Thesis submitted for obtaining the degree of

doctorate in electrical engineering.

Option: Signals and Communications.

# Metaheuristics for Robust Object Tracking in Video Sequences (Multi-object real time tracking)

Presented by

**CHAREF-KHODJA Djemai.**

Thesis defended publicly on: 22/03/2022

The jury members:

| | | | |
|---|---|---|---|
| President: | Pr. BAARIR Zineddine | Prof | University of Biskra |
| Supervisor: | Pr. TOUMI Abida | Prof | University of Biskra |
| Co-supervisor: | Pr. SBAA Salim | Prof | University of Biskra |
| Examiner: | Pr. ABDOU Latifa | Prof | University of Batna |
| Examiner: | Pr. BENZID Redha | Prof | University of Batna |
| Examiner: | Pr. TERKI Nadjiba | Prof | University of Biskra |

University year: 2021-2022

# DEDICATION

I dedicate this modest work to:

My beloved parents: father and mother.

May Allah grant them health, and happiness.

My brothers, sisters, and my children.

My wife.

For her precious assistant and encouragement, when I needed moral support.

All my friends and colleagues.

All my teachers of the department of electrical engineering.

All those I respect and appreciate.

# ACKNOWLEDGEMENTS

First and foremost, I gratefully acknowledge my gratitude and indebtedness to ALLAH.

Then, I would like to extend my warmest thanks and appreciation to my supervisor Pr. TOUMI Abida, for her endless support, encouragement, guidance, and advices in academic avenues.
For my co-supervisor Pr. SBAA Salim, for his support and guidance, to complete this thesis.

I also thank them, for their patience and for the trust they have always placed in me.

I thank all the thesis jury members, who agreed to judge my work:

- Pr. BAARIR Zineddine.
- Pr. ABDOU Latifa.
- Pr. BENZID Redha.
- Pr. NADJIBA Terki.

I also want to express my sincere thanks and gratitude in particular to Dr. MEDOUAKH Saadia, who guided me at the beginning of my work.

## ABSTRACT

In these recent years, object tracking has attracted the interest of many researchers, because it offers many challenges as a scientific problem, as well as its potential industrial and commerce applications in human life, robotics, and surveillance. Moreover, it is a part of many prominent levels in computer vision problems, such as motion analysis, and activity recognition. Visual object tracking is a challenging task, as many approaches have been proposed, it is still unresolved problem because of the high number of the challenging factors. By considering the research trends in this line, three lines of researches were developed namely: the object presentation, the search mechanism, and the updating model. In this thesis, we are specifically focused in the study and the developing of metaheuristic based searching techniques used for tracking. The role of metaheuristic search algorithm is to find the most similar candidate to a previous defined template. Many similar related works have been proposed in this line, their main disadvantage is the convergence at local minima which make them unable to find the exact position; to overcome this drawback, we proposed four different tracking frameworks, whose three, are single object tracking based methods, and one is multi-object tracking based method. The first part of this thesis is dedicated to implement Stochastic Fractal Search (SFS) algorithm, study and analyze the effect of using different population sizes; also the implementation results of the Harris Hawks Optimizer (HHO), in the same context. When the second part of this thesis is dedicated to implement, study and analyze the effect of using both population sizes, and iterations number on the tracking accuracy with WOA algorithm, combined with a very discriminative appearance model, to improve the efficiency and robustness of the tracking results under difficult environment, such as blurred motion, small similarity between the target object and the background, and the changes in illuminations. The last and the third part of this thesis is dedicated to the implementation of Equilibrium Optimizer (EO) on multi-object tracking framework.

**Keywords:** Visual object tracking, SOT, Multi-object tracking, MOT, SFS, HHO, WOA, EO, CBWH, Metaheuristics.

**Résumé**

Dans ces dernières années, le suivi d'objets a attiré l'attention de nombreux chercheurs, car il présente de nombreux défis en tant que problème scientifique, ainsi que ses applications industrielles et commerciales potentielles dans la vie humaine, la robotique et la surveillance. De plus, il fait partie de nombreux niveaux importants dans les problèmes de vision par ordinateur, tels que l'analyse de mouvement et la reconnaissance d'activité. Le suivi visuel des objets est une tâche difficile, bien que de nombreuses approches ont été proposées, ce problème reste non résolu en raison du nombre élevé de facteurs difficiles. En considérant les tendances de recherche dans cette ligne, trois axes de recherche ont été développés à savoir: la présentation de l'objet, le mécanisme de recherche et le modèle de mise à jour. Dans cette thèse, nous nous concentrons spécifiquement sur l'étude et le développement de techniques de recherche méta-heuristiques utilisées pour le suivi. Le rôle de l'algorithme de recherche métaheuristique est de trouver le candidat le plus similaire à un modèle défini au préalable. De nombreux travaux similaires ont été proposés dans cette ligne, leur principal inconvénient est la convergence aux minima locaux qui les empêchent de trouver la position exacte; pour surmonter cet inconvénient, nous avons proposé quatre cadres de suivi différents, dont trois sont des méthodes basées sur le suivi d'un seul objet, et un est une méthode basée sur le suivi multi-objets. La première partie de cette thèse est consacrée à la mise en œuvre de l'algorithme de recherche fractale stochastique (SFS), à l'étude et à l'analyse de l'effet de l'utilisation de différentes tailles de population; également les résultats de la mise en œuvre de l'optimiseur des faucons de Harris (HHO), dans le même contexte. La deuxième partie de cette thèse est consacrée à implémenter, étudier et analyser l'effet de l'utilisation à la fois de la taille de la population et du nombre d'itérations sur la précision de suivi avec l'algorithme d'optimisation de baleine (WOA), combiné avec un modèle d'apparence très discriminant, pour améliorer l'efficacité et la robustesse du suivi dans un environnement difficile, tel que la présence d'informations de base importantes dans le l'objet modèle, petite similitude entre l'objet cible et l'arrière-plan et les changements d'illumination. La dernière et la troisième partie de cette thèse sont consacrées à l'implémentation de l'optimiseur d'équilibre (EO) sur un cadre de suivi multi-objets.

**Mots clés:** Suivi d'objet visuel, Suivi multi-objet, SFS, HHO, WOA, EO, CBWH, Méta-heuristiques.

# الملخص

في هذه السنوات الأخيرة، اجتذب تتبع الكائنات اهتمام العديد من الباحثين، لأنه يطرح العديد من التحديات كمشكلة علمية، إضافة إلى التطبيقات الصناعية والتجارية المحتملة في حياة الإنسان، كالروبوتية، والمراقبة. علاوة على ذلك ، فهو جزء من العديد من المستويات البارزة في مشاكل الرؤية للكمبيوتر، مثل تحليل الحركة والتعرف على النشاط. يعد تتبع الكائن المرئي مهمة صعبة للغاية، بالرغم من أنه تم اقتراح العديد من التقنيات، غير أنه لا تزال مشكلة لم يتم حلها بسبب العدد الكبير من العوامل الصعبة. من خلال النظر في اتجاهات البحث في هذا الإطار، تم تطوير ثلاثة محاور للبحث وهي: طريقة عرض الكائن، آلية البحث، ونموذج تحديث القالب. في هذه الأطروحة، نركز بشكل خاص على دراسة وتطوير تقنيات البحث القائمة على تقنيات الاستدلال الفوقي (metaheuristic) المستخدمة في التتبع. يتمثل دور خوارزمية البحث كهذه، في العثور على المرشح الأكثر تشابهًا مع قالب محدد سابقًا. تم اقتراح العديد من الأعمال المماثلة ذات الصلة في هذا المحور، ومن عيوبها الرئيسية الوقوع في حلول محلية غير شاملة، مما يجعلها غير قادرة على العثور على الموضع الصحيح بدقة؛ للتغلب على هذا العيب، اقترحنا أربعة أطرعمل مختلفة للتتبع، ثلاثة منها، هي أساليب تعتمد على تتبع كائن واحد، وواحدة تعتمد على تتبع متعدد الكائنات. الجزء الأول من هذه الأطروحة مخصص لتطبيق خوارزمية البحث العشوائي الفركتالي (SFS)، مع دراسة وتحليل تأثير استخدام أحجام مختلفة للسكان. أيضًا نتائج تطبيق محسن صقور هاريس (HHO)، في نفس السياق. تم تخصيص الجزء الثاني من هذه الأطروحة لتنفيذ ودراسة وتحليل تأثير استخدام كل من أحجام السكان ، وعدد التكرارات على دقة التتبع باستخدام خوارزمية محسن حوت البالين (WOA) ، مركبا مع نموذج مظهر شديد التمييز، لتحسين كفاءة ومتانة تتبع النتائج في ظل بيئة صعبة، مثل وجود معلومات بارزة في الكائن القالب، والتشابه القليل بين الكائن الهدف والخلفية والتغيرات في الإضاءة. تم تخصيص الجزء الأخير والثالث من هذه الأطروحة لتطبيق خوارزمية محسن التوازن (EO) في إطار عمل تتبع الكائنات المتعددة.

**الكلمات الاستدلالية**: تتبع الكائن المرئي، تتبع متعدد الكائنات، الميتا استدلالية, CBWH, WOA, EO, SFS, HHO

## Publications in journals

- **Charef-Khodja, D.**, Toumi, A., Medouakh, S., Sbaa, S.: A novel visual tracking method using stochastic fractal search algorithm. SIViP 15(2), 331-339 (2021).
- **Charef-Khodja, D.**, Toumi, A., Medouakh, S., Sbaa, S.: Efficient visual tracking approach via whale optimizer and corrected background weighted histogram. Multimed Tools Appl 80(14), 21381-21407 (2021).

## Publication in international conference

- **Charef-Khodja**, **D.,** Abida, T., Sbaa, S., Medouakh, S.: Robust visual tracking method based on Harris Hawks algorithm. In: 020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), IEEE. 180-185 (2020).

# Table of contents

# List of figures

# List of tables

# List of acronyms

| Acronyms | |
|---|---|
| AI | Artificial intelligence |
| BA | Bat algorithm |
| CBWH | Corrected background weighted histogram |
| CS | Cuckoo search |
| DBT | Detection based tracking |
| DFT | Detection free tracking |
| DPR | Distance precision rate |
| DSST | Discriminative Scale Space Tracker |
| EO | Equilibrium optimizer |
| Fig | Figure |
| FPA | Flower pollination algorithm |
| HHO | Harris hawks optimizer |
| HOG | Histogram of orient gradient |
| HSV | Hue saturation value color space |
| KCWH | Kernel color weighted histogram |
| KLT | Kanade-Lucas-Tomasi |
| KMS | Kernel mean-shift |
| LBP | Local binary pattern |
| LPQ | Local phase quantization |
| MB-LBP | Multi-Block LBP |
| MFPA | Modified flower pollination algorithm |
| MOT | Multi-Object tracking |
| MTT | Multi-target tracking |
| NCC | Normalized Cross-Correlation |
| OPE | One pass evaluation metric |
| OSR | Overlap success rate |
| OTB | Object tracking Benchmark |
| PSO | Particle swarm optimization |
| RGB | Red Green Blue color space |
| SFS | Stochastique fractal Search |
| SOT | Single Object tracking |
| Struck | Structured Output Tracking with Kernels |
| SVM | Support vector machine |
| TLD | Tracking Learning and Detection |
| VOT | Visual Object tracking |
| WOA | Whale optimization algorithm |

# Introduction

Computer vision is the key of any artificial vision system, as it is a branch of artificial intelligence; its objective is to allow a machine to analyze and processes, many data within images taken by camera(s). Object tracking is one of the most important and difficult research topics in the computer vision field as it has been successfully applied in several ranges of real world applications, such as video surveillance, and robotics, etc.

Nowadays, tracking objects in video footage is ranked among the most active research field. The main purpose of tracking is to estimate over time the location of the target object in each frame of a video clip.

Tracking objects whether single or multiple, remains a complex and difficult task with no last words, and far from being closed, and this is due to the many challenges [1-3], related to the limitations of vision sensors (low resolution, low frame rate, and noise, etc.), objects (non-rigid objects, occlusions between objects, size changes, etc.), the requirements of application scenarios (real-time applications, etc) and for environment (lighting conditions, occlusions by the environment, etc.). To face these challenges many object tracking algorithms have been proposed in the literature, to ensure good tracking results.

By considering the progress in this field, online tracking algorithms typically include three fundamental lines of research which present on most of the cases the core components of any tracking algorithm, namely: the appearance model, the search mechanism, and the update model [1].

Through the appearance model, some authors proposed to use generative models, discriminative models, or even a hybrid models which is a combination of the two previous models [1].

By looking at the research trends in the search mechanisms, we can categorize them into three categories [2], according to their mechanisms; they can be deterministic [4, 5], probabilistic [6-7], or metaheuristic [2, 8, 9]. Deterministic methods are iterative approaches which maximize a fitness function between a template and candidates in each frame [1]. Probabilistic methods consider the searching process as a state of solving a problem under Bayesian framework[6], while metaheuristic techniques are another form of stochastic approaches, but with a higher level of optimization, designed to find the most similar candidate to a predefined template, by maximizing or minimizing a fitness function[2,9].

## Problematique

Deterministic methods such as, mean-shift (MS) [4,5], trust region [10] and Snakes model [11], are not effective, and easily got distracted by object's size change, out of plane rotation, or full occlusion, while probabilistic mechanisms often suffer from getting trapped in a local optimum [12]. Compared to object appearance researches, there are not so many on search models [8]. Formulating localization problem in video sequences can be interpreted as an optimization problem, and solved using metaheuristic algorithms. And it is in this context that our study was developed.

Metaheuristics are stochastique optimization techniques, allowing the obtainment of satisfying optimization results within a reasonable time. They are based on the minimization or maximization of an objective function, and on random mechanisms to explore the search space, in order to determine or get close to a global optimum.

Metaheuristics have been applied successfully in several research fields such as medical applications networks, telecommunications, multimedia applications, image processing and especially in object tracking. In this line, we decided to implement four tracking methods based on metaheuristic techniques. By solving the problem of tracking objects in video frames using metaheuristic algorithms, and finding out the most similar candidate(s) to the target object(s) through an effective expression of the object, so far we obtain the complete motion trajectory of the moving object(s).

We aim in this thesis to conceive new robust object tracking methods based on metaheuristic techniques which are capable of obtaining satisfying tracking results in terms of precision, and under difficult environment.

## Contributions

The main objective of this work is to propose object tracking techniques by combining powerful metaheuristic algorithms, as a searching techniques, with color histogram as an object presentation, firstly in two generative approaches [2, 13], with statistical studies of the parameters tuning on the performance of the metaheuristic based trackers, secondly in a discriminative approach [9], and at last in a multi-object tracking approach.

Our main contributions include:

- Two generative single object tracking approaches:

  1. By using stochastique fractal search (SFS), and study its population size parameter, for better tracking results and we got good tracking performance.
  2. By using Harris Hawks optimizer (HHO) algorithm and we got promising tracking results.

- One discriminative single object tracking approach, by combining a new tracking algorithm: whale optimization algorithm (WOA) with a discriminative feature called as "corrected background weighted histogram" (CBWH); WOA parameters such as, population size and number of iterations are studied experimentally and statistically in an original way.

- Apply a new metaheuristique algorithm called as: Equilibrium Optimizer (EO) in multiple objects tracking framework.

This thesis is structured as follows:

- Chapter 1 is an introduction to visual tracking, by giving an overview of this field; the chapter illustrates the state of the art of single object tracking (SOT) techniques, their classifications, and their main components.
- Chapter 2 gives an overview of the most commonly datasets, used in the field of SOT.
- Chapter 3 is reserved for multi-object tracking techniques, their applications, along with their challenges and categories, in order to show the diversity of the design of the developed approaches. The chapter also discusses the available multi-object tracking datasets, and their evaluation metrics used to assess the performance of individual trackers.
- Chapter 4 is devoted to metaheuristic techniques. First we describe the optimization problem, and then we mainly focus on metaheuristic optimization techniques by detailing its principles, and their classifications.
- Chapter 5 presents the experimental results of our four contributions including the application of SFS, HHO, and WOA metaheuristic algorithms for SOT, and the EO for multi-object tracking. In this chapter, we are interested in understanding their application combined with color histogram as object appearance model, as well as studying and analyzing the different tuning parameters, for better tracking efficiency and accuracy. The comparison of the effectiveness of our proposed approaches with other state-of-the-art methods are presented and discussed using the OTB2015 dataset.
- Finally, we conclude the work of this thesis and we present future works that can be carried out.

# CHAPTER 1

# STATE OF THE ART OF SINGLE OBJECT TRACKING TECHNIQUES

## 1.1. Introduction

Before talking about object tracking, it is necessary to make a brief introduction to the computer vision's field. To understand computer vision definition, we need to understand what human vision is. There is a saying that: "**to see is to know what is where by looking**". The aim of computer vision field is to solve the same visioning problem using a computer. Consequently, computer vision field is considered as a part of artificial intelligence (AI) domain. Therefore **a computer must demonstrate human intelligence**, by answering many questions regarding an image that can be answered by a human. The most asked questions about an image, are what objects are seen in the image and where are they located? If these questions are asked for a video sequence, this is called object tracking.

Because we are humans, some objects are more important for us than others, and the most important object for us are humans and their faces. This is why pedestrian and face are the most rigorously searched objects in the field of object detection & tracking, and also the most rigorously researched topics in the computer vision area.

Object tracking has always been one of the most popular research areas, and remain the core computer vision problem, owing to the significant commercial potential of tracking applications based, as well as the research challenges this field offers. Through examining the literature, compared to SOT, where lots of source codes are publicly available, there seem not many public sources for multi-object tracking (MOT); certainly, the progress in SOT is larger than that of MOT [3]. One reason can be that, many researchers in SOT have made their codes publicly available, as opposed to MOT researchers [3].

Although object tracking is a promising and active field of research, books in this area are very scarce; at present the most recent book for object tracking have just been published in 2019 by Huchuan Lu et al.[1], the book provides excellent support for recent research and trends in the field of online single object tracking. Before 2019, the most cited works of surveys and reviews related to the state of art of object tracking are of two, the works are done by Yilmaz et al. in 2006 [14], and Li et al. in 2013 [15], but their main disadvantages are the categorizations, definitions, and also the references are very old.

In this thesis, we will base our categorization according to the new book "**Online object**

**tracking**"[1], and on the most recent papers, in order to provide the most recent tracking algorithms classifications and definitions.

This chapter is devoted to SOT framework, including definitions of the object tracking problem, a description of the key components that any tracker should have, and the difficulties that tracking algorithms are likely to encounter.

## 1.2. Object tracking definition

Let's consider a video sequence with some moving objects inside; the video sequence can be captured using static or moving camera; then the process of object tracking can be defined as the process of state estimation of moving object [14, 15] or multiple objects over time. The output of the tracking algorithm is the "**object track**" which can be defined as the object's states (e.g., position and size) in each frame of the video sequence. Depending on number of objects to be tracked, object tracking problem can be classified into: single object tracking (SOT), or multi-object tracking (MOT), each of both have different challenges and applications.

Due to object tracking’s close relationship with object's detection and classification, and the ambiguity caused by the diversity of definitions and vocabularies that can be found in the literature, and may generate possible confusion of these terms, we have agreed on the following definitions:

### 1.2.1. Object detection

Object detection is one of the most important key problems in the computer vision field. The aim of the object detection process is to disclose the existence of object(s) from a certain class [16], such as: face, pedestrian, dog, text, etc; and to find their exact position in the image. The detected objects can be of certain shape and size.

The object detector algorithm constructs a feature model for one object class by using a set of training examples [16]. We can specify the location of the object with a bounding box like in Fig.1.1, described by the position of one of the corners, and the width and height of that box.

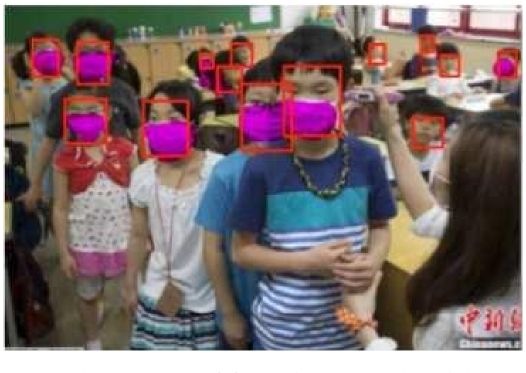


Fig. 1.1 The output of face detector algorithm[17].

"It is worth mentioning here that tracking by detection is a term used **in general**, for a certain type of multi-object tracking technique".

### 1.2.2. Object classification

The process of comparison and assigning to each located object its proper class name is called as classification; the various objects within an image have different properties presented by their coded features, the classification process is carried by comparison with those in a database [18].

Compared to **object detection algorithm,** the output of the object classifier algorithm is **structured**, and every object is generally marked with a bounding box and class label Fig.1.2.

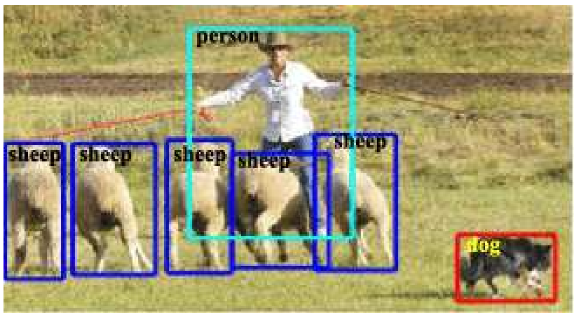


Fig.1.2 An example of the output of object classifier algorithm [19]

## 1.3. Single object tracking are model free

It is worth mentioning that, when we simply put: "**Visual Object Tracking (VOT)**" we are implicitly referring to the problem of **online single object tracking (SOT)** in the video sequences.

To the best of our knowledge all single object tracking methods are considered model free or detection free trackers. **The model-free term**, means that there is no specific model for the object, and the target object is specified only by the bounding box in the first frame of the video, and the only information we have, is the object's location in the first frame.

Initially and before **2018**, most SOT algorithms are considered model-free **short term**, and causal trackers [14, 15, 20-27]; because the appearance of the object is changing over time, we generally assume a **short-term** tracking and in a short video sequences, thus we cannot build a detector or redetect the target object in other sequences if the object is lost. Causality or online tracking means that the tracker cannot use any future frames.

In 2018, the VOT tracking algorithms [28-30] were extended to another category called as: model free long-term trackers. Long-term tracking algorithms refer to the trackers that employ re-detection mechanism if the target has been lost.

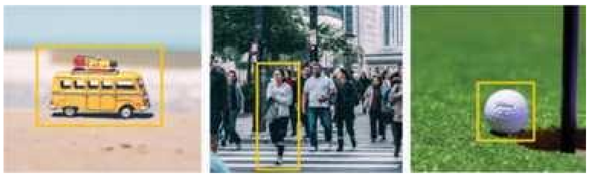


Fig.1.3 Model free single object tracking

## 1.4. Single object tracking applications

Object tracking is among the most studied issues in the recent years, as it has great interest in the past decade due to the variety of its fields of application [14,20,31,21], such as:

- Military applications (ex: missile target tracking).
- Sport analysis.
- Robotics: following a target and avoidance of obstacles.
- Monitoring of visual cues in a task controlled by vision.
- Medical imaging.
- Recognition based on motion.
- Video indexing, for automatic retrieval and annotation of the video's data in video databases.
- Human-computer interaction, such as: gesture recognition, eye gaze tracking for identification's system.

  Fig.1.4 depicts some of these applications

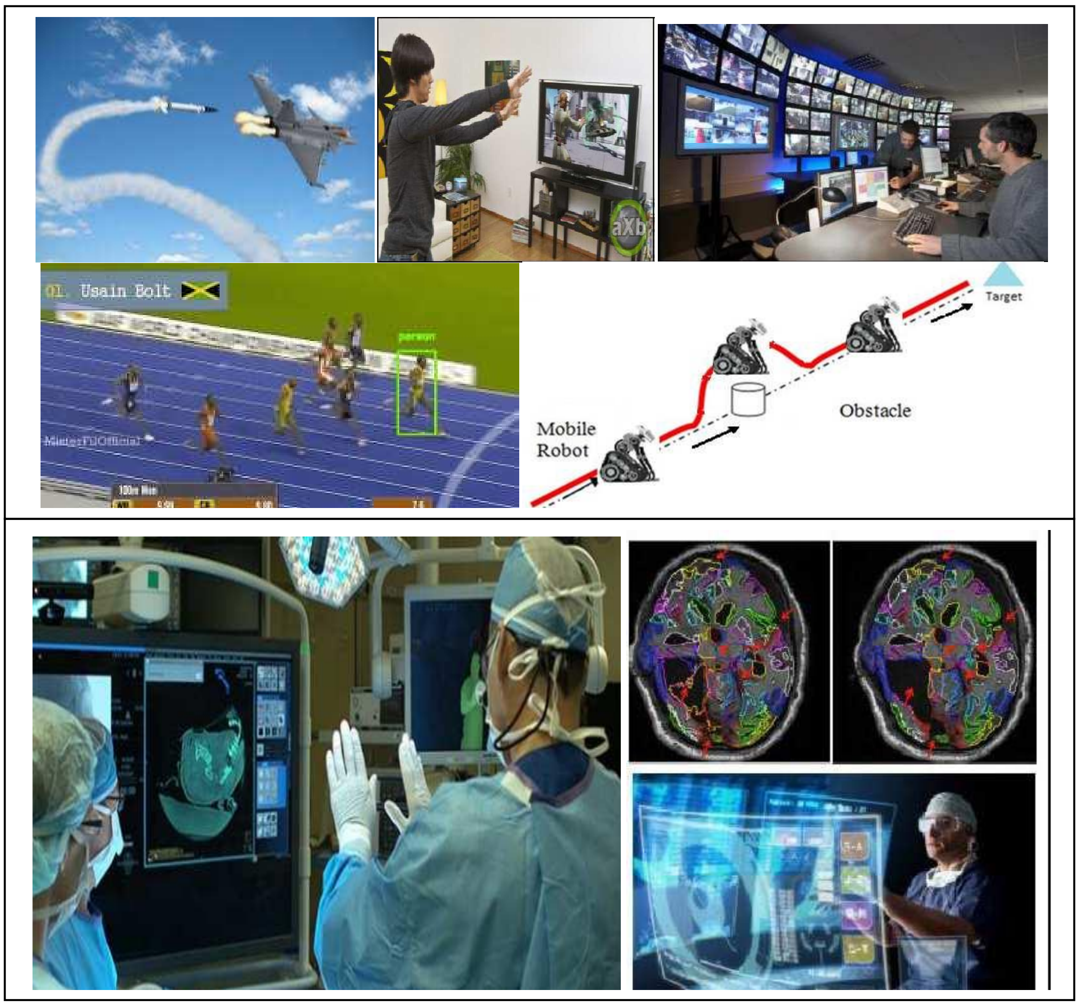


Fig.1.4 Some SOT applications.

## 1.5. Single object tracking challenges

Because the target objects inevitably undergo different changes over time. There are numerous challenges in developing SOT methods, the most common challenges are, either defined in the literature or in datasets such as: OTB Benchmark[20], or VOT[23-30], these factors influence the performance of SOT algorithm such as:

- **Partial or full occlusion**: object is partially or completely occluded in some frames.
- **Blurred motion**: the object may be blurred due to its fast motion or camera motion.
- **Size changes**: object in a frame can change scale dramatically, due to camera zoom, or because of approaching or moving away from the camera for example.
- **Illumination changes**: lighting conditions in a frame can have a big effect on how objects look and can make it harder to consistently track them.
- **Computational load:** this problem arises in real-time processing, because we need to process a whole lot of frames.
- **Rotation:** there are two types of rotation: in plane rotation, where the target rotates in the image plane; and out of plane rotation where the target rotates out of the image plane.
- **Background clutter:** The background of the object can be similar to the color or the texture of the target object of interest, or also can be similar to other object; in this case, the SOT algorithm have to discern between these different objects.
- **Low resolution:** the number of pixels inside the bounding box of the target object is very small.
- **Object is out of view:** The target object may leave the scene for a long time.
- **Fast motion:** the motion of the object between adjacent frames is very fast.
- **Object noise:** the target object may suffer from a noise caused by the environment or by to the quality of camera.
- **Camera motion:** tracking using a fixed camera is a way easier than a moving camera.
- **Deformation:** the target object is non-rigid.

Fig.1.5 shows some difficulties taken from the OTB2013 [20] dataset.

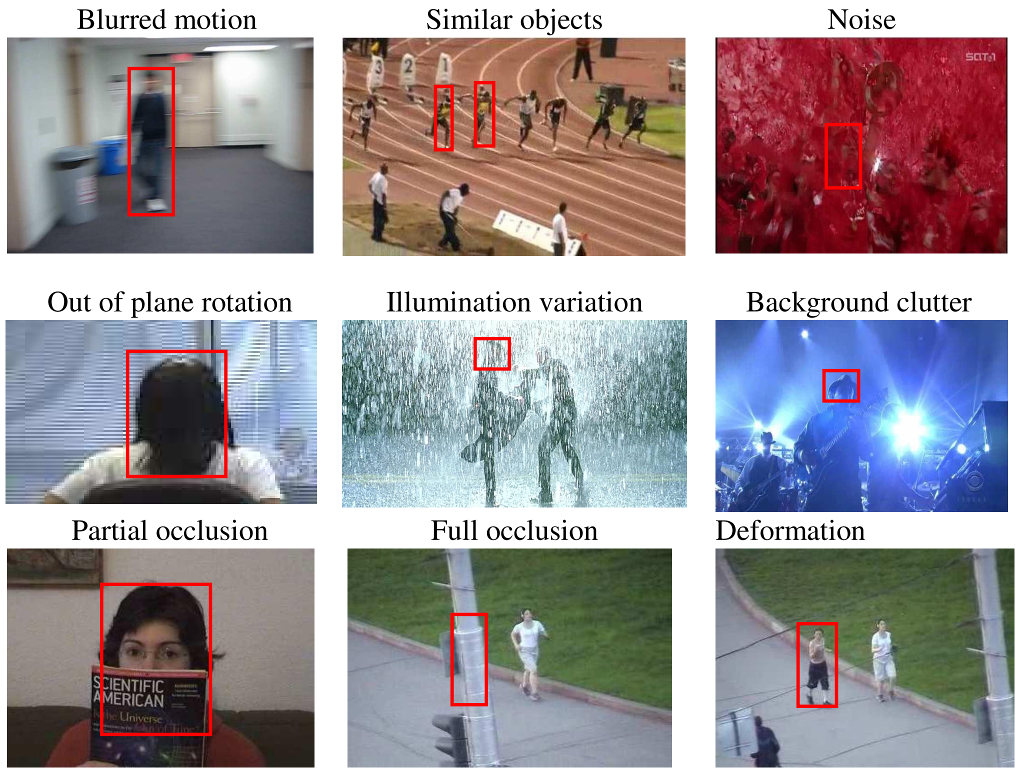


Fig.1.5 Some difficulties in SOT [20]

## 1.6. Basic components of single object tracking algorithm

Object tracking in video sequences is a problem that requires the extraction and processing of information from complex images. There is a large number of tracking methods in the literature. This number is due to the large number of problems to be solved as well as to the diversity of the applications' types. Thus, each of the methods can deal with some applications and fail in others.

By considering the development in this field, and as it is already mentioned in the introduction section, SOT algorithms are of three main components namely [1]:

1. The feature presentation
2. The searching mechanism
3. The model update

### 1.6.1. The feature presentation

According to Yilmaz et al. [14], an object can be presented by its shape for example in Fig.1.6: Point (a, b), primitive geometric shape (c, d), articulated (e), skeletal (f), or silhouette and contour (g, h, i).

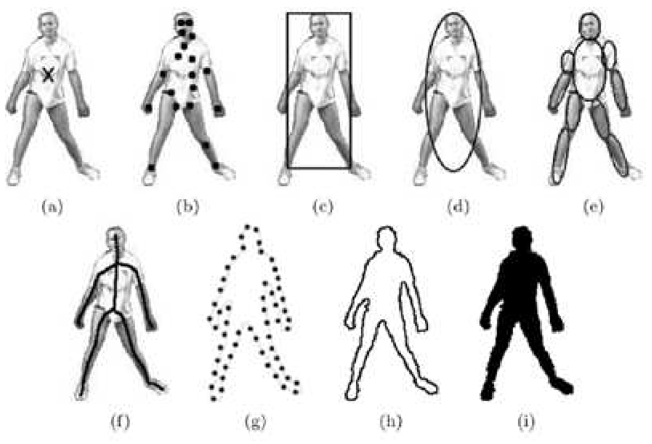


Fig.1.6 Object presentations by shape [14]

In Fig.1.6 (c), the rectangle shape is the most recent trend and the most used presentation used for visual tracking.

Selecting the right features plays an important role in conceiving any tracking algorithm. In most of the cases, the visual feature used for tracking falls into one of two categories [1, 19, 31]; they are either generative, or discriminative.

#### 1.6.1.1. Generative methods

Generative methods formulate the tracking problem as finding image regions that are most similar to a target model by considering only the bounding box information of the target in interest [1, 32].

Object generative representation is one of the most used types for visual tracking systems; the main generative appearance models that can be found in the literature are:

##### a. Grey level template

Grayscale template is the most basic example of representation of an object and basic approach for correlation-based tracking [33]. The tracking based technique is called as: template matching (TM); in this technique we can search for the new position of the template in the next frame by brute force method of searching through scanning the whole image by sliding window.

When doing the sliding window, we make a comparison per-pixel between the template and candidate and generate a new image, which maps the distance metric results. The minimum is the distance in the image map, which gives the new position of the template.

The Sum squared error, and the normalized cross correlations (NCC) are two examples of such metrics. Matlab software help mentioned that template matching technique, using NCC metric is the baseline visual tracking method. Additionally, the map image based on NCC can locate objects in texture-rich images.

Despite its simplicity, such method can be efficient for short term tracking in some real world applications; such as to track people faces between detections, or a very simple smart-phone remote control for taking selfie photos, by tracking of a human palm, like in Fig.1.7 & Fig.1.8, where Fig.1.7 represents an advertisement of a smartphone.

TM tracking drawback is that it is an exhaustive tracking technique, because it becomes computationally expensive when the template size or the search region is large. However, more effective tracking algorithms based on TM have been proposed [34, 35]. NCC metric is one of the important components in many developed trackers such as tracking learning detection (TLD) tracker [36] and parallel robust online simple tracking (PROST) tracker [37]. D. Oliva et al. [38] used TM with electromgnetique optimizer (EMO) in visual object tracking.

Another limitation of this presentation is in the deformation of non rigid objects, and the rotation which cause a lot of information loss.

Fig.1.7 Selfie photo taking by tracking palm hand.

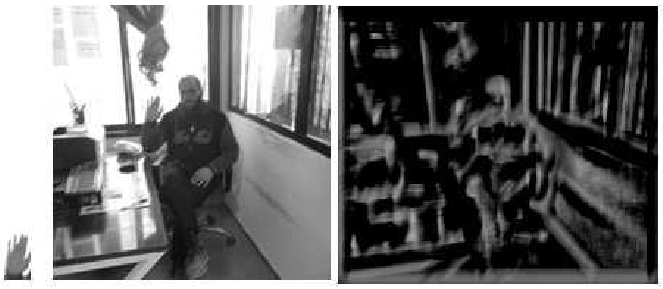


Fig.1.8 Palm hand as object template

From the left to the right, the palm hand, the test image, and the image score map.

### b. Optical flow

Since the first work of Lucas-Kanade (LK) [39], on optical flow, (based on raw intensity values) holistic template has been widely used for SOT [40]. However, when the visual properties of a target in interest change significantly, the LK approaches do not work well; to tackle this problem Matthews et al. [41] developed a template update approach by exploiting the first frame's information to correct drifts. Subspace-based presentation is a weighted sum of several feature vectors, where each vector model an appearance feature have been proposed to effectively account for appearance changes; in this line Hager and Belhumeur [42], proposed an efficient tracking algorithm using LK and low dimensional representations for tracking under different illumination conditions to improve tracking robustness. Misra, R et al. [43], used Quantum particle swarm optimizer (QPSO), through calculating dominant points of object using LK optical flow algorithm; their algorithm was tested on both static and dynamic environment and they have achieved 90% faster results by applying QPSO approach compared to the basic PSO.

### c. Color

Color is the most intuitive and used feature in the literature for a large number of tracking methods, because it is the most conserved feature of the object. Color feature is invariant to object shape changes, but its robustness decrease when facing regions with similar color distributions. In the literature, Color feature can be exploited using two methods:

**1-Kernel color weighted histogram (KCWH):** This presentation is the same as used with mean-shift algorithm by comaniciu et al. [5], and it is the most used for tracking. Collins [44] improved the mean shift tracking algorithm to handle the size variation of the object in interest.

Another trend used so much this presentation, by combining it with metaheuristic searching mechanism, including our proposed approaches [2, 9, and 12], Gao et al. [8], Bae et al. [45], Pranay Kate et al. [46], H. Nenavath et al. [47], M. Narayana et al. [48], etc.

**2-Traditional color histogram:** Compared to the previous presentation, this presentation is less used for tracking, as it requires high number of bins for robust presentation, therefore computationally expensive, Guang Liu et al [49], Henry et al. [50], and Ong Kok Meng el al. [12], combined this presentation with metaheuristic algorithms to create powerful SOT trackers.

### d. Shape

One of the basic object shape detection method is called: histogram of gradients or HOG detector, Fig.1.9. Originally, HOG detector have been adopted by Dalal and Triggs, in 2005 [51] for pedestrian detection, by training support vector machine (SVM) classifier. Thereafter it has been widely opted for tracking [52, 53]. HOG descriptor is extracted from the spatial analysis of the light intensity of the image. An important property of the HOG is its robustness to certain noises such as: background clutter, and illumination changes, but it cannot handle deformation or occlusion. The contours resulting from the HOG are exploited in many object tracking algorithms. In this line: Zhang et al. combined this feature with other metaheuristic methods such as: Extended Cuckoo search (ECS) algorithm with kernel correlation filter (ECS-KCF)[54], Moth-Flame Optimizer (MFO) [52], Grasshopper Optimizer (GOA) [53], hybrid Extended Ant-lion optimize with sine cosine algorithm (EALO- SCA) [56]; which have been proven to provide better accuracy than other trackers.

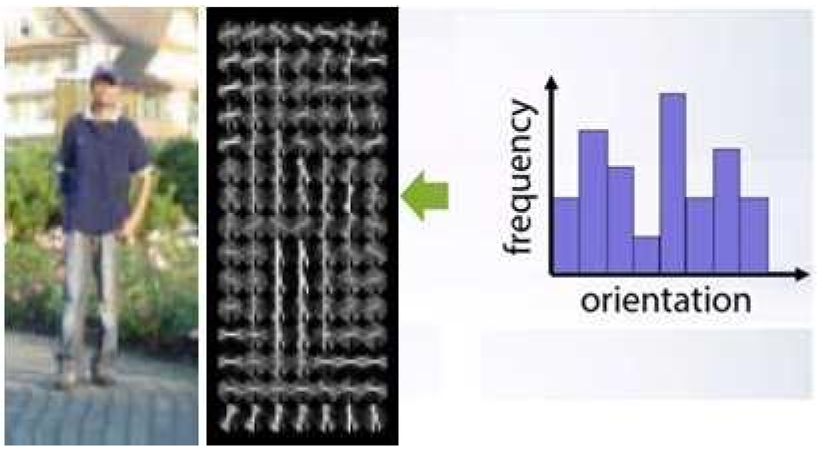


Fig.1.9 Computed HOG descriptor[51]

At the left is the test image, at the middle is it's at the right is an example of HOG vector of a small region.

