est le développement d'algorithmes d'aide au placement d'un réseau de caméras pour un système de capture de mouvement optique avec des marqueurs passifs. Les principales contributions sont:

- L'utilisation de la méta-heuristique "OEP" pour l'optimisation de placement bidimensionnel pour le recouvrement d'un seul marqueur.
- L'augmentation du nombre de marqueurs et l'introduction d'obstacles pour tester les limites de la technique choisie.
- Le passage à un environnement tridimensionnel adéquat avec le domaine MoCap.
- Une nouvelle application de la méta-heuristique "RFS" dans le domaine MoCap.

1.4 Organisation de la thèse

Cette thèse est organisée comme suit:

Le **chapitre 2** traite l'état de l'art des systèmes de capture de mouvement où la recherche dans ce domaine a débutée en 1973. Nous focaliserons l'étude sur les systèmes de capture de mouvement optiques ainsi que sur d'autres systèmes non-optiques (systèmes magnétiques, mécaniques, inertiels et acoustiques). Nous présenterons aussi un aperçu sur la modélisation de la caméra basée sur l'étude de ses paramètres intrinsèques (qui déterminent la qualité de l'image) et de ses paramètres extrinsèques (qui déterminent le placement de la caméra).

Le **chapitre 3** consiste à présenter brièvement la théorie de la complexité qui s'intéresse à l'étude de la difficulté des problèmes en informatique. Cette théorie choisit l'algorithme de résolution du problème à l'aide de la classification de sa complexité qui peut être de type P, NP ou NP-complet. Dans ce chapitre, nous présenterons "l'optimisation combinatoire" qui admet que n'importe quel problème peut être formulé comme un problème d'optimisation. Elle regroupe une grande classe de problèmes d'optimisation dans de nombreux domaines. Dans notre thèse, nous avons choisi l'étude de deux techniques d'optimisation qui appartiennent à la classe des méta-heuristiques à population, l'OEP et la RFS.

Dans le **chapitre 4**, nous présenterons les contraintes nécessaires utilisées pour assurer le recouvrement bidimensionnel d'un marqueur par un réseau de caméras même avec la présence d'un obstacle qui empêche la visibilité. Ces contraintes sont utilisées pour fonder une fonction de fitness robuste. La technique OEP se base sur cette fonction pour trouver le placement optimal de l'ensemble de quatre caméras comme une première application. Pour une deuxième application, nous appliquerons la technique OEP pour recouvrir trois mouvements différents d'un cube dans une salle rectangulaire où huit marqueurs sont fixés sur celui-ci. Pour l'un de ces mouvements et avec la présence de deux obstacles, nous essayerons de trouver le placement optimal d'un réseau de caméras en utilisant trois variantes de l'OEP où chaque marqueur est capturé par trois caméras.

Le **chapitre 5** présente l'étude du problème dans le domaine tridimensionnel pour le placement optimal d'un réseau de caméras autour d'un cube en mouvement. Comme première application, l'algorithme exige que chaque marqueur doit être capturé uniquement par deux caméras

(condition exprimée par $C_{seuil} = 2$). Comme deuxième application, la condition est d'exiger trois caméras au minimum ($C_{seuil} = 3$) pour extrapoler la complexité de l'optimisation du placement. Dans ce chapitre, nous utiliserons la méthode CPSO et sa variante modifiée (MCPSO), mais vu leurs limitations en 3D, nous utiliserons la méta-heuristique RFS développée par [12] qui donne une nette amélioration par rapport aux deux premières.

Nous terminerons cette thèse par une conclusion générale sur les travaux effectués et quelques perspectives jugées intéressantes pour des travaux futurs, éventuellement.

2

Systèmes de capture de mouvement

MoCap : Etat de l'art

2.1 Historique

Parmi les premiers travaux effectués dans le domaine de la capture de mouvement (ou captation de mouvement ou tout simplement MoCap de Motion Capture, en anglais) sont les analyses d'un mouvement cinématographique étudiant la locomotion animale et humaine par prise de vues en chronophotographie, réalisées par Eadweard James Muybridge (photographe) et Etienne Jules Marey (psychologue), dans les années 1860 (voir figure 2.1). [13]

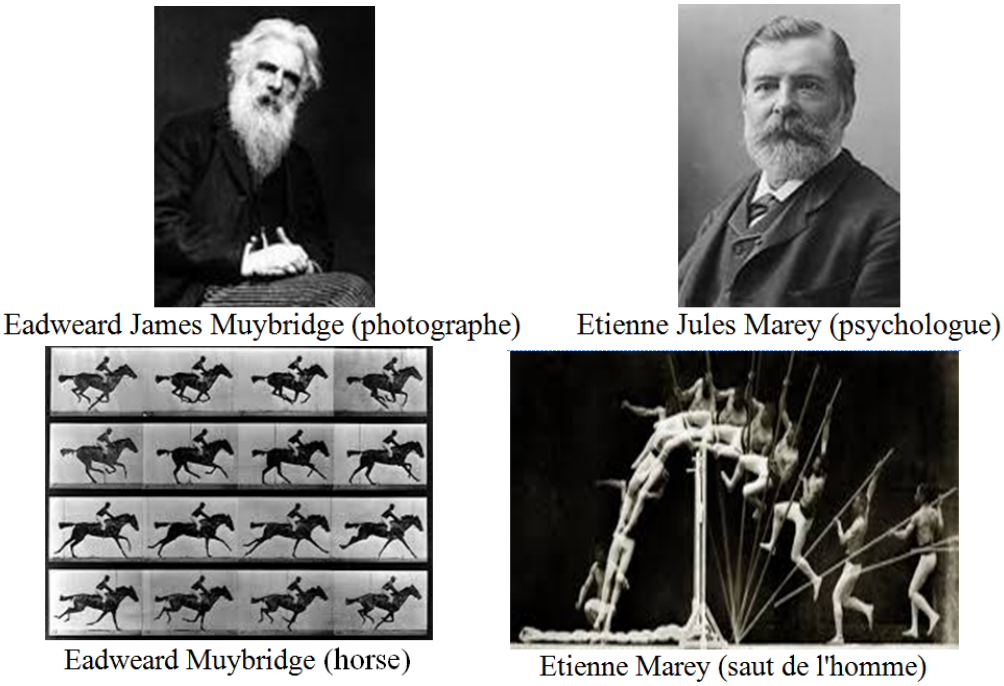


Figure 2.1: Muybridge et Marey.

En 1973, le psychologue Gunnar Johansson a introduit une méthode basée sur le mouvement biologique appelée "Point-light Displays" où un nombre de petits points représentant les articulations principales d'une personne en marche [14]. Cette technique consiste à fixer des marqueurs réflecteurs sur des corps humains pour analyser les trajectoires des articulations. Ces expérimentations représentent les premiers pas d'un travail de recherche intensif [15].

Nous pouvons aussi citer "Christian Wilhelm Braüne" (anatomiste allemand) et son étudiant "Otto Fischer" (figure 2.2), qui ont mis en place la première méthodologie d'analyse biomécanique de mouvement en 3D [16].

Dans le domaine de la réalisation des films animés ou de la production de jeux vidéo et

Figure 2.2: Méthodologie de Braüne et Fischer.

avec le développement de la photographie et de la vidéo numérique, les animateurs ainsi que la communauté de recherche des domaines de la vision par ordinateur et l'informatique graphique sont particulièrement intéressés par l'analyse du mouvement humain. Leur but est de créer un mouvement réaliste, naturel et adapter pour un personnage virtuel auquel le spectateur ou le joueur doit croire [13] [15].

2.2 Introduction

La capture de mouvement est une technique utilisée comme un processus d'enregistrement d'un mouvement complet ou partiel d'un corps humain, animal ou un objet réel, afin de le traduire en un terme mathématique exploitable. Cette technique utilise des points clés appelés "Marqueurs" qui représentent les parties intéressantes du sujet (articulations, extrémités ...etc.) pour une modélisation numérique en trois dimensions (3D) de son comportement par ordinateur. Cette méthode est utilisée dans des domaines très variés, tels que : *le sport* (pour augmenter les performances des athlètes), *le diagnostique médical et réadaptation* (analyse du système d'équilibre ou connaissance de la physiopathologie de l'appareil squelettique et locomoteur), *la biomécanique* (possibilité de créer des dispositifs avec une conception plus confortable et utile), *les films, animation et les jeux vidéo* (pour créer des films d'animation ou des jeux vidéo

avec des mouvements et des actions plus naturelles), *la justice* (reconstitutions d'une scène de crime) ...etc.

La discipline de capture de mouvement étudie le mouvement du corps humain, afin d'avoir une mesure objective et précise [17]:

- Des segments du corps en mouvement (cinématique)
- Des forces de réaction au sol (cinétique)
- Du signal électrique d'activités musculaires (électromyographie).

2.3 Systèmes MoCap

Il y a trois grandes familles pour les systèmes de capture de mouvement : Les systèmes optiques, les systèmes non-optiques et les systèmes hybrides (voir figure 2.3).
Les systèmes MoCap optiques, utilise la lumière comme une source de donnée et les caméras comme capteurs. Parmi ces systèmes, on cite:

- Les systèmes optiques avec marqueurs (passif ou actif)
- Les systèmes optiques sans marqueurs (2D ou 3D)

Les systèmes MoCap non-optiques utilisent d'autres approches (mécanique, inertielle, magnétique et acoustique). Les systèmes hybrides s'intéressent à combiner entre deux ou plusieurs systèmes MoCap existants, pour créer un nouveau système qui intègre les avantages de ces systèmes et/ou pour diminuer les inconvénients ou de franchir leurs limites. L'un des exemples d'hybridation des systèmes MoCap est le système de capture de mouvement inertiel qui utilise un capteur magnétique (magnétomètre) pour la stabilisation horizontale (voir 2.3.3.C) [17].

Selon [5], une autre classification des systèmes de capture de mouvement est basée sur le capteur et la source des données, qu'ils soient placés sur le corps ou à l'extérieur de celui-ci :

- **Système Extérieur-interne :** Il utilise des capteurs externes pour collecter les données à partir de sources placées sur le corp du sujet (système MoCap à base de caméras (capteurs) et marqueurs réflecteurs (sources)).

- **Système Intérieur-externe :** Les capteurs placés sur le corps collectent les données d'une source externe (système MoCap électromagnétique, où les capteurs se déplacent dans un champ électromagnétique externe).

- **Système Intérieur-interne :** Les capteurs et les sources sont placés sur le corps (système MoCap à base de combinaisons électromécaniques où les capteurs sont des potentiomètres et les sources sont les articulations à l'intérieur du corps).

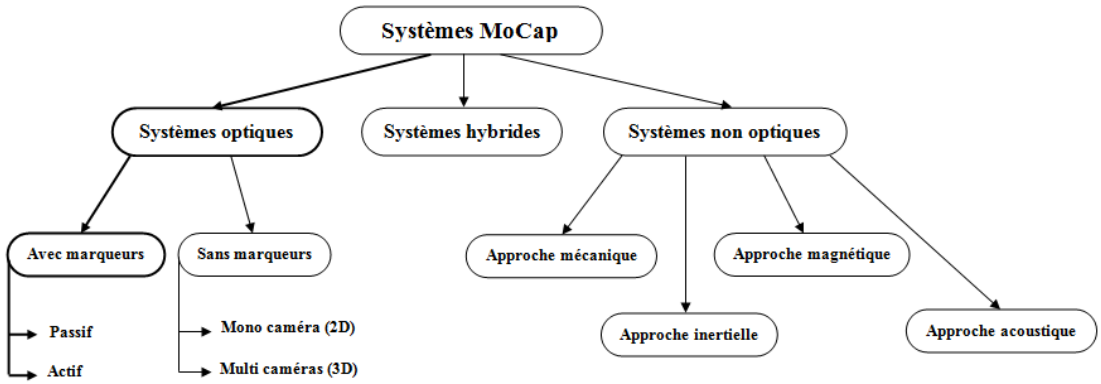


Figure 2.3: Différents systèmes de capture de mouvement.

2.3.1 Système MoCap optique avec marqueurs

Le système le plus précis et le plus utilisé pour la capture de mouvement est le système MoCap optique à marqueurs, qui se base sur la photographie ou l'enregistrement vidéo d'une scène de mouvement d'un acteur équipé de marqueurs optiques, par des caméras pour obtenir des données sous forme d'image 2D puis les faire traiter une unité de calcul.

L'acteur réalise sont mouvement dans un espace appelé "Volume de capture" (figure 2.4). La taille de ce volume est choisi d'une façon que n'importe quel marqueur à l'intérieur du volume soit vu par plusieurs caméras. Selon cette taille, le nombre de caméras à déployer augmente ou diminue.

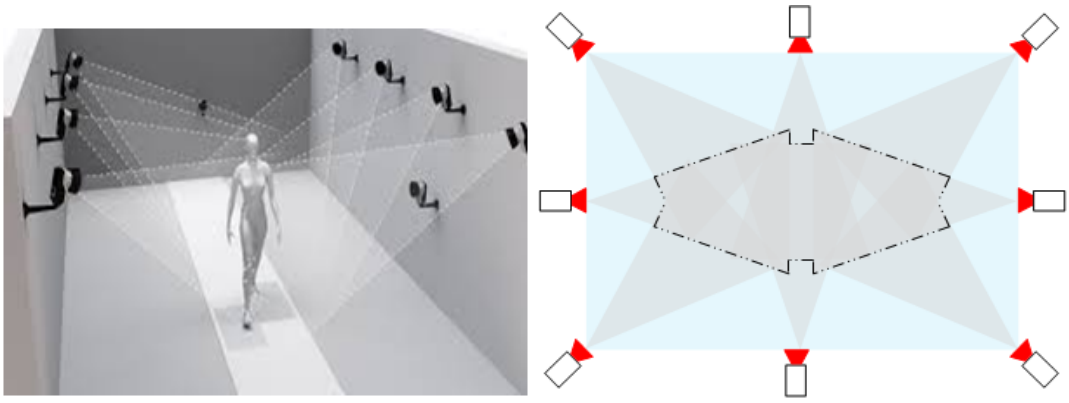


Figure 2.4: Volume de capture

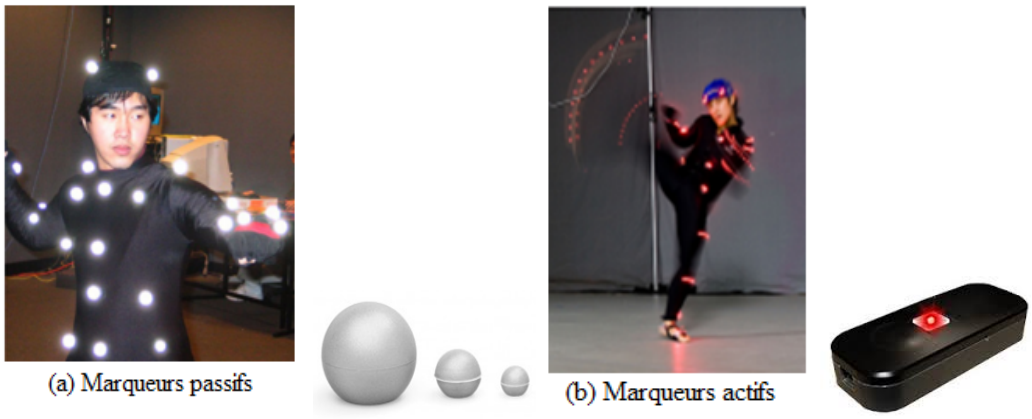
Selon la figure précédente (figure 2.3), le système MoCap optique est divisé suivant le type des marqueurs utilisés, en deux systèmes:

1. Le système MoCap optique avec marqueurs passifs;

2. Le système MoCap optique avec marqueurs actifs.

Pour les deux systèmes, les marqueurs (qu'ils soient actifs ou passifs) sont placés sur les articulations principales du corps de l'acteur. Ce dernier porte un costume spécialement conçu pour fixer les marqueurs aux bonnes articulations. Les marqueurs dits "passifs", sont des petites sphères de différents diamètres (figure 2.5(a)) recouvertes d'une matière rétro-réfléchissante éclairée par des sources lumineuses placées à coté des caméras. Pour identifier chaque marqueur, il faut avoir les positions de marqueurs statiques définis par un "modèle de position". La source de lumière préférée pour ce type de marqueur est la lumière infrarouge (figure 2.6) à cause de sa faible distorsion visuelle pour l'utilisateur (invisible). Par contre, les marqueurs dits "actifs", (figure 2.5(b)) sont des Diode Electro Luminescente (DEL) ou (Light Emitting Diode (LED), en anglais) alimentées par des petites batteries, émettent leurs propre lumière avec une seule longueur d'onde facilement détectable par les caméras (figure 2.7(a)) ou avec différentes longueurs d'onde (figure 2.7(b)) pour que les caméras peuvent différentier chaqu'un des marqueurs par un marquage codé de la lumière.

L'idée d'utiliser des marqueurs optiques est qu'ils sont facilement détectables dans les images enregistrées par les caméras placées d'une manière stratégique autour du volume de capture pour bien suivre le mouvement d'un acteur ou d'un objet (figure 2.8).


(a) Marqueurs passifs

(b) Marqueurs actifs

Figure 2.5: Marqueurs passifs et actifs. [1]

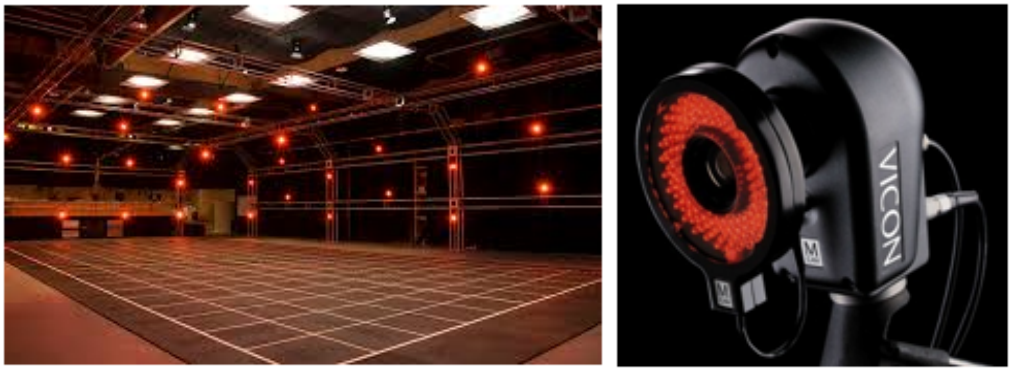


Figure 2.6: Caméras à LED infrarouge.

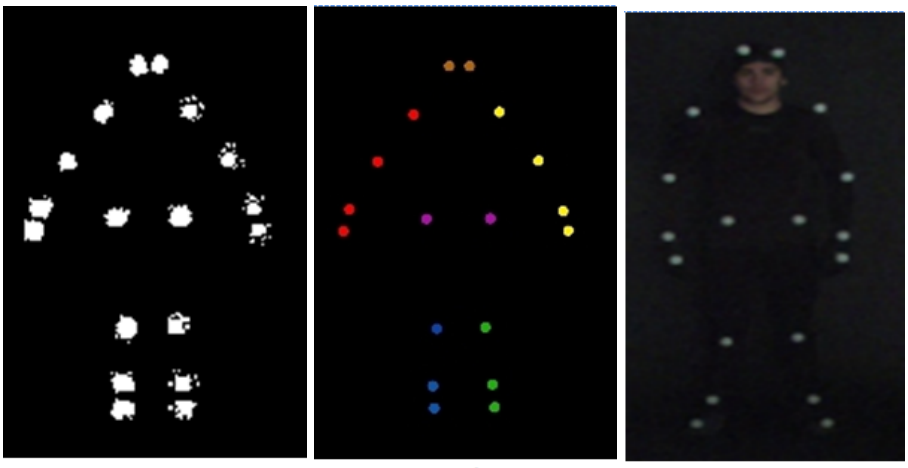

(a) Une seule λ (b) Plusieurs λ (c) Vue réelle

Figure 2.7: Différentes longueurs d'onde pour les marqueurs actifs

Avant l'acquisition des données du mouvement, l'acteur se met en une position modèle un peu spéciale appelée "T-pose" (figure 2.9). Cette position permet à l'opérateur de réaliser les

Figure 2.8: Caméras placées autour de l'acteur [2].

correspondances entre marqueurs et les articulations du modèle à animer, ce qui lui permet aussi d'estimer la position et l'orientation de chaque segment du modèle [15].

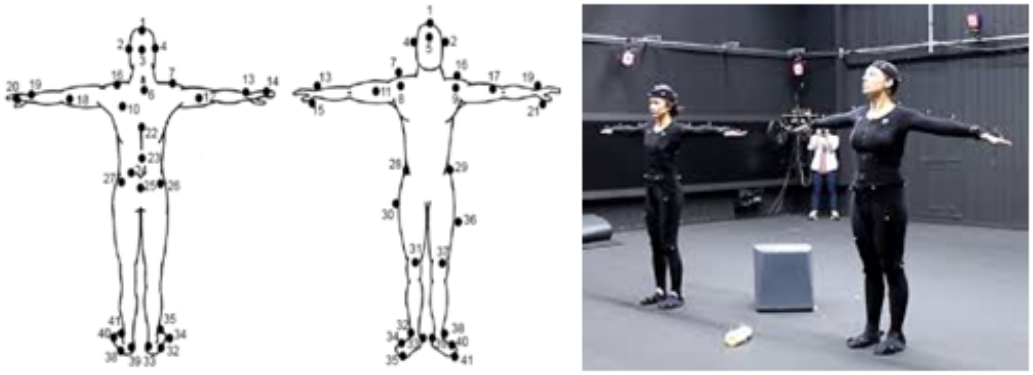


Figure 2.9: T-pose et placements des marqueurs [3].

Un système MoCap nécessite le déploiement d'un nombre minimal de 4 caméras et 32 caméras comme un nombre maximal [5]. Généralement, c'est le volume de capture et le nombre de marqueurs ainsi que la complexité du mouvement à réaliser qui influent sur le nombre de caméras. Cette condition du nombre minimal de caméras, est liée au principe de la reconstitution des coordonnées 3D par "triangulation", de chaque marqueur à partir d'au moins deux de ces images 2D capturées.

Le système soufre par fois d'un phénomène dit "Auto-occlusions" par l'apparition et la disparition de quelques marqueurs dans certaines caméras à cause de l'opacité du corps de l'acteur ou à cause d'un objet placé à l'intérieur de l'espace de travail.

Dans un système MoCap optique, les caméras placées autour de l'acteur captent la lumière

réfléchit par les marqueurs; ainsi, ces derniers apparaissent sur les images comme une série de pixels très lumineux par rapport à l'arrière plan [18]. Avant que la capture d'une scène ne débute, il faut passer par une phase dite "Calibrage" des caméras. Elle consiste à extraire les paramètres extrinsèques et intrinsèques de chaque caméra (pose et position, distance focale, position du centre optique, paramètres de distorsion ...etc. (voir section 2.4) par la combinaison de toutes les images capturées d'un objet dit "kit de calibration" (figure 2.10). Il s'agit d'une règle de forme de "T" dont les dimensions sont connues, placée dans le repère de la salle.

Figure 2.10: Kit de calibration.

Toutes ces informations sont nécessaires pour effectuer la reconstruction 3D (figure 2.11) par une unité de calcul. La calibration doit être faite pour un volume proportionnel au mouvement de l'acteur à étudier. Si le volume est trop grand, la précision sera dégradée et si le volume est trop petit, le mouvement ne peut pas être enregistré dans sa totalité [17].

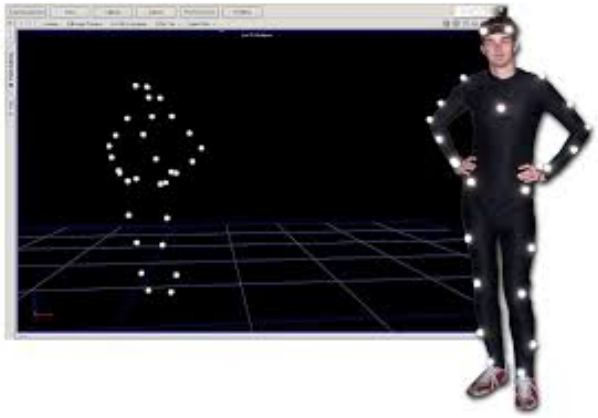


Figure 2.11: Reconstitution 3D.

A. **Avantages:**

Parmi les avantages d'un système MoCap optique, on cite l'absence de câbles ou structures métalliques pouvant gêner le mouvement de l'acteur, ainsi que la légèreté et la petite taille des marqueurs, qui permet à l'acteur de se déplacer librement et à des grandes distances et hauteurs, tant qu'il reste à l'intérieur du volume de capture. Le système permet d'utiliser un grand nombre de marqueurs et une segmentation facile des images capturées ce qui conduit à une bonne précision dans la reconstitution. La haute fréquence de capture (images/second) d'un tel système, qui se traduit par une vitesse d'acquisition suffisamment rapide (surtout pour les mouvements sportifs).

B. **Inconvénients:**

Le système MoCap optique n'est pas un système parfait, il présente quelques inconvénients telle que l'augmentation du nombre de marqueurs aux articulations pour pouvoir obtenir l'orientation, ce qui augmente aussi le temps de calcul. Aussi, tous les marqueurs capturés sont de couleur blanche (marqueurs passifs), ce qui rend l'identification de chaque marqueur presque impossible. Ce problème exige un prètraitement coûteux en temps de calcul. Les surfaces réfléchissantes et les lumières extérieures au système comme celles du soleil (qui renferme de l'infrarouge) provoquent des points parasites. Le problème de l'auto-occlusion durant un temps long engendre des erreurs dans la reconstitution et la perte de données dans les trajectoires et cela demande parfois l'intervention de l'ingénieur. N'oublions pas le coût matériel exhaustif qui varie (entre 100000€ et 250000€ [5]) en fonction du nombre de caméras utilisées.

2.3.2 Système MoCap optique sans marqueurs

Nous avons vu dans le paragraphe précédent les avantages et les inconvénients du système MoCap optique à marqueurs. Le principal avantage d'un tel système est sa précision de reconstruction 3D à cause de la haute résolution des caméras utilisées. Malheureusement, le coût exhaustif de ces mêmes caméras présente l'inconvénient majeur de ce système.

Ici, vient le rôle d'un système MoCap optique sans marqueurs comme une solution aux problèmes dont souffre le système MoCap optique à base de marqueurs. Cette technique ap-

pelée : "Markerless Motion Capture (MMC)" en anglais, consiste à reconstruire le mouvement d'un acteur à partir d'une ou de plusieurs images 2D enregistrées d'une façon à retrouver les positions et les orientations des parties du corps de l'acteur au fil du temps. Chaque partie est modélisée par un "voxel" (un objet intermédiaire entre le pixel et le vecteur), qui représente un nuage de cubes colorés positionnés dans l'espace. Le volume de chaque cube défini approximativement une des parties du corps, pour enfin, extraire le squelette virtuel tout entier de l'acteur (figure 2.12) [19].

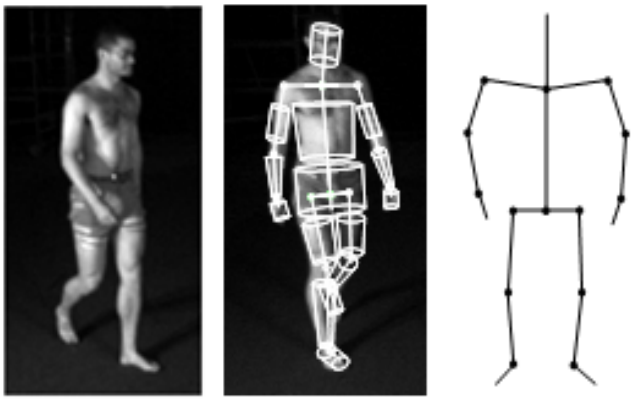


Figure 2.12: Modélisation de squelette par voxel [4].

De ce fait, cette technique utilise un modèle géométrique : sous forme de "Squelette", (composé de cubes ou cylindres articulés), sous forme d'une grille de surfaces ou sous forme de représentations de densité de probabilité liées au corps de l'acteur [20], pour une identification facile et simple du corps de l'acteur en mouvement à partir des images 2D acquises.

Les images 2D représentent des données complexes à traiter à cause de multiples facteurs, tels que : textures, lumières, occultations, manque d'informations de profondeur, calibration des caméras.

Selon [15], l'auteur propose de différencier les systèmes MoCap optiques sans marqueurs en deux classes de familles, suivant la nature de l'information extraite:

1. Les systèmes MoCap optiques sans marqueurs à estimation de mouvement 2D : cette approche analyse le mouvement humain par l'extraction de ces caractéristiques (forme, type de mouvement, trajectoire et position des objets d'intérêt ...etc.) à partir des images 2D.

Généralement, un modèle 2D explicite de la forme à suivre est recalé sur le mouvement observé dans les images par l'utilisation de différentes caractéristiques définies par le type de modèle (contours, régions de l'image, simples points ...etc.). Ces approches sont utilisées pour des applications qui ne nécessitent pas une estimation précise (reconnaissance de geste, surveillance ...etc.) [15].

2. Les ystèmes MoCap optiques sans marqueurs à estimation de mouvement 3D : Ces approches donnent une meilleure compréhension du mouvement en estimant ces paramètres, par l'utilisation d'un modèle 3D du corps humain. Le squelette d'animation formé est habillé par différentes primitives géométrique simples, de façon à modéliser les propriétés physiques du corps humain [15].

Selon [21], les systèmes MoCap sont divisés en trois grandes familles, suivant les algorithmes qui tentent de calquer le modèle géométrique à l'image pour retrouver sa posture au cours du temps:

1. Les algorithmes basés sur les modèles de mouvement : Ils utilisent les connaissances sur le mouvement pour prédire les images futures.

2. Les algorithmes utilisant la vidéo inverse : Ils utilisent la vidéo inverse pour inverser la projection et retrouver une posture approchante du squelette à partir de l'image vidéo.

3. Les algorithmes utilisant la stéréoscopie : Avec plusieurs points de vue d'une même scène, ces algorithmes essayent de limiter les problèmes dûs aux occultations en analysant les images.

Un système MoCap optique sans marqueurs, utilise généralement des techniques à base de deux types de caméras : soit des caméras "Red Green Bleu (RGB)" simples ou des caméras 3D *Kinect de Microsoft* (figure 2.13).

Pour la technique qui utilise le premier type de caméras RGB, on exige une calibration initiale des caméras en détectant d'abord l'arrière plan de la scène. Ensuite, pour les nouvelles images capturées, l'arrière plan est soustrait afin d'isoler le sujet en question. De cette façon,

Figure 2.13: Caméra 3D Kinect de Microsoft et prise d'images de profondeur.

le squelette de l'acteur est extrait et on peut extraire ainsi les positions des articulations dans l'espace. Malheureusement, il n'y a aucune information de profondeur fournie par une caméra RGB, ce qui limite la précision de positions des articulations pour cette technique.

L'autre technique (à base de caméras 3D), utilise un modèle articulé (squelette) du corps humain pour une bonne précision de capture de mouvement. Une façon facile offerte par cette technique, pour l'extraction du squelette (figure 2.14) à partir des images de profondeur fournies pas ce type de caméras. La résolution de la caméra 3D utilisée, limite la précision de la capture pour cette technique [22].

Figure 2.14: Extraction des articulations.

Le processus de capture de mouvement est complètement effectuer à travers des logiciels, en supprimant toutes les limitations physiques ce qui impose des contraintes de calculs [23].

L'avantage majeur de ce type de système est son coût matériel minime, ce qui permet de l'utiliser pour des applications à faible budget. Dans un tel système, le sujet ne présente aucune contrainte physique, ce qui permet à l'acteur de bouger librement dans la scène; donc

peu de contraintes sont imposées pour l'environnement (possibilité de prendre des mesures à l'extérieur).

Ce type de système n'est pas parfait et il souffre de quelques problèmes, telle que l'analyse complexe des images par des techniques de traitement d'images et de vision par ordinateur et qui nécessite parfois l'intervention de l'utilisateur. Il souffre aussi du problème d'occultations et de manque de l'information de profondeur surtout pour les systèmes mono-caméra. De tels systèmes doivent avoir de la robustesse et un temps d'exécution raisonnable pour qu'ils puissent être appliqués dans le monde de l'animation ou de l'étude du mouvement humain [13].

2.3.3 Systèmes MoCap non optiques

Dans la pratique, nous rencontrons différents systèmes de capture de mouvement, qui sont basés sur d'autres grandeurs physiques que la lumière (pour remédier à leurs limitations) afin de construire un système MoCap plus précis et moins coûteux. Parmi les différentes grandeurs physiques, il y a des systèmes MoCap à base de capteurs (magnétique, mécanique, inertiel, acoustique et autres).

2.3.3.A Système MoCap magnétique

Un système MoCap magnétique est basé sur le principe de mesure et identification de positions et orientations en exploitant les propriétés d'un champ magnétique (généré par un émetteur électromagnétique), par des capteurs placés aux articulations du corps d'un acteur en mouvement.

Figure 2.15: Système MoCap magnétique [5].

Un système MoCap magnétique est composé d'un émetteur (figure 2.15) qui génère trois champs magnétiques (perpendiculaires les uns sur les autres) à chaque cycle de mesure, puisque l'émetteur est formé de trois bobines perpendiculaires traversées en séquence par un courant. Chaque capteur magnétique 3D peut mesurer la force de ces champs qui est proportionnelle à la distance entre le capteur et l'émetteur (de 3 à 15 mètres), ce qui permet de calculer les positions de chaque capteur [17].

Les capteurs sont placés sur le corps (figure 2.16) pour capter un champ électromagnétique de basse fréquence (100Hz) et sont connectés à travers des câbles (individuels dans la plupart des cas), transmettant un courant électrique filtré et amplifié à une unité de contrôle électronique. Cette unité envoie les données à un calculateur central pour calculer les coordonnées cartésiennes et les orientations à l'aide d'un logiciel approprié. Un autre logiciel utilise ces dernières données pour les convertir à une chaine hiérarchique avec une seule position et plusieurs orientations afin d'être utilisées pour créer une animation du mouvement [5].

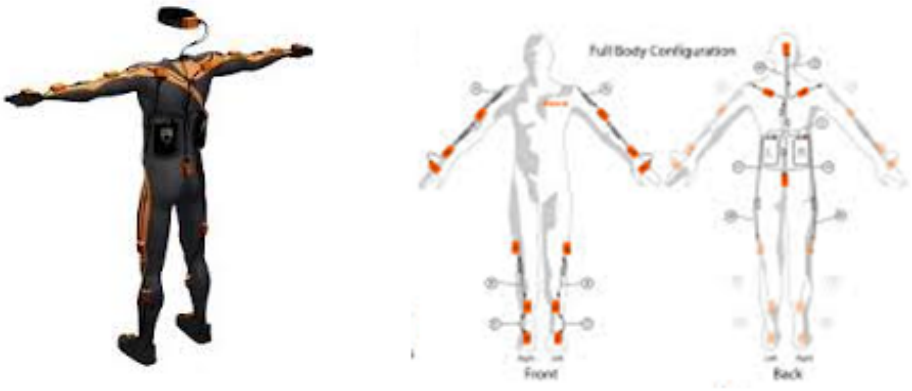


Figure 2.16: Positionnement des capteurs sur le corps (système XSENS) [6].

Les systèmes MoCap magnétiques, peuvent être divisés en deux types, selon le courant du champ électromagnétique (courant direct (DC) ou courant alternatif (AC) [5]).

Le système MoCap magnétique présente plusieurs avantages par rapport à d'autres systèmes MoCap. Le principal avantage, est qu'il n'a aucun problème vis à vis l'occultation des capteurs (puisque il s'agit de champs électromagnétiques qui traversent des corps non-métalliques) ce qui permet à plusieurs acteurs d'interagir simultanément. L'autre avantage, est que les positions et les orientations des capteurs sont disponibles sans traitement, ce qui minimise les calculs pour les prochaines étapes; ainsi, un tel système est dit "système à temps réel", permettant la visualisation des résultats en temps réel, ce qui permet aussi au réalisateur de vérifier et de corriger les erreurs sur scène. Pour le coût d'un tel système, on peut dire qu'il est moins cher qu'un système MoCap optique à marqueurs.

Les objets et les surfaces métalliques peuvent provoquer des irrégularités et des interférences dans les lignes des champs magnétiques par induction de courant dans le métal, ce qui génère un nouveau champ électromagnétique appelé "eddy currents" ou "courant de Foucault" et les données capturées seront ainsi erronées. Le problème d'interférence peut être provoqué par les câbles reliant les capteurs et ces mêmes câbles peuvent gêner le mouvement de l'acteur. Ce dernier problème est résolu par l'utilisation de liaison sans fils (figure 2.17) entre les capteurs et l'unité de contrôle mais un autre problème dit "latency" ou "latence" exprimé par la croissance du temps de calcul entre la collection des données et l'affichage des résultats du mouvement.

Figure 2.17: Simplification du mouvement par l'utilisation de liaison sans fils [6].

La fréquence d'acquisition d'un tel système est considérée comme insuffisante pour certains mouvements sportifs (mouvement rapide) à cause du bruit de mesure, ce qui la diminue jusqu'à 15Hz à cause du filtrage [13]. On trouve aussi une difficulté dans le changement de configuration des capteurs posés sur le corps de l'acteur.

2.3.3.B Système MoCap mécanique

L'un des premiers systèmes utilisés pour la MoCap a été un électro goniomètre (un instrument qui permet de mesurer les angles) [17]. Un système MoCap mécanique est un système dirigé par les mouvements du corps humain et est constitué d'un ensemble de tiges légères modélisant les parties rigides du corps (les os) et des potentiomètres placés entre les tiges modélisant les joints (articulations), constituant l'ensemble (tiges et potentiomètres) s'appelé "l'exosquelette" (figure 2.18).

Le potentiomètre placé à un joint est considéré comme un capteur qui mesure les rotations angulaires des tiges et ces rotations sont convertis en angles d'articulations (figure 2.19) utilisant un modèle cinématique [24].

Dans le potentiomètre il y a un curseur qui se déplace le long d'un élément de résistance et produit une lecture de potentiel de tension variable, en fonction du pourcentage de la résistance totale appliquée à la tension d'entrée. Les potentiomètres utilisés pour les capteurs de mouvement sont parfois appelés "capteurs angulaires" analogiques ou numériques [5].

Le coût modéré d'un système MoCap mécanique est un avantage par rapport aux systèmes

Figure 2.18: Exosquelette pour un système MoCap mécanique [7].

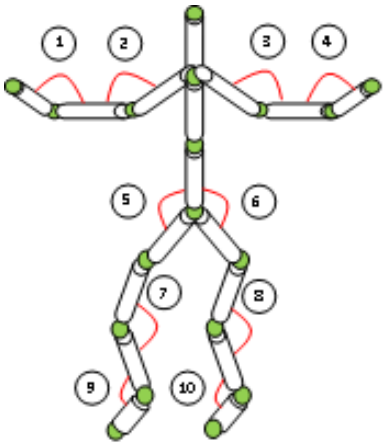


Figure 2.19: Modélisation par mesure d'angles.

optiques et magnétiques. La légèreté de l'armature mécanique (figure 2.20) fabriquée en métal ou en plastique, la rend portative ce qui permet à l'acteur de ce déplacer dans un volume de capture très large. A partir de la nature des capteurs, on peut dire qu'un tel système ne peut souffrir du problème d'occultations ce qui permet la capture rapide de mouvements pour un multiple d'acteurs, puisque les données sont disponibles et moins coûteuses.

Le plus grand inconvénient d'un système MoCap mécanique est son incapacité de mesurer les translations globales. Pour pallier à ce problème, on utilise généralement un capteur électromagnétique pour récupérer la translation globale du squelette [21]. Cet ajout engendre l'inconvénient de l'influence du métal sur le système électromagnétique. N'oublions pas la rigidité de l'armature

Figure 2.20: Armature pour un système MoCap mécanique [8].

qui gêne le mouvement de l'acteur surtout pour les mouvements rapides, ce qui est traduit par une fréquence d'acquisition faible pour un tel système. Aussi, les capteurs placés sur cette armature sont fixes et on ne peut changer la configuration de ces capteurs (figure 2.20).

2.3.3.C Système MoCap inertiel

Le système MoCap inertiel n'est pas un système de capture de mouvement du corps humain seulement, mais il a été initialement utilisé pour suivre l'orientation de véhicules aériens. Le principe de ce système est de mesurer l'orientation d'un segment du corps équipé d'une unité dite "Unité de Mesure Inertielle (UMI)" ou "Inertial Measurement Unit (IMU) en anglais", qui contient à la fois : un accéléromètre et un gyroscope. Pour d'autre système "Xsens MVN" de capture de mouvement inertiel, l'unité contienne : gyroscope 3D, accéléromètre 3D et magnétomètre 3D (figure 2.21, [25]). Ce qui lui permet de mesurer la position et l'orientation (en 3D) des segments du corps (en 6 degrés de liberté) dans un système de coordonnées globales [26]. Tous ces capteurs sont installés ensemble dans un petit boîtier pour simplifier sa fixation à un objet ou à une personne (figure 2.22) [20].

