


AIX-MARSEILLE UNIVERSITE

ÉCOLE CENTRALE MARSEILLE

Ecole Doctorale Physique et Sciences de la Matière

IntuiSense

Institut Fresnel

Thèse présentée pour obtenir le grade universitaire de docteur

Discipline : Optique, Photonique et Traitement d'Image

Benoit MARTIN

Méthode d'optimisation mixte bio-inspirée : Application à l'imagerie multi-spectrale et à la mesure d'audience

Soutenue le XX/10/2018 devant le jury composé de :

M. Pascal CHARGÉ	Polytech Nantes	Rapporteur
M. Camel TANOUGAST	Université de Lorraine	Rapporteur
M. Ahmed BOURIDANE	Northumbria University	Examinateur
Mme Caroline FOSSATI	École Centrale de Marseille	Examinateur
M. Salah BOURENNANE	École Centrale de Marseille	Directeur de thèse
M. Julien MAROT	Aix Marseille Université	Co-Directeur de thèse

Numéro national de thèse/suffixe local :


AIX-MARSEILLE UNIVERSITE

ÉCOLE CENTRALE MARSEILLE

Ecole Doctorale Physique et Sciences de la Matière

IntuiSense

Institut Fresnel

Thesis presented to obtain the degree of Doctor

Scientific field : Optics, Photonics and Image Processing

Benoit MARTIN

Mixed bio-inspired optimization method : Application to multispectral image processing and audience measurement.

presented on XX/10/2018
COMMITTEE :

M. Pascal CHARGÉ	Polytech Nantes	Rapporteur
M. Camel TANOUGAST	Université de Lorraine	Rapporteur
M. Ahmed BOURIDANE	Northumbria University	Examinateur
Mme Caroline FOSSATI	École Centrale de Marseille	Examinateur
M. Salah BOURENNANE	École Centrale de Marseille	Directeur de thèse
M. Julien MAROT	Aix Marseille Université	Co-Directeur de thèse

Numéro national de thèse/suffixe local :

Acknowledgment

This CIFRE thesis is the result of a collaboration between the "Groupe Signaux Multidimensionnels" (GSM) team of the Fresnel Institute, in Marseille (FRANCE), and the company IntuiSense, in Gemenos (FRANCE).

I am sincerely grateful to Professor Salah BOURENNANE, head of the GSM team, and Doctor Julien MAROT, Associated Professor of Aix-Marseille Université, for their supervision. Julien always succeeded in transmitting me its huge motivation and in guiding me or giving me hints and advices all along these 3 years. Salah always managed to point out the path to explore in this work, while bringing a good mood and laughters in the office. Moreover, Salah's meticulous readings of the articles that have been published during this thesis greatly improved their quality.
I am also grateful to Régis LE BRAS, Alexandre JAUD and Frédéric GUÉRAULT all from or previously from the company IntuiSense for their technical supervision. Their support helped me a lot in order to have a more industrial viewpoint of my research work than I would have by working solely in a laboratory.
I was honored by all members of the jury who examined this dissertation :

— M. P. CHARGÉ,
— M. C. TANOUGAST,
— M. A. BOURIDANE,
— Ms. C. FOSSATI.

I would like to thank all the people from the GSM team and the people from the Institut Fresnel as a whole for making these 3 years enjoyable.
I would like to especially thank my friends from the office 227 : Marion, Khaled, Nicola and Hind for their psychological support, their happiness and the moments we spent together....even if we still have to do a picnic on the beach. I hope the future will be full of success and happiness for the four of you, I know you deserve it and it will happen.
I would like to thank my colleagues from IntuiSense : Tonton Manu, Kéké, Chauve, Capitaine, S.Normal, Mémé, Figue, Régis and the newly arrived Christophe and Paître for their constant good and light-hearted mood and all the futsal and ping-pong matches we had during these 3

years. I hope you will keep this state of mind after my departure. I would also like to make a little dedication to the people who left us and those who only passed by : Nathalie, Thibault, Lucie, David, Fred, Jean-Paul and Christophe, wherever they are.
I thank my parents who educated me, for better or for worse, and allowed me, through the constant moving, to have an open-mind on the world who surround us. I also thank my two little brothers Arnaud and Hugo for having supported me for 18 years (or is it the contrary ?).
I want to thank all the other friend I made in Marseille : Lucas, Lisa, Chiva, Tangui, Juliette, Armand, Nina, Romane and Loïc for the good moments we spent in Marseille, whether it was just for taking a stroll, having a drink, playing card games or watching football games (We have two stars !). I also want to thank my friends Loïc, Simon and Clément from high school for their good humour and their love for rock and metal music which contaminated me to the core. I hope you are all doing well in your respective places and that we will finally manage to meet once again very soon.
I also thank the retkipumput mestarit I met in Oulu : Giulia, Jan, Reeta, Vincent and Marius who welcomed us in San Daniele and with who I share unforgettable memories.
I want to warmly thank my favorite pangolins from Rennes : Kevin, Quentin, Abderrahmane, Bruno and Thibaud for being my friends. Although we are scattered all around France, and soon all around the world, I know that we will remain as the AN4 group. I still remember what Thibaud said when we started our studies in the 1-D group at the INSA : "You know what folks ? People say that the friends you make after high school remain your friends for life".
And most importantly, I want to sincerely thank Rozenn, Tsuki and Kaboom for their constant love and support. I thank them for supporting my Martin's jokes, whether they were perfect or awful, and my mood, whether it was good or bad. I thank them for welcoming me in Marseille and for living this 3 years experience with me, although i am still sad to have missed Maiden and At the Gates because of our PhDs...
Finally, I would like to thank all the people I did not name here but still participated in the accomplishment of this work or were a part of what I am today, whether it was in France, in Djibouti, in Brittany, in Finland or somewhere else around the globe.

Table of contents

Nomenclature

Notations used

a or A	scalars
$\mathbf{a}$	vectors
$\mathbf{A}$	matrices
$\mathbb{A}$	manifolds
$\mathcal{A}$	tensors

Parameters used

N	Number of expected parameters
i	Index of the expected parameter with $i = 1, \cdots, N$
K_i	i-th expected parameter
$f(\cdot)$	Function to minimize, also called criterion
a	Phase distingisher
iter	current iteration
$\mathrm{T_{max}}$	Maximum number of iterations
η	Phase emphasizer
Q	Number of wolves in the Herd
$\mathbf{x}_q^{\mathbf{k}}$(iter)	Position of the q-th wolf, with $q = 1, \cdots, Q$
$\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, $\mathbf{x}_\delta^{\mathbf{k}}$	Positions of the leaders α, β and δ
$\mathbf{x}_{\rho 1}^{\mathbf{k}}$, $\mathbf{x}_{\rho 2}^{\mathbf{k}}$	Positions of the random wolves $\rho 1$ and $\rho 2$
$\hat{K}_i$	i-th estimated parameter

Parameters solely used for Improve Discrete GWO

H_i	Number of candidate value for the parameter K_i
$\mathbf{d}_i^{val} = \left[K_i^1, \ldots, K_i^{h_i}, \ldots, K_i^{H_i}\right]^T$	Candidates values for the parameter K_i
$\mathbf{d}_i^{ind} = [1, \ldots, h_i, \ldots, H_i]^T$	Indexes of each candidates values for the parameter K_i
$\mathbf{h}_q$(iter)	Vector of indexes associated with the wolf $\mathbf{x}_q^{\mathbf{k}}$(iter)
$\mathbf{h}_\alpha$, $\mathbf{h}_\beta$, $\mathbf{h}_\delta$	Vector of indexes associated with the leaders α, β and δ
$\mathbf{h}_{\rho 1}$, $\mathbf{h}_{\rho 1}$	Vector of indexes associated with the random wolves $\rho 1$ and $\rho 2$
l	Refer to the selected leader, either α, β, δ, $\rho 1$ or $\rho 2$
Δ	Displacement factor

Parameters solely used for Global Continuous GWO

y_i^l	Contribution of a leader l for the parameter K_i

Abstract

This thesis proposes a novel bio-inspired optimization method based on the GWO algorithm, with the purpose of solving mixed optimization problems, i.e. problems with both continuous and discrete variables. This novel method is named mixedGWO and is applied to 2 distinct problematics.