### e. Texture

The texture of an object is a characteristic that can model an object. Texture is a measure of the variation in color arrangement or intensity in an image or selected region of an image [14]. Unlike color, texture requires processing steps to generate descriptors. There are different texture descriptors in the literature, including covariance region descriptors [56]; wavelet transform [57], Haar-like features [58], and Gabor filter [59]. Although its effectiveness, Gabor filter is difficult to use in real-time applications. Ojala et al. [60] developed Local Binary Patterns LBP (Local Binary Patterns) descriptor. LBP Texture operator is defined as a grayscale invariant texture measure. Many variants of LBP exist in the literature, such as Multi-Block LBP (MB-LBP) [61], local ternary patterns (LTP) [62], and local phase quantization (LPQ) [63], etc. Texture characteristics are less sensitive to changes in illumination compared to color. Texture feature have been utilized to model object appearance for tracking in many works [64, 65].

### f. Combination of features

In object tracking, some features can also be combined in a generative manner to build a robust visual tracker. Texture descriptor can be fused with color or other different types of features for example:

Comaniciu et al. [5] extended his previous work by using a joint spatial (LBP texture)-color histogram, with the same manner Saadia et al. [4] used mean-shift and combined color feature with LPQ texture feature to conceive a joint feature texture color for robust visual tracker against background clutter and blurred motions. Somayyeh S. et al. [66] used color histogram and the variance of the object to define the fitness function, and hybridized genetic algorithm GA with PF (GA-PF), to solve sample impoverishment of particle filter(PF), their tracking results demonstrated that their approach provide better performance compared to other tracking methods. Sardari F, et al. [67], combined color histogram and HOG feature to model the object appearance and proposed to use PF with modified galaxy-based search algorithm (PF-MGbSA), which is proven to be more accurate and more robust than other trackers, but it cannot handle occlusion.

#### 1.6.1.2. Discriminative methods

Despite the successes of generative models, they have encountered difficulties for the target object without considering the background information, especially when the appearance of the target object changes dramatically and / or the background plan is cluttered, or when the first detected region contain salient background information. On the contrary, discriminative models seek for a target appearance model by separating it from its surrounding background. Unlike generative methods, the target's information and its background are used simultaneously [9, 32], as shown in Fig.1.10.

They aim to maximize the separation between object and non-object regions in a discriminatory manner. In addition, they focus on discovering very informative characteristics for visual tracking. Therefore, they are more robust to complex scenarios by explicitly modeling the background as negative samples [32]. Methods based on the discriminative model have evolved rapidly in these recent years; in VOT2020 [30] challenge for example, 92% of the trackers are discriminative and only 8% are generative models.

Pranay Kate et al. proposed to enhance firefly algorithm (FA) [46], in their work they have implemented a discriminative approach by modeling the foreground by a spatial kernel color histogram and the background by its color histogram for better localization. Comparison results demonstrated that their algorithm outperform other eight baseline trackers, including: firefly algorithm(FA), particle filter (PF), Matrioska Flow (MatFlow), Local-Global Tracker (LGT), Incremental Learning Tracker (IVT), Multiple Instance Learning (MIL), Pixel Tracker (PT), and Best Displacement Flow (BDF).

In some models, a binary classifier is learned online to separate the target from its background; an example of these classifiers including [1]: SVM, ranking SVM, structured output SVM, semi-boosting, boosting, online multi-instance boosting. Avidan [68] trained SVM classifier with optical flow framework for tracking, to handle appearance changes. In [44], trained SVM classifier to learn online the most discriminative feature combination and thereafter builds a confidence map in each frame to separate a target object from its background. In [68], an ensemble of online learned weak classifiers was used to determine whether a pixel belongs to the target region or background. Hare et al. [69] designed a SOT based on a kernelized structured SVM, by exploiting the constraints of predicted outputs.

There are also approaches based on multiple representation schemes to effectively handle appearance changes. Stenger et al. [70] combined multiple observation models online in a cascaded or parallel manner. Recently, Kwon and Lee [71] developed SOT decomposition algorithm that uses multiple observations and motion models to score large appearance variations caused by fast motion and drastic lighting changes. This approach has been further extended to search for appropriate trackers by particle filter sampling [72].

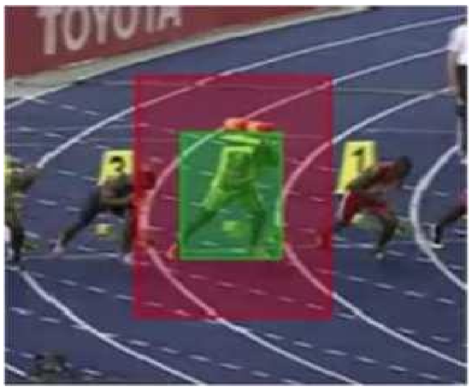


Fig.1.10 Discriminative approach [20]

The green bounding box denotes the object while the red one denotes its background.

Recently, discriminative tracking methods based on the correlation filter (KCF) proposed by Henriques et al. [73], have been shown to run super real time high speed and robust tracking performance, the reason behind its high speed is that the computation is performed in the Fourier domain.

The basic idea behind correlation filter is the estimation of an output image produced by correlation peaks for each target of interest in the scene while giving low background responses. The typical filter response is generally of a Gaussian shape centered at the target location, so the score decreases with the distance.

The filter is trained by shifting instances of the target patch. The filter is trained online and updated with every frame in order for the tracker to adapt to target changes.

The framework of a typical correlation filter-based monitoring method can be summarized as in Fig.1.11. The correlation filter is trained from the first frame by a given position of the target. Then, for each subsequent frame, different characteristics can be extracted from the raw input data, and a cosine window is usually applied to smooth the border effects, as in Fig.1.11. Subsequently, a response map is generated by a Fast Fourier Transform (FFT). The position with the maximum value in this map is the predicted the new location of the target object. Finally, the appearance at the estimated position is extracted for training and updating the correlation filter. Zhang et al [54], have integrated Extended Cuckoo search (ECS) algorithm with kernel correlation filter (KCF) tracker, for better tracking under fast motion, and showed better accuracy than other seven trackers.

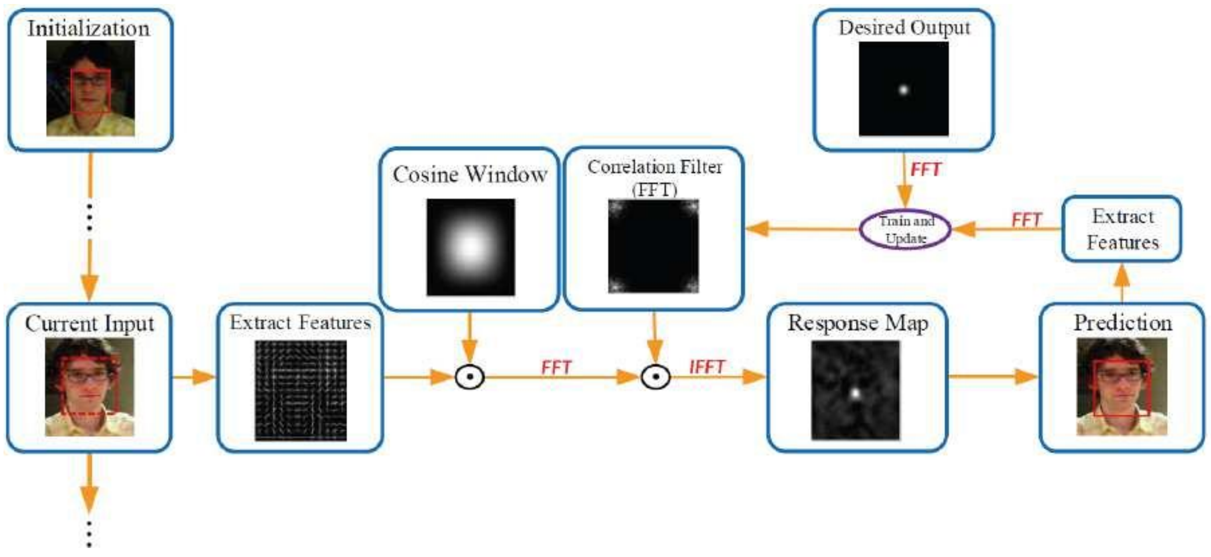


Fig.1.11 A general workflow for tracking method based on correlation filter [73]

### 1.6.2. The searching mechanism

The searching methods have been developed to estimate object states such as position and size. In general, there are three well known search strategies [2, 9, 12] namely: deterministic, probabilistic, and metaheuristic methods.

#### 1.6.2.1. Deterministic methods

Deterministic methods [1, 14] define a fitness function by associating to each object in frame $t-1$ to a single object in frame $t$ using a set of motion constraints. According to the fitness function a minimization or maximization of the correspondence is formulated as a deterministic optimization problem. The solution consists of the most optimal correspondence among all possible associations, which can be obtained by optimal assignment methods, for example, the mean-Shift algorithm [5], trust region [10], or snake search model [11], etc. The following constraints are usually imposed in this line [14]:

- ***Proximity:*** assumes that the future position of the object would not change considerably, from one frame to another.
- ***Maximum velocity:*** defines the lower and the upper bound on the object velocity, to limit all the possible locations around the object.
- ***Small velocity change:*** (smooth motion), low speed object between adjacent frames, assumes the direction and speed of the object does not change drastically.

However, we should mention here that these constraints are not specific to the deterministic methods, and some of them can also be used in the context of visual tracking using probabilistic or metaheuristic searching methods. It is worth mentioning here that the output results of deterministic tracking methods is constant when running the methods multiple times, while that output results of probabilistic and metaheuristic tracking methods are different in different running trials.

#### 1.6.2.2. Probabilistic methods

Object's movements can be subject to random disturbances; furthermore the acquired data by the video sensors always contains some noises. Probabilistic methods are defined [14] as, statistical matching methods which solve these tracking problems by adding uncertainty to the motion and the model of the object. Probabilistic methods use the state space approach to model object properties such as position, velocity, and acceleration. During tracking, the object's condition is estimated by a dynamic transition model, updated and corrected during tracking by taking measurements of the image. Some methods of estimating condition in this context include the Kalman filter (KF), particulate filter (PF), etc. But it should be noted that these methods can be used in general to estimate the state of any time varying system.

In the context of visual object tracking the information representing the object, for example, its position, is defined by a sequence of states $X^{t=\ 1,\ 2,\ \dots}$ then the change in state over time can be expressed by the following equations:

- $X^t=f^t(X^{t-1})+W^t$ (1.1)
- $Z^t = h^t(X^t, N^t)$ (1.2)

Where: $f^t$ is the state transition model applied to the previous state $X^{t-1}$; $W^t$: t = 1, 2, . . . is white noise; $Z^t$ is the measurement equation which specify the relationship between the measurement and the state; and $N^t$ is also a white noise but independent of $W^t$ .

The objective of tracking is to estimate $X^t$ given all the measurements or, evenly, construct the probability density function (pdf), $P(X^t | Z^{1,\dots,t})$.

A theoretical optimal solution is given by a recursive Bayesian filter to solve the problem in two steps namely: predictive step and correction step.

The prediction step uses the dynamic equation and the already computed pdf of the state at time t−1 to derive the prior pdf of the current state, that is $p(X^t | Z^1,\dots,^{t-1})$.

The correction step, employs the probability function: $P(Z^t | X^t)$ of the current measurement to evaluate the posterior pdf $p(X^t | Z^{1,\dots,t})$. In the case of SOT, the measurements required are the two steps as defined.

Kalman filter [74], and particle filter [75], and their variants [76], are a typical examples of probabilistic based techniques. Compared to deterministic methods, probabilistic methods such as particle filter for example [77] have been largely used because they are less sensitive to local minimum than mean-shift, and computationally efficient. Particle filters based methods recently have been developed using effective observation models [78] with demonstrated success.

#### 1.6.2.3. Metaheuristic methods

Metaheuristic methods are our related works, consider the visual tracking task, as an optimization problem, the same manner as deterministic methods, but resolved using metaheuristic optimization techniques [2, 9, 13]. The tracking process starts by modeling the target in interest by a chosen feature from the first frame and stored as a reference model for future comparison using a distance metric which is the fitness function. Then the metaheuristic algorithm is used as a localization method, to find the most similar region to the reference template. Subsequently, the fitness is measured between the template and random generated candidates, in which optimization is sought. Compared to the previous mentioned methods, metaheuristic tracking approaches are only of a few [2, 8].

### 1.6.3. Model update

Template update plays an importance role and an essential step for robust visual tracking in order to deal with pose variations and illumination conditions.

Matthews et al. [79] resolved the model update problem of the Lucas-Kanade tracking algorithm by using a combination of fixed reference template extracted from first frame and the most recent frame.

Online mixture models was also an effective update technique which has been proposed in the form of: incremental subspace updating [80], or online boosting [81].

Although considerable progress has clearly been made in model update line, developing a model update without drifts remains to be difficult.

## 1.7. Tracking based on deep learning

Trackers based on deep learning are a full class on its own; while the previous mentioned traditional tracking algorithms usually focus on developing robust appearance model using handcrafted features, searching mechanisms, online learning algorithms, or even both; deep learning based tracking algorithms are detection based methods [24], which focus on automatic extraction of features using multi-layer nonlinear transformations.

Most of the existing trackers based on deep learning use convolutional neural network (CNN). In the tracking task, CNN model is very appropriate for developing robust appearance model, due to its powerful ability on extracting features and image classification. Similar to the traditional tracking methods, CNN-based tracking methods can also be either generative or discriminative [1].

The reports on large-scale benchmark evaluations (OTB-2015) demonstrate that the traditional tracking algorithms performances are far from the requirement of realistic applications [1].

In the literature several deep-learning-based trackers (e.g., FCNT [82], STCT [83], LSART [84]) have emerged and demonstrated a significant tracking performance. As a result, the performance on the OTB-100 [85] dataset is constantly refreshed by the tracking methods based on deep learning. MDNet [86] for example was the winner of VOT2015 [25] competition. All the top-4 trackers in the VOT2020 competition are based on deep neural networks. In this competition, (86%) of the trackers used deep features, which shows that the field has moved away from using traditional features [30].

However, there some practical challenges and drawbacks when using deep learning on computer vision projects in general, including [87]:

- Many variables need to be tuned, that can impact performance, as we need to run many experiments to discover the best hyper-parameter values for our problem.
- As it requires a lot of computing capacity, running many experiments need to scale up multiple machines and GPUs. Providing machines and setting them up to run deep learning projects is time & resources consuming.

In the case of object tracking algorithms, we need to train on lots of videos, with very bulky training sets we need to copy these data to each training machine, which takes time and hurts productivity.

## 1.8. Conclusion

In this chapter, we have defined and clarified concepts very used in the field of object tracking such as object detection and classification; we realized that single object tracking are model free which means, it is not necessary for each tracking method to have an object detection mechanism, and we can conceive a single object tracking algorithm without any detection. We will see in the third chapter that detection based trackers can be just a type of multi-object tracking techniques.

Finally, in this chapter, we have also seen that VOT is a hard task, due to many factors that can affect the object appearance (occlusions, illumination conditions, deformation, etc.), and to overcome such difficulties, state of art of single object tracking techniques, and their main components, are presented.

# CHAPTER 2

# AVAILABLE DATASETS FOR SINGLE OBJECT TRACKING

## 2.1. Introduction

In order to effectively and objectively compare between tracking algorithms, it is necessary to assess them on the same unbiased dataset with a standardized protocol. Until 2013, datasets for SOT were still rare, and many of them were self made or case specific (e.g. for people surveillance, sport analytics, car tracking, aerial cameras, etc).

The performance's evaluation of any tracker demonstrates its strengths and weaknesses and helps to identify future research directions in this field to design more robust tracking algorithms.

In this chapter, we briefly introduce the commonly used datasets for evaluating the performance of different online visual trackers.

## 2.2. Single Object tracking benchmarks

### 2.2.1. OTB DATASET

The Object Tracking Benchmark (OTB) database is the most commonly used dataset in the literature. OTB was published in 2013 by Wu et al. [20], it contains 50 video sequences fully annotated Fig.2.1. The dataset is freely available in the website: www.visual-tracking.net, and provides an application-independent benchmark and protocol for evaluation.

The updated OTB published in 2015 contains 100 video sequences; it expands the sequences in OTB-2013 to include 100 target objects in the tracking benchmark OTB100 dataset [85], Fig.2.2. As humans are the most important target objects in practice, we can notice that the OTB-100 dataset contains more sequences of this category (36 body and 26 face/head videos) than in other datasets.

For a satisfactory analysis of the strengths and weaknesses of any tracking algorithms, OTB-100 annotates sequences according to 11 difficulties, namely: illumination variation (IV), scale variation(SV), occultation (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out of view (OV), background clutter (BC) and low resolution (LR). Each difficulty attribute represents a specific challenging factor in visual tracking. One sequence can be annotated with many attributes, and some attributes may occur more frequently than others do. The characteristics of tracking algorithms can be clearly analyzed from the sequences with the same attributes. For example, to evaluate how well the tracker handles occlusion, one may use 49 sequences (29 in OTB-100) annotated with the occlusion attribute.

Table 2.1 gives a description of the challenging factors in OTB2015 [85].

Table 2.1 Description of challenging factors in visual tracking adopted from [85].

| Challenge | Description |
|---|---|
| Scale variation (SV) | The ratio between the bounding boxes of the first frame and the current frame is out of range Rg, Rg =2. |
| Deformation (DEF) | The object has non-rigid deformation |
| Fast motion(FM) | The object's motion between adjacent frames is larger than tm pixels (tm=20). |
| Motion blur (MB) | The target region is blurred due to the motion of the target or the camera. |
| Background clutter (BC) | The background near the target has a similar color or texture as the target. |
| In-plane rotation (IPR) | The target rotates in the image plane. |
| Out-of-plane rotation (OPR) | The target rotates out of the image plane. |
| Illumination variation(IV) | The illumination in the target region is significantly changed. |
| Out-of-view(OV) | Some portion of the target leaves the view |
| Low resolution (LR) | The number of pixels inside the ground truth bounding box is very small, less than 400 pixels. |
| Occlusion(OCC) | The target is partially or fully occluded. |

These sequences are part of those commonly used in visual tracking. The first frame of each sequence in OTB2013 and OTB2015 is shown in Fig.2.1 and Fig.2.2, respectively. In order to present the progress of the tracking algorithms and to define a general benchmark, tracking results of 29 trackers are provided for comparison [13, 84]. Two adopted evaluation methodologies are used: precision plots Fig.2.3 and success plots Fig.2.4. The precision plots reflect the Euclidean error distance between the predicted center of the target and the ground truth (Fig.2.3). It is measured as the percentage of frames whose predicted object location (center of the predicted box) is at a distance varying between 0 and 50 pixels from the center of the ground truth bounding box. The legend of the precision plots contains threshold scores at 20 pixels.

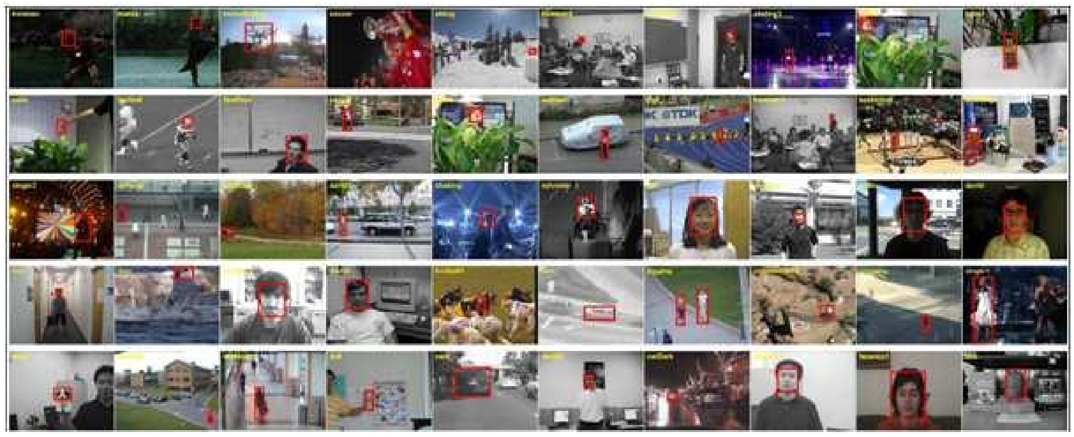
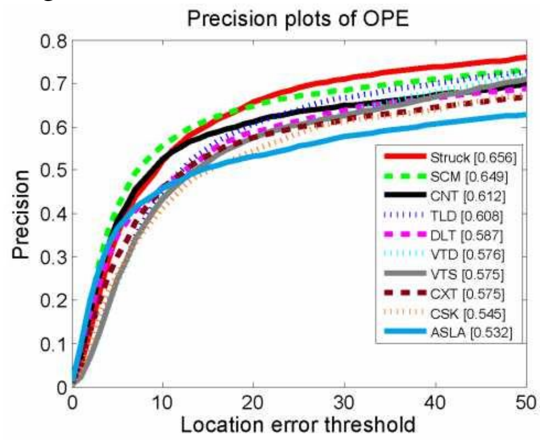


Fig. 2.1 The 50 video sequences of OTB2013 dataset [20].

The images correspond to the first frame of each sequence with the object of interest surrounded by a red bounding box.

Fig. 2.2 The newly added 50 video sequences of the OTB2015 database [85].

The images correspond to the first frame of each sequence with the object of interest surrounded by a bounding box.



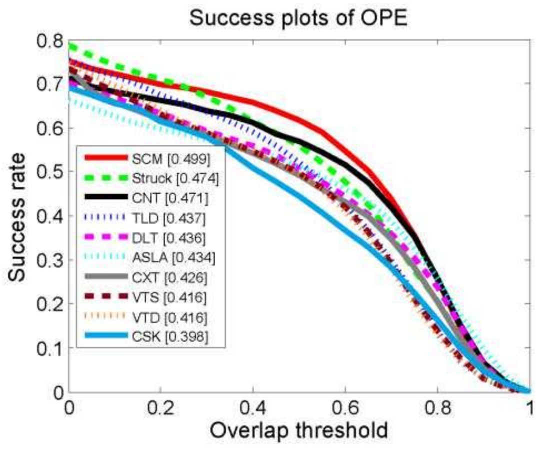
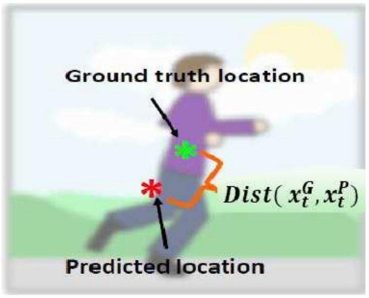


Fig. 2.3 The precision plots of OPE for the top 10 trackers [85]

Fig. 2.4 The success plots of OPE for the top 10 trackers [85]

The success plots (Fig.2.4) shows the percentage of frames whose score is defined as the Overlap ratio, which is the ratio of area of intersection of predicted in ground truth bounding boxes to the area of the union on these boxes as shown in the Fig.2.6 the score is between 0 and 1, and the legend of overlap success contains area under-curve threshold at 0.5.

When comparing between trackers, the success plots are preferred over precision plots, since precision plots uses only the bounding box locations, and ignores the size and overlap [13].



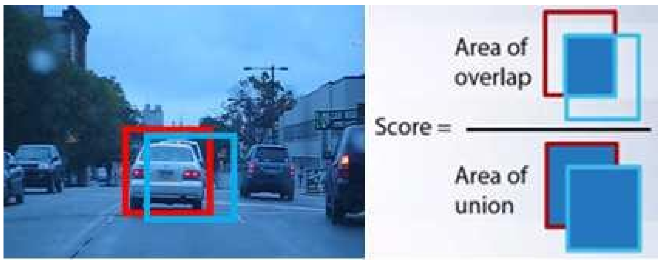


Fig.2.5 The center location error between the predicted center and the ground truth [22].

Fig.2.6 Illustration of the overlap ratio score [20].

### 2.2.2. The VOT DATASET

The VOT (Visual Object Tracking) benchmark was introduced in 2013, right after the first publication of the OTB2013 with the aim of providing a standardized platform for evaluating single camera, single object, model-free, and short-term causal tracking algorithms. A new edition is organized every year, and about 50 trackers are submitted for ranking [23-30].

The VOT challenge scoring scheme uses precision and robustness measures to compare between trackers, due to their high level of interpretability [23-30]. To assess the accuracy of tracking, we calculate the average overlap during successful tracking between ground-truth boxes and predicted bounding box. When the tracker overlap reaches zero, it is re-initialized and tracking is continued. So, when tracking misses off target, the tracking accuracy is penalized.

- The expected average overlap is used to merge between accuracy and robustness of a tracker into one metric.
- The robustness of an algorithm is defined as the number of times a tracker loses the target.

So, both, robustness and accuracy can be performed in one measure.

The bases of evaluation of VOT2013 [23] (16 videos in Fig.2.7) and VOT2014 [24] (25 videos) consist of sequences selected from OTB and ALOV ++ dataset [88], according to five challenges (occlusion, illumination changes, size change, object movement, camera movement). Whereas, VOT2015 [25] consists of 60 short video sequences selected from the OTB, ALOV ++, PTR [89] datasets, and annotated with 6 different challenges, namely: occlusion, illumination changes, motion change, size change, camera motion, and unassigned, and 30 other sequences annotated according to 11 global challenges. The VOT2016 database [26] is made up of the same VOT2015 videos, but the ground truth has been re-annotated more precisely.

The main difference between OTB-2015 and VOT2015 is that VOT2015 challenge provides are supervised running, which means it include a re-initialization protocol (i.e., the tracker is reset with ground truth in the middle of the evaluation if a tracking failure is observed).

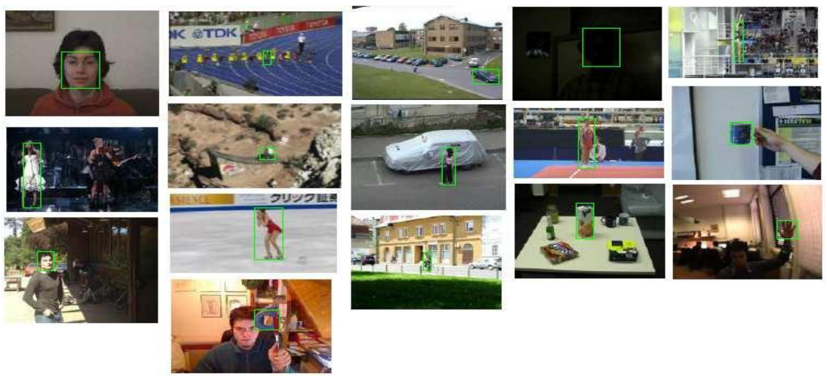


Fig. 2.7 Sequences of the VOT2013 database [23].

The images correspond to the first frame of each sequence with the object of interest surrounded by a green bounding box.

### 2.2.3. ALOV++ DATASET

The Amsterdam Library of Ordinary Videos (ALOV++) dataset [88] is published in 2013, it contains 315 video sequences, among them are 250 new, and 65 sequences have been reported from PETS Dataset [90]. In ALOV++ dataset, the preference was given too many short videos over a few longer ones to maximize diversity. The major source of the data is a set of real-life videos from YouTube with 64 different types of targets, including: ball, octopus, microscopic cells, human face, person, plastic bag, and a can. The collection is categorized by 13 levels of difficulty, from "hard" to "very hard" videos.

The data in ALOV++ are annotated by a rectangular bounding box with a flexible size along the main axes every five frames. In rare cases, in fast motion, the annotation becomes frequent. The ground truth bounding box in the first frame is specified for the trackers, but for other frames, in the middle of the sequences, it is calculated by linear interpolation.

### 2.2.4. TColor-128 DATASET

TColor-128 [91] is a large dataset published in 2015 with 128 color sequences; it is especially devoted to color visual tracking. Video sequences in TColor-128 mostly come from new color sequences newly collected from the internet collections and 50 other sequences from the previous datasets, such as OTB-2013 and VOT2013. The new added 78 sequences increase the diversity and difficulty of the previous 50 sequences as they involve various circumstances, such as: airport terminals highways, railway stations, concerts, and so on. Similar to the OTB dataset, each sequence in TColor-128 is annotated with 11 challenging attributes.

### 2.2.5. PTB-TIR DATASET

Thermal Infrared-Pedestrian Tracking Benchmark (PTB-TIR) [92], have numerous applications in the field of computer vision such as, the ability to track pedestrians in total darkness. PTB-TIR is the first dataset published in 2018 which aims to assess objectively the performance of tracking algorithm captured by thermal infrared devices. The benchmark contains 60 manual annotated video sequences; each sequence has nine attributes for evaluation and nine public available trackers for comparison.

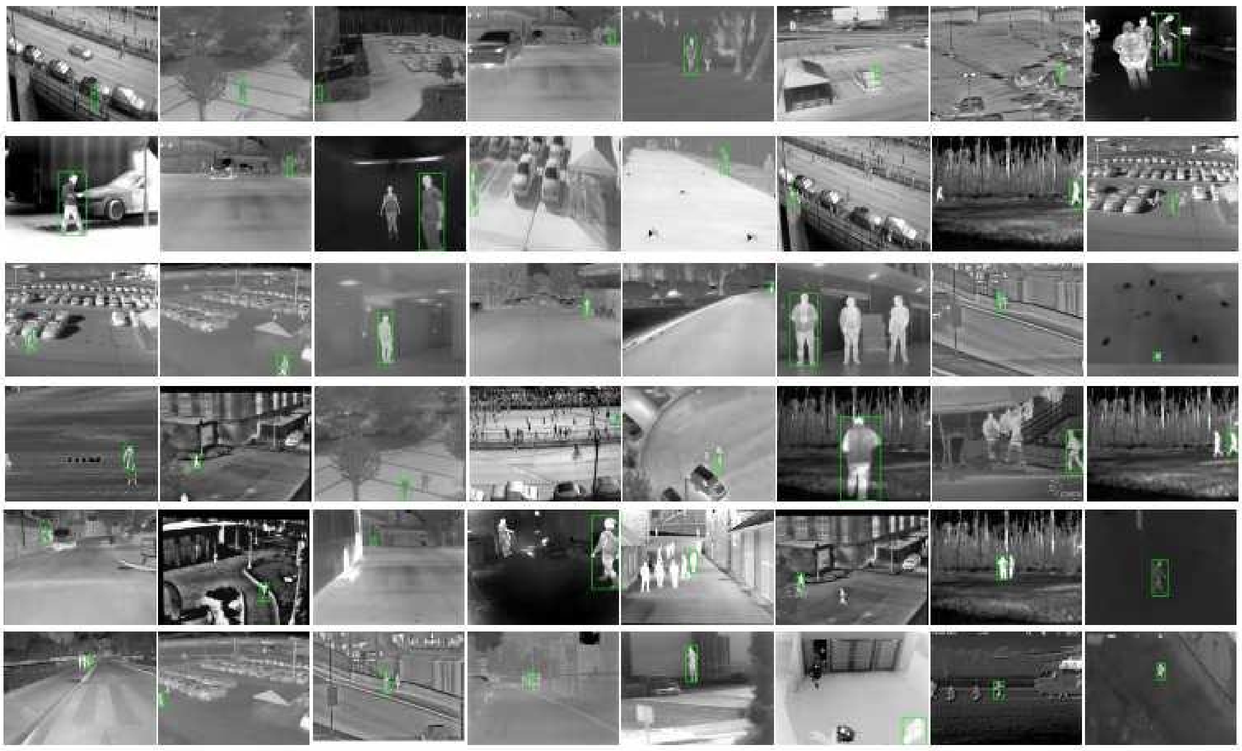


Fig.2.8 The first frame from some selected sequences of the PTB-TIR Dataset.
The target is surrounded by a green bounding box [92].

## 2.3. Conclusion

The purpose of this chapter is to show the most recent datasets and their corresponding evaluation metrics. Today, among the presented datasets the two benchmarks VOT & OTB15 are the reference datasets to evaluate any SOT algorithms. In the sixth chapter, our experiments and analysis will be based on the OTB2015 dataset, on which lots of researchers usually benchmark their trackers.

# CHAPTER 3

# MULTI-OBJECT TRACKING TECHNIQUES

## 3.1. Introduction

Multi-Object Tracking (MOT) or Multi-target tracking (MTT) is an important task in the field of computer vision which has earned an important attention due to its academic challenges and commercial potential.

There are two main differences between SOT and MOT. The first is, obviously in MOT we need to handle multiple objects simultaneously. The second is, MOT goal is long-term tracking instead of short-term tracking in SOT. The ultimate goal for MOT tracking is to work 24/24 and 7/7[3].

Generally, in MOT objects to track are of one class [3], for example: vehicles on the road, pedestrians on the street, sport players on the court, or groups of animals (birds, ants, fish**,** etc.).

Multiple objects can also be considered as different parts of a single object [93].

As it has many practical applications, such as: visual surveillance [94-95], action recognition [96], behavior analysis [96, 97], human computer interaction [98], and virtual reality [99]. These practical applications have sparked enormous interest in this topic**.**

## 3.2. MOT applications

MOT has attracted much attention due to its potential in many applications [3], such as:

- Video surveillance (monitoring, people activity recognition, intruder's detection & tracking).
- Sport analysis.
- Automated surveillance that is, monitoring a scene to detect suspicious activities or unlikely events.
- Traffic control.
- Human-computer interaction.
- Self driving car.
- Robotics: followed by obstacles during an avoidance phase.
- Video compression.
- Video conferencing (monitoring of interlocutors).
- Traffic management and analysis (monitoring a car or the edges of a road from a camera on board a vehicle).
- Robotics: followed by obstacles during an avoidance phase.
- Monitoring of visual cues in a task controlled by vision.

- Monitoring of an operator (body, face, hand, etc.) to define advanced modes of interaction between man and machine, etc.
- **Medical imaging.**
- Vehicle navigation that is, video-based path planning and obstacle avoidance capabilities.
- **Human-computer interaction:** such as, gesture recognition, eye gaze tracking for identification's system.

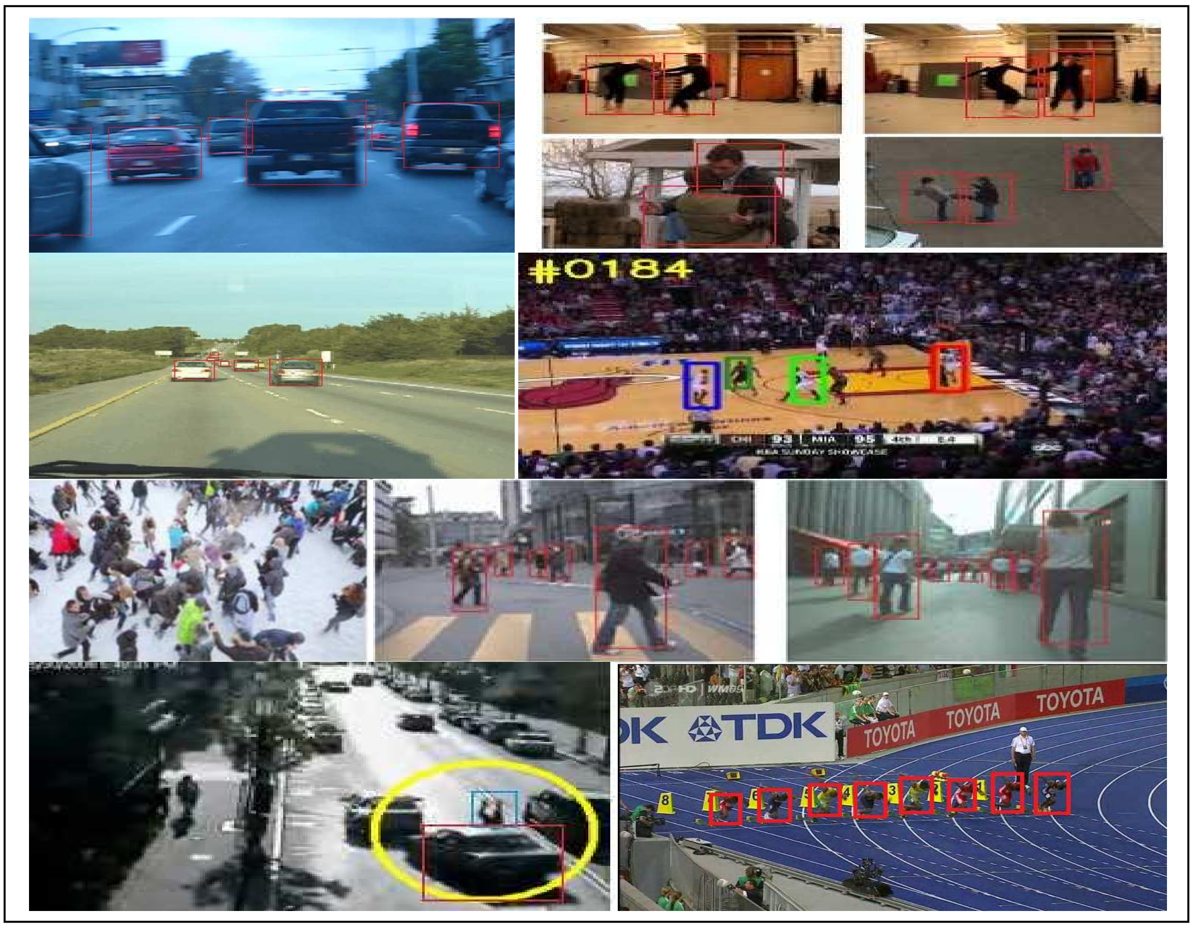


Fig.3.1 Some multi-object tracking applications [85, 91, 96-98].

## 3.3. MOT challenges

Compared to SOT task which aim to design a sophisticated search mechanism and/or appearance models to deal with challenging factors such as scale variations, rotations and illumination changes, MOT task is much harder [3], as its job is partitioned to positioning multiple objects, keeping their identities, and saving their individual tracks given an input video. Apart from the common challenges in both MOT and SOT, further key problems can make MOT much harder including [3]:

1) **Frequent occlusions:** objects are partially or completely occluded in some frames, as other objects appear in front of them and cover them up; occlusion handling is perhaps the most critical challenge in MOT as it is a primary cause for ID switches.

2) **Identity switches**: When two objects cross each other, we need to discern between them.

3) Appearance and disappearance of objects as they can move into or out of the frame unpredictably and we need to connect them to objects previously seen in the video.

4) Initialization and termination of tracks.

5) **Background clutter:** The background of the object can be similar to the object of interest, or also similar appearance between objects.

6) **Re-identification**: connecting an object in one frame to the same object in the subsequent frames.

7) **Motion blur**: objects may look different due to their own motion or camera motion

8) **Viewpoints**: objects may look very different from different viewpoints, and we have to consistently identify the same object from all perspectives.