Le rôle de l'accéléromètre est de mesurer l'accélération par le déplacement d'une petite masse à l'intérieur du capteur due à la force du mouvement du segment et par contribution de la gravité. Ainsi, la position est obtenue par double intégration de la donnée d'accélération en fonction du temps.

Le rôle du gyroscope est de mesurer la vitesse angulaire du capteur, et avec une seule intégration de cette donnée en fonction du temps, on obtient l'orientation du segment sur le

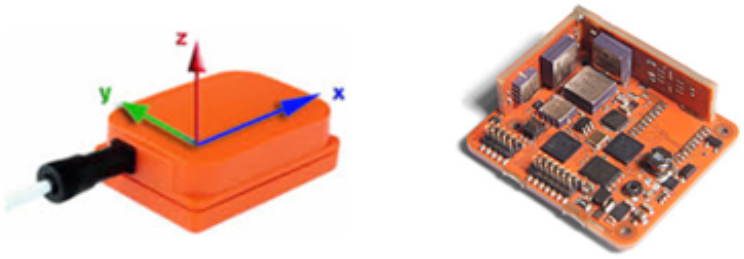


Figure 2.21: Unité de mesure inertielle.

quel est placé le capteur.

Le rôle du magnétomètre est d'assurer la stabilité dans le plan horizontal, en détectant la direction du champ magnétique terrestre comme une boussole [27]. La plupart des UMI utilisent les mesures du capteur de champ magnétique pour dériver la direction canonique "Nord" [20].

Figure 2.22: Placement des capteurs inertiels.

Le système MoCap inertiel présente des avantages par rapport à d'autres systèmes MoCap. Le premier avantage est qu'il est autonome, c'est à dire, qu'il n'est relié à aucune source externe (lumière Infra Red (IR), champs magnétique, ...etc.). Le second, est qu'il n'y a aucun facteur externe qui peut gêner sont fonctionnement (système sans magnétomètre). Le troisième, est qu'il est portatif puisque les capteurs en petit boîtier sont très légers. L'exécution du système en temps réel avec une mise à jour maximale de 120Hz comme fréquence d'acquisition, est un autre avantage. Cet avantage exprime aussi un large volume de capture même à l'extérieur.

Comme d'autres systèmes MoCap, ce système présente aussi quelques inconvénients. En premier, nous pouvons citer le problème qu'un tel système ne peut mesurer la position et

l'orientation directement mais par calcul mathématique des grandeurs mesurées, ce qui provoque une accumulation d'erreurs le long du temps à cause de la dérive d'intégration, mais ils sont stables si les mesures sont prises à des courts intervalles de temps. Un autre inconvénient vient avec l'ajout du magnétomètre, est que le champ magnétique mesuré diffère d'un capteur à un autre.

2.3.3.D Système MoCap acoustique

Un système de capture de mouvement acoustique ou sonore, est un système basé sur l'utilisation des ondes (ou des impulsions) ultrasoniques. La génération de ces ondes, s'effectue par des petits émetteurs placés sur les articulations principales du corps de l'acteur. Tandis, que la réception de ces ondes est réalisée par seulement trois récepteurs (microphones) placés dans la zone de captation (figure 2.23). Les récepteurs sont connectés à une unité de contrôle électronique. La distance entre les émetteurs et les récepteurs est limitée par la vitesse de propagation de l'onde acoustique dans l'air qui varie avec la pression et la température de l'air.

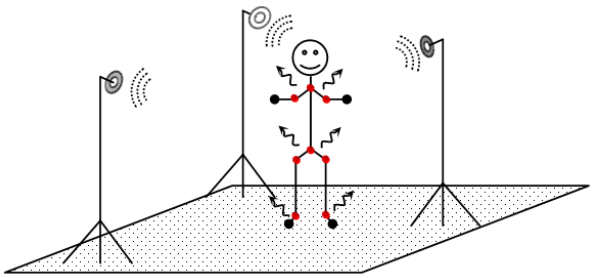


Figure 2.23: Système MoCap acoustique.

Le principe d'un tel système, est de calculer la position de chaque émetteur en utilisant la longueur d'onde du signal acoustique généré et sa vitesse de propagation. Selon [28], il y a deux types de systèmes:

- Les systèmes à impulsions mesurent la différence en temps (temps de vol) entre l'émission du signal à partir de l'émetteur et sa réception par le microphone du capteur.
- Les systèmes à ondes continues sont basés sur la cohérence de phase et mesurent la différence de phase entre le signal à l'extrémité d'émetteur et l'extrémité de récepteur.

Selon [29], ces systèmes peuvent être basés uniquement sur des signaux à ultrasons ou par une combinaison (hybridation) avec des ondes Radio Fréquence (RFq) pour améliorer le système. Pour le dernier cas, le récepteur est monté dans un équipement portatif et détermine sa propre position.
Malgré que ce type de système présente l'avantage d'être à bon marché, il possède par contre plusieurs inconvénients. Parmi les inconvénients des systèmes à ultrasons, on cite :

- Le faible débit binaire qui engendre une faible fréquence d'acquisition
- La réduction du niveau du signal à cause du bruit de fond
- La réflexion des ondes par les murs et les objets à l'intérieur de la zone de travail
- Les problèmes causés par l'effet "multi trajets" du même signal (écho)
- La présence de gaz et variations de la température, humidité et pression d'air
- La latence de mesure d'une position causée par le déplacement lent du son (de l'émetteur jusqu'au récepteur)
- L'alignement presque impossible entre l'émetteur et le récepteur pour des meilleures mesures.

2.3.3.E Comparaison entre les systèmes MoCap

Le tableau 2.1 donne une comparaison entre les différents systèmes de capture de mouvement vus dans les paragraphes précédents.

Tableau 2.1: Comparaison entre les différents systèmes de capture de mouvement

Système MoCap	Optique	Magnétique	Mécanique	Inertiel	Acoustique
Temps réel	Oui	Oui	Oui	Oui	Oui
Coût	Très cher	Moins cher	Raisonnable	Moins Cher	Moins cher
Occultation	Oui	Non	Non	Non	Oui
Prés calcul	Oui	Non	Non	Non	Oui
Multiple acteurs	Oui	Oui	Oui	Oui	Oui
Sensibilité	Lumière calibrage	Métal	Rigidité	Accumulation d'erreurs	Pression
Fréquence d'acquisition	Haute (500Hz)	100Hz	Faible	Haute	Basse
Volume de capture	Variable	Limité (3-15m)	Variable	Variable	Très limité
Changement de config des capts	Facile	Difficile	Difficile	Facile	Facile
Contrainte câbles	Non	Oui	Non	Non	Non
Précision	Grande	Grande	Faible	Faible	Faible
Nombre de capteurs	Illimité	Limité	Limité	Limité	Limité
Confort	Oui	Oui	Non	Oui	Oui

2.4 Caméra

Pour un système de capture de mouvement (MoCap), la caméra présente l'élément essentiel pour l'acquisition des données (images des marqueurs) issues du déplacement d'un objet ou d'un humain dans une scène de surface bien déterminée. Les images capturées par la caméra présentent une projection du monde (système de coordonnées 3D) sur une image (système de coordonnées 2D). Pour que le système soit opérationnel et se rapproche au comportement physique (projection perspective), il faut définir un modèle géométrique pour la caméra basé sur un ensemble de paramètres décrivant le comportement physique du capteur. L'extraction de ces paramètres est dite "phase de calibration".

2.4.1 Modélisation de la caméra

Plusieurs modèles décrivent les caractéristiques de la caméra d'une façon plus ou moins proche à la manière que notre œil perçoit visuellement le monde qui nous entoure. Le modèle le plus utilisé par la communauté de la vision par ordinateur et qui représente une projection perspective est dit "modèle sténopé" ou encore "modèle de trou d'épingle" (en anglais, pinhole model) (voir le dispositif optique responsable de la formation de l'image (figure 2.24) [9]). Ce modèle présente un compromis entre l'approximation du phénomène de la vision physique et la simplicité des équations de la modélisation. Ce modèle est loin d'être novateur puisque le phénomène était déjà décrit par Aristote puis Léonardo de Vinci et enfin Cardan [30].

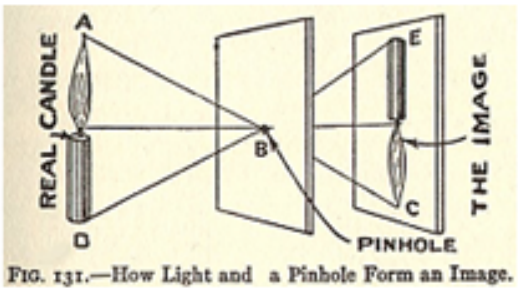


Figure 2.24: Principe d'une caméra sténopée illustré en 1925 dans "The Boy Scientist" [9].

Le modèle sténopé (sur un plan géométrique), peut être décomposé en un ensemble de points 3D "M" situés dans une scène aux coordonnées (M_{X_m},M_{Y_m},M_{Z_m}) dans le repère monde (O_m)

et sa projection sur un plan image R est appelée "rétine", en un point 2D "m" de coordonnées (x, y) exprimées en pixels. Selon la figure 2.25, l'image "m" est l'intersection de la droite CM avec le plan image. Cette droite est appelée "droite de projection" ou "ligne de vue". Tandis que la droite qui passe par le point focal C (appelé aussi "centre optique") et le centre du plan image (point c ou "point principal"), est appelée "axe optique".

Si le point focal est placé devant le plan image, l'image obtenue est une projection inverse de la scène (figure 2.24). Dans le cas contraire, l'image obtenue est une projection de la scène (figure 2.25) [31]. On peut définir la distance focale f (en mm), comme la distance entre le point focal C de coordonnées (x_C,y_C,z_C) et le point principal c de coordonnées (u_0,v_0) exprimé en pixels.

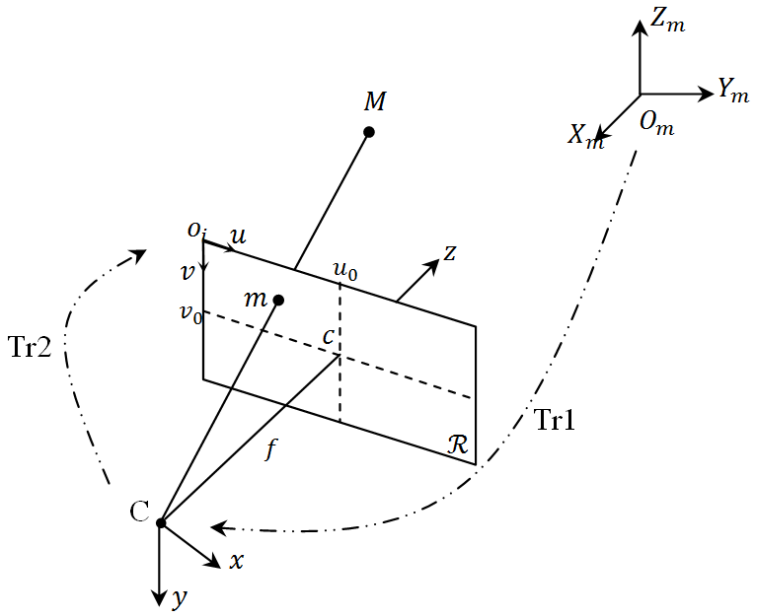


Figure 2.25: Modèle sténopé d'une caméra.

2.4.2 Paramètres extrinsèques

Les paramètres extrinsèques expriment une relation d'un mouvement rigide entre le repère de caméra (C,x_C,y_C,z_C) et le repère monde (O_m,X_m,Y_m,Z_m). Cette relation est définie par la position du repère caméra, donnée par le vecteur de translation t et par l'orientation du même repère par la matrice de rotation R lors de la prise de vue.

La position du point focal C par rapport au repère monde O_m est donnée par le vecteur de translation, comme suit:

$$t = \begin{vmatrix} t_x \\ t_y \\ t_z \end{vmatrix} \tag{2.1}$$

La matrice d'orientation R qui exprime l'orientation des axes de la caméra (x_C,y_C,z_C) par rapport au repère monde O_m peut être calculée par le produit matriciel de trois matrices de rotation, comme suit:

$$R = \begin{vmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{vmatrix} \tag{2.2}$$

Les angles d'Euler α, β et γ sont associés aux axes x_C,y_C et z_C respectivement.

On peut écrire la notation de cette matrice d'une manière plus simple à l'aide de ces trois vecteurs d'orientation r_i constitués de trois composantes r_{ij} pour chacun, comme suit:

$$R = \begin{vmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{vmatrix} = \begin{vmatrix} r_1 \\ r_2 \\ r_3 \end{vmatrix} \tag{2.3}$$

Les vecteurs r_i sont unitaires orthogonaux et orthonormaux, c'est à dire:

$$r_i.r'_j = \begin{cases} 0 & Si \quad i \neq j \\ 1 & Si \quad i = j \end{cases}$$

Enfin, les deux repères (C et O_M) sont reliés par l'équation 2.4 suivante:

$$\begin{vmatrix} x_C \\ y_C \\ z_C \end{vmatrix} = R. \begin{vmatrix} X_m \\ Y_m \\ Z_m \end{vmatrix} + t \tag{2.4}$$

A partir des coordonnées (M_{X_m},M_{Y_m},M_{Z_m}) du point M dans le repère monde, on peut calculer ces coordonnées (M_{x_C} ,M_{y_C},M_{z_C}) dans le repère caméra à l'aide de l'équation (2.5), comme suit:

$$\begin{vmatrix} M_{x_C} \\ M_{y_C} \\ M_{z_C} \end{vmatrix} = R. \begin{vmatrix} M_{X_m} \\ M_{Y_m} \\ M_{Z_m} \end{vmatrix} + t \tag{2.5}$$

On tenant compte des coordonnées homogènes, la relation (2.5) peut être exprimée par une seule matrice A, comme suit :

$$\begin{vmatrix} M_C \\ 1 \end{vmatrix} = A. \begin{vmatrix} M_m \\ 1 \end{vmatrix} + t \tag{2.6}$$

Avec:

$$A = \begin{vmatrix} R_{3\times3} & t_{3\times1} \\ 0_{1\times3} & 1 \end{vmatrix}$$

La transformation rigide (Tr1) est représentée par la matrice A, qui présente à son tour les trois composantes de rotation et les trois composantes de translation. Elle exprime donc les six paramètres extrinsèques (α, β, γ, t_x, t_y et t_z).

2.4.3 Paramètres intrinsèques

La deuxième transformation est une transformation affine permettant de passer du repère caméra (C,x_C,y_C,z_C) au repère image (o_i,u,v) comme un changement d'échelle et une translation. A l'aide des paramètres intrinsèques de la caméra, nous pouvons effectuer une relation entre un point M_C (M_{x_C}, M_{y_C}, M_{z_C}) en 3D et son projeté $m(x, y)$ en 2D sur le plan image.

A l'aide des relations entre triangles similaires (voir figure 2.25), nous pouvons calculer les coordonnées (x, y) du point image m, comme suit:

$$x = f\frac{M_{x_C}}{M_{z_C}} \tag{2.7}$$

$$y = f\frac{M_{y_C}}{M_{z_C}} \tag{2.8}$$

En coordonnées homogènes, les équations (2.7) et (2.8) peuvent être écrites sous forme matricielle, comme suit:

$$\begin{vmatrix} x \\ y \\ 1 \end{vmatrix} \sim \begin{vmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} M_{x_C} \\ M_{y_C} \\ M_{z_C} \end{vmatrix} = \begin{vmatrix} fM_{x_C} \\ fM_{y_C} \\ M_{z_C} \end{vmatrix} \tag{2.9}$$

L'égalité vectorielle n'est définie qu'à un facteur scalaire près ($\sim$). Ceci est un vecteur de coordonnées homogènes.

$$\begin{vmatrix} f\frac{M_{x_C}}{M_{z_C}} \\ f\frac{M_{y_C}}{M_{z_C}} \\ 1 \end{vmatrix} \sim \begin{vmatrix} fM_{x_C} \\ fM_{y_C} \\ M_{z_C} \end{vmatrix} \tag{2.10}$$

La matrice de projection homogène $diag(f,f,1)[I \mid 0]$ permet le passage entre le repère caméra (C,x_C,y_C,z_C) et le repère image (o_i,u,v), c'est le changement d'échelle.

Pour la translation, il s'agit du passage aux coordonnées pixeliques. Les équations (2.7) et (2.8) prennent comme origine le point principal c de coordonnées (u_0,v_0), mais en réalité l'origine des coordonnées image (en pixels) est le point (o_i,u,v) situé au coin supérieur gauche du plan image R. Il faut prendre aussi en considération les deux facteurs k_u et k_v, dits facteurs d'échelles horizontal et vertical respectivement, exprimés en pixels/mm. Ces facteurs sont pris en compte pour modéliser les imperfections des pixels qui ne sont pas orthogonaux et ne possèdent pas la même échelle sur les deux axes qui constituent le capteur (Charged Coupled Device (CCD) et Complementary Meatl Oxide Semiconductor (CMOS)) [9].

Alors, les nouveaux coordonnées sont données par:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} k_u & 0 & 0 \\ 0 & k_v & 0 \\ 0 & 0 & 0 \end{vmatrix} \begin{vmatrix} x \\ y \\ 1 \end{vmatrix} + \begin{vmatrix} u_0 \\ v_0 \\ 1 \end{vmatrix} \tag{2.11}$$

Alors:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x \\ y \\ 1 \end{vmatrix} = K. \begin{vmatrix} x \\ y \\ 1 \end{vmatrix} \tag{2.12}$$

Avec :

$$K = \begin{vmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 0 \end{vmatrix}$$

est une matrice de transformation linéaire.

Si en remplace par l'équation (2.9), l'équation (2.12) devient:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} M_{x_C} \\ M_{y_C} \\ M_{z_C} \\ 1 \end{vmatrix} = \begin{vmatrix} fk_u & 0 & u_0 & 0 \\ 0 & fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} M_{x_C} \\ M_{y_C} \\ M_{z_C} \\ 1 \end{vmatrix}$$

D'où:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} M_{x_C} \\ M_{y_C} \\ M_{z_C} \\ 1 \end{vmatrix} = I. \begin{vmatrix} M_{x_C} \\ M_{y_C} \\ M_{z_C} \\ 1 \end{vmatrix} \tag{2.13}$$

Tel que:

$\alpha_u = fk_u$ et $\alpha_v = fk_v$ désignent la focale de la caméra en nombre de pixels suivant les directions x et y respectivement.

La matrice I est une application linéaire entre le repère caméra et le plan image. Cette matrice exprime les paramètres intrinsèques de la caméra (α_u, α_v, u_0 et v_0).

2.4.4 Modèle sténopé complet (sans distorsion)

Finalement, nous pouvons décrire la relation de projection d'un point "M" dans le repère monde et sa projection 2D dans le plan image par le produit de deux matrices. La première matrice "I" qui modélise le capteur par ces paramètres intrinsèques et la seconde matrice "A"

qui exprime la pose de la caméra par les paramètres extrinsèques.

Remplaçons l'équation (2.6) dans (2.13), on obtient:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} R_{3\times3} & t_{3\times1} \\ 0_{1\times3} & 1 \end{vmatrix} \begin{vmatrix} M_{X_m} \\ M_{Y_m} \\ M_{Z_m} \\ 1 \end{vmatrix}$$

On remplace $R_{3\times3}$ et $t_{3\times1}$ par leurs équivalent (2.3) et (2.1) respectivement:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} M_{X_m} \\ M_{Y_m} \\ M_{Z_m} \\ 1 \end{vmatrix} \tag{2.14}$$

A partir de l'équation (2.14), on obtient donc les formules suivantes:

$$\begin{aligned} u &= u_0 + \alpha_u . \frac{r_{11}M_{X_m} + r_{12}M_{Y_m} + r_{13}M_{Z_m} + t_x}{r_{31}M_{X_m} + r_{32}M_{Y_m} + r_{33}M_{Z_m} + t_z} \\ v &= v_0 + \alpha_v . \frac{r_{21}.M_{X_m} + r_{22}.M_{Y_m} + r_{23}.M_{Z_m} + t_y}{r_{31}.M_{X_m} + r_{32}.M_{Y_m} + r_{33}.M_{Z_m} + t_z} \end{aligned} \tag{2.15}$$

Ou tout simplement:

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = I.A \begin{vmatrix} M_{X_m} \\ M_{Y_m} \\ M_{Z_m} \\ 1 \end{vmatrix} = L \begin{vmatrix} M_{X_m} \\ M_{Y_m} \\ M_{Z_m} \\ 1 \end{vmatrix}$$

Avec:

$$L = \begin{vmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} r_1 & t_x \\ r_2 & t_y \\ r_3 & t_z \\ 0 & 1 \end{vmatrix} = \begin{vmatrix} \alpha_u.r_1 + u_0.r_3 & \alpha_u t_x + u_0 t_z \\ \alpha_v r_2 + v_0 r_3 & \alpha_v t_y + v_0 t_z \\ r_3 & t_z \end{vmatrix} \tag{2.16}$$

Nous pouvons aussi écrire la matrice L dite "matrice de projection perspective", sous la forme

suivante:

$$L = \begin{vmatrix} l_{11} & l_{12} & l_{13} & l_{14} \\ l_{21} & l_{22} & l_{23} & l_{24} \\ l_{31} & l_{32} & l_{33} & l_{34} \end{vmatrix} = \begin{vmatrix} l_1 & l_{14} \\ l_{21} & l_{24} \\ l_{31} & l_{34} \end{vmatrix} \tag{2.17}$$

Avec l_i représentatif d'un vecteur ligne à trois composantes:

$$l_i = \begin{vmatrix} l_{i1} & l_{i2} & l_{i3} \end{vmatrix}$$

A partir des formules de la matrice L, nous pouvons écrire un ensemble d'équations (2.18) qui permettent de calculer les paramètres intrinsèques et extrinsèques en fonction des coefficients de L, comme suit:

$$\begin{cases} r_3 = l_3 \\ u_0 = l_1.l_3' \\ v_0 = l_2.l_3' \\ \alpha_u = \sqrt{l_1.l_1' - u_0^2} \\ \alpha_v = \sqrt{l_2.l_2' - v_0^2} \\ r_1 = \frac{l_1 - u_0.l_3}{\alpha_u} \\ r_2 = \frac{l_2 - v_0.l_3}{\alpha_v} \\ t_z = l_{34} \\ t_x = \frac{l_{14} - u_0.l_{34}}{\alpha_u} \\ t_y = \frac{l_{24} - v_0.l_{34}}{\alpha_v} \end{cases} \tag{2.18}$$

Ces formules représentent les trois paramètres extrinsèques de rotation autour des axes (r_1,r_2,r_3), ceux de translation (t_x,t_y,t_z), ainsi que les quatre paramètres intrinsèques (α_u,α_v,u_0,v_0).

Nous pouvons procéder à deux étapes pour trouver les paramètres du modèle. La première étape consiste à estimer les coefficients de la matrice L et la deuxième est d'extraire les paramètres à l'aide de ces coefficients donnés par les formules (2.18) [31].

Le modèle sténopé décrit jusqu'à maintenant est un modèle projectif simple de la caméra. Pour des objets de taille faible par rapport à leurs distances au centre de projection C, un modèle dit "modèle linéaire affine" décrit la caméra comme une projection parallèle de la scène sur le plan image [30].

2.4.5 Modèle sténopé complet (avec distorsion)

Dans le cas d'un modèle sténopé ou affine, le ratio entre l'épaisseur de la lentille et le rayon de la courbure des ses faces est faible et ce modèle ne prend pas en considération les distorsions non linéaires puisqu'elles sont considérées comme des distorsions rectilignes où les lentilles sont dites "minces" [9]. Pour les caméras réelles, trois types de distorsions de différentes natures se présentent [30]:

- La distorsion radiale causée par les défauts de courbures des lentilles de la caméra
- La distorsion de décentrage causée par le mauvais alignement des centres optiques des lentilles
- La distorsion prismatique causée par l'inclinaison des lentilles les unes par rapport aux autres.

Ces distorsions doivent êtres prisent en compte pour corriger l'erreur de positionnement par rapport au modèle parfait. Dans [30], une étude sur ces différentes distorsions est bien détaillée.

2.5 Conclusion

Dans les années 1860, des premiers travaux sont effectués dans le domaine MoCap pour analyser des mouvements cinématographiques afin d'étudier la locomotion animale et humaine. En 1973, une méthode basée sur le mouvement biologique consiste à attacher des marqueurs réflecteurs afin d'analyser les trajectoires des articulations a été introduite dans un tel système. Par la suite, ce système (MoCap) est développé pour toucher plusieurs disciplines telles que: le sport, la médecine, la biomécanique ...etc. En général, les systèmes sont classés comme des systèmes optiques, non optiques ou hybrides. L'objectif de ce chapitre a été d'expliquer les différents systèmes MoCap et de présenter les avantages ainsi que les limites de chacun d'eux.

Dans notre travail, nous nous sommes intéressés aux systèmes MoCap optique à base de marqueurs passifs. Ce système se base sur l'utilisation de la caméra comme moyen d'acquisition des données. Pour cette raison, une partie de ce chapitre a été consacrée à l'étude de la modélisation de la caméra. Cette étude nous a permis de mieux comprendre les paramètres

intrinsèques modélisant le capteur et les paramètres extrinsèques modélisant la position et l'orientation de la caméra.

Le chapitre suivant sera consacré à l'étude de deux méthodes d'optimisation : L'optimisation par essaim de particules (OEP) et la recherche fractale stochastique (RFS).

3

Métaheuristiques d'Optimisation par OEP et par RFS: Etat de l'art

3.1 Introduction

Actuellement, plusieurs domaines sont en perpétuel changement, surtout dans les domaines de la science et de l'industrie (ingénieurs, décideurs, ...etc.), soit pour la conception d'un système mécanique, informatique, ou électronique. Ce changement est dû aux améliorations technologiques qui provoquent à leur tour l'augmentation de la quantité des données ainsi que leurs types. Les gens sont confrontés à divers problèmes dont la complexité augmente de jour en jour. Par conséquent, il faudrait que les méthodes de traitement utilisées doivent suivre cette évolution. Au début, la tache a été d'accumuler le maximum de données possibles pour une exploitation directe, mais aujourd'hui, il faudrait utiliser des méthodes complexes pour l'extraction de l'information utile.

Malheureusement, ces méthodes exhaustives sont trop coûteuses en termes de temps et espace mémoire. Aussi, pour un problème d'optimisation traditionnel simple, il peut être résolu analytiquement avec des méthodes déterministes; par contre, il devient difficile (voir impossible) de résoudre un problème d'optimisation global qui n'est pas différentiable, non-linéaire, implicite, ou possède plusieurs solutions optimales locales par les méthodes déterministes [32].

Il est impossible d'atteindre les solutions optimales globales exactes et des algorithmes sont nécessaires pour les adapter aux environnements si ces derniers sont dynamiques. Un autre type de problèmes d'optimisation globale ont un grand nombre d'optima locaux, en particulier dans un espace de recherche à grande dimension. Cette propriété devient impossible vu que les méthodes déterministes énumèrent tous les optima locaux dans un délai non acceptable en raison de la grande complexité de l'espace de recherche. Par conséquent, les méthodes non-déterministes sont nécessaires pour obtenir de meilleures solutions possibles qui peuvent avoir une qualité moins inférieure à l'optimum global dans un délai envisageable [32].

Heureusement, il y a une alternative à ces inconvénients qui consiste à utiliser des méthodes dont la recherche de la solution du problème est plus rapide et moins encombrante. L'une de ces techniques formalise le problème sous forme d'un problème d'optimisation. Les méthodes d'optimisation dites "approchées", peuvent trouver une solution satisfaisante pour un temps acceptable là ou les méthodes exhaustives sont trop lentes. Alors, de nombreux problèmes difficiles, peuvent être formulés comme des problèmes d'optimisation [10].

3.2 Théorie de la complexité

La théorie de la complexité s'intéresse à l'étude formelle de la difficulté des problèmes en informatique [33]. La question qui se pose est si ces problèmes peuvent être résolus d'une façon efficace ou non suivant l'estimation théorique du temps de calcul et l'espace mémoire disponible. En réalité, il y a une multiplicité d'algorithmes pour résoudre un problème donné. Alors, comment juger qu'un algorithme est plus performant qu'un autre? Pour répondre à cette question, il faudrait effectuer une comparaison entre ces algorithmes selon le temps d'éxécution nécessaire pour chacun. Ce temps dépend des caractéristiques de la machine, le langage de programmation, le système d'exploitation et l'expérience du programmeur.

Selon [33], pour définir plus rigoureusement ce qu'est l'efficacité d'un algorithme, on doit donc considérer une approche complètement indépendante de toutes ces considérations. L'approche de [34] a alors été de calculer une mesure de la complexité d'un algorithme, afin d'obtenir un ordre de grandeur du nombre d'opérations élémentaires nécessaires pour que l'algorithme fournisse la solution du problème à l'utilisateur.

Les problèmes peuvent aussi se distinguer entre des problèmes de décision ou d'optimisation. Pour le problème de décision, se pose une question dont la réponse sera "oui" ou "non". Tandis que, pour le problème d'optimisation, son objectif est de chercher une solution de bonne qualité selon un ou plusieurs critères et objectifs. La complexité d'un problème d'optimisation est liée à la complexité d'un problème de décision qu'on peut associer au premier, dont le but est de déterminer s'il existe une solution pour laquelle la fonction objectif soit supérieure (ou inférieure) ou égale à une valeur donnée [33].

3.2.1 Complexité algorithmique

La complexité algorithmique d'un problème quelconque est définie comme une estimation du nombre d'instructions nécessaires (affectation, comparaison, opération algébrique, lecture et écriture, ...etc.) à exécuter pour résoudre un problème. Cette complexité algorithmique exprime l'efficacité (le temps d'exécution) des algorithmes de résolution pour différents problèmes. La notion en O() (notation en grand $'O'$) est utilisée généralement pour exprimer la complexité

d'un algorithme donné (par exemple $O(n^2)$, $O(logn)$,..). Dans la littérature, plusieurs classes de problèmes ont été définies en fonction de leur résolution (facile ou difficile) [35] [33].

Définition : Algorithme à temps polynomial

Un algorithme est dit "Algorithme à temps polynomial" si sa complexité est $O(p(n))$, où $p(n)$ est une fonction polynomiale de n.

Une fonction polynomiale de degré k peut être définie comme suit:

$$p(n) = a_k \cdot n^k + \cdots + a_j \cdot n^j + \cdots + a_1 \cdot n + a_0$$

Avec: $a_k > 0$ et $a_j \geq 0$, $1 \leq j \leq k-1$. L'algorithme correspond à une complexité polynomiale de $O(n^k)$ [10].

3.2.2 Classification de la complexité

En fonction de la difficulté intrinsèque d'un problème, la classification de la complexité aide au choix de l'algorithme de résolution de ce problème.

La théorie de complexité classe les problèmes en fonction de leurs complexités en trois classes courantes : P, NP et NP-complet :

a) Classe P (de Polynomial time) : Un problème est classé comme problème polynomial, s'il est résoluble par un algorithme de complexité polynomiale. Un tel algorithme permet de trouver une solution optimale pour toutes ces instances en un temps polynomial par rapport à la taille de l'instance. Cette classe est la classe des problèmes dits faciles.

b) Classe NP (de No-deterministic Polynomial time) : C'est la classe des problèmes (dits difficiles) pour lesquels la solution (exponentielle ou pire qu'exponentielle) peut être trouvée par un algorithme non-déterministe en un temps polynomial par rapport à la taille de l'instance. Pendant l'exécution de l'algorithme, il y aura une autorisation à des choix non déterministes.

c) Classe NP-complet : C'est la classe des problèmes plus difficiles que les NP. On dit qu'un problème donné appartient à NP-complet s'il est dans NP et si tout autre problème dans NP peut s'y réduire. Pour prouver qu'un nouveau problème appartient à NP-complet, on montre d'abord qu'il est dans NP, ensuite le réduire en un problème connu dans NP-complet. Beaucoup de problèmes importants et fréquents sont NP-Complets et sont cités dans [33].

Une autre classe donnée par [36] (dont la complexité est aussi difficile que la classe NP-complet) est dite "NP-difficile". Elle englobe les problèmes de décision et d'optimisation en même temps. Si un problème de décision associé à un problème d'optimisation P est un NP- Complet alors P est un NP-Difficile. Par conséquent, afin de prouver qu'un problème d'optimisation est NP-Difficile, il suffit de montrer que le problème de décision associé à P est NP-Complet. Il est à noter que jusqu'à maintenant, aucun algorithme polynomial n'est connu pour résoudre ce type de problèmes (i.e. NP- Difficiles).

Plus de détails pour la théorie de complexité sont données dans [10].

3.3 Optimisation combinatoire

3.3.1 Définition

L'optimisation existe depuis toujours, et elle est l'une des plus anciennes questions scientifiques. Jusqu'aux technologies de pointe dans notre vie quotidienne, nous pouvons toujours aborder un problème, en tentant d'obtenir un profit maximum avec un coût minimal [32].

L'optimisation (ou Optimisation Combinatoire (OC)) occupe une place très importante en recherche opérationnelle et en informatique et elle regroupe une grande classe de problèmes d'optimisation dans de nombreux domaines tels que la gestion, l'ingénierie, la production, les télécommunications, les transports, l'énergie et les sciences sociales. Ces problèmes peuvent être modélisés comme des applications telles que : le problème de voyageur de commerce, l'ordonnancement de tâches, ou le problème de la coloration de graphes ...etc. Ces applications sont décrites en détails dans [33].

Aussi, dans [33] une définition générale de l'optimisation est donnée par [37] [33] comme: " L'optimisation c'est l'art de comprendre un problème réel, de pouvoir le transformer en un modèle mathématique que l'on peut étudier afin d'en extraire les propriétés structurelles et de caractériser les solutions du problème. Enfin, c'est l'art d'exploiter cette caractérisation afin de déterminer des algorithmes qui les calculent mais aussi de mettre en évidence les limites sur l'efficience et l'efficacité de ces algorithmes".

Ainsi, un problème d'optimisation nécessite l'étude de quatre points principaux afin de le

résoudre (figure 3.1):

1. L'analyse du problème (caractérisation, modélisation, représentation informatique et codage)
2. L'expression de l'objectif à évaluer
3. Le choix de la méthode d'optimisation
4. Implémentation de la solution.

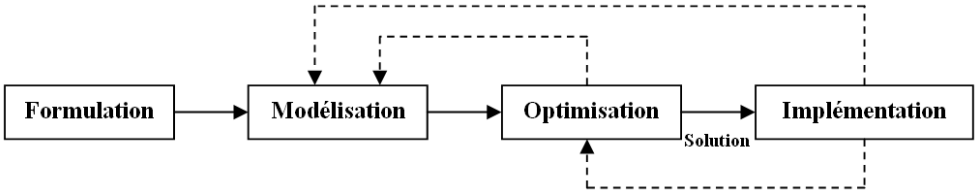


Figure 3.1: Processus classique de décision [10].

Une des classes des problèmes les plus populaires sont les problèmes d'optimisation combinatoire. Cette classe de problèmes est caractérisée par des variables de décision discrètes et un espace de recherche finie. Cependant, la fonction d'objectif et les contraintes sont de natures différentes et peuvent prendre toutes formes (non linéaire, non analytique, boîte noire, ...etc.) [10].

La résolution d'un problème d'optimisation combinatoire se ramène à la mise en examen d'un ensemble S fini de combinaisons (solutions $x = (x_1, x_2, \ldots, x_n)$) dites réalisables (formant ainsi l'espace de recherche ou de décision D), et une fonction de coût ou d'objectif F qui associe à chaque combinaison $x \in S$ le nombre réel (ou entier) $F(x)$ i.e., $F : S \rightarrow R$. Alors, résoudre un problème d'optimisation revient à trouver une combinaison réalisable $x' \in S$ de coût minimal (ou maximal) de la fonction d'objectif dans D.

Un problème d'optimisation est défini comme suit :

- Le vecteur à éléments réels ou entiers de dimension n : $x = (x_1, x_2, \ldots, x_n)$
- Le domaine des éléments $D = (D_1, D_2, \ldots, D_n)$
- La (ou les) fonction(s) d'objectif(s) à minimiser ou à maximiser : $F_j(x)$ avec $j = 1, 2, \ldots, M$
- Les contraintes du problème, soit d'égalité comme $\phi_g(x) = 0$ avec $g = 1, 2, \ldots, G$ ou d'inégalité comme $\psi_k(x) < 0$ avec $k = 1, 2, \ldots, K$ qui peuvent être linéaires ou non

linéaires.

Une solution x' qui minimise la fonction d'objectif F tout en respectant les contraintes ϕ et ψ est appelée solution optimale ou un optimum global si: $F(x') \leq F(x), \forall x \in S$. Il est à noter que pour les problèmes de maximisation, il suffit de multiplier la fonction d'objectif par (-1).

3.3.2 Classification des problèmes d'optimisation

Il y a plusieurs critères utilisés pour la classification de différents problèmes d'optimisation combinatoire:

1. Selon le nombre d'objectifs :
   (a) Un seul objectif $M = 1$
   (b) Plusieurs objectifs $M \geq 2$
2. Selon les contraintes :
   (a) Avec contraintes d'égalité : $G \geq 1$ et $K = 0$
   (b) Avec contraintes d'inégalité : $G = 0$ et $K \geq 1$
   (c) Avec contraintes d'égalité et d'inégalité : $G \geq 1$ et $K \geq 1$
   (d) Sans contraintes
   (e) Avec contraintes dynamiques
3. Selon la linéarité :
   (a) Linéaire (F, ϕ et ψ sont linéaires)
   (b) Non linéaire
4. Selon la fonction d'objectif :

(a) Uni-modal (un seul optimum local ≡ optimum global)

(b) Multi-modal (plusieurs optimum)

5. Selon le type de la variable de décision :

(a) Discret ou combinatoire

(b) Continue

6. Selon la variable de décision et la fonction d'objectif :

(a) Déterministe

(b) Stochastique

La problématique pour résoudre un problème d'optimisation est de mettre en œuvre un algorithme efficace en temps de calcul (raisonnable) et en espace mémoire (minimal).

3.3.3 Méthodes d'optimisation

Plusieurs méthodes de résolution des problèmes d'optimisation combinatoires sont mentionnées dans la littérature, qui les divise en trois grandes catégories de nature différente (voir figure 3.2):

- Méthodes exactes
- Méthodes approchées (ou heuristiques)
- Méthodes hybrides.

3.3.3.A Méthodes exactes

Les méthodes d'optimisation exactes sont des méthodes déterministes, qui peuvent résoudre des problèmes d'optimisation et trouver une solution optimale dans un temps acceptable si le problème est de taille raisonnable. Les problématiques auxquelles ces méthodes doivent se

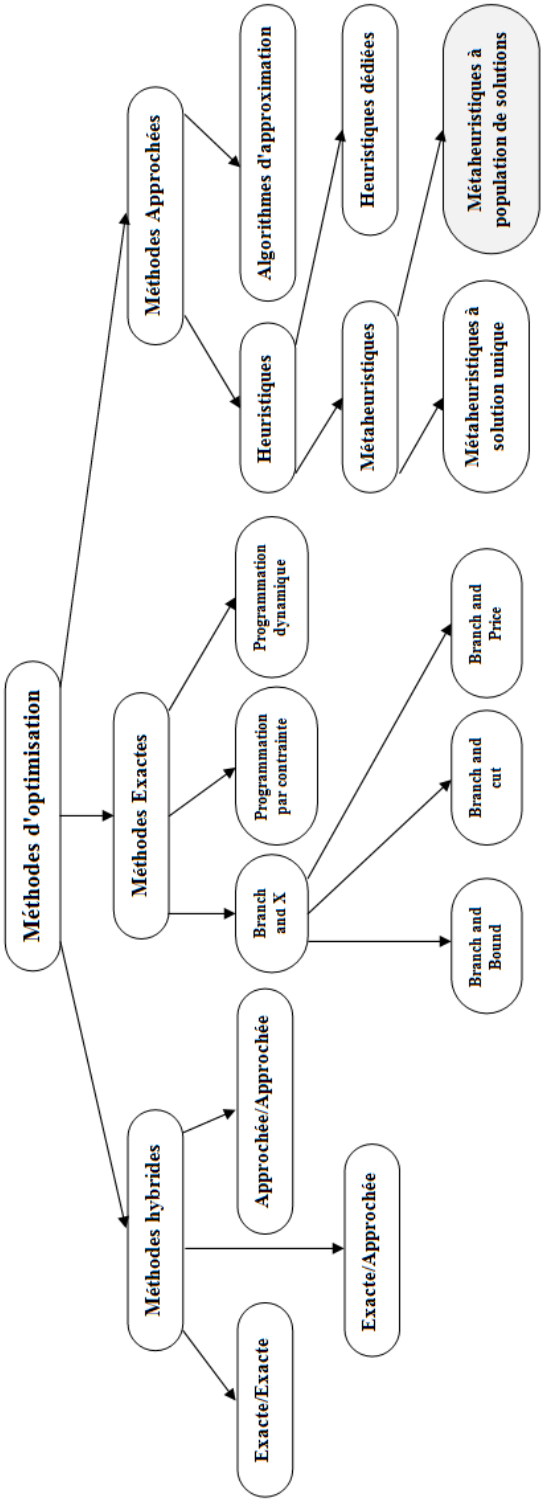


Figure 3.2: Hiérarchie pour les méthodes d'optimisation [11].

confronter sont basées sur la fonction d'objectif qui ne doit pas présenter les caractéristiques de l'absence de convexité, la discontinuité, la non-dérivabilité, la présence de bruit et en plus elle doit être définie avec précision. Le principe de travail de tels algorithmes est de parcourir la totalité de l'espace de recherche de façon à garantir l'obtention de toutes les solutions et d'en choisir la meilleure. Dans le cas d'une augmentation de la taille du problème, le temps de calcul pour que l'algorithme trouve une solution optimale risque d'augmenter exponentiellement. Dans cette situation le problème sera dit *"difficile"*, puisque aucune des méthodes exactes ne peut le résoudre dans un temps acceptable. Dans [38], l'auteur divise l'optimisation difficile en deux types de problèmes:

- *Les problèmes d'optimisation à variables discrètes* qui consistent à trouver une solution dans un ensemble discret. Le problème est qu'il existe un grand nombre de solutions réalisables, ce qui rend la recherche d'une solution optimale, une tache très difficile en un temps acceptable. La plupart des métaheuristiques sont conçues pour faire face à ce type de problèmes.