Firstly, the mixedGWO should permit to improve the quality of image classification by SVM. Indeed, a SVM accuracy will depend of its training parameters, and there is non empirical and non exhaustive method to define these parameters for a given classification problem. Therefore, the mixedGWO can be used as a solution to this parametring problem. The improve classification should allow the company IntuiSense to add the gender recognition feature to its audience measurement tool ISAM.

Secondly, the mixedGWO is used for joint denoising and unmixing of spectra in multi-spectral and hyper-spectral image processing. Indeed, the unmixing's quality is strongly dependent of the denoising quality : doing these 2 steps simultaneously permits a gain of time and a results' accuracy way better than if they are done one after the other.

Keywords : Audience Measurement, bio-inspired-optimisation, image processing, computer vision, multi-spectral images, image classification, face detection, face recognition

Résumé

Cette thèse propose une nouvelle méthode d'optimisation bio-inspirée basée sur le GWO avec pour but de pouvoir résoudre des problèmes d'optimisation dits mixtes, c'est-à-dire des problèmes composés de variables continues et discrètes. Cette nouvelle méthode baptisée mixedGWO est ensuite appliquée à 2 problématiques distincts.

Tout d'abord, le mixedGWO pourra permettre d'améliorer la qualité de la classification d'image par SVM. En effet, la fiabilité d'un SVM va dépendre de ses paramètres d'entraînement, et il n'existe pas de méthode non empirique et non exhaustive permettant de définir ces paramètres pour un problème de classification donné. Le mixedGWO se propose comme une solution à ce problème de paramétrage. La classification doit permettre à l'entreprise IntuiSense d'ajouter une brique de reconnaissance de genre à son outil de mesure d'audience ISAM.

Ensuite, le mixedGWO est employé pour faire du débruitage et du démêlage de spectres en simultanée sur des images multi-spectrales ou hyper-spectrales. En effet, la qualité du démêlage des spectres va être particulièrement dépendant de la qualité du débruitage de l'image : faire ces 2 étapes simultanément permet donc un gain de temps et une fiabilité des résultats bien plus intéressants que les faire l'une après l'autre.

Mots clefs : Mesure d'Audience, optimisation bio-inspirée, traitement d'images, vision par ordinateur, images multi-spectrales, classification d'images, détection faciale, reconnaissance faciale

Introduction

Audience Measurement is the act of measuring the number of people who are in an audience. This term is usually used for medias, in whichever form, and more recently for websites. The company IntuiSense, located at Gemenos in France, is specialized in vending machine interface and would like to adapt the concept of audience measurement on these vending machine using computer vision. That means a vending machine should be able, through a embedded camera, to detect its customers, count them and characterize them, $i.e.$ recognize a person gender or its age. Such a tool must be able to work with real-time performances and have the lowest computational cost as possible while conserving an acceptable accuracy. A classic and fast method used for gender recognition is to do image classification using a Support Vector Machine (SVM). However, a SVM's accuracy is strongly dependent of its input training parameters and there is no global method to determine the best training parameters for a certain classification problem. These parameters are usually continuous, except in the case of a polynomial kernel, where the polynomial degree is a discrete value.
In image processing, people usually work on images composed of the 3 main channels Red, Green and Blue (RGB), which corresponds approximatively to the wavelength of 690 nm, 550 nm and 450 nm respectively in the light spectrum. In multi-spectral and hyper-spectral image processing, images are composed of more than just the 3 basic channels. Moreover, they do not necessarily constraint themselves to the visible part of the light spectrum but can go further in the Ultra-Violet part or in the Infra-Red part. To perform spectra unmixing on an multi-spectral or hyper-spectral image, the target image must first be denoised. However, it should not be done randomly. Indeed, an image denoised too strongly will lose important information useful for the spectra unmixing while an image not denoised enough will contain too much information. Therefore, it would be better to perform the denoising and the spectra unmixing simultaneously in order to find the best balance between these two.
There is a common limit shared by the two application described previously : gender recognition, and more precisely image classification by SVM, and the joint denoising and unmixing of multi-spectral images. These two application would benefit from the addition of an optimization method to determine their best parameters.
An optimization problem consists of maximizing or minimizing a function by systematically

choosing input values from within an allowed set and computing the output value of the function. As the size of the said problems increase, classic optimization methods require more and more computation power. Therefore, new methods which are computationally efficient were born, such as bio-inspired optimization methods [16]. As their name implies, the bio-inspired optimization methods take inspiration from Mother Nature way of working. The most famous algorithm from the bio-inpsired family is the Genetic Algorithm (GA) [45], which take its inspiration from the Darwinian evolution theory. This thesis will emphasized on the swarm-type optimization method, whose main state-of-the-art method is the Particle Swarm Optimization (PSO) algorithm [56]. The swarm optimization algorithm will simulate the functioning of living organisms who work together. As example, the Ant Colony Optimization (ACO) method will simulate the way ants behave when they are looking for food [29] and the Grey Wolf Optimizer (GWO) method will simulate the hunting behaviour of a grey wolves pack [92]. However, as both applications' input parameters are composed of continuous and discrete value, the optimization method to apply should be designed for mixed problem. However, such a method does not exist as of now.

This manuscript holds two main purposes :

• to present an Audience Metrics tool working with computer vision and image processing and able to work at real-time speed with a low computational effort ;

• to propose a novel bio-inspired optimization method dedicated able to tackle fully continuous, fully discrete and mixed optimization problems.

Chapter 1 is an overview of the state-of-the-art in the different fields studied throughout this thesis. Firstly, object detection, and more particularly face detection, in image processing and computer vision is studied. Then, we will focus on object tracking in computer vision and video processing. Several aspects of object recognition through machine learning are then described. Finally, a broad overview of bio-inspired optimization method is done, with a clear differentiation between the methods designed for continuous space and those designed for discrete space.

Chapter 2 presents the theory of the proposed bio-inspired optimization method named mixedGWO, which is able to tackle fully continuous, fully discrete and mixed problems. The chapter will start by describing the continuous aspect of the algorithm, then the discrete aspect is studied and finally the mixed aspect is explained, along with comparison to state-of-the-art methods on several benchmark functions. This discrete part of this method has been published in [86] and the complete method first version is currently reviewed under minor revisions in [83].

Chapter 3 is focused on the applications developed during this thesis. Firstly the IntuiSense Audience Metrics (ISAM) tool developed with the company IntuiSense will be presented with quantitative and qualitative results, completing the publication [85]. Then, an extension to gender recognition improved by bio-inspired optimization is presented, published in [84]. Finally, the proposed mixedGWO method is applied to a multi-spectral image processing pro-

blematic : the joint denoising and unmixing of multi-spectral images. The result obtained and presented in this subsection are currently reviewed under minor revisions in [83].