9) **Scale change:** objects in a video can change scale dramatically, due to camera zoom for example.

10) **Illumination changes:** lighting changes in a video can have a big effect on how objects look and can make it harder to consistently detect them.

11) Interactions among multiple objects.

12) Real time processing.

## 3.4. Note on real time MOT

In the literature there is no specific definition of real time tracking [100,101] either for SOT or MOT. Also there is no reference speed required for a tracker to be said a real-time tracker.

Instead, it is said that a real-time model if it can give an output faster than or as fast as, the frame rate of the video input. Then, a real-time tracker is so, if it can scan 60 seconds of video sequence in less than one minute or else, it is not [101].

However, the video is varying in time, as it is recorded at a determined frame-rate, and the frames are inputted to the tracker with a constant time. Because online tracking models are needed to give an output before the next input is provided, the waiting of the recording system is directly connected to the rate in which frames are scanned.

In fact, for some real-world tracking applications, it is not necessary to analyze at a certain frame rate (frames per second=FPS); we can just regulate the frame-rate of the input to correspond to the tracker's speed and say that anything above a certain FPS is acceptable[101]. But, for some applications such as robotics and surveillance, shorter delay and higher frame-rate might be necessary, and because of the challenges mentioned previously, MOT is considered to be very difficult. In this case, one way to attain real-time execution is skipping all incoming frames until the tracker has finished analyzing a given frame, so that the input is all the time the most recent recorded one. In this case, the delay is stable, and we can say that the tracker is real-time. To the best of our knowledge, there have not been so much works on this line, to investigate how skipping frames would affect the tracking performance [101].

## 3.5. MOT Categories

A wide range of MOT approaches have been proposed in the past few decades, to deal with all the mentioned key challenges. These trackers are based on different aspects of a MOT system. Based on initialization there are mainly two variants of MOT problems [3]: detection based tracking (DBT), and detection free tracking (DFT).

### 3.5.1. Detection based tracking (DBT)

Commonly referred as "tracking-by-detection", in this category, as shown in Fig.3.2 (top), the object detector is applied to each frame in the video. Objects are detected at each frame and then linked into trajectories. Given a sequence, a specific object detection approach is applied to obtain objects' states (positions, sizes), and then the tracking process is conducted to link detected instances into trajectories.

#### 3.5.1.1. Detection based tracking steps

There are three steps in DBT procedure namely: detection, prediction, and data association [3].

##### Step one: Detection

The selection of the suitable approach to detect objects of interest relies mainly on the camera type whether it is stationary or not, and what we want to track.

**A- Detecting objects using a stationary camera**

To detect moving objects using a stationary camera, we can implement background subtraction technique. Background subtraction approach is efficient if the camera is stationary.

**B- Detecting objects using a Moving Camera**

To detect moving objects using a moving camera, we can use sliding-window detection technique. Typically, this approach is slower than the previous approach, and it generally use a specific class of object.

There are two main issues worth noting in this category: First, most of them focus on single class of objects in interest, such as vehicles, pedestrians or faces, and the object detector is trained beforehand. Second, their performances are highly depending on the performance of the combined object detector.

##### Step two: Prediction

Tracking objects over time means that we must predict its state in the next frame. The most obvious prediction method is to assume that the objects will move near the previous location. This method is efficient for high frame rates. However, it may fail for objects that move at varying speeds.

The prediction is based on both appearance models and searching methods, using the same principles previously seen in the chapter of SOT. The searching technique of each object for example is estimated by a specific approach, the same searching methods types implemented for SOT can be applied here.

#### Step three: Data association

It is the process of associating the detected objects to their corresponding predicted state at each frame. A track representation includes the whole state history of the object.

Data association must also take into consideration the fact that new objects can appear or leave the field of view.

In a given frame, some detections may be assigned to tracks for update, while other detections and tracks may considered unassigned and set as new tracks. Each track is saved in consecutive frames, when it is unassigned for a specific threshold number, it is assumed that the object has left the field of view.

Although some methods diverge from this approach slightly, doing these three steps in different ways [101].

Problems emerges in the data association task, when new objects enter the scene, target objects are not detected or leave the scene, and the detector give false positives i.e. wrong detection or detecting an object that doesn't exist.

**Assignment problem** consists of finding the optimal matching between the predicted states and those detected; Hungarian algorithm [101] is generally used to solve such problem.

### 3.5.2. Detection free tracking (DFT)

Similar to SOT, in this category we have no idea about objects of interest, except their bounding boxes in the first frame. As shown in Fig.3.1 (bottom), [102-105] this category requires manual initialization of a fixed number of objects in the first frame, then localize these objects in subsequent frames. This category is less used but more popular because new objects are discovered and disappearing objects are terminated automatically. Its main disadvantage is that it cannot deal with the case that objects appear in other frames except the first frame. However, they are free of pre-trained object detectors. Table 3.2: depicts the main differences between DFT and DBT.

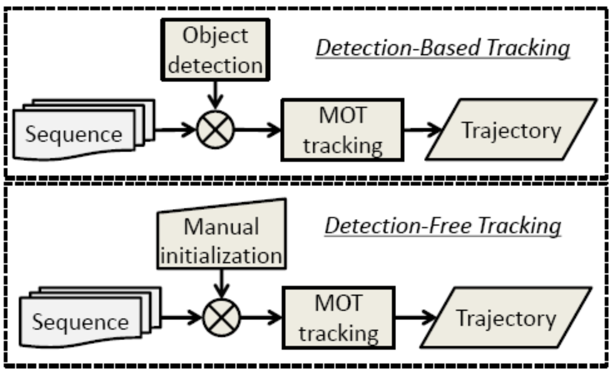


Fig.3.2 A procedure flow of two MOT types,

Detection-Based Tracking (DBT) and Detection-Free Tracking (DFT) [3].

Table 3.1 Comparison between DFT and DBT, adapted from [3].

| Item | DFT | DBT |
|---|---|---|
| Initialization | Manual, Perfect | Automatic, Imperfect |
| Number of objects | Fixed | Variable |
| Applications | Any type of objects (Multi-class) | One type of objects (in most cases) |
| Advantages | Free of object detector | Can handle variable number of objects |
| Drawbacks | Manual initialization. | Tracking performance depends on objects detector |

## 3.6. Multi-Object Tracking Evaluation

To assess the performance quantitatively of an MOT approach, metrics and datasets are required. This is necessary for two major reasons. From one standpoint, it is important to evaluate the effect of different parameters and components on the overall tracking performance to conceive the best system. On the other point, it is desirable to have a direct comparison to other methods. MOT's performance evaluation is not straightforward, as SOT and we will see this in the 8th section.

## 3.7. Multi-object tracking benchmarks

Works on standardizing the MOT evaluation was rather limited and most of the published datasets, even today, are application specific [101]. Multiple benchmarks exist for evaluating tracking models; the most recent and commonly used in the literature are presented below:

### 3.7.1. MOT Challenge

Since 2015, MOT challenge [106] was the most used benchmark for MOT, it focuses on pedestrian tracking. It is a yearly competition benchmark, and its website has a scoreboard always open to new tracker submissions.

The dataset video sequences are labeled with bounding boxes for each pedestrian.

The Dataset is a collection of annotated video sequences from multiple sources; the sequences differ in lightning conditions, resolution, and frame rate. The online public scoreboard is provided to promote competitions and comparison of different tracking models. In this benchmark there are two sets of challenges. In one assignment, you are given raw video sequences, and you need to detect and track the objects; in the other assignment you are given detection annotated sequences, and the assignment is to conceive an as accurate tracker as possible.

Although the first assignment's results are generally better than the second, because the detections are not state of arts, the second assignment is undoubtedly more interesting, because it measures the tracker performance, without regard to detection accuracy [101].

A free public tool is provided to compute 13 metrics for any data. An example of an annotated frame is shown in Fig.3.3.

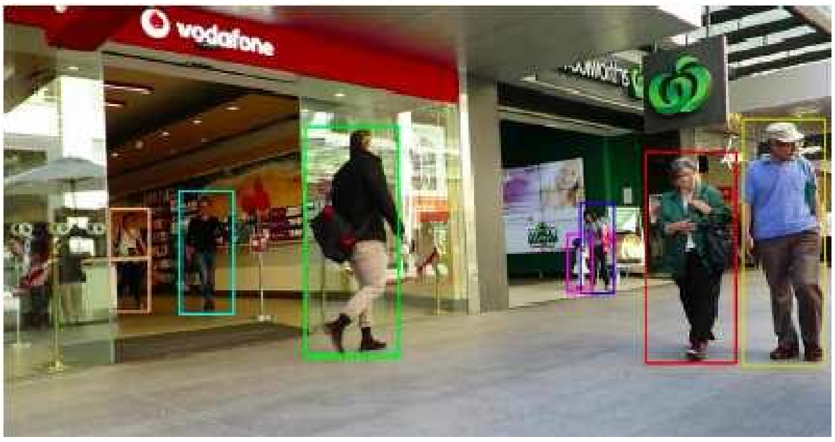
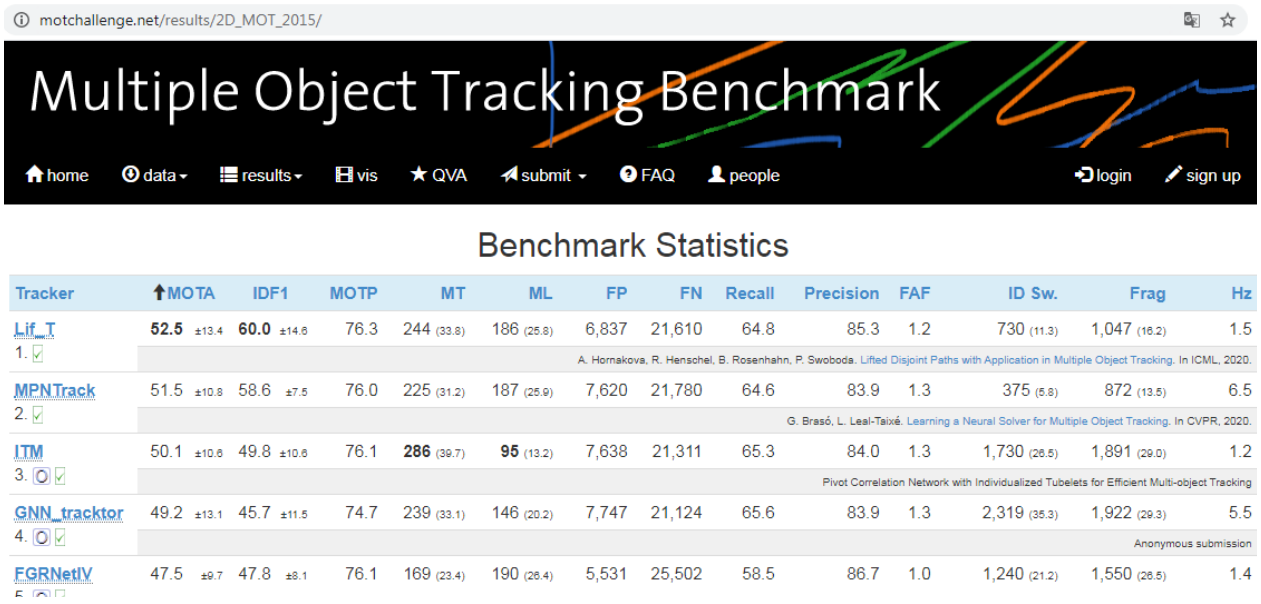


Fig. 3.3 An annotated frame from MOTChallenge [106].
The Ground truth bounding boxes are displayed, each color represent an ID.

In the figure below we show the 13 metrics scoreboard of the top 5 trackers on the MOTchallenge website (www.motchallenge.net).


motchallenge.net/results/2D_MOT_2015/
Multiple Object Tracking Benchmark
home
data
results
vis
QVA
submit
FAQ
people
login
sign up

Benchmark Statistics

| Tracker | MOTA | | IDF1 | | MOTP | MT | ML | FP | FN | Recall | Precision | FAF | ID Sw. | Frag | Hz |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lif_T 1. | **52.5** | ±13.4 | **60.0** | ±14.6 | 76.3 | 244 (33.8) | 186 (25.8) | 6,837 | 21,610 | 64.8 | 85.3 | 1.2 | 730 (11.3) | 1,047 (16.2) | 1.5 |
| | A. Hornakova, R. Henschel, B. Rosenhahn, P. Swoboda. Lifted Disjoint Paths with Application in Multiple Object Tracking. In ICML, 2020. | | | | | | | | | | | | | | |
| MPNTrack 2. | 51.5 | ±10.8 | 58.6 | ±7.5 | 76.0 | 225 (31.2) | 187 (25.9) | 7,620 | 21,780 | 64.6 | 83.9 | 1.3 | 375 (5.8) | 872 (13.5) | 6.5 |
| | G. Brasó, L. Leal-Taixé. Learning a Neural Solver for Multiple Object Tracking. In CVPR, 2020. | | | | | | | | | | | | | | |
| ITM 3. | 50.1 | ±10.6 | 49.8 | ±10.6 | 76.1 | **286** (39.7) | **95** (13.2) | 7,638 | 21,311 | 65.3 | 84.0 | 1.3 | 1,730 (26.5) | 1,891 (29.0) | 1.2 |
| | Pivot Correlation Network with Individualized Tubelets for Efficient Multi-object Tracking | | | | | | | | | | | | | | |
| GNN_tracktor 4. | 49.2 | ±13.1 | 45.7 | ±11.5 | 74.7 | 239 (33.1) | 146 (20.2) | 7,747 | 21,124 | 65.6 | 83.9 | 1.3 | 2,319 (35.3) | 1,922 (29.3) | 5.5 |
| | Anonymous submission | | | | | | | | | | | | | | |
| FGRNetIV 5. | 47.5 | ±9.7 | 47.8 | ±8.1 | 76.1 | 169 (23.4) | 190 (26.4) | 5,531 | 25,502 | 58.5 | 86.7 | 1.0 | 1,240 (21.2) | 1,550 (26.5) | 1.4 |

Fig. 3.4 Scoreboard of MOT challenge.

The trackers are ranked according to their MOTA score.

### 3.7.2. ImageNet

ImageNet [107] was first used for competition in image recognition, for now it provide challenges on both object detection in images and object detection and tracking in videos simultaneously.

The annotated video sequences contain 30 different objects categories; and only one metric is used in this dataset which is the mean average precision.

A frame example with instances of the class bicycle is shown in Fig.3.5.

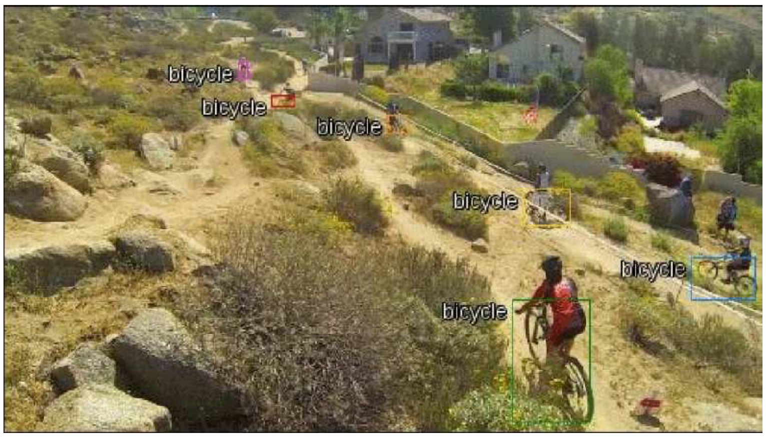


Fig. 3.5 An example of annotated frame from ImageNet [107].
Ground truth bounding boxes are displayed, with their class label, each color represent an ID.

### 3.7.3. Okutama-Action

Okutama-Action [108] is the first dataset for action detection published just recently, consists of 43 minute-long fully-annotated aerial-view video sequences with 12 action classes captured from Unmanned aerial vehicles (UAVs). The benchmark implements some undiscussed challenges not present in the current datasets, for example significant changes in scale, abrupt camera motions, the aspect ratio dynamic transition of actions, different varying altitude, and view-angles. The altitude, ranges from 10 to 45 meters. The video sequences (example in Fig.3.6) are provided with two distinct annotations: one for pedestrian tracking and one for action recognition. There are 12 action classes available for each action recognition task. Because the dataset is recently published, no available results are available for comparison.

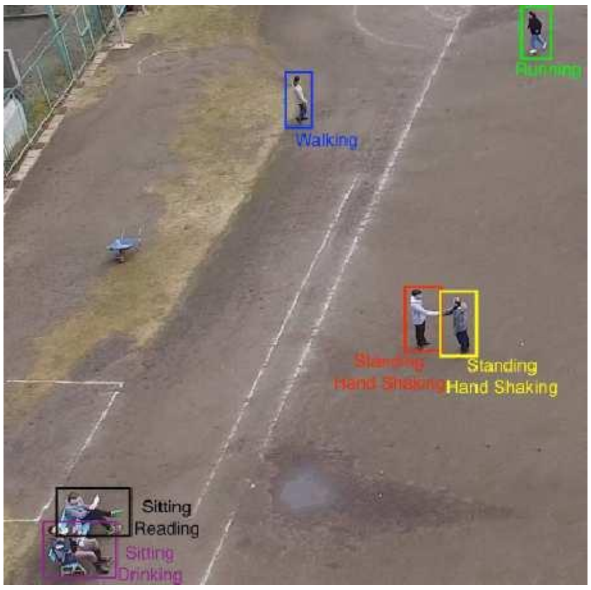


Fig. 3.6 A sample frame from Okutama-Action [108]

Ground truth bounding boxes are displayed, along with the actions, each color represent an ID.

## 3.8. MOT Evaluation metrics

Evaluating an MOT approaches are an important results, because they give a fair meaning of quantitative comparison. An overview of the MOT assessment metrics is presented in this section.

Performance metrics vary according to the dataset used and the MOT type implemented whether it is DFT or DBT. For example DFT use the same performance metrics used in SOT, such as: the average overlap, average Euclidean distance for each object, and the number of successfully tracked frame, etc [102-105]. On the other side DBT approaches use the tracking-by-detection framework, where detection performance is often measured as well as tracking performance. Therefore Object detection metrics are also used in an MOT approach.

To better understand the aim of these metrics [3], we need to explain what we expect from an ideal multi-target tracker:

First, it must get the right objects' number present in the scene and estimate the precise state (position, size); it is worth mentioning here that orientation, and contour are not considered in MOT.

Second, it should also assign to each object a unique track ID, all over the sequence (despite temporary occlusion, etc).

This leads to the following performance evaluation metrics:

• The ability to judge the tracker’s precision to determine the accurate objects' locations.

• The ability to correctly track down object tracks, by producing one trajectory per object.

Additionally, we expect that useful evaluation metrics to be:

• Clear, and understandable, with few simple parameters, adjustable thresholds, in order to make straightforward comparison and keep results comparable.

• Expressive and few in number, so they can be used in large evaluations where many different systems can be compared.

Based on what we have mentioned before, MOT metrics for DBT can be categorized into two sets of metrics: detection and tracking respectively [3]; in the following we will illustrate the most used metrics in this line:

### 3.8.1. Metrics for detection

The detection performance evaluation metrics can be grouped into two subclasses [3]. One class measures accuracy, and the other one measures precision; "**detection accuracy**" depicts how many errors the detector made in terms of misdetection, false positives, mismatches, and so forth; and the "**detection precision**" which expresses how well accurate positions of objects are predicted.

#### 3.8.1.1. Detection accuracy

To measure the detection accuracy, the commonly used precision metric is given by comparing the predicted bounding box with ground truth bounding box.

The intersection over union (IoU), also named as Overlap ratio or Jaccard index, is the ratio of the area of the intersection of the predicted bounding box with the ground truth bounding boxes divided by the area of the union on these boxes[101].

Generally **IoU** is defined by a threshold to define detection correctness. The larger the threshold, the more precision the detector localize objects. Usually, the threshold is set to 0.5. The detection output is a set of detection proposals as shown in Fig.3.7. A **true positive (TP)** detection is observed when **IoU** prediction score exceeds the predefined threshold.

A **false positive** detection indicates that **IoU** score is lower than the threshold, in this case the ground truth is marked as **misdetection** or **false negative**.

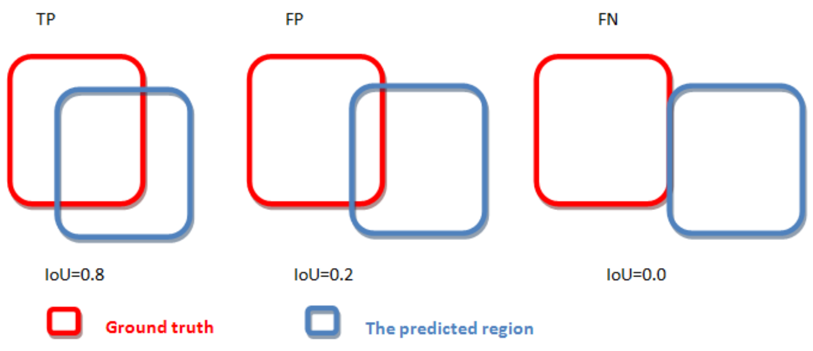


Fig. 3.7 Object detection IoU

**FN** and **FP** are detection metrics used in MOTChallenge Dataset [106], and defined as follow

- **FP:** is the total number of false positive detections.
- **FN:** is the total number of false negative detections (missed targets).

Choi et al. [109] used the average false positive per frame index (**FPPI**) metric to evaluate detection accuracy in MOT. Another metric which called as Multiple Object Detection Accuracy (**MODA**), considers the relative number of false positives and miss detections [110].

- **Precision:** Precision metric is the ratio between the accurately detections and the entire detections, as in the below equation[3]:

- $$\frac{TP}{TP+FP} \tag{3.1}$$

#### 3.8.1.2. Detection precision

- **N-MODP**: Normalized multiple object detection precision metric measures the quality of alignment between predicted detections and the ground truths [111] for an entire video, its formula is as follow:

- $$\text{N-MODP}(t)=\frac{1}{N}\sum_{t,i}\frac{IoU_t^i}{N_t} \tag{3.2}$$

Where $IoU_t^i$ is the intersection over union between the predicted and the ground truth position of an object i in the frame t, N is the total number of frames, $N_t$ is the number of objects found in a frame t.

### 3.8.2. Metrics for tracking

Metrics for tracking are classified into four subsets by different attributes as follows [3]:

#### 3.8.2.1. Tracking accuracy

It depicts how many mistakes made by the tracker concerning:

- Misses.
- False positives.
- Mismatches.
- Failures to recover tracks.

There are four metrics used in the MOTChallenge dataset for tracking accuracy defined as below:

1. **ID_SW:** Identity switches are how often is assigned a new ID for an object in its track.
2. **MOTA:** Multiple object tracking accuracy, in 2008 Bernardin and Stiefelhagen [110], proposed two new metrics: **MOTA** and **MOTP** for multiple object tracking precision, and they are referred as the CLEAR MOT metrics; **MOTA** perfect score is equal to 100%; and it is computed by a combination of the three previously mentioned errors as below:

- $$\mathbf{MOTA}=1-\frac{\sum_t(FP_t+FN_t+IDS\_W_t)}{\sum_t G_t} \tag{3.3}$$

Where: **t** is the frame's number, $\mathbf{G_t}$: is the objects' number in the t frame.

3. **Recall:** Is the ratio of correct detections to total number of ground truth (*GT*) boxes, formulated as follows:

- $TP/\sum Gt$ (3.4)

4. **FAF:** refer to false alarm per frame, it evaluate the average number of **FP** in each frame.

#### 3.8.2.2. Tracking precision

Tracking precision expresses how well accurate objects' states are estimated.

1. **MOTP**: refers to MOT precision, it measures the proper positioning of the predicted bounding boxes with the ground truth. Primarily, the formula proposed by [111] was:

- $MOTP=\frac{\sum_{i,t} d_t^i}{\sum_t C_t}$ (3.5)

Where $d_t^i$ is the Euclidean distance between the predicted and the ground truth position of an object i in the frame t; $C_t$ is the number of matches found in a frame t. In the original formula(3.5) the perfect score is zero, however it was modified in MOTChallenge, changing the numerator in the equation (3.5), by the IoU between the predicted positions and the ground truth, so that a perfect score has MOTP = 100%.

1. **IDF1:** proposed by ristani et al. [112] permits positioning all trackers on one scale that balances identification precision and recall through their harmonic mean; it is defined by the ratio:

- $\frac{2*TP}{(2*TP+FP+FN)}$ (3.6)

#### 3.8.2.3. Tracking completeness

There are other trajectory-based metrics [3,101] that can estimate the performance of trajectories instead of frame-by-frame errors. A target is often tracked accurately, for a certain period of time and not for its complete existence in the video. To quantify this metric, a trajectory can be classified into:

1. **MT:** mostly tracked, a target is considered mostly tracked when more than 80% of its ground truth trajectory is detected.
2. **ML:** mostly lost, a target is mostly lost when it is detected during less than 20% of its presence.

Fig.3.8. illustrate these two trajectory metrics.

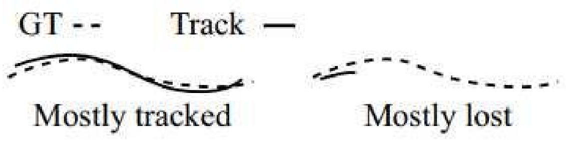


Fig. 3.8 Trajectory based metrics [3]

3. **Frag:** Fragmentation metric computes how often an object is lost in a frame, then redetected in a future frame, consequently the track is fragmented [106].

#### 3.8.2.4. Tracking speed

The tracker's speed metric is measured in Hz (number of frames per second), this metric is provided by the authors and not officially evaluated by the MOTChallenge; the detector's speed is not mentioned [106].

In the following table we present the 13 metrics used in the MOTChallenge Benchmark. The second column indicates if the performance is better when the value is higher (resp. lower).

Table 3.2 The 13 metrics used in MOT challenge[106].

| N° | Measure | Better score | Perfect score |
|---|---|---|---|
| 01 | MOTA | Higher | 100% |
| 02 | MOTP | Higher | 100% |
| 03 | FP | Lower | 0 |
| 04 | FN | Lower | 0 |
| 05 | ID Sw. | Lower | 0 |
| 06 | FAF | Lower | 0 |
| 07 | MT | Higher | 100% |
| 08 | ML | Lower | 0% |
| 09 | Frag | Lower | 0 |
| 10 | Hz | Higher | Inf. |
| 11 | Recall | Higher | 100% |
| 12 | Precision | Higher | 100% |
| 13 | IDF1 | Higher | 100% |

## 3.9. The Correspondence between MOT output system and the ground-truth reference

Finding the appropriate mapping between the MOT system output and ground-truth reference is an important task [3,101], for both detection and tracking. Mapping strategy is needed for precise measures and concise computation of the previously mentioned metrics such as FN, FP, and ID-SW, etc. The optimal mapping employs an error optimization algorithm to find the most acceptable matching according to a metric. Generally, the spatial area under curve between ground-truth and MOT system output is used as a metric. The mapping result is a matrix (NxM); where N is the number of ground-truth objects, and M is number of MOT system output objects). The metric score value between the ground truth $G_i$, and the predicted system output $D_j$ is denoted by $X_{ij}$, Fig.3.9.

$$\begin{array}{c|cccc}
 & \mathbf{D_1} & \mathbf{D_2} & \mathbf{...} & \mathbf{D_M} \\
\hline
\mathbf{G_1} & \mathbf{X_{11}} & & & \mathbf{X_{1M}} \\
\mathbf{G_2} & & & & \\
* & & & & \\
* & & & & \\
* & & & & \\
\mathbf{G_N} & \mathbf{X_{N1}} & & & \mathbf{X_{NM}}
\end{array}$$

Fig. 3.9 The mapping matrix

The maximum score is attained by optimally assigning system output and ground truth pairs. This problem can be identified as an optimization problem and generally solved using Hungarian algorithm [113].

## 3.10. Conclusion

In this chapter we provided a brief overview of MOT techniques, its applications and the challenges that may affect MOT approaches; we also held a brief discussion on real-time tracking.

We also presented the most commonly used datasets for MOT, as well as the most used evaluation metrics that can be found in the literature; we realized that most of them are reserved for detection based MOT approaches.

In this thesis, we mainly focus on the research on DFT tracking based on metaheuristic searching method. To the best of our knowledge, not many approaches have been proposed in this line, and it still remains challenging due to factors like there is no dataset for this type.

# CHAPTER 4

# STATE OF ART OF METAHEURISTIC ALGORITHMS

## 4.1. Introduction

Metaheuristic algorithms are stochastic optimization methods designed to solve difficult problems when deterministic methods fail, or require an important computation time. Generally, metaheuristic techniques are based on random mechanisms to explore the search space, to find or give an approximate global optimum. Metaheuristics have been widely used in various domains such as: business, engineering, economics, image processing, mechanics, encryption, and telecommunications,... etc. In this chapter we will present, optimization problems classifications, and metaheuristic based optimization techniques and their categorizations.

## 4.2. Optimization problem components

An optimization problem, in its general sense, consists of determining a solution x which belongs to a search space X. This solution minimizes or maximizes an objective function f. In addition, an optimization problem can present constraints of equality and / or inequality on candidate solutions; generally these are additional conditions that restrict the search space [114].

The formulation of the optimization problem remains very ambiguous because the diversity of vocabularies and the possible confusion that could be generated. So we have agreed to adopt the following vocabulary:

**The search space**, is also called the state space or the search domain, is the set of areas of definition of the different variables of the problem.

**The variables of the problem**, also called variables of designated problem, which can be of different nature (real, integer, or Boolean, etc.) and express qualitative or quantitative data, in this thesis we are interested in the integer case.

**The objective function**, the fitness function or even called the cost function, defines the goal to be achieved, we seek to minimize or maximize it.

A multimodal function has several minima (local and global). While a uni-modal function has only one minimum, which is the global minimum.

**The set of constraints** is generally a set of equalities with/or inequalities that the variables must satisfy. These constraints limit search space.

Optimization methods search to find the minimum points (or maximum) local (or global) of an objective function, that satisfy all the constraints. Fig.4.1. gives an example of minimum and maximum global and local.

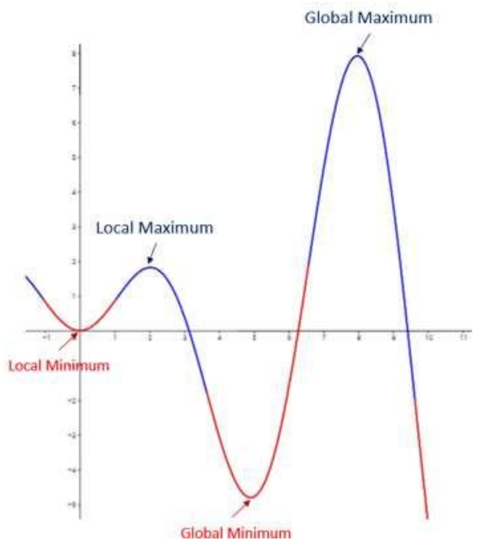


Fig. 4.1 The local and global minima and maxima of a function

## 4.3. Example of mathematical formulation of single objective optimization problem

Mathematically, in the case of minimization, a single-objective optimization problem comes in the following form [114]:

Finding x ∈ X, where x=$\begin{pmatrix} x1 \\ x2 \\ \vdots \\ xn \end{pmatrix}$ is variable of the problem, which is the solution that minimizes f(x), and n is the dimension of the problem, and under the constraints:

- $$\begin{cases} h_i(x) = 0, i = 1, \dots, m \\ g_j(x) \leq 0, j = 1, \dots, m \\ x_L \leq x \leq x_U \end{cases} \tag{4.1}$$

Where $h_i(x)$ and $g_j(x)$, are respectively the constraints of equality and of inequality and $x_L, x_U$ are respectively the lower and upper bounds of the variables research domain.

It is possible to go from a minimization problem to a problem of maximization thanks to the following property:

- $$\underset{x \in X}{\text{Min}}\, f(x) = \underset{x \in X}{\text{Max}}\, -f(x) \tag{4.2}$$

## 4.4. Optimization problems classification

According to the nature of f(x), the constraints $g_j(x)$ and the variable x, the corresponding optimization problem goes by various names. We distinguish more particularly the following cases [114]:

- ✓ **According to the number of objective function:** Single-objective optimization problem is defined by a set of variables, a unique objective function and a set of constraints, while multi-objective optimization problem is defined by a set of variables, a set of objective functions and a set of constraints.
- ✓ **According to the existence of constraints:** The optimization problem can be with or without constraints.
- ✓ **According to the nature of the variables:** The optimization problems can be classified into two categories:
  - In the first category, the problem is to find the optimal value of a parameter.
  - In the second category, the problem is to find the optimal value of a parameter which depends on another parameter. This type of problem where each variable is a function of one or more parameters is known as a dynamic or trajectory optimization problem.
- ✓ **According to the nature of the equations involved:** This classification groups optimization problems in four categories: linear, non-linear, geometric and quadratic.
- ✓ **Linear:** If the objective function and all the constraints are linear functions, the problem is called a linear programming (LP) problem.
  - $$\begin{cases} \min f(x) = \sum_{i=1}^{d} a_i x_i \\ g_i(x) = \sum_{i=1}^{d} c_{ji} x_i \ , \ j = 1, \ldots, k \end{cases} \tag{4.3}$$
- ✓ **Non-linear:** If a function among the objective functions or the constraints is non-linear, the problem is called a non-linear programming problem (NLP). Practically this category cover all the categories, which make the others, appear as special cases.
- ✓ **Geometric:** In the geometric programming problem (GMP), the objective function f(x) and the constraints $\mathrm{g_j}(\mathrm{x})$ are expressed as posynomial in X. A posynomial of f(x) of N terms can be expressed as follows:
  - $$f(x) = \sum_{j=1}^{N} c_j \prod_{i=1}^{n} x_i^{aji} \tag{4.4}$$

Where: $c_j$ and $x_i$ are positive real numbers, and the exponents $aji$ are real numbers.

- ✓ **Quadratic:** If the objective function f (x) is quadratic and the constraints $\mathrm{g_j}(\mathrm{x})$ are linear, the problem is called quadratic; usually it is formulated as follows:
  - $$\begin{cases} \min f(x) = c + \sum_{i=1}^{d} a_i x_i + \sum_{i=1}^{d} a_{ij} x_i x_j \\ g_i(x) = \sum_{i=1}^{d} c_{ji} x_i \ , j = 1, \ldots, k \ \ x_i \geq 0, i = 1, \ldots, n. \end{cases} \tag{4.5}$$
  - ✓ **According to the type of variables:** The variables of the optimization problem can be continuous or discrete values, and consequently, we find continuous optimizations problems and discrete ones.

- ✓ **According to the deterministic nature of the variables:** If some or all of the variables of an optimization problem are probabilistic (non-deterministic), the problem is stochastic programming, otherwise the problem is deterministic programming.

## 4.5. Optimization techniques

Optimization techniques can be classified into two categories according to the evolution of the resolution method whether or not it includes a random aspect, they are deterministic techniques or non-deterministic techniques (approximate) [115].

### 4.5.1. Deterministic techniques

Deterministic methods or exact techniques are introduced to solve optimization problems in a predictable manner; they guarantee that the absolute optimal solution is obtained. Among these methods, we can cite the Newton method, simplex method, and the gradient method, etc. The use of these methods is often successful. However, several disadvantages limit their use including [115]:

- Deterministic methods require convex objective function, continuous and differentiable.
- They are applicable as long as the number of variables is small.
- The computation time is often very long.

### 4.5.2. Non-deterministic techniques

When deterministic methods fail to solve an optimization problem or require a very long computation time, it is probably necessary to use non deterministic techniques (approximate techniques). Approximate or heuristic techniques use random numbers to explore the search space more efficiently to provide an approximate solution, without guarantee of optimality, in favor of a reduced computation time. Among these methods, we can cite constructive methods, local searches and metaheuristics [115].

In the following we will focus more particularly on metaheuristic techniques.

## 4.6. Metaheuristics

Metaheuristics are more advanced stochastic optimization algorithms designed to solve difficult optimization problems. The term metaheuristic has been invented by Fred Glover in 1986; it is derived from the composition of two Greek words:

- **Meta** which is a suffix meaning *beyond*, or *a higher level.*

-**heuristic** which comes from the verb heuriskein (euriskein) and which means *find* ;

The main quality of these methods is that they are adaptable to a large number of optimization problems. Usually, these methods start with a random number of candidate solutions, they change their positions iteratively before they converge to an optimal solution.

The term metaheuristics is used as opposed to heuristics. In fact, metaheuristics can be used for several types of problems, while a heuristic is suitable for a given problem. Metaheuristics have common characteristics through their stochastic characters, which mean the search part is conducted in such a random way. In addition to the stochastic basis, metaheuristics are generally iterative, i.e. the same schema search is applied several times during optimization [115]. They are direct, i.e. they do not use the information of the gradient of the function goal, and derive their interest in particular from their ability to avoid the local optima, either by accepting a degradation of the objective function during their progression, either by using a population of points as a search method. In general metaheuristic algorithms follow the flowchart shown in Fig.4.2

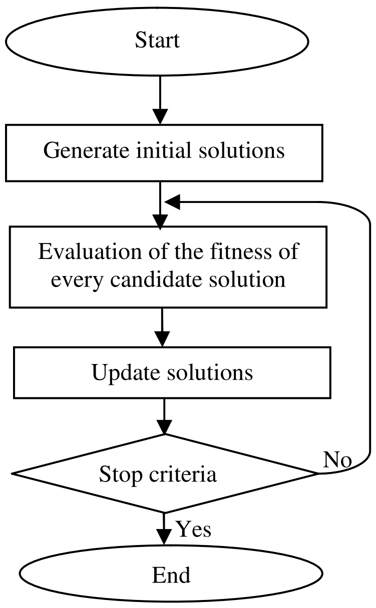

Start
Generate initial solutions
Evaluation of the fitness of every candidate solution
Update solutions
Stop criteria
No
Yes
End

Fig. 4.2 Flowchart of metaheuristic algorithms

The satisfying resolution of an optimization problem using metaheuristics depends greatly on its ability to provide a good balance between exploration (diversification) and exploitation (intensification) [115].

- **Exploration** is the algorithm's ability to visit different regions in the search space, to avoid trapped in local optimum.
- **Exploitation** is the ability of the algorithm to focus the search in the promising regions of the search space

## 4.7. Metaheuristic algorithms classification

A large number of metaheuristic algorithms have been published in these recent years; they can be categorized into three main categories, according to the number of solutions, to the search history or according to the inspiration [115], Fig.4.3.