- *Les problèmes d'optimisation à variables continues* où le problème est moins formalisé que le premier, avec une fonction objectif accessible par simulation uniquement.

- Il existe des problèmes dits à *variables mixtes* (variables discrètes et continues pour le même problème).

Parmi les méthodes exactes nous pouvons citer quelques unes (ce n'est pas notre vocation): l'algorithme du simplexe, la programmation dynamique, l'algorithme A*, les algorithmes de séparation et évaluation (comme : Branch and Bound, Branch and Cut, ...etc.), la méthode de coupes planes (Cutting-Plane), et la méthode de la génération de colonnes (voir figure 3.2). Ainsi, les méthodes exactes ne sont pas adaptées à toutes les problématiques rencontrées et il est nécessaire d'avoir recours à des méthodes d'optimisation approchées dites *"heuristiques"* ou encore *"métaheuristiques"*, dont le temps de calcul est acceptable mais qui ne garantissent pas l'optimalité des résultats.

3.3.3.B Méthodes approchées

Les méthodes approchées constituent une alternative très intéressante pour traiter les problèmes d'optimisation de grande taille si l'optimalité n'est pas primordiale. De nombreuses méthodes approchées ont été proposées et utilisées depuis longtemps par de nombreux praticiens. On peut citer *"les méthodes gloutonnes"* et *"l'amélioration itérative"* : Par exemple, la méthode de Lin et Kernighan qui resta pendant longtemps la doyenne des algorithmes pour le problème du voyageur de commerce [39]. Elles sont plus efficaces que les méthodes exactes pour la résolution de problèmes difficiles en un temps acceptable. Nous pouvons classer les méthodes approchées en deux classes :

- Les heuristiques;
- Les algorithmes d'approximation.

(A) **Les heuristiques :**

Plusieurs définitions ont été données dans la littérature. D'abord, une heuristique est une méthode spécifique pour un problème donné, ce qui lui nécessite d'avoir des connaissances sur le domaine du problème à traiter en s'appuyant sur sa structure sans garantir la qualité de la solution. Elle est définie comme une règle empirique ou une règle d'estimation qui aide à découvrir la solution d'un problème parmi plusieurs d'autres pour améliorer l'efficacité d'un système à atteindre son but.

Le mot "heuristique" dérivé du mot grec ancien heuriskêin signifie "trouver".

Une heuristique est un algorithme approché qui permet de fournir rapidement et en un temps polynomial au moins une solution approchée (pas obligatoirement optimale) à un problème d'optimisation difficile. Elle peut accélérer le processus de résolution exacte. Par exemple, le fonctionnement d'une heuristique gloutonne est similaire à celui d'un algorithme glouton exact [38]. Selon [40], ces méthodes peuvent être classées en deux catégories:

Les méthodes constructives qui génèrent des solutions à partir d'une solution initiale en essayant d'en ajouter petit à petit des éléments jusqu'à ce qu'une solution complète soit obtenue.

Les méthodes de fouilles locales qui démarrent avec une solution initialement complète (probablement moins intéressante), et de manière répétitive essaie d'améliorer cette solution en explorant son voisinage.

La différence particulière entre une heuristique et une métaheuristique est la suivante:
Une métaheuristique est une heuristique générique qu'il faut adapter afin d'optimiser une large gamme de problèmes donnés, tandis que, une heuristique est une technique de résolution spécialisée pour le problème à traiter.

(B) **Les métaheuristiques :**

Le mot "métaheuristique" est composé de deux mots : le préfixe "méta" qui signifie "au-delà" ou bien "dans un niveau supérieur", et le mot "heuristique" signifie l'art de découvrir de nouvelles stratégies (règles) pour résoudre des problèmes [41] [10].

La recherche d'une solution pour les limites des heuristiques, a donné naissance à une autre méthode, applicable pour résoudre une grande variété de problèmes d'optimisation, qui est "la métaheuristique". Les métaheuristiques, qui sont apparues à partir des années 80, résoudront des problèmes dits "difficiles", avec une grande exploration de solutions dans l'espace de recherche par ces agents (particules). Ainsi, on peut dire qu'une métaheuristique est une méthode générique pour la résolution de problèmes combinatoires NP-difficiles. La solution est atteinte par une réduction de la taille et une exploration efficace de l'espace de recherche. Les avantages de ces méthodes sont la simplicité de conception et la flexibilité, puisqu'elles se basent sur trois objectifs principaux : la résolution rapide des problèmes, la résolution de grands problèmes et la robustesse des algorithmes. Ainsi, ces méthodes sont applicables et adaptables à une grande classe de problèmes.

Les métaheuristiques sont des algorithmes concurrents pour cette classe de problèmes afin d'obtenir de bonnes solutions pour les cas jugés trop complexes pour être résolus d'une manière exacte. Une métaheuristique peut également être utilisée pour générer de bonnes bornes inférieures ou supérieures pour les algorithmes exacts et d'améliorer leur efficacité [10].

Les approches de type métaheuristique sont basées sur des métaphores collectives d'insectes

sociaux et elles sont à haut niveau de granularité. Dans ce cas, les agents sont associés aux solutions possibles. Un agent construit ou améliore une solution et ce faisant parcourt séquentiellement l'espace de recherche, quant au nombre d'agents utilisés, il n'est pas nécessairement en relation avec la taille du problème. Les plus connues de ces métaheuristiques sont les colonies de fourmis (ou Ant Colony Optimization (ACO), en anglais) et les essaims de particules (OEP). Le but recherché est la résolution efficace du problème d'optimisation complexe [42].

Ces métaheuristiques possèdent quelques inconvénients majeurs communs, tels que:

- Le réglage difficile de paramètres
- L'impossibilité de prévoir l'efficacité avec un problème donné.

Les métaheuristiques reposent sur un ensemble de concepts [43]:

1. La mémoire (stockage de l'information recueillie par les agents)
2. L'intensification (amélioration de la pertinence de l'information par une recherche locale)
3. La diversification (exploration de nouvelles zones de l'espace de recherche, pour augmenter la quantité d'informations)

(C) **Extensions des métaheuristiques :**

Vu l'incapacité des métaheuristiques de résoudre des problèmes de différentes diversités, [38] et [43] ont cité divers extensions de métaheuristiques :

- Les métaheuristiques d'optimisation multi-objective : Dans ce cas, il y a plusieurs objectifs contradictoires à satisfaire, alors il n'existe pas un seul optimum global mais un ensemble d'optima, qui constituent tous un ensemble (qu'on appelle "surface de compromis") du problème à optimiser.

- Les métaheuristiques d'optimisation multimodale : Ici, on cherche tout un ensemble d'optimums globaux ou locaux.
- Les métaheuristiques d'optimisation de problèmes bruités : C'est le cas où il y a une incertitude dans les calculs de la fonction d'objectif pour la recherche de l'optimum.
- Les métaheuristiques d'optimisation dynamique : A cause de la variation temporelle de la fonction d'objectif durant le processus d'optimisation, ce type nécessite une flexibilité pour approcher au mieux la solution optimale à chaque pas de temps.
- Les métaheuristiques hybrides : C'est une hybridation de différentes méthodes d'optimisation, afin de tirer profit des avantages et faire face aux inconvénients
- Les métaheuristiques parallèles : Il y a recours à cette technique quand on cherche à accélérer les calculs par la répartition des tâches de calcul entre plusieurs processeurs. Il suffit bien choisir et adapter les métaheuristiques pour qu'elles soient bien distribuées sur plusieurs machines.

(D) **Classification des métaheuristiques :**

Les métaheuristiques sont des algorithmes stochastiques itératifs, qui peuvent manipuler une ou plusieurs solutions à la fois pour rechercher l'optimum.

Une des classifications les plus populaires des métaheuristiques est la classification selon le type de population (métaheuristique à population de solutions ou à une seule solution). La figure 3.3 donne quelques exemples pour ces deux types.

1. **Les métaheuristiques à base de solution unique** (souvent appelées **"Méthodes à recherche locale"** ou **"Méthodes de trajectoire"**) cherchent à améliorer itérativement une seule solution tant qu'il est possible; ces méthodes ont un pouvoir d'exploitation (intensification) fort. Ce type de métaheuristique adopte donc la recherche par voisinage qui est une recherche locale se basant sur la construction d'une trajectoire dans l'espace de solutions en se dirigeant vers la solution optimale. Ces techniques souffrent du problème de convergence rapide (ou convergence prématurée) vers l'optimum local, à l'exception de celles qui utilisent des techniques d'échappement.

2. **Les métaheuristiques à base de population de solutions** (souvent appelées **" Méthodes évolutives"**) se basent sur l'augmentation de la qualité moyenne d'un ensemble de solutions par la manipulation d'une population initiale selon un mécanisme de recherche bien défini. La recherche de l'optimum global s'effectue en parcourant un vaste espace de recherche; alors ces méthodes ont un fort pouvoir d'exploration (diversification). Ces métaheuristiques sont inspirées généralement de la nature (physique, biologie, éthologie, ...etc.) et on peut en distinguer deux types d'approches : l'approche utilisant des mécanismes d'évolution naturels (Algorithmes Génétiques (AG), Algorithmes Evolutionnaires (AE), ...etc.) et l'approche utilisant l'intelligence collective (OEP, Optimisation par Colonie de Fourmis (OCF), ...etc.) [41].

   Dans la littérature des métaheuristiques, nous somme souvent confrontés à deux termes : *l'intensification* (ou l'exploitation) et *la diversification* (ou l'exploration).

   L'intensification est une concentration sur le voisinage d'une solution élue, qui consiste à explorer en détails la région de l'espace de recherche comprenant des solutions de propriétés approchées.

   La diversification consiste à obliger le processus de rechercher à explorer d'autres régions dans l'espace de recherche non visités afin de découvrir d'autres solutions différentes de celles rencontrés auparavant pour accroître la quantité de ces informations.

3. **Autres classifications :** La figure 3.4, éllustre d'autres types de classifications des métaheuristiques rencontrées dans la littérature:

   - Selon l'inspiration d'un phénomène naturel ou non;
   - Selon la fonction d'objectif quelle soit statique durant le processus de recherche ou quelle soit dynamique. Pour ce cas, la fonction d'objectif change durant la progression de la recherche en faisant aussi changer la topologie de voisinage.
   - Selon l'usage de l'historique ou non. Pour les méthodes qui n'utilisent aucune mémoire sont dites à processus markovien et pour les méthodes à mémoire on distingue deux types : méthodes ayant une mémoire à court terme ou à long terme.

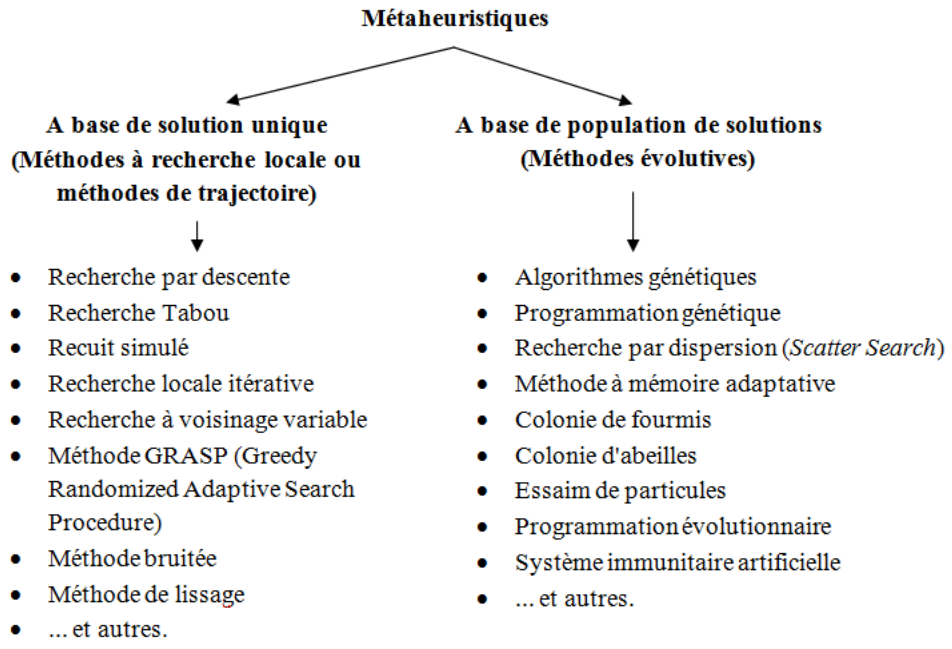


Figure 3.3: Classification populaire des métaheuristiques (avec exemples).

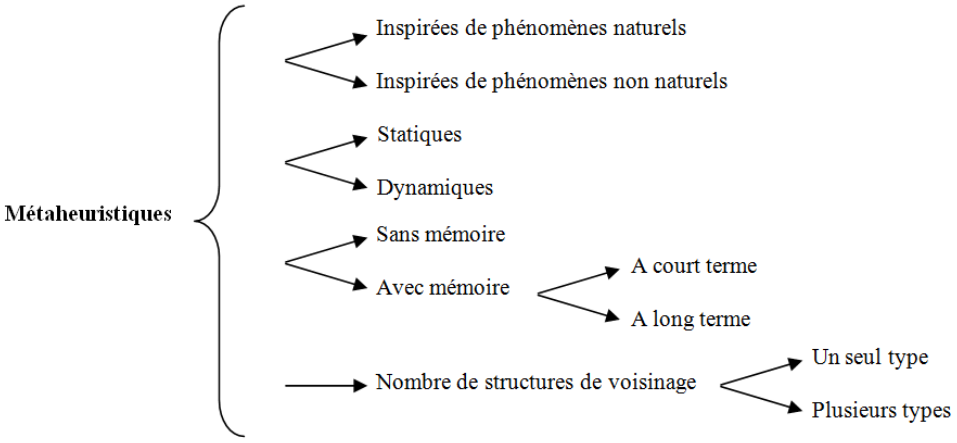


Figure 3.4: Divers classifications de métaheuristiques.

- Selon le voisinage d'un seul type ou à plusieurs types de voisinages.

3.3.3.C Méthodes hybrides

Une nouvelle classe des méthodes d'optimisation dites "Méthodes hybrides", a été créé par une combinaison des méthodes exactes et des méthodes approchées. Selon [36], des chercheurs ont envisagé des combinaisons de méthodes afin de bénéficier des points forts de chacune et

de proposer des alternatives beaucoup plus efficaces et performantes. Nous pouvons distinguer trois différents types d'hybridations:

- Hybridation exacte/exacte : Elle consiste à faire une hybridation entre des méthodes exactes et/ou entre leurs principes.
- Hybridation exacte/approchées : Elle consiste à faire une hybridation entre des méthodes exactes et des méthodes approchées et/ou entre leurs principes.
- Hybridation approchées/approchées : Elle consiste à faire une hybridation entre des méthodes approchées et/ou entre leurs principes.

3.4 Technique d'optimisation par essaim de particules

La méthode d'Optimisation par Essaim de Particules (OEP) (ou Particle Swarm Optimization (PSO) en anglais) est une métaheuristique basée sur l'intelligence par essaim (SI, de Swarm Intelligence en anglais), et a été proposée par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 [44]. Au début, d'après [45], J. Kennedy et R. Eberhart, cherchaient à simuler la capacité des oiseaux à voler de façon synchrone et leur aptitude à changer brusquement de direction tout en restant en une formation optimale. Le modèle qu'ils ont proposé a ensuite été étendu en un algorithme simple et efficace d'optimisation. Mais d'après [36], l'origine de l'idée de l'optimisation par essaim de particules a débuté en 1983, lorsque W.T.Reeves [46] donna une description d'un système à particules comme une collection d'un nombre de particules qui représentent ensemble un objet flou. Au cours d'une période de temps, des particules sont générées dans un système, se déplacent, changent de forme à l'intérieur de ce système et meurent dans le système. Quand une particule est générée, elle a une durée de vie et à chaque pas de temps, cette durée de vie est décrémentée jusqu'à ce qu'elle atteint zéro où la particule sera dite "tuée". D'autre part, Craig Reynolds [47] a tenté d'améliorer l'idée de Reeves en rendant le comportement des particules plus dynamique et mieux organisé par l'ajout de la notion d'orientation et la notion de communication inter-particules pour que le groupe vol correctement (sans collision entre les congénères); alors, il a proposé la notion de voisinage, en respectant les trois règles suivantes [48]:

Cohésion : Elle entraîne la particule à observer pour rester proche de ses voisines

Séparation : Elle oblige la particule à éviter les collisions en orientant loin les particules qui sont perçus comme trop proches

Alignement : Il imite le désir de synchroniser la vitesse et le cap avec les particules à proximité.

Dans un système OEP, un ensemble d'agents (essaim de particules) sont mis en jeu pour la résolution d'un problème d'optimisation. L'ensemble est appelé "essaim" et les agents sont appelés "particules". Chaque particule est une solution candidate au problème posé. Chaque membre de l'essaim survole ou parcourt l'espace de recherche avec une vitesse et ainsi il occupe une nouvelle position. Cette dernière est principalement influencée par deux meilleures positions:

- Sa meilleure position visitée par elle-même, qui représente sa propre expérience appelée "composante cognitive".

- La position visitée par la meilleure particule de son voisinage, qui représente l'expérience des particules voisines, appelée "composante sociale". Selon le voisinage d'une particule, l'algorithme est appelé "gbest" OEP si son voisinage est l'essaim toutes entier. Par contre, si des voisinages de plus petites tailles (locales) sont utilisés, l'algorithme est appelé "lbest" OEP.

Une autre composante influence le déplacement de la particule est sa propre vitesse appelée *"composante d'inertie ou physique"*.

La figure 3.5 donne la stratégie du déplacement d'une particule dans l'espace de recherche :

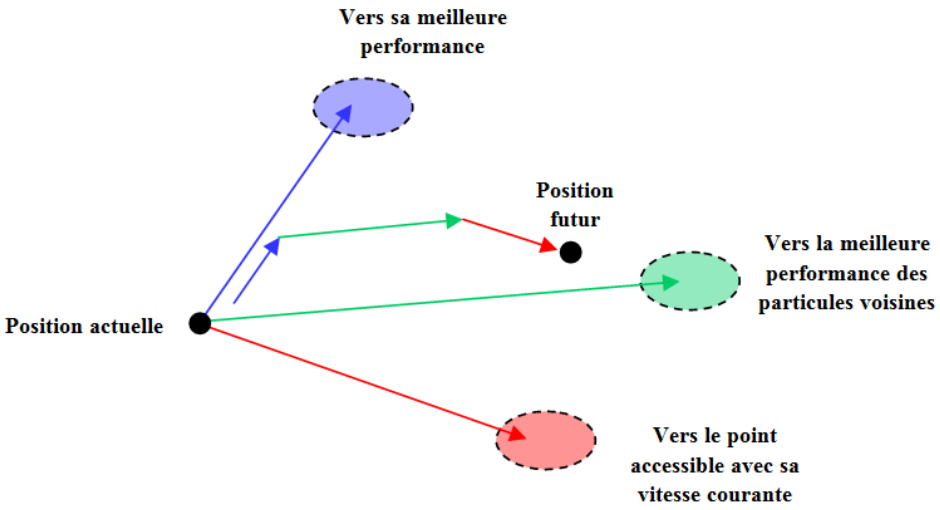


Figure 3.5: Déplacement d'une particule.

L'algorithme commence avec une initialisation de l'essaim dans l'espace de recherche.

3.4.1 Mise à jour de la position

La position de chaque particule i est modélisée par son vecteur position $\overrightarrow{x}_i^d = (x_i^1, x_i^2, \ldots, x_i^D)$ dans un espace de recherche de dimension D, et son mouvement par le vecteur vitesse $\overrightarrow{v}_i^d = (v_i^1, v_i^2, \ldots, v_i^D)$, selon l'équation (3.1) :

$$\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \overrightarrow{v_i}(t+1) \tag{3.1}$$

Quant à la qualité de sa position, elle est pondérée par la valeur de la fonction de Fitness F (ou fonction d'objectif) en ce point (§3.4.4). La fonction de fitness $F(x_i^d)$ de la particule x_i^d est une fonction spéciale au problème traité.

3.4.2 Mise à jour de la vitesse

La nouvelle vitesse $\overrightarrow{v_i}(t+1)$ qui permet de calculer la nouvelle position $\overrightarrow{x_i}(t+1)$ est donnée par l'équation (3.2) :

$$\overrightarrow{v_i}(t+1) = \overrightarrow{v}_i^d(t) + C_1.r_1^d(P_{best_i}^d - \overrightarrow{x}_i^d(t)) + C_2.r_2^d(G_{best}^d - \overrightarrow{x}_i^d(t)) \tag{3.2}$$

Avec :

$\overrightarrow{x}_i$: Position de la $i^{\text{ème}}$ particule dans la $d^{\text{ème}}$ dimension.

$\overrightarrow{v}_i$: Vitesse de la $i^{\text{ème}}$ particule dans la $d^{\text{ème}}$ dimension.

$P^d_{best_i}$: Meilleure position personnelle historique trouvée par la particule i.

G^d_{best} : Meilleure position globale trouvée par tout l'essaim.

C_1 et C_2 : Deux constantes appelées coefficients d'accélération ou paramètre cognitif et social, respectivement. Elles peuvent êtres non constantes dans certains cas [49] [50] [36].

r^d_1 et r^d_2 : Deux nombres aléatoires tirés uniformément dans l'intervalle [0,1], à chaque itération t et pour chaque dimension d.

Les trois composantes mentionnées dans l'équation (3.2) sont représentées par les termes suivants :

- $\overrightarrow{v}^d_i(t)$: modélise la tendance de la particule à se déplacer dans la même direction. En plus, elle empêche les particules de rester dans les minima locaux.
- $C_1.r^d_1(P^d_{best_i} - \overrightarrow{x}^d_i(t))$: correspond à la composante cognitive et représente une attraction linéaire vers la meilleure position jamais trouvée par la particule $P^d_{best_i}$. Cette composante est appelée "mémoire", "connaissance de soi", "nostalgie" ou "souvenir" et qui reflète le comportement d'auto-apprentissage de la particule.
- $C_2.r^d_2(G^d_{best} - \overrightarrow{x}^d_i(t))$: correspond à la composante sociale. Cette composante est appelée "coopération", "connaissance du groupe", "connaissance sociale" ou "information partagée" et qui reflète le comportement de l'apprentissage social de la particule.

3.4.3 Mise à jour des meilleures particules

Chaque particule mettra à jour (potentiellement) la meilleure solution locale:

$$\text{Si F}(x^d_i) \quad \text{Meilleure que} \quad \text{F}(P^d_{best_i}), \text{ alors : } P^d_{best_i} = x^d_i$$

En outre, la meilleure solution globale de l'essaim est mise à jour:

$$\text{Si F}(x^d_i) \quad \text{Meilleure que} \quad \text{F}(G^d_{best}), \text{ alors : } G^d_{best} = x^d_i$$

Par conséquent, à chaque itération, chaque particule va changer sa position en fonction de sa propre expérience et celle des particules voisines [10].

3.4.4 Fonction d'objectif F

La fonction objective F formule l'objectif à atteindre. Elle associe à chaque solution de l'espace de recherche une valeur réelle qui décrit la qualité de la solution. Ensuite, elle représente une valeur absolue et permet un classement complet de toutes les solutions de l'espace de recherche. La fonction objective est un élément important dans la conception d'une métaheuristique. Elle guide la recherche vers les «bonnes» solutions de l'espace de recherche. Si la fonction objective est mal définie, elle peut conduire à des solutions non acceptables quelle que soit la métaheuristique utilisée.

Pour certains problèmes d'optimisation, la définition de la fonction objective est simple. Elle spécifie la fonction initialement formulée comme objectif.

Pour d'autres problèmes, la définition de la fonction objective est une tâche difficile et constitue une question cruciale. La fonction objective doit être transformée pour une meilleure convergence de la métaheuristique. La nouvelle fonction objective guide la recherche d'une manière plus efficace [10].

3.4.5 Manipulation des contraintes

Faire face à des contraintes dans les problèmes d'optimisation est un autre sujet important pour la conception efficace des métaheuristiques. En effet, de nombreux problèmes d'optimisation continus et discrets sont contraintés, et il n'est pas trivial de faire face à ces contraintes. Les contraintes peuvent être de toute nature: linéaire ou non linéaire et à égalité ou à inégalité.

Les stratégies de manipulation de contrainte agissent principalement sur la représentation des solutions ou la fonction objective. Ils peuvent être classés comme des stratégies de rejet, de pénalisation, de réparation, de décodage, et de préservation [10].

3.4.6 Algorithme OEP de base

L'algorithme de base de l'OEP peut être donné par le pseudo code suivant :

Algorithme 1 : OEP de base

1. **Initialiser aléatoirement une population de N particules (position et vitesse) dans l'espace de recherche.**

2. **Evaluer la fonction de fitness F(x_i^d) pour toutes les particules i.**

3. **Poser $P_{best_i}^d = x_i^d$, pour chaque particule i.**

4. **Choisir G_{best}^d meilleur des $P_{best_i}^d$ (maximisation ou minimisation).**

5. **Répéter jusqu'à ce que le critère d'arrêt soit satisfait**
   5.1 Déplacer les particules par la MAJ des vitesses et des positions.
   5.2 Evaluer la fonction de fitness pour chaque particule.
   5.3 Calculer les nouveaux $P_{best_i}^d$ et G_{best}^d.

6. **Afficher la solution du problème d'optimisation considérée G_{best}^d.**

3.4.7 Sélection des paramètres

L'algorithme OEP a plusieurs paramètres qui doivent être spécifiés avant de lancer le processus d'optimisation. Ceux-ci comprennent la taille de l'essaim, le nombre maximum d'itérations ainsi que les coefficients (d'inertie w, cognitif C_1 et social C_2) de l'équation de la vitesse. Ces paramètres affectent les trajectoires des particules. Si la variante "OEP locale" est utilisée, une topologie de voisinage doit également être spécifiée. La taille de l'essaim affecte la capacité de recherche de l'algorithme OEP et il est choisi en fonction de la taille de l'espace de recherche D et de la difficulté des problèmes [51].

3.4.7.A Choix de la taille d'essaim

Le choix de la taille de l'essaim ou la taille de la population est l'une des étapes importantes pour l'initialisation de l'algorithme OEP. Pour une grande taille, il y aura une génération de grandes parties de l'espace de recherche à explorer par itération, ce qui augmente aussi la complexité de calcul et ainsi le temps d'exécution. Par contre, ce grand nombre de particules générées peut réduire le nombre d'itérations nécessaire pour une bonne optimisation. De nombreuses études empiriques ont montré que la plupart des implémentations de l'OEP utilisent un intervalle de $N \in [20, 60]$ de la taille des essaims [52].

Essentiellement, il y a deux paramètres sur les quels dépend le choix du nombre de la taille de l'essaim pour résoudre un problème d'optimisation :

1. La taille D de l'espace de recherche.
2. Le compromis entre la capacité de calcul de la machine et le temps maximal de recherche de solutions.

Il n'y a aucune règle qui fixe ce nombre, mais plusieurs essais permettent l'appréhension de ce nombre selon le problème posé.

Dans [53], la taille de l'essaim est calculée automatiquement par la formule (3.3) qui dépend de la dimension D du problème:

$$N = 10 + \lceil 2\sqrt{D} \rceil \tag{3.3}$$

Avec:

$\lceil u \rceil$: partie entière du nombre réel u.

3.4.7.B Choix du nombre maximal d'itérations

Pour un bon résultat d'optimisation, il faut un choix adéquat du nombre maximal d'itérations. Pour un petit nombre (< 20) le processus de recherche peut s'arrêter prématurément, tandis qu'un grand nombre d'itérations (> 40) peut provoquer une complexité inutile et par conséquent plus de temps de calculs [52].

Un choix adéquat des paramètres est très important pour une bonne convergence de l'algorithme de l'OEP. Selon [54], Clerc a donné quelques directives pour choisir la bonne combinaison de valeurs qui fonctionnent bien dans un large éventail de problèmes. Il a proposé d'utiliser les valeurs suivant le tableau 3.1 :

Paramètres	Symbole	Intervalle	Préférence
Taille de l'essaim	N	$[20, 40]$	20
Cognitif	C_1	$[0, 1]$	0.7
Social	C_2	~ 1.5	1.43
Vitesse maximale	v_{max}	$\sim (x_{max} - x_{min})/2$	**

Tableau 3.1: Directives de Clerc pour le choix de paramètres

3.4.7.C Coefficients d'accélération "C_1" et "C_2"

La combinaison des paramètres w, C_1 et C_2 permet de régler l'équilibre entre les phases de diversification et d'intensification du processus de recherche [38]. Les coefficients d'accélération C_1 et C_2 contrôlent le mouvement de chaque particule pour rechercher autour de sa meilleure position et autour de la meilleure position globale respectivement. Une petite valeur limite le mouvement de la particule, tandis qu'une grande valeur peut causer la divergence de la particule [55]. La multiplication des coefficients C_1 et C_2 par les valeurs aléatoires r_1 et r_2 respectivement, maintient l'influence stochastique des composantes cognitive et sociale de la vitesse de la particule [52].

Résumant le comportement des particules selon les propriétés de C_1 et C_2, dans le tableau 3.2:

Les auteurs ont conclu que, par une augmentation de la valeur de la constante d'accélération $c = C_1 + C_2$, la fréquence des oscillations autour du point optimal augmente. Pour des petites valeurs, la trajectoire est similaire à une forme d'onde sinusoïdale. La trajectoire tend vers l'infini pour des valeurs supérieures à $4,0$ [55].

Une initialisation incorrecte de "C_1" et "C_2" peut entraîner un comportement divergent ou cyclique. À partir des différentes recherches empiriques, il a été proposé que les deux constantes d'accélération doivent être $C_1 = C_2 = 2$ [52] [55].

C_1	C_2	**Comportement des particules**
$= 0$	$= 0$	Toutes les particules continuent à survoler avec leurs vitesses actuelles jusqu'à ce qu'elles atteignent la limite de l'espace de recherche.
> 0	$= 0$	Toutes les particules sont indépendantes.
$= 0$	> 0	Toutes les particules sont attirées par un seul point G_{best} dans tout l'essaim.
$= C_2$	$= C_1$	Toutes les particules sont attirées vers la moyenne de G_{best} et P_{best}.
$\gg C_2$	$--$	Chaque particule est plus fortement influencée par sa meilleure position personnelle.
$--$	$\gg C_1$	Toutes les particules sont beaucoup plus influencées par la meilleure position globale, ce qui provoque la convergence des particules prématurément vers l'optimum.

Tableau 3.2: Comportement des particules selons C_1 et C_2.

3.4.7.D Vitesse maximale "v_{max}"

Selon [55], la vitesse de la particule est une variable stochastique et est donc soumise à la création d'une trajectoire non contrôlée, ce qui oblige les particules à suivre des cycles plus larges dans l'espace de recherche. Pour amortir ces oscillations, les limites supérieures et inférieures peuvent être définies pour la vitesse $v_i^d(t)$:

$$\begin{cases} v_i^d(t) = v_{max} & \text{Si } v_i^d(t) > v_{max} \\ v_i^d(t) = -v_{max} & \text{Si } v_i^d(t) < -v_{max} \end{cases} \tag{3.4}$$

Si la vitesse maximale v_{max} est trop grande, les particules peuvent se déplacer de façon erratique et sauter par-dessus la solution optimale. D'autre part, si v_{max} est trop petit, le mouvement de la particule est limité et l'essaim peut ne pas explorer suffisamment ou peut se coincer dans un optimum local [52].

Aussi, la valeur v_{max} dépend des caractéristiques du problème. Dans les travaux de recherche effectués dans [55], les auteurs ont montré qu'une modification dynamique appropriée peut améliorer les performances de l'algorithme OEP. En outre, pour assurer une vitesse uniforme dans toutes les dimensions, [55] ils ont proposé une vitesse maximale donnée par :

$$v_{max} = \frac{x_{max} - x_{min}}{L} \tag{3.5}$$

Où L est le nombre d'intervalles dans la $d^{\text{ème}}$ dimension sélectionnée par l'utilisateur, et

"x_{max}" et "x_{min}" sont des valeurs minimales et maximales trouvées jusqu'à présent par les particules.

Le paramètre système v_{max} a l'effet bénéfique de la prévention de l'explosion et redimensionne l'exploration de la recherche de la particule. Malheureusement, le choix d'une valeur pour v_{max} dépend d'une connaissance du problème. Par exemple, si un grand pas de v_{max} est nécessaire afin d'échapper à un optimum local, la particule sera prise au piège. En outre, dans l'approche d'un optimum, il serait préférable de prendre des petits pas [56].

Ainsi, l'avantage de la limitation (verrouillage) de la vitesse est de contrôler l'explosion de la vitesse dans l'espace de recherche. D'autre part, l'inconvénient est que la meilleure valeur de v_{max} doit être choisie pour chaque problème d'optimisation différent en utilisant des techniques empiriques.

Enfin, v_{max} a été introduite pour éviter les explosions et la divergence. Cependant, il est devenu nécessaire pour la convergence l'utilisation du facteur d'inertie w ou le facteur de constriction χ [52].

3.4.7.E Facteur d'inertie "w"

Dans la mise à jour de la vitesse, le facteur d'inertie w est ajouté à l'équation (3.2) par multiplication avec la vitesse précédente, comme suit:

$$\overrightarrow{v_i}(t+1) = \omega * \overrightarrow{v}_i^d(t) + C_1 * r_1^d(P_{best_i}^d - \overrightarrow{x}_i^d(t)) + C_2 * r_2^d(G_{best}^d - \overrightarrow{x}_i^d(t)) \tag{3.6}$$

Ce facteur est considéré pour remplacer v_{max} par l'ajustement de l'influence des vitesses précédentes et aussi par le contrôle de la dynamique de la particule par mesure de la contribution de la vitesse précédente [52]; ainsi, il contrôle l'exploration de l'espace de recherche.

Une grande valeur de ce facteur (typiquement $0,9$) permet une exploration globale pour trouver l'optimum global alors qu'une valeur réduite de ce facteur (habituellement $0,4$) favorise une exploitation locale dans l'espace de recherche.

Le tableau 3.3, résume le comportement des particules suivant le facteur d'inertie :

$\omega \geq 1$	Les vitesses augmentent avec le temps et les particules ne peuvent changer leur direction pour retourner vers l'optimum et ainsi l'essaim se diverge.
$\omega \ll 1$	Un petit dynamisme est conservé à partir de la vitesse précédente ainsi il provoque un changement rapide de la direction des particules.
$\omega = 0$	Suppression des vitesses et les particules se déplacent sans connaître la vitesse précédente à chaque itération.

Tableau 3.3: Comportement des particules en fonction de ω

Van den Bergh et Engelbrecht Trelea, ont défini une condition qui garantie la convergence :

$$1 \geq \omega > \frac{1}{2} * (C_1 + C_2) - 1 \tag{3.7}$$

Un comportement divergent ou cyclique peut se produire dans le processus si cette condition n'est pas satisfaite [52] [45].

Une amélioration de l'OEP de base [57], est de décrémenter ce facteur linéairement au cours des itérations d'une valeur maximale w_{max} jusqu'à une valeur minimale w_{min}, en vue d'affiner la recherche au fur et à mesure de la convergence de l'algorithme, comme suit :

$$w(t) = w_{max} - \frac{(w_{max} - w_{min}).t}{T} \tag{3.8}$$

Tel que:

t: Itération courante.

T: Nombre d'itérations maximal.

3.4.8 Coefficient de constriction "χ"

Cette technique introduit un nouveau paramètre, nommé coefficient de constriction χ. Ce coefficient a été développé par Clerc [52]. Cette variante d'OEP, qui a été largement utilisée dans la littérature, est connue sous le nom de "Canonical PSO ou CPSO" [38]. Ce coefficient est très important pour contrôler l'équilibre entre l'exploration et l'exploitation, pour assurer la convergence du comportement, et pour écarter le facteur d'inertie w et la vitesse maximale w_{max} [52].

L'équation de la mise à jour de la vitesse pour la méthode de Clerc est donnée comme suit:

$$\overrightarrow{v_i}(t+1) = \chi * [\overrightarrow{v}_i^d(t) + \phi_1^d(P_{best_i}^d - \overrightarrow{x}_i^d(t)) + \phi_2^d(G_{best}^d - \overrightarrow{x}_i^d(t))] \tag{3.9}$$

Avec:

$$\chi = \frac{2}{|2 - \phi - \sqrt{(\phi^2 - 4 * \phi)}|}, \quad \text{avec : } \phi = \phi_1 + \phi_2, \phi_1 = C_1 * r_1^d, \phi_2 = C_2 * r_2^d$$

Dans [38] et [55] : $\phi_1 = C_1$, $\phi_2 = C_2$.

Pour cette méthode, si $\phi \geq 4$, les particules tournent autour de la meilleure solution dans l'espace de recherche sans assurer la convergence; par contre, si $\phi > 4$, les particules convergent rapidement et surement [52].

En général, le facteur de constriction améliore la convergence des particules au cours du temps en amortissant les oscillations (figure 3.6), si la particule est centrée sur le meilleur point dans la région optimale. L'inconvénient majeur de cette méthode est que les particules peuvent suivre des cycles plus larges et peuvent ne pas converger quand la meilleure position personnelle soit loin à celle de la meilleure position globale (deux différentes régions) [55].

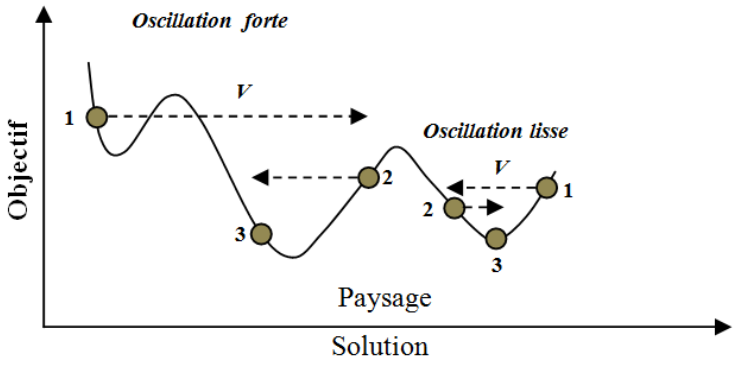


Figure 3.6: Oscillations fortes contre oscillations lisses dans un essaim de particules [10].

Plusieurs auteurs ont proposé des valeurs spécifiques de paramètres OEP ou des techniques pour obtenir des valeurs de paramètres appropriés. Ce sont des valeurs heuristiques qui ont été jugées contraires à des exigences de convergence de l'OEP. Il a été montré que les trajectoires

des particules peuvent être convergentes, cycliques ou divergentes. Des modifications ont été introduites pour réduire les problèmes de divergence de particules, telles que, l'utilisation de verrouillage de vitesse (ou restriction) et l'utilisation des facteurs d'inertie autres que l'unité. Ces modifications conduisent à de meilleures caractéristiques de convergence de l'algorithme OEP [51].

3.4.9 Topologie de voisinage

Dans les méthodes à base de population comme l'OEP, le regroupement, les connections et les distances entre les membres de la population deviennent des facteurs signifiants affectant leurs performances. Les topologies de voisinage (détermineés par le groupement des particules dans des sous-groupes) dans l'OEP déterminent la suffisance et la fréquence des flux d'information, et le coût de calcul de communication entre les particules [58].

Une particule à l'intérieur d'un voisinage peut communiquer et échanger de l'information à propos de l'espace de recherche avec les particules de son voisinage.