CHAPTER 1

Bibliography study

1.1 Face detection in images

Object detection in image processing consist of having a computer automatically determining if a target object is present in an input image and if so showing where in this object is in the image. The most basic and natural way of performing this task would be by analyzing what composed the target object, what are its visual attributes. For example, in the case someone want to detect an orange fruit in an image, one way could be to detect the area in the image which share the same color as the fruit and then check among all the detected area which one are rounded.
Around the second half of the 90s, the same kind of methods have been firstly used for human face detection. In [48], a skin detection model combined with an elliptical shape detection is proposed while in [22] a skin detection model is combined with an features detection model in order to detect the shape of the nose, the eyes and the mouth. The problem of these methods reside mostly in the complexity of the target object : the human face. Indeed, the human face, contrarily to an orange fruit, is a deformable object as its shape change according to the face orientation [24]. Moreover, there are as many faces as there are people on earth, e.g. skin color, and even a same person appearance can quickly change just by opening his mouth or putting glasses on.
All this factors make the face detection topic a really challenging topic, which was splendidly tackled in 2001 by Paul Viola and Michael Jones with their reference detector [115]. Afterwards, several promising detection methods such as the Histogram of Gradient (HOG) detector or the SIFT and SURF descriptors were proposed in the literature.

1.1.1 Viola-Jones method

In 2001, Paul Viola and Michael Jones described in [115] a novel object detection method they applied to face detection, by using pseudo-Haar features, AdaBoost and an image representation named Integral image. This method, well-known as the Viola-Jones

method, has been a huge step forward in the automatic object detection domain and is considered as one of its cornerstone [123]. Moreover, it is able to work in almost real-time constraints and present a low complexity mainly depending of the input image size [55].

By searching through an image for a target object, the Viola-Jones detector wil determine where this object is or is not, multiple detection being possible. To do so, the target object will be described using pseudo-Haar features. Figure 1.1 showcases the two first pseudo-Haar features describing a human face.

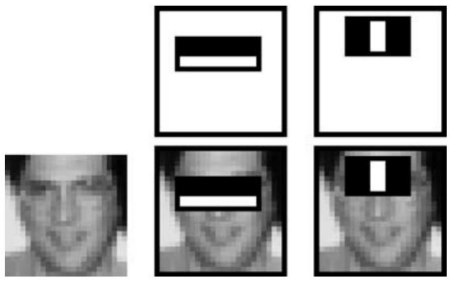


Figure 1.1 — First two Haar-Features for human faces's detection

The Viola-Jones detector is a cascade classifier which has been created using the AdaBoost method and has been enhanced by using Integral images [116]. Each stage of this cascade classifier are sub-classifiers consisting of an ensemble of pseudo-Haar features and their corresponding acceptance threshold. The workflow of a cascade classifier is illustrated on Figure 1.2.

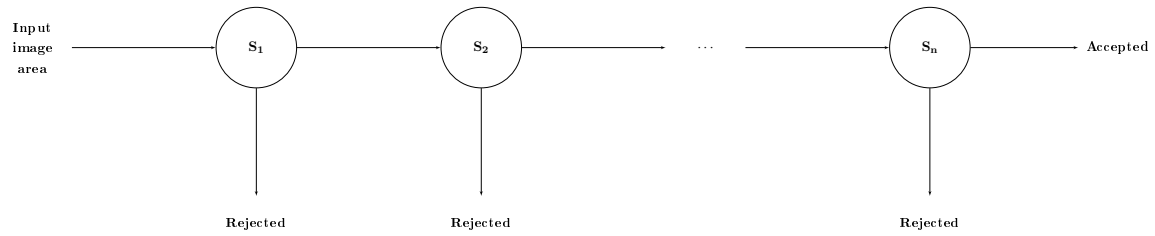


Figure 1.2 — Workflow of a Cascade Classifier

The first stages of this cascade classifier will consist of the simplest features, as the one showcased on Figure 1.1, in order to eliminate the quickest as possible the majority of the image area without the target object. On the other hand, the last stages will be the most complex ones. Only by passing through all of the stages will the studied area of an image be considered as the target object. Some example results of the Viola-Jones algorithm are displayed on Figure 1.3.

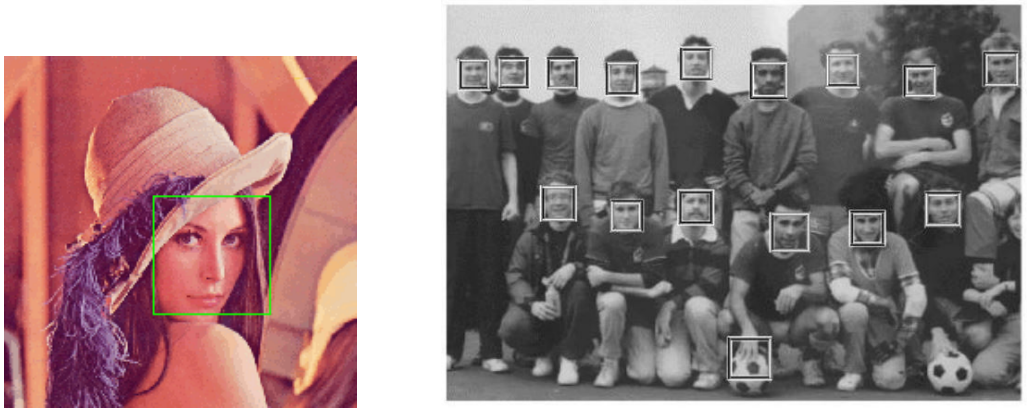


Figure 1.3 — Example results of the Viola-Jones algorithm

1.1.2 Histogram of Oriented Gradient (HOG) : Semi-local Approach

• **Histogram of Oriented Gradient (HOG) descriptors** are features widely used by the object detection and object recognition community. They have been shown to be distinctive and robust under small affine transformations and illumination changes. They are constructed by dividing the image into a dense grid of uniformly spaced cells and then computing the orientation histograms of the image gradient values on each cell. The illumination and contrast changes are taken into account by local normalization of the gradient strengths which requires grouping the cells together into larger, spatially-connected blocks.
The HOG descriptor is then the vector of the components of the normalized cell histograms for all the block regions. Dalal *et al.* [99] have proposed Histogram of Oriented Gradients in the case of human detection.

(a) (b)

Figure 1.4 — Example of HOG image, with : (a) Original Image and (b) HOG image

1.1.3 SIFT and SURF descriptors : Local Approach

• **The Scale Invariant Feature Transform (SIFT)** is a well known local descriptor created in 1999 by Lowe [78], allowing to detect and extract features which are invariant to rotation and scale and robust to some variations of illuminations, viewpoints and noise. The SIFT descriptor is computed in four steps. The two first stages correspond to the choice of keypoints, first identifying potential interest points that are scale and rotation invariant and then rejecting the ones that have low contrast and stability. The two last stages correspond to the descriptor vector computation, assigning one or more orientations to each selected keypoint based on local image gradient directions and using a 4×4 location Cartesian grid to compute the gradient on each location bin on the patch around the keypoint.
The SIFT descriptor gives good results in the case of object recognition when it can find relevant keypoints. It has been used by Wang *et al.* [117] for hand posture recognition with the objective of human-robot interaction.

• **Speeded Up Robust Feature (SURF)** was first presented by Bay *et al* in 2006 [11]. Partly inspired by the SIFT descriptor, SURF also consists in interesting points localization followed by feature descriptors computation. In both cases, the output is a representation of the neighborhood around an interest point as a descriptor vector. SURF is based on the distribution of first order Haar wavelet responses [43]. One of the principal advantages of SURF is to be several times faster than SIFT while having more discriminative power. It uses the integral images to simplify and to accelerate the computations. Yielding a lower dimensional feature descriptor, it reduces the time for feature computation and matching.