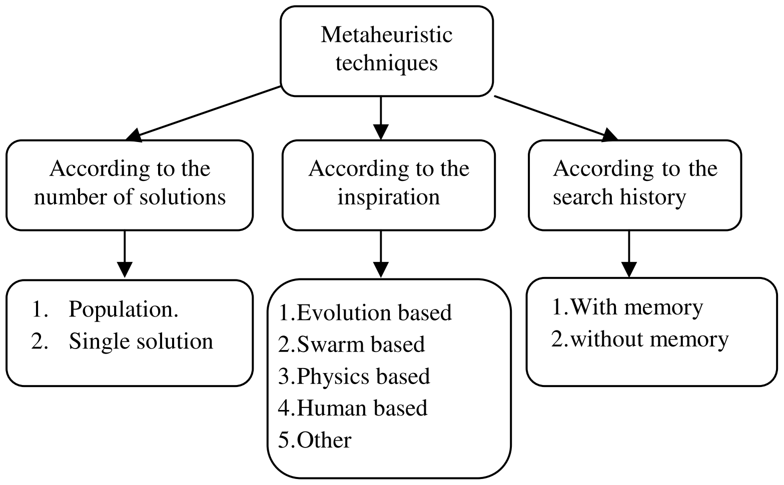


Fig. 4.3 Metaheuristics classification

According to the number of candidate solutions, we can find:

- **Single-solution based methods**, also called trajectory methods; they start with a single initial solution and gradually move away from it into building a trajectory in the research space. Among these methods we can cite: simulated annealing (SA) [116], taboo search [117], Greedy randomized adaptive search procedure [118], variable neighborhood search [119] and iterated local search [120].
- **Population based solutions**, manipulate a set of solutions. Among these methods, we can find stochastic fractal search (SFS) [121], particle swarm optimization (PSO) [122], sine-cosine algorithm [123], etc.

The common feature between population-based metaheuristic algorithms is the search process which can be divided into two phases [115]:

- **Exploration:** In this phase, the algorithm must globally explore the search space i.e., population motion should be randomized as most as possible.
- **Exploitation:** In this process, the algorithm investigates in detail promising areas of the search space. Finding a good balance between exploration and exploitation is the most challenging task in developing any metaheuristic algorithm due to the stochastic nature of the optimization process.

Generally, metaheuristics based on a single solution are more oriented toward exploitation, while population-based metaheuristics are oriented toward exploration [115].

According to the use of the search history or not, many metaheuristic techniques use their search history to guide optimization through iterations following, whether in the short term (solutions visited recently) or in the long term (memorization of a set of synthetic parameters describing the search) [115]; the taboo search method, for example, keeps in memory the last solutions visited and prohibits the return to them; and PSO method saves the best global solution and the best solution reached by each particle.

According to the inspiration we have:

- **Evolution-Based:** are inspired by biological evolution such as mutation, reproduction, recombination, and selection the biological evolution of species, the search process begins with a random population that evolves over generations. The best individuals in these methods are always combined together to generate the next population of individuals. This permits the optimization of the population over the course of generations; some of the most popular evolutionary-based techniques are genetic algorithms (GA) [124], genetic programming (GP) [125], evolution strategy (ES) [126], biogeography-based optimizer (BBO) [127], and differential evolution (DE) [128].
- **Physics-based:** In this category the search agents communicate between each other, to move throughout the search space according to physical rules of the universe such as: electromagnetic, force gravitational force, inertia force, and so on. The most popular algorithms are simulated annealing (SA) [116], black hole (BH) [129], colliding Bodies Optimization (CBO) [130], equilibrium optimizer [131], etc.
- **Swarm-based methods:** This category is based on the collective behavior of social creatures. The collective intelligence is inspired by the interaction of swarm with each other and their environment. The most famous algorithm of swarm inspired technique is Particle Swarm Optimization (PSO). Another popular swarm- intelligence technique is ant colony Optimization [132], monkey search [133], cuckoo search (CS) [134], bat-inspired Algorithm (BA) [135], and firefly algorithm (FA) [136].
- **Human behaviors:** there are also other metaheuristic methods inspired by human behaviors in the literature, some of the popular algorithms are harmony search (HS) [137], teaching-learning-based optimization (TLBO) [138], taboo search (TS) [117], group search optimizer (GSO) [139], imperialist competitive algorithm (ICA) [140], league championship algorithm (LCA) [141], firework algorithm [142].
- **Other inspiration:** there are also some techniques inspired by other sources, like simulated kalman filter [142], and sine cosine algorithm [123], etc**.**

According to "**no free lunch**" (NFL) theorem [144], if we consider the set of all possible optimization problems, there is no algorithm better than one other. In fact, the comparison of metaheuristic algorithms can only take place once the problem treated has been specified [145]. According to [146], optimization is not just a mathematical theory but there is experience which guides the user in the choice of the algorithm to implant.

According to recent research results of applying metaheuristics in different domains [115], it has been shown that the use of evolutionary algorithms provides many advantages, and they performed better than other metaheuristics, in very complex problems. This superiority is due to the fact that different solutions are selected randomly and then combined to create new solutions. The weighted combination of good partial solutions can produce very good overall results. The problem with these methods is the use of multiple updating processes which may increase the number of fitness function evaluations.

## 4.8. Extensions of metaheuristics

A very interesting characteristic of metaheuristic techniques is that they are naturally ready to extensions whose aim is to adapt them to a large number of optimization problems [147]. Among these extensions, we can cite:

- **Multi-objective:** this type of method does not seek a single optimal solution but a set of optimal solutions, called **Pareto** optimal. One solution is Pareto optimal if the improvement in one of the objective functions invariably leads deterioration relative to another objective function.
- **Multi-population:** algorithms with several populations (or swarms) use several subpopulations to perform different tasks, with or without the presence of overlap between subpopulations.
- **Parallel:** this implementation is used to speed up execution by the distribution calculation tasks with several processors.
- **Hybridization:** it consists of combining metaheuristic techniques, with other optimization methods, whether deterministic or metaheuristic in order to take advantage of respective advantages.

## 4.9. Metaheuristic algorithms used in the thesis

### 4.9.1. Stochastic fractal search algorithm

This subsection is devoted to the SFS algorithm, we start by describing SFS metaheuristic algorithm.

#### 4.9.1.1. The principle of SFS

Stochastic Fractal Search (SFS) is a metaheuristic technique population based, and belongs to the family of evolutionary algorithms. It is proposed by Hamid Salimi in 2015 [121], and inspired by the natural phenomenon of growth which uses the mathematical concept of fractal.

#### A. The initialization process

The SFS algorithm begins with a random initialization of a number Np of solutions candidates in the search space, on the basis of problem limitation, the initialization equation of the *i-th* point is addressed as follows:

- $P_i(\mathrm{j}) = \mathrm{P_{min}} - \mathrm{r}.(\ \mathrm{P_{max}} - \mathrm{P_{min}})$ (4.6)

Where $\mathrm{P_{min}}$ and $\mathrm{P_{max}}$ are the lower and the upper bounds limiting the search window, respectively.

After the initialization of the population, the fitness function is evaluated to measure the quality of each solution and determine the best solution. The SFS technique subsequently uses three main processes, a diffusion process and two update processes, to converge iteratively towards the best solution overall.

### B. The diffusion process

In the diffusion process, each candidate solution is diffused to generate new others solutions. The Fig.4.4 illustrates an example of the diffusion of a particle. To achieve this process, SFS uses two Gaussian random approaches, defined by equations: (4.7), (4.8) by making a random permutation between the two.

- $GP_i = Gaussian(\mu, \sigma) + (r.BP - r'.P_i)$ (4.7)
- $GP_i = Gaussian(Pi, \sigma)$ (4.8)
- $\sigma = \left|\frac{Log(g)}{g}(P_i - BP)\right|$ (4.9)

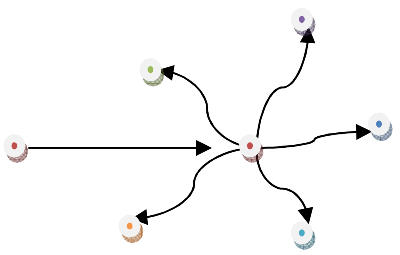


Fig. 4.4 Particle under diffusion process.

Where $\mathrm{P_i}$ is a candidate solution and BP is the best point solution found in the current iteration. The two random numbers r and r' are drawn uniformly from the interval [0.1]. μ and $\sigma$ are respectively the mean and the standard deviation of the Gaussian equations. The mean μ in equation (4.7) is the best global solution BP; and the term $\left|\frac{Log(g)}{g}\right|$ in (4.9) is used to decrease the size of Gaussian walks.

The first Gaussian equation generates a new solution $\mathrm{GP_i}$ in the neighborhood of the best overall solution BP. While, the second Gaussian equation generates a solution in the neighborhood of the candidate solution. The random permutation between these two equations allows exploiting once the zone of the best solution and again the zone of the candidate solution.

The standard deviation $\sigma$, calculated by equation (4.9) takes an increasing value at the beginning so that the generated solutions are far from the candidate solutions, in order to explore different regions in the search space. Then, $\sigma$ takes a decreasing value to concentrate the search in the promising areas.

At the end of this process, only the best points are considered, and the rests are discarded.

### C. The first updating process

In the first update process, a probability value $Pa_i$ is given to each candidate solution using equation (4.10), where rank ($P_i$) represents the ranking of the solution $P_i$. Then, if a candidate solution has a probability value less than a random number r in [0, 1], ($Pa_i$<r), its position is updated according to equation (4.11).

- $Pa_i = \frac{rank(P_i)}{Np}$ (4.10)
- $P_i'(j) = P_r(j) - r.(P_t(j) - P_i(j))$ (4.11)

Where $P_i'(j)$ is the new position of $P_i(j)$, $P_r(j)$ and $P_t(j)$ are random selected points in the group.

### D. The second updating process

In the second updating process, the probability values are calculated the same way as in the first process. Then, each solution that satisfies the condition $Pa_i$ <r, its position is updated by equations (4.12) and (4.13) by making a random permutation between them.

- $P_i''(j) = P_i'(j) - r.( P_t'(j) - BP)\ if\ r' \leq 0.5$ (4.12)
- $P_i''(j) = P_i'(j) - r.( P_t'(j) - P_r')\quad if\ r' > 0.5$ (4.13)

Where $P_t'$, $P_r'$ are random solutions selected from the first procedure. The new solution $P_i''$ update the solution $P_i'$ if its fitness function value is better than $P_i'$ . Both the processes diffusion and updating are repeated until the maximum iteration number is achieved. Fig.4.6 illustrates the pseudo-code of the SFS algorithm.

```
1.  Initialize a population of $N$p random points.
2.  Evaluate fitness function of the $N$p points.
3.  Sort all points Based on fitness Value
4.  While i < IterMax Do
5.      For i = 1 to Np Do
6.          Update solutions based on diffusion process.
7.      end For
8.      For i = 1 to Np Do
9.          Evaluate fitness function.
10.     end For
11.     Sort all points based on fitness value
12.     For i = 1 to Np Do
13.         Apply the first updating process.
14.         Evaluate fitness function.
15.     end For
16.     Sort all points based on fitness value
17.     For i = 1 to Np Do
18.         Apply the second updating process.
19.     end For
20. end While
21. Get the best solution
```

Fig. 4.5 Pseudo code of SFS algorithm

The SFS metaheuristique technique is a very efficient evolutionary algorithm, as it incorporates an efficient exploration of the search space by providing a good balance between exploitation and exploration.

The diffusion process generates new solutions in the neighborhood of each solution which allows the exploitation of promising regions. In addition, the standard deviation decreases during iterations, which allows the search region to be decreased, therefore be balanced between exploration and exploitation. On the other hand, the first and the second updating processes make it possible to diversify the search which improves exploration.

The author Hamidi [121], compared the optimization results obtained by SFS technique with those of other metaheuristics and he has shown that this method surpasses many other methods in case of uni-modal and multimodal test functions. In addition, SFS technique has been used in various applications and the results have been satisfactory [2,148,149].

The main disadvantage of SFS technique is that the number of evaluations of the fitness function is high.

### 4.9.2. Harris hawks optimizer

Harris Hawks Optimizer (HHO) is a novel nature-inspired, gradient free, and population-based optimization algorithm, proposed in 2019 by Ali Asghar Heidari et al. [150].

The algorithm mathematically mimics the Harris hawks group hunting strategy. The basic idea behind their smart strategy consists of cooperative attacks of the prey by surprise from different directions, in order to make the prey fully exhausted, then increasing the probability of capturing the prey. During hunting process, Harris Hawks may display different attacking motions and switching activities cooperatively, according to the prey's motion patterns.

Based on what was mentioned above, there are three phases for HHO:

#### 4.9.2.1. Exploration phase

In this phase Harris' hawks, detect and tracker their prey by moving randomly on some locations based on two strategies, formulated in two equations (4.14).

- $$P(t+1) = \begin{cases} P_{rand}(t) - r_1 |P_{rand}(t) - 2r_2 P_{rand}(t)| & q \geq 0.5 \\ \left(P_{rabbit}(t) - P_m(t)\right) - r_3\left(Lb + r_4(Ub - Lb)\right) & q < 0.5 \end{cases} \quad (4.14)$$

Where P(t + 1) is the hawks' position in the t+1 iteration, P(t) is the current position of the hawks, $P_{rabbit}$(t) is the current position of the prey; $r_1$, $r_2$, $r_3$, $r_4$, and q are random sets inside [0,1], updated every iteration. Lb and Ub are the lower and upper bounds of variables. $P_m$ is the average calculated position of the current population of hawks. $P_{rand}$(t) is a randomly selected hawk from the current population.

#### 4.9.2.2. Transition from exploration to exploitation

In this phase, the prey's energy will be decreased during chasing's process. The prey escaping energy can be modeled as follows:

- $$E = 2E_0\left(1 - {}^{t}/_{IT}\right) \quad (4.15)$$

$E_0$ is the prey initial energy; IT is the maximum number of iteration, and *E* is the prey escaping energy.

#### 4.9.2.3. Exploitation phase

In this process, the hawks organize a surprising and confusing attack to capture the prey. There are four steps to illustrate the attack motion pattern:

##### A. Step one Soft besiege

If *r* &|*E*| >=0.5: the prey have enough energy to escape, the hawks positions is modeled by the following equations:

- $$P(t+1) = \Delta P(t) - E\ |J\ P_{rabbit}(t) - P(t)| \quad (4.16)$$
- $$\Delta P(t) = P_{rabbit}(t) - P(t) \quad (4.17)$$

Where ΔP(t) is the distance between the location of the prey and the current position of the hawk, in the current iteration t; J=2(1-$r_5$) simulate the random jump of the rabbit. $r_5$ is random number inside [0,1].

### B. Step two Hard besiege

If $r$ &$|E|$ <0.5: The prey is tired, then the hawks positions is modeled by the following equations:

- $$P(t+1) = P_{rabbit}(t) - E\ |\Delta P(t)| \quad (4.18)$$

### C. Step three soft besiege with progressive rapid dives

If r < 0.5 & $|E| \geq 0.5$, the prey still have sufficient energy to escape, then the hawks positions are given by the following equation:

- $$X = P_{rabbit}(t) - E\ |J\ P_{rabbit}(t) - P(t)| \quad (4.19)$$
- $$Y = X + V \,.\, LF(D) \quad (4.20)$$

Where, D is the problem's dimension, V is a random vector with a size of 1x D; *LF(D)* is the Levy Flight function [149]. Thus the final action to update the hawks' positions is given by the following formula:

- $$P(t+1) = \begin{cases} X & if\ F(X) < F(P(t)) \\ Y & if\ F(Y) < F\big(P(t)\big) \end{cases} \quad (4.21)$$

### D. Step four, hard besiege with progressive rapid dives

If r & $|E|$ < 0.5, the rabbit have not enough energy to escape, then the hawks perform the action modeled as follow:

- $$P(t+1) = \begin{cases} X = P_{rabbit}\ (t) - E|J.P_{rabbit} - P_m(t)| & if\ F(X) < F(P(t)) \\ Y = X + V.LF(D) & if\ F(Y) < F\big(P(t)\big) \end{cases} \quad (4.22)$$

```
1. Initialize the position of N Hawks.
2. While i < IterMax Do
3. Evaluate fitness values of N Hawks.
4. Set $P_{rabbit}$ as the best Position of the rabbit from the best fitness value.
5. for j=1 to N do
6. Update E, E0 and J using Eq (4.14).
7. Update the E using Eq. (4.15)
8. if ( |E≥ 1|) then                                  Exploration phase
9. Update Hawks position using Eq (4.15).
10.     if ( |E< 1|) then.                            Exploitation phase
11.         if (r ≥0.5 and (|E|≥ 0.5 ) then           Soft besiege
12.         Update Hawks position using Eq. (4.16)
13.         else if (r ≥0.5 and (|E| < 0.5 ) then . Hard besiege
14.         Update Hawks position using Eq. (4.18)
15.         else if (r <0.5 and |E| ≥0.5 ) then .   Soft besiege with
                                                   progressive rapid dives
16.         Update Hawks position using Eq. (4.21)
17.         else if (r <0.5 and (|E| < 0.5 ) then .  Hard besiege
with progressive rapid dives
18.         Update Hawks position using Eq. (4.22)
19.     Return $P_{rabbit}$
```

Fig. 4.6 Pseudo Code of HHO

### 4.9.3. Whale optimization algorithm

Whale Optimization Algorithm (WOA) is a recent metaheuristique algorithm [151], proposed by Seyedali Mirjalili in 2016. The algorithm mimics the bubble-net hunting strategy of humpback whales, by searching of the global optimum, through encircling the prey then attacking it. Humpback whales prefer to hunt preys that are close to the surface using its unique hunting technique called as bubble-net feeding by creating a 9 shape bubbles or a circle as shown in Fig.4.8.

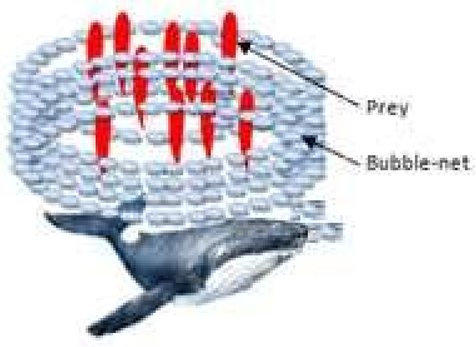


Fig. 4.7 Bubble-net feeding of the humpback whales

#### 4.9.3.1. Algorithm initialization:

The algorithm starts with a random initialization of N search agent candidate in the search space, this action is modeled by the following equation:

- $P_i=P_{min}+R*(P_{max}-P_{min})$ (4.23)

Where $P_i$ is the *i-th* initial position of the searching agent, $P_{min}$ and $P_{max}$ are respectively the lower, and upper bound which define the search space, and R is a random floating number in [0,1].

After initiating the N search agent, their fitness functions are evaluated to determine the best agent $P(t)_{best}$.

#### A.The hunting strategy

The bubble-net hunting strategy can be illustrated in the following three steps:

#### a.1. 1st step, Encircling the prey

Humpback whales can identify the preys' position and surround them. WOA suppose that the current $P(t)_{best}$ solution is near the optimum prey position. After identifying $P(t)_{best}$, the other positions are updated around it. This action is modeled as follows:

- $D=|C.P(t)_{best}-P(t)|$ (4.24)
- $P(t+1)= P(t)best-A.D$ (4.25)

Where**:** **A** and **C** are coefficient vectors calculated using the following formulas:

- $A=2.a.r-a$ (4.26)
- $C=2.r$ (4.27)

- $a=2-t*(2/iter\_Max)$ (4.28)

t indicates the current iteration, Pbest(t) is updated in every iteration, P(t) is the current position vector, | | denotes the absolute value, r is a random set in [0,1], a is decreased real number from 2 to 0 during iterations, iter_Max denotes maximum number of iteration.

### a.2. 2nd Step: Bubble-net attacking (Exploitation phase)

The bubble-net behavior is modeled by the following two stages as follows:

- **Stage one, shrinking encircling mechanism**

This conduct is realized by the fluctuation of the value of A in the equation (4.26), caused by the decreasing in value of a in Equation (4.28), then the updated position of the search agent can be set randomly, between its original position and the current best position in equation (4.25).

- **Stage two, spiral updating mechanism**

The distance between the positions of the whale and its prey is modeled by applying a spiral (a 9 shape), as in Fig.4.8, this is mathematically modeled as follows:

- $P(t+1)= D'.e^{bl}.\cos(2\pi l) +P_{best}(t)$ (4.29)
- $D'=|P_{best}(t)-P(t)|$ (4.30)

Where **b** is constant to set the logarithmic spiral shape, and l is random number between − 1 and 1

To update whale agents' positions there are two equal possibilities. The first possibility is to swim near the prey with a shrinking circle, and the second is to use spiral motion shape, these actions are modeled as follows:

- $P(t+1)\begin{cases} P_{best} - A.D & \text{if } q < 0.5 \\ D'.e^{bl}.\cos(2\pi.l) + P_{best}(t) & \text{if } q \geq 0.5 \end{cases}$ (4.31)

Where q is a random number in [0,1].

- **Step three, searching for the prey (Exploration phase)**

In this phase, the position of a search agent are updated using on a random chosen search agent. This behavior is carried when the absolute value of A is superior or equal to 1. The position of the search agent is updated as follows:

- $D''=|C.P_{rand}(t)-P(t)|$ (4.32)
- $P(t+1)=P_{rand}-A.D''$ (4.33)

Where $P_{rand}$ is a random whale position.

```
1. Initialize the position of N whale position.
2. Evaluate fitness values of N search agent.
3. pick the $P_{best}$ as the best agent position.
4. While i < IterMax Do
5. Update $a$ using Eq (4.28).
6. for j=1 to N do
7. Update $A,C,l,q$ parameters in Eq (4.26,27,31).
8. if ( q<0.5) then                 % Shrinking encircling
9.     if ( |A|< 1) then.          %Exploitation phase
10.        Update search agent position using Eq. (4.25)
11.    else if (|A| ≥1) then       % Exploration phase
12.        Update search agent position using Eq. (4.33)
13.    end if
14. if ( q≥0.5) then               %Spiral updating mechanism
15.         Update search agent position using Eq. (4.29)
16. end if
17. end for
18. Check that all search agents are inside the search space
19. i=i+1
20. Return $P_{best}$
```

Fig. 4.8 Pseudo Code of WOA

### 4.9.4. Equilibrium Optimizer

Equilibrium Optimizer (EO), is a recent metaheuristic algorithm, physical inspired, and population-based optimization algorithm proposed in 2019 by Afshin Faramarzi, et al. [131].

EO mathematically mimics the adjustment of mass balance to estimate both dynamic state and equilibrium state.

The simple inspiration of EO approach is a, well-mixed mass balance on a control volume.

Wherein the mass balance equation is employed to describe the concentration of a non-reactive component in a system of controlled volume. The mass balance equation provides the basic physics for maintaining a mass entry, exit, and composition into a control volume system.

In EO algorithm, each agent tune its position depending on the best agents found so far, to reach optimum equilibrium state. A strategy named as "**Generation rate**" is exploited to improve the ability of EO in both exploration, and exploitation.

#### 4.9.4.1.Initialization of agents and evaluation of fitness function

In this phase, every agents (position) simulates a concentration; EO begins with a random initial position (concentration) of every agents *i*, as follows:

- $CT_i^{initial}=CT_{min}+R.(CT_{max}-CT_{min})\ _{i=1,2,...n}$ (4.34)

Where $CT_i^{initial}$ indicates the initial position of the agent *i*, $CT_{max}$ & $CT_{min}$ represents respectively, the maximum and minimum boundary values of the search space, R is a random number in [0,1], and n denotes the population size. Subsequently, the fitness function of every agent is evaluated and sorted to decide the balance candidates.

#### 4.9.4.2. Equilibrium pool

The global optimum is reached when the equilibrium state is achieved.

Let's denote by $CT_{eq5}$ the arithmetic average of the best four agents ($CT_{eq1,...4}$) identified and updated during optimization process. $CT_{eq5}$ agent helps in exploitation, while the other agents ($CT_{eq1,...4}$) helps in exploration.

These five agents are called as equilibrium candidates, and they are employed to create another vector called as the equilibrium pool.

- $CT_{eq,Pool}=\{CT_{eq1}, CT_{eq2}, CT_{eq3}, CT_{eq4}, CT_{eq5}\}$ (4.35)

#### 4.9.4.3. Exponential term

To balance between exploration and exploitation, EO use the exponential term (F), to update search agent positions, as follow:

- $F=a_1. \text{Sign}(R-0.5)*(\exp^{-\lambda.t}-1)$ (4.36)
- $t = (1 - Iter \,/\, IterMax)^{(a2\frac{Iter}{IterMax})}$ (4.37)

Where **IterMax** and **Iter,** stand for respectively the maximum number of iterations, and the current iteration, λ is random vector in [0, 1], $a_1$, $a_2$ are constant numbers used respectively for managing exploration and exploitation.

#### 4.9.4.4. Generation rate

Enhancing exploitation is supported by the generation rate $G_r$. Its major goal is to give the best solution, it is expressed as follows:

- $Gr = Gr_0.F$ (4.38)

Where:

- $Gr_0=GCP.(C_{eq}- \lambda.\ CT_i)$ (4.39)
- $GCP = \begin{cases} 0.5 * r_1 & r_2 \geq GP \\ 0 & r_2 < GP \end{cases}$ (4.40)

Where: $r_1$, $r_2$ are random real numbers in [0,1], GCP stands for generation rate control parameter, is the vector that controls the Generation rate, GP is Generation probability, it defines how many search agents use the GCP parameter.

*GP*=0.5 allows to reach a good equilibrium between exploitation and exploration. At last, the updating rule for EO search agents is formulated as:

- $CT = CT_{eq} + (CT - CT_{eq}).F + \frac{Gr}{\lambda.V}(1 - F)$ (4.41)

Where: F is the same defined in Eq.(4.36), and $V=1$.

#### 4.9.4.5. The memory saving of the search agents

EO have a similar strategy of $P_{best}$ in PSO, where each agent with its fitness value in the current iteration, update it position and fitness value, if it was better than the previous one, this strategy support the exploitation's ability.

The pseudo code of EO is displayed in Fig.4.10

```
1.  Initialize the position of N search agents.
2.  Assign values for the parameters $v$,$a_1$,$a_2$,GP.
3.  While Iter < IterMax Do
4.  For i=1 to N do
5.  Evaluate fitness values of the i-th search agent
6.  Update ($CT_{eq1}$ , $CT_{eq2}$, $CT_{eq3}$ ,$CT_{eq4}$) and their corresponding Fitness according to B.
7.  End For .
8.  Set $CT_{ave}$ as the average position of the 4 best search agent.
9.  Set the equilibrium pool as in (4.35)
10. if (Iter >1) then establish memory saving.
11. Set t as in (4.37)
12. for j=1 to N do
13. choose a random candidate from $CT_{eq,Pool}$
14. Generate $\lambda$ & R.
15. Set F as in (4.36)
16. Set GCP vector as in (4.40)
17. Set $G_{r0}$ as in (4.39)
18. Set $G_r$ as in (4.38)
19. Update CT as in (4.41)
20. End for
21. Iter=Iter+1
22. End while
23. Return $CT_{eq1}$
```

Fig. 4.9 Pseudo Code of EO

## 4.10. Conclusion

In this chapter we have presented optimization techniques, whether deterministic or non-deterministic. Metaheuristics are presented as non deterministic solution method to some optimization problems. Metaheuristics are stochastic methods that aim to solve a wide range of problems, without any need to modify their structure by the user. These methods are categorized according to many considerations. Metaheuristics quickly met with great success thanks to their simplicity employment but also to their high modularity. We have seen all along in this chapter that, according to the nature of the problem posed (continuous / discrete, single-objective / multi-objective, etc.), metaheuristics are easily extended and / or hybridized in order to obtain the best possible performance. We conclude that the richness of the different optimization methods leads us to choose the right compromise of methods and algorithms in order to solve a problem.

# CHAPTER 5

# PROPOSED OBJECT TRACKING TECHNIQUES BASED ON METAHEURISTICS

## 5.1. Introduction

In the context of visual object tracking, many techniques have been proposed. To better understand them, we divided the components of a tracker into two components, the feature presentation and the search technique.

Compared to the research on feature presentation, there are not so many on search techniques[8].

Among them are methods based on metaheuristics as a searching technique. According to the literature, the results of these tracking methods are better than many other deterministic and probabilistic methods, especially in terms of precision and under difficult challenges, such as fast motion, and occlusion, etc. However, the major problem with the proposed techniques in this context have some limitations, for example tracking methods based on quantum particle swarm optimization (QPSO)[43], moth-flame Optimization algorithm (MFO) [52], cuckoo search (CS) [152], Modified Flower Pollination Algorithm (MFPA) [12], Simplified Swarm Optimization(SSO)[49], BAT algorithm (BA) [8], Particle Swarm Optimization (PSO) [153], etc, suffer from the problem of premature convergence, and they can fall on local optima.

In this chapter, a three new VOT algorithms based on a metaheuristics such as: the stochastic fractal search (SFS), Harris Hawks optimization (HHO), Whale optimization(WOA) algorithm, and a MOT approach based on Equilibrium optimizer (EO) is also proposed. The choice of SFS, HHO, WOA, and EO is mainly due to the fact that these techniques can provide a good balance between exploitation and exploration as they provide good tracking results. The developed algorithms are implemented directly, and combined with color histogram as object presentation. For these proposed trackers Bhattacharyya distance is measured between the two histograms of the template and the candidates to define the fitness distance, in which optimization is sought.

## 5.2. Color histogram as a feature used in VOT

VOT is the process of locating the object region over time in a video sequences. The problem of object tracking can be expressed as a dynamique optimization problem and solved using metaheuristic techniques. Metaheuristic based trackers are algorithms that uses the same presentation for both the target template and the candidates, and search for the most similar candidate to a previously defined template in each frame.

Comaniciu et al. [5] used a **histogram weighted by an isotropic kernel** calculated over a rectangular region to represent the object, and Mean-shift as a search mechanism.

In our work, we used the same presentation, and metaheuristic techniques such as: SFS, and HHO, are used instead of mean-shift algorithm. The principle behind metaheuristic based tracker is to maximize the similarity of appearance by comparing the histograms of the target model and candidates inside a defined window around the hypothesis target position.Fig.5.1. The similarity between two histograms is defined by the Bhattacharyya coefficient. At each search agent, the fitness distance is optimized by calculating the similarity between the histograms.

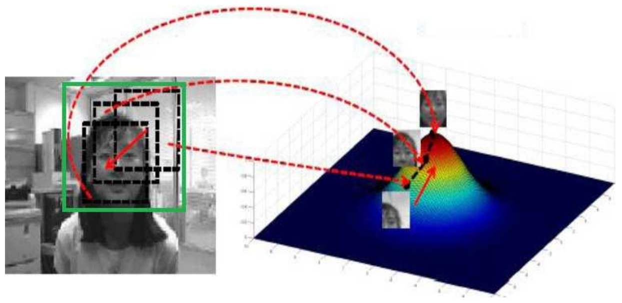


Fig. 5.1 Illustration of the search process of a metaheuristic algorithm
The green rectangle defines the search region [5].

## 5.3. The Kernel color weighted histogram (KCWH) presentation

Usually in VOT, the target is represented by a rectangular region of size R=($H_x$, $W_y$), where $H_x$ and $W_y$ are its height and width respectively. An HSV space is generally used to evaluate the histogram of the distribution of pixels of the target region. The histogram is represented by the target model: $\hat{q} = \{\hat{q_u}\}_{=1\ldots m}$ of m color bins, which used to describe the appearance of the object located in the target region.

We denote by $\{x^*_i\}_{i=1\ldots n}$ the set of coordinates of n pixels of the target model, centered at x, and normalized by the radius of the rectangle $H_x$ and $W_y$. The color histogram is calculated using a convex (isotropic kernel) profile function $(x)$. Fig.5.2 is a typical example of that kernel.

The kernel $(x)$ gives more importance to pixels near the center rather than those at the edges.

The function $b$: $R^2 \to \{1\ldots m\}$ associates with each pixel at $x^*_i$ the index $b\ (x^*_i)$ of its color in the m-histogram. The probability of the color $u$= 1… $m$ in the target model is calculated as follows:

- $$\hat{q} = C \sum_{i=1}^{n} k\left(\| \frac{x-x_i^*}{R} \|^2\right) \delta[b(x_i^*) - u] \qquad (5.1)$$

Where $\delta$ is the Kronecker function. The normalization constant C is given by imposing the condition $\Sigma\hat{q_u} = 1$. The constant C is given by:

- $$C = \frac{1}{\sum_{i=1}^{n} k\left(\|\frac{x-x_i^*}{R}\|^2\right)} \qquad (5.2)$$

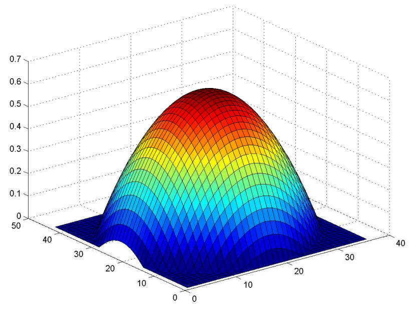


Fig. 5.2 Three-dimensional surface plot of the kernel used

The use of the presented histogram makes it possible to limit the influence of the background and focus on relevant information. The weighting defined by the spatial kernel Fig.5.2 gives a greater weight to the pixels closer to the center of the object, while a lower weight to the pixels far from the center of the object (background pixels). Fig.5.3 gives a good illustration of the kernel color weighted histogram presentation.

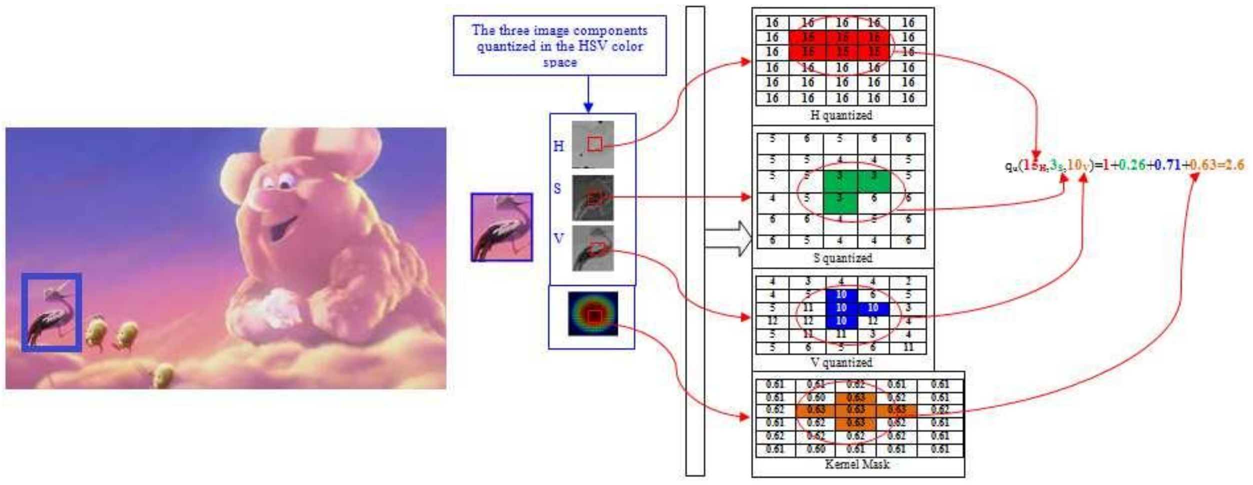

The three image components quantized in the HSV color space
H
S
V
H quantized
S quantized
V quantized
Kernel Mask
$q_u(15_H,3_S,10_V)=1+0.26+0.71+0.63=2.6$

Fig. 5.3 Example of the kernel CWH presentation.

The presentation of the center location of the object with distribution of color $\hat{q}_u(U_H=15,U_S=3,U_V=10)$ using one summation[9].

## 5.4. The kernel function

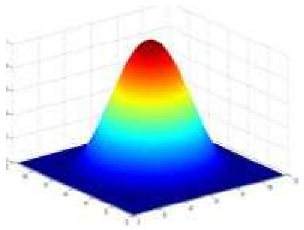
A kernel is a function defined on a space ε with real positive values which satisfies the following properties [153]:

- $\int_{\varepsilon} k(x)dx = 1$ (5.3)
- $\forall x \in \varepsilon, K(-x)=K(x)$ (5.4)

There are a multitude of spatial kernels used; Fig.5.4 shows examples of the most commonly used kernel functions. Our proposed algorithms uses Epanechnikov's kernel, which gives greater weights to the pixels near the center of the object, because the most important information is on most of the cases in the center.

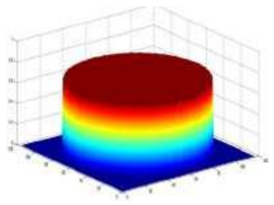
- **Gaussian kernel:**

$$k(x) = \begin{cases} \lambda e^{(-\frac{1}{2}|x^2|)} & si \quad |x| \le h \\ 0 & si \quad |x| > h \end{cases}$$



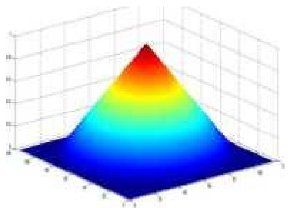
- **Uniform kernel**

$$k(x) = \begin{cases} \lambda & si \quad |x| \le h \\ 0 & si \quad |x| > h \end{cases}$$



- **Triangular kernel**

$$k(x) = \begin{cases} \lambda\,(1 - |x|) & si \quad |x| \le h \\ 0 & si \quad |x| > h \end{cases}$$



- **Epanechnikov's kernel:**

$$k(x) = \begin{cases} \lambda(1 - |x^2|) & si \quad |x| \le h \\ 0 & si \quad |x| > h \end{cases}$$

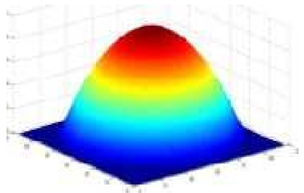


Fig. 5.4 Four different types of kernel functions [153]

## 5.5. The similarity measure

The similarity measure between the two histograms of target model and target candidate is given by the Bhattacharyya distance. This distance represents the overlap between two sets of samples. Bhattacharyya distance is known in the statistical world as a measure of similarity between two statistical distributions. The Bhattacharyya coefficient is defined by the following equation:

- $d(y) \equiv d[h_1(y), h_2] = \sum_{n=1}^{m} \sqrt{h_1(y).h_2}\ , \forall\ h_1(y), h_2,\ 0 \le d \le 1$ (5.5)

This distance has been used by several works to measure the similarity between two histograms in the context of visual tracking. Locating the desired region means minimizing the said distance d.

By using the Bhattacharya distance as the fitness function to be optimized, the metaheuristic algorithm such as SFS & HHO searches for the most similar candidate around the most probable area surrounding the previous target region.

## 5.6. Object tracking by SFS and KCWH

In this section, we present a new approach of object tracking using stochastique fractal search algorithm(SFS) and Harris hawks optimization algorithm(HHO). These two algorithms has been proven to be robust against getting trapped in local optimum, and combined with kernel KCWH, they have given a good tracking results under partial occlusion, size change, rotation and non-rigid target deformation.

## 5.7. The relationship between optimization and visual tracking

According to oxford dictionary, optimization is defined as " **the action of making the best or the most effective use of a situation or resource**."