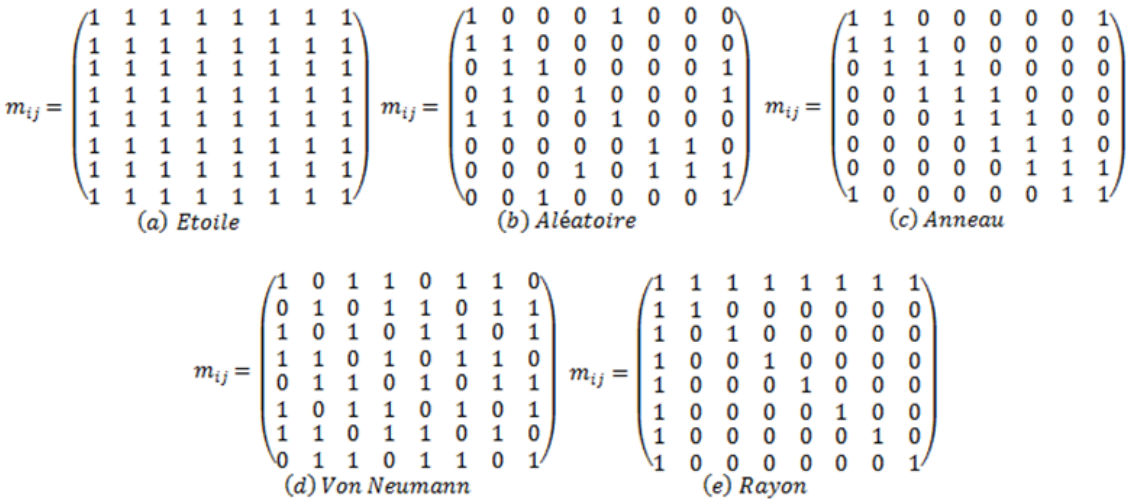


Figure 3.7: Matrices d'adjacence pour cinq topologies différentes.

Une particule j est dite à l'intérieur du voisinage d'une particule i, s'il y a un "lien" de la particule i (dite particule informée) vers la particule j (dite particule informante) [51].

Si les deux particules peuvent communiquer entre elles, ce lien est dit "bidirectionnel", sinon ce lien est "unidirectionnel" [58].

Selon [51], la topologie du voisinage est définie par une matrice adjacente (matrice de contiguïté) m_{ij}, où les lignes correspondent aux particules informantes et les colonnes correspondent aux particules informées (figure 3.7).

En général, cette matrice est asymétrique binaire, où l'entrée "1" indique que la particule i est à l'intérieur du voisinage de la particule j (c'est à dire la particule i informe la particule j). Cette matrice a des "1" dans la diagonale, puisque chaque particule appartient à son propre voisinage. Par l'utilisation de cette matrice, il est possible de définir les différents types de topologies de voisinage [51].

Principalement, il y a deux algorithmes de base pour l'OEP, le premier à base de voisinage global (gbest de Global Best) et le deuxième à base voisinage local (lbest de Local Best).

3.4.9.A OEP à base de voisinage global (Global best)

Les particules, dans l'OEP sous sa version "gbest", sont attirées vers la meilleure position trouvée par n'importe quel membre de l'essaim [55]; ainsi, chaque particule est influencée par la particule la mieux positionnée dans l'essaim tout entier. Parmi les topologies utilisant la version "gbest", nous citons : Topologie en étoile (Star Topology, figure 3.8(a)) et la topologie aléatoire (Random Topology, figure 3.8(b)) où chaque particule est la voisine de toutes les autres particules de l'essaim. Les topologies étoile, anneau, Von Neumann et rayon (figure 3.8(a), (c), (d), (e) respectivement) sont des topologies de voisinage fixées (les liens entre particules ne changent pas durant les itérations) tandis que la topologie aléatoire (figure 3.8(b)) est une topologie de voisinage local (les liens entre particules peuvent varier avec les itérations).

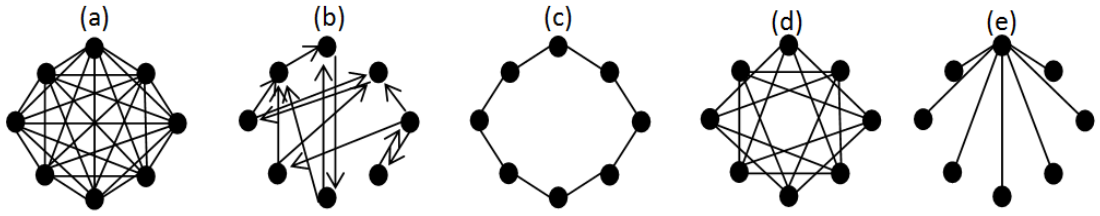


Figure 3.8: Topologies de voisinage:(a) étoile, (b) aléatoire, (c) anneau, (d) Von Neumann, (e) rayon.

Dans cette méthode, chaque particule individuelle, $i \in [1,\ldots,n]$ avec $n > 1$, a une position courante x_i dans l'espace de recherche, une vitesse courante v_i, et une meilleure position

personnelle P_{best_i} dans l'espace de recherche. La position P_{best_i} correspond à la position dans l'espace de recherche quand la particule i avait la grande valeur déterminée par la fonction de fitness F, si on considère un problème de maximisation. En outre, la position qui donne la plus grande valeur parmi toutes les meilleures positions personnelles P_{best_i} est dite "meilleure position globale" et est notée G_{best}. Les équations suivantes (3.10 et 3.11) donnent la mise à jour des valeurs de P_{best_i} et G_{best} respectivement:

$$P_{best_i}^{t+1} = \begin{cases} P_{best_i}^{t} & Si \quad F(x_i^{t+1}) < P_{best_i}^{t} \\ x_i^{t+1} & Si \quad F(x_i^{t+1}) \geq P_{best_i}^{t} \end{cases} \tag{3.10}$$

Considérant un problème de maximisation.

Et la meilleure position globale G_{best} est calculée comme suit:

$$G_{best} = max\left\{P_{best_i}^{t}\right\}, \text{avec} : i \in [1,\ldots,n] \quad \text{et} \quad n > 1 \tag{3.11}$$

Pour cette méthode, l'équation de la mise à jour de la vitesse est identique à l'équation (3.2) [52].

Les études sur les topologies de voisinage confirment que la topologie de voisinage globale tend à converger rapidement à un point dans l'espace de recherche à cause d'une exploration insuffisante, mais le risque de tomber dans un minimum local est élevé (convergence prématurée) [58]. La matrice d'adjacence est une matrice avec des "1" pour toutes les entrées [51].

3.4.9.B OEP à base de voisinage local (Local best)

Pour améliorer la convergence, d'autre variantes de la version originale dites version Local "lbest" ont été proposées dans la littérature de l'OEP. Dans ces versions, le terme G_{best} est remplacé par les termes L_{best_i}, où pour chaque particule i, on définit un ensemble de voisinage (i.e., l'information partagée est la meilleure solution trouvée dans le voisinage de chaque particule L_{best_i}) [38].

Seule la méthode "meilleur local OEP" (ou "lbest") permet à chaque particule d'être in-

fluencée par la particule la mieux positionnée choisie dans son voisinage. Ici, l'information sociale échangée dans le voisinage de la particule, ce qui dénote la connaissance locale de l'environnement [52]. Dans ce cas, la vitesse de la particule i est calculée par :

$$\overrightarrow{v_i}(t+1) = \omega * \overrightarrow{v}_i^d(t) + C_1 * r_1^d\Big(P_{best_i}^d - \overrightarrow{x}_i^d(t)\Big) + C_2 * r_2^d\Big(L_{best_i}^d - \overrightarrow{x}_i^d(t)\Big) \tag{3.12}$$

Où:

$L_{best_i}^d$ est la meilleure position que toute particule a trouvée dans le voisinage de la particule i à partir de l'initialisation jusqu'à l'itération t.

Parmi les topologies "lbest", nous citons: La topologie en anneau (Ring Topology) où chaque particule est connectée avec deux de ces voisines, alors chaque voisinage contient trois particules au total (figure 3.8.c), la topologie en rayon (Wheel Topology) dans la quelle toutes les particules sont séparées les unes des autres et toutes les informations sont transmises vers une particule focale (figure 3.8.e).

Les topologies de voisinage local ont tendance à converger lentement, mais elles ont la chance de converger vers un maximum global ou à des meilleurs points dans l'espace de recherche [58].

Il y a plusieurs autres types de topologies (Pyramid Topology, Clusters Topology, ... etc.), mais aucune d'elles n'est meilleure que l'autre puisque ça dépend du problème d'optimisation à résoudre.

3.4.10 Critères d'arrêt

La technique d'optimisation par essaim de particules OEP, comme d'autres méthodes itératives, nécessite une condition à satisfaire par laquelle la recherche doit être interrompue. Cette condition est dite "Critère d'arrêt". Il y a plusieurs critères d'arrêt, nous pouvons citer quelques uns:

1) Selon le nombre d'itérations maximal: Pour certains cas, il suffit de dépasser le nombre maximal des itérations pour que le processus de recherche s'arrête. Cette condition est

généralement utilisée pour l'observation du processus d'optimisation.

2) Selon la fonction d'objectif: Si l'utilisateur connaît d'avance la valeur de la solution à atteindre, il suffit de mettre cette valeur comme critère d'arrêt.

3) Selon la progression de l'optimisation: C'est le cas où il n'y a aucune amélioration significante après un certain nombre d'itérations (Par exemple, si la moyenne de la vitesse reste nulle pendant plusieurs étapes [52]).

3.5 Technique d'optimisation par recherche stochastique fractale

3.5.1 Introduction

La base de l'algorithme de Recherche Stochastique Fractale (RFS ou SFS, en anglais) est la Recherche Fractale (RF) (ou Fractal Search (FS), en anglais), proposé par Benoît Mandelbort en 1975 (figure 3.9). Cet algorithme est un algorithme dynamique qui contient un certain nombre d'agents de recherche qui sont modifiés, d'où la RF demande plus d'effort [59]. Chaque agent de recherche tente d'imiter la répartition des propriétés de branchement (trop de paramètres d'entrée sont nécessaires). Ceci permet à l'algorithme RF de résoudre plus efficacement les problèmes d'optimisation globale [60].

Figure 3.9: Benoit Mandelbort.

3.5.2 Algorithme RFS

Un algorithme d'optimisation puissant nommé RFS, récemment proposé par [12] (en 2015); ce nouvel algorithme métaheuristique a été créé à partir d'un concept mathématique fondamental appelé "fractal". Cette technique est basée sur deux processus principaux: (i) le processus de diffusion et (ii) le processus de mise à jour. Dans le premier processus, similaire à la RF, chaque agent (particule) diffuse et crée de nouvelles particules au hasard autour de sa position actuelle pour remplir la propriété d'intensification. Cette étape étend la possibilité de déterminer les minima (ou maxima) globaux et évite d'être coincé dans les minima (ou maxima) locaux [61].

Deux méthodes sont utilisées pour créer de nouvelles particules, y compris le vol de Levy (Levy flight, en anglais) et la marche gaussienne (Gaussian Walk, en anglais). Dans la technique RFS, une série de la seconde méthode est utilisée pour effectuer un processus de diffusion, en changeant le processus d'itération pour générer des fractals aléatoires. L'avantage d'utiliser la marche gaussienne est due à sa rapidité à converger et est capable de trouver les minima globaux [62] [61]. Dans le processus de mise à jour, les positions des agents de recherche (points) sont mises à jour en fonction de la position des autres particules dans le groupe. Deux processus de mise à jour sont utilisés pour identifier le meilleur agent de recherche. Dans ce processus, peu de meilleurs agents de recherche sélectionnés par le processus de diffusion sont considérés, et le reste d'entre eux sont rejetés. Ces agents de recherche donnent l'emplacement optimal des particules [59]. De plus, certaines procédures statistiques sont utilisées dans la RFS pour éviter le piégeage dans les minima locaux [60].

L'algorithme de recherche fractale stochastique peut être décomposé en trois étapes principaux [63] [64]:

Étape 1: Initialisation de la taille de la population N_p, génération maximale (ou maximal Generation (Gen)), Nombre Diffusif Maximal (NDM) (ou maximum diffusion number), taille de la dimension N_{dim}, la position de chaque agent de recherche (éq. 3.13) au hasard, qui dépend du problème en spécifiant les limites inférieures et supérieures (Lower Boundary (LB) et Uper Boundary (UB) respectivement) comme:

$$P_i = LB + rand(UB - LB) \quad \text{avec} : i = 1, \cdots, N_p \tag{3.13}$$

P_i: Vecteur des agents de recherche initiale.

$rand$: crée un nombre uniformément réparti dans l'intervalle $[0, 1]$.

Ensuite, pour trouver le meilleur point (Best Point (BP), en anglais) parmi tous les agents de recherche, une fonction de fitness doit être évaluée pour chaque agent de recherche du groupe.

Etape 2: pour le processus de diffusion, l'algorithme utilise la marche gaussienne (3.14 et 3.15) pour générer de nouveaux agents de recherche. Deux séquences de la marche gaussienne aléatoire (Gaussian Random Walks (GRW)) utilisées dans cette étape, sont données par:

$$GRW_1 = \text{Gaussian}(|BP|, \sigma) + (rand \times BP - rand' \times P_i) \tag{3.14}$$

$$GRW_2 = \text{Gaussian}(|P_i|, \sigma) \tag{3.15}$$

Où, P_i est le $i^{\text{ème}}$ agent de recherche dans le groupe, $rand$ et $rand'$ sont des nombres aléatoires distribués et limités à $[0, 1]$, σ est un écart type calculé par l'équation 3.16 comme suit:

$$\sigma = \left| \frac{\log(g)}{g} \times (P_i - BP) \right| \tag{3.16}$$

Où g est le numéro de génération.

Le terme log g / g est utilisé pour diminuer la taille du saut gaussien tout en augmentant le nombre de générations [64].

Étape 3: avant la mise à jour, chaque agent de recherche est classé en fonction de ses valeurs de fitness par:

$$P_{P_i} = \frac{rank(P_i)}{N_p} \tag{3.17}$$

Où P_{P_i} est la valeur de probabilité de P_i.

Dans le premier processus de mise à jour, la nouvelle position de mise à jour $P_i^1(j)$ des

agents de recherche $P_i(j)$ est calculée selon cette équation:

$$P_i^1(j) = P_\alpha(j) - rand[P_\beta(j) - P_i(j)] \tag{3.18}$$

Où $P_\alpha(j)$ et $P_\beta(j)$ sont des agents de recherche choisis arbitrairement dans le groupe.

Dans le deuxième processus de mise à jour, la position de mise à jour P_i^1 est modifiée en P_i^2 selon que la condition $P_{P_i} < rand_g$ est satisfaite, sinon elle reste la même. Ces positions sont calculées selon les équations 3.19 et 3.20, comme suit:

$$P_i^2 = P_i^1 - rand(P_\alpha^1 - BP) \quad \text{if} \quad rand_g \leq 0.5 \tag{3.19}$$

$$P_i^2 = P_i^1 - rand(P_\alpha^1 - P_\beta^1) \quad \text{if} \quad rand_g > 0.5 \tag{3.20}$$

Où P_α^1 et P_β^1 sont des positions choisies au hasard, et $rand_g$ est un nombre aléatoire calculé par approche gaussienne. Si la valeur de fitness de P_i^2 est meilleure que P_i^1, alors P_i^1 sera remplacée par P_i^2.

3.5.3 Organigramme de la RFS

L'organigramme de la technique RFS (donné par la figure 3.10), débute par une phase d'initialisation de la taille de la population et l'évaluation de la fonction de fitness pour chaque particule générée puis de choisir la meilleure particule (BP) dans le groupe. Après la vérification du critère d'arrêt, l'algorithme exécute le processus de diffusion après avoir fixé le NDM. Selon ce nombre, l'algorithme commence le processus de diffusion en utilisant la marche gaussienne (3.14 et 3.15) et le processus se termine par le choix de la meilleure particule. Ensuite, l'algorithme exécute le processus de mise à jour, qui se divise en deux processus : (1) Dans le premier processus de MAJ, la nouvelle position $P_i^1(j)$ est mis à jour par l'équation 3.18. (2) Dans le deuxième processus de MAJ, la position de la particule P_i^1 est mis à jour et modifiée en P_i^2 selon les équations 3.19 et 3.20.

Le processus d'optimisation s'itère jusqu'à la satisfaction du critère d'arrêt; ainsi, les résultats sont affichés et peuvent être visualisés.

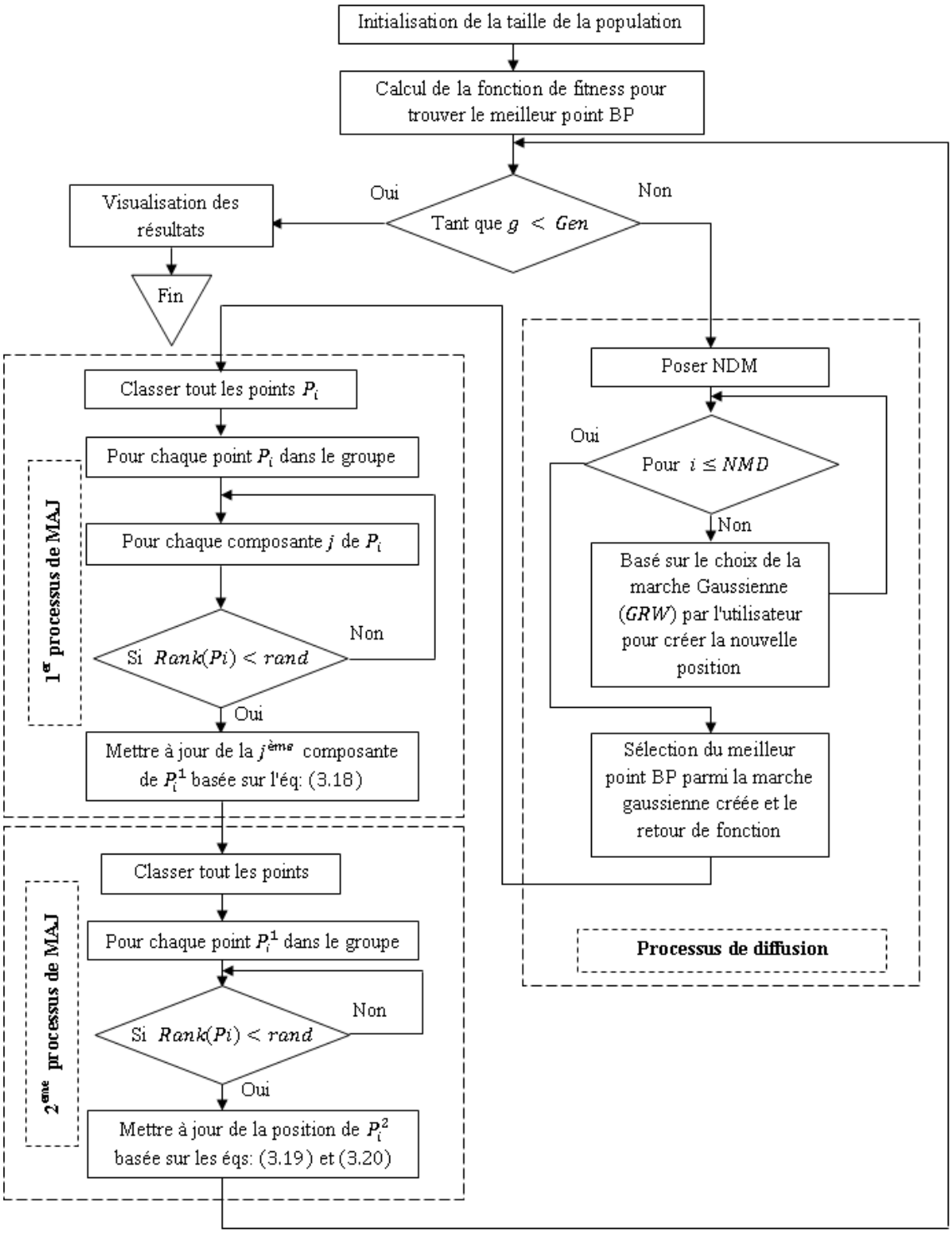


Figure 3.10: Organigramme de la RFS

3.6 Conclusion

Nous avons présenté dans ce chapitre l'état de l'art de deux métaheuristiques à base de population, qui sont des méthodes d'optimisation multi objectifs statiques. La première méthode est l'optimisation par essaim de particules ou OEP et la deuxième est l'optimisation par recherche fractale stochastique ou RFS. Tout d'abord, nous avons abordé la théorie de complexité pour donner une idée sur les différentes classes de difficulté des problèmes. La résolution de problèmes classés difficiles nécessite l'utilisation d'algorithmes dont la complexité augmente avec la dimension du problème. Les chercheurs cherchent à résoudre ces problèmes, en les transformant en un problème d'optimisation.

Ensuite, nous avons défini l'optimisation comme un moyen de résolution de problèmes; ainsi que les différents types de sa classification. Parmi les méthodes d'optimisation qui ne nécessitent pas beaucoup de temps de calcul ni une grande mémoire de stockage, il y a les méthodes dite "approchées" dont les "métaheuristiques" font partie. L'optimisation par essaim de particules "OEP" est une métaheuristique à population de solutions, a fait l'objet d'une étude détaillée dans ce chapitre.

Le choix de cette méthode en particulier est dû essentiellement à sa simplicité et sa facilité de mise en œuvre. D'un autre côté, notre application consiste à calculer un nombre important de paramètres de réglage pour atteindre l'objectif voulu, ce qui explique l'utilisation d'une métaheuristique à population. Cette méthode nécessite un réglage approprié de ces propres paramètres et le choix de la bonne topologie afin de bénéficier de ses bonnes performances.

4

Optimisation de placement 2D d'un réseau de caméras par PSO

4.1 Introduction

Il est important d'observer que la capture de mouvement en utilisant un système MoCap [3] est liée aux dispositions favorables des caméras lors de leurs prises de vue. Beaucoup de recherches ont été menées pour trouver ces dispositions; l'objectif principal est de maximiser la couverture ou de couvrir des points spécifiques. Cependant, mêmes avec ces recherches, il nous manque une méthode générale pour résoudre ce problème en considérant l'espace continu, et la présence des obstacles en employant l'OEP standard.

L'optimisation par essaim particulaire [43] est une technique d'optimisation très récente, dérivée de la famille d'algorithmes métaheuristiques et a été proposée par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 [44]. Son origine vient des observations faites lors des simulations informatiques des interactions sociales entre les agents de certains systèmes naturels tels que la métaphore et les oiseaux. Ces simulation ont mis en valeur la notion d'un essaim et ont également révélé l'importance de l'intelligence collective pour converger vers la solution optimale globale du problème traité, à partir d'optimums locaux et empiriques de chaque agent [43], [65].

4.2 Travaux connexes

Le problème de dispositions optimales des caméras a été traité depuis des décennies par différentes techniques. On peut citer les travaux suivants :

Dans [66], ils ont relié le problème de placement des caméras par le problème de placement des gardiens (AGP) et ils ont établi un théorème connu sous le nom de théorème d'art galerie (AG) en déclarant qu'afin de garder une galerie sous la forme d'un polygone simple avec n sommets couvertes, il est nécessaire d'arranger le nombre n/3 des gardiens. Ce théorème a été basé sur la géométrie informatique et a mené le principe de la triangulation.

Dans [67], ils ont divisé la scène en un ensemble de points de grille et ont dérivé un modèle de programmation linéaire; d'après les résultats, la couverture complète de la scène n'est assurée qu'avec une fréquence d'échantillonnage élevée; une telle technique ne peut être utilisée dans les grandes scènes puisque le nombre de variables binaires à utiliser est limité par la

mémoire et par le système.

Mais récemment, avec l'apparition des métaheuristiques et leurs utilisations intense dans différents domaines, plusieurs chercheurs ont appliqué la PSO pour trouver les dispositions optimales des caméras. Par exemple dans [68], ils ont présenté un algorithme qui optimise les positions de caméras en combinant la couverture globale et locale et ils ont définit la couverture globale par deux paramètres: la densité du pixel et la qualité de vision, ainsi que la couverture locale qui concerne la couverture des points d'intérêt comme les portes, les fenêtres…etc., par une fonction exponentielle négative de la couverture globale tout en utilisant la technique PSO. Dans [69], ils ont employé la PSO binaire inspirée par la probabilité pour l'optimisation des placements des caméras en deux et trois dimensions; cette technique a pu optimiser la couverture, même dans les grandes scènes et résoudre le problème de la capacité mémoire avec la programmation linéaire.

Nous visons dans ce chapitre à optimiser les dispositions 2D des caméras utilisées dans un système (MoCap), d'une manière géométrique en considérant l'espace continu et la présence des obstacles par la technique PSO standard et autres variantes.

4.3 Notre proposition

Dans notre proposition, nous devons optimiser le placement d'un certain nombre de caméras (de quatre à dix caméras) dans un système MoCap pour couvrir un mouvement virtuel d'un objet. Le but du placement d'un réseau de caméras est que le réseau doit couvrir les marqueurs placés sur l'objet par trois caméras en même temps durant chaque trame de la scène.

Le premier avantage, pour lequel nous avons choisi cette métaheuristique, est dû au fait que la technique PSO est une technique rapide en raison de sa simplicité algorithmique. Le second, est que nous avons quelques paramètres à ajuster (la position (C_{x1},C_{x2}), la pose φ, de quatre à dix caméras, c'est à dire 12 à 30 paramètres) et quelques contraintes à respecter (marqueur à l'intérieur du champ de vision de la caméra, marqueur vu par trois caméras dans la même trame, angle critique, éviter l'obstacle).

Les contributions les plus importantes de ce chapitre sont:

- Optimisation des positions et des orientations d'un réseau de caméras, en utilisant les coordonnées 2D d'un objet virtuel dans un système MoCap.

- Optimisation des positions et des orientations selon une condition principale (chaque marqueur est vu par trois caméras au moins dans chaque image durant tout le mouvement).

- Utilisation de trois variantes de la technique PSO, pour une étude comparative à base des résultats obtenus.

4.4 Critères de placement de la caméra

Le problème de positionnement de caméras peut être défini comme la manière de les poser (sous certaines contraintes) dans des endroits appropriés afin de maximiser la couverture des marqueurs tout en se déplaçant dans l'espace de travail. Ces contraintes peuvent être divisées en trois types principaux: La contrainte de tâche comprend une couverture complète des marqueurs, la contrainte de la caméra inclut les paramètres intrinsèques (longueur de focalisation, taille de la diode à couplage de charge (CCD), ...etc.) et la contrainte de la scène inclut la zone observable (2D, avec ou sans obstacles, ...etc.).

4.4.1 Modélisation du champ de vision d'une caméra

Dans un espace 2D, selon [67], le champ de vision d'une caméra (Field of View (FoV), en anglais) est décrit par un triangle comme le montre la figure 4.1.

Dans [38], la représentation du champ de vision de la caméra (FoV) est décrite par son angle d'ouverture α (α dans le cas 2D ou α_h et α_v dans le cas 3D) et sa position d'origine. Le calcul de la projection du FoV d'une caméra dans un plan 2D en fonction des paramètres intrinsèques de la caméra est décrit dans [70].

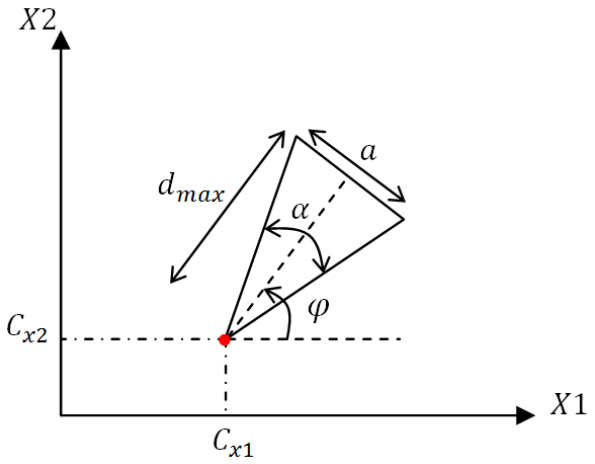


Figure 4.1: Paramètres du FoV dans un plan 2D.

La modélisation du champ de vision de chaque caméra est inspirée par [71], en utilisant un triangle isocèle (figure 4.2).

Nous supposerons dans ce qui suit, qu'un réseau de N_C caméras ($C_i, \quad i = 1, \cdots, N_C$) doit être déployé dans un espace de travail bien défini. Chaque caméra C^i est déployée dans sa position (C_{x1},C_{x2}), et posée avec un angle horizontal φ à la bissection de l'angle de vue α, et enfin la distance de travail de la caméra d_{max}.

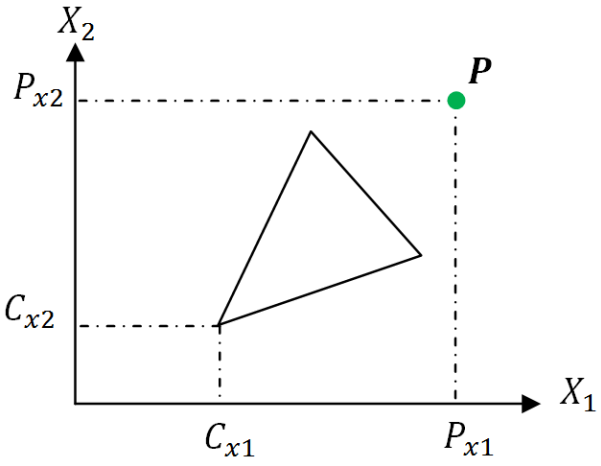


Figure 4.2: Position du point P dans un plan 2D.

Ainsi, un point P positionné à (P_{x1},P_{x2}) (voir figure 4.2) peut être visualisé par n'importe

quelle caméra C^i, si les contraintes ci-dessous (4.1 et 4.2) sont satisfaites [38].

$$fct_1 \leq d_{max} \tag{4.1}$$

$$\left(\frac{-a}{2.d_{max}}\right).fct_1 \leq fct_2 \leq \left(\frac{+a}{2.d_{max}}\right).fct_1 \tag{4.2}$$

Avec:

$$fct_1 = \cos\varphi.(P_{x1} - C^i_{x1}) + \sin\varphi.(P_{x2} - C^i_{x2})$$
$$fct_2 = \cos\varphi.(P_{x2} - C^i_{x2}) - \sin\varphi.(P_{x1} - C^i_{x1})$$

Deux régions représentées par deux plans (N de Near et F de Farther en anglais, voir figure 4.3) du champ de vision définies par [70], sont appelées "Zones mortes". Le plan N (le plus proche de la caméra) correspond à la hauteur minimale (d_{min}) et le plan F (le plus éloigné de la caméra) correspond à la hauteur maximale (d_{max}). La zone qui se trouve entre les deux plans est la "région visible". Si deux points proches l'un de l'autre se trouvent à l'intérieur de cette région, la résolution qui permet de distinguer entre eux est considérée.

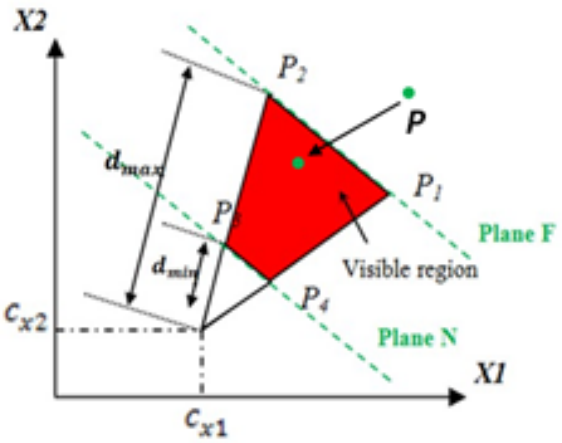


Figure 4.3: Région visible trapézoïdale.

4.4.2 Contrainte d'angle critique

Dans [43], la contrainte d'angle critique (figure 4.4) garantit que l'angle θ entre deux caméras voisines qui distinguent le même point $P(P_{x1}, P_{x2})$ ne doit pas dépasser $\theta_{i,i+1} = \theta_{min}$. Ce critère assure que le positionnement de deux caméras adjacentes donne deux images identiques, c'est à dire une redondance d'information.

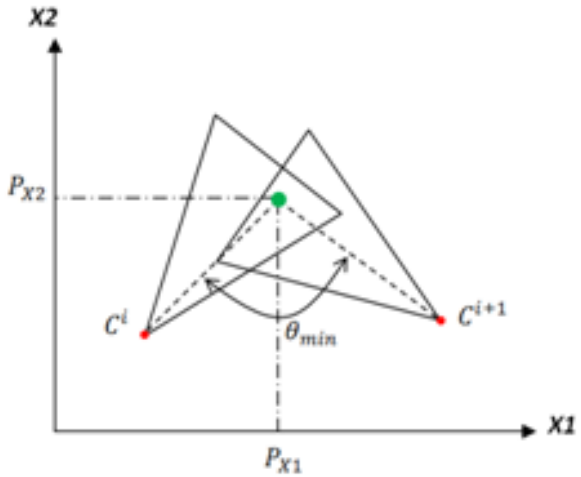


Figure 4.4: Angle critique θ_{min}.

L'angle critique est donné par l'équation 4.3 suivante:

$$\theta_{i,i+1} = \arccos \frac{|C^i - P|.|C^{i+1} - P|}{\|C^i - P\|.\|C^{i+1} - P\|} \tag{4.3}$$

Avec:

C^i,C^{i+1}: Coordonnées de deux caméras adjacentes.

$P = (P_{x1}, P_{x2})^T$: Coordonnées du point vu par les deux caméras.

4.4.3 Considération des obstacles

Dans le cas réel, il faut prendre en compte les occlusions (dues aux obstacles, figure 4.5) qui empêchent la capture du marqueur par la caméra.

Dans notre application, les obstacles sont des occlusions externes (présence d'un objet opaque entre la caméra et le marqueur) et ils sont simulés par des segments noirs (murs). La

décision qu'un élément est invisible à une caméra est qu'il y aura une intersection entre la ligne définie par les extrémités du segment (S^1,S^2) et celle définie par les coordonnées du point et de la caméra (P,C^i).

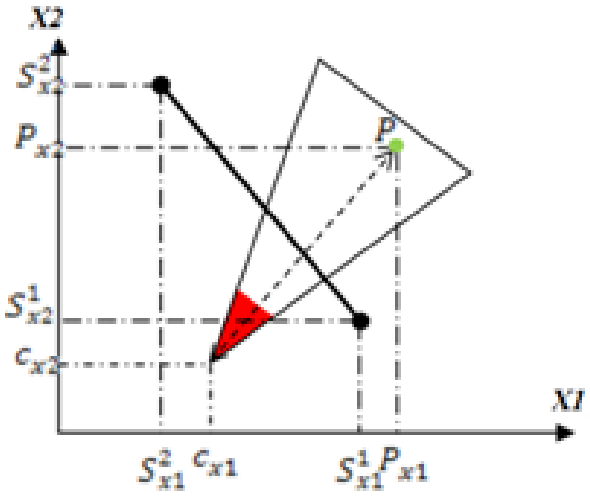


Figure 4.5: Considération des obstacles (P n'est pas visible par la caméra en raison de la présence d'un obstacle entre eux).

Dans ce qui va suivre, nous présenterons dans la première application nos travaux d'optimisation de placement d'un réseau de quatre caméras par la technique WPSO. Nous montrerons aussi dans cette application le comportement de la particule pour différentes déclarations et choix des paramètres propres à la WPSO. Dans la deuxième application, nous essayerons d'augmenter la difficulté d'optimisation par l'introduction d'un objet équipé de plusieurs marqueurs à ses extrémités et le déploiement de plus de quatre caméras dont le placement nécessite beaucoup de paramètres. Ainsi, nous créerons plus d'un scénario avec différents types de trajectoires et d'obstacles.

4.5 Application 1 : Recouvrement d'un point

Dans cette application, nous tentons d'optimiser la disposition de quatre caméras (une par une) utilisées dans un système MoCap afin qu'un point donné (ou une trajectoire du point) soit à l'intérieur du champ de vision de deux caméras au minimum.

Pour la réalisation de cette application, nous avons utilisé comme langage de programmation le logiciel Matlab sous sa version V.7.10 (32 bits) installé sur un PC utilisant un système d'exploitation Windows 7 et équipé d'un processeur Intel Atom^{TM} de fréquence 1,6 GHZ et une RAM de 2GB DDR3.

Le modèle de notre système MoCap comprend un nombre de caméras positionnées sur des rails où chacune ne peut se déplacer que sur le sien. Cette limitation des positions peut introduire le problème de la portée insuffisante pour l'observation du point. Cette considération nous a permis d'améliorer notre algorithme par l'ajout de l'optimisation de la portée (d_{max}).

Pour évaluer notre algorithme, nous proposons d'étudier trois scénarios différents, le tableau 4.1 donne une description de chacun :

Tableau 4.1: Scénarios proposés

Scénario N°	Description
1	Un point à couvrir par quatre caméras dans une scène sans obstacles
2	Même scénario N°1 avec l'ajout d'un obstacle
3	Un groupe de points à couvrir (scène sans obstacle)

4.5.1 Description de l'organigramme

Notre application peut être structurée selon l'organigramme de la figure 4.6.

a) Initialisation:

La PSO est un algorithme de population, qui commence avec une initialisation aléatoire de l'essaim et une déclaration des paramètres de réglage.

La taille d'essaim a été fixée pour 20 particules, et les valeurs des coefficients de confiance (C_1 et C_2) ont été fixées d'une manière empirique, où nous avons constaté après plusieurs essais que les valeurs ($C_1 = 0.2$ et $C_2 = 1.2$) donnent de très bons résultats concernant l'optimisation pour cette application.