1.2 Object tracking in images

Object tracking can be regarded as the process of estimating the state, e.g. the position or the scale, along the time of a moving object in videos or image sequences [125]. It is an important problem in computer vision and has a wide range of applications, such as visual surveillance, robot navigation or autonomous driving [120].
In a object tracking problem, knowing the physical state, $i.e.$ color or shape, of the object to track beforehand can be useful. In [59], the colorimetry information of the target object are used , while in [60] prior shape information are used. However, information about the target object physical state is not necessarily mandatory. Indeed, the tracking method can content itself solely with the initial position of the object to track. It can either be given manually in controlled cases or done with an object detection method such as the ones presented previously in section 1.1. As an example, the Kalman filters [52], which will be presented further in subsection 1.2.1, need the position of the object to track at the start and

still need it to be given during a short number of iterations in order to adapt correctly its tracking model. In the same way, the Lucas-Kanade Optical Flow, proposed in [81] and detailed in subsection 1.2.2, need the position of a set of points to track from one image to another.

1.2.1 Kalman Filters

The Kalman filters is a tracking method in video processing which has been proposed in [52]. The Kalman filters is a recursive estimator modeled on a Markov chain and whose purpose is to estimate the state of a linear system where the state is assumed to be distributed by a Gaussian [110].

By applying a Kalman filter, it is assumed that the studied system follows a linear rule where its true state x_k at time k is given by Eq. (1.1) :

$$x_k = \mathbf{F_k} x_{k-1} + \mathbf{B_k} \mathbf{u_k} + w_{k-1} \tag{1.1}$$

with $\mathbf{F_k}$ the state transition model, $\mathbf{B_k}$ the control-input model, $\mathbf{u_k}$ the control vector and w_{k-1} a random value representing the process' noise.
The true state's measurement is given by Eq. (1.2) :

$$z_k = \mathbf{H_k} x_k + vk \tag{1.2}$$

with $\mathbf{H_k}$ the observation model and v_k a random value representing the measurement noise. The random noise values w_k and v_k are assumed to be independent of each other and to follow a normal probability distribution.

As shown on Figure 1.5, a Kalman filter is composed of 2 steps which continually works alternatively : the Prediction step and the Correction (or Update) step. The state of a Kalman filter is represented by two values : $\hat{\mathbf{x}}_{k|k}$ the posteriori state's estimation until time k and $\mathbf{P}_{k|k}$ the posteriori error covariance matrix.

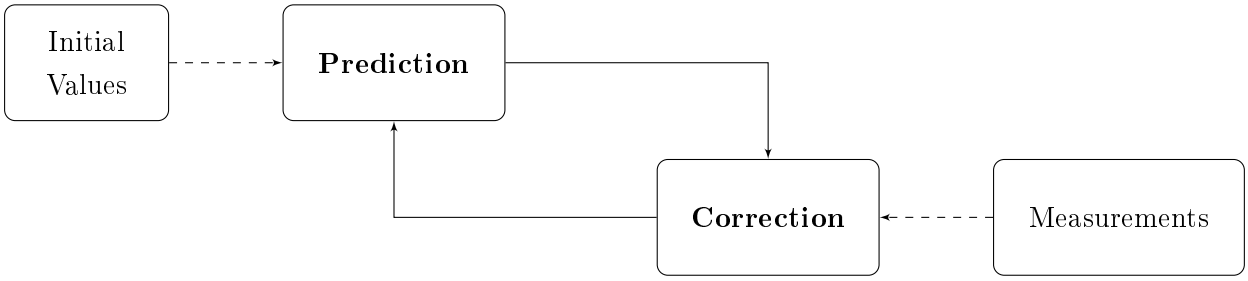


Figure 1.5 — The Kalman Filters' functioning

At the beginning of the studied time frame or studied video ($k = 0$), the initial values, *i.e.* the target object's position, are fed to the prediction step.
At each step k, the Prediction step will start the process by trying to predict the object's position using Eq. (1.3) :

$$\begin{cases} \hat{\mathbf{x}}_{\mathbf{k|k-1}} = \mathbf{F_k}\hat{\mathbf{x}}_{\mathbf{k-1|k-1}} + \mathbf{B_k}\mathbf{u_k} \\ \mathbf{P_{k|k-1}} = \mathbf{F_k}\mathbf{P_{k-1|k-1}}{\mathbf{F_k}}^T + \mathbf{Q_k} \end{cases} \tag{1.3}$$

with $\mathbf{Q_k}$ the process noise's covariance matrix.
Afterwards, the Correction step will use the information given by measurements z_k of the studied system and the Prediction step in order to update the model in order for further predictions to be closer to the true state of the studied system.
The Correction step will compute the innovation $\overline{\mathbf{y}}_\mathbf{k}$, the innovation covariance $\mathbf{S_k}$ and the optimal Kalman gain $\mathbf{K_k}$ using Eq. (1.4) :

$$\begin{cases} \overline{\mathbf{y}}_\mathbf{k} = z_k - \mathbf{H_k}\hat{\mathbf{x}}_{\mathbf{k|k-1}} \\ \mathbf{S_k} = \mathbf{R_k} + \mathbf{H_k}\mathbf{P_{k|k-1}}{\mathbf{H_k}}^T \\ \mathbf{K_k} = \mathbf{P_{k|k-1}}\mathbf{H_k^T}\mathbf{S_k^{-1}} \end{cases} \tag{1.4}$$

with $\mathbf{R_k}$ the observation noise's covariance matrix.
The Correction step then compute the new values $\hat{\mathbf{x}}_{k|k}$ and $\mathbf{P}_{k|k}$ using the Eq. (1.5) :

$$\begin{cases} \hat{\mathbf{x}}_{\mathbf{k|k}} = \hat{\mathbf{x}}_{\mathbf{k|k-1}} + \mathbf{K_k}\overline{\mathbf{y}}_\mathbf{k} \\ \mathbf{P_{k|k}} = (\mathbf{I} - \mathbf{K_k}\mathbf{H_k})\mathbf{P_{k|k-1}} \end{cases} \tag{1.5}$$

1.2.2 Optical Flow

Optical flow is the pattern of motion, as it appears to a camera, of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer (an eye or a camera) and the scene. The concept of optical flow was introduced by the American psychologist James J. Gibson in 1950 to describe the visual stimulus provided to animals

moving through the world [39]. The optical flow may often want to assess motion between two frames (or a sequence of frames) without any other prior knowledge about the content of those frames. Typically, the motion itself is what indicates that something interesting is going on. Movement, characterized by optical flow, has been exploited by roboticists, who use optical flow techniques for image processing and control of navigation [12, 3].

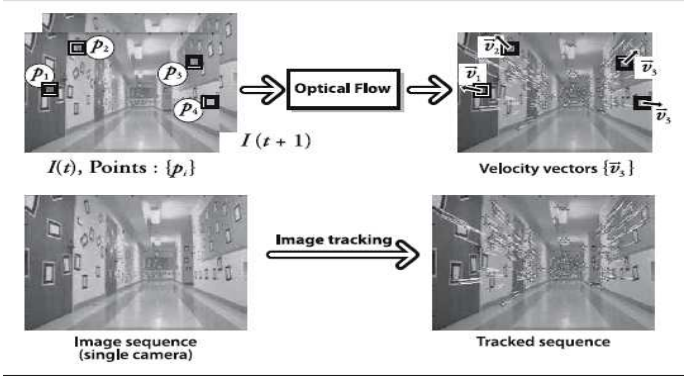


Figure 1.6 — Optical Flow

As already mentioned, optical flow may often want to assess motion between two frames (or a sequence of frames) without any other prior knowledge about the content of those frames. A result that can be obtained by optical flow is illustrated in Figure 1.6.