At the light of this definition, resolving an optimization problem can be defined as the process of selecting the best element among certain sets of available alternatives under certain constraints. All optimization problems that have an explicit objective function can be expressed as nonlinearly constrained optimization problems in the following form [152]:

- Minimize or maximize f(x), $x \in R^n$ (5.6)

Where f is the cost function or the objective function. $R^n$ is the search space, and x is the decision vector varying in an N-dimensional space $R^n$.

In the tracking process, and under the assumption [2,9,12] that the target's speed, can't exceed the target dimensions(width and height) between adjacent frames to define the search space, so the constraints; the observation distance between the object in interest and the candidate form the objective function (similarity function). Therefore the location of the target can be interpreted as an optimization problem by maximization or minimization of the similarity function.

Thereafter visual tracking can be considered as the process of finding the most similar candidate region to a previously defined template target defined from the first frame.

Like what we have mentioned in the state of art chapter, generally any visual tracking algorithm, have at least two essential components, namely: feature representation, and a target search strategy.

The target in our case is presented using KCWH defined in the 3rd previous section, which is an efficient target representation. To measure the similarity between the target and the candidates Bhattacharyya distance (BC $[H_1,H_2]$), is defined as follows:

- $BC[H_1, H_2] = 1 - \sum_{u=1}^{m} \sqrt{H_1(u)H_2(u)}$ (5.7)

Where: m=16, is the number of bins in the histograms, $H_1$ and $H_2$ are respectively the histograms of template and the candidate, being compared. It is noted that $BC[H_1,H_2]$ is small when the histograms are similar, and large when they are very different.

In the tracking process, the object in interest can be localized by searching and measuring every possible candidate. This process is very challenging, because the object in interest undergoes different changes such as: deformation, size change, rotation, and occlusion, etc. Therefore, an efficient searching technique for state estimation is essential to get the exact the target's state.

## 5.8. SFS implemented for visual tracking

To implement SFS in visual tracking, the target state must be initiated manually from the first frame using a bounding frame around it. Defined by a vector $P_i = (x_i, y_i, w_i, h_i)$ where $\{x_i, y_i\}$ denote the 2-D translation coordinate of the target, and $\{w_i, h_i\}$ are respectively its initial width and height.

Second, the object is modeled using its color histogram explained in the third previous section.

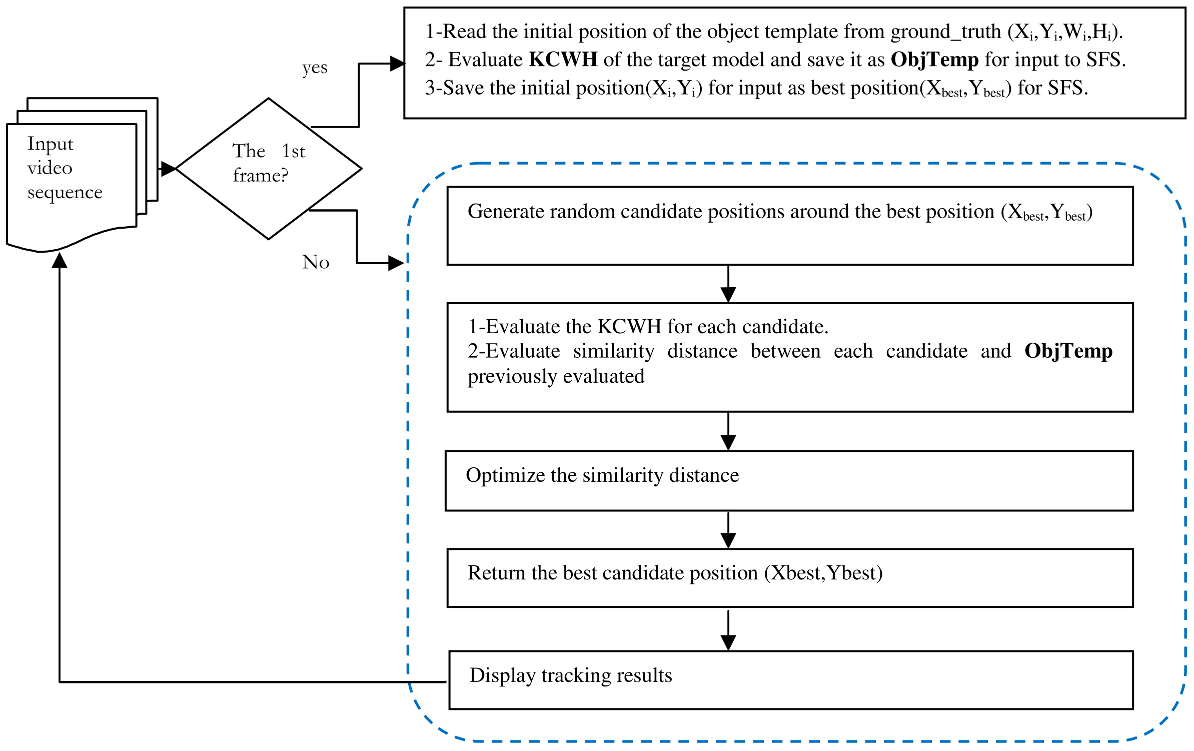
The target localization procedure starts from the position of the target in the first frame, then on each frame it is assigned to SFS to find the best candidate region using Bhattacharyya Coefficient as fitness function.

The constraints defined in this case of visual tracking are the definition of the search window. There are two parameters to define it, which are lower boundary $P_{min}$, and the upper limit $P_{max}$ defined by the following relation:

- $P_{min}=P_i(t)-w_i$ (5.8)
- $P_{max}=P_i(t)+w_i$

Where: Pi(xi,yi) represents the current position of the target object.

Fig.5.5 shows the SFS flowcharts and its implementation, while the search space of SFS tracking algorithm is illustrated in Fig.5.6. It can be described as follows, for each current object position in the current frame, the algorithm searches for the most similar region in the next frame within a search window of size (2wi × 2wi). The best-matched region is the candidate region that optimizes the fitness function.



1-Read the initial position of the object template from ground_truth ($X_i$,$Y_i$,$W_i$,$H_i$).
2- Evaluate **KCWH** of the target model and save it as **ObjTemp** for input to SFS.
3-Save the initial position($X_i$,$Y_i$) for input as best position($X_{best}$,$Y_{best}$) for SFS.
yes
Input video sequence
The 1st frame?
No
Generate random candidate positions around the best position ($X_{best}$,$Y_{best}$)
1-Evaluate the KCWH for each candidate.
2-Evaluate similarity distance between each candidate and **ObjTemp** previously evaluated
Optimize the similarity distance
Return the best candidate position (Xbest,Ybest)
Display tracking results

Fig. 5.5 SFS tracking algorithm flowchart.

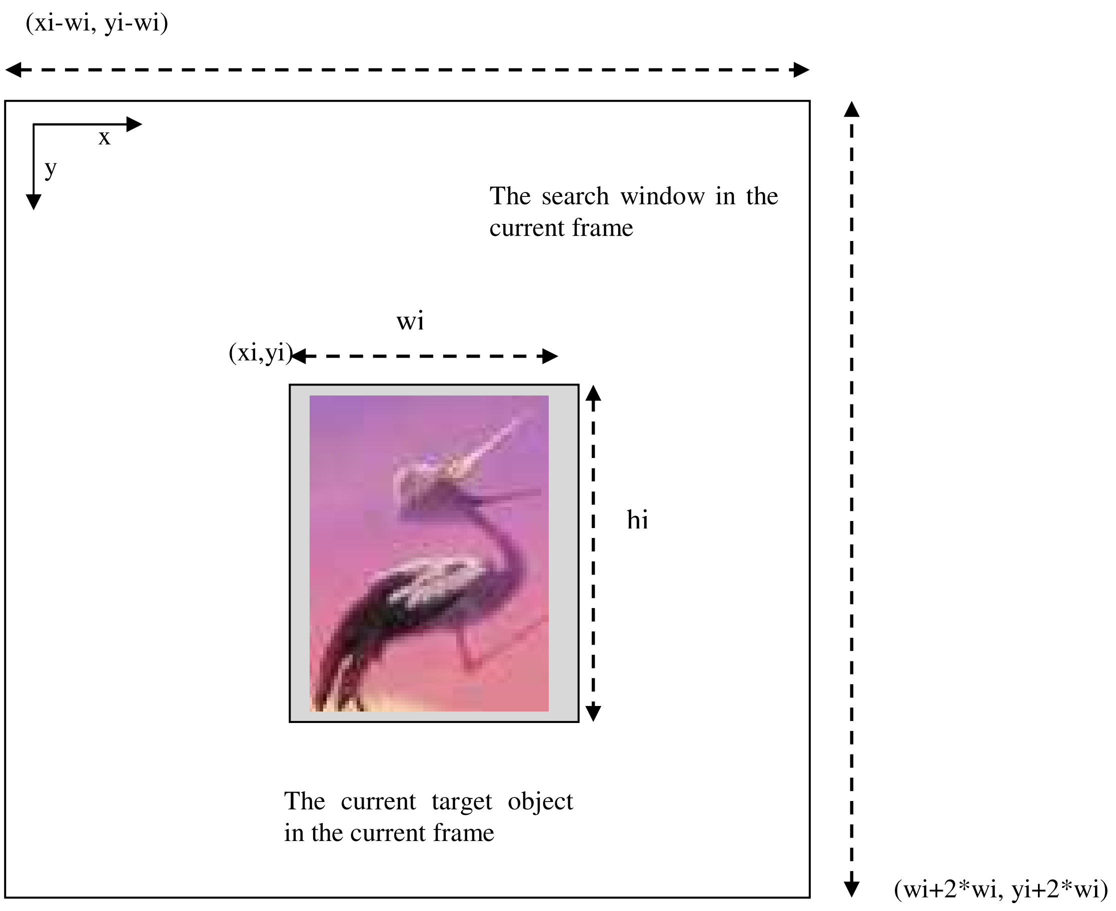

(xi-wi, yi-wi)
x
y
The search window in the current frame
wi
(xi,yi)
hi
The current target object in the current frame
(wi+2*wi, yi+2*wi)

Fig. 5.6 The horizontal and vertical displacements margins of the target object

### 5.8.1. SFS algorithm tuning parameters

There are two difficult factors involved in tuning the parameters of any metaheuristic algorithm.

The first is the algorithm's ability to converge by the smallest number of populations, and second, its ability to converge to the best candidate solution.

In our case, we tried to adjust the size of the population, in order to find the optimal value of this parameter to get the smallest number of fitness function's evaluation.

We did our analysis using the Boy sequence Fig.5.7. by varying the size of the population N. The size of the population varied from 10 to 50 with a spacing of 10.

Fig. 5.7 Boy sequence used for adjusting population size.

The boy sequence undergo many challenges such as: scale change, blurred motion,

fast motion, in/out of plane rotation.

The algorithm was run 10 times on the Boy sequence Fig.5.7 and the average tracking performance results were displayed in Fig.5.8.

Performance is evaluated using the accuracy rate, which we have defined as the predicted number of frames whose overlap is region is greater or equal to 0.5 over the total length of the video sequence.

The overlap region is calculated between the predicted bounding box and ground truth.

Fig.5.8 shows the performance evaluation by using different values of N. According to Fig.5.8 performance curve, when the population size N increases from 10 to 30, the accuracy improves almost linearly, but when the population is superior or equal to 30 the accuracy is at its maximum, and no improvement, thus in the experiment section, N chosen value is equal to 30, which present the maturity of the algorithm to achieve its maximum performance.

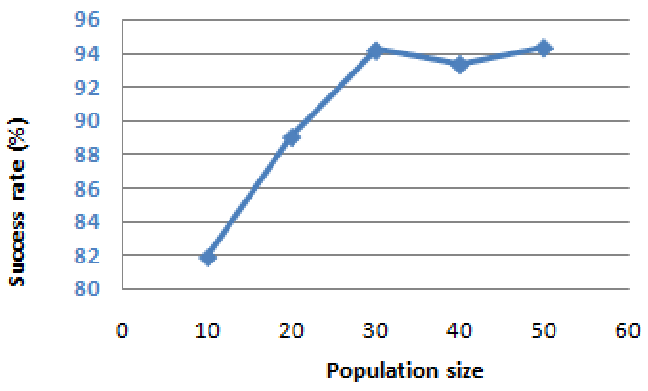


Fig. 5.8 Tracking performance with different population size

### 5.8.2. Platform implementation

The proposed tracking framework is implemented in Matlab R2013 on a lap top with Intel I5-5200U 2.20 GHZ CPU with 6 GB RAM, and without GPU.

The performance of SFS based tracker is given by using 20 video sequences from OTB-15 Benchmark [84], publicly available on the website (www.visual-tracking.net).

### 5.8.3. Tracking comparison

The evaluation of the proposed tracker is made by comparison with other 11 state-of-the-art trackers, categorized into three classes according to their searching techniques:

**a. Metaheuristic methods:** Six approaches, including SFS, QPSO [43], MFO [52], SPSO [153], MFPA [12], (CS) [152].

**c. Probabilistic methods:** Three approaches, including Color-Based particle filter (CPF) [155], Adaptive Structural Local Sparse Appearance Model (ASLA) [78], and Sparsity-based Collaborative Model (SCM) [21].

**c. Deterministic methods:** Three approaches, including Kernel Correlation Filter (KCF) [73], the traditional kernel mean shift tracking KMS [5], and Structured output tracking with kernels (Struck)[69].

Concerning metaheuristic trackers, the same target model (KCWH) are used for all the trackers, and the same initialization; for the parameter setting of each algorithm, they are implemented as their authors used in their experiments, expect population size, which is taken as 30 for a fair comparison.

The parameter setting of each metaheuristique algorithms compared are depicted in table 5.1.

Table 5.1 Sensitivity parameters of every algorithm

| Algorithm | Parameters |
|---|---|
| **SFS** | Maximum diffusion number is set to 1. |
| **QPSO** | No specific parameter. Same as [43]. |
| **MFO** | Constant of logarithmic spiral shape, b = 2. Same as in [52] |
| **SPSO** | W, C1 and C2 are set the same as in [153]. |
| **MFPA** | No specific parameter. Same as [12]. |
| **CS** | Discovering probability, pa =0.4. Same as in [152]. |

### 5.8.4. Performance evaluation

The effectiveness of the proposed tracking algorithm is carried extensively using quantitative and qualitative comparison.

Through quantitative comparison we have used three methods:

• The first one by using OPE on 20 video sequences.

• The second one by using average overlap+ average Euclidean distance of 10 video sequences.

• The third one is attribute based comparison.

Through qualitative comparison, the tracking results are displayed using snapshots of 7 video sequences.

#### 5.8.4.1. Quantitative evaluation

##### a. The first quantitative evaluation

This evaluation is given using two metrics namely:

1. **The overlap ratio**: which represents the average number of frames whose predicted overlap between the estimated bounding box and the given one (ground-truth) at a given threshold 0.5.

2. **The center location error:** which is the average value of the frames whose predicted positions are within the threshold of 20 pixels.

The experiments are carried by following the protocol in [84], by using the same parameter values for all the sequences.

Fig.5.9 and Fig.5.10, depicts the overall performances of SFS tracker, by showing the results under one-pass evaluation (OPE), using the two mentioned metrics. The figures are given using the toolkit publicly available at the website (http://www.visual-tracking.net). For clarity purposes, we only showed the performances of the top 10 trackers.

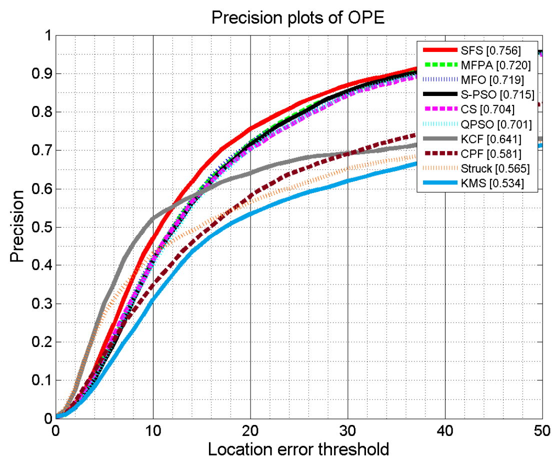


Fig. 5.9 The results under OPE metric, using location error threshold.

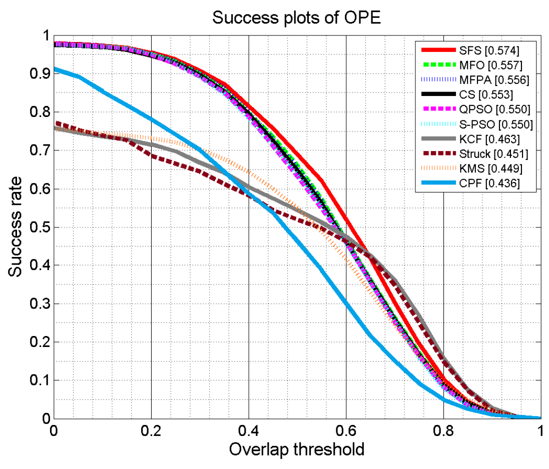


Fig. 5.10 The results under OPE metric, using Overlap threshold.

As shown in Fig.5.9 and Fig.5.10, SFS based tracker performs effectively against the 11 reference trackers in both distance precision rate (DPR) of 75.6% and overlap success rate (OSR) of 57.4%.

The efficiency of SFS tracker is demonstrated with an improvement gain of **3.6%** in DPR and **1.7%** in OSR. Among the existing reference methods, it is clear that SFS-Tracker outperforms the others in terms of DPR and OSR.

We can also notice that the convergence rate of the SFS is the best on the two curves. Additionally, all the reference metaheuristic outperforms deterministic and probabilistic approaches.

MFO and MFPA performance can considered the same, as they rank the second and third best results with (72%) DPR for MFPA, and (71.9%) DPR for MFO, and (55.7%) OSR for MFO, and (55.6%) OSR for MFPA.

### b. The second quantitative evaluation

The results in table 5.2 summarize the comparison between our proposed algorithm and the ten reference trackers on 10 video sequences that contain many challenges. This is another quantitative performance validation methodology by using both the average overlap and the average Euclidean distance from the central location error.

Table 5.2 shows that our algorithms (SFS-tracker) have the best average performance (the average minimum location distance error and maximum overlap ratio) than the comparison methods.

KCF and Struck provided the best results in three sequences namely: BlurFace, Boy and Deer.

Table 5.2 The average overlap and center location error in 10 video sequences.

| Sequence | Struck | KCF | QPSO | MFO | CS | MFPA | SFS |
|---|---|---|---|---|---|---|---|
| Boy | ***0.75/3.84*** | **0.78/2.67** | 0.63/7.06 | 0.64/6.63 | 0.64/6.73 | 0.63/6.81 | 0.68/5.62 |
| Skating2.1 | 0.24/57.88 | 0.35/30.76 | **0.51**/***27.85*** | **0.51**/28.52 | 0.50/28.98 | **0.51/27.74** | **0.51**/28.19 |
| Girl2 | 0.22/144.35 | 0.06/264.58 | ***0.64***/17.41 | ***0.64/17.17*** | ***0.64***/17.48 | ***0.64***/17.84 | **0.65/16.95** |
| BlurFace | 0.47/42.35 | **0.80/8.36** | 0.66/18.03 | 0.66/17.86 | 0.67/17.61 | 0.67/17.88 | ***0.74/12.78*** |
| Jogging-2 | 0.20/107.69 | 0.13/144.03 | 0.71/10.11 | 0.71/10.09 | ***0.72/9.67*** | 0.70/10.08 | **0.72/9.27** |
| DragonBaby | 0.20/69.26 | 0.31/50.40 | 0.54/15.40 | ***0.56***/14.77 | ***0.56***/14.89 | ***0.56/14.49*** | **0.59/13.13** |
| Lemming | 0.48/37.75 | 0.38/77.97 | ***0.63***/14.36 | 0.62/14.53 | ***0.63***/14.34 | ***0.63/14.03*** | **0.65/13.15** |
| Bolt2 | 0.22/86.41 | 0.01/329.81 | ***0.46***/37.87 | 0.45/***37.49*** | 0.45/37.51 | 0.43/39.44 | **0.47/37.03** |
| BlurBody | 0.72/13.97 | 0.43/64.11 | ***0.63/24.93*** | **0.63**/25.14 | ***0.63***/25.80 | ***0.63***/25.73 | **0.65/22.55** |
| Deer | **0.73/5.27** | 0.62/21.27 | 0.48/25.6 | 0.59/15.30 | 0.52/21.18 | 0.56/18.48 | ***0.63/11.90*** |
| Average | 0.43/56.88 | 0.39/99.40 | 0.59/19.86 | ***0.60/18.75*** | ***0.60***/19.42 | ***0.60***/19.25 | **0.63/17.06** |

The bold numbers indicates the best performance, while the italic bold indicates the second best.

The other four reference meta-trackers primarily provide the second best results for the rest of the sequences. This is due to the strong metaheuristic search capacity compared to deterministic.

### c. The third quantitative evaluation

In this section, we analyze in more detail our tracker performance on 11 different challenges (for example, background clutter, occlusion, and abrupt motion). Fig.5.11 shows the OPE for 20 video attributes. From this figure we have the following observations:

**First,** the superiority of the proposed tracker SFS compared to the other meta-trackers can be justified by its good exploration/exploitation mechanism, which differs it from the other metaheuristic trackers.

And the superiority of metaheuristic based trackers compared to deterministic and probabilistic ones in situations such as: Fast motion, Deformation, Blurred motion, and in plane rotation, is due to the strong search ability of the metaheuristic searching techniques compared to deterministic and probabilistic ones.

**Second,** our tracking approach is effective in managing probably the most difficult situation in visual tracking which is: occlusion, as our algorithm locally search for the best candidate around the previous found position, when the target object is out of view or fully occluded generally it reappears around its last position, thus our approach can handle better occlusion than the reference trackers.

**Third,** in out of plane rotation situation, generally the target object color do not change considerably, in this case the performance of the search model intervenes, which makes metaheuristic based searching techniques the top trackers.

**Fourth,** in situations such as: Background clutter, and illumination changes, the sequences used do not undergo much of these situations, which justify of the obtained results.

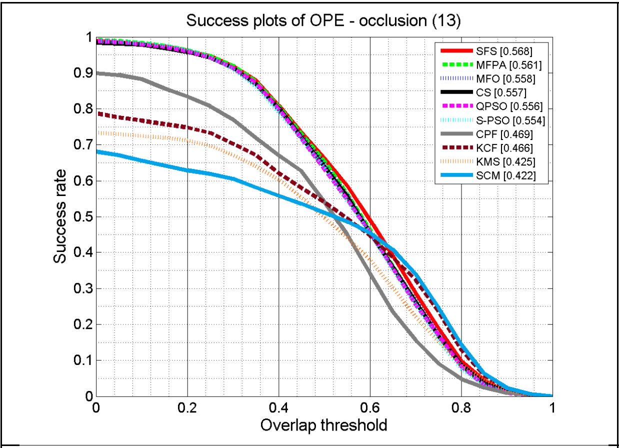
In low resolution situation, the performances of our proposed approach are not the best but provided satisfying tracking results. In overlap success rate, Struck algorithm has a score of **61.5%,** KCF: **55.5%**, and SFS: **51.9%**.

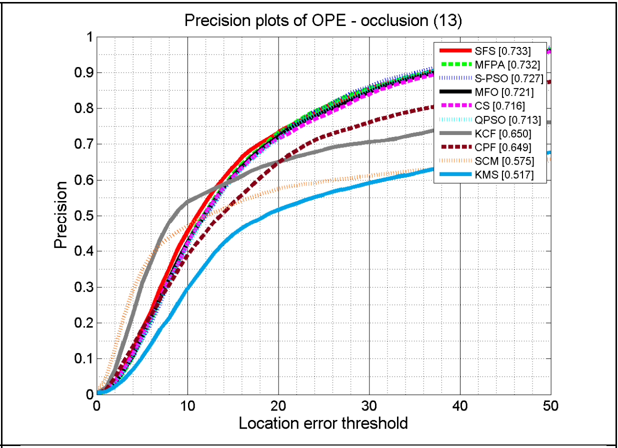
To overcome this issue we can propose to increase the number of bins in the evaluated histogram and get better tracking results.

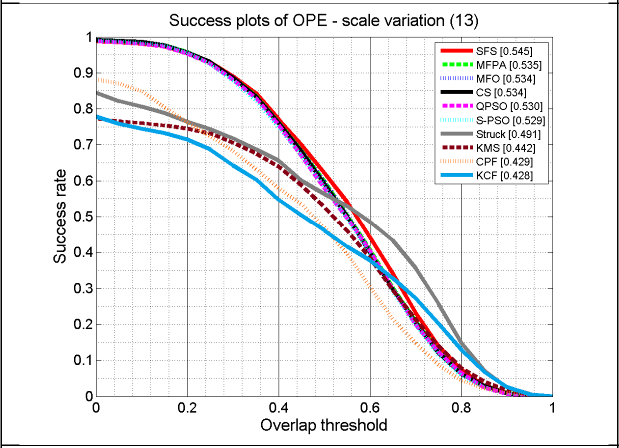
In situations such as: scale variation our algorithm, performs favorably against the reference trackers even without implementing scale variation estimation, this is justified by a problem found in the benchmark itself, because if we implement a mechanism that can handle size variation, the performance of the tracking will be degraded.

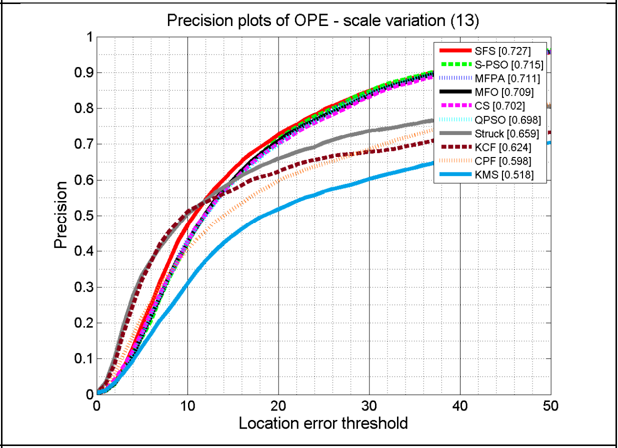
In fact, this issue can be explained by the frequent occlusions, if the object undergo partial occlusion, the tracking algorithm will change the size according to the visible part of the object, but the object in reality did not change in size, thus the performance of any tracking algorithm that implement size variation handling will degrade according to the results displayed; and this is why we have not implemented a size handling mechanism in our tracking approach.

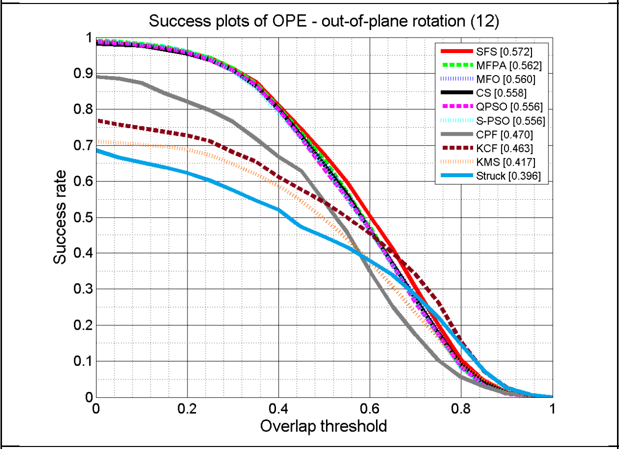
To remedy to this problem the researcher must choose only sequences with size variations and do not include occlusions, to compare his tracking results.

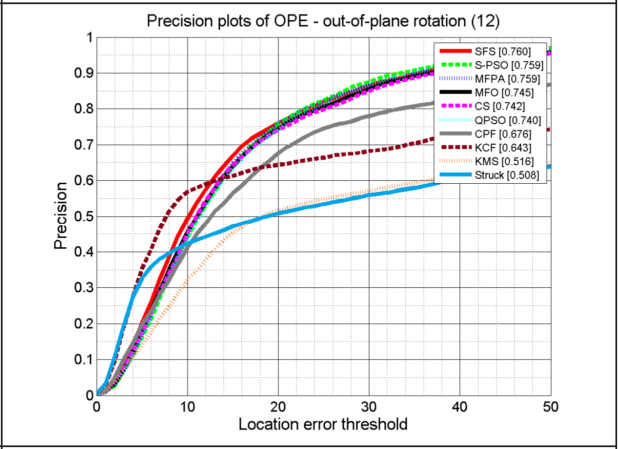

Success plots of OPE - occlusion (13)
SFS [0.568]
MFPA [0.561]
MFO [0.558]
CS [0.557]
QPSO [0.556]
S-PSO [0.554]
CPF [0.469]
KCF [0.466]
KMS [0.425]
SCM [0.422]
Success rate
Overlap threshold

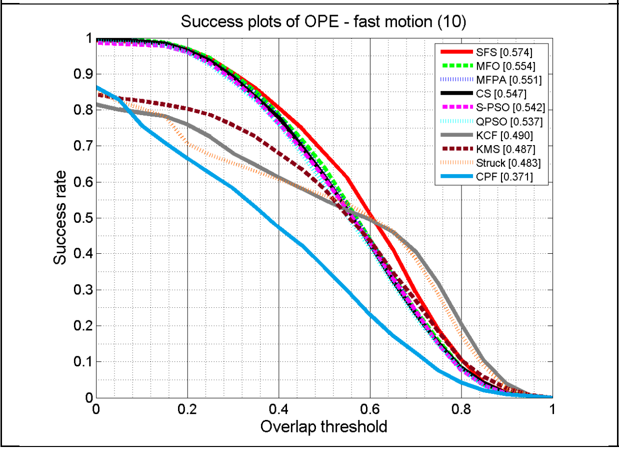

Precision plots of OPE - occlusion (13)
SFS [0.733]
MFPA [0.732]
S-PSO [0.727]
MFO [0.721]
CS [0.716]
QPSO [0.713]
KCF [0.650]
CPF [0.649]
SCM [0.575]
KMS [0.517]
Precision
Location error threshold

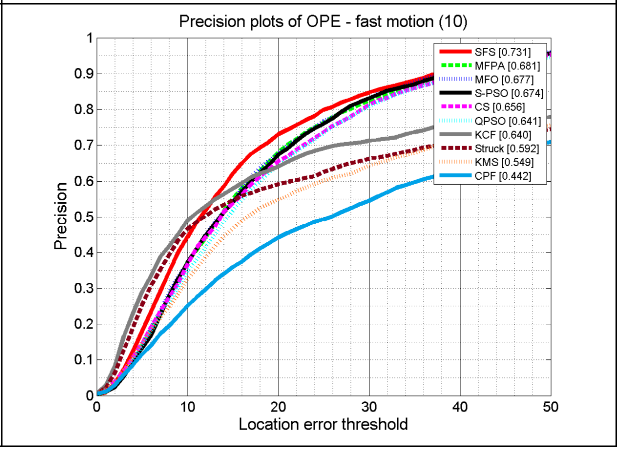

Success plots of OPE - scale variation (13)
SFS [0.545]
MFPA [0.535]
MFO [0.534]
CS [0.534]
QPSO [0.530]
S-PSO [0.529]
Struck [0.491]
KMS [0.442]
CPF [0.429]
KCF [0.428]
Success rate
Overlap threshold

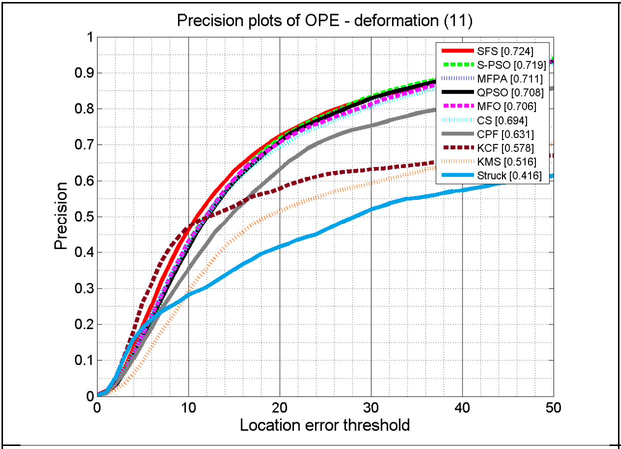

Precision plots of OPE - scale variation (13)
SFS [0.727]
S-PSO [0.715]
MFPA [0.711]
MFO [0.709]
CS [0.702]
QPSO [0.698]
Struck [0.659]
KCF [0.624]
CPF [0.598]
KMS [0.518]
Precision
Location error threshold

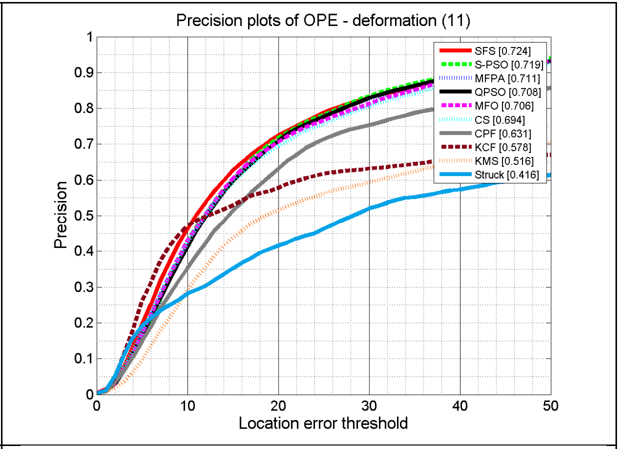

Success plots of OPE - out-of-plane rotation (12)
SFS [0.572]
MFPA [0.562]
MFO [0.560]
CS [0.558]
QPSO [0.556]
S-PSO [0.556]
CPF [0.470]
KCF [0.463]
KMS [0.417]
Struck [0.396]
Success rate
Overlap threshold

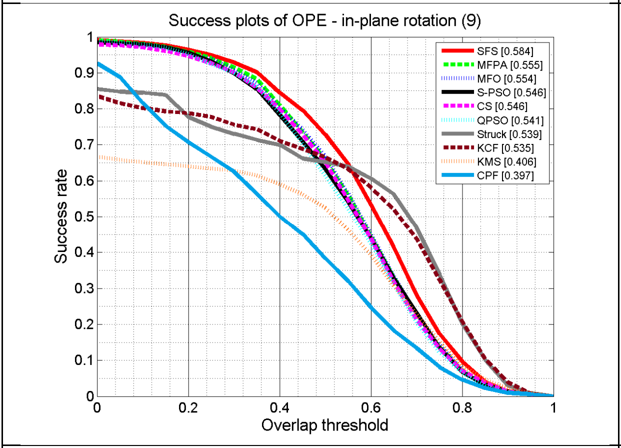

Precision plots of OPE - out-of-plane rotation (12)
SFS [0.760]
S-PSO [0.759]
MFPA [0.759]
MFO [0.745]
CS [0.742]
QPSO [0.740]
CPF [0.676]
KCF [0.643]
KMS [0.516]
Struck [0.508]
Precision
Location error threshold

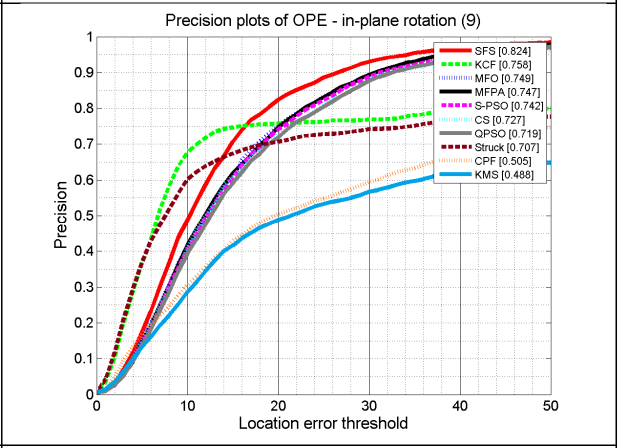

Success plots of OPE - fast motion (10)
SFS [0.574]
MFO [0.554]
MFPA [0.551]
CS [0.547]
S-PSO [0.542]
QPSO [0.537]
KCF [0.490]
KMS [0.487]
Struck [0.483]
CPF [0.371]
Success rate
Overlap threshold

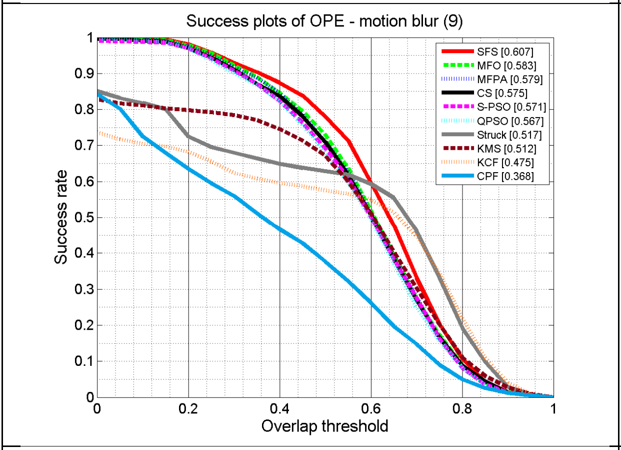

Precision plots of OPE - fast motion (10)
SFS [0.731]
MFPA [0.681]
MFO [0.677]
S-PSO [0.674]
CS [0.656]
QPSO [0.641]
KCF [0.640]
Struck [0.592]
KMS [0.549]
CPF [0.442]
Precision
Location error threshold

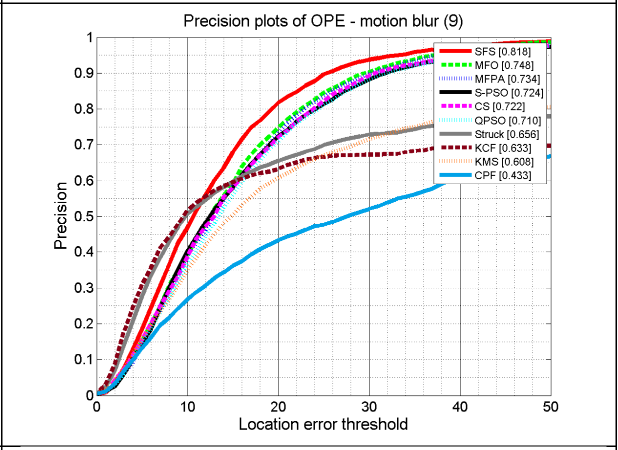

Precision plots of OPE - deformation (11)
SFS [0.724]
S-PSO [0.719]
MFPA [0.711]
QPSO [0.708]
MFO [0.706]
CS [0.694]
CPF [0.631]
KCF [0.578]
KMS [0.516]
Struck [0.416]
Precision
Location error threshold