Le coefficient d'inertie varie généralement dans l'intervalle [0.4,1.4]. Une grande valeur de ($\omega > 1$) est synonyme d'une grande amplitude de mouvement et donc, exploration. Au contraire, une faible valeur de ($\omega < 1$) est synonyme d'une faible amplitude de mouvement

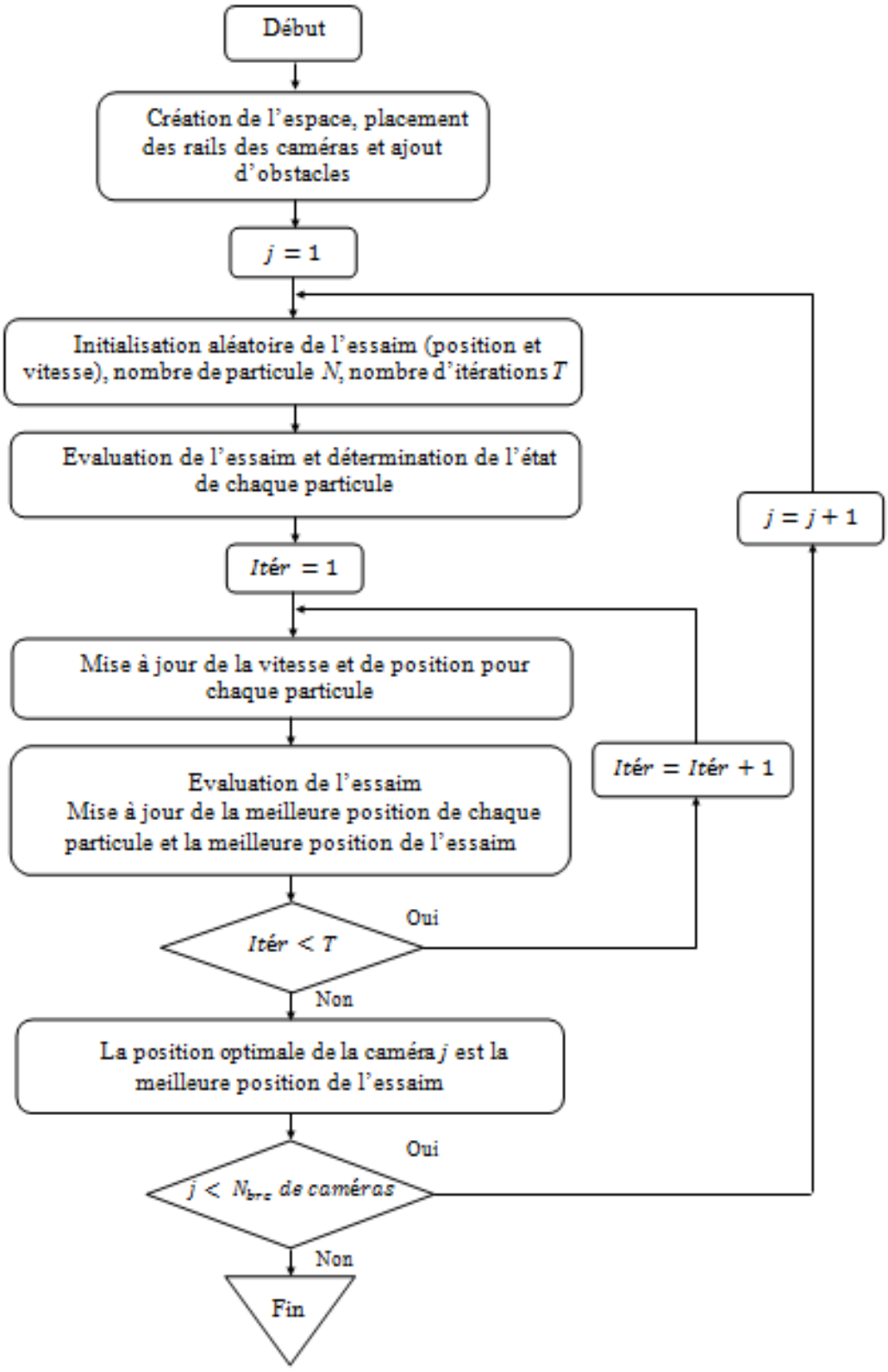


Figure 4.6: Organigramme de la 1ère application (Recouvrement d'un point).

et donc d'exploration locale (exploitation); afin de trouver un compromis entre l'exploration globale et locale [71], nous pouvons le fixer à la valeur 0,9.

b) Espace de recherche:

Notre objectif pour cette application est de trouver la solution optimale parmi plusieurs solutions proposées. Ces dernières sont choisies dans un espace de recherche qui définit les limites maximale et minimale pour chaque paramètre (C_{X1}, C_{X2}, orientation φ et portée d_{max}) pour chacune des quatre caméras. Le tableau 4.2 résume les limites de l'espace de recherche:

Tableau 4.2: Espace de recherche

Caméra N°	**Position [m]**				**Orientation [°]**		**Portée [m]**	
	$C_{X1_{min}}$	$C_{X1_{max}}$	$C_{X2_{min}}$	$C_{X2_{max}}$	φ_{min}	φ_{max}	$dmax_{min}$	$dmax_{max}$
1	0	20	20	120	270	90	50	70
2	20	120	20	40	0	180	50	70
3	100	120	20	120	90	270	50	70
4	0	120	100	120	180	360	50	70

Pour l'angle d'ouverture α, sa valeur est fixée à 30°.

c) Fonction de fitness :

La stratégie utilisée par la PSO pour la recherche de la meilleure solution est de minimiser une fonction de fitness; cette solution devrait être capable d'optimiser la position, l'orientation et la portée de caméras (une par une). Avec cette considération, notre fonction de fitness est la somme des trois fonctions suivantes :

La fonction $ftns_1$ mesure la distance entre les coordonnées de la caméra et le point du rail le plus proche du point à couvrir.

$$ftns_1 = \sqrt{\left(|x - c_x|^2 + |y - c_y|^2\right)} \tag{4.4}$$

La fonction $ftns_2$ calcule les coordonnées (x_1,y_1) et (x_2,y_2) des points P_1 et P_2 respectivement (figure 4.7) pour chaque particule; ensuite, elle mesure la distance entre ces points et le

point à couvrir, comme suit :

$$\begin{aligned} x_1 &= c_x + D.\cos\left(\varphi + \left(\frac{\alpha}{2}\right)\right) \\ y_1 &= c_y + D.\sin\left(\varphi + \left(\frac{\alpha}{2}\right)\right) \\ x_2 &= c_x + D.\cos\left(\varphi - \left(\frac{\alpha}{2}\right)\right) \\ y_2 &= c_y + D.\sin\left(\varphi - \left(\frac{\alpha}{2}\right)\right) \end{aligned} \tag{4.5}$$

Avec:

$D = \sqrt{d_{max}^2 + \frac{a^2}{4}}$

Donc:

$$ftns_2 = \sqrt{(y-y_1)^2 + (x-x_1)^2} + \sqrt{(y-y_2)^2 + (x-x_2)^2} \tag{4.6}$$

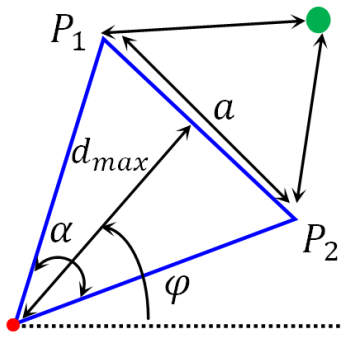


Figure 4.7: Extrimités du FoV.

La fonction $ftns_3$ mesure la distance entre la caméra et le point; ensuite, elle calcule la différence entre cette distance et la longueur de la portée d_{max}.

$$ftns_3 = \left| d_{max} - \sqrt{(x-c_x)^2 + (y-c_y)^2)} \right| \tag{4.7}$$

Donc, notre fonction de fitness globale ftns est donnée comme suit:

$$ftns = ftns_1 + ftns_2 + ftns_3 \tag{4.8}$$

c) Mouvement des particules:

La modification du comportement de chaque particule se base sur la mise à jour de la vitesse et de la position à chaque itération.

La particule combine linéairement ses trois tendances (expliquées auparavant) afin de trouver sa nouvelle vitesse et effectuer son mouvement par conséquent.

Le modèle du voisinage utilisé dans cette application est le G_{best} où toutes les particules de l'essaim ont la même information sociale.

La procédure de recherche sera arrêtée lorsque le nombre courant d'itérations devient supérieur ou égal à la valeur maximale d'itérations (T =100); dans ce cas, la dernière valeur G_{best} est considérée comme une solution au problème.

4.5.2 Résultats de simulation

Les résultats de simulation de la première application, selon trois scénarios différents, sont donnés par les figures suivantes (4.8, 4.9 et 4.10).

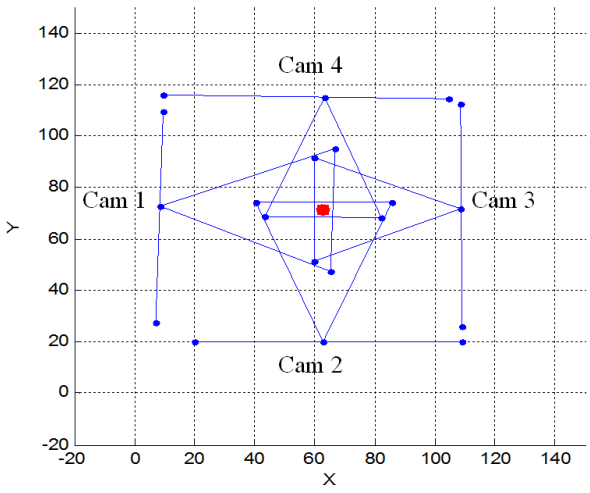


Figure 4.8: Résultats du 1^{er} scénario.

Le premier scénario exprime un seul point couvert par quatre caméras sans la présence d'un obstacle (figure 4.8).

Le deuxième scénario montre le même point mais cette fois avec la présence d'un obstacle qui gène le placement de la caméra N°2 (figure 4.9).

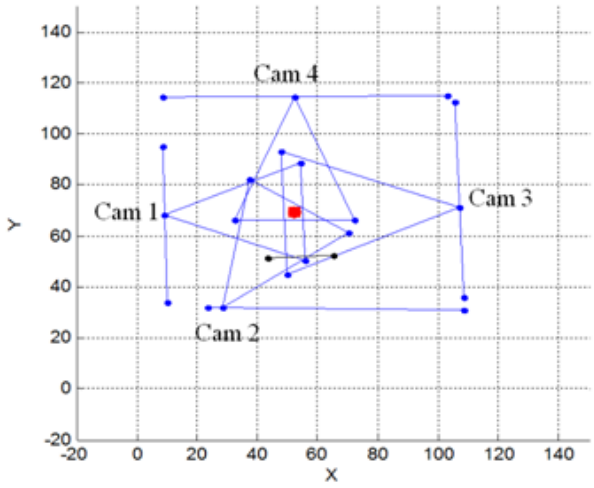


Figure 4.9: Résultats du 2ème scénario.

Enfin, le troisième scénario est un ensemble de points (figure 4.10) qui peuvent être traduits comme un mouvement d'un seul point dans la scène (sans obstacle).

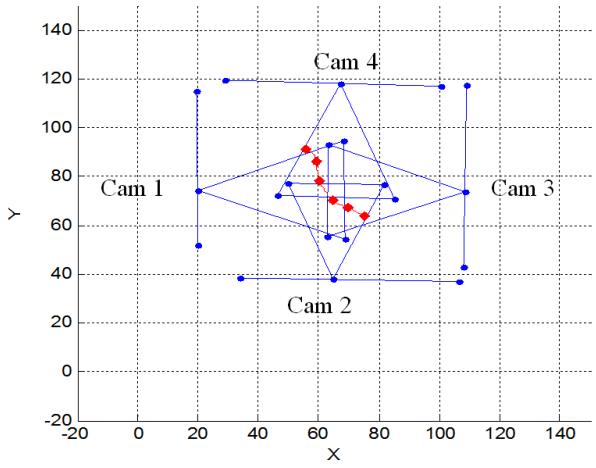


Figure 4.10: Résultats du 3ème scénario.

Les valeurs numériques de ces résultats, sont données sous forme de tableaux (4.3, 4.4 et 4.5) pour les trois scénarios, respectivement.

D'après les résultats obtenus, notre objectif est d'observer un point avec deux caméras au minimum, même avec l'ajout d'obstacles. Le système a convergé à une position qui permet la visibilité du point, non forcément la plus proche de ce point, mais il faut noter que si l'obstacle

Tableau 4.3: Résultats du 1^{er} scénario

Caméra N°	Position [m]		Orientation [°]	Portée [m]
	C_{X1}	C_{X2}		
1	8.4662	72.8588	358.2359	61.6358
2	62.7632	20.0148	89.8894	57.9773
3	108.7282	71.4777	180.0808	51.9683
4	63.4525	115.101	269.053	49.5167

Tableau 4.4: Résultats du $2^{\text{ème}}$ scénario

Caméra N°	Position		Orientation[°]	Portée[m]
	C_{X1}	C_{X2}		
1	9.2381	68.1743	1.5285	48.5859
2	28.6252	31.8843	57.5573	49.8398
3	107.113	71.3598	182.2764	62.1244
4	52.0853	114.687	270.2908	51.3443

Tableau 4.5: Résultats du $3^{\text{ème}}$ scénario

Caméra N°	Position [m]		Orientation[°]	Portée[m]
	C_{X1}	C_{X2}		
1	19.8402	73.9765	0.3768	51.8020
2	64.9935	37.7706	88.9103	40.9920
3	108.410	73.7690	179.3874	48.0777
4	67.2652	118.099	268.0760	49.3583

est très grand, on ne pourrait pas trouver de solutions qui permettent le recouvrement du point (marqueur) dans ce cas.

Ensuite, nous avons considéré un groupe de points; d'après les résultats, presque tous les points sont observés par deux caméras au minimum, ce qui nous conduit à dire que l'objectif est atteint.

Le temps de calcul est proportionnel au nombre de caméras placées, au nombre d'obstacles et au choix d'un point ou un groupe de points. En général, le temps de calcul est très satisfaisant (quelques minutes) et cet avantage revient aux particularités de la PSO.

4.5.3 Convergence du système

La trajectoire de chaque particule est une courbe sinusoïdale amortie (figure 4.11), illustrant la position d'une particule dans l'espace de recherche en fonction du nombre d'itérations, dont l'amplitude et la fréquence sont fonctions du choix des paramètres et des conditions initiales (position et vitesse initiales) [72]. La courbe suivante illustre (comme exemple) la trajectoire d'une particule dans un scénario choisi.

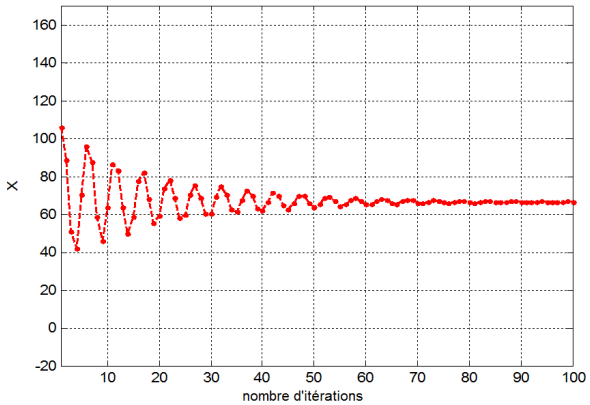


Figure 4.11: Trajectoire d'une particule.

Notre application minimise une fonction de fitness, la figure 4.12 illustre les valeurs de fitness de disposition d'une caméra au cours de l'exécution.

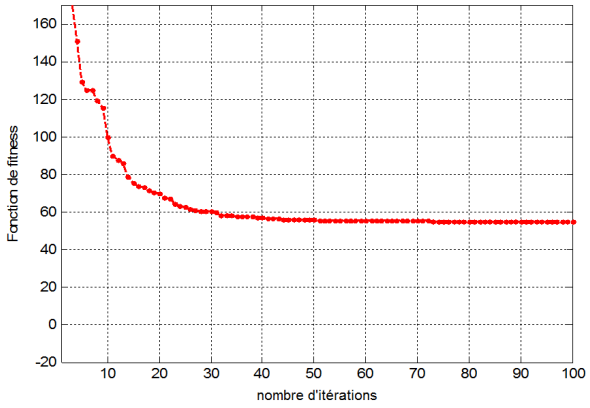


Figure 4.12: Minimisation d'une fonction de fitness.

La convergence du système dépend des paramètres de réglage de la PSO où une modification d'un seul paramètre peut induire un comportement complètement différent du système. Dans cette partie, nous analysons les trajectoires d'une particule sous différentes conditions afin de trouver le bon réglage de ses paramètres d'une manière à obtenir des bons résultats d'optimisation.

4.5.4 Influence des coefficients de confiance

Le choix des coefficients de confiance est très important pour la convergence de l'algorithme, la figure 4.13 illustre les trajectoires d'une particule dans les cas suivants :

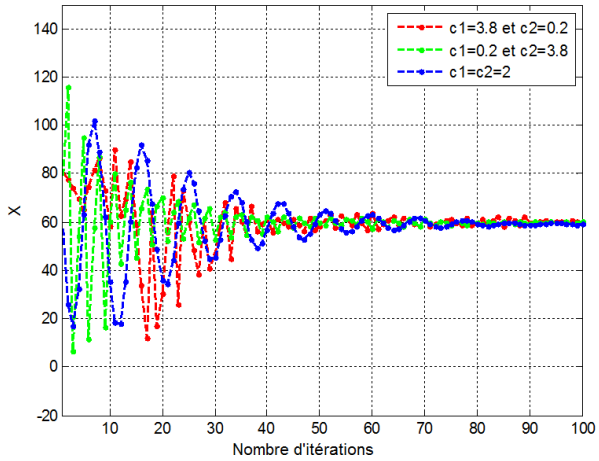


Figure 4.13: Trajectoires d'une particule sous différentes combinaisons entre C_1 et C_2.

- $C_1 \ll C_2$: La particule est influencée par la meilleure position globale, la courbe en

vert affirme cette notion, où dès le départ de l'exécution la particule présente une grande amplitude et fréquence d'oscillation. Elle est vite attirée vers la meilleure solution globale et ses positions sont autour de cette solution.

- $C_1 \gg C_2$: La particule est influencée fortement par sa meilleure position, la courbe en rouge affirme cette notion, où ses positions ne sont pas autour de la solution globale, mais cette tendance a permis aux particules à mieux exploiter la solution.
- $C_1 = C_2$: Toutes les particules sont attirées vers la moyenne entre P_{best_i} et G_{best_i}, la courbe en bleu illustre ce cas.

4.5.5 Influence du coefficient d'inertie

Le choix du coefficient d'inertie w permet de déterminer la qualité de recherche dans l'espace. Généralement il est compris entre [0.4,1.4]. Dans la figure 4.14, nous avons étudié l'influence de trois valeurs possibles de w sur la trajectoire de la particule :

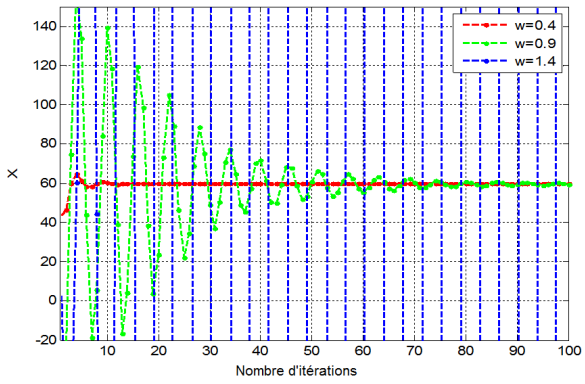


Figure 4.14: Trajectoires d'une particule sous différentes valeurs de w.

- La courbe en rouge illustre le cas d'une petite valeur ($w = 0.4$); cette valeur permet de concentrer la recherche sur un petit espace (exploitation).
- La courbe en bleu illustre le cas d'une grande valeur ($w = 1.4$); cette valeur permet une grande exploration de l'espace, mais avec une telle valeur le processus diverge.
- La courbe en vert illustre le cas d'une valeur médiane ($w = 0.9$); cette valeur permet de trouver un compromis entre l'exploration globale et locale.

Il faut noter que le coefficient d'inertie w peut-être constant ou fonction linéaire décroissante du temps, donnée par l'équation 4.9 :

$$w_{t+1} = w_{max} - \left(\frac{(w_{max} - w_{min})}{T} \right).t, \qquad \text{avec:} \quad w_{max} > w_{min} \tag{4.9}$$

w_{max} et w_{min} sont les valeurs initiale et finale respectivement du facteur d'inertie.

T est le nombre d'itérations maximal.

t est le nombre d'itérations courant.

La figure 4.15 illustre la trajectoire d'une particule avec des valeurs décroissantes de w.

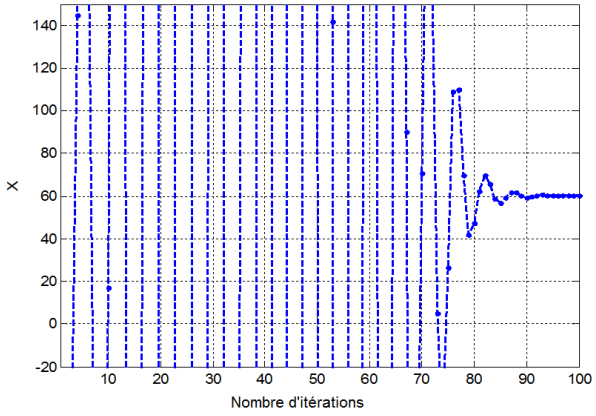


Figure 4.15: Trajectoire d'une particule avec des valeurs décroissantes de w.

D'après cette figure, on peut remarquer que l'utilisation des valeurs décroissantes de w permet au début à la particule de couvrir un grand secteur de l'espace de recherche; par la suite, le facteur w commence à amortir progressivement sa vitesse et il force ainsi la particule à effectuer une recherche locale plus fine.

La figure 4.16 illustre la fonction de fitness avec des valeurs décroissantes de w. Nous pouvons constater qu'avec cette variation de w, nous avons décalé le processus du mode exploration au mode exploitation et nous avons pu éviter la convergence prématurée.

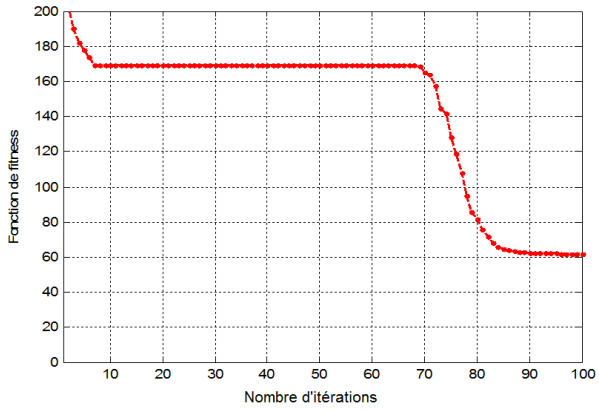


Figure 4.16: Fonction de fitness avec des valeurs décroissantes de w.

4.5.6 Influence de la taille d'essaim

Il est bien évident qu'un nombre élevé de particules a l'avantage d'offrir plus de chance pour converger vers l'optimal au prix d'un long temps de calcul.

La figure 4.17 illustre les fonctions de fitness obtenues pour trois nombres de particules, allouées à la résolution de notre application.

D'après les résultats, notre algorithme basé sur la PSO n'est pas sensible à la taille de l'essaim; la fonction de fitness obtenue avec 20 particules ne diffère pas beaucoup de celle obtenue avec 80 ou 160 particules. Cependant, il y a l'inconvénient de la nécessité d'un long temps de calcul et de convergence.

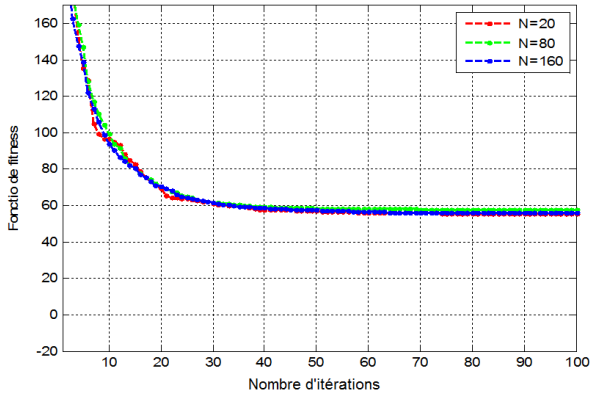


Figure 4.17: Fonctions de fitness avec différentes tailles d'essaim.

Presque toutes les métaheuristiques rencontrent l'enjeu du choix des paramètres de réglage

pour l'adapter au mieux à un problème posé, et sa convergence vers un optimum global ou un optimum local n'est pas assurée; elle est assurée uniquement vers la meilleure position visitée par tout l'essaim; ces inconvénients ont une influence importante sur la robustesse de l'algorithme.

4.6 Application 2 : Recouvrement d'un cube

4.6.1 Introduction

Dans cette application, nous essayerons d'augmenter la difficulté d'optimisation par l'introduction d'un objet (cube) équipé de huit marqueurs à ses extrémités et le déploiement de quatre jusqu'à dix caméras avec plusieurs paramètres réglant leurs placements. Afin, de créer plusieurs scénarios de différents types de trajectoires et d'obstacles. La version standard de la PSO (Standard Particle Swarm Optimization (SPSO)) a atteint ses limites pour une telle application et pour atteindre la solution optimale de placement 2D du réseau de caméras, nous essayerons d'appliquer deux autres variantes (WPSO et CPSO) pour atteindre l'optimum.

4.6.2 Plate-forme expérimentale

Pour cette application, nous avons préparé une plate-forme différente de celle de la première application. Nous présentons d'abord la modélisation de la scène, puis la modélisation du cube et enfin trois différents scénarios de mouvement du cube à l'intérieur de la scène en présence de quelques obstacles.

a) Modélisation de la scène :

La scène est de forme rectangulaire, avec une taille de (18m $\times$ 12m), présentée en figure 4.18. La salle peut être équipée de dix caméras infrarouges 3D au maximum; ces caméras sont soit montées sur des trépieds amovibles ou montées sur des supports fixés au mur. La scène est simulée comme suit:

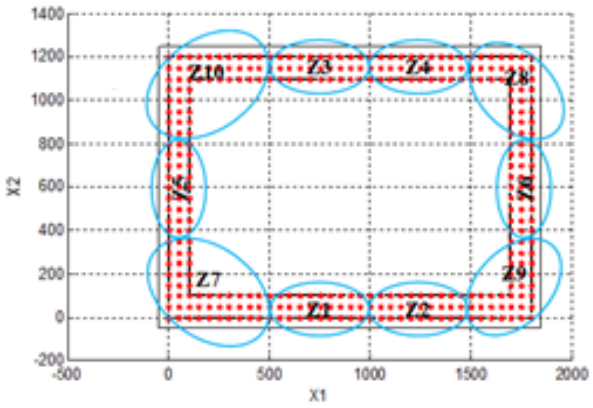


Figure 4.18: Organisation des emplacements pour les dix caméras considérées (une seule caméra par zone).

Les caméras sont placées sur les bords de la salle, où les points rouges représentent les régions possibles pour le placement des caméras. Dans notre cas, cette région est divisée en dix zones (de Z1 jusqu'à Z10). Une zone représente l'intervalle de placement d'une seule caméra. Cette plateforme est conçue selon une salle d'expériences du système MoCap disponible au laboratoire LAMIH de l'Université de Valenciennes en France.

b) Modélisation du cube :

Un cube transparent (de dimension 50cm × 50cm, figure 4.19(a)) se déplace aléatoirement dans la scène (figure 4.19(b)). Aux extrémités de ce cube, sont placés huit marqueurs (quatre en haut et quatre en bas). Dans le cas 3D, les images obtenues par les caméras sont binaires (huit points blancs sur un fond noir, figures 4.19(c) et 4.19(d), tracées à l'aide du logiciel gratuit Mokka). Ces deux dernières figures représentent une reconstruction 3D d'un cube fourni en fichier *.C3d par le LAMIH.

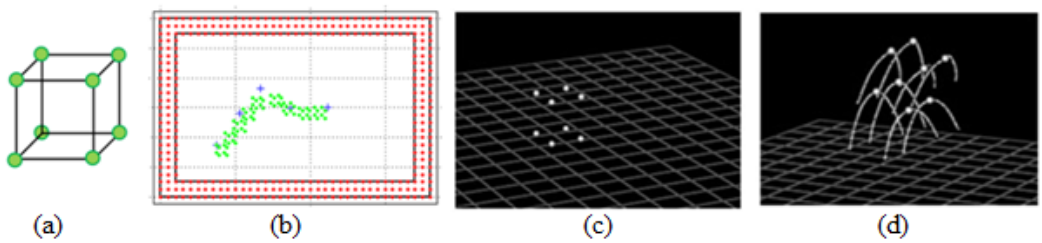


Figure 4.19: Mouvement du cube avec une trajectoire aléatoire.

c) Scénarios proposés :

Nous proposons trois scénarios différents pour le mouvement du cube à l'intérieur de la salle avec la présence d'obstacles pour tester la robustesse de l'algorithme.

- *Premier scénario:* Le cube se déplace en une ligne droite entre deux obstacles (figure 4.20(a)).
- *Deuxième scénario:* Le cube se déplace en une ligne droite vers un grand obstacle (mur) (figure 4.20(b)).
- *Troisième scénario:* Le cube se déplace entre deux piliers (figure 4.20(c)).

Chaque scénario est une séquence de vidéo divisée en plusieurs images. Dans notre cas, le nombre de trames diminu ou augmente avec la vitesse du déplacement du cube.

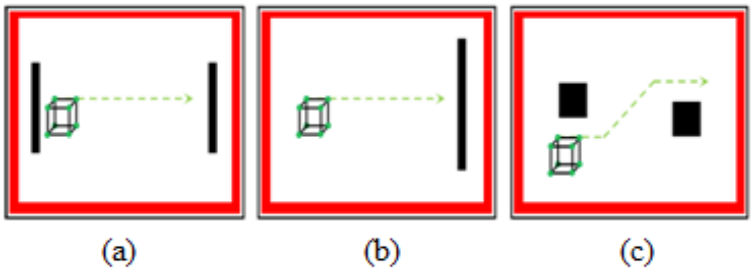


Figure 4.20: Trois scénarios différents.

4.6.3 Contraintes de la fitness

a) Contraintes d'optimisation du placement 2D :

Pour formuler l'équation de la fonction de fitness, nous devons définir d'abord un certain nombre de variables. Premièrement, le tenseur $I_{ij,k}$ qui décide si le marqueur j est à l'intérieur du champ de vision de la caméra C^i durant la trame k ou non, comme suit:

$$I_{ij,k} = \begin{cases} 1 & \text{Si le marqueur } j \text{ est à l'intérieur de } C^i \\ 0 & \text{Ailleurs} \end{cases} \tag{4.10}$$

$I_{ij,k}$ est un tenseur de dimension $[N_C \times N_t \times N_f]$, Avec: $j = 1,\cdots,N_t$, $i = 1,\cdots,N_C$ et $k = 1,\cdots,N_f$.

N_C: Nombre de caméras, avec : $N_C = 1,\cdots,10$.

N_t: Nombre de marqueurs, pour le cube: $N_t = 8$.

N_f: Nombre de trames.

Le marqueur j est jugé visible ou non par la caméra C^i en présence d'un obstacle l entre eux durant la trame k, si la condition 4.11 est satisfaite. Le tenseur I devient:

$$I_{ij,k} = \begin{cases} 1 & \text{Si le marqueur } j \text{ est visible à } C^i \text{ en présence de } l \text{ durant } k \\ 0 & \text{Ailleurs} \end{cases} \tag{4.11}$$

La condition principale dans notre problème d'optimisation est que chaque marqueur j ne doit être vu que par C_{seuil} caméras dans chaque image à la fois; ensuite, nous additionnons le tenseur précédent et nous obtenons:

$$A_{j,k} = \sum_{i=1}^{N_c} I_{i,j,k} \tag{4.12}$$

Avec:

$A_{j,k}$ est un tenseur de taille $[N_t \times 1 \times N_f]$, qui représente le nombre d'apparition de chaque marqueur j durant toutes les trames, c'est à dire durant tout le mouvement.

Finalement:

$$OF_{j,k} = \begin{cases} 1 & \text{Si } A_{j,k} \geq C_{seuil} \\ 0 & \text{Ailleurs} \end{cases} \tag{4.13}$$

Où:

$OF_{j,k}$ est une matrice de taille $[N_t \times N_f]$, qui présente la condition que chaque marqueur j doit être visible au moins par les C_{seuil} caméras (condition de reconstruction) durant tout le mouvement.

b) Fonctions de pénalité :

Avant de calculer la fitness (qualité) de la solution proposée, il faut la vérifier. La solution est pénalisée par la fonction $Pnlt1$ (équation 4.14); lorsqu'une caméra est située à l'extérieur de la scène.

$$Pnlt1 = \begin{cases} 1 & \text{Si } C_{X1_{min}} \leq C^i_{x1} \leq C_{X1_{max}} \\ & \text{et Si } C_{X2_{min}} \leq C^i_{x2} \leq C_{X2_{max}} \\ 0 & \text{Ailleurs} \end{cases} \tag{4.14}$$

Où:

$C_{X1_{min}}$ et $C_{X1_{max}}$ sont les extrémités en abscisses de la salle.

$C_{X2_{min}}$ et $C_{X2_{max}}$ sont les extrémités en ordonnées de la salle.

La deuxième fonction de pénalité $Pnlt2$ est liée à la condition d'angle critique donnée par l'équation (4.3). Cette pénalité est exprimée selon l'équation 4.15:

$$Pnlt2 = \begin{cases} +\lambda & \text{Si} \quad \theta_{i,i+1} > \theta_{min} \\ -\lambda & \text{Ailleurs} \end{cases} \tag{4.15}$$

Avec: $\lambda = 0.5$ et θ_{min} =20°, des valeurs expérimentales.

c) Fonction de fitness :

L'équation adéquate adoptée pour notre application est donnée par:

$$Ftnes = (ftn1 + Pnlt2).Pnlt1 \tag{4.16}$$

Avec:

$$ftn1 = \frac{1}{N_t}\frac{1}{N_f}\sum_{j=1}^{N_t}\sum_{k=1}^{N_f} OF_{j,k} \tag{4.17}$$

4.6.4 Optimisation et résultats

a) Processus d'optimisation :

La simulation commence par le déploiement de quatre caméras (en même temps); le choix de la trajectoire du cube et la pose d'obstacles. La phase d'optimisation ne s'arrête tant que le

nombre maximal des itérations n'est pas atteint ($CA1 \geq T$, première condition d'arrêt). En cas de recouvrement complet ($CA2 = 100\%$, deuxième critère d'arrêt), les résultats sont affichés, sinon le nombre de caméras augmente et la simulation redémarre, tout en conservant les mêmes paramètres de la PSO.

L'organigramme de la figure 4.21 donne un aperçu sur les démarches du processus d'optimisation.

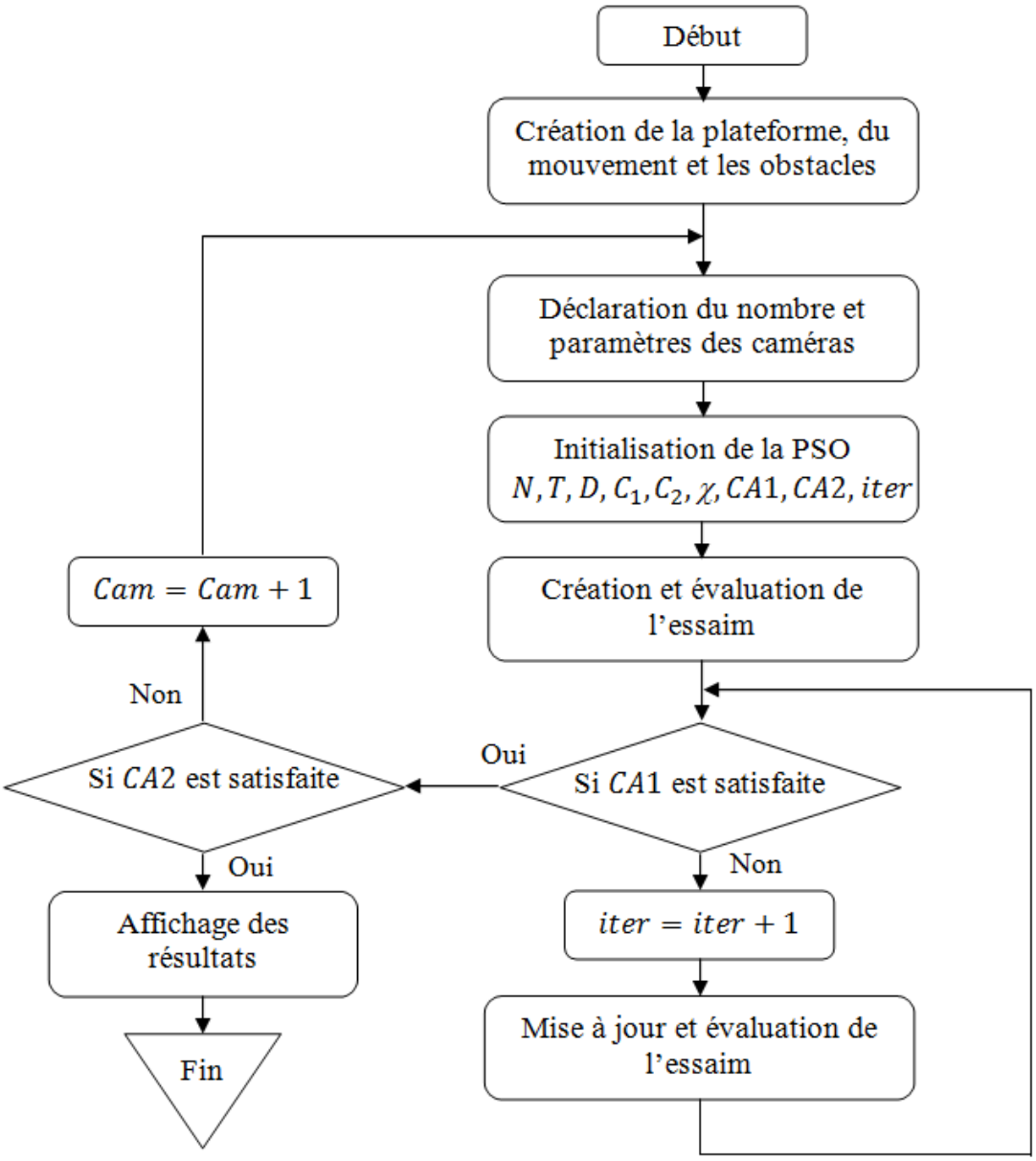


Figure 4.21: Organigramme de la 2ème application (Recouvrement d'un cube).

b) Contribution :

Dans les figures 4.22, 4.23 et 4.24, nous présentons la solution finale qui donne les positions optimales et l'orientation du réseau de caméras pour chacun des scénarios.

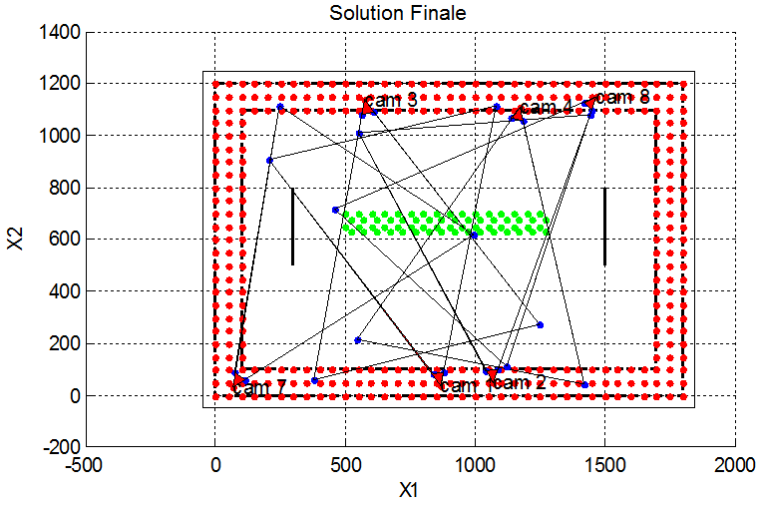


Figure 4.22: Résultats du 1^{er} scénario.

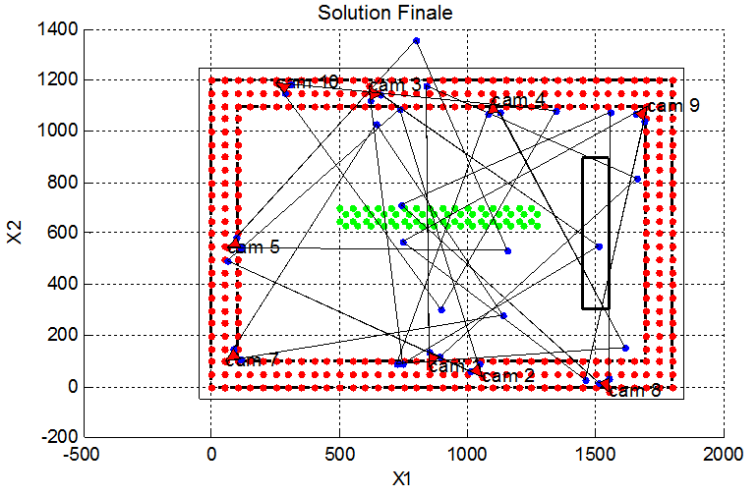


Figure 4.23: Résultats du $2^{\text{ème}}$ scénario.

Dans cette deuxième application, la technique standard (SPSO, voir 3.4.2) et la technique WPSO (voir 3.4.7.E) n'ont pas abouti à des résultats satisfaisants. L'utilisation de la troisième technique CPSO (voir 3.4.8) a donné de meilleurs résultats par l'introduction d'une petite mod-

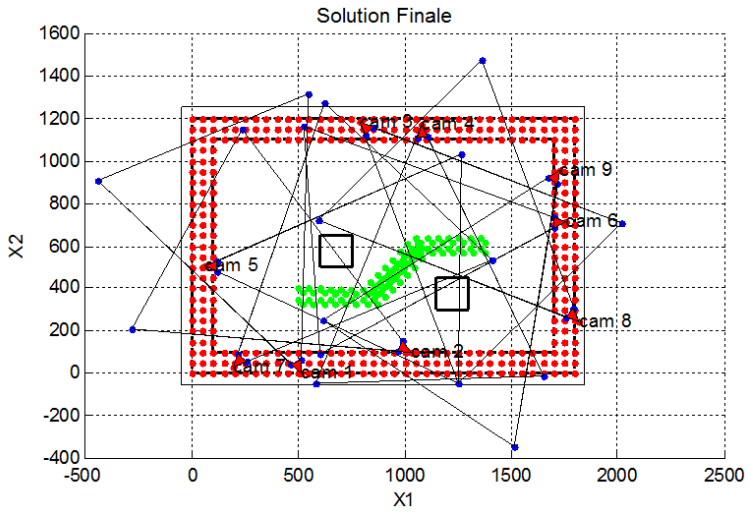


Figure 4.24: Résultats du 3ème scénario.

ification sur les valeurs des facteurs cognitifs C_1 et C_2 au cours des itérations (figure 4.25) par inspiration de l'équation 3.8. Les équations suivantes montrent cette modification:

$$C_1 = \frac{(c_{min} - c_{max}).t}{T} + c_{max} \tag{4.18}$$

$$C_2 = \frac{(c_{max} - c_{min}).t}{T} + c_{min} \tag{4.19}$$

Avec :

$c_{min} = 0.6$ et $c_{max} = 3.5$.

t: Itération courante.

T: Nombre d'itérations maximal.

Au début de la simulation, cette contribution favorise le comportement d'exploration des particules. Le processus est inversée au fur et à mesure que le nombre d'itérations augmente pour favoriser le comportement d'exploitation; ainsi, à la fin de la simulation l'algorithme exploite mieux les résultats obtenus durant la phase d'exploration.

c) Résultats de la simulation:

Nous pouvons observer en analysant les figures (4.22, 4.23 et 4.24), que chaque marqueur est couvert par au moins $C_{seuil} = 3$ caméras.