The principles of optical flow are as follows : if color images are considered, a conversion to one channel is done. For instance, we can select the Cr component of the $YCbCr$ representation, but this is valid only when white hand are considred. We also can retain only the luminance component from the HSL (Hue, Saturation, Lightness) representation of the RGB image. We can associate some kind of velocity with each pixel in the frame or, equivalently, some displacement that represents the distance a pixel has moved between the previous frame and the current frame. It associates a velocity with every pixel in an image. There are two approaches to calculate the optical flow.

The first approach is the dense technique which tries to match large windows around each pixel of an image to another, as the algorithm of Horn and Schunk [47]. This algorithm was developed in 1981 ; it puts aside the hypothesis of constancy of brightness by minimizing the regularized Laplacian of optical flow velocity components. This turns as a valid one the hypothesis of smoothness constraint on the velocities. Also, there exists a whole class of similar algorithms in which the image is divided into small regions called blocks [12, 50].

These blocks are generally square and may overlap. These algorithms attempt to divide

the two previous and current images in blocks and then calculate the movement of these blocks. Such algorithms are of great interest in many video compression techniques and in computer vision. Black and Anadan have created dense optical flow techniques [17, 18] that are often used in movie production, where, for the sake of visual quality, the movie studio is willing to study in detail the flow information, in practice the movement of the actors or objects. The block-matching algorithms operate on aggregates of pixels, not on individual pixels.

If the overlap between blocks is very important, the returned images of "flow" are usually of a lower resolution than the input images. Algorithms of this approach have superior quality but are slow and cannot be applied in real time and cannot resolve the case of large displacements. In practice, calculating dense optical flow is not easy. Let's consider the motion of a white sheet of paper. Many of the white pixels in the previous frame will simply remain white in the next. Only the edges may change, and even then only those orthogonal to the direction of motion. Hence the idea of creating a sparse optical flow, developed originally in [81].
The second approach is a popular sparse tracking technique, Lucas-Kanade (LK) optical flow. This version of optical flow relies on some means of specifying beforehand the subset of points that are to be tracked. If these points have certain desirable properties, such as the "corners", then the tracking will be relatively robust and reliable. The LK algorithm [81], as originally proposed in 1981, was an attempt to produce dense results. However, because the method is easily applied to a subset of the points in the input image, it has become an important sparse technique. The LK algorithm can be applied in a sparse context because it relies only on local information that is derived from some small windows surrounding each of the points of interest. This contrasts with the intrinsically global nature of the Horn and Schunck algorithm.

The basic idea of the Lucas-Kanade algorithm is based on three assumptions (see Fig. 1.7) :
• *Brightness constancy* : A pixel from the image of an object in the scene does not change in appearance as it (possibly) moves from frame to frame. For grayscale images (LK can also be done in color), this means we assume that the grey level of a pixel does not change as it is tracked from frame to frame.
• *Temporal persistence or "small movements"* : The image motion of a surface patch changes slowly in time. In practice, this means the temporal increments are fast enough, relative to the scale of motion in the video sequence, to prevent the object from moving much from frame to frame.
• *Spatial coherence* : Neighboring points in a scene belong to the same surface, have similar motion, and project to nearby points on the image plane.

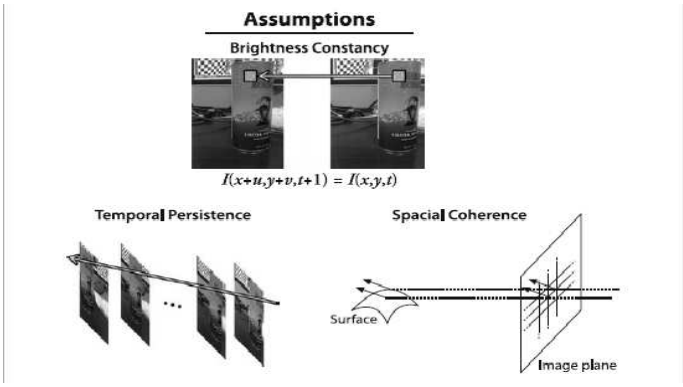


Figure 1.7 — Assumptions behind Lucas-Kanade optical flow

As mentioned above, the disadvantage of using small local windows in Lucas-Kanade approch is that large motions can move points outside of the local window and thus become impossible for the algorithm to find. Indeed large and non-coherent motions are often observed in practice. The key idea in the Lucas-Kanade approach is to avoid this problem, by tracking first over larger spatial scales, by using an image pyramid and then by refining the initial motion velocity assumptions by working its way down the levels of the image pyramid until it arrives at the raw image pixels.
Hence, this problem led to the development of the "pyramidal" LK algorithm, which tracks an object starting from the highest level of an image pyramid (lower detail resolution) and working down to lower levels (finer detail resolution). Thus we minimize the violations of our motion assumptions and we can track faster and longer motions. This more elaborated function is known as "pyramidal Lucas-Kanade" optical flow and is illustrated in Figure 1.8. Hence, tracking along the resolution levels as downhill along pyramids allows large motions to be characterized by local windows.
In [51], a forward-backward error is introduced to the classical optical flow in order to increase its precision. This type of optical flow is called the Median Flow.

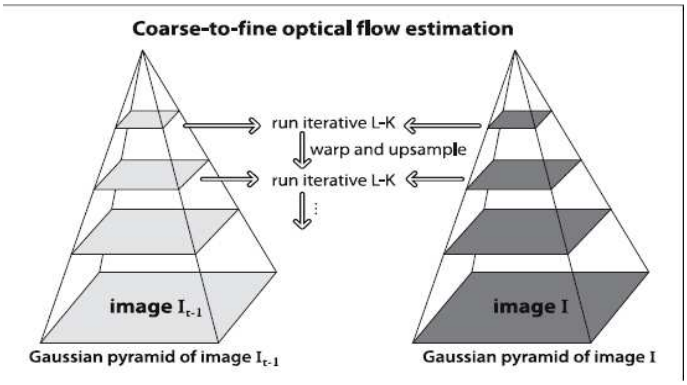


Figure 1.8 — Pyramid Lucas-Kanade optical flow

1.3 Object recognition in image processing

Object recognition in image processing and computer vision is being able to automatically label an object's image correctly, e.g. label a face as a woman's face or a man's face. To perform such feat, image classification is necessary. Image classification is usually done with machine learning algorithms such as Conventional Neural Networks (CNN) [65] or Support Vector Machines (SVM) [69]. Object recognition can also be done in a simpler way by using classic distances (e.g. euclidian or mahalanobis) [19] or by using a Nearest-Neighbours classifier [2].

To perform image classification, training images are needed, *i.e.* a set of images representing the object that the classification algorithm will use in order to "learn" the object. The number of training images used is an important factor for object recognition, indeed a sufficient number of images will enable the recognition system to achieve a good recognition rate while a low number of images will decrease it [36]. Several databases have been created for the problems about face recognition and characterization, such as the FERET database [103], the YALE Face database [37], the ORL Database of Faces [108], the Labeled Face in the Wild database [49] and a lot more.
Moreover, an object recognition system will not work from an image as whole : it needs to extract the object's representation, *i.e.* useful information from the images whether they are the training images or the input images [109]. A classic approach would be to apply a statistic method such as a Principal Component Analysis [113] or a Linear Discriminant Analysis [34] to the images. Another approach is to use descriptors describing either the texture or the shape of the studied object. Among the most used descriptors in face recognition and characterization we can cite the Local Binary Patterns (LBP) [2], the Histogram of oriented Gradients (HOG) [74] and the SIFT descriptors [35]. In gender recognition, in order to increase the recognition system's performances, these features have been combined with Color Histograms (CH) or Gabor features [109].