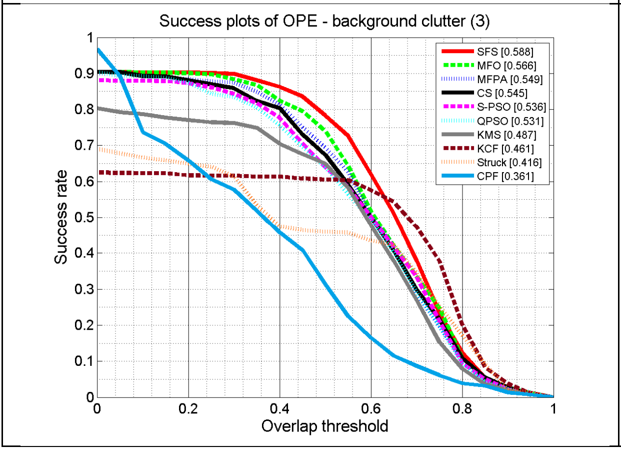

Precision plots of OPE - deformation (11)
SFS [0.724]
S-PSO [0.719]
MFPA [0.711]
QPSO [0.708]
MFO [0.706]
CS [0.694]
CPF [0.631]
KCF [0.578]
KMS [0.516]
Struck [0.416]
Precision
Location error threshold

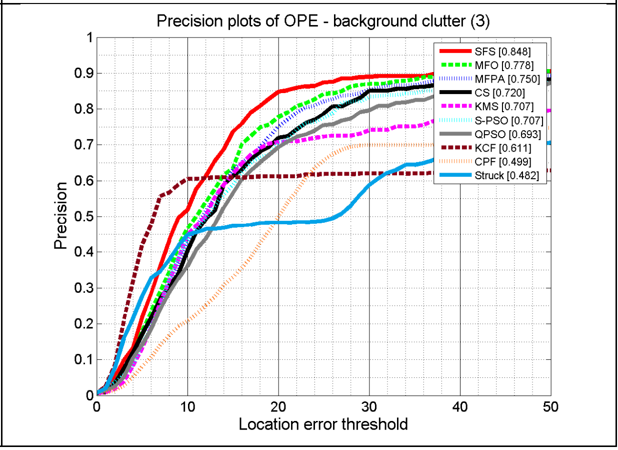

Success plots of OPE - in-plane rotation (9)
SFS [0.584]
MFPA [0.555]
MFO [0.554]
S-PSO [0.546]
CS [0.546]
QPSO [0.541]
Struck [0.539]
KCF [0.535]
KMS [0.406]
CPF [0.397]
Success rate
Overlap threshold


Precision plots of OPE - in-plane rotation (9)
SFS [0.824]
KCF [0.758]
MFO [0.749]
MFPA [0.747]
S-PSO [0.742]
CS [0.727]
QPSO [0.719]
Struck [0.707]
CPF [0.505]
KMS [0.488]
Precision
Location error threshold


Success plots of OPE - motion blur (9)
SFS [0.607]
MFO [0.583]
MFPA [0.579]
CS [0.575]
S-PSO [0.571]
QPSO [0.567]
Struck [0.517]
KMS [0.512]
KCF [0.475]
CPF [0.368]
Success rate
Overlap threshold


Precision plots of OPE - motion blur (9)
SFS [0.818]
MFO [0.748]
MFPA [0.734]
S-PSO [0.724]
CS [0.722]
QPSO [0.710]
Struck [0.656]
KCF [0.633]
KMS [0.608]
CPF [0.433]
Precision
Location error threshold


Success plots of OPE - background clutter (3)
SFS [0.588]
MFO [0.566]
MFPA [0.549]
CS [0.545]
S-PSO [0.536]
QPSO [0.531]
KMS [0.487]
KCF [0.461]
Struck [0.416]
CPF [0.361]
Success rate
Overlap threshold


Precision plots of OPE - background clutter (3)
SFS [0.848]
MFO [0.778]
MFPA [0.750]
CS [0.720]
KMS [0.707]
S-PSO [0.707]
QPSO [0.693]
KCF [0.611]
CPF [0.499]
Struck [0.482]
Precision
Location error threshold

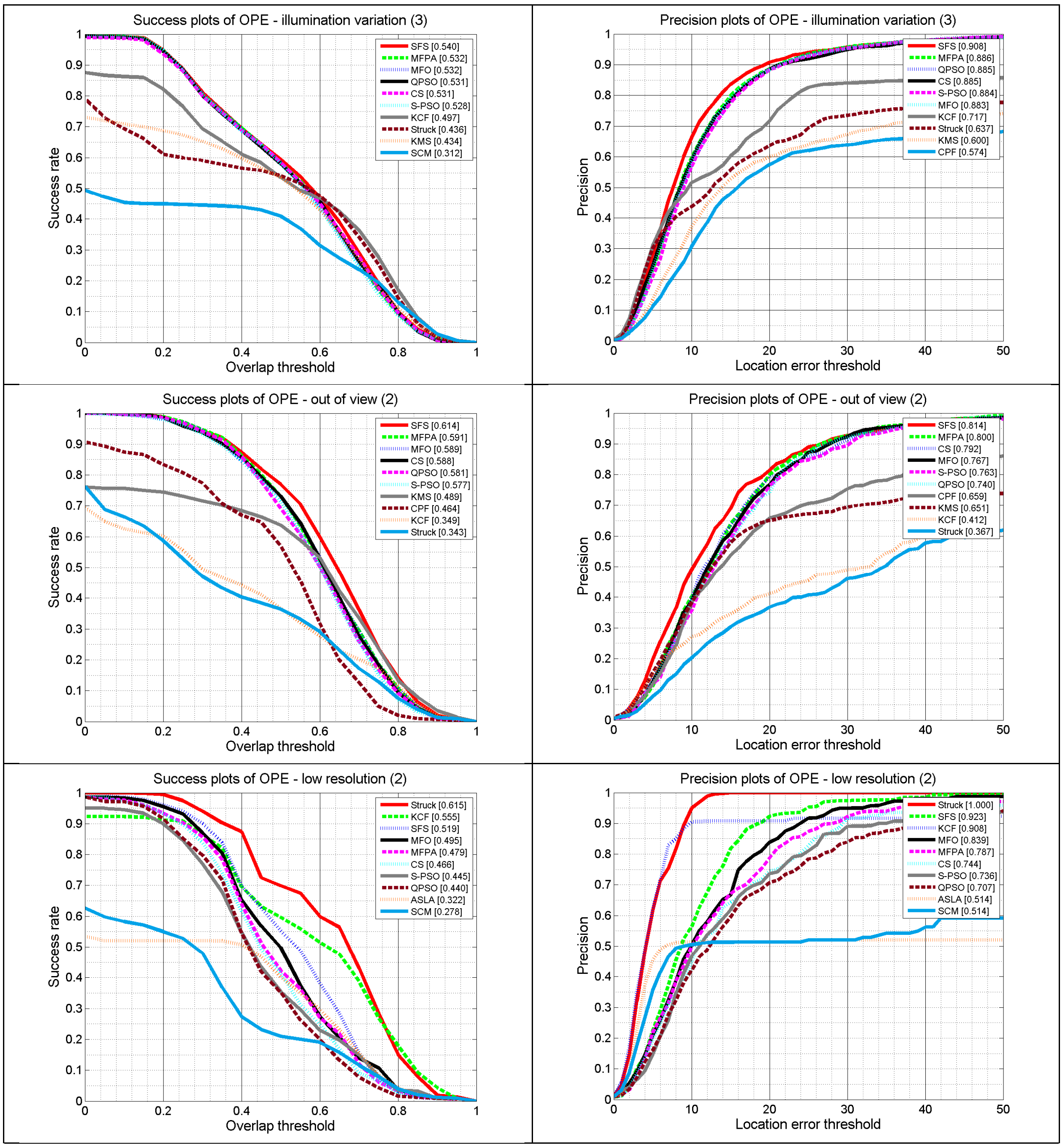


Fig. 5.11 Distance precision plots over 11 tracking challenges.

The legend contains the scores at a threshold of 20 pixels for each tracker. and an overlap success curve which contains area under- the-curve score for each tracker thresholded at 0.5.

The legend of distance precision contains threshold scores at 20 pixels, while the legend of overlap success contains area under-curve score at 0.5 for each tracker.

Our proposed algorithm performs favorably against the 11 reference trackers in 10 challenging attributes.

### 5.8.4.2. Qualitative comparison

In order to perform qualitative evaluation, Fig.5.12 shows of the tracking results of the top four performing trackers such as: SFS, MFO, MFPA, and CS. The tracking snapshots are taken using 7 difficult video sequences taken as an example: (**a**) Boy, (**b**) DragonBaby, (**c**) Blurface, (**d**) BlurBody, **(e)** Jogging-2, (**f**) Bolt2, **(g)** Lemming.

The table.5.3 illustrates the challenging attributes that characterize these sequences.

The object of interest in these sequences are of different types, including three human face sequences, Fig.5.12.(a,b,c), three human body sequences: Fig.5.12 (d, e, f), and a doll sequence in Fig.5.12.(g).

In Fig.5.12(a, b, c), the sequences Boy, DragonBaby and BlurFace were under SV, MB, FM, IPR, OPR, OCC, and OV.

In Fig.5.12 (a) the target object is the face of a Boy in a indoor environment, SFS tracker provided the best tracking performance. In seq. #80, the boy's face undergoes fast motion, SFS and MFO were the only trackers that successfully could follow the object, in contrast CS and MFPA slightly drifted. In the three sequences #268, #362, and #408, the challenge here is a combined attributes such as rotation, fast motion, and blurred motion all the other trackers slightly drifted, except SFS.

In Fig.5.12 (b) the object of interest is the face of baby. In seq. #13, only MFPA lost the target due to the target fast motion, and the rest of the trackers could successfully follow the target. In seq. #41, the challenge here is a fast and blurred motion together, all the trackers successfully got the right position, except MFO failed to follow the object. In seq. #54, the target was under out of plane rotation and abrupt motion, only MFPA missed the target. At last, in seq. #100 despite the combined challenges such as: out of plane rotation, fast motion, and partial occlusion, SFS was the only tracker that was able to follow the target.

In Fig.5.12 (c), SFS tracker was the only tracker who was able to follow the target, the challenges here are: in plane rotation, fast motion, and blurred motion.

In Fig.5.12 (d): The BlurBody sequence. In seq. #16, #170, and #331, Only SFS tracker could follow the target under continuous fast blurred motion, and deformation, unlike all the other trackers MFPA, MFO and CS who have lost target. In seq. #148, SFS and CS were able to follow the target, In contrast MFPA and MFO lost the target because of the fast and blurred motion.

In Fig.5.12 (e) The Jogging-2 sequence, the main challenge in seq. #112, #161, #246, and #248, is a continuous deformation, and we can notice that only SFS has the capacity to follow the target with precision despite this challenge, the other trackers have missed slightly without getting the exact position, this is due to the difference in exploration and exploitation.

In Fig.5.12 (f), Bolt2 sequence undergone slight background clutter in the first frames, and deformation, in #72, CS already lost the target, while in #177, SFS got the right position, while the three other trackers drifted.

In # 192, MFPA lost the target due to a similar object near the target. As of frame # 213, all trackers including SFS have lost the target due to the huge color similarity between runners in the last frames. This is due to the disadvantage of the color function in the background clutter.

In the Lemming sequence Fig.5.12 (g), the task here is the most difficult due to the frequent appearance changes in size with out-of-plane rotation, abrupt motion and occlusion, in very long video, even though SFS was the best, we can see a slight difference in tracking performance between the others. In seq. # 227 MFPA, and MFO lost part of the object due to the blurred motion and out-of-plane rotation. In seq. # 554, only SFS could follow the target accurately, the others lost half of the object in out-of-plane rotation and partial occlusion.

Finally, in the last seq. # 1046, all trackers were able to find the precise position because the object underwent a slight change in size with rotation from how it looked in the first frame.

Table 5.3 Description of the challenges that characterize the sequences

| Sequence | sequence length | Challenging Attribute |
|---|---|---|
| Boy | 602 | Scale Variation, Motion Blur, Fast Motion, In plane rotation, Out of Plane Rotation. |
| DragonBaby | 113 | Scale Variation, Motion Blur, Fast Motion, In plane rotation, Out of Plane Rotation, Occlusion, Out of View. |
| Blurface | 493 | Motion Blur, Fast Motion, In plane rotation. |
| BlurBody | 334 | Scale Variation, Motion Blur, Fast Motion, In plane rotation, Deformation. |
| Jogging-2 | 307 | Occlusion, Deformation, Out of Plane Rotation. |
| Bolt2 | 293 | Deformation, Background Clutter |
| Lemming | 1336 | Scale Variation, Occlusion, Fast Motion, Out of Plane Rotation, Out of View. |

### 5.8.5. Justification of the chosen sequences

As the proposed tracking algorithm used color histogram as a feature, its main disadvantage situations including: background clutter, illumination variation, and low resolution.

Table.5.4 display the 20 video sequences used and their challenging attributes.

Table.5.5 display the number of occurrence of every attribute for the 20 chosen video sequence.

We can observe that the chosen sequences have the least unfavorable situations such as: background clutter, illumination variation, low resolution, and out of view.

The most favorable situations for our tracking algorithm are sorted as follows: partial and full occlusion, scale variation, out of plane rotation, fast motion, deformation, in plane rotation, and blurred motion.

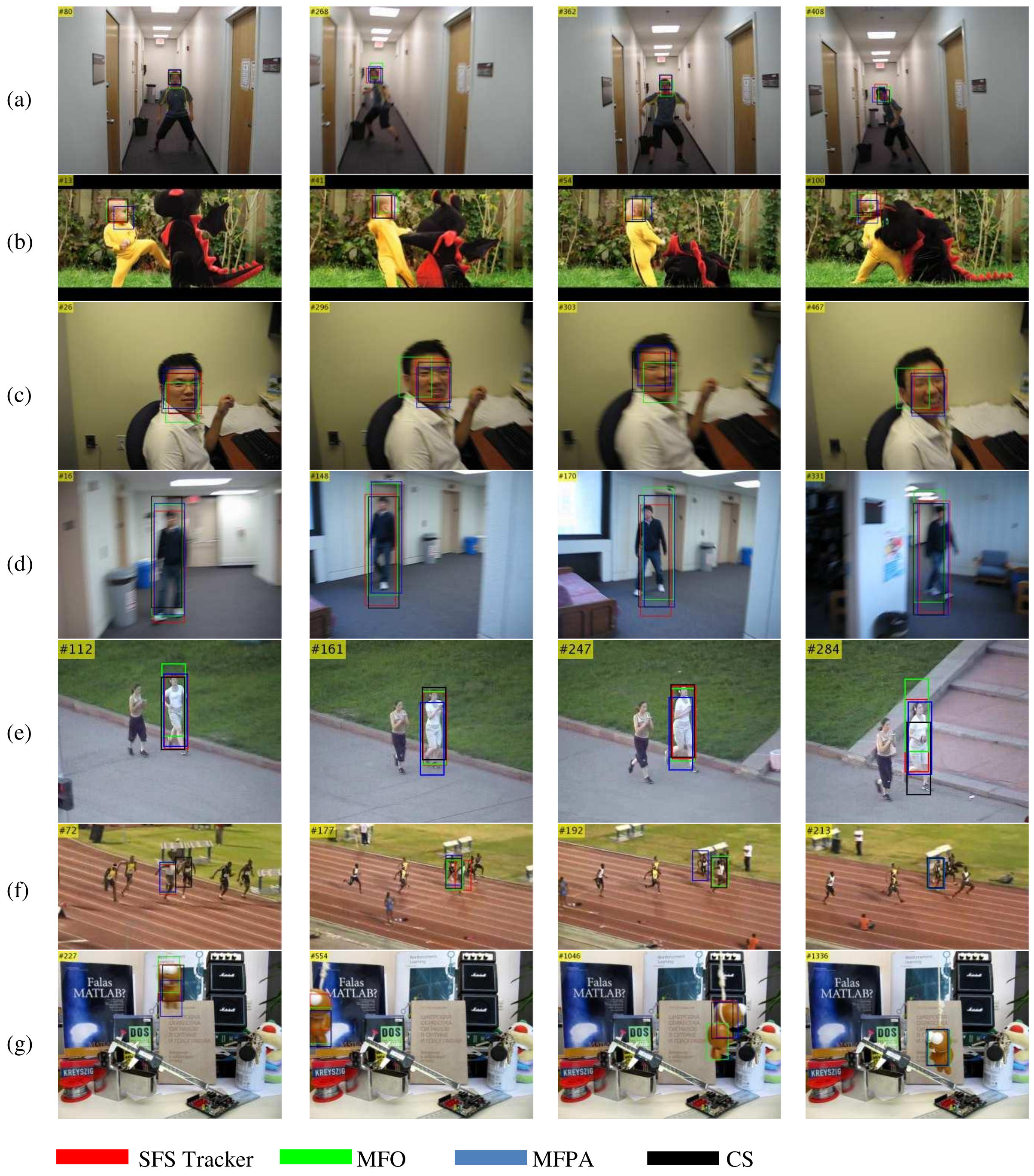


Fig. 5.12 Tracking results of four metaheuristic algorithms.

Frame numbers are shown in the top left of each figure. Sequences are: **(a)** Boy, **(b)** DragonBaby, **(c)** Blurface, **(d)** BlurBody, **(e)** Jogging-2, **(f)** Bolt2, **(g)** Lemming.

Table 5.4 The 20 video sequences used in the experiment

| Sequence | Challenges |
|---|---|
| Boy | SV, MB, FM, IPR, OPR |
| Skating2.1 | SV, OCC, DEF, FM, OPR |
| Skating2.2 | SV, OCC, DEF, FM, OPR |
| Girl2 | SV, OCC, DEF, MB, OPR |
| BlurFace | MB, FM, IPR |
| Jogging-2 | OCC, DEF, OPR |
| DragonBaby | SV, OCC, MB, FM, IPR, OPR, OV |
| Lemming | IV, SV, OCC, FM, OPR, OV |
| Bolt2 | DEF, BC |
| BlurBody | SV, DEF, MB, FM, IPR |
| Deer | MB, FM, IPR, BC, LR |
| Bolt | OCC, DEF, IPR, OPR |
| Girl | SV, OCC, IPR, OPR |
| David3 | OCC, DEF, OPR, BC |
| Woman | IV, SV, OCC, DEF, MB, FM, OPR |
| FaceOcc1 | OCC |
| BlurOwl | SV, MB, FM, IPR |
| Walking | SV, OCC, DEF |
| RedTeam | SV, OCC, IPR, OPR, LR |
| Human9 | IV, SV, DEF, MB, FM |

Table 5.5 The 11 challenging attributes and their number of occurrences

| Attribute acronym | Attribute full name | Number of occurrences |
|---|---|---|
| OCC | occlusion | 13 |
| SV | Scale variation | 13 |
| OPR | Out of plane rotation | 12 |
| FM | Fast motion | 11 |
| DEF | Deformation | 11 |
| IPR | In plane rotation | 09 |
| MB | Motion blur | 09 |
| BC | Background clutter | 03 |
| IV | Illumination variation | 03 |
| LR | Low resolution | 02 |
| OV | Out of View | 02 |

## 5.9. HHO experimental evaluation and tracking results

Harris hawks optimizer (HHO) was found to provide an efficient exploration of the search space, with a higher accuracy compared to other metaheuristic techniques. In this work we propose to apply it for the first time as a searching mechanism, in SOT framework.

To implement HHO algorithm for visual tracking, the same assumptions as SFS tracking algorithm were made, the same presentation, and the same platform, the using 20 video sequences from the same OTB2015 dataset, and following the same protocol.

The proposed HHO tracking algorithm performances are given with comparisons to 11 reference trackers. According to their searching mechanisms, they can be categorized into three classes:

1. **Deterministic methods:** Including: the traditional mean shift tracking KMS [5], and Fragments-based tracker (Frag) [155], structured output tracking with kernels (Struck) [68].
2. **Probabilistic methods:** Including color-based particle filter (CPF) [154], locally orderless tracking (LOT) [156], and multi-task sparse learning (MTT)[157].
3. **Metaheuristic methods:** Including our HHO, BAT algorithm (BA) [8], particle swarm optimization (PSO) [152], Cuckoo Search (CS) [151], simplified swarm optimization (SSO) [48], and modified flower pollination algorithm (MFPA) [11].

Concerning metaheuristic based tracking techniques, the table 5.6 resume their tuning parameters, these parameters are used as their authors recommended except population size, which is set as N=30, for a fair comparison.

The reported quantitative evaluation are shown in Fig.5.13, and Fig.5.14

Table 5.6 The parameter setting of every algorithm used during simulation

| Algorithm | Parameter | Value |
|---|---|---|
| **HHO** | N: Population size | 30 |
| **CS** | N: Population size | 30 |
| **MFPA** | N: Population size | 30 |
| **PSO** | N: Population size | 30 |
| | W | |
| | C1 | |
| | C2 | |
| **SSO** | N: Population size | 30 |
| | Cg | 0.8 |
| | Cp | 0.35 |
| | Cw | 0.1 |
| **BA** | N: Population size | 20 |
| | Fmin | -20 |
| | Fmax | 20 |
| | α: constant for loudness update | 0.8 |
| | γ: constant for emission rate update | 0.8 |
| | $r_o$: initial pulse emission rate | 0.001 |

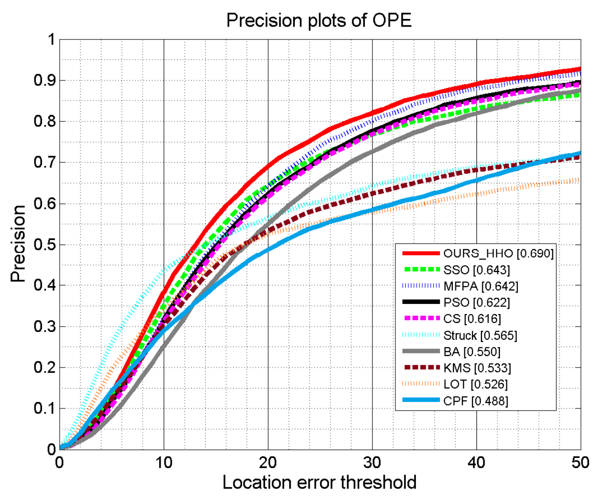


Fig. 5.13 The results under OPE metric, using location error threshold.

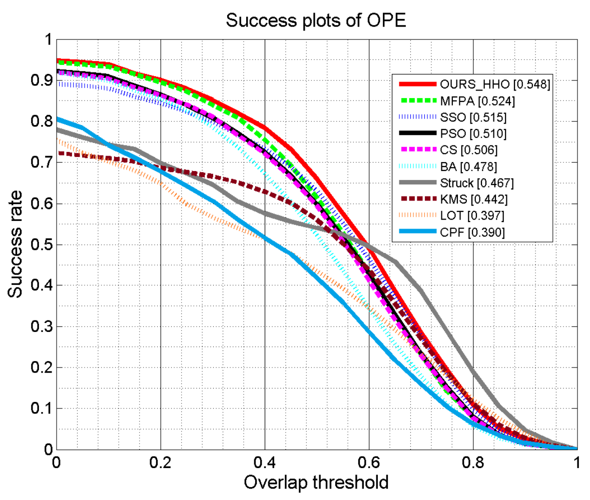


Fig. 5.14 The results under OPE metric, using Overlap threshold.

As shown in Fig.5.13 and Fig.5.14, HHO based tracker performs favorably against the 11 reference trackers in both distance precision with (69%) and overlap success with (54.8%).

MFPA and SSO tracking performance can be considered the same, as they rank the second and third best results with (64.2%) distance precision rate for MFPA, and (64.3%) distance precision rate for SSO, and (52.4%) overlap success rate for MFPA, and(51.5%) overlap success rate for SSO.

PSO rank the fourth among all the reference trackers with an overlap success rate (51 %) and distance precision rate (62.2%).

From both Fig.5.13 and Fig.5.14, the six compared metaheuristic based tracking methods perform favorably against the three deterministic, and the three probabilistic methods.

## 5.10. Discriminative framework of single object tracking using WOA

In this work WOA is implemented in SOT, and the parameters such as: population size and iteration numbers are studied experimentally and statistically to be adjusted in visual tracking, WOA is used because of its high accuracy and lower computation complexity which differs it from other metaheuristic techniques.

The problematic to be solved in this framework is little background clutter, and the presence of salient background information inside the first initiated object to be tracked, which often effect the tracking accuracy, because the traditional generative presentation KCWH do not take in consideration background information.

To solve such problem WOA algorithm is combined with a high discriminative presentation called corrected background weighted histogram (CBWH). The presentation corrects the background of the object template, to improve the tracking accuracy in difficult environment.

### 5.10.1. WOA implemented for visual tracking

In this section, we show how to implement the WOA into SOT framework.

The same manner as SFS and HHO trackers are implemented, WOA is implemented with one exception.

The corrected background weighted histogram (CBWH) of the object template presentation is carried in the first frame which will be explained in the next section. CBWH is calculated in the HSV space, then stored as the reference target model for comparison.

The object candidates are presented using the traditional kernel color weighted histogram (KCWH) fully explained in the third previous section.

The same assumptions were made, such as that the object's speed cannot exceed its initial size ($W_i$,$H_i$), which drive us to the most probable area of search is defined by the lower bound $P_{min}$ and upper bound $P_{max}$ in the following formula:

- $P_{max}=P_{best}+Min(W_i,H_i)$ (5.9)
- $P_{min}=P_{best}-Min(W_i,H_i)$

Where $P_{best}$ is the best position found so far and updated at the end of every iteration.

During tracking process, WOA tracker is called, and the KCWH of candidates are generated, then compared to the object model histogram using the Bhattacharyya distance as fitness function in which optimization is sought.

The flow chart of the tracking algorithm is displayed as follows:

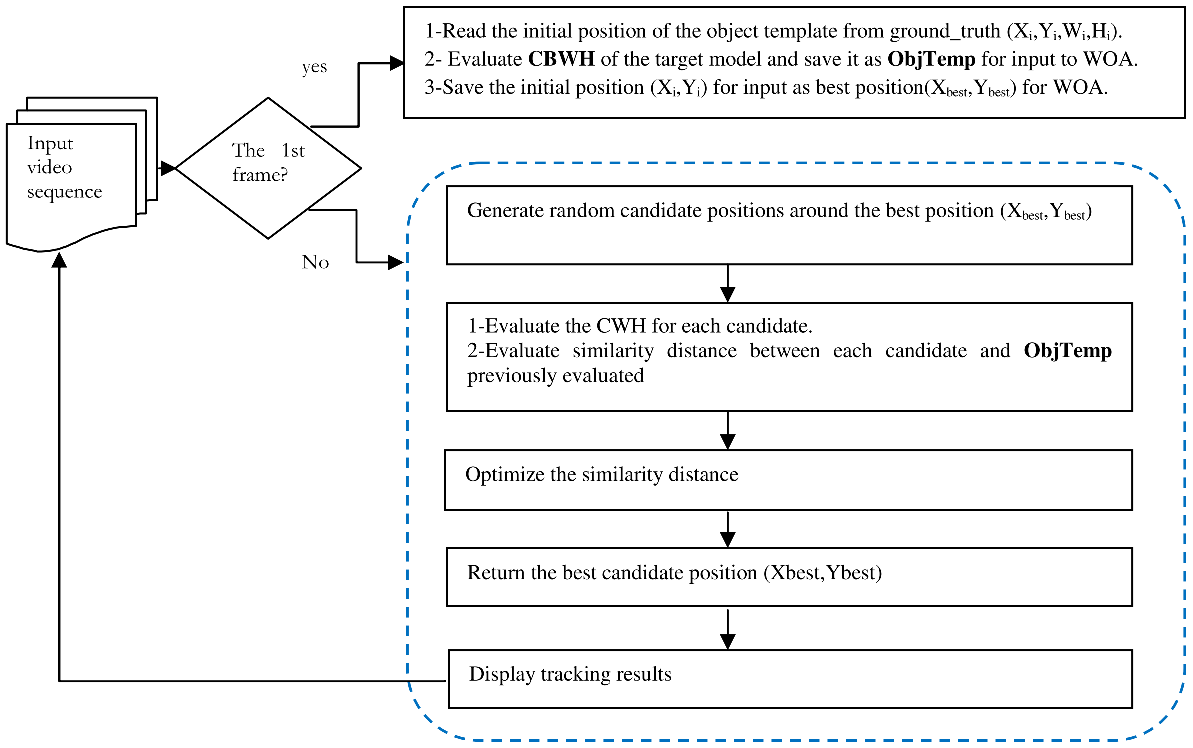

Input video sequence
The 1st frame?
yes
No
1-Read the initial position of the object template from ground_truth ($X_i$,$Y_i$,$W_i$,$H_i$).
2- Evaluate **CBWH** of the target model and save it as **ObjTemp** for input to WOA.
3-Save the initial position ($X_i$,$Y_i$) for input as best position($X_{best}$,$Y_{best}$) for WOA.
Generate random candidate positions around the best position ($X_{best}$,$Y_{best}$)
1-Evaluate the CWH for each candidate.
2-Evaluate similarity distance between each candidate and **ObjTemp** previously evaluated
Optimize the similarity distance
Return the best candidate position (Xbest,Ybest)
Display tracking results

Fig. 5.15 Proposed WOA Tracking algorithm.

### 5.10.2. WOA-CBWH Tuning parameters

In any metaheuristic algorithm, there are two challenging factors that measure the algorithm computation complexity. The first one is the number of populations, and second, is the number of iterations. Those two must be tuned to find the algorithm best performance defined by its accuracy to find the best solution.

To carry on the study of WOA-CBWH performances, we have statistically tuned the two parameters: population size, and iterations number, with the aim to find the lowest number of fitness computations.

Our study was given using Boy sequence in Fig.5.16 from the benchmark OTB2015. This sequence have the higher number of challenging attributes such as: fast motion, blurred motion, scale variation, in plane rotation and out of plane rotation. The red bounding box shows the background, where the blue one indicate the target model.

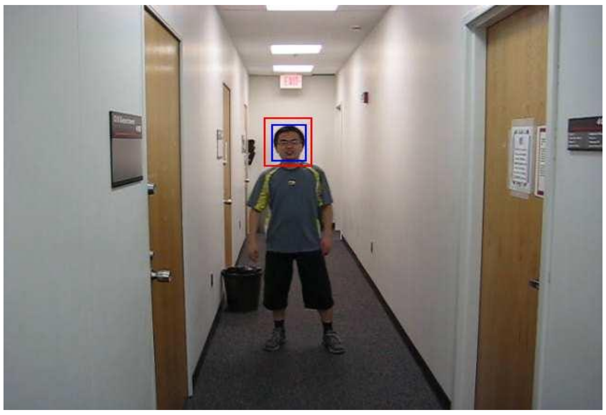


Fig. 5.16 Boy sequence used for adjusting WOA parameters.

WOA-CBWH tracker was run 10 times, and its average performances are evaluated by varying both population size from 6 to 24 with a step of 2, and iterations number from 1 to 6.

The performance used was the average overlap, which can be defined in a frame I by:

- $$Overlap = \frac{Area(B_I^p \cap B_I^g)}{Area(B_I^p \cup B_I^p)} \tag{5.10}$$

Where: $B_I^g$ is the ground-truth bounding box, and $B_I^p$ is the predicted bounding box.

### Discussion of the obtained results

Fig.5.17 and Fig.5.18 display the performance evaluation of six different iterations with 10 variable population sizes.

In Fig.5.17, it can be observed that except the case of single iteration, the tracking performance is enhanced as iteration number and population size increase.

WOA algorithm computation complexity is related to how many times objective function is calculated. In the initialization process, the computational complexity of N Whale agents is O(N), while in the updating process it is O(N*Iter_Max), where Iter_Max is the maximum number of iterations.

Therefore, the computation complexity of WOA is O(N*(IT+ 1)).

Fig.5.18 displays the results under One-way ANOVA analysis, to test statistically WOA-CBWH performance with different iterations and population size. As we are searching for the lowest computation complexity, with the smallest standard deviation value (Std), and an acceptable mean value of objective function. The graphs on Fig.5.17 and Fig.5.18 can depict that, at N=12, and Iter_Max=2, WOA-CBWH algorithm have statistically no important improvement in the best objective value, where the mean value equals to 0.6612, and Std=$3.10^{-3}$, and a total number of objective function evaluations equals to 12x(2+1)=36, which is acceptable.

But when the computation complexity is not considered, the algorithm can achieve an average tracking performance up to 0.7164 (with N=24, Iter_Max=6), but with a total number of fitness evaluations equals to 168, which is 4.7 times the number of evaluations in the first case (N=12, Iter_Max=2), and the gain in this case is only 5% better.

In the next experiment we have chosen **N=12**, and **Iter_Max=2**, as a result of this study.

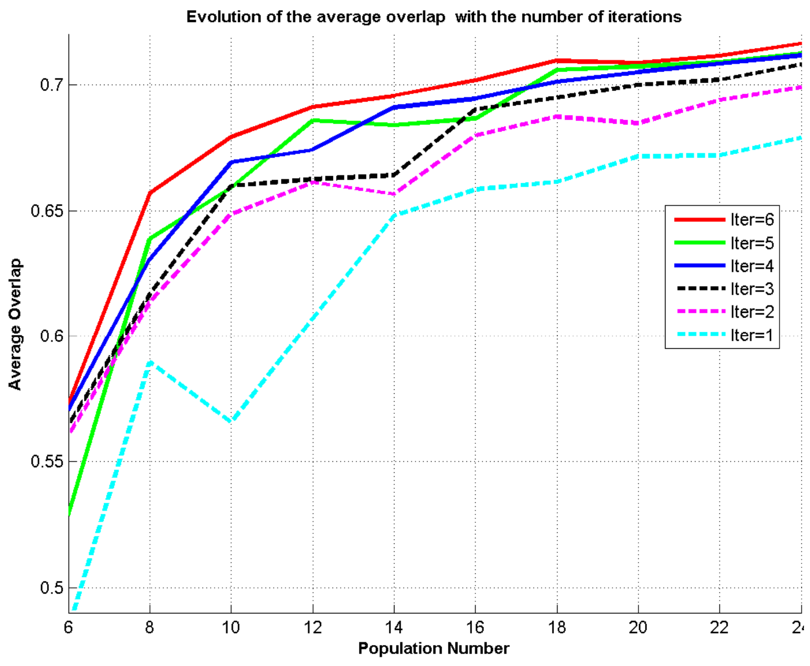
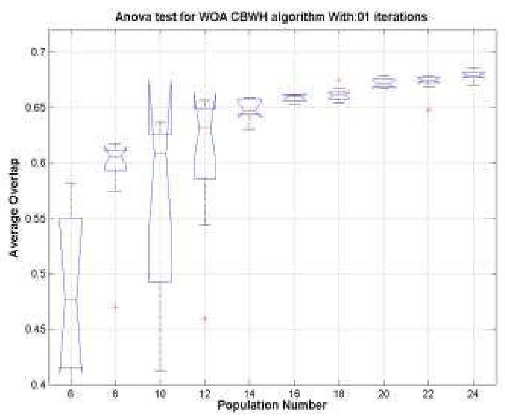
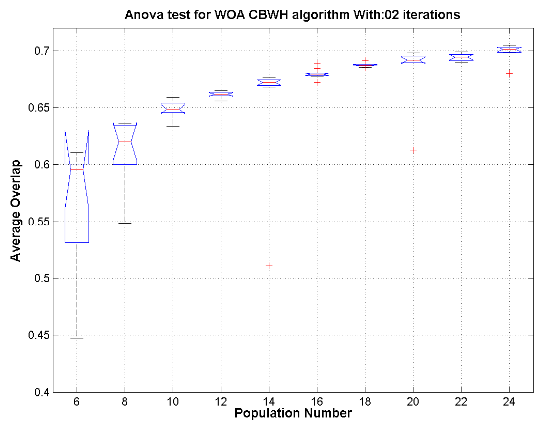


Fig. 5.17 Tracking performance under different iterations and population size.

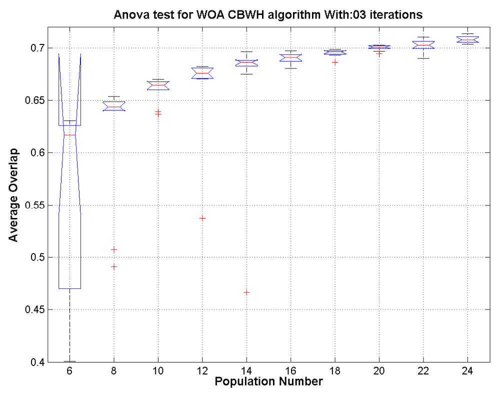
(a)



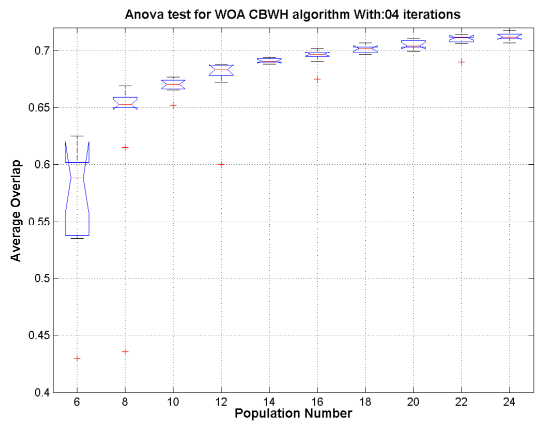
(b)



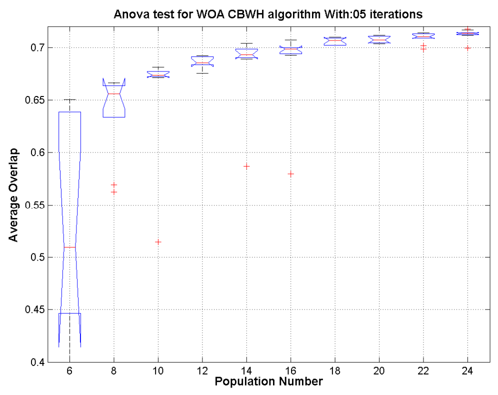
(c)



(d)



(e)



(f)

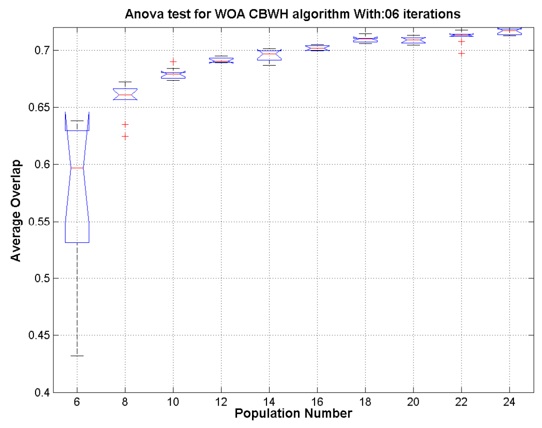


Fig. 5.18 Anova test of 10 runs, each figure depicts a number of iterations.

### 5.10.3. Object Template presentation based on corrected background weighted histogram

Background information is often inside the initiated template region and often affects the tracking accuracy. To decrease its interference, comaniciu et al, proposed a presentation named as background weighted histogram (BWH), to decrease the influence of background on target localization and improve object localization. Thereafter, Corrected BWH is the name given by Jifeng [159], to the same presentation initially created by comaniciu et al, by proving that BWH should only used in the object model and not the object candidates, to get better tracking results.

Comaniciu et al. [5] proposed a formula by reducing background information in the first initiated template. They used the background that surrounds three times the target region of the ground truth. In our work we propose the same formula, but with a different window which is 1.4 times the minimum between (width and height) of the ground truth of the object in the first frame, and by modeling only the object model and not the object candidates.