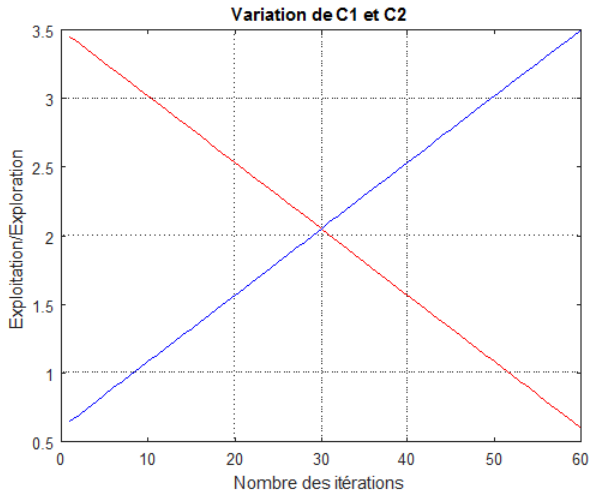


Figure 4.25: Modification de C_1 et C_2 durant les itérations.

L'étape de reconstruction en 3D peut être jugée possible si le marqueur est capté par trois caméras ($C_{seuil} = 3$), ainsi le pourcentage de recouvrement peut atteindre la valeur 100%. Cette étape (décrite dans le premier chapitre) utilise deux images au minimum pour calculer les coordonnées 3D d'un marqueur à partir de ses coordonnées 2D.

Remarque: Dans ce qui suivra, nous présenterons les résultats obtenus pour le premier scénario.

L'allure de la fonction de fitness présentée en figure 4.26, représente l'évaluation de la recherche de la solution optimale dans le premier scénario. Cette fonction exprime le nombre de trames (8 trames maximum pour ce scénario) où tous les marqueurs sont couverts par trois caméras. D'un côté, l'évolution de la courbe montre bien que c'est un problème de maximisation. D'un autre côté la courbe monte rapidement au cours des premières itérations ($\sim$ 5 itérations) jusqu'à $\sim$ 6 trames ce qui exprime une bonne exploration, puis une évolution plus lente (jusqu'à l'itération 40) qui exprime l'exploitation de la solution obtenue dans la phase d'exploration. Enfin, la stabilité de la courbe (jusqu'à l'itération 61) qui assure la décision de la solution optimale.

La figure 4.27, montre l'optimisation des positions des huit caméras déployées (par différentes couleurs). Nous remarquerons bien que les caméras N°5 et N°6 sont inutilisables à cause des obstacles et sont retirées de la configuration finale (figure 4.22).

La courbe de la figure 4.28, vérifie que nous avons obtenu le placement optimal pour

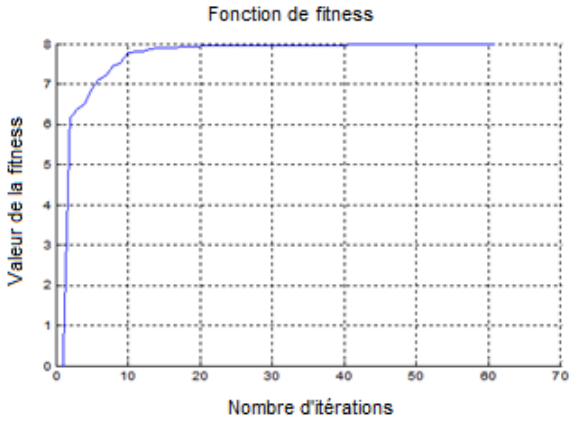


Figure 4.26: Evaluation de la fonction de fitness.

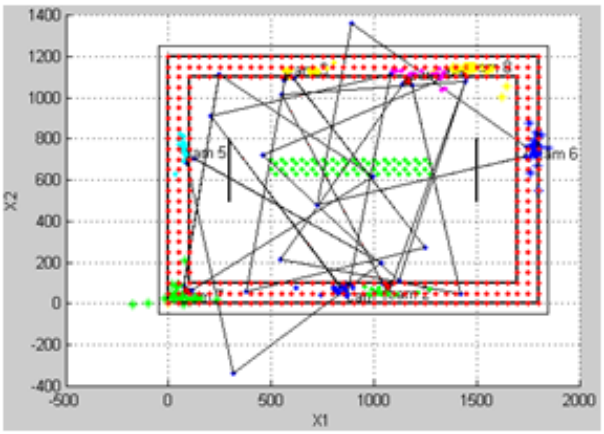


Figure 4.27: Mouvement des particules.

cette situation. Elle présente le recouvrement des marqueurs (en pourcentage %) dans chaque itération de l'algorithme. La formule utilisée pour calculer le pourcentage est donnée comme suit:

$$CP_{[\%]} = \frac{1}{N_t}.\frac{1}{N_f}.ftn1.100 \tag{4.20}$$

La courbe de la figure 4.29, représente la valeur d'erreur de recouvrement par trames, qui exprime le nombre de trames où les marqueurs ne sont pas recouverts par trois caméras. Nous remarquons aussi, l'annulation de l'erreur à la $24^{ème}$ itération puis la perte de cette solution, mais l'algorithme continue sa progression jusqu'à ce que l'erreur devienne nulle et stable.

Une représentation globale de nos résultats est montrée par un histogramme groupé de la

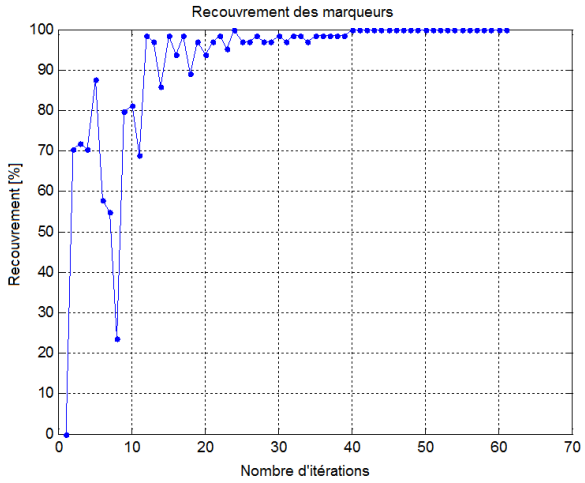


Figure 4.28: Evaluation du recouvrement des marqueurs.

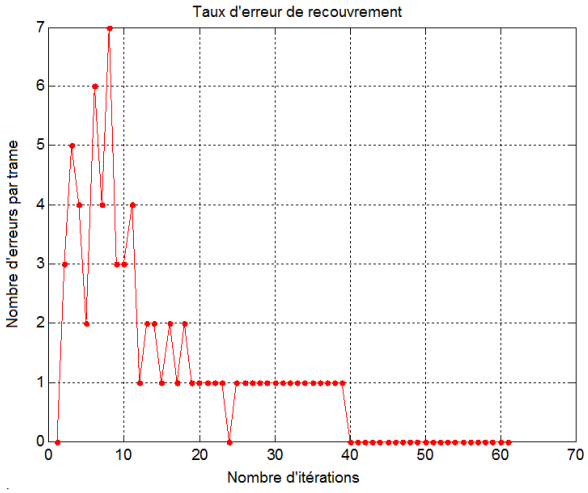


Figure 4.29: Evaluation de l'erreur de recouvrement.

figure 4.30. Le 2ème scénario présente le temps de cacul (Temps Calc [sec]) le plus court, car il n'y a qu'un seul obstacle et seule la 5ème caméra qui n'est pas utilisée. Dans le premier scénario, le temps augmente car il y a deux obstacles (5ème et 6ème caméras ne sont pas utilisées). Le dernier scénario, présente un long temps de calcul à cause du nombre de trames (19 trames) et de la complexité des obstacles; mais en retour, toutes les caméras déployées sont utilisées et le nombre d'itérations est acceptable. Dans tous les scénarios, le nombre de particules $N = 30$, $C_{seuil} = 3$, et le même pourcentage de recouvrement (objectif atteint).

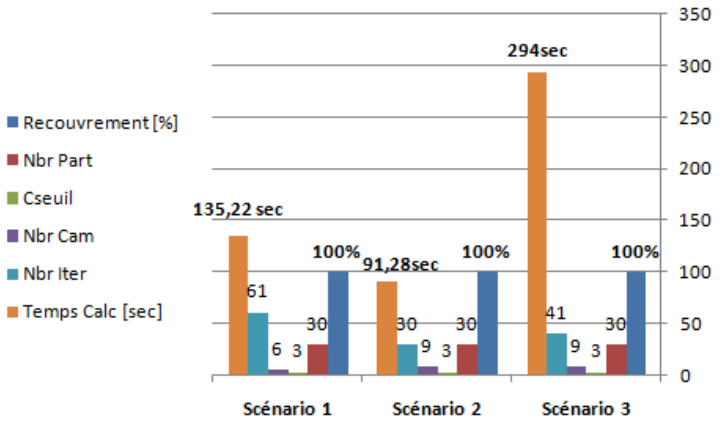


Figure 4.30: Résultats globals pour les trois scénarios.

4.7 Conclusion

D'après les résultats obtenus dans la première application, nous pouvons constater que notre algorithme permet d'optimiser la position, l'orientation et la portée de chaque caméra pour l'observation du point (marqueur).

Il apparaît clairement que notre algorithme a été développé (utilisation de la WPSO avec ω vraiable) pour résoudre un problème de quatre dimensions ($D = 4$), et cela n'a pas empêcher l'obtention de résultats acceptables (pour les scénarios proposés : un seul point, un point et un obstacle et mouvement d'un point) en assurant une convergence très rapide avec une population plus réduite et un nombre réduit d'itérations.

Dans la deuxième application, nous avons appliqué une variation des coefficients cognitifs C_1 et C_2 de la technique CPSO (pour un passage fluide entre l'exploration et l'exploitation). Nous avons utilisé cette technique dans un système MoCap pour optimiser le placement d'un réseau de caméras dans un simple environnement 2D (pour trois différents scénarios).

Il existe certains travaux d'optimisation de couverture utilisant la technique PSO dans la littérature, mais il s'agit de couvrir une surface pour des tâches de monitoring comme dans [69] ou pour couvrir un espace de travail comme dans [73].

D'un autre côté, notre travail consiste à couvrir des points particuliers sous la condition d'une reconstruction possible du mouvement en 3D. Nous proposons dans le chapitre suivant, l'utilisation d'un champ de vision 3D pour augmenter la dimension D du problème d'optimisation

(position 3D (x, y, z), deux poses (φ_v et φ_h), deux profondeurs (d_{min} et d_{max}), deux angles d'ouverture (α_v et α_h) et le déploiement de quatre à dix caméras).

5

Optimisation du placement 3D d'un réseau de caméras par PSO et RFS

5.1 Introduction

Dans le chapitre précédent nous avons optimisé le placement 2D d'un réseau de caméras pour le recouvrement d'un mouvement virtuel d'un cube qui utilise huit marqueurs. Le recouvrement de ces marqueurs a été effectué sous deux conditions: la première est que chaque marqueur doit être recouvert par deux caméras durant chaque trame du mouvement et la deuxième condition est qu'il soit recouvert par trois caméras à la fois. Ici, nous avons considéré que la hauteur de toutes les caméras est unifiée pour que le résultat soit acceptable. Ce cas a été étudié, en simplifiant la complexité du système MoCap, pour se familiariser avec l'optimisation par PSO pour un tel système.

Dans le cas réel, un système MoCap est un système tridimensionnel, c'est à dire que le champ de vision des caméras infrarouges et le mouvement d'un objet ou d'un humanoïde à l'intérieur de la scène doit être pris dans son cas naturel en 3D.

Dans ce chapitre, nous essayerons de résoudre le problème d'optimisation de placement 3D d'un réseau de caméras pour un mouvement le long de la diagonale de la salle du même cube. Pour cette raison, nous utiliserons deux variantes de la métaheuristique PSO (CPSO et MCPSO) et une autre métaheuristique appelée RFS.

5.2 Matériels et logiciels

Le logiciel de calcul utilisé pour nos simulations est le logiciel MATLAB sous la version R2016b.

La machine utilisée pour nos simulations :

Un PC de bureau équipé d'un processeur Inter(R) Core(TM) i5-3550 CPU avec une horloge de 3.30GHz, une mémoire RAM de 16 Go, un système d'exploitation windows 7 professionnel de 64 bits.

5.3 Modélisation du champ de vision 3D d'une caméra

5.3.1 Forme du champ de vision

Le champ de vision 3D d'une caméra est modélisé sous forme d'une pyramide selon la figure 5.1. Mais la base de cette pyramide n'est pas vraiment plate. Nous la modélisons ici par une autre pyramide pour que la somme des hauteurs des ces deux pyramides soit égale à d_{max} .

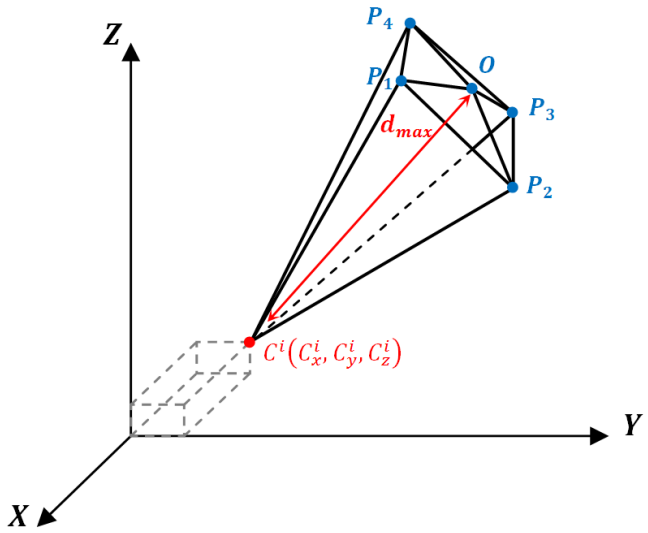


Figure 5.1: Modélisation en 3D du champ de vision.

Les cinq extrémités formant cette pyramide sont les points (O,P_1,P_2,P_3,P_4), dont les coordonnées en 3D sont calculées à partir de la position et la pose de la caméra $C^i(C_x^i, C_y^i, C_z^i)$, selon les équations (5.1 jusqu'à 5.5), comme suit:

a) Le point $O(O_x, O_y, O_z)$ ***:***

$$\begin{aligned} O_x &= C_x^i + \left(d_{max}\cos(\varphi_h^i)\cdot\cos(\varphi_v^i)\right) \\ O_y &= C_y^i + \left(d_{max}\sin(\varphi_h^i)\right) \\ O_z &= C_z^i + \left(d_{max}\sin(\varphi_v^i)\right) \end{aligned} \tag{5.1}$$

b) Le point $P_1(x_1, y_1, z_1)$ ***:***

$$
\begin{aligned}
x_1 &= C_x^i + \left(d_{max} \cdot \cos(\varphi_h^i) + \left| \frac{\alpha_h^i}{2} \right| \cdot \cos(\varphi_v^i) - \left| \frac{\alpha_v^i}{2} \right| \right) \\
y_1 &= C_y^i + \left(d_{max} \cdot \sin(\varphi_h^i) + \left| \frac{\alpha_h^i}{2} \right| \right) \\
z_1 &= C_z^i + \left(d_{max} \cdot \sin(\varphi_v^i) - \left| \frac{\alpha_v^i}{2} \right| \right)
\end{aligned} \tag{5.2}
$$

d) Le point $P_2(x_2, y_2, z_2)$ ***:***

$$
\begin{aligned}
x_2 &= C_x^i + \left(d_{max} \cdot \cos(\varphi_h^i) - \left| \frac{\alpha_h^i}{2} \right| \cdot \cos(\varphi_v^i) - \left| \frac{\alpha_v^i}{2} \right| \right) \\
y_2 &= C_y^i + \left(d_{max} \cdot \sin(\varphi_h^i) - \left| \frac{\alpha_h^i}{2} \right| \right) \\
z_2 &= C_z^i + \left(d_{max} \cdot \sin(\varphi_v^i) - \left| \frac{\alpha_v^i}{2} \right| \right)
\end{aligned} \tag{5.3}
$$

e) Le point $P_3(x_3, y_3, z_3)$ ***:***

$$
\begin{aligned}
x_3 &= C_x^i + \left(d_{max} \cdot \cos(\varphi_h^i) + \left| \frac{\alpha_h^i}{2} \right| \cdot \cos(\varphi_v^i) + \left| \frac{\alpha_v^i}{2} \right| \right) \\
y_3 &= C_y^i + \left(d_{max} \cdot \sin(\varphi_h^i) + \left| \frac{\alpha_h^i}{2} \right| \right) \\
z_3 &= C_z^i + \left(d_{max} \cdot \sin(\varphi_v^i) + \left| \frac{\alpha_v^i}{2} \right| \right)
\end{aligned} \tag{5.4}
$$

f) Le point $P_4(x_4, y_4, z_4)$ ***:***

$$
\begin{aligned}
x_4 &= C_x^i + \left(d_{max} \cdot \cos(\varphi_h^i) - \left| \frac{\alpha_h^i}{2} \right| \cdot \cos(\varphi_v^i) + \left| \frac{\alpha_v^i}{2} \right| \right) \\
y_4 &= C_y^i + \left(d_{max} \cdot \sin(\varphi_h^i) - \left| \frac{\alpha_h^i}{2} \right| \right) \\
z_4 &= C_z^i + \left(d_{max} \cdot \sin(\varphi_v^i) + \left| \frac{\alpha_v^i}{2} \right| \right)
\end{aligned} \tag{5.5}
$$

Avec :

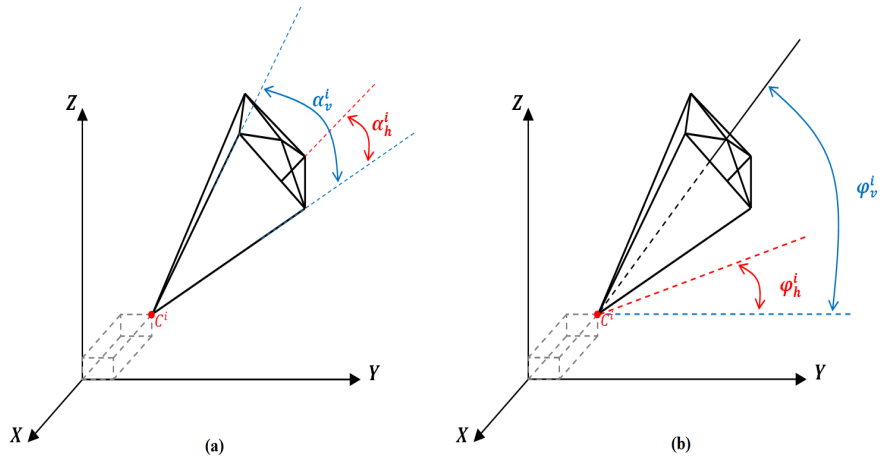


Figure 5.2: (a) Angles d'ouverture (b) Angles de pose de la caméra.

d_{max}: la portée maximale du champ de vision;

α_h^i et α_v^i: les angles d'ouverture horizontale et verticale respectivement du FoV de la caméra C^i;

φ_h^i et φ_v^i: les angles de pose horizontale et verticale respectivement de la caméra C^i.

Remarque: Tous les angles sont prisent en degré (°).

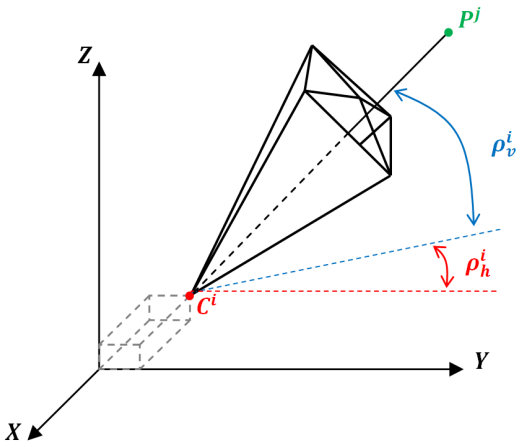


Figure 5.3: Angles de pose du point P^j.

5.3.2 Vérification du recouvrement

Pour vérifier le recouvrement du point P^j par une caméra C^i, il faut calculer les poses horizontale ρ_h et verticale ρ_v de ce point dans l'espace, afin de les comparées avec celles de la

caméra considérée (φ_h^i et φ_v^i).

a) Calcul de la pose horizontale ρ_h ***du point*** P^j ***:***

La pose horizontale peut être calculée comme suit:

$$\rho_h = \arctan\left(\frac{P_y^j - C_y^i}{P_x^j - C_x^i}\right) \tag{5.6}$$

L'angle de la pose horizontale ρ_h est modifié selon la condition d'orientation du segment (C^i, P^j), (voir figure 5.4). Les équations suivantes donnent les modifications en détails:

$$\rho_h = \begin{cases} \rho_h & Si \quad P_y^j > C_y^i \quad et \quad P_x^j > C_x^i \\ 180 - |\rho_h| & Si \quad P_y^j > C_y^i \quad et \quad P_x^j < C_x^i \\ |\rho_h| - 180 & Si \quad P_y^j < C_y^i \quad et \quad P_x^j < C_x^i \\ -|\rho_h| & Si \quad P_y^j < C_y^i \quad et \quad P_x^j > C_x^i \end{cases}$$

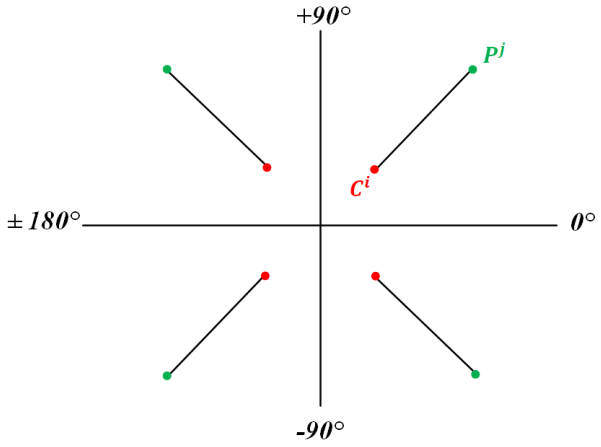


Figure 5.4: Orientation du segment (C^i, P^j).

b) Calcul de la distance entre C^i ***et*** P^j ***:***

La distance l_{ij} entre une caméra C^i et un point P^j est donnée par l'équation (5.7) suivante:

$$l_{ij} = \sqrt{(P_x^j - C_x^i)^2 + (P_y^j - C_y^i)^2 + (P_z^j - C_z^i)^2} \tag{5.7}$$

c) Calcul de la pose horizontale ρ_v ***du point*** P^j***:***

L'angle de pose verticale ρ_v est calculé comme suit:

$$\rho_v = \arcsin\left(\frac{P_z^j - C_z^i}{l_{ij}}\right) \tag{5.8}$$

d) Condition de recouvrement :

Le point P^j est dit recouvert par la caméra C^i ou non, si la condition V_{ij} est vérifiée par l'équation (5.9) comme suit:

$$V_{ij} = \begin{cases} 1 & \text{Si} \quad V_{ij}^1.V_{ij}^2.V_{ij}^3 = 1 \\ 0 & Ailleurs \end{cases} \tag{5.9}$$

Avec:

$$V_{ij}^1 = \begin{cases} 1 & \text{Si } d_{min} \leq l_{ij} \leq d_{max} \\ 0 & \text{Ailleurs} \end{cases}$$

$$V_{ij}^2 = \begin{cases} 1 & \text{Si } \rho_h \leq \left(\varphi_h + \frac{\alpha_h}{2}\right) \text{ et } \rho_h \leq \left(+\varphi_h - \frac{\alpha_h}{2}\right) \\ 0 & \text{Ailleurs} \end{cases}$$

$$V_{ij}^3 = \begin{cases} 1 & \text{Si } \rho_v \leq \left(\varphi_v + \frac{\alpha_v}{2}\right) \text{ et } \rho_v \leq \left(+\varphi_v - \frac{\alpha_v}{2}\right) \\ 0 & \text{Ailleurs} \end{cases}$$

5.3.3 Modélisation du FoV sous MATLAB

A l'aide des équations précédentes (de 5.1 jusqu'à 5.9), nous avons modélisé le champ de vision d'une caméra FoV sous MATLAB (voir figure 5.5) et nous pouvons vérifier si n'importe quel point P^j est recouvert par n'importe quelle caméra C^i ou non.

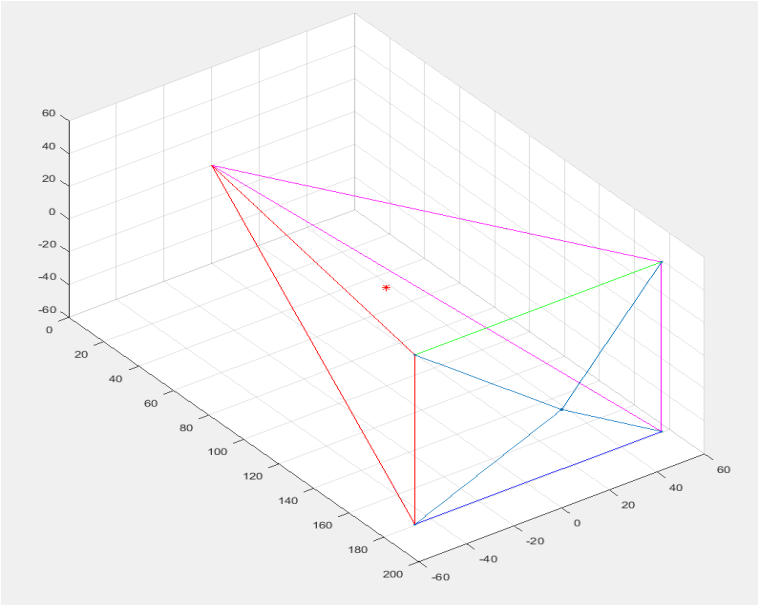


Figure 5.5: Modélisation du FoV sous MATLAB.

5.4 Espace de recherche

L'espace de recherche définit les intervalles des valeurs numériques pour les paramètres à optimiser. La position de chaque caméra est limitée par la longueur ($0 \leq X \leq 18m$), la largeur ($0 \leq Y \leq 12m$) et la hauteur ($0 \leq Z \leq 3m$) de la salle. Aussi, la pose de chaque caméra est limitée par les deux angles vertical et horizontal (-20° $\leq \varphi_v \leq$ +20° et -180° $\leq \varphi_h \leq$+180°), respectivement. L'intervalle de l'angle de pose verticale φ_v est choisi d'une façon à ne pas avoir une image inversée de haut en bas, tandis que φ_h permet un balayage horizontal complet. Les angles d'ouverture verticale α_v et horizontale α_h ainsi que les portées maximale d_{max} et minimale d_{min} sont limitées selon des valeurs données pour avoir une bonne résolution de capture.

Par conséquent, chaque paramètre est limité par sa valeur maximale et minimale à ne pas dépasser. L'espace de recherche pour notre application est donné en détails suivant le tableau 5.1.

Tableau 5.1: Espace de recherche (limites des paramètres)

Caméra N°	**X [cm]**		**Y [cm]**		**Z[cm]**		d_{max} **[cm]**		d_{min} **[cm]**		α_h **[]**		α_v **[°]**		φ_h **[°]**		φ_v **[°]**	
	min	*max*	*min*	*max*	*min*	*max*	*min*	*max*	*min*	*max*	*min*	*max*	*min*	*max*	*min*	*max*	*min*	*max*
1	400	900	0	100	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
2	900	1400	0	100	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
3	400	900	1100	1200	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
4	900	1400	1100	1200	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
5	0	100	400	800	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
6	1700	1800	400	800	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
7	0	400	0	400	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
8	1400	1800	0	400	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
9	1400	1800	800	1200	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20
10	0	400	800	1200	0	300	600	1000	120	200	30	40	30	40	-180	180	-20	20

5.5 Nouvelle contribution de la PSO

Nous avons modifié les équations des coefficients cognitifs C_1 et C_2 selon les équations (5.10 et 5.11) suivantes:

$$C_1 = \left(\frac{(c_{max} - c_{min}).t}{T} \right) + c_{min} \tag{5.10}$$

$$C_2 = \left(\frac{(c_{min} - c_{max}).t}{T} \right) + c_{max} \tag{5.11}$$

Avec:

$c_{max} = 3.5$ et $c_{min} = 0.6$.

Pour tester le comportement local et global des particules, nous avons utilisé une fonction de test F_{test} pour $(N = 20, T = 200, D = 2)$ afin d'évaluer ce comportement. La fonction F_{test} utilisée est :

$$F_{test} = (x + 1.42)^2 + (y + 0.8)^2 \tag{5.12}$$

Les figures suivantes (5.6(a) et 5.6(b)) montrent bien le comportement des particules pour la recherche d'une solution locale où la couleur en rouge représente le début de la simulation et en bleu la fin de la simulation:

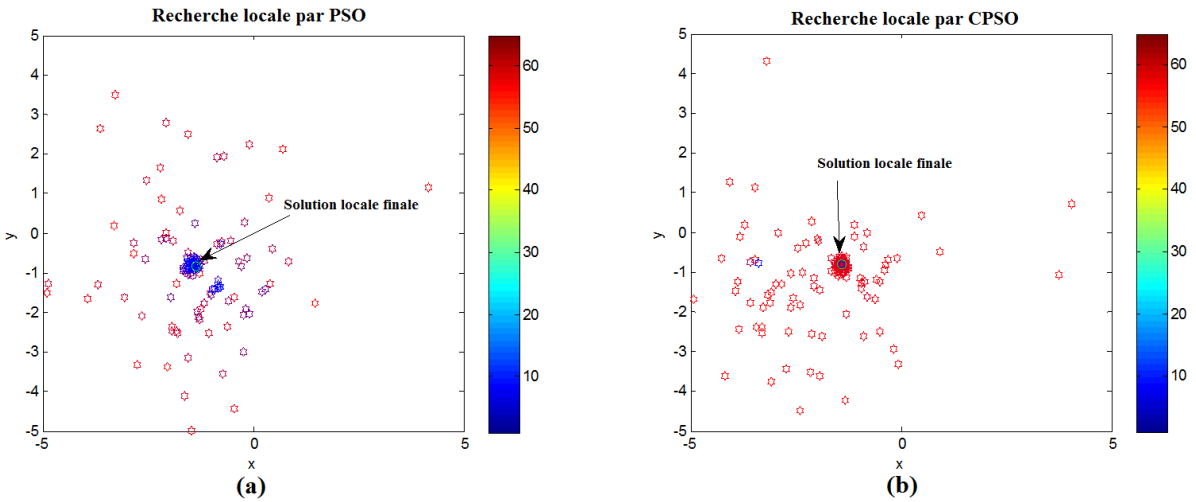


Figure 5.6: Comparaison entre la PSO et la CPSO pour une recherche locale.

D'après la figure 5.6(a), nous remarquons que les particules sont éloignées de la solution optimale (plus dispersées), puis ces particules (en bleu et en violet) se rapprochent lentement vers la solution local optimale (cercle en bleu courtois). Par contre, la majorité des particules dans la figure 5.6(b), sont plus regroupées autour de la solution optimale finale. Ici, nous pouvons dire que les particules covergent rapidement (par rapport à la première) et elles sont plus concentrées autour de la solution locale optimale. Dans ce cas, nous pouvons conclure que la CPSO est meilleure que la PSO pour la recherche locale.

Pour voir le comportement des particules pour la recherche globale, nous pouvons analyser les figures 5.7(a) et 5.7(b).

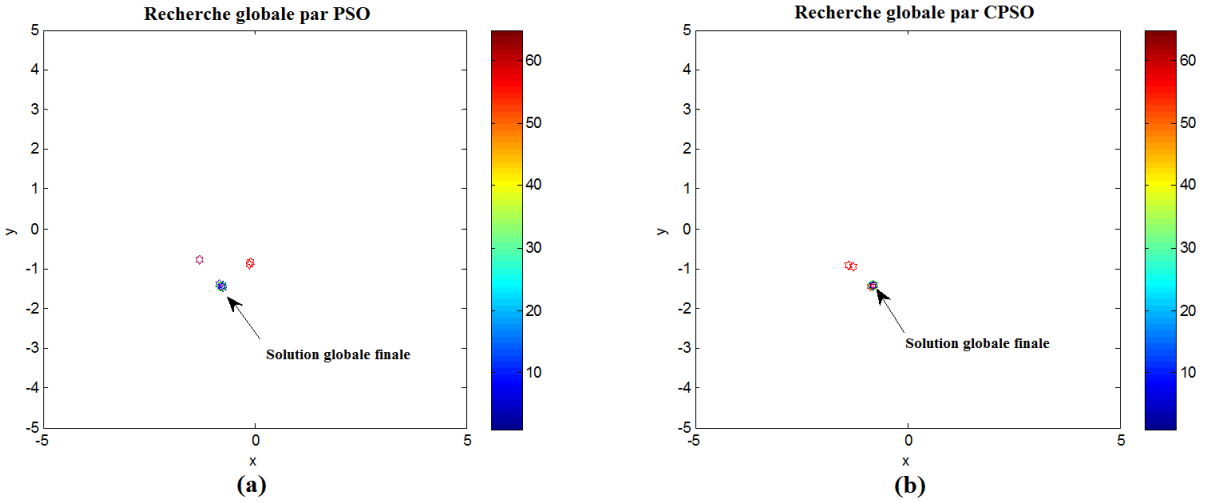


Figure 5.7: Comparaison entre la PSO et la CPSO pour une recherche globale.

Pour la recherche de la solution globale optimale (cercle en bleu courtois), la CPSO présente une convergence rapide (présence de particules en couleur bleu) avec un minimum d'erreurs par rapport à l'optimisation par PSO qui présente des erreurs comme s'il y a d'autres optimums.

On peut dire aussi que la CPSO est meilleurs pour la recherche de l'optimum global.

5.6 Application de la condition de recouvrement $C_{seuil} = 2$

Dans cette application, nous essayerons de déployer un réseau de quatre jusqu'à dix caméras pour recouvrir un mouvement d'un cube (utilisant huit marqueurs) afin que chaque marqueur

soit vu par deux caméras ($C_{seuil} = 2$) durant chaque trame du mouvement. Ce mouvement est choisi pour sa complexité, puisqu'il traverse la salle diagonalement; ainsi, il couvre presque un tiers de sa surface.

Notre problème se résout comme un problème d'optimisation de placement 3D d'un réseau de caméras. Pour atteindre notre objectif, nous utiliserons deux métaheuristiques d'optimisation. La première est la PSO et la deuxième est l'optimisation par recherche stochastique fractale RFS.

Pour la première méthode (PSO), nous utiliserons deux variantes : la première est la technique CPSO (Canonical PSO) et la deuxième est une technique modifiée de la CPSO que nous avons appelée MCPSO (de Modified Canonical PSO).

Pour évaluer les performances de ces variantes, nous avons choisi de comparer les résultats obtenus par la CPSO et la MCPSO avec les résultats obtenus par la méthode d'optimisation RFS.

Nous présentons dans ce qui va suivre, les résultats de simulations suivant lePourcentage de Recouvrement (PR) des marqueurs (en %) et le Temps de Calcul (TC en min) pour les différentes techniques (CPSO, MCPSO et RFS). Pour une évaluation convenable entre ces techniques, nous avons fixé le paramètre de population ($N_P = 50$), le nombre des itérations ($T = 500$), le même mouvement exprimé par le nombre de trames ($Tr = 26$) et la condition que chaque marqueur est vu par deux caméras ($C_{seuil} = 2$), pour toutes les techniques.

Durant toutes les simulations avec les deux variantes de la technique PSO, nous avons rencontré le problème où certaines caméras ne couvrent aucun marqueur. C'est pour cette raison que la simulation est répéter jusqu'à ce que toutes les caméras déployées seront utilisées (même si elle couvre un seul marqueur). C'est pour cette raison qu'on retrouve parfois une dégradation du recouvrement malgré l'incrémentation du nombre de caméras.

5.6.1 Recouvrement par la technique CPSO

Dans cette simulation, nous avons utilisé la technique canonicale de la PSO (CPSO) pour rechercher le recouvrement total des marqueurs du cube. Les coefficients cognitifs changent de valeur avec l'évolution des itérations, selon les équations (5.10) et (5.11).

La figure 5.8 présente les résultats d'optimisation par la technique CPSO pour couvrir les marqueurs du cube. Malgré le déploiement de toutes les caméras, le recouvrement total n'est pas atteint par cette méthode (77.88% seulement). Mais nous pouvons noter que le temps de calcul est très court (environ 1.49 minutes seulement).

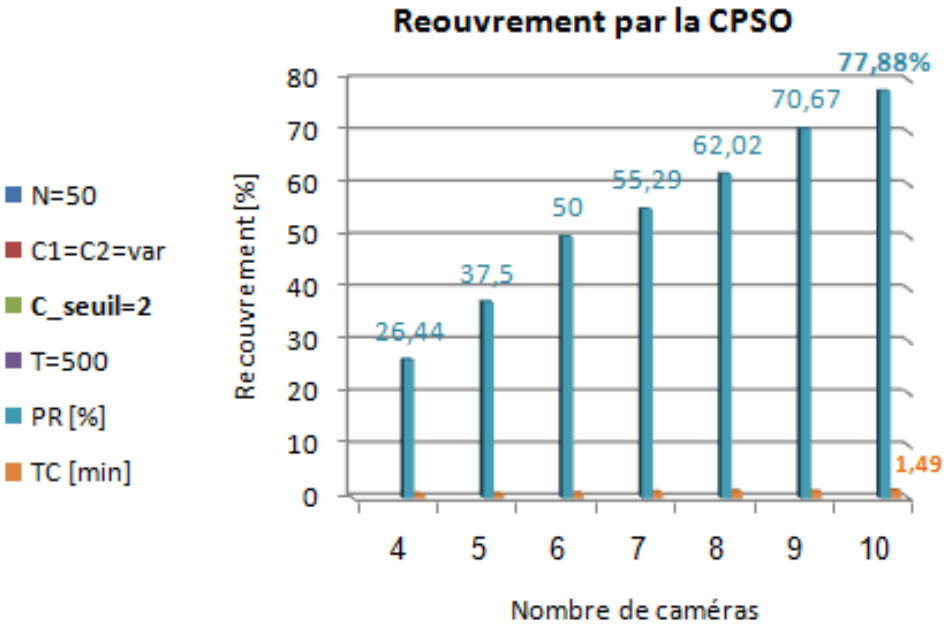


Figure 5.8: Recouvrement par CPSO pour $C_{seuil} = 2$.

Les résultats du déploiement de dix caméras sont donnés par une vue globale en 3D par la figure 5.9(a). La vue de haut (figure 5.9(b)) simplifie la compréhension de nos résultats et on peut remarquer que les marqueurs au milieu de la salle ne sont couverts que par la caméra N°6 ou N°9 seulement; par contre, les autres marqueurs sont couverts par plus de deux caméras, ce qui explique pourquoi le recouvrement total n'est pas atteint. La figure 5.9(c) nous a permet de vérifier qu'il n y'a aucune divergence et qu'aucune caméra ne sorte de l'espace de recherche (sous-sol).

Les valeurs numériques des résultats de placement par la technique CPSO sont données par le tableau 5.2. Nous pouvons constater que les valeurs obtenues ne divergent pas de l'espace de recherche défini par le tableau 5.1.

Notre problème est la maximisation du recouvrement. La figure 5.10(a) présente l'évaluation de la fonction de fitness en fonction des itérations. Selon cette figure, la solution est coincée

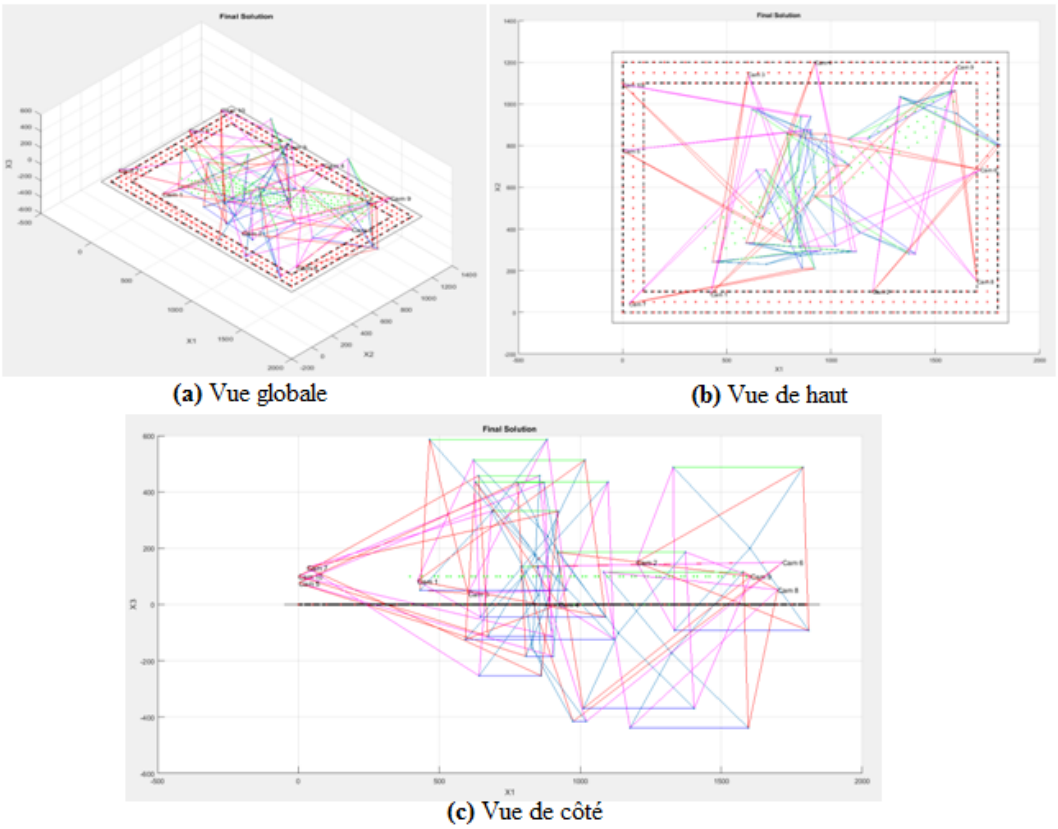


Figure 5.9: Déploiement final du réseau de dix caméras par la CPSO.

dans un maximum local dès le début de la recherche, malgré le passage entre le comportement d'exploration et le comportement d'exploitation des particules (selon la figure 5.10(b)).