1.3.1 Object recognition by Distances

Easy to compute and of low complexity, distances are metrics used to compare two distributions of the same weights. Therefore, it could be used for object recognition problems.
In [19], the authors aimed to classify hands images, and more precisely to recognize a hand's shape using computer vision and image processing, *i.e.* detect the number of fingers shown by the hand. To classify these hands, they used a contour detection tool and compute its signature components matrix $\mathbf{Z}$ using a signal generation process, as shown on Figure 1.9.

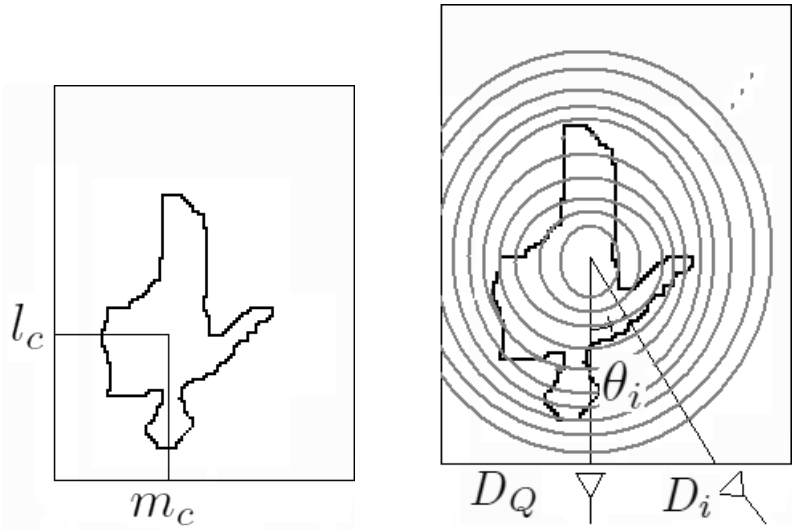


Figure 1.9 — (a) Image and edge model; (b) signal generation process

The signature matrix $\mathbf{Z}$ is then vectorized into a vector $\mathbf{x}$. For each image, their vector $\mathbf{x}$ is computed and they are then classified by minimizing the Mahalanobis distance applied to a compressed vector of data $\mathbf{U}_h^T\mathbf{x}$, obtained with PCA, using Eq. (1.6) :

$$\mathcal{D}_m = (\mathbf{U}_h^T\mathbf{x} - \boldsymbol{\mu}_h)^T(\boldsymbol{\Lambda}_h + \epsilon)^{-1}(\mathbf{U}_h^T\mathbf{x} - \boldsymbol{\mu}_h) \tag{1.6}$$

with $\boldsymbol{\mu}_h$ the mean invariant vector, $\boldsymbol{\Lambda}_h$ the covariant matrix for each class h, and ϵ a regularization constant.
Still in [19], the authors also proposed to use a classic Euclidian distance instead of Mahalanobis, using Eq. (1.7) :

$$||\mathbf{U}_h^T\mathbf{x} - \boldsymbol{\mu}_h|| \tag{1.7}$$

Another object recognition technique using distances is the k-Nearest Neighbors (k-NN) classifiers [4]. When giving an object to label to such a classifier, its distances relative to the known objects are computed. The k shortest distances, $i.e.$ the k nearest neighbors of the input object, permit to determine to which class the input object belongs.
In [2], a k-NN classification is used to classify histograms of LBP images for face recognition. Several distances are proposed to computed this k-NN classifier :

• Histogram intersection :

$$D(S,M) = \sum_i \min(S_i, M_i) \tag{1.8}$$

• Log-likehood statistic :

$$L(S,M) = \sum_i S_i \log M_i \tag{1.9}$$

• Chi square statistic (χ^2) :

$$\chi^2(S,M) = \sum_i \frac{(S_i - M_i)^2}{S_i + M_i} \tag{1.10}$$

• Weighted χ^2 statistic :

$$\chi^2_\omega(S,M) = \sum_i \omega_j \frac{(S_{i,j} - M_{i,j})^2}{S_{i,j} + M_{i,j}} \tag{1.11}$$

where S and M are computed form the input histograms.
In [67], the authors achieved a good face recognition rate higher than 95% by combining the linear transformation techniques FLDA with the Earth Mover's Distance (EMD). The EMD compares two distributions A and B using the following Eq. (1.12) :

$$EMD(A,B) = \min_{f \in F} EMD(A, f(B)) \tag{1.12}$$

1.3.2 Conventional Neural Network

Proposed in 1990 by Lecun et al. [65], the Conventional Neural Network (CNN) is arguably the most famous deep learning method. In its first version, it was used to classify and recognize handwritten digits in an image [65]. A basic CNN structure for gender recognition is shown on Figure 1.10.

A CNN is composed of 3 kinds of layers :
• *The input layer* : It contains basic information about the studied object, such as the pixel values of the input image,
• *the output layer* : It is the final layer of the CNN, it is the layer that take the final decision of the CNN, *e.g.* if the input image corresponds to a man's face or a woman's face,
• *The hidden layer* : All the layers between the input layer and the output layer. They will compute correlations between all the informations given in the input layer and correlate these correlations until the output layer is able to take a decision.

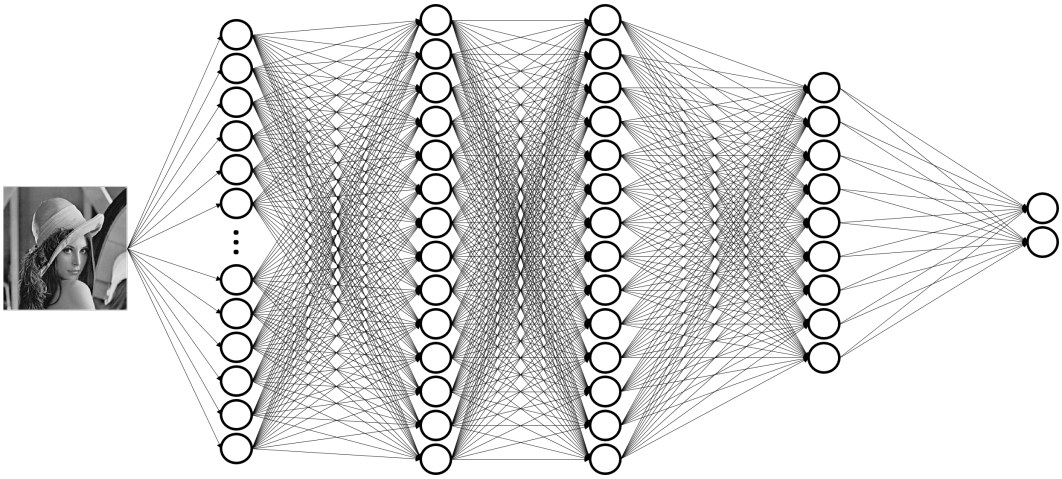


Figure 1.10 — Structure of a CNN

In the output layer o, the goal is to have, for each node k, an activation value a_k^o with $0 \leq a_k^o \leq 1$. The output layer's node with the highest value is considered as the result given by the CNN, *e.g.* in the basic case presented in Figure 1.10, the first node in the output layer corresponds to the answer *woman's face* while the second node corresponds to the answer *man's face*.