To carry on the object model presentation (CBWH), we have followed the two following steps:

**The First step: Evaluation of the object kernel color weighted histogram (KCWH)**

This step is explained in the previous section 3, let's denote by $H_u$ this presentation.

**The Second step: Evaluation of the Corrected background weighted histogram (CBWH)**

We denote by $\{\widehat{BG}_U\}_{u=1\ldots m}$(with $\sum_{u=1}^{m}\widehat{BG}_U = 1$) the traditional normalized histogram of the background that surrounds the bounding box's (BB) of the object model manually initiated from the first frame. The BB of the object is marked with red color in Fig.5.19.

Let $BG^*$ the smallest nonzero value of $\widehat{BG}_U$, the weight $\left\{V_u = min\left(\frac{\widehat{BG}^*}{\widehat{BG}_u}, 1\right)\right\}_{u=1\ldots m}$ is evaluated and used to lower the background's features present in the traditional presentation KCWH. The new object model presentation denoted by $\widehat{H}_u$ is called as CBWH, and defined by:

- $\widehat{H}_u = C * V_u * H_u$ (5.11)

Where * is the wise multiplication; if we replace $H_u$ by its value, C is normalization constant expressed as:

- $C = \frac{1}{\sum_{i=1}^{n} k(\|x_i^*\|^2) \sum_{u=1}^{m} V_u \delta[b(x_i^*)-u]}$ (5.12)

For a better illustration of the equation (5.11), Fig.5.19 depicts an example on how to evaluate the center location distribution of color: Hu($U_H$=15,$U_S$=3,$U_V$=10), using one summation.

Through the equation 5.11, and the flow chart in Fig.5.19, we can see that the probabilities of the color that are part of the background are inversely proportional to the final presentation $\widehat{H}_u$.

Since both the target object and target candidates are histograms, Bhattacharyya distance (BD) is also used as similarity metric.

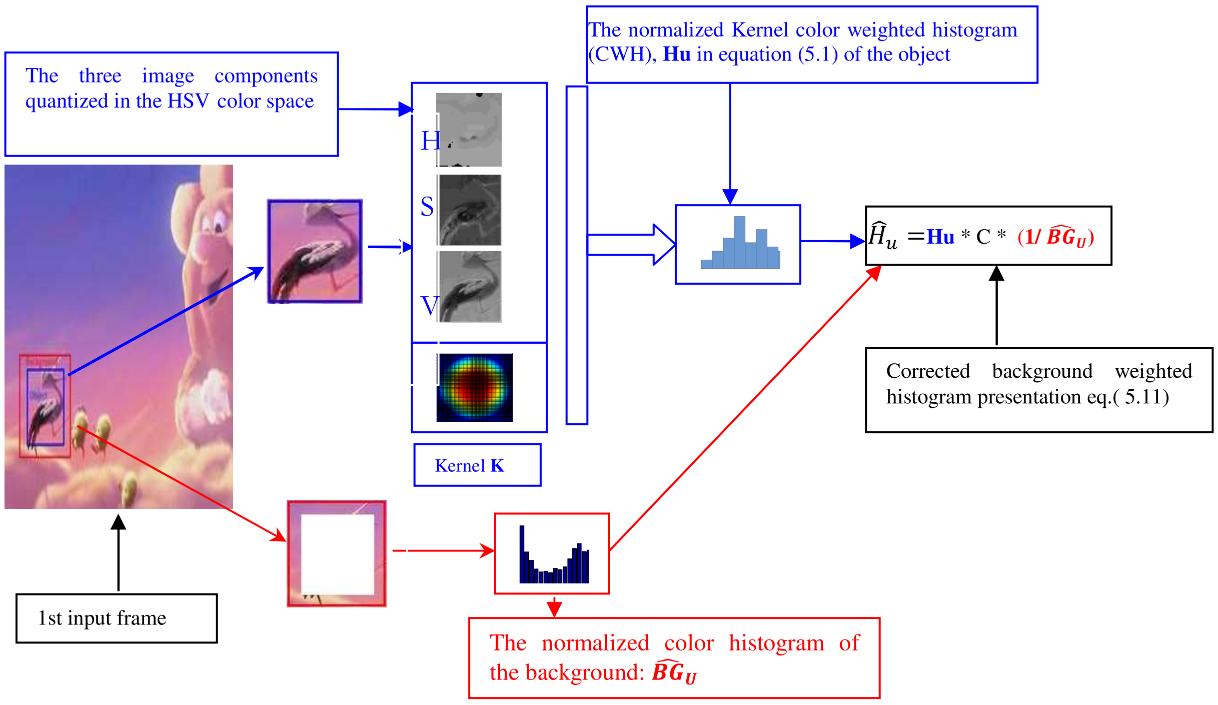

The three image components quantized in the HSV color space
H
S
V
The normalized Kernel color weighted histogram (CWH), Hu in equation (5.1) of the object
$\widehat{H}_u$ =Hu * C * (1/ $\widehat{BG}_U$)
Corrected background weighted histogram presentation eq.( 5.11)
Kernel K
1st input frame
The normalized color histogram of the background: $\widehat{BG}_U$

Fig. 5.19 Flowchart of the extraction of CBWH carried for the first input frame. Vu is replaced by its value, and BG* is assumed to be equal to 1 in this presentation[9]

### 5.10.4. Experimental evaluation

The efficiency of the proposed tracking approach is carried extensively using quantitative and qualitative comparisons, using 20 video sequences from the OTB15 dataset, and with other trackers following the protocol in [85].

Through quantitative comparison we have used two methods:

- The first one by using OPE metric on 20 video sequences.
- The second one by using average overlap and average Euclidean distance of 10 video sequences.

Through qualitative comparison:

- The first results represent snapshots of the tracking results over 8 video sequences
- The second results and for the same 8 video sequences the Euclidean distance plot per sequence is given.

The performance comparison is given with 6 reference trackers. According to their searching mechanisms, they can be categorized into three classes:

- **Probabilistic methods:** including, color-based probabilistic tracking (CPF) [155], and locally orderless tracking (LOT) [157].
- **Deterministic methods:** including, discriminative scale space tracker (DSST)[160], and Kernel mean-Shift with joint texture color (KMS_LPQ) [4].

- **Metaheuristic methods:** including WOA-CBWH, WOA-CWH, modified flower pollination algorithm (MFPA)[12], and BAT Algorithm (BA)[8].

The comparison of the proposed WOA-CBWH tracker is carried with other available tracker's results provided by their authors, and the available source codes of the metaheuristic trackers that have used traditional Kernel color histogram to model the object template.

Except WOA-CBWH, the other meta-trackers such as WOA-CWH, MFPA, and BA, the same target presentation (kernel-based spatial color histogram), and the same initialization, are used for a fair evaluation. The other sensitivity parameter of each algorithm is displayed in table 5.7.

Table 5.7 The sensitivity parameters of each algorithm

| Algorithm | Parameter | Value | Iterations |
|---|---|---|---|
| WOA-<br>CBWH | N: Population size<br>b: in Eq.7 | 12<br>1 | 2 |
| WOA-CWH | N: Population size<br>b: in Eq.7 | 12<br>1 | 2 |
| MFPA | N: Population size | 12, | 2 |
| BA | N: Population size<br>(Fmin, Fmax)<br>α: constant for loudness update<br>γ: constant for emission rate update<br>$r_o$: initial pulse emission rate | 12<br>(-20,20)<br>0.8<br>0.8<br>0.001 | 2 |

### 5.10.4.1. Quantitative comparison

#### a. The first quantitative comparison

Fig.5.20, and Fig.5.21, display the overall results under OPE plots, using the distance precision rate, and the overlap success rate.

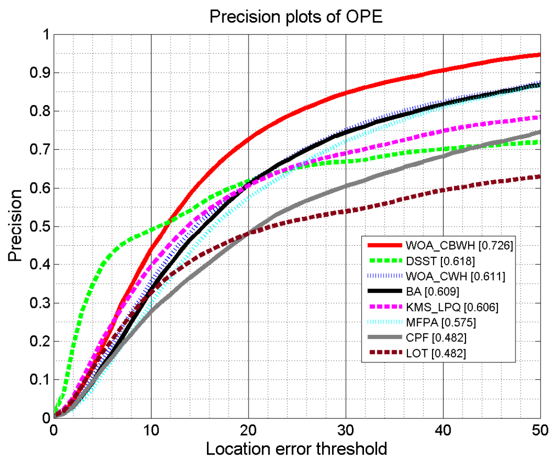


Fig. 5.20 The results under OPE metric, using location error threshold.

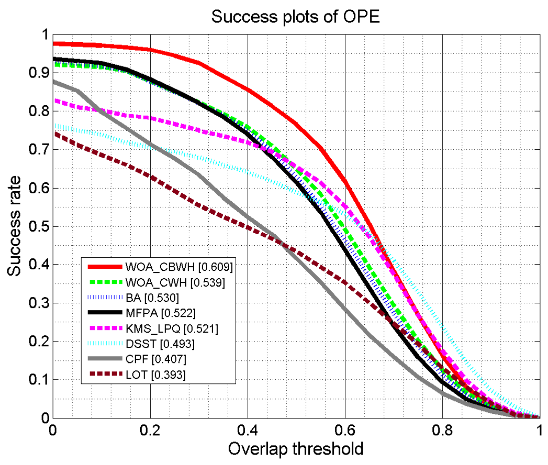


Fig. 5.21 The results under OPE metric, using Overlap threshold.

The legend of overlap success rate rank all the trackers at threshold of 0.5, while the legend of location error metric ranks the trackers at threshold equals to 20 pixels. It can be observed that

WOA-CBWH algorithm has better tracking performance against the 6 reference methods on both overlap score (60.9%) and distance precision (72.6%).

WOA-CBWH has the highest convergence rate compared to all the 7 mentioned trackers with a gain of 7.9% in overlap success rate, compared to BA tracker in Overlap metric, and 10.8% gain in distance precision rate compared to the second ranked DSST tracker in distance precision metric.

It is also remarkable in both Fig.5.20 and Fig.5.21, that the four metaheuristic trackers provides good performance compared to the based probabilistic approaches such as particle filter.

The WOA-CWH that used the traditional kernel color weighted histogram performed positively against the other two metaheuristic trackers, as it ranked the second in distance precision (61.1%) and the third in overlap success rate (53.9%) after DSST tracker. WOA-CWH tracker have roughly similar performance behavior compared to BA, with a gain of 0.2% in distance precision and of 0.9% in overlap success score. BA algorithm have a 60.9% score in distance precision and 53% overlap score.

DSST tracker achieved the second best result in distance precision with a score (61.8%), and ranked sixth in overlap with a score (49.3%).

### b. The second quantitative comparison

The performance of the proposed approach is depicted in table 5.8. The table summarize the average Euclidean distance between the true center position, and the predicted position by the tracker, the same manner for the average overlap, using 10 video sequences,

Table 5.8 The average Euclidean distance and average overlap obtained over 10 video sequences.

| Sequence | WOA-CBWH | WOA-CWH | BA | MFPA | KMS_LPQ | CPF | LOT | DSST |
|---|---|---|---|---|---|---|---|---|
| Lemming | 0.63/15.77 | ***0.65***/13.39 | 0.62/14.64 | 0.59/16.33 | **0.67**/***12.25*** | 0.56/**11.71** | 0.59/19.12 | 0.37/81.55 |
| Bird2 | **0.73/9.08** | 0.39/30.31 | 0.37/31.82 | 0.40/30.01 | 0.07/133.85 | 0.40/31.50 | 0.08/108.69 | ***0.59/19.21*** |
| BlurOwl | **0.70/12.06** | ***0.57***/23.04 | 0.56/***20.63*** | 0.50/27.87 | 0.48/30.09 | 0.25/61.03 | 0.21/81.91 | 0.20/131.39 |
| DragonBaby | **0.59/13.15** | **0.59**/***13.37*** | *0.56*/15.48 | 0.44/35.12 | ***0.56***/35.51 | 0.37/43.63 | 0.53/26.82 | 0.05/150.42 |
| Board | **0.72/18.40** | 0.64/44.71 | 0.63/46.79 | 0.63/33.93 | ***0.70/21.20*** | 0.54/49.63 | 0.21/182.39 | 0.68/30.99 |
| Girl2 | **0.67**/***13.37*** | ***0.65***/17.12 | 0.64/17.13 | 0.64/17.53 | 0.61/36.03 | 0.64/**8.41** | 0.58/34.15 | 0.50/46.54 |
| Liquor | 0.67/42.43 | 0.67/43.39 | 0.67/41.92 | ***0.73/19.20*** | 0.62/43.34 | 0.47/72.30 | **0.83/8.53** | 0.44/82.68 |
| Trellis | ***0.49***/20.21 | 0.40/30.37 | 0.39/30.67 | 0.39/30.32 | 0.39/29.50 | 0.24/28.50 | 0.31/47.74 | **0.63/2.59** |
| Woman | ***0.65***/**8.35** | 0.62/***9.44*** | 0.62/10.61 | 0.60/11.80 | 0.14/164.44 | 0.07/124.58 | 0.09/130.90 | **0.69**/11.16 |
| Human9 | **0.41**/9.18 | ***0.36***/**8.67** | ***0.36/8.89*** | 0.29/22.65 | 0.19/40.76 | 0.28/22.79 | 0.13/42.47 | 0.34/27.64 |
| Average | **0.63/16.20** | ***0.55/23.38*** | 0.54/23.86 | 0.52/24.48 | 0.44/54.70 | 0.38/45.41 | 0.36/68.27 | 0.45/58.42 |

The underlined bold numbers indicates the best performance, while the italic bold is the second best.

The average overall minimum Euclidean distance and maximum overlap ratio in the 10 video sequences are given by **WOA-CBWH** tracker, with 16.20 pixels average distance to the true center (ADTC), and 63% average area under curve (AUC), which we consider as acceptable.

We can also notice an important gain margin by 9% in AUC, and 7.66 pixels ADTC, between the first ranking **WOA-CBWH** and **BA** as a reference tracker.

It worth here to mention here that, there is no much as difference in performance between **WOA-CWH** and **BA** performance, in the 10 mentioned sequences.

Among the 10 mentioned sequences: **Bird2**, **BlurOwl**, **Board**, **Human9** and **Girl2** present a considerable correlation between the first initiated target and its background, that's why **WOA-CBWH** provided the best results, the other trackers did not get the best results.

For the sequences such as **Liquor**, **DragonBaby**, and **Lemming**, etc, there is not an important correlation between the template and its background, **WOA-CBWH** and **WOA-CWH** performance were roughly similar, which means that **CBWH** presentation did not affect considerably the tracking performance.

In the **Trellis** sequence, the proposed tracker did not get the best tracking performance due to its use of color feature under background clutter; **DSST** tracker**,** used correlation filters and object feature using texture, it has the best tracking results because it uses HOG feature; which is strong against background clutter, when the deformation of the object is not important, which is the case in this sequence.

In the **Liquor** sequence, Our tracker has not ranked the first because of the frequent out of view situation, **LOT** tracker has the best tracking performance, because it used the particle filter to lock its target, and it seems to recover better from the out of view situation than the other reference tracker.

### c. Qualitative Evaluation

In this evaluation we outlined our comparison to the best four reference trackers such as: **WOA-CBWH, BA, MFPA**, and **KMS-LPQ**. The first results here are screenshots of the tracking results using 8 video sequences in Fig.5.22 (a, b, c, d, e, f, g, h), and the second results displays the Euclidean distance plot per sequence of the same 8 video sequences Fig.5.23 (a, b, c, d, e, f, g, h).

Table.5.9. illustrates the challenging attributes that characterize each of the 8 used sequences.

For the first four sequences in Fig.5.22 and Fig.5.23: **(a)Bird2, (b)BlurCar2, (c)BlurOwl, (d)Trellis**, the presence of background information is important in the object template region, and the accuracy of WOA-CBWH is very important, which is the contrary for the other trackers.

In the **Bird2** sequence, **KMS-LPQ** lost the target definitely from the beginning, while the other trackers such as **MFPA** and **BA** were following the head of the stork, because they were using the traditional presentation CWH. In this sequence only **WOA-CBWH** was able to get the position accurately.

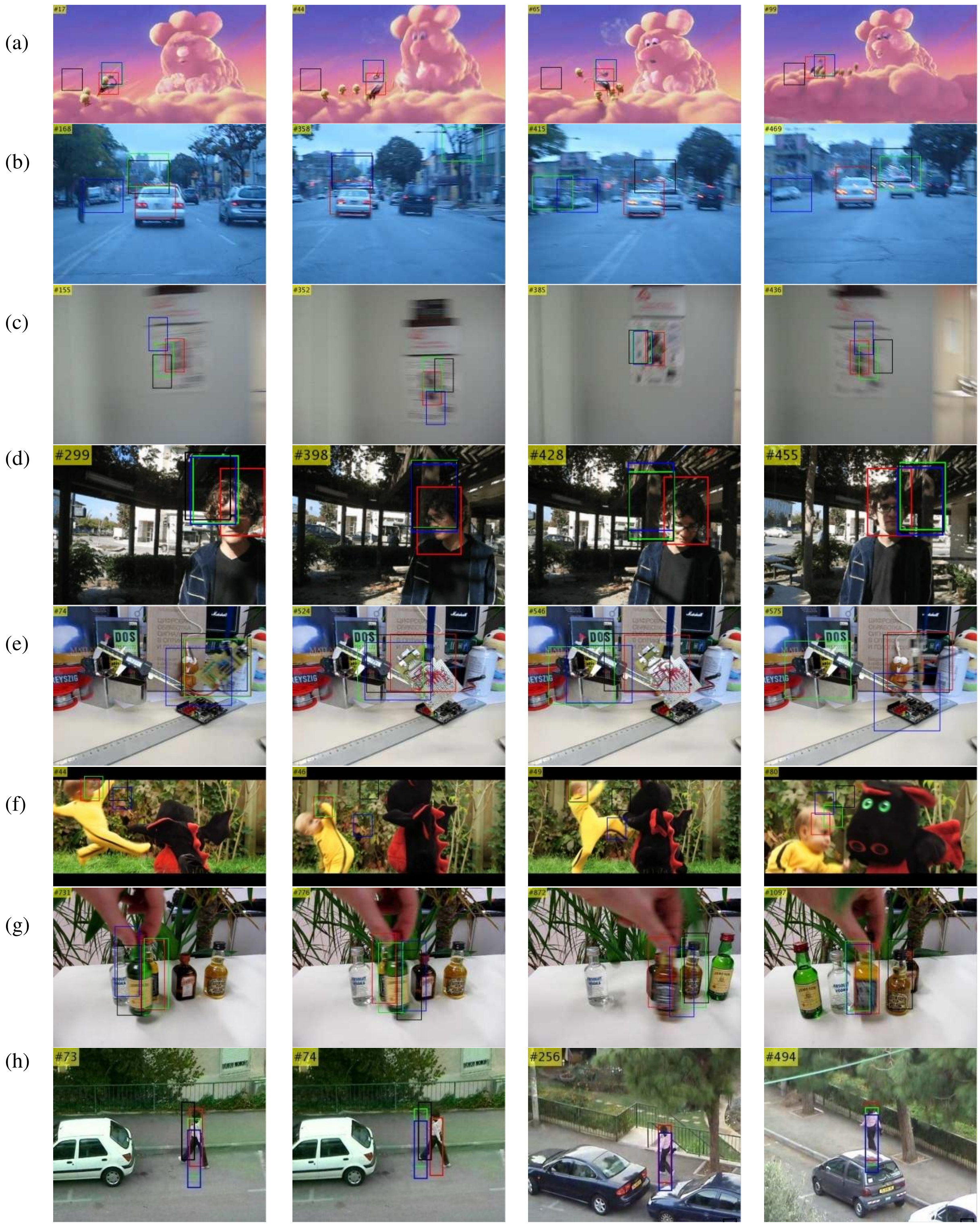
In the **BlurCar2** & **BlurOwl** sequences, the mentioned frames underwent significant blurred and fast motion, **BA** & **MFPA** lost the target completely, and even **KMS** combined with the joint texture-color missed the object on most of the sequences, in contrast to **WOA-CBWH.**

In **Trellis** sequence, even in the illumination changes and background clutter situation that characterize this sequence **WOA-CBWH** successfully could follow the target accurately, while the other trackers such as, **BA, MFPA, and KMS-LPQ** lost half part of the object. In **Board** sequence frame #524, an out of plane rotation, made all the trackers miss the target, except **WOA-CBWH**, and in #546, **BA & MFPA** drifted because of the presence of similar object near the target, per contra **KMS-LPQ & WOA-CBWH**. **WOA-CBWH** have not been affected, because the similar colored object was situated inside the background of the object model.

In **DragonBaby** sequence, in the first frames such as #44, #49, **BA** and **WOA-CBWH** showed similar tracking performance because there were no correlation between the object model and its background. In #46, the target's motion between the adjacent frames #45 and #46 was not continues (out of View), all the trackers failed to get the baby's head right position. In latest frames such as #80, only **WOA-CBWH** was able to follow the baby's because of its better searching ability compared to the other trackers. In **Liquor** sequence, the aim here is to follow a certain bottle. In seq.#731,#776, only **BA** and **WOA-CBWH** was able to follow it, because it underwent half occlusion. But in #872, despite the presence of very similar object near the target, only **WOA-CBWH** was able to get the right position. At last, in seq.#1097, **KMS-LPQ** was the only tracker that failed to get the exact position, because it was distracted by a similar bottle near the target. In the first images of **Woman** sequences, such as seq. # 73 and # 74, all trackers except WOA-CBWH were distracted and lost part of the target's body due to lighting variation. KMS LPQ lost completely the target in the rest of the sequences and only the other meta-trackers were able to follow the woman.

Table 5.9 Description of the challenges characterizing the sequences

| Sequence | Sequence length | Challenging Attribute |
|---|---|---|
| **Bird2** | 99 | Occlusion, Deformation, Fast Motion, In plane rotation, Out of Plane Rotation |
| **BlurCar2** | 585 | Scale Variation, Motion Blur, Fast Motion, |
| **BlurOwl** | 631 | Scale Variation, Motion Blur, Fast Motion, |
| **Board** | 698 | Scale Variation, Motion Blur, Fast Motion, Out of Plane Rotation, Out of View, Background Clutter |
| **DragonBaby** | 113 | Scale Variation, Motion Blur, Fast Motion, In plane rotation, Out of Plane Rotation, Occlusion, Out of View. |
| **Liquor** | 1741 | Illumination Variation, Scale Variation, Occlusion, Motion Blur, Fast Motion, Out of Plane Rotation, Background Clutter, out of view. |
| **Trellis** | 569 | Illumination Variation, Scale Variation, In plane rotation, Out of Plane Rotation, Background Clutter |
| **Woman** | 597 | Illumination Variation, Scale Variation, Occlusion, Deformation, Motion Blur, Fast Motion, Out of Plane Rotation |

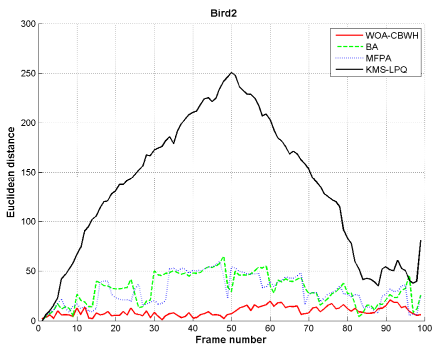


🟥 WOA-CBWH 🟩 BA 🟦 MFPA ⬛ KMS-LPQ

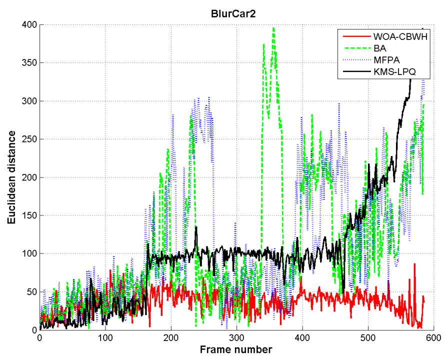
Fig. 5.22 Tracking results of four states of art trackers.

Frame indexes are shown in the top left of each figure. Sequences names are: (a) Bird2, (b) BlurCar2, (c) BlurOwl, (d) Board (e) DragonBaby, (f) Liquor, (g) Trellis, (h) Woman.

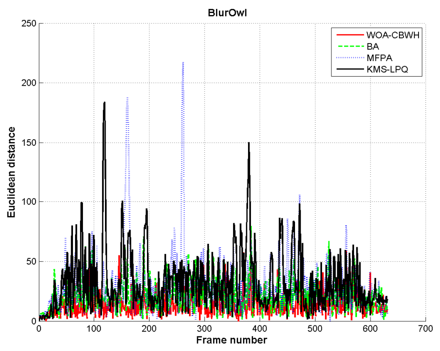
(a)



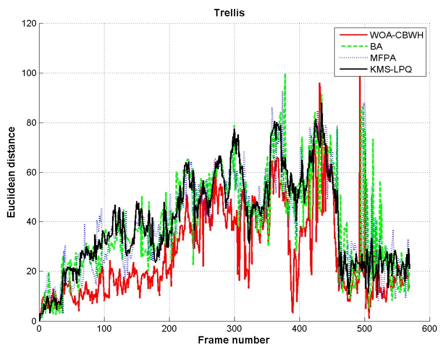
(b)



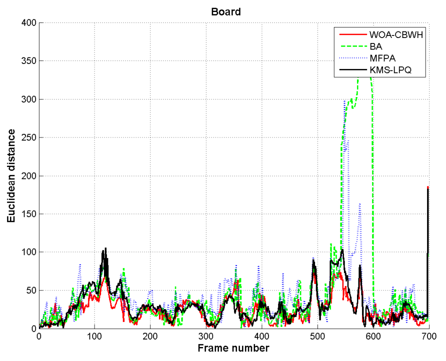
(c)



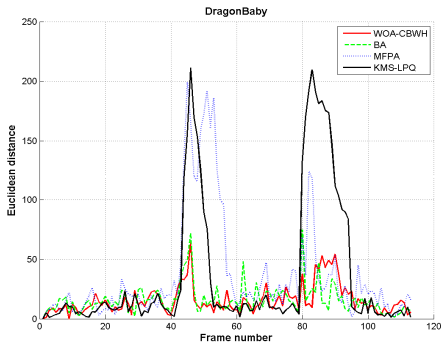
(d)



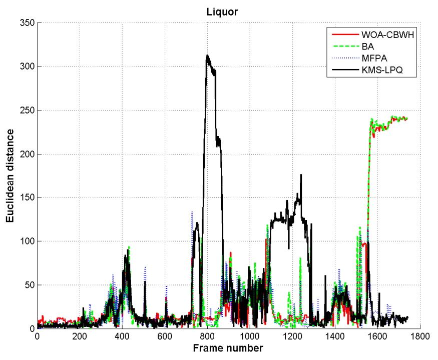
(e)



(f)



(g)



(h)

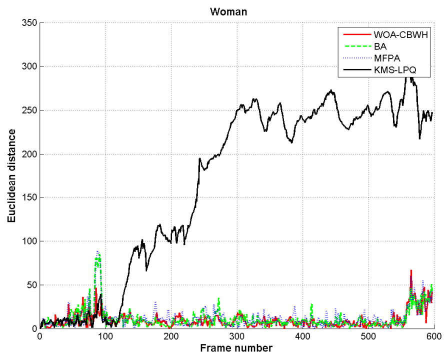


Fig. 5.23 Euclidean distance plots of 4 different trackers in 8 video sequences.

## Discussion of the obtained results

Basically, WOA-CBWH has many advantages, first it exploits the decrease the interference of background information in the object template, which lead to a better localization, which means this approach is more robust for local distractors such as background clutter and illumination variation, compared to the traditional KCWH presentation. Second, it is proved experimentally that metaheuristic searching techniques provide better efficient exploration of the search space than deterministic and probabilistic approaches.

In conclusion, with the many challenges such as: abrupt movement, little background clutter, blurred motion, deformation, rotation and occlusion, the use of CBWH combined with WOA provided the best performance compared to all the mentioned trackers.

#### *5.10.4.2.* Time Cost of tracking

Despite the difference in the implementation platforms, the time cost of tracking is affected by different factors such as the size of the bounding box (BB) that surround the object, the feature used to present the object, the number of particles and the number of iterations used in a metaheuristic algorithm. The time cost of tracking indicates the complexity of the calculation which is directly related to the number of objective function calculations, in both in the initialization and updating process. In the previous experiments the number of fitness function evaluations in WOA was 36. In the previous experiments, the number of fitness function evaluations in WOA was 36 in every frame, table 5.10 shows the comparison of average time costs of the three meta-trackers in 8 video sequences.

Table 5.10 The comparison results of average time costs of the three meta-trackers in 8 video sequences.

| **Video sequences** | **Average Time Cost (S) per frame** | | | | |
|---|---|---|---|---|---|
| | **Bounding Box size** | **WOA** | | **BA** | **MFPA** |
| | | **CBWH** | **CWH** | | |
| **BlurCar2** | 12078 | 0.5641 | 0.5691 | 0.5720 | 1.3619 |
| **Trellis** | 6868 | 0.3574 | 0.3665 | 0.3599 | 0.6858 |
| **BlurOwl** | 5600 | 0.3553 | 0.3349 | 0.3285 | 0.7260 |
| **Bird2** | 5037 | 0.2964 | 0.3041 | 0.3038 | 0.6079 |
| **David3** | 4585 | 0.3000 | 0.2918 | 0.2939 | 0.5825 |
| **DragonBaby** | 3640 | 0.2617 | 0.2509 | 0.2472 | 0.5251 |
| **Boy** | 1470 | 0.1782 | 0.1770 | 0.1825 | 0.2981 |
| **Crossing** | 850 | 0.1375 | 0.1347 | 0.1356 | 0.2075 |

According to table 5.10 above, both WOA and BA have roughly the same the average time cost, because they have the same computation complexity, and the average time cost is proportional to BB size. MFPA have an important average time cost, then an important computation complexity, compared to WOA and BA. Between WOA-CWH and WOA-CBWH there is difference in average time cost, because the CBWH presentation is carried only in the first frame.

#### *5.10.4.3.* The limitation of the proposed tracker

- ✓ No size estimation of the object is studied in this algorithm.
- ✓ A fast motion that exceed object size (width, height), could result a tracking failure.
- ✓ When the target is out of view situation, and changes its position beyond its original size, the proposed approach could result tracking failure, as it doesn't implement a redetection mechanism. (Example: Liquor, DragonBaby, etc).
- ✓ A problem could be encountered in real time, is when the target size is important, fitness evaluation becomes computationally expensive.
- ✓ The use of color feature to model the object makes our algorithm, vulnerable against background clutter (Example: Trellis sequence).
- ✓ Template update mechanism is not implemented, which make our algorithm fail in dynamic scenes, as it does not consider inevitable appearance changes.

## 5.11. Detection Free Tracking of Multiple Objects via Equilibrium Optimizer

Multiple objects tracking (MOT) is a difficult task, as it usually requires high computation complexity. In this section, we present a new framework of MOT by using of equilibrium optimizer (EO) algorithm as a solver of such problem in the detection free tracking framework. First, the target objects are initialized or detected in the first frame, and then modeled by their kernel color histogram to establish a feature model. The Bhattacharya distances between the histogram of object models and other candidates are used as the fitness function to be optimized. Multiple agents are generated by EO, according to the number of the target objects to be tracked. EO Algorithm is used because of its efficiency and lower computation cost compared to other algorithms in global optimization.

To implement EO in MOT DFT framework, we evaluate the displacement vectors of all the objects simultaneously, we propose to use a parallel implementation of EO, then the task of MOT in every frame is considered as a parallel processing optimization problem.

Instead of a single population, several sub-populations are used, each of which has its associated fitness function.

Fig.5.24 shows the EO parallel implementation in the case of k fitness functions.

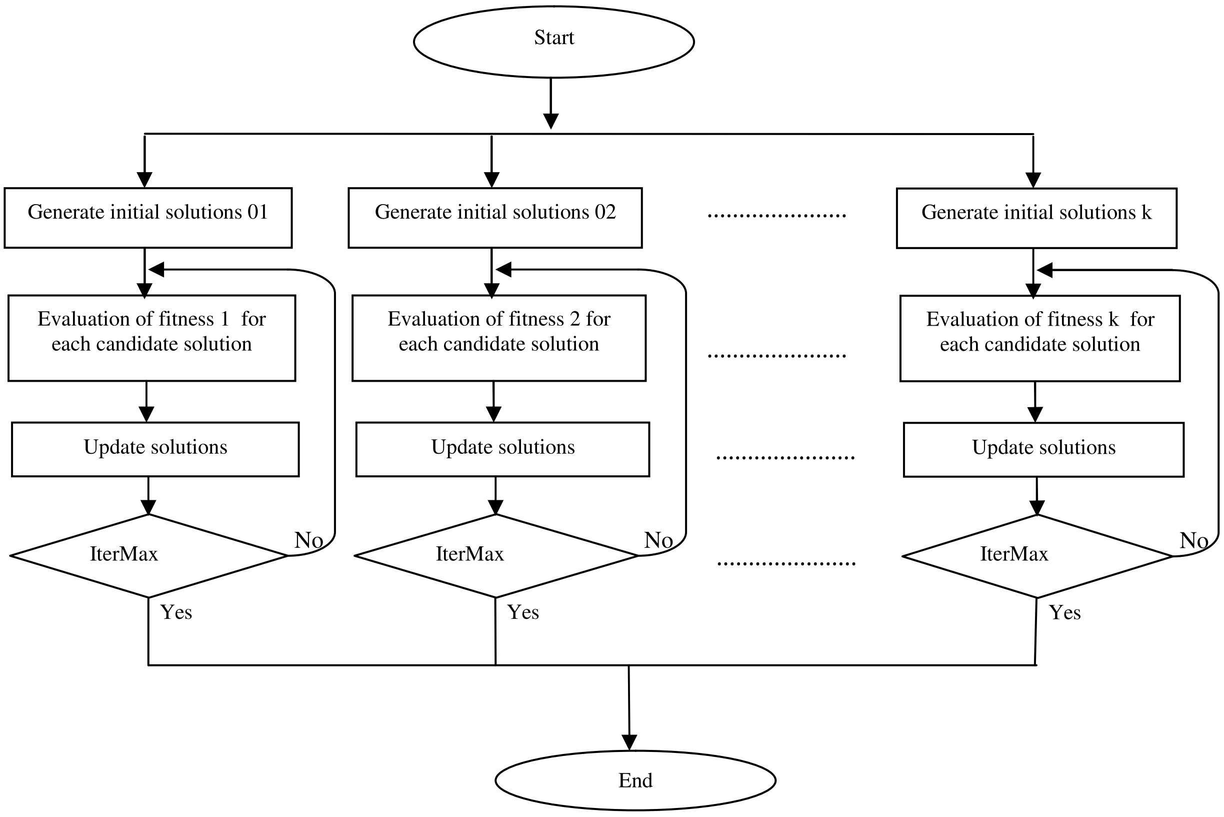

Start
Generate initial solutions 01
Generate initial solutions 02
Generate initial solutions k
Evaluation of fitness 1 for each candidate solution
Evaluation of fitness 2 for each candidate solution
Evaluation of fitness k for each candidate solution
Update solutions
Update solutions
Update solutions
IterMax
IterMax
IterMax
No
No
No
Yes
Yes
Yes
End

Fig. 5.24 Flowchart of EO with a parallel implementation

Unlike the standard version of EO, the parallel implementation Fig.5.24 initializes randomly several sub-populations in different search spaces. Each subpopulation is designated to find a different optimum; this is why each of them is evaluated with its proper fitness function. Subpopulations are then updated with all their corresponding fitness function. The whole processes are repeated until the maximum iteration number is achieved.

In EO MOT multiple agents are generated according to the number of object to be tracked.

In the following, we discuss the steps to implement EO MO.

The first step, include the selection of the targets in interest, either manually from the ground truth or from the screen, then in the second step, the feature models for all objects, are evaluated and stored as a reference. The third and last step includes the parallel optimization, where the agents concentrate on their own target separately.

In EO MO tracker, each target *i* is surrounded by a bounding box $BB_i = (x_i, y_i, w_i, h_i)$ where $\{x_i, y_i\}$ denotes its cartesian coordinate, and $\{w_i, h_i\}$ its initial high and width respectively. EO algorithm search space is specified, by its boundary, $UB_i$ as its upper boundary, and $LB_i$ as its lower boundary, as follow:

$$\bullet\ LB_i = C_i(t) - 1.7 * w_i. \qquad (5.13)$$

$$\bullet\ UB_i = C_i(t) + 1.7 * w_i$$

The factor 1.7 is multiplied to the width area to widen the search space, to recover quickly from occlusion.

```
1.   Read the first sequence of video I
2.   Read target templates initial positions
     from their grounding_truth
3.   Evaluate kernel CWH of the templates
4.   for i=2: frame_lenght(I)
5.   Call EO-MOT
6.   Display tracking results
7.   end for
8.   Save tracking results
```

Fig. 5.25 Pseudo Code of EO-MO tracker

Where $C_i$, refers to the predicted location of the object *i*, updated at the end of every optimization, and set manually at the beginning of every optimization in the initialization process.

### 5.11.1. Fitness function

For both the object template and object candidate, the KCWH explained in the 3rd previous section is performed to model the target in interest.

To measure the similarity or dissimilarity between the histograms Bhattacharya distance is carried.

### 5.11.2. Experimental validation

To illustrate the efficiency of the proposed method, qualitative and quantitative evaluations of its performance are given using two video sequences from OTB2015 Benchmark [85]. The used sequences are '**Jogging**', '**Skating2**', Fig.5.26.

Weight adjusted particle swarm optimizer WAPSO is proposed by Guang Liu et al. [161] and traditional PSO, proposed by M. Thida et al. [162] serves as reference trackers for comparison with our proposed approach. For a fair evaluation of the proposed approaches, the same object model is used, with same sensitivity parameters used by their authors, and the same population size. These parameters are illustrated in table 5.11.

Based on the same statistical study illustrated in the previous chapter using Anova test. We realized through our experiences, that the combination number of agent N=8 and IterMax=2, permits good and stable results of EO-MOT algorithm.

Table 5.11 The parameters setting of every algorithm

| Algorithm | Parameter | Value |
|---|---|---|
| **EO** | N: Population size | 8 |
| | Iter: Iteration number | 2 |
| | $a_1$ | 2 |
| | $a_2$ | 1 |
| | GP | 0.5 |
| | V | 1 |
| **WAPSO** | N: Population size | 8 |
| | Iter: Iteration number | 2 |
| | Wi | 1 |
| | C1 | 2 |
| | C2 | 2 |
| | $\alpha$, $\beta$ | 0.5 |
| **PSO** | N: Population size | 8 |
| | Iter: Iteration number | 2 |
| | C1 | 2 |
| | C2 | 2 |
| | $W_{max}$ | 0.9 |
| | $W_{min}$ | 0.2 |

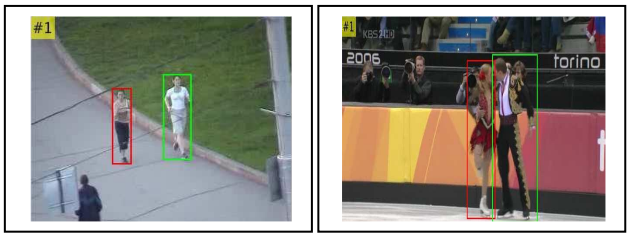


Fig. 5.26 The first frames of the two used videos

From left to right They are named by: Jogging, Skating2.