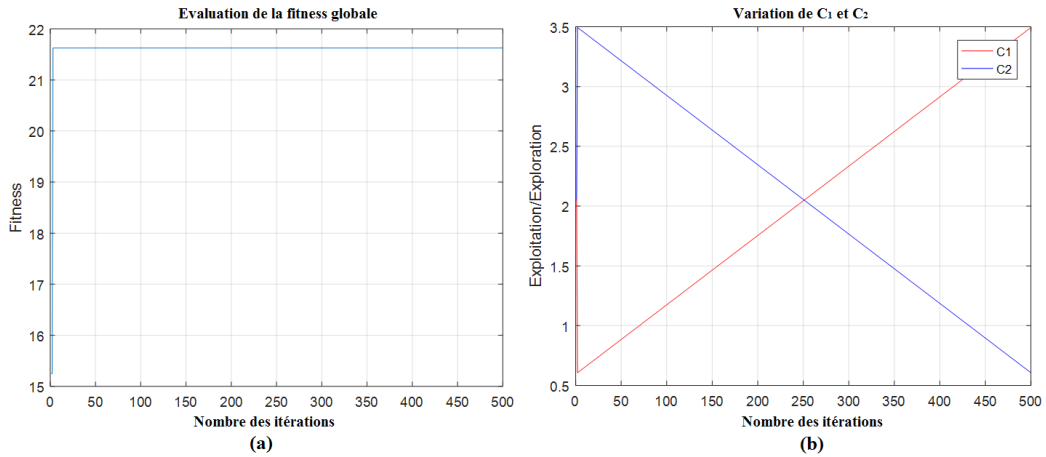


Figure 5.10: Evaluation (a) de la fonction de fitness (b) des coefficients cognitifs pour la CPSO.

Tableau 5.2: Résultats de placement par la CPSO pour $C_{seuil} = 2$

Caméra N°	X[cm]	Y[cm]	Z[cm]	φ_h[°]	φ_v[°]	α_h[°]	α_v[°]	d_{max}	d_{min}
1	421.6	89	83.9	59	10	33.2	36.1	912.8	167.4
2	1200	100	150	64.8	3	33.5	35.8	944.6	169
3	601.4	1143	39.9	-84.1	18.6	33.1	35.9	916.5	167.8
4	922.8	1200	0	-94.4	10.1	33.4	35.6	929.4	165.5
5	8	775.3	76.4	-10.7	3.1	33.5	38.2	944.9	165.3
6	1718	684.1	150	-173.8	-18.8	33.3	35.8	944.5	166.3
7	30.4	42.2	131.1	27.1	-6.0	33.2	36.9	928.1	165.9
8	1700	150	52.5	114.9	-14.1	34.6	36.2	921.2	165.8
9	1604.2	1179.8	100	-121.2	-12.5	33.4	35.7	928	165.3
10	0	1093.4	100	-26	4.6	33.2	35.5	938.7	165.3

5.6.2 Recouvrement par la technique MCPSO

Dans cette section, nous essayerons d'améliorer les résultats précédents par l'application de la MCPSO. Cette nouvelle contribution consiste à limiter l'incrémentation et la décrémentation des coefficients cognitifs (figure 5.11(b)) à 2.1, selon les équations (5.13) et (5.14). Cette limitation consiste à favoriser le comportement d'exploration du comportement d'exploitation au début des itérations puis rendre l'équilibre entre elles à la fin des itérations.

$$C_1 = \left(\frac{(2,1 - c_{max}).t}{T} \right) + c_{max} \tag{5.13}$$

$$C_2 = \left(\frac{(2,1 - c_{min}).t}{T} \right) + c_{min} \tag{5.14}$$

Avec:

$c_{max} = 3.5$ et $c_{min} = 0.6$.

Nous pouvons remarquer (selon la figure 5.11(a)) qu'il y a une amélioration de la qualité à l'itération ~ 20 puis ~ 100 et enfin à ~ 400. Cette amélioration est due au changement d'information entre les particules. Nous pouvons dire qu'il y a un léger avantage par rapport à la technique CPSO.

Le graphe de la figure 5.12 présente les nouveaux résultats obtenus par cette technique, où le meilleur recouvrement a atteint 89.42% pour le déploiement du même nombre de caméras et

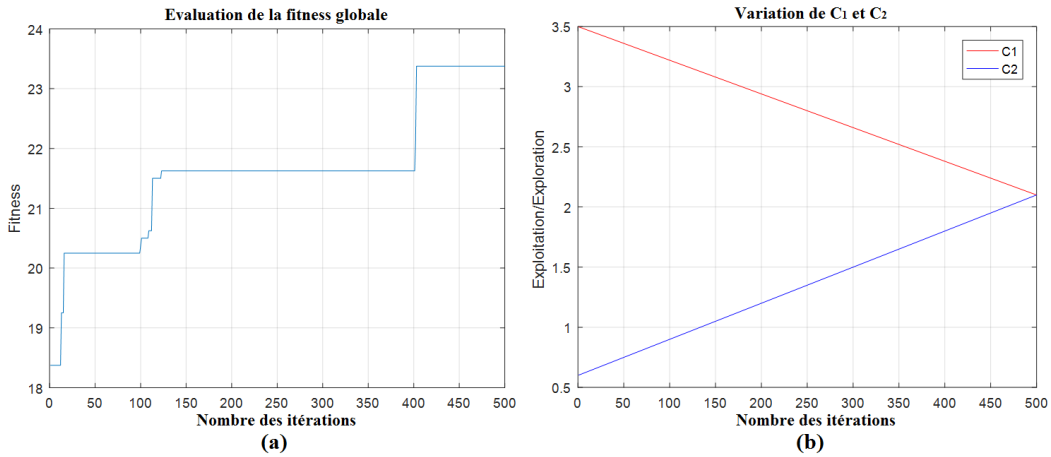


Figure 5.11: Evaluation (a) de la fonction de fitness (b) des coefficients cognitifs pour la MCPSO.

avec un temps de calcul médiocre de 1.24 minutes (réduit que celui de la CPSO).

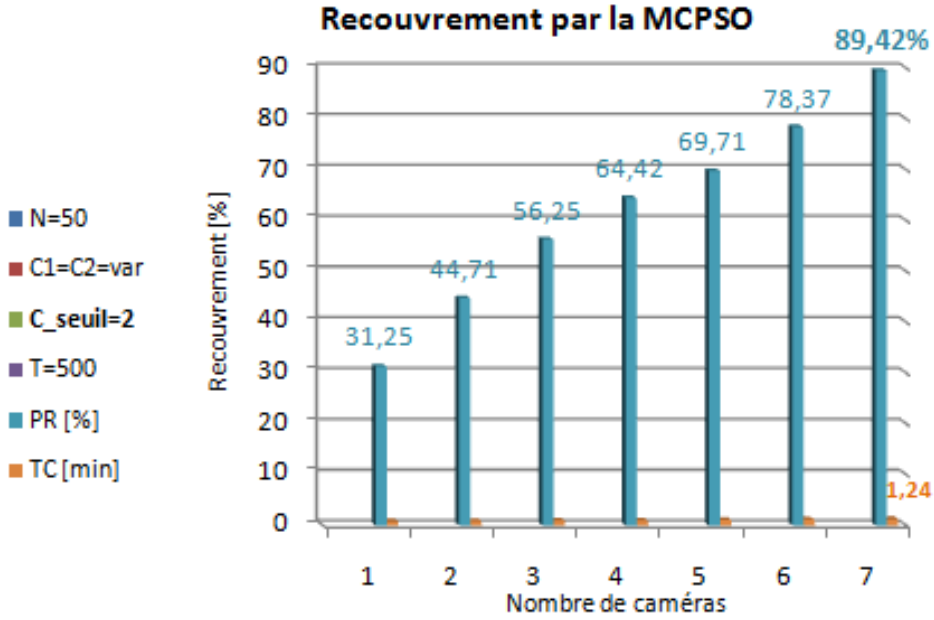
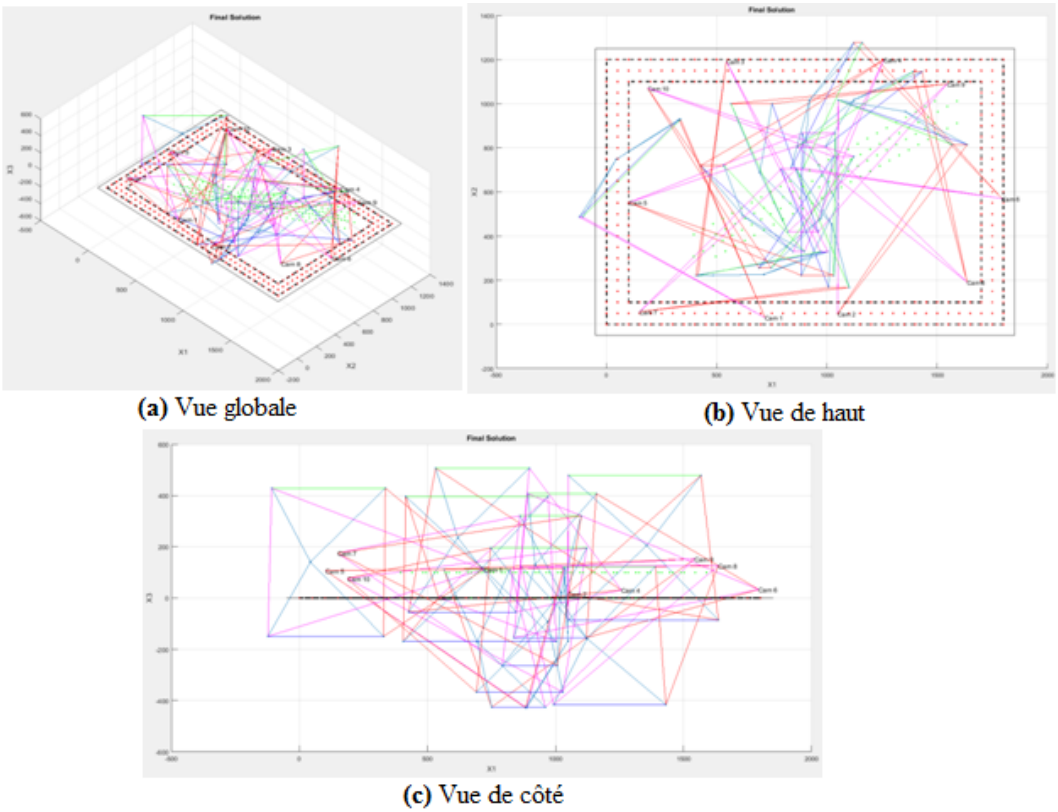


Figure 5.12: Recouvrement par la MCPSO pour $C_{seuil} = 2$.

Les figures (5.13(a), 5.13(b) et 5.13(c)) présentent la vue globale, la vue de haut et la vue de côté respectivement, pour le recouvrement de dix caméras par la technique MCPSO sous la condition $C_{seuil} = 2$.

Les valeurs numériques du résultat de placement par la technique MCPSO sont données par le tableau A.1. Nous pouvons toujours constater que les valeurs obtenues ne diffèrent pas de



(a) Vue globale (b) Vue de haut

(c) Vue de côté

Figure 5.13: Déploiement final du réseau de dix caméras par la MCPSO.

l'espace de recherche défini par le tableau 5.1.

Remarque: les figures et les tableaux illustrant les valeurs numériques de placement seront donnés dans l'annexe A de cette thèse.

5.6.3 Recouvrement par la technique RFS

La figure 5.14 présente les résultats du recouvrement obtenus par l'utilisation de la technique d'optimisation RFS. D'après le graphe, nous pouvons remarquer une grande amélioration par rapport aux deux techniques précédentes, tel que le recouvrement total est atteint pour le déploiement de seulement huit caméras, mais avec un temps de calcul d'environ 3.99 minutes et avec un nombre d'itérations inférieur à 500.

Au début de l'utilisation de la technique RFS, nous avons remarqué que le positionnement final de quelques caméras chevauche avec la trajectoire du cube (figure 5.15) et ce chevauchement pose un grand problème pour la phase de reconstruction du mouvement en 3D.

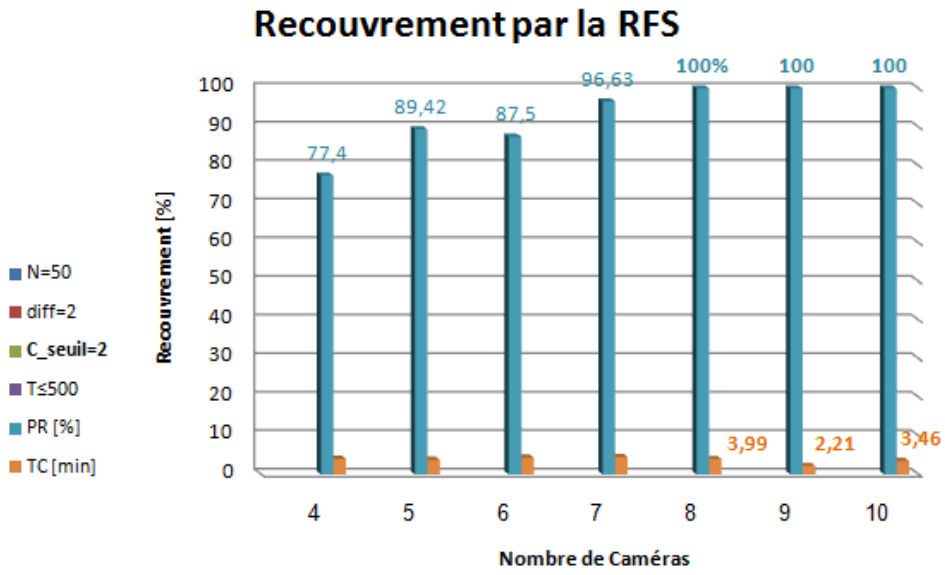


Figure 5.14: Recouvrement par la RFS pour $C_{seuil} = 2$.

Pour les deux premières techniques, ce problème n'est pas posé, puisque chaque caméra est limitée dans sa propre zone. Pour remédier à ce problème, nous avons défini une zone de travail, modélisée par le rectangle noir représenté dans la figure 5.16, où le placement d'une caméra à l'intérieur de cette zone pénalise la solution par la remise à zéro de sa valeur de fitness.

Les figures (A.1(a), A.1(b) et A.1(c)) représentent les résultats obtenus pour le placement de huit caméras avec un recouvrement total de 100%. Selon ces figures, nous pouvons remarquer qu'il y a un problème dans le placement final de la caméra N°2, où il y a toujours un seul chevauchement avec la trajectoire du cube à l'intérieur de la scène, ce qui va être considéré comme un obstacle qui gène le mouvement réel du cube une fois l'opération d'acquisition est lancée.

Dans le domaine de la MoCap, ce problème peut être résolu manuellement par l'opérateur, tout en surlevant un peu la caméra en question, puisque la solution donnée par notre système est une solution approchée et non exacte.

La figure 5.17 représente l'évolution de la fonction de fitness par RFS, pour le déploiement de 8 caméras, où l'objectif est atteint ($PR = 100\%$) aux alentours de l'itération 415. Nous remarquons au début de la simulation, qu'il y a une dégradation de la fonction de fitness, qui

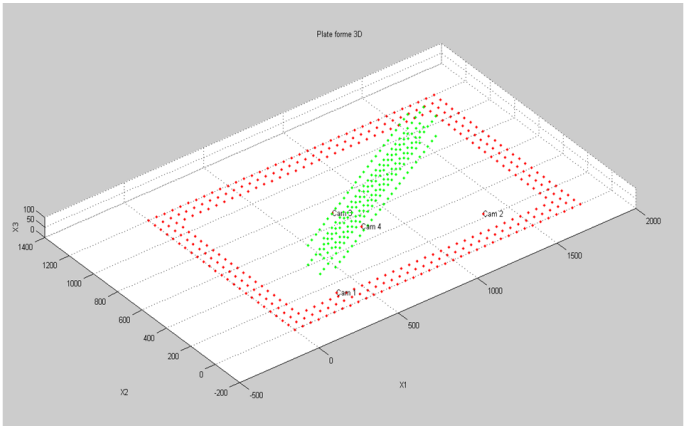


Figure 5.15: Chevauchement entre le cube et les caméras.

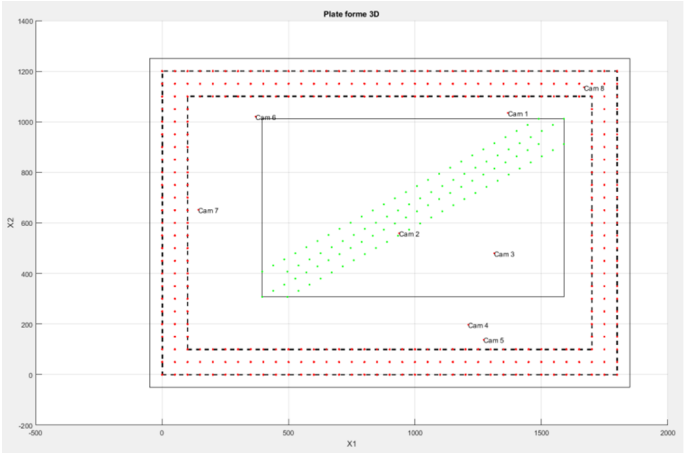


Figure 5.16: Limitation de la zone de travail pour la technique RFS.

peut être causée par le positionnement des caméras à l'intérieur de la zone de travail ou par des mauvaises poses, puis une amélioration rapide à partir de l'itération ~ 100. Le retour de la courbe vers le maximum exprime bien la fiabilité de cette technique.

5.6.4 Comparaison globale des résultats (pour $C_{seuil} = 2$)

Le graphe de la figure 5.18 résume les résultats de simulations obtenus par les trois techniques d'optimisation (CPSO, MCPSO et RFS). Nous remarquons bien que la méthode RFS a donnée les meilleurs résultats par rapport aux deux techniques PSO, où l'écart entre les courbes est bien clair. Bien que les deux méthodes PSO n'aient pas abouti au recouvrement total, les deux courbes montrent bien que la méthode MCPSO présente une légère amélioration de re-

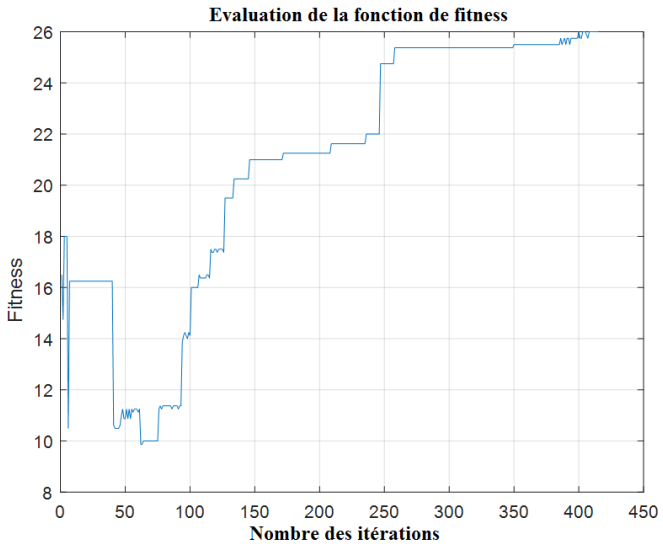


Figure 5.17: Evaluation de la fitness par la RFS pour $C_{seuil} = 2$.

couvrement par rapport à la technique CPSO.

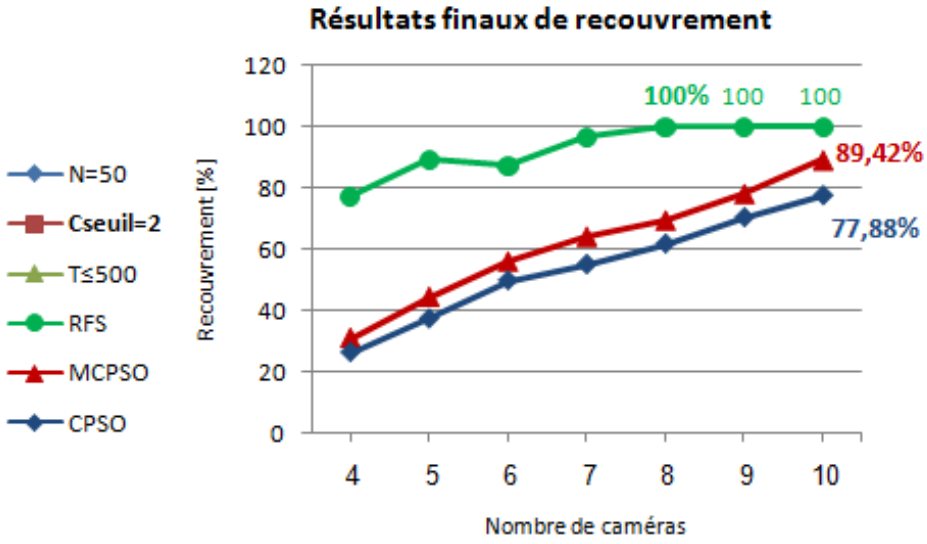


Figure 5.18: Résultats finaux de recouvrement ($C_{seuil} = 2$) pour les trois méthodes.

Pour mieux tester l'algorithme d'optimisation RFS pour notre application, nous allons étendre le nombre d'itérations jusqu'à 5000 itérations. L'histogramme empilé de la figure 5.19 présente en détails les différents résultats de simulation (Recouvrement, temps de calcul et nombre d'itérations) pour la condition $C_{seuil} = 2$.

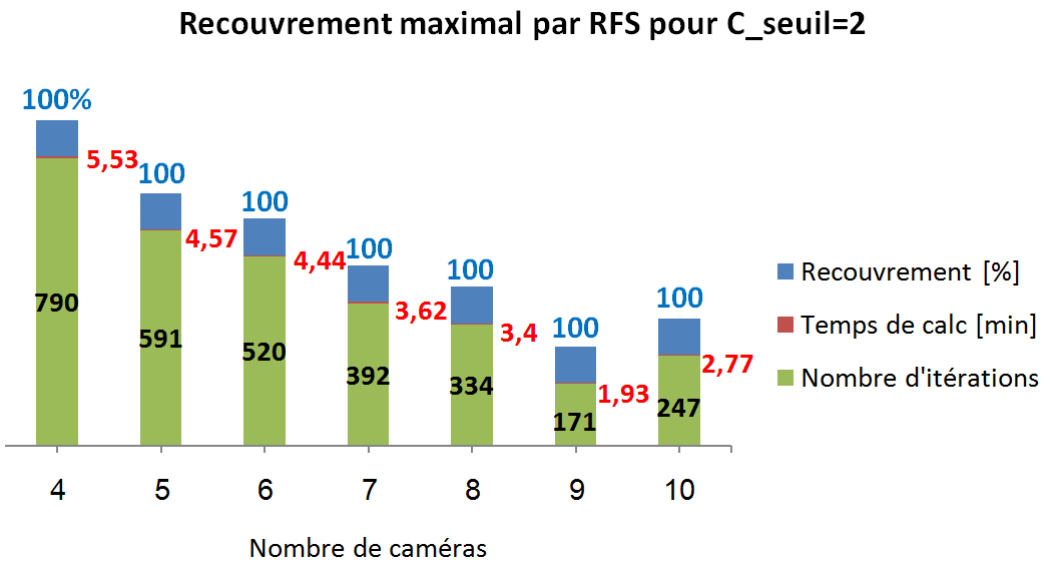


Figure 5.19: Evaluation des résultats en fonction des itérations.

Cet histogramme montre bien la diminution du temps de calcul et du nombre d'itérations tout en augmentant le nombre de caméras, avec un recouvrement total assuré pour la condition $C_{seuil} = 2$. Nous remarquons une importante amélioration entre le déploiement de huit et neuf caméras, qui s'explique par la chance de tomber sur une solution très proche de l'optimal, ce qui a permis à l'algorithme de trouver la solution dans un court délai. Par contre nous remarquons une dégradation entre le déploiement de neuf et dix caméras qui n'est pas vraiment une dégradation si nous la comparons avec le résultat de huit caméras, puisque le nombre d'itérations est inférieur.

5.6.5 Vérification par rapport à l'espace de recherche

Si nous prenons par exemple la solution de placement de la caméra N°1 dans le tableau 5.2. La figure 5.20 montre bien que la solution est bien compatible avec les limites de l'espace de recherche données par le tableau 5.1. La courbe en rouge indique les limites maximaux des paramètres à optimiser et la courbe en bleu montre les limites minimaux. D'après cette figure, les valeurs optimisées de la solution sont présentées par la courbe en vert pour la CPSO et par la courbe en violet pour la MCPSO. La courbe en bleu clair présente la solution par la méthode RFS qui diverge de ces limites seulement pour les coordonnées X et Y; mais pratiquement, ça

ne pose aucun problème.

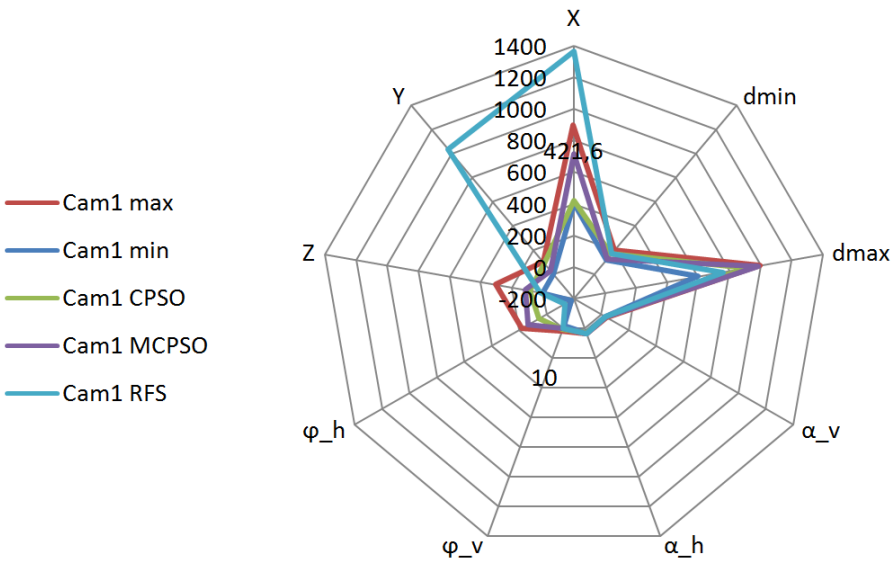


Figure 5.20: Les limites de l'espace de recherche.

5.7 Application de la condition de recouvrement $C_{seuil} = 3$

Pour augmenter la chance d'avoir plus de captures pour chaque marqueur et par conséquence d'assurer la phase de reconstruction 3D du mouvement, nous avons augmenté la condition que chaque marqueur doit être vu par trois caméras durant chaque trame du mouvement. Dans ce qui suit, nous allons résoudre le problème avec les mêmes méthodes décrites dans le cas $C_{seuil} = 2$.

5.7.1 Recouvrement par la technique CPSO

Le graphe de la figure 5.21 présente le pourcentage de recouvrement et le temps de calcul dans le cas de l'optimisation par la technique CPSO. L'augmentation de la condition de recouvrement à trois caméras a provoquée des résultats dégénérés par cette méthode. Nous pouvons dire que cette technique a atteint ces limites d'optimisation pour le cas $C_{seuil} = 3$.

Les figures (A.2(a), A.2(b) et A.2(c)) représentent la vue globale, la vue de haut et la vue de côté respectivement, pour le recouvrement de dix caméras par la technique CPSO sous la condition $C_{seuil} = 3$.

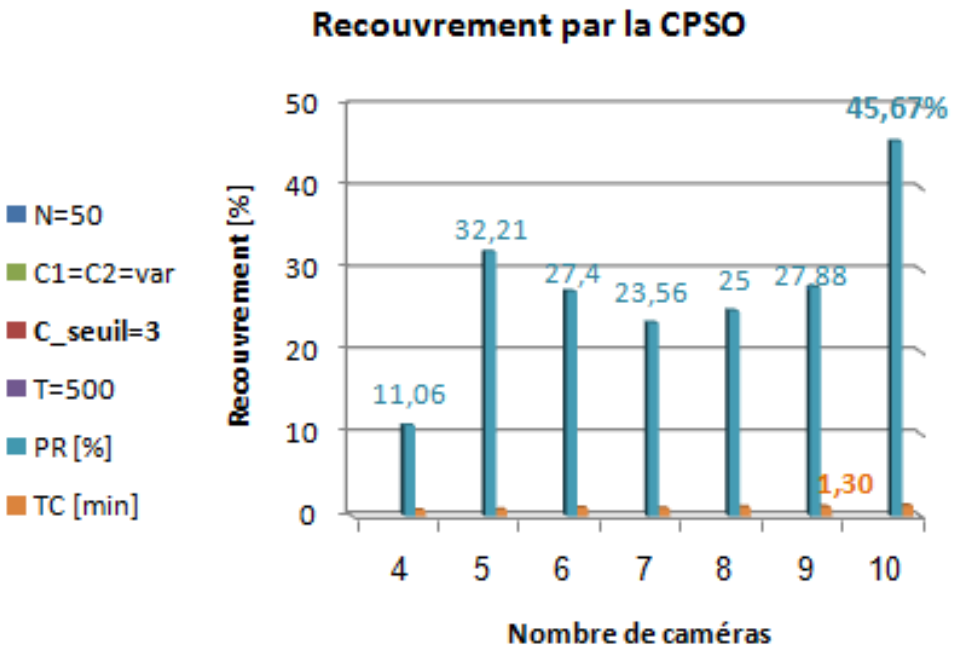


Figure 5.21: Recouvrement par la CPSO pour $C_{seuil} = 3$.

La fonction de fitness donnée par la courbe de la figure 5.22 présente une longue stabilité de la valeur de fitness (~ 12.75 frames) c'est à dire un recouvrement de $\sim 49\%$ ($\sim 100 * 12.75/26$) et ça peut être traduit comme un coincement de la solution dans un maximum local, puisque il n'y a aucune amélioration du résultat jusqu'à la fin des itérations.

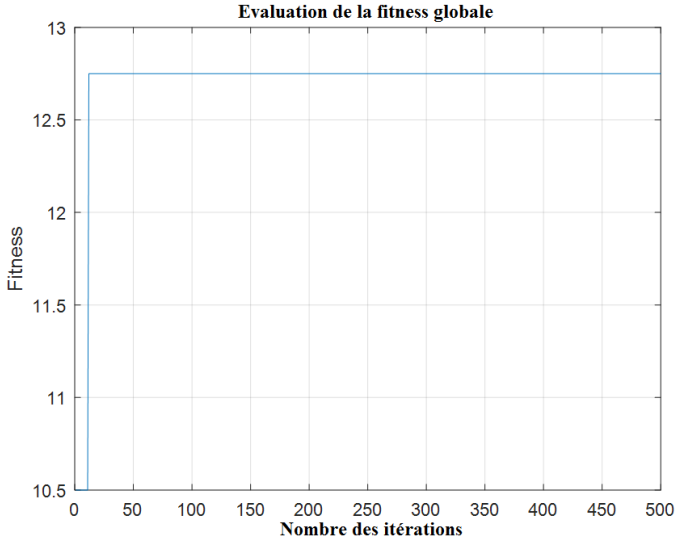


Figure 5.22: Evaluation de la fitness par la CPSO pour $C_{seuil} = 3$.

5.7.2 Recouvrement par la technique MCPSO

L'histogramme de la figure 5.23, donne les résultats de simulation du déploiement d'un réseau de caméras par la méthode MCPSO sous la condition $C_{seuil} = 3$. Nous pouvons facilement remarquer une légère amélioration par rapport à la CPSO pour la même condition. Mais malheureusement, le recouvrement total n'est toujours pas atteint par la technique MCPSO.

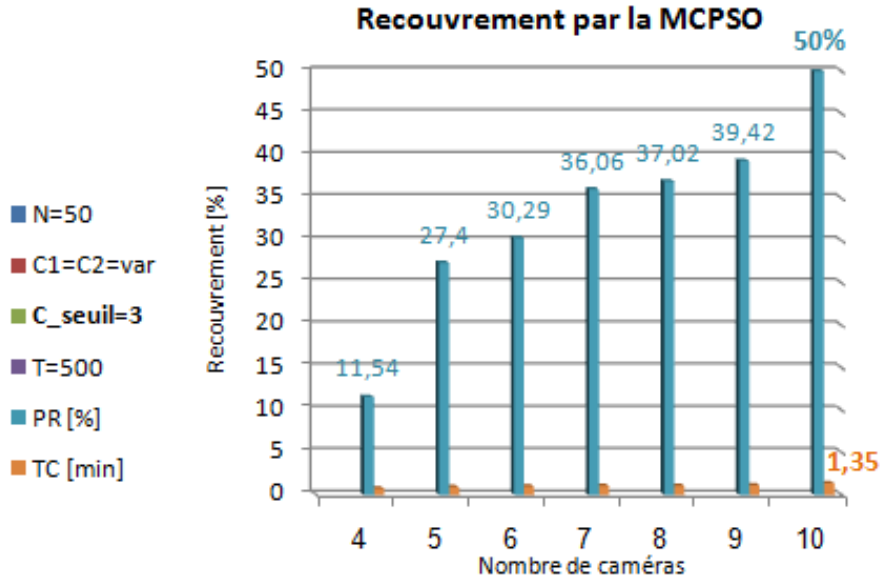


Figure 5.23: Recouvrement par la MCPSO pour $C_{seuil} = 3$.

Les figures (A.3(a), A.3(b) et A.3(c)) représentent la vue globale, la vue de haut et la vue de côté respectivement, pour le recouvrement de dix caméras par la technique MCPSO sous la condition $C_{seuil} = 3$.

Cette fois la fonction de fitness donnée par la figure 5.24 présente une graduation (marches d'escalier) de la valeur de la fitness globale. L'allure de cette courbe s'explique par la localisation, par les particules, d'une meilleure solution par rapport à la précédente en fonction des itérations, même si l'objectif final n'est pas atteint.

5.7.3 Recouvrement par la technique RFS

L'exigence de la condition $C_{seuil} = 3$ a rendue la tache très difficile pour les techniques PSO, mais pas pour la méthode RFS (figure 5.25), qui présente une légère dégradation par rapport à

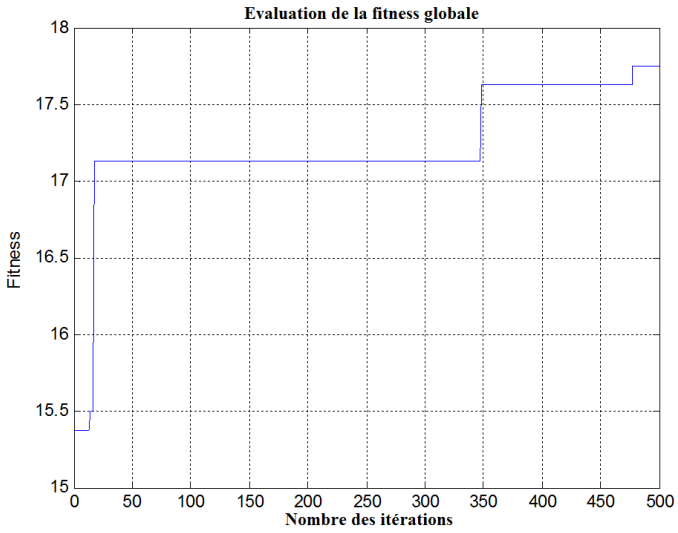


Figure 5.24: Evaluation de la fitness par la MCPSO pour $C_{seuil} = 3$.

la condition $C_{seuil} = 2$, mais l'objectif final est atteint même avec le déploiement d'une caméra de plus.

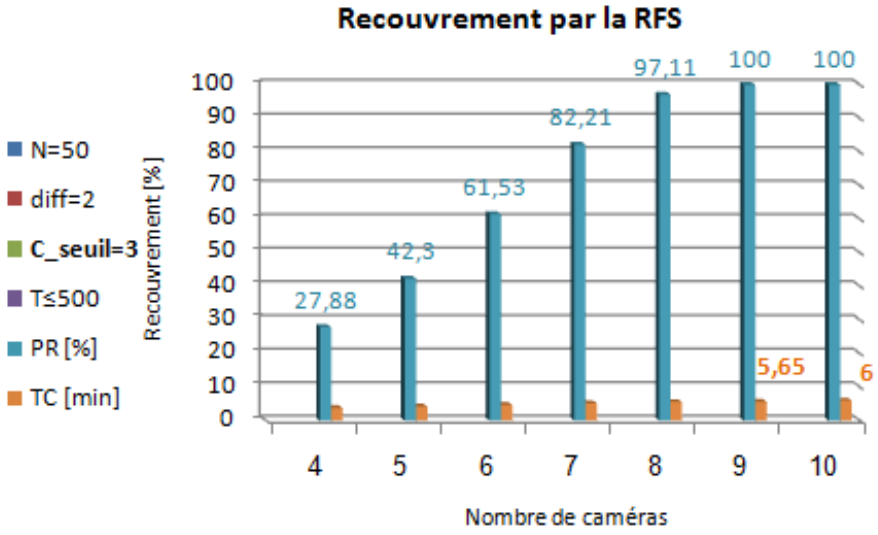


Figure 5.25: Recouvrement par la RFS pour $C_{seuil} = 3$.

Les figures (A.4(a), A.4(b) et A.4(c)) représentent la vue globale, la vue de haut et la vue de côté respectivement, pour le recouvrement de neuf caméras par la technique RFS sous la condition $C_{seuil} = 3$.

La figure 5.26 représente l'amélioration successive de la qualité de la solution exprimée par

sa valeur de fitness. Cette courbe a une allure linéaire, si nous excluons les deux dégradations aux itérations ~ 20 et ~ 400. Cette courbe exprime une recherche continue de la solution optimale et le retour à la meilleure solution malgré le passage par ces deux dégradations.

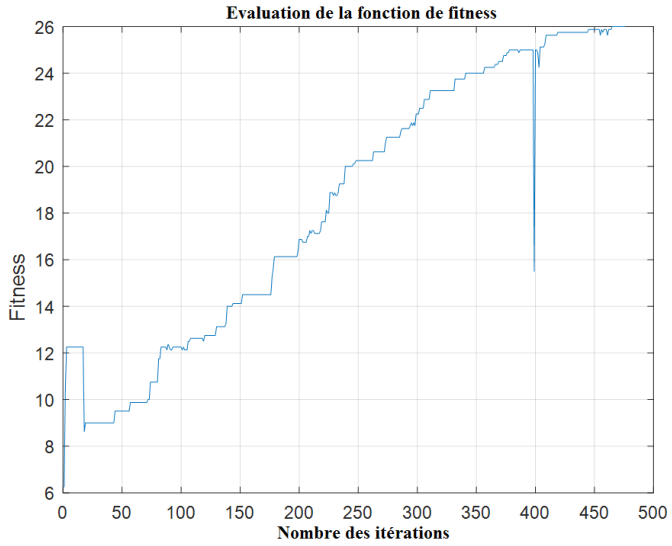


Figure 5.26: Evalluation de la fitness par la RFS pour $C_{seuil} = 3$.

5.7.4 Comparaison globale des résultats (pour $C_{seuil} = 3$)

Une meilleure vue globale des résultats obtenus par les trois méthodes sous la condition $C_{seuil} = 3$, est donnée par les courbes de la figure 5.27.