The CNN will use the hidden layers in order to compute activation values which can be seen as correlations between the informations given in the input layer. Each node k of the layer l, which is either from a hidden layer or from the output layer, will compute its activation value a following the Eq. (1.13) :

$$a_k^l = A({\omega_k^l}^T \cdot \mathbf{x}^{l-1} + b_k^l) \tag{1.13}$$

with ω_k^l the weight vector, b_k^l the bias term and $\mathbf{x}^{l-1}$ the activation values of the previous layer, if it is a hidden layer, or the information value if the previous layer is the input layer. $A(\cdot)$ is the nonlinear activation function used to scale the activation value in the correct range. This function is usually a sigmoid function [42], a tanh function [66] or a ReLU function [64].

1.3.3 Support Vector Machine

A support vector machine (SVM) is a concept in statistics and computer science for a set of related supervised learning methods that analyze data and recognize patterns. When used for classification, SVM seeks to find an optimal hyperplane as a decision function in high-

dimensional space [114, 101]. For a two-class pattern recognition problem, if we suppose that the classes are linearly separable, SVM selects one linear decision boundary which minimizes the generalization error. The selected decision boundary is represented by a hyperplane in the feature space and it leaves the greatest margin between the two classes. Here, margin is defined as the sum of the distances from the closest cases of the two classes to hyperplane.

In case of the simplest scenario, that is to say the classes are linearly separable, the training data set comprises l cases : $\{\mathbf{x}_i, y_i\}, i = 1, \cdots, l$, where $\mathbf{x} \in \mathbb{R}^N$ is an N-dimensional space and $y \in \{-1, +1\}$ is the class label. The training patterns are linearly separable if there exists a vector $\mathbf{w}$ and a scalar b satisfying

$$y_i(\mathbf{w}^T \cdot \mathbf{x}_i) - b \geq 0$$

where $\mathbf{w}$ defines the orientation of a discriminating plane and b determines the offset of the discriminating plane from the origin. The hypothesis space can be defined by the set of functions given by

$$f_{\mathbf{w}^T,b} = \text{sign}((\mathbf{w}^T \cdot \mathbf{x}_i) + b). \tag{1.14}$$

SVM finds the separating hyperplanes for which the distance between classes is maximized by solving the constrained optimization problem :

$$\min_{\mathbf{w}} \frac{1}{2} \parallel \mathbf{w} \parallel^2 .$$

For linearly nonseparable classes, SVM seeks to find the hyperplane that maximizes the margin and minimizes a quantity proportional to the number of misclassification errors by introducing a relaxation variable $\varsigma_i \geq 0$. Thus, for nonseparable data the constrained optimization problem is :

$$\min_{\mathbf{w},\varsigma_1,\cdots,\varsigma_l} \left[\frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_{i=1}^{l} \varsigma_i \right] \tag{1.15}$$

where $0 < C < \infty$ is a positive constant to trade off between margin and misclassification error.

For nonlinear decision surfaces, a feature vector $\mathbf{x} \in \mathbb{R}^N$ is mapped into a higher dimensional Euclidean space (feature space) $\mathcal{F}$ via a nonlinear function $\varphi : \mathbb{R}^N \mapsto \mathcal{F}$ [101]. The optimal margin problem in $\mathcal{F}$ can be written by replacing $\mathbf{x}_i^T$ with $\varphi(\mathbf{x}_i)^T$. To address this problem, Vapnik [114] suggested a way to create nonlinear SVM by applying the kernel function to maximum-margin hyperplanes. A kernel function $\mathcal{K}$ is defined as

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^T \cdot \varphi(\mathbf{x}_j) \tag{1.16}$$

and with the kernel function above, Eq. (1.14) becomes

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i \in \mathbb{S}} \zeta_i y_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b \right) \tag{1.17}$$

where ζ_i is a nonzero Lagrange multiplier, $\mathbb{S}$ is a subset of indices $\{1, 2, \cdots, l\}$ corresponding to ζ_i and defines the so-called support vectors (SVs). A detailed discussion of the computational aspects of SVM can be found in [114] and [101], with many examples also in the remote sensing literature [6, 88].

By introducing the kernel, SVM gains flexibility in the choice of the form of the threshold (hyperplane) separating solvent from insolvent classes, which needs not be linear and even needs not have the same functional form for all data. If the parameters are appropriately chosen, for instance parameter C, SVM can be robust even when the training sample has some bias, *i.e.* it is robust to noise. An SVM delivers a unique solution, since the optimality problem is convex [7]. Two other types of kernels are also mainly used for a SVM : the gaussian (RBF) kernel and the polynomial kernel whose functions are Eq. (1.18) and Eq. (1.19 respectively).

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{||\varphi(\mathbf{x}_i) - \varphi(\mathbf{x}_j)||^2}{2\gamma^2}\right) \tag{1.18}$$

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \left(\varphi(\mathbf{x}_i)^T \cdot \varphi(\mathbf{x}_j) + 1\right)^d \tag{1.19}$$

One remaining problem in SVM is the high algorithmic complexity and extensive memory requirements of the required quadratic programming in large-scale tasks. Another issue is the optimal choice of the kernel. Indeed it has great influence on the effectiveness of SVM classification [20].

1.4 Bio-inspired Optimization

An optimization problem consists of maximizing or minimizing a real fitness function by systematically choosing input values from within an allowed set and computing the value of the function. As the size of the said problems increase, classic optimization methods require more and more computation power. Therefore, bio-inspired optimization methods were born, as they are computationally efficient [16].
In subsection 1.4.1, bio-inspired optimization methods for continuous problems will be studied, with a highlight on the swarm-based methods Particle Swarm Optimization (PSO) method, the Grey Wolf Optimizer (GWO), the Tree-Seed Algorithm (TSA) and the Artificial Bee Colony (ABC) method.
Finally, subsection 1.4.2 will be about the bio-inspired optimization method for discrete problems, with a highlight of the Ant Colony Optimization (ACO) method and the Multi-Objective Discrete Grey Wolf Optimizer (MODGWO).

1.4.1 Bio-inspired Optimization in continuous space

To solve an optimization problem, an interesting field of applied mathematics has attracted much interest during the past few years : Meta-heuristics. These methods exhibit a good capacity in solving computationally expensive numerical problems, with limited function evaluations [8]. Meta-heuristics may be classified into three main classes : physics-based, evolutionary, and swarm intelligence (SI) algorithms. Physics-based algorithms include for instance gravitational search algorithm [105, 91, 89], where the search agents are provided with a mass which depends on their fitness, and wind driven optimization (WDO) [15]. Simulated annealing [77] can also be seen as a physics-based algorithm. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. This notion of slow cooling implemented in the simulated annealing algorithm is interpreted as a slow decrease in the probability of accepting worse solutions as the solution space is explored. Evolutionary algorithms [45], together with swarm intelligence algorithms [56, 31, 92, 95], compose the bio-inspired optimization algorithms. They differ for instance in their way to encode the agents which search the space composed by the allowed sets of values. The convergence success of a bio-inspired optimization algorithm depends on directing and balancing of its so-called 'exploration' and 'exploitation' abilities [124]. Among evolutionary algorithms, the most popular are the genetic algorithms (GA). This algorithm was proposed by Holland in 1992 [45] and simulates Darwinian evolution concepts. An often cited, now well-known reference [94] introduces genetic algorithms in the context of evolutionary computation which implies the evolution of a population which is inspired by Darwin's natural selection theory. Another largely cited reference presents basics about genetics, the hierarchical genetic algorithm, and applications to H∞ control, neural network, and speech recognition [82]. The seminal work about swarm intelligence is particle swarm optimization (PSO) [56, 31], proposed by Kennedy and Eberhart, who got inspired by [106] where the term 'particle swarm' was chosen to define the members of a population or test set. In the PSO paradigm, the population members are mass-less and volume-less. Their evolution is described through position, speed, and acceleration parameters. The concept of swarm got first inspired by the behavior of birds, where one leader guides the flock. In [92], a method involving three leaders was proposed, which gets inspired by the behavior of wolves, namely grey wolf optimization : the GWO algorithm mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. Four types of grey wolves such as alpha, beta, delta, and omega are employed to simulate the leadership hierarchy. In addition, the three main steps of hunting, searching for prey, encircling prey, and attacking prey, are implemented.
A number of variants are also proposed to improve the performance of vanilla GWO that include a hybrid version of GWO with PSO [53], and a 'modified GWO' (mGWO) [95] which includes a slightly modified update rule for the wolves. In [93] grey wolf optimization is