In the first picture of Fig.5.26, on the left, two women are jogging, in this video EO-based MOT performance is evaluated under three challenging circumstances such as deformation, full occlusion, and out of plane rotations, which are very real world difficult scenarios. In the second image of Fig.5.26. on the right, the objects to be tracked are couple of man and woman doing a skating, this video is much harder than the previous, as it undergoes very challenging situations such as frequent full Occlusion, Scale Variation, Out of plane rotation, Fast Motion, and Deformation. The ID of the first object (Obj.1) is given by a red grounding box, while the second (Obj.2) by a green one.

#### 5.11.2.1. Quantitative comparison

In this part, we report two quantitative assessments, table 5.12 reports the average Euclidean distance and average overlap ratio of the previously mentioned sequences. The underlined bold numbers show the best performance, while the bold ones show the second best. Table 5.13 reports all the trackers performance using the speed metric by showing average time cost and computation complexity. It can be noticed from table 5.12. That the performances of all the trackers are very close in both average overlap and Euclidean distance, and EO-MOT is either the best or the second best in terms of precision in tracking.

However, if we examine table 5.13, EO-MOT has the superiority in computation complexity compared to PSO and WAPSO; as it reaches good tracking performances by evaluating only 32 fitness functions, which is exactly two times the number calculated by PSO and WAPSO; thus explains the average time cost between the three trackers.

Table 5.12 The average performance of three trackers

| | | EO | WAPSO | PSO |
|---|---|---|---|---|
| Jogging | **Obj. 1** | **0.497/22.047** | **0.500/22.318** | **0.485/22.455** |
| | **Obj. 2** | **0.662/12.711** | **0.635/14.593** | **0.627/14.430** |
| Skating2 | **Obj. 1** | **0.495/28.390** | **0.487/28.125** | **0.489/29.097** |
| | **Obj. 2** | **0.397/48.895** | **0.384/53.351** | **0.404/44.995** |
| **Average** | | **0.512/28.011** | ***0.502*/29.597** | **0.501/27.744** |

Table 5.13 Average fitness evaluation's number and time cost of three trackers.

| **Algorithm** | **EO** | **WAPSO** | **PSO** |
|---|---|---|---|
| Number of fitness evaluation | 32 | 64 | 64 |
| Average Time cost per frame in jogging sequence | 0.1693 | 0.4228 | 0.4280 |
| Average Time cost per frame in Skating sequence | 0.7537 | 1.9928 | 2.0662 |

In the video **Skating2**, Obj.2 is fully occluded in so many frames, which explains the important average the small overlap ratio and Euclidean distance for all the trackers.

The tracking speed or the average time cost is mainly linked to the platform of implementation, object presentation, bounding box size, and number of particles in a metaheuristic algorithm. In **Skating2** sequence all the algorithms have an important time because of the bounding box big size. The target objects have a pixel resolution (64x236) for Obj.1 and (103x251) for Obj.2; while in the **Jogging** video sequence, all the algorithm provided a lesser time cost than in the previous sequence, and EO provided the best acceptable time cost (0.169s) equivalent to 6 frames per second or 6HZ, consequently we can assume that EO is better convenient to be implemented in real time than WAPSO and PSO.

### 5.11.2.2. Qualitative comparison

To illustrate qualitatively the tracking outcome of the three trackers, table 5.14 and table 5.15 report four print-screens, in different difficult scenarios.

In table 5.14, **Jogging** sequence. In frame #30, WAPSO and EO obtained the exact location of the **obj.2** while PSO missed a part of the object. In frame #74, **obj.1** was in full occlusion which causes tracking failure of all the trackers. In frames #203 and #194, the two targets suffered deformations, only EO was able to get the precise objects' location, while the others missed.

In table 5.15, **Skating2** sequence. In frame #18, **Obj.2** was completely occluded behind **Obj.1** which leads the non success of all the trackers. In frame #198, #220 only WAPSO missed **Obj.1**, as it experienced deformation and out of plane rotation. In Frame #320, only WAPSO missed the accurate location of **Obj.1** as it experienced partial occlusion and out of plane rotation.

Table 5.14 Tracking results in jogging sequence.

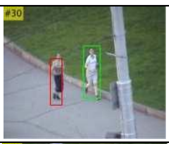
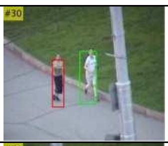
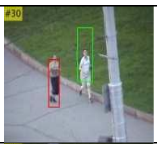
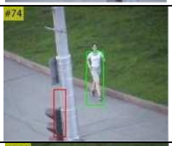
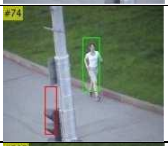
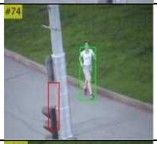
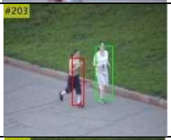
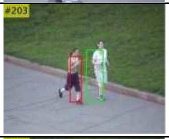
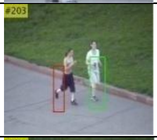
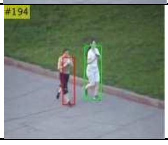
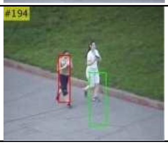
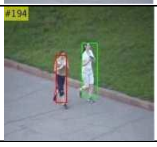
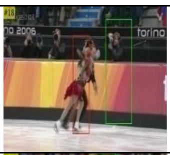
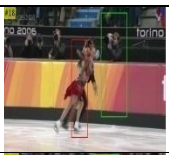
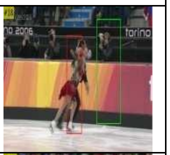
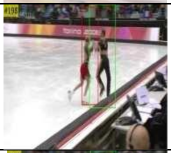
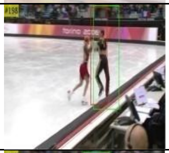
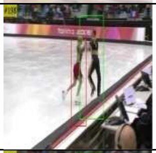
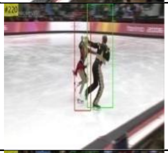
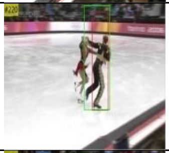
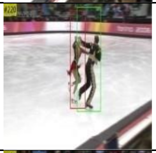
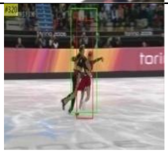
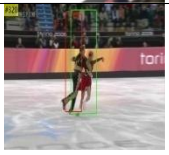
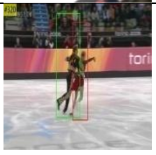
| Frame N° Challenge | EO | WAPSO | PSO |
|---|---|---|---|
| 30 Deformation |  |  |  |
| 74 Full occlusion |  |  |  |
| 203 Deformation |  |  |  |
| 194 Deformation |  |  |  |

Table 5.15 Tracking results in Skating2 sequence

| Frame N° Challenge | EO | WAPSO | PSO |
|---|---|---|---|
| 18 Full occlusion + Out of plane rotation |  |  |  |
| 198 Deformation + Out of plane rotation |  |  |  |
| 220 Deformation + Out of plane rotation |  |  |  |
| 320 Partial occlusion + Out of plane rotation |  |  |  |

## 5.12. Conclusion

In this chapter, we have proposed four new efficient and robust methods, which three are SOT and one for MOT.

For SOT framework, SFS-Tracker and HHO-Tracker used the generative presentation named as Kernel color weighted histogram, as a feature presentation for the object to be tracked, while for WOA-CBWH, is used in a discriminative framework; using CBWH with WOA permits to achieve better tracking results through discriminating background from the target object and its allowed faster convergence rate. We also statistically studied its parameters such as population size and iterations in an original way. To demonstrate the tracking precision of the proposed trackers, a quantitative and qualitative comparative study is carried, using sequences from the OTB15 dataset.

The performance is compared with many reference trackers, classified into three: deterministic, probabilistic and metaheuristic methods. The comparative results show that the SFS, HHO, WOA-CBWH based trackers outperforms the other reference trackers in certain conditions, and under different difficult situations in terms of accuracy. The proposed tracking approaches, are single feature based, although they are robust to some challenges such as: rotation, partial occlusion, non-rigid deformation, and camera motion, they are very sensitive some situations such as: heavy background clutter, and low resolution, etc.

To the best of our knowledge, this is the first time that SFS, HHO, and WOA algorithms has been implemented in VOT framework, and the first results have proven to be a promising and powerful tracking algorithms. To decrease the tracking time cost, for objects with high resolution, future research could be focused on optimizing object feature presentation for less time consumption, so that real-time tracking could be achieved.

For MOT framework, we have applied EO algorithm in MOT DFT framework in very competitive environments such as: rotation, occlusion, deformation, and fast motion, by suggesting a newly architecture of MOT by using EO. To demonstrate the tracking performance of the proposed approach, quantitative and qualitative evaluations with WAPSO and PSO, as reference trackers demonstrated that EO based MOT is better in tracking than PSO, and WAPSO; we can conclude that EO can be applied for real-time MOT in certain conditions, and future works can be dedicated to implementing EO in DBT MOT.

# CONCLUSION AND FUTURE WORK

The problem of object tracking in video sequences has been a very active research topic in the world of computer vision in the recent decades.

Until today, there is no tracker capable of overcoming all the difficult situations that can appear such as: changes in illumination, scale variation, occlusions, camera movement, object deformation, and background clutter, etc. To solve such problems, researchers on this field in most of the cases, are focused on two main components: object representation and object search model.

In this thesis, we have presented a state of the art and a classification of object tracking methods according to new trends and recent references.

So that the present work may help students, and researchers who wants to start the field of object tracking in a simple way by giving the appropriate and simple definitions largely used in the field.

The objective of the thesis was the use of metaheuristic optimization techniques as a searching technique. The reason behind is, it is their simplicity and efficiency in implementation.

The performance of this type of method in terms of precision and complexity of calculation depend on the algorithm adopted. On the other hand, metaheuristic optimization techniques represent efficient tools for solving a large number of optimization problems.

Therefore, in our work we have developed object tracking techniques based on metaheuristics, which allowed us to choose four metaheuristique algorithms that were appropriate for the problem solved:

- The first is by using the stochastic fractal search (SFS) technique, and the second is by using Harris Hawks Optimizer (HHO) as search mechanism, in the single object tracking framework.
- The third is WOA-CBWH which is based on whale optimization algorithm as search mechanism and corrected background weighted histogram (CBWH) applied to the object template, to overcome difficult environments such as: little background clutter, and the presence of salient background information in the object model, etc.
- The fourth method is based on equilibrium optimizer (EO), proposed in the detection free tracking (DFT) of multiple objects framework, which consists in solving the problem of MOT computation complexity, using the EO technique, in a parallel implementation to estimate the state of all the objects simultaneously.

For all our proposed trackers, color histogram is used as feature and Bhattacharyya distance is measured between the two histograms of the template and the candidates to define the fitness distance, in which optimization is sought.

The proposed tracking approaches are single feature based, although they are robust to some challenges such as: rotation, partial occlusion, deformation, and camera motion of the target, they are very sensitive to background clutter, and out of view situation.

The proposed approaches are tested on a large number of video sequences from the OTB2015 dataset, and the obtained simulation results show the superiority of our proposed methods in tracking accuracy compared to other tracking techniques categorized according to their searching mechanism, whether they are deterministic, probabilistic, or metaheuristic.

## Future works

Object tracking is a hot topic and far from getting closed, as perspectives, we propose the use of the EO-MOT in the detection based framework, in a new promising applications such as traffic monitoring of public pedestrian, to verify the respect of the safety distance of 1 meter away between people, and the verification of wearing a mask to slow the spread of COVID-19.

# BIBLIOGRAPHY

[1] H. Lu, D. Wang.: Online Visual Tracking, Springer, (2019).

[2] Charef-Khodja, D., Toumi, A., Medouakh, S., Sbaa, S.: A novel visual tracking method using stochastic fractal search algorithm. SIViP 15(2), 331-339 (2021).

[3] W Luo, X Zhao, and TK Kim. Multiple object tracking: A review. arXiv preprint arXiv:1409.7618, 2014.

[4] Medouakh, S., Boumehraz, M., Terki, N.: Improved object tracking via joint color-LPQ texture histogram based mean shift algorithm. SIViP 12, 583–590 (2018).

[5] Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. PAMI 25(5), 564–577 (2003)

[6] Mihaylova, L., Carmi, A.Y., Septier, F., Gning, A., Pang, S.K., Godsill, S.: Overview of Bayesian sequential Monte Carlo methods for group and extended object tracking. Digit. Sig. Process.: Rev. J. 25(1), 1–16 (2014).

[7] Zhou Z, Zhou M, Li J.: Object tracking method based on hybrid particle filter and sparse representation. Multimed Tools Appl 76(2),2979–299 (2017).

[8] Gao, M., Shen, J., Yin, L., et al.: A novel visual tracking method using bat algorithm. Neurocomputing 177, 612-619 (2016).

[9] Charef-Khodja, D., Toumi, A., Medouakh, S., Sbaa, S.: Efficient visual tracking approach via whale optimizer and corrected background weighted histogram. Multimed Tools Appl (2021). https://doi.org/10.1007/s11042-021-10691-9

[10] TL Liu, HT Chen.: Real-time tracking using trust-region methods. IEEE Trans Pattern Anal Mach Intell 26(3), 397–402 (2004).

[11] Yu, S., Lu, Y., Molloy, D.: A dynamic-shape-prior guided snake model with application in visually tracking dense cell populations. IEEE Trans. Image Process. 28(3), 1513–1527 (2019).

[12] Ong, K.M., Ong, P., Sia, C.K., Low, E.S.: Effective moving object tracking using modified flower pollination algorithm for visible image sequences under complicated background. Appl. Soft Comput. 83, 105625 (2019).

[13] Charef-Khodja, D., Abida, T., Sbaa, S., & Medouakh, S.: Robust visual tracking method based on Harris Hawks algorithm. In: 020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), IEEE. 180-185 (2020).

[14] Yilmaz, A., Javed, O., Shah, M.: Object tracking: a survey. ACM Comput. Surv. 38(4), 1–45 (2006).

[15] Li, X., Hu, W.M., Shen, C.H., Zhang, Z.F., Dick, A., Hengel, A.V.D.: A Survey of Appearance Models in Visual Object Tracking. ACM Trans. Intell. Syst. Technol, 4, 58 (2013).

[16] Amit Y., Felzenszwalb P., Girshick R.: Object Detection. In: Ikeuchi K. (eds) Computer Vision. Springer, Cham. (2020).

[17] Chen, Y., Song, L., Hu, Y., He, R.: Adversarial occlusion-aware face detection. In IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS) (1-9),(2018).

[18] Sundararajan D.: Object Classification. In: Digital Image Processing. Springer, Singapore. https://doi.org/10.1007/978-981-10-6113-4_12 (2017).

[19] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. International journal of computer vision, 128(2), 261-318 (2020).

[20] Y.Wu, J. Lim, and M.-H. Yang.: Online object tracking: A benchmark. In CVPR (2013).

[21] Zhong, W., Li, H., Yang, M.: Robust object tracking via sparsity-based collaborative model. In: IEEE Conference Computer Vision and Pattern Recognition, 1838–1845 (2012)

[22] Q. Wang, F. Chen, W. Xu, M. Yang, An experimental comparison of online object tracking algorithms, in: Proceedings of SPIE: Image and Signal Processing, 1-11(2011).

[23] Kristan, M., et al.: The visual object tracking vot2013 challenge results. In: ICCV2013 Workshops, Workshop on Visual Object Tracking Challenge, 98–111 (2013).

[24] Kristan, M., et al.: The visual object tracking vot2014 challenge results. In: ECCV2014 Workshops, Workshop on Visual Object Tracking Challenge (2014).

[25] Kristan, M., et al.: The visual object tracking vot2015 challenge results. In: ICCV2015 Workshops, Workshop on

Visual Object Tracking Challenge (2015)

[26] Kristan, M., et al.: The visual object tracking vot2016 challenge results. In: ECCV2016 Workshops, Workshop on Visual Object Tracking Challenge (2016)

[27] Kristan, M., et al.: The visual object tracking vot2017 challenge results. In: ICCV2017 Workshops, Workshop on Visual Object Tracking Challenge (2017)

[28] Kristan, M., et al.: The visual object tracking vot2018 challenge results. In: ECCV2018 Workshops, Workshop on Visual Object Tracking Challenge (2018)

[29] Kristan, M., et al.: The seventh visual object tracking vot2019 challenge results. In: ICCV2019 Workshops, Workshop on Visual Object Tracking Challenge (2019)

[30] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Kämäräinen, J. K., ... & Ma, Z.: The eighth visual object tracking VOT2020 challenge results. In European Conference on Computer Vision Springer, Cham, 547-601 (2020).

[31] Q. Wang, F. Chen, W. Xu, M. Yang, An experimental comparison of online object tracking algorithms, In: Proceedings of SPIE: Image and Signal Processing, 1-11(2011)

[32] Dou, J., Qin, Q., & Tu, Z..: Robust visual tracking based on generative and discriminative model collaboration. Multimedia Tools and Applications, 76(14), 15839–15866 (2016).

[33] Chantara, W., Mun, J. H., Shin, D. W., & Ho, Y. S.: Object tracking using adaptive template matching. IEIE Transactions on Smart Processing and Computing, 4(1), 1-9 (2015).

[34] K. Briechle and U. D. Hanebeck.: Template Matching Using Fast Normalized Cross Correlation. In: Proceedings of SPIE: Optical Pattern Recognition XII. 4387, 95–102 (2001).

[35] Haim Schweitzer, JW Bell, and Feng Wu.: Very fast template matching. In: Proceedings of European Conference on Computer Vision, 358–372 (2002).

[36] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas.: Tracking-Learning-Detection. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 34(7), 1409–1422 (2012).

[37] Jakob Santner et al.: Prost: Parallel robust online simple tracking. In: Proceedings of Conference on Computer Vision and Pattern Recognition, 723–730 (2010).

[38] D. Oliva and E. Cuevas, Advances and Applications of Optimized Algorithms in Image Processing, Intelligent Systems Reference Library 117,(2017).

[39] Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence, 2, 674–679 (1981).

[40] Alt, N., Hinterstoisser, S., Navab, N.: Rapid selection of reliable templates for visual tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1355–1362 (2010)

[41] Matthews, L., Ishikawa, T., Baker, S.: The template update problem. IEEE Trans. Pattern Anal. Mach. Intell. 26(6), 810–815 (2004)

[42] Hager, G.D., Belhumeur, P.N.: Efficient region tracking with parametric models of geometry and illumination. IEEE Trans. Pattern Anal. Mach. Intell. 20(10), 1025–1039 (1998).

[43] Misra, R., & Ray, K. S.: Object tracking based on quantum particle swarm optimization. In Ninth International Conference on Advances in Pattern Recognition (ICAPR), IEEE, 1-6 (2017).

[44] Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. IEEE Trans. Pattern Anal. Mach. Intell. 27(10), 1631–1643 (2005).

[45] Bae, C., Kang, K., Liu, G., & Chung, Y. Y.: A novel real time video tracking framework using adaptive discrete swarm optimization. Expert Systems with Applications, 64, 385-399. (2016).

[46] Kate, P., Francis, M., & Guha, P.,.: Visual tracking with breeding fireflies using brightness from background-foreground information. In 24th International Conference on Pattern Recognition (ICPR) IEEE, 2570-2575. (2018).

[47] Nenavath, H., Jatoth, R. K., & Das, S. A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. Swarm and Evolutionary Computation, 43, 1-30, (2018).

[48] Narayana, M., Nenavath, H., Chavan, S., & Rao, L. K.: Intelligent visual object tracking with particle filter based on Modified Grey Wolf Optimizer. Optik, 193, 162913,(2019).

[49] Liu, G., Chung, Y. Y., & Yeh, W. C..: A simplified swarm optimization for object tracking. In International Joint

Conference on Neural Networks (IJCNN) IEEE, 169-176, (2016).

[50] Yeung, H. W. F., Liu, G., Chung, Y. Y., Liu, E., & Yeh, W. C.: Hybrid gravitational search algorithm with swarm intelligence for object tracking. In International Conference on Neural Information Processing. Springer, Cham, 213-221 (2016).

[51] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition, 886–893 (2005).

[52] Zhang, H., Zhang, X., Qian, X., Chen, Y., & Wang, F..: A novel visual tracking method based on moth-flame optimization algorithm. In Chinese Conference on Pattern Recognition and Computer Vision (PRCV) Springer, Cham,284-294, (2018).

[53] Zhang, H., Gao, Z., Zhang, J., & Yang, G.: Visual tracking with levy flight grasshopper optimization algorithm. In Chinese Conference on Pattern Recognition and Computer Vision (PRCV), Springer, Cham.217-227, (2019).

[54] Zhang, H., Zhang, X., Wang, Y., Qian, X., & Wang, Y.: Extended cuckoo search-based kernel correlation filter for abrupt motion tracking. IET Computer Vision, 12(6), 763-769, (2018).

[55] Zhang, H., Gao, Z., Zhang, J., Liu, J., Nie, Z., & Zhang, J.: Hybridizing extended ant lion optimizer with sine cosine algorithm approach for abrupt motion tracking. EURASIP Journal on Image and Video Processing, 1, 4. (2020).

[56] Tuzel, O., Porikli, F., & Meer, P..: Region covariance: A fast descriptor for detection and classification. In European conference on computer vision, Springer, 589-600 (2006).

[57] Patil, D.O., Hamde, S.T. Automated detection of brain tumor disease using empirical wavelet transform based LBP variants and ant-lion optimization. Multimed Tools Appl (2021). https://doi.org/10.1007/s11042-020-10434-2

[58] Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vis. 57(2), 137–154 (2004).

[59] Wouwer, G., Scheunders, P. and Dyck, D.: Statistical texture characterization from discrete wavelet representations, IEEE Trans. Imag. Process. 8(4), 592–598 (1999).

[60] Ojala,T., Pietikäinen, M.,Mäenpää,T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. PatternAnal. Mach. Intell. 24(7), 971–987(2002).

[61] Lun Zhang, Rufeng Chu, Shiming Xiang, Shengcai Liao, and Stan Z Li. Face detection based on multi-block lbp representation. In Advances in biometrics, Pp 11–18. Springer, (2007).

[62] Tan, X., Triggs, B.: Enhanced local texture feature sets for face recognition under difficult lighting conditions. Trans. Img. Proc. 19(6):1635–1650 (2010).

[63] Ojansivu, V., Heikkila, J.: Blur insensitive texture classification using local phase quantization. In: Proc. Image and Signal Processing ICISP, Cherbourg-Octeville, 236-243 (2008).

[64] T. B. Dinh, N. Vo, and G. Medioni. Context Tracker: Exploring Supporters and Distracters in Unconstrained Environments. In CVPR, (2011).

[65] Z. Kalal, J. Matas, and K. Mikolajczyk. P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints. In CVPR, (2010).

[66] Moghaddasi, S. S., & Faraji, N.: A hybrid algorithm based on particle filter and genetic algorithm for target tracking. Expert Systems with Applications, 147, 113188, (2020).

[67] Sardari, F., & Moghaddam, M. E.: A hybrid occlusion free object tracking method using particle filter and modified galaxy based search meta-heuristic algorithm. Applied Soft Computing, 50, 280-299, (2017).

[68] Avidan, S.: Ensemble tracking. IEEE Trans. Pattern Anal. Mach. Intell. 29(2), 261–271 (2007).

[69] Hare, S., Saffari, A., Torr, P.H.: Struck: structured output tracking with kernels. In: IEEE International Conference on Computer Vision, pp. 263–270 (2011)

[70] Stenger, B., Woodley, T., Cipolla, R.: Learning to track with multiple observers. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2647–2654 (2009).

[71] Kwon, J., Lee, K.M.: Visual tracking decomposition. In: IEEE Conference on ComputerVision and Pattern Recognition, pp. 1269–1276 (2010).

[72] Kwon, J., Lee, K.M.: Tracking by sampling trackers. In: IEEE International Conference on Computer Vision, pp. 1195–1202 (2011).

[73] Henriques, J. F., Caseiro, R., Martins, P., & Batista, J.: High-speed tracking with kernelized correlation filters. IEEE transactions on pattern analysis and machine intelligence, 37(3), 583-596, (2014).

[74] Kalman, R. E.: A new approach to linear filtering and prediction problems. Journal of Fluids Engineering, 82(1),35–45 (1960).

[75] Ma, L., Liu, J., Wang, J., Cheng, J., Lu, H.: A improved silhouette tracking approach integrating particle filter with graph cuts. In: IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), IEEE,1142–1145. (2010).

[76] Zhou, N., Meng, D., & Lu, S. (2013). Estimation of the dynamic states of synchronous machines using an extended particle filter. IEEE Transactions on Power Systems, 28(4), 4152-4161.

[77] Poppe, R.: Condensation-conditional density propagation for visual tracking. Comput. Vis. Image Underst. 108, 4–18 (2007).

[78] Jia, X., Lu, H., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: IEEE Conference on Computer Vision and Pattern Recognition, 1822–1829 (2012).

[79] Matthews, L., Ishikawa, T., Baker, S.: The template update problem. IEEE Trans. Pattern Anal. Mach. Intell. 26(6), 810–815 (2004).

[80] Ross, D.A., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. Int. J. Comput. Vis. 77(1–3), 125–141 (2008)

[81] Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: The British Machine Vision Conference, vol. 1, p. 6 (2006)

[82] Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: IEEE International Conference on Computer Vision, 3119–3127 (2015).

[83] Wang, L., Ouyang, W., Wang, X., Lu, H.: Stct: Sequentially training convolutional networks for visual tracking. In: IEEE Conference on Computer Vision and Pattern Recognition (2016).

[84] Sun, C.,Wang, D., Lu, H., Yang, M.H.: Learning spatial-aware regressions for visual tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, 8962–8970 (2018).

[85] Wu, Y., Lim, J., Yang, M.: Object tracking benchmark. IEEE Trans. Pattern Anal. Mach. Intell.37(9), 1834–1848 (2015).

[86] Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, 4293–4302 (2016).

[87] Zhou, Z., Liao, H., Gu, B., Huq, K. M. S., Mumtaz, S., & Rodriguez, J.: Robust mobile crowd sensing: When deep learning meets edge computing. IEEE Network, 32(4), 54-60, (2018).

[88] Smeulders, A.W.M., Chu, D.M., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.: Visual racking: an experimental survey. IEEE Trans. Pattern Anal. Mach. Intell. 36(7), 1442–1468 (2013).

[89] T. Vojir, J. Noskova, and J. Matas. Robust scale-adaptive mean-shift for tracking. Image Analysis, 652–663, (2013).

[90] J. Ferryman and A. Shahrokni, "PETS2009: Dataset and challenge," 2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, pp. 1-6, doi: 10.1109/PETS-WINTER.2009.5399556(2009).

[91] Liang, P., Blasch, E., Ling, H.: Encoding color information for visual tracking: algorithms and benchmark. IEEE Trans. Image Process. 24(12), 5630–5644 (2015).

[92] Liu, Q., He, Z., Li, X., & Zheng, Y.: PTB-TIR: A thermal infrared pedestrian tracking benchmark. IEEE Transactions on Multimedia, 22(3), 666-675, (2019).

[93] Duan, G., Ai, H., Cao, S., & Lao, S.: Group tracking: Exploring mutual relations for multiple object tracking. In European conference on computer vision. Springer, 129-143(2012).

[94] Elhoseny, M.: Multi-object Detection and Tracking (MODT) Machine Learning Model for Real-Time Video Surveillance Systems. Circuits Syst Signal Process 39, 611–630 (2020). https://doi.org/10.1007/s00034-019-01234-7.

[95] Gaikwad, B., & Karmakar, A.: Smart surveillance system for real-time multi-person multi-camera tracking at the edge. Journal of Real-Time Image Processing, 1-15 (2021).

[96] Borges, P. V. K., Conci, N., & Cavallaro, A.: Video-based human behavior understanding: A survey. IEEE transactions on circuits and systems for video technology, 23(11), 1993-2008 (2013).

[97] Mabrouk, A. B., & Zagrouba, E..: Abnormal behavior recognition for intelligent video surveillance systems: A review. Expert Systems with Applications, 91, 480-491(2018).

[98] Candamo, J., Shreve, M., Goldgof, D. B., Sapper, D. B., & Kasturi, R..: Understanding transit scenes: A survey on human behavior-recognition algorithms. IEEE transactions on intelligent transportation systems, 11(1), 206-224 (2009).

[99] Uchiyama, H., & Marchand, E..: Object detection and pose tracking for augmented reality: Recent approaches. In 18th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV), (2012).

[100] Sen M Kuo, Bob H Lee, and Wenshun Tian.: Real-time digital signal processing: fundamentals, implementations and applications. John Wiley & Sons, 2013.

[101] Murray, S.. Real-time multiple object tracking-a study on the importance of speed. (2017) arXiv preprint arXiv:1709.03572.

[102] Hu, W., Li, X., Luo, W., Zhang, X., Maybank, S., & Zhang, Z..: Single and multiple object tracking using log-Euclidean Riemannian subspace and block-division appearance model. IEEE transactions on pattern analysis and machine intelligence, 34(12), 2420-2440 (2012).

[103] L. Zhang and L. van der Maaten,.: Structure preserving object tracking," in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.,1838–1845 (2013)

[104] Zhang, L., & Van Der Maaten, L.. Preserving structure in model-free tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(4), 756-769 (2013)

[105] Yang, M., Yu, T., & Wu, Y.. Game-theoretic multiple target tracking. In 2007 IEEE 11th International Conference on Computer Vision. IEEE, 1-8 (2007).

[106] Leal-Taixé, L., Milan, A., Reid, I., Roth, S., & Schindler, K.. Motchallenge 2015: Towards a benchmark for multi-target tracking. (2015), arXiv preprint arXiv:1504.01942.

[107] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L.: Imagenet large scale visual recognition challenge. International journal of computer vision, 115(3), 211-252 (2015).

[108] Barekatain, M., Martí, M., Shih, H. F., Murray, S., Nakayama, K., Matsuo, Y., & Prendinger, H..: Okutama-action: An aerial view video dataset for concurrent human action detection. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 28-35, (2017).

[109] Choi, W., & Savarese, S..: Multiple target tracking in world coordinate with single, minimally calibrated camera. In European Conference on Computer Vision. Springer, 553-567, (2010).

[110] Kasturi, R., Goldgof, D., Soundararajan, P., Manohar, V., Garofolo, J., Bowers, R., ... & Zhang, J.: Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(2), 319-336(2008).

[111] Bernardin, K., & Stiefelhagen, R.: Evaluating multiple object tracking performance: the clear mot metrics. EURASIP Journal on Image and Video Processing, 1-10(2008).

[112] Ristani, E., Solera, F., Zou, R., Cucchiara, R., & Tomasi, C.: Performance measures and a data set for multi-target, multi-camera tracking. In European conference on computer vision, Springer, Cham,17-35 (2016).

[113] Harold W Kuhn.: The hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2):83–97, 1955

[114] Smairi, N.:Optimisation par essaim particulaire: adaptation de tribes à l'optimisation multiobjectif. Doctoral dissertation, Paris Est..(2013).

[115] Betka, A.: Estimation de mouvement par les techniques métaheuristiques. LMD doctoral dissertation, uni-Biskra.(2019).

[116] Van Laarhoven, P. J., & Aarts, E. H.: Simulated annealing. In Simulated annealing: Theory and applications (pp. 7-15). Springer, Dordrecht. (1987).

[117] Cvijović, D., & Klinowski, J.: Taboo search: an approach to the multiple minima problem. Science, 267(5198), 664-666. (1995)

[118] Feo, T. A., & Resende, M. G.: Greedy randomized adaptive search procedures. Journal of global optimization, 6(2), 109-133 (1995).

[119] Mladenović, N., & Hansen, P.: Variable neighborhood search. Computers & operations research, 24(11), 1097-1100. (1997)

[120] Lourenço, H. R., Martin, O. C., & Stützle, T.: Iterated local search. In Handbook of metaheuristics (pp. 320-353). Springer, Boston, MA. (2003).

[121] Salimi, H.: Stochastic fractal search: a powerful metaheuristic algorithm. Knowledge-Based Systems, 75, 1-18

(2015).

[122] Kennedy, J., & Eberhart, R.: Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks, IEEE, 4, 1942-1948 (1995).

[123] Mirjalili, S.: SCA: a sine cosine algorithm for solving optimization problems. Knowledge-based systems, 96, 120-133 (2016)

[124] Holland, J. H.: Genetic algorithms. Scientific american, 267(1), 66-73(1992).

[125] Blickle, T., & Thiele, L.: Genetic programming and redundancy. choice, 1000, 2. (1994).

[126] Rechenberg I.: Evolutions strategien. Springer Berlin Heidelberg, 83–114 (1978).

[127] Simon D .: Biogeography-based optimization. IEEE Trans Evol Comput, 12:702–13 (2008).

[128] Storn R, Price K .: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim, 11:341–59 (1997).

[129] Hatamlou A .: Black hole: a new heuristic optimization approach for data clustering. Inf Sci, 222, 175–84 (2013).

[130] Kaveh A .: Colliding bodies optimization. Advances in metaheuristic algorithms for optimal design of structures. Springer, 195–232, (2014).

[131] Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S.: Equilibrium optimizer: A novel optimization algorithm. Knowledge-Based Systems, 191, 105190 (2020).

[132] Dorigo M, Birattari M, Stutzle T .: Ant colony optimization. IEEE Comput Intell,1:28–39 (2006).

[133] Mucherino, A., & Seref, O.: Monkey search: a novel metaheuristic search for global optimization. In AIP conference proceedings. American Institute of Physics, 953(1), 162-173 (2007).

[134] Yang X-S, Deb S.: Cuckoo search via Lévy flights. In: Proceedings of the world congress on nature & biologically inspired computing, NaBIC, 210–14 (2009).

[135] Yang X-S .: A new metaheuristic bat-inspired algorithm. In: Proceedings of the workshop on nature inspired cooperative strategies for optimization (NICSO). Springer, 65–74 (2010).

[136] Yang X-S .: Firefly algorithm, stochastic test functions and design optimisation. Int J Bio-Inspired Comput;2:78–84 (2010).

[137] Geem, Z. W., Kim, J. H., & Loganathan, G. V.: A new heuristic optimization algorithm: harmony search. simulation, 76(2), 60-68 (2001).

[138] Rao RV, Savsani VJ, Vakharia DP .: Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. Inf Sci, 183,1–15 (2012).

[139] He S, Wu Q, Saunders J .: A novel group search optimizer inspired by animal behavioural ecology. In: Proceedings of the IEEE congress on evolution- ary computation, CEC, 1272–8 (2006) .

[140] Atashpaz-Gargari E, Lucas C . Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: Pro- ceedings of the IEEE congress on evolutionary computation, CEC; 4661–7 (2007).

[141] Kashan AH .: League championship algorithm: a new algorithm for numerical function optimization. In: Proceedings of the international conference on soft computing and pattern recognition, SOCPAR’09, 43–8 (2009).

[142] Tan Y, Zhu Y . Fireworks algorithm for optimization. Advances in swarm intel- ligence. Springer, 355–64 (2010).

[143] Ibrahim, Z., Aziz, N. H. A., Aziz, N. A. A., Razali, S., & Mohamad, M. S.: Simulated Kalman filter: a novel estimation-based metaheuristic optimization algorithm. Advanced Science Letters, 22(10), 2941-2946 (2016).

[144] Wolpert, D.H.: The lack of a priori distinctions between learning algorithms. Neural Comput. 8(7), 1341–1390 (1996).

[145] H. Hachimi.: Hybridations d’algorithmes métaheuristiques en optimisation globale et leurs applications. Doctoral dissertation: Optimisation et Fiabilité en Mécanique des Structures, l’université INSA de Rouen, (2013).

[146] G. Cohen.: Convexité et optimisation. Course, Ecole Nationale des Ponts et Chaussées, 139, 2000.

[147] A. El Dor.: Perfectionnement des algorithmes d’optimisation par essaim particulaire: applications en segmentation d’images et en électronique”. Doctoral dissertation: Images, Signaux et Systèmes Intelligents, l’université Paris-Est, (2012).

[148] Betka, A., Terki, N., Toumi, A., Hamiane, M., & Ourchani, A.: A new block matching algorithm based on stochastic fractal search. Applied Intelligence, 49(3), 1146-1160 (2019).

[149] Dhal, K. G., Gálvez, J., Ray, S., Das, A., & Das, S.: Acute lymphoblastic leukemia image segmentation driven by stochastic fractal search. Multimedia Tools and Applications, 1-29 (2020).

[150] Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H.: Harris hawks optimization: Algorithm and applications. Future generation computer systems, 97, 849-872 (2019).

[151] Mirjalili, S., & Lewis, A.: The whale optimization algorithm. Advances in engineering software, 95, 51-67 (2016).

[152] M.L. Gao, L.J. Yin, G.F. Zou, H.T. Li, W. Liu,.: Visual tracking method based on cuckoo search, Opt. Eng. 54 (7) (2015) 073105.

[153] Zhang, X., Hu, W., Maybank, S., Li, X., & Zhu, M.: Sequential particle swarm optimization for visual tracking. In IEEE conference on computer vision and pattern recognition, IEEE, 1-8 (2008).

[154] Medouakh, S.: Détection et suivi d'objets. Dotoral dissertation, uni-Biskra. (2019).

[155] Pérez, P., Hue, C., Vermaak, J., & Gangnet, M..: Color-based probabilistic tracking. In European Conference on Computer Vision . Springer, 661-675 (2002).

[156] Adam, A., Rivlin, E., & Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In IEEE Computer society conference on computer vision and pattern recognition (CVPR'06), IEEE, 1, 798-805 (2006).

[157] Oron, S., Bar-Hillel, A., Levi, D., & Avidan, S.: Locally orderless tracking. International Journal of Computer Vision, 111(2), 213-228 (2015).

[158] Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. Robust visual tracking via structured multi-task sparse learning. International journal of computer vision, 101(2), 367-383(2013).

[159] Ning, J., Zhang, L., Zhang, D., & Wu, C.: Robust mean-shift tracking with corrected background-weighted histogram. IET computer vision, 6(1), 62-69 (2012).

[160] Danelljan, M., Häger, G., Khan, F. S., & Felsberg, M..: Discriminative scale space tracking. IEEE transactions on pattern analysis and machine intelligence, 39(8), 1561-1575(2016).

[161] Liu, G., Chen, Z., Yeung, H. W. F., Chung, Y. Y., & Yeh, W. C.: A new weight adjusted particle swarm optimization for real-time multiple object tracking. In International Conference on Neural Information Processing. Springer, Cham. 643-651 (2016).

[162] Thida, M., Remagnino, P., & Eng, H. L.: A particle swarm optimization approach for multi-objects tracking in crowded scene. In 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops. IEEE. 1209-1215(2009).