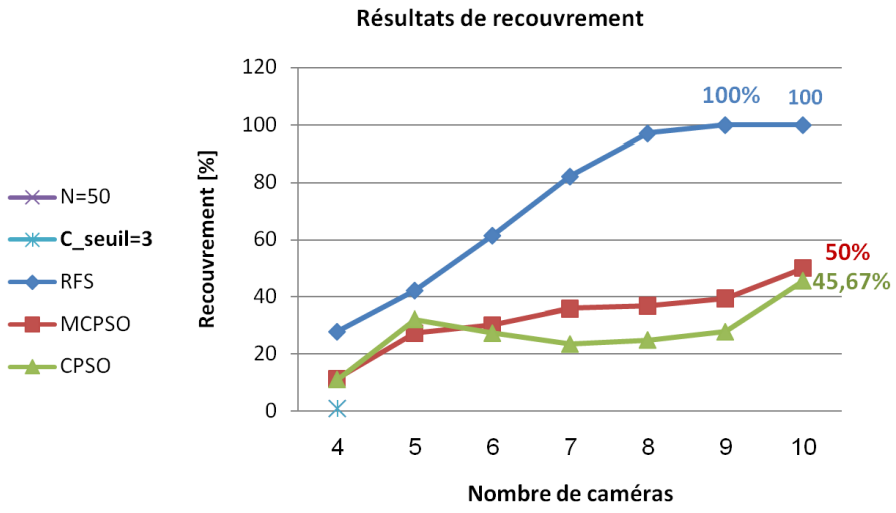


Figure 5.27: Résultats de recouvrement par les trois méthodes avec $C_{seuil} = 3$.

Les deux méthodes CPSO et MCPSO présentent presque les mêmes résultats, avec une légère dégradation de la CPSO par rapport à la MCPSO et ça est du au grand écart entre le comportement d'exploration et d'exploitation adopter par la CPSO. La réduction de cet écart a aidé la méthode MCPSO à exploiter rapidement les solutions trouvées dans la phase d'exploration et de sortir avec des solutions légèrement optimles par rapport à la CPSO.

La méthode RFS présente les meilleurs résultats. Pour mieux voir les opportunités de cette méthode pour notre application, nous avons répété les simulations pour un nombre plus grand d'itérations (5000 itérations).

Les résultats obtenus sont présentés par l'histogramme de la figure 5.28, montre bien une grande amélioration au niveau de notre objectif (recouvrement total), où il est atteint par le déploiement de six caméras, mais au prix d'un large temps de calcul (36,32 minutes ou 4356 itérations). D'autres solutions acceptables pour leurs temps de calcul et le nombre d'itérations, allons de 12 minutes pour le placement de sept caméras jusqu'à 7 minutes pour le placement de neuf caméras (1314 à 677 itérations respectivement). Le déploiement du nombre total des caméras donne le plus court temps de calcul et de nombre d'itérations.

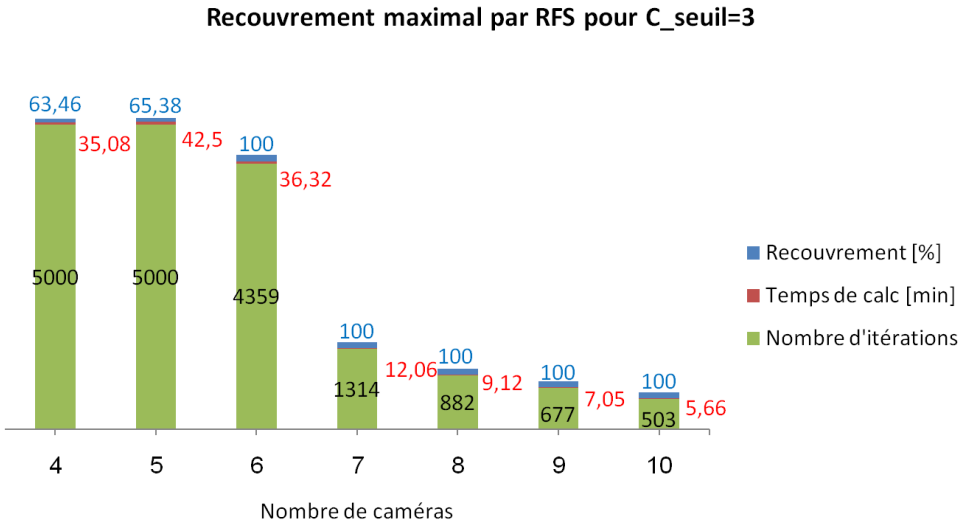


Figure 5.28: Limites du recouvrement total par la RFS pour $C_{seuil} = 3$.

Pour le cas de 4 et 5 caméras, le résultat obtenu est le maximal c'est à dire que le recouvrement à 100% est impossible pour ces deux cas. Ce que montrent les deux courbes de fitness pour le déploiement de 4 et 5 caméras (respectivement figure 5.29(a) et 5.29(b)).

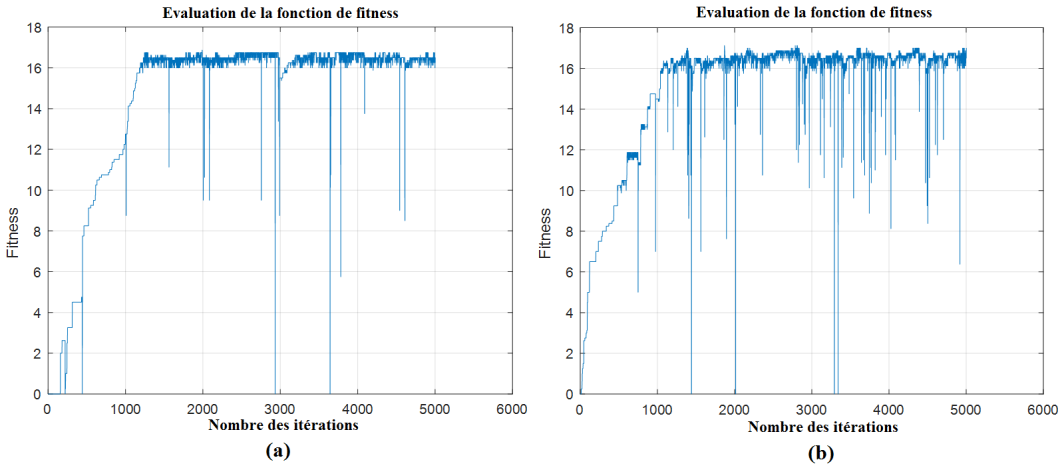


Figure 5.29: Fonction de fitness pour (a) 4 caméras (b) 5 caméras

5.8 Conclusion

Dans ce chapitre, nous avons crée un environnement virtuel en 3D qui est compatible avec un système MoCap optique. Dans notre environnement, nous avons négligé le problème d'occlusion, quelle soit interne ou externe, causée par l'opacité du cube ou par la pose d'obstacles dans la scène.

La première phase été de modéliser le champ de vision en 3D d'une caméra et de vérifier si n'importe quel marqueur peut être vu par une caméra donnée ou non. Sur cette base une simulation d'un mouvement d'un cube est réalisée afin d'optimiser le placement d'un réseau de caméras ainsi que leurs nombres, dont la condition est de couvrir chaque marqueur par deux ($C_{seuil} = 2$) ou par trois ($C_{seuil} = 3$) caméras durant chaque trame du mouvement.

L'objectif principal de notre optimisation est de placer un nombre minimal de caméras dans un temps le plus réduit possible pour atteindre le recouvrement total.

Par l'application de la première ou de la deuxième condition, les deux variantes de la PSO n'ont pas atteint l'objectif final (recouvrement à 100%). Par contre la méthode RFS a atteint l'objectif final pour les deux conditions, mais avec une relation inverse entre le nombre d'itérations et le nombre de caméras déployées.

6

Conclusion générale et perspectives

6.1 Conclusion générale

Nous avons traité dans cette thèse le problème du placement d'un réseau de caméras autour d'un objet muni de marqueurs pour capturer son mouvement dans une scène. De tels systèmes, sont dits systèmes de capture de mouvement ou MoCap. Dans cette thèse, nous avons choisi un système de capture de mouvement optique. Ainsi, nous avons donné une étude sur le modèle sténopé complet décrit par la matrice de projection perspective qui modélise le capteur (la caméra).

Notre problème de placement a été traité comme un problème d'optimisation combinatoire, puisqu'il s'agit de trouver une meilleure solution parmi plusieurs combinaisons, tout en respectant des contraintes contradictoires avec l'objectif principal du problème. Notre travail s'est orienté vers l'étude des méthodes d'optimisation qui cherchent des solutions approchées avec un temps d'exécution acceptable par rapport aux méthodes exactes; la métaheuristique d'optimisation par essaim de particule "OEP" avec ses variantes (SPSO, WPSO et CPSO) et aussi la recherche fractale stochastique "RFS", ont fait l'objet d'une étude détaillée dans ce travail.

Deux applications de la métaheuristique "OEP" (avec ses variantes) dans un système de capture de mouvement virtuel ont fait l'objet d'une optimisation de placement 2D d'un réseau de caméras. Comme première application, nous avons pu optimiser (à l'aide de la WPSO) le placement 2D d'un réseau de quatre caméras dont l'objectif est de recouvrir un seul point , un point avec obstacle et puis son mouvement à l'intérieur d'une scène tout en respectant les contraintes imposées. L'objectif de recouvrement ($PR = 100\%$) a été atteint pour cette application. Comme une deuxième application, nous avons utilisé la CPSO (avec la variation de C_1 et C_2) pour optimiser le placement 2D d'un réseau de caméras afin de recouvrir le mouvement des marqueurs d'un cube pour trois différents scénarios.

Les méta-heuristiques CPSO, MCPSO et la RFS, ont été appliquées pour l'optimisation de placement 3D d'un réseau de caméras pour le recouvrement des marqueurs d'un cube (pour un scénario unique). Dans le domaine 3D, le nombre de paramètres a augmenté par rapport au cas 2D, ce qui a rendu le processus d'optimisation plus difficile. Les paramètres à chercher pour le placement d'un réseau de caméras ont été optimisés avec deux principales applications de

recouvrement ; la première application est que chaque marqueur est recouvert par deux caméras ($C_{seuil} = 2$) où la technique RFS a atteint le recouvrement total ($PR = 100\%$ par le déploiement de 8 caméras); tandis que, la technique MCPSO n'a pas donnée un résultat satisfaisant, mais elle a présenté une légère amélioration de résultats (avec un $PR = 89.42\%$) par rapport à un $PR = 77.88\%$ pour la CPSO (déploiement de 10 caméras pour chacune). La deuxième application exige que le recouvrement doit être assuré par trois caméras ($C_{seuil} = 3$). Cette condition rend la tâche de trouver une solution optimale plus difficile que l'application précédente, d'où le faible pourcentage ($PR = 45.67\%$ et $PR = 50\%$ pour la CPSO et la MCPSO respectivemet); Ici, et pour cette application, la technique RFS amiliore nettement les résultats précédents et assure un recouvrement total ($PR = 100\%$ par le déploiement de 9 caméras seulement).

6.2 Perspectives

Le domaine de la capture de mouvement est un domaine vaste et prometteur. Le problème de placement optimal reste toujours pertinent, puisqu'on cherche toujours une solution approchée influencée par plusieurs critères (mouvement à effectuer, nombre de personnes, présence d'obstacles fixes ou mobiles, sources de lumière ...etc.).

Dans notre application 3D, nous avons négligé le problème de la présence d'obstacles qui provoquent les occlusions (internes ou externes) des marqueurs.

Comme première perspective, nous proposons d'abord l'implémentation de notre algorithme "RFS" dans un environnement virtuel tridimensionnel, pour qu'il soit plus proche à l'environnement réel du domaine de la capture du mouvement (Par exemple : VRML de MATLAB ou autre).

Une deuxième perspective, consiste à la reconstruction du mouvement en 3D comme fonction d'objectif pour vérifier et valider les résultats de placement du réseau de caméras par simulation.

Une troisième perspective consiste à comparer les résultats de simulation avec un mouvement réel enregistré dans un laboratoire équipé d'un système MoCap optique.

Bibliographie

[1] R. Parent, *Computer animation: algorithms and techniques*. Newnes, 2012.

[2] http://cgicoffee.com/blog/2017/04/first-production-motion-capture-session report. [Online]. Available: 05/01/2018

[3] S. Tirakoat, "Optimized motion capture system for full body human motion capturing case study of educational institution and small animation production," in *Digital Media and Digital Content Management (DMDCM), 2011 Workshop on*. IEEE, 2011, pp. 117–120.

[4] A. J. Davison, J. Deutscher, and I. D. Reid, "Markerless motion capture of complex full-body movement for character animation," in *Eurographics Workshop on Animation and Simulation*. Springer, 2001, pp. 3–14.

[5] R. Parent, D. S. Ebert, D. Gould, M. Gross, C. Kazmier, C. J. Lumsden, R. Keiser, A. Menache, M. Müller, F. K. Musgrave *et al.*, *Computer animation complete: all-in-one: learn motion capture, characteristic, point-based, and Maya winning techniques*. Morgan Kaufmann, 2009.

[6] www.semanticscholar.org. [Online]. Available: 05/01/2018

[7] http://metamotion.com/gypsy/gypsy-motion-capture system.htm. [Online]. Available: 05/01/2018

[8] https://lukebeech.wordpress.com/motion capture/. [Online]. Available: 05/01/2018

[9] F. Rameau, "Système de vision hybride à fovéation pour la vidéo-surveillance et la navigation robotique," Ph.D. dissertation, Dijon, 2014.

[10] E.-G. Talbi, *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009, vol. 74.

[11] J.-C. Boisson, “Modélisation et résolution par métaheuristiques coopératives: de l’atome à la séquence protéique,” Ph.D. dissertation, Université Lille 1, 2008.

[12] H. Salimi, “Stochastic fractal search: a powerful metaheuristic algorithm,” *Knowledge-Based Systems*, vol. 75, pp. 1–18, 2015.

[13] D. Knossow, “Paramétrage et capture multicaméras du mouvement humain,” Ph.D. dissertation, Institut National Polytechnique de Grenoble-INPG, 2007.

[14] N. F. Troje, C. Westhoff, and M. Lavrov, “Person identification from biological motion: Effects of structural and kinematic cues,” *Attention, Perception, & Psychophysics*, vol. 67, no. 4, pp. 667–675, 2005.

[15] B. Michoud, “Reconstruction 3d à partir de séquences vidéo pour l’acquisition du mouvement de personnages en temps réel et sans marqueur,” Ph.D. dissertation, Université Claude Bernard-Lyon I, 2009.

[16] N. Fusco, “Analyse, modélisation et simulation de la marche pathologique,” Ph.D. dissertation, Université Rennes 2, 2008.

[17] M. Cognolato, “Experimental validation of xsens inertial sensors during clinical and sport motion capture applications,” 2012.

[18] S. Hachem, “Analyse du mouvement humain À l’aide d’un systÈme de capture de mouvement,” Master’s thesis, UNIVERSITÉ DE SHERBROOKE, Mai 2015.

[19] P. F. G. L. M. T. H. VASSEUR, *OÙ EN SONT LES TECHNOLOGIES NUMÉRIQUES AU CINÉMA ?*, INSA Rennes, 2013.

[20] T. Helten, “Processing and tracking human motions using optical, inertial, and depth sensors,” 2013.

[21] S. Ménardais, “Fusion et adaptation temps réel de mouvements acquis pour l’animation d’humanoïdes synthétiques,” Ph.D. dissertation, Rennes 1, 2003.

[22] D. Ménard, “Génération d’animations par capture de mouvements avec multiples caméras 3d,” Ph.D. dissertation, École Polytechnique de Montréal, 2014.

[23] P. Nogueira, “Motion capture fundamentals: A critical and comparative analysis on real-world applications,” in *v zborniku: 4th International Conference on Information Society and Technology*, 2011.

[24] T. Shiratori, H. S. Park, L. Sigal, Y. Sheikh, and J. K. Hodgins, "Motion capture from body-mounted cameras," in *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4. ACM, 2011, p. 31.

[25] http://www.auvsi.org/tiny-and-everywhere-unmanned-mems-movement 0. [Online]. Available: 05/01/2018

[26] A. Shen and T. Ting, "Marker-less motion capture for biomechanical analysis using the kinect sensor," B.S. thesis, Universitat Politècnica de Catalunya, 2014.

[27] D. Roetenberg, H. Luinge, and P. Slycke, "Xsens mvn: full 6dof human motion tracking using miniature inertial sensors," *Xsens Motion Technologies BV, Tech. Rep*, 2009.

[28] S. P. Storing, "Motion capture."

[29] C. Pradalier and D. Jud, "Motion tracking systems."

[30] P. Clerc, "Mesure de champs de déplacements et de déformations par stéréovision et corrélation d'images numériques," Ph.D. dissertation, Villeurbanne, INSA, 2001.

[31] C. Cauchois, E. BRASSART, and C. DROCOURT, "Calibration du capteur de vision omnidirectionnelle syclop," *DEA Report*, 1998.

[32] C. Li, "Particle swarm optimization in stationary and dynamic environments," Ph.D. dissertation, University of Leicester, 2011.

[33] L. SAID, "Méthodes bio-inpirées hybrides pour la résolution de problèmes complexes," Ph.D. dissertation, Thèse Doctorat en Sciences en Informatique, Université Constantine 2, Tunisie, vol., n" Avril, 2013.

[34] D. E. Knuth, "Fundamentalalgorithms, volume 1 ofthe art of computer programming," 1973.

[35] C. Papadimitriou, "Computational complexity addison-wesley reading," *Massachusetts Google Scholar*, 1994.

[36] A. Gherboudj, "Méthodes de résolution de problèmes difficiles académiques," Ph.D. dissertation, Thèse de Doctorat, Université de Constantine 2, Algérie, 2013.

[37] V. Paschos, "Optimisation combinatoire. 1, concepts fondamentaux," *Hermès Sci. Lavoisier Paris*, 2005.

[38] A. El Dor, "Perfectionnement des algorithmes d'optimisation par essaim particulaire: applications en segmentation d'images et en électronique," Ph.D. dissertation, Université Paris-Est, 2012.

[39] J.-K. Hao, P. Galinier, and M. Habib, "Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes," *Revue d'intelligence artificielle*, vol. 13, no. 2, pp. 283–324, 1999.

[40] M. Douiri, S. Elbernoussi, and H. Lakhbab, "Cours des méthodes de résolution exactes heuristiques et métaheuristiques," *Université Mohamed V, Faculté des sciences de Rabat*, 2009.

[41] F. Bekkari, "Résolution des problèmes difficiles par optimisation distribuée," Ph.D. dissertation, 2009.

[42] J.-C. Créput, "Hybridation de métaheuristiques pour la résolution distribuée de problèmes d'optimisation spatialisés," Ph.D. dissertation, Université de Bourgogne, 2008.

[43] M. Clerc and P. Siarry, "Une nouvelle métaheuristique pour l'optimisation difficile: la méthode des essaims particulaires," *J3eA*, vol. 3, p. 007, 2004.

[44] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. IEEE, 1995, pp. 39–43.

[45] A. Dutot and D. Olivier, "Optimisation par essaim de particules application au problème des n-reines," *Laboratoire Informatique du Havre, Université du Havre*, p. 8, 2002.

[46] W. T. Reeves, "Particle systems—a technique for modeling a class of fuzzy objects," *ACM Transactions on Graphics (TOG)*, vol. 2, no. 2, pp. 91–108, 1983.

[47] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *ACM SIGGRAPH computer graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[48] C.-O. Erneholm, "Simulation of the flocking behavior of birds with the boids algorithm," 2011.

[49] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on evolutionary computation*, vol. 8, no. 3, pp. 240–255, 2004.

[50] I.-H. Kuo, S.-J. Horng, T.-W. Kao, T.-L. Lin, C.-L. Lee, T. Terano, and Y. Pan, "An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model," *Expert systems with applications*, vol. 36, no. 3, pp. 7027–7032, 2009.

[51] J. Onwunalu, "Optimization of field development using particle swarm optimization and new well pattern descriptions," Ph.D. dissertation, Stanford University, 2010.

[52] S. Talukder, "Mathematicle modelling and applications of particle swarm optimization," 2011.

[53] M. Clerc, "Standard particle swarm optimisation from 2006 to 2011 (2011)," *URL¡ www. particleswarm. info*.

[54] N. Nouaouria, "Une approche d'optimisation par essaim de particules pour la recherche en mémoire de cas," Ph.D. dissertation, Université du Québec à Montréal, 2013.

[55] Y. Del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. G. Harley, "Particle swarm optimization: basic concepts, variants and applications in power systems," *IEEE Transactions on evolutionary computation*, vol. 12, no. 2, pp. 171–195, 2008.

[56] R. C. Eberhart, Y. Shi, and J. Kennedy, *Swarm intelligence*. Elsevier, 2001.

[57] G. Wu, D. Qiu, Y. Yu, W. Pedrycz, M. Ma, and H. Li, "Superior solution guided particle swarm optimization combined with local search techniques," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7536–7548, 2014.

[58] S. Devarakonda, "Particle swarm optimization," Ph.D. dissertation, University of Dayton, 2012.

[59] S. Padhy and S. Panda, "A hybrid stochastic fractal search and pattern search technique based cascade pi-pd controller for automatic generation control of multi-source power systems in presence of plug in electric vehicles," *CAAI Transactions on Intelligence Technology*, vol. 2, no. 1, pp. 12–25, 2017.

[60] S. Khalilpourazari, S. H. R. Pasandideh, and S. T. A. Niaki, "Optimization of multi-product economic production quantity model with partial backordering and physical constraints: Sqp, sfs, sa, and wca," *Applied Soft Computing*, vol. 49, pp. 770–791, 2016.

[61] H. Mosbah and M. El-Hawary, "Optimization of neural network parameters by stochastic fractal search for dynamic state estimation under communication failure," *Electric Power Systems Research*, vol. 147, pp. 288–301, 2017.

[62] P. K. Rawlings, "Modes of a gaussian random walk," *Journal of statistical physics*, vol. 111, no. 3, pp. 769–788, 2003.

[63] I. Khanam and G. Parmar, "Order reduction of lti systems using sfs," 2017.

[64] H. Mosbah and M. El-Hawary, "Power system tracking state estimation based on stochastic fractal search technique under sudden load changing conditions," in *Electrical and Computer Engineering (CCECE), 2016 IEEE Canadian Conference on*. IEEE, 2016, pp. 1–6.

[65] G. Calas, "Optimisation par essaim de particules," *Une*, vol. 3, p. 3, 2008.

[66] W. R. Franklin, "Art gallery theorems and algorithms (joseph o'rourke)," *SIAM Review*, vol. 31, no. 2, p. 342, 1989.

[67] E. Horster and R. Lienhart, "Approximating optimal visual sensor placement," in *Multimedia and Expo, 2006 IEEE international conference on*. IEEE, 2006, pp. 1257–1260.

[68] K. R. Konda and N. Conci, "Global and local coverage maximization in multi-camera networks by stochastic optimization," *Infocommunications Journal*, vol. 5, no. 1, pp. 1–8, 2013.

[69] Y. Morsly, N. Aouf, M. S. Djouadi, and M. Richardson, "Particle swarm optimization inspired probability algorithm for optimal camera network placement," *IEEE Sensors Journal*, vol. 12, no. 5, pp. 1402–1412, 2012.

[70] N. Smairi, "Optimisation par essaim particulaire: adaptation de tribes à l'optimisation multiobjectif," Ph.D. dissertation, Université Paris-Est, 2013.

[71] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3. IEEE, 1999, pp. 1945–1950.

[72] M. Hanafi, "Analyse du mouvement humain par vison artificielle pour consoles de jeux vidéos," Ph.D. dissertation, Université Laval, 2012.

[73] R. Malik and P. Bajcsy, "Automated placement of multiple stereo cameras," in *The 8th workshop on omnidirectional vision, camera networks and non-classical cameras-OMNIVIS*, 2008.

A

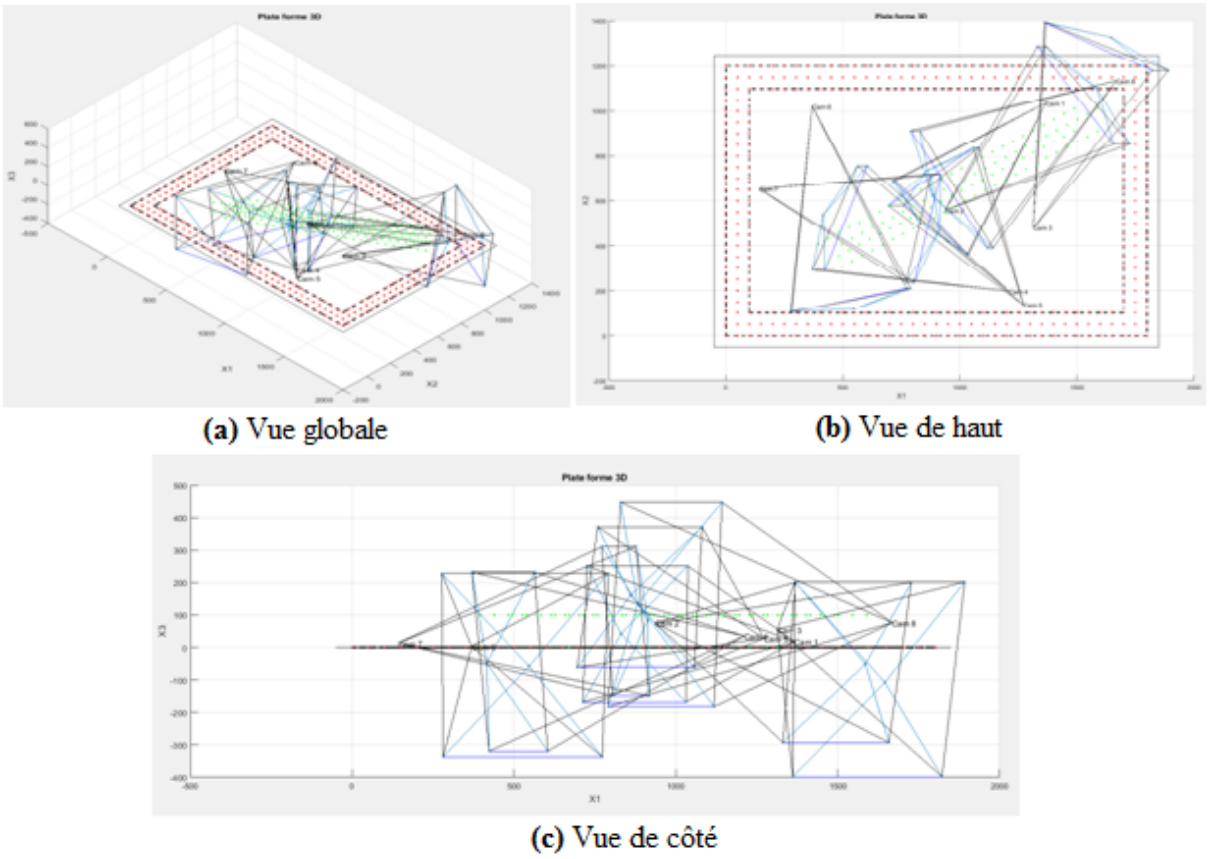
Tableaux et figures

Tableau A.1: Résultats de placement par la MCPSO pour $C_{seuil} = 2$

Caméra N°	X [cm]	Y [cm]	Z [cm]	φ_h [°]	φ_v [°]	α_h [°]	α_v [°]	d_{max}	d_{min}
1	718.6	30.6	112	133.3	1.7	38.2	34.2	985.1	125.7
2	1048.2	48.7	17.1	71.1	11.2	37.5	34.6	967.7	136.7
3	543.4	1191.4	2.3	-79.9	6.8	36.7	33.8	980.2	131.5
4	1257.4	1198.5	31.2	-132.8	12.2	34.7	34.7	962.0	140.4
5	101.8	552.3	106.9	-0.5	-16.1	38.1	33.4	986.5	127.3
6	1793.4	570.4	33.2	153.1	5.7	37.8	33.4	983.7	125.0
7	149.1	57.1	175.2	24.2	-9.1	35.4	35.6	968.6	147.5
8	1635.6	191.2	126.2	122.9	-16.8	37.1	33.3	984.8	126.9
9	1543.9	1089.4	153.2	-156	-19.2	37.7	33.6	984.7	127.8
10	186.5	1070.3	76.3	-36.7	-9.8	37.1	33.5	991.5	130.7

Tableau A.2: Résultats de placement par la RFS pour $C_{seuil} = 2$

Caméra N°	X [cm]	Y [cm]	Z [cm]	φ_h [°]	φ_v [°]	α_h [°]	α_v [°]	d_{max}	d_{min}
1	1367.2	1034.9	18.4	-135.1	1.8	35.7	32.2	757.5	172.5
2	937.0	558.8	75.0	39.7	-8.5	38.9	34.1	851.9	154.3
3	1313.6	480.0	55.7	68.3	-10.3	36.5	39.1	913.6	192.9
4	1211.1	197.7	35.5	156.9	-5.5	33.8	37.2	868.5	199.6
5	1272.4	137.9	28.3	125.0	10.5	35.6	35.1	730.1	179.9
6	369.1	1020.5	3.2	-78.9	-3.8	34.1	35.9	920.2	136.9
7	142.7	652.0	13.5	-13.3	5.1	36.4	33.8	795.0	134.1
8	1668.8	1136.0	76.5	-146.9	3.6	38.8	39.2	940.6	183.7



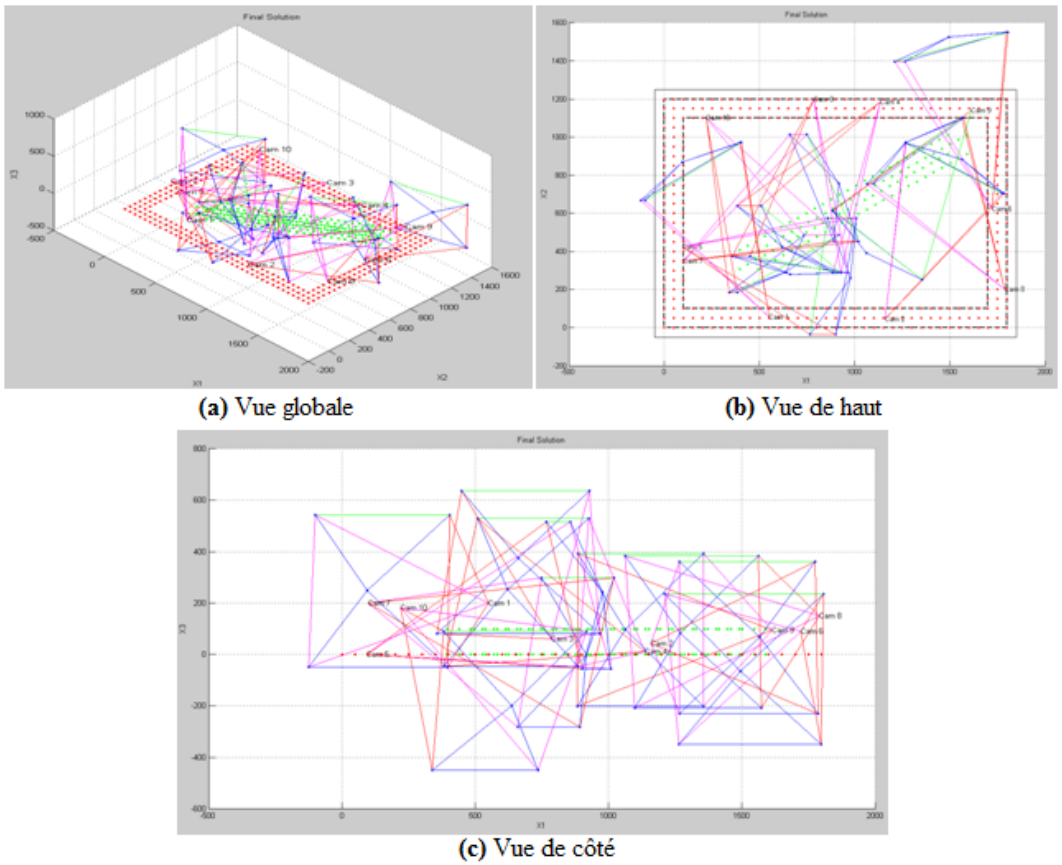
(a) Vue globale **(b)** Vue de haut

(c) Vue de côté

Figure A.1: Déploiement final du réseau de huit caméras par la RFS pour $C_{seuil} = 2$.

Tableau A.3: Résultats de placement par la CPSO pour $C_{seuil} = 3$

Caméra N°	X [cm]	Y [cm]	Z [cm]	φ_h [°]	φ_v [°]	α_h [°]	α_v [°]	d_{max}	d_{min}
1	678.1	50.0	99.1	71.2	8.3	34.2	33.8	846.9	169.1
2	948.9	85.3	84.3	70.5	12.4	34.1	34.7	847.1	170.2
3	411.2	1100	160.6	-67.8	9.6	33.9	34.4	846.9	168.6
4	1208	1100	91.5	-129.1	0.0	35.2	33.8	855.5	169.0
5	17.0	465.6	198.5	0.6	-13.0	33.6	33.9	854.5	167.8
6	1700	800	146.4	-165	-16.3	35.4	34.4	853.1	176.8
7	299.7	18.1	25.1	61.4	7.4	34.4	33.8	850.5	169.4
8	1729.2	220.3	196.8	115.7	-21.1	35.0	33.8	854.3	168.5
9	1700	853.6	93.7	-154.1	15.9	35.2	33.8	867.2	167.6
10	28.4	936	102.2	-63.5	14.3	35.2	33.7	849.4	169.7



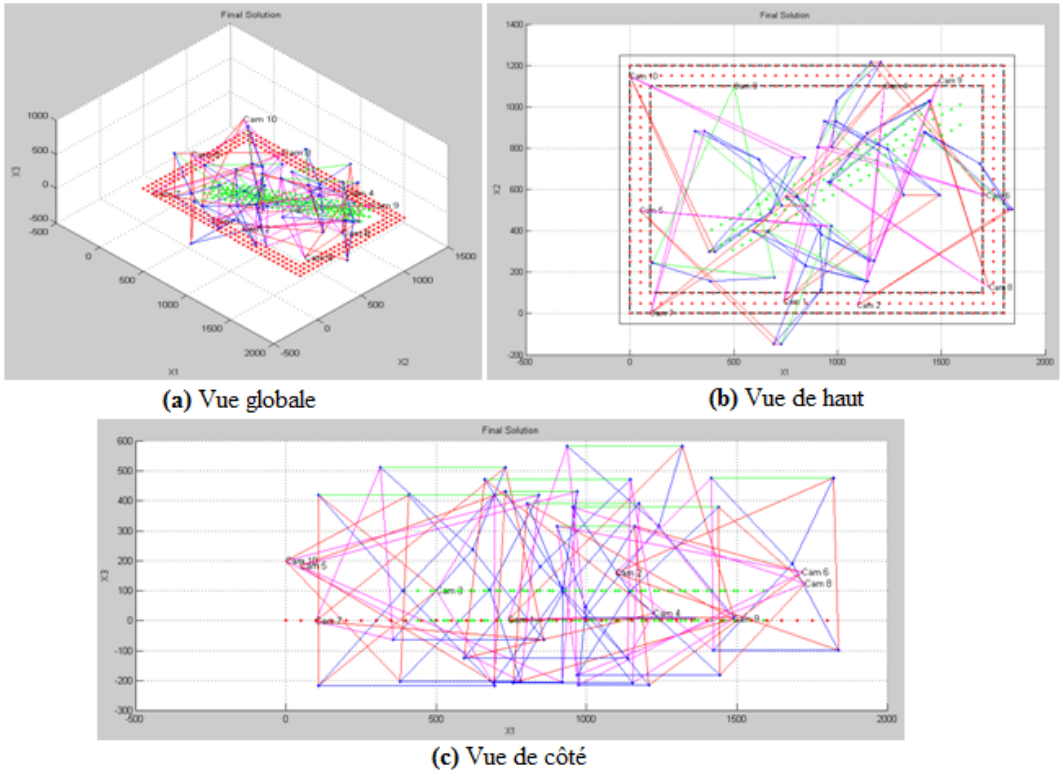
(a) Vue globale (b) Vue de haut

(c) Vue de côté

Figure A.2: Déploiement final du réseau de dix caméras par la CPSO pour $C_{seuil} = 3$.

Tableau A.4: Résultats de placement par la MCPSO pour $C_{seuil} = 3$

Caméra N°	X [cm]	Y [cm]	Z [cm]	φ_h [°]	φ_v [°]	α_h [°]	α_v [°]	d_{max}	d_{min}
1	739.0	56.5	3.3	54.2	20.0	39.3	39.1	911.5	131.7
2	1095.2	40.8	158.7	49.3	2.0	37.0	37.2	903.5	132.9
3	500.0	1100.0	100.1	-96.7	0.1	38.6	39.1	949.4	151.2
4	1224.5	1100.0	24.3	-113.9	9.6	36.9	37.0	949.2	121.3
5	48.1	497.4	183.4	-23.4	-4.4	38.1	39.1	955.2	132.0
6	1717.0	568.8	161.3	147.6	-7.8	32.9	36.3	860.9	137.5
7	100.1	0.3	-0.1	54.1	14.9	39.1	37.6	920.4	131.7
8	1727.3	125.5	123.8	127.9	-1.5	39.3	34.3	950.3	120.3
9	1486.7	1127.3	05.0	-127.3	5.6	31.8	37.4	937.5	110.8
10	0.6	1149.7	200.0	-44.1	-5.8	39.2	38.5	948.5	132.5



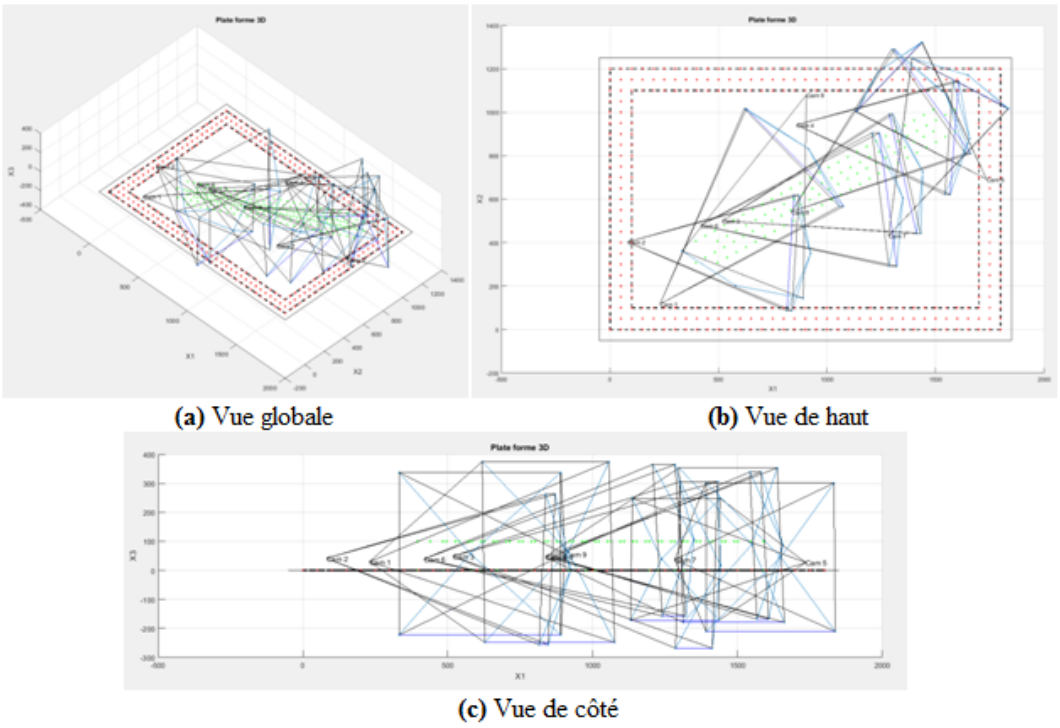
(a) Vue globale **(b)** Vue de haut

(c) Vue de côté

Figure A.3: Déploiement final du réseau de dix caméras par la MCPSO pour $C_{seuil} = 3$.

Tableau A.5: Résultats de placement par la RFS pour $C_{seuil} = 3$

Caméra N°	X [cm]	Y [cm]	Z [cm]	φ_h [°]	φ_v [°]	α_h [°]	α_v [°]	d_{max}	d_{min}
1	230.1	117.2	29.9	46.1	2.0	38.2	36.7	989.8	130.7
2	81.3	402.8	41.7	-3.6	-2.7	36.7	35.8	845.5	182.4
3	517.4	500.4	48.7	13.7	-1.9	34.5	35.2	952.0	199.8
4	860.1	941.9	45.9	-4.6	3.0	38.0	37.0	800.7	125.3
5	1735.9	691.8	27.1	135.1	1.0	37.4	34.8	703.3	126.2
6	417.9	479.5	39.3	7.7	4.1	38.3	32.3	942.7	157.1
7	1283.4	432.4	36.9	63.6	0.6	36.9	36.2	824.0	191.3
8	836.6	543.0	46.2	37.0	2.7	39.2	34.7	893.8	195.1
9	904.8	1079.6	57.0	-110.4	0.0	39.0	34.8	936.1	170.0



(a) Vue globale **(b)** Vue de haut

(c) Vue de côté

Figure A.4: Déploiement final du réseau de neuf caméras par la RFS pour $C_{seuil} = 3$.