adapted for multi-objective optimization problems. In [62] a 'chaotic' GWO is proposed which aims at accelerating the global convergence speed of GWO. Compared to PSO and other prevailing techniques, GWO has the ability to converge to a better quality near-optimal solution and possesses better convergence characteristics [53]. Also, GWO has a good balance between exploration and exploitation that results in high local optima avoidance [90]. It has been successfully applied, as mentioned in [95], for solving economic dispatch problems, optimal design of double layer grids, time forecasting, flow shop scheduling problem, optimal power flow problem, and optimizing key values in the cryptography algorithms. Some other variants of the GWO algorithm can be found in [124, 104, 44] and in the references inside.

Within bio-inspired optimization algorithms, Artificial Bee Colony (ABC) [54] and Tree Seed Algorithm (TSA) [61] exhibit a common property : they rule the displacement of the search agents through the choice of a random leader, selected among the whole population of search agents.
In the ABC algorithm [54], the colony of artificial bees contains three groups of bees : employed bees, onlookers and scouts. A bee waiting on the dance area for making decision to choose a food source is called an onlooker and a bee going to the food source visited by itself previously is named an employed bee. A bee carrying out random search is called a scout. In the original ABC algorithm, one half of the bees are employed bees, and the other half constitutes the onlookers. In order to update the position of the current employed bee, one of the other employed bees is selected at random. As the difference between the location of the current and the random bee decreases, the displacement of the current bee decreases too. Thus, as the search approaches to the optimum solution in the search space, the displacement magnitude is adaptively reduced : exploration is preferred at the beginning of the algorithm, and exploitation is preferred at the end of the algorithm.
In TSA [61], a new seed location is produced through one of the two principal equations of the algorithm (see [61, 9] for details). In the first equation, the global best position obtained so far rules the creation of a new seed. In the second equation, a tree selected at random among all but the current tree rules the creation of a new seed. The decision of choosing either the first equation or the second one is made through the 'search tendency' parameter (ST). The higher ST, the more probable is the choice for the first equation. The smaller ST, the more probable is the choice for the second equation. A high value of ST yields a powerful local search and a high convergence speed, whereas a low value of ST causes slow convergence but powerful global search. In other words, the exploration and exploitation capabilities of the TSA are controlled by ST parameter. We notice that this capability is set *a priori* for an entire run of the algorithm.
A common and interesting property of ABC and TSA is that they base the evolution of the population of search agents on the selection of a random agent. However, they can only

handle continuous search spaces.

Mainly, the rest of the swarm intelligence techniques which mimic the behavior of groups of animals are as follows (see [92] and references inside) : Marriage in Honey Bees Optimization Algorithm (MBO) in 2001, Artificial Fish-Swarm Algorithm (AFSA) in 2003, Termite Algorithm in 2005, Wasp Swarm Algorithm in 2007, Monkey Search in 2007, Bee Collecting Pollen Algorithm (BCPA) in 2008, Dolphin Partner Optimization (DPO) and Cuckoo search (CS) [122] in 2009, Firefly Algorithm (FA) in 2010, Bird Mating Optimizer (BMO) in 2012, Krill Herd (KH) in 2012, Fruit fly Optimization Algorithm (FOA) in 2012, Glowworm Swarm Optimization [68] in 2016, artificial algae algorithm (AAA) [10] in 2018.

We now focus on methods which are based on the division of the search agents into 'groups' also called 'tribes'. Tribes PSO has first been proposed in [26] and further developed in [23]. Its performance has been analyzed in [27]. The principles of tribes PSO is to reduce the number of parameters in PSO, through an adaptive process which modifies autonomously the number of particles. In [121] a grouped grey wolf optimizer is proposes which is a 'tribe version' of GWO : the grey wolves are divided into two groups, including a cooperative hunting group with four types of wolves and a random scout group with two types of wolves. Contrary to the tribes in [23], these two tribes of wolves are not defined symmetrically. The first group (hunters) is meant for both 'exploration' in a first phase of the algorithm, and then 'exploitation', that is, concentration around the global minimum when it is assumed to be found. The second group is meant for exploration, to randomly look for a potential prey.
In summary, as evoked by the "No Free Lunch" theorem [118], the diversity of meta-heuristic algorithms comes from the fact that general applicability comes at the cost of domain specific performance, and that there is no one best approach for all domains. Furthermore, there are no guarantees of finding a globally optimal solution, even within a specified tolerance.

1.4.1.1 Highlights of the Particle Swarm Optimization

The basic PSO algorithm [56] is inspired by the behavior of fish or bird swarms. The basic principles of PSO algorithm is to simulate the communication between animals of a swarm, which aim at locating food for instance. PSO updates the behavior of such individuals of the swarm, called particles, through iterations. As shown in Algorithm 1, it is implemented as an iterative algorithm which, for the current iteration number iter, computes two characteristics of the particles : velocity, and position.

Here are some details for PSO algorithm :

At each iteration iter, $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$ is the current position of particle q, and $\mathbf{v}_q^{\mathbf{k}}(\mathrm{iter})$ is its

Algorithm 1 Pseudo-code : Particle Swarm Optimization for multiple parameter estimation

Inputs : fitness function, small factor ϵ set by the user, to stop the algorithm, lower and upper bounds for each parameter.

1. Set iteration number iter $= 1$, create an initial population composed of Q random particles with all required parameter values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$. This initial population takes the form of a matrix with Q rows and N columns.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each particle $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Update the local best particles $\mathbf{p}_q^{\mathbf{k}}$ $(q = 1, \ldots, Q)$, and the global best particle $\mathbf{g}^{\mathbf{k}}$,
4. Repeat steps for each particle q, $q = 1, \ldots, Q$:
   (a) Compute displacement also called velocity $\mathbf{v}_q^{\mathbf{k}}(\text{iter} + 1)$
   (b) Compute position $\mathbf{x}_q^{\mathbf{k}}(\text{iter} + 1)$
5. Exchange current population from new one, obtained at step 4.
6. If iter $< \mathrm{T}_{\max}$ or $||\mathbf{x}_q^{\mathbf{k}}(\text{iter} + 1) - \mathbf{x}_q^{\mathbf{k}}(\text{iter})|| > \epsilon$, increase iter, and go to step 2.

Output : estimated parameter values $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$

velocity.

At step 1, the initial population is stored in a $Q \times N$ matrix. Each row of the matrix is a $1 \times N$ vector with three random values. The n^{th} coefficient of this vector is a random value in the interval of authorized values for the corresponding parameter. Therefore, at iteration 1, the q^{th} row of the matrix is $\mathbf{x}_q^{\mathbf{k}}(1)$.

In the following, we exemplify the step 4 of Algorithm 1, with one particle with index q. We still assume without loss of generality that N unknowns $K_1, K_2, \ldots, K_N$ are expected. This particle is expressed as $\mathbf{x}_q^{\mathbf{k}}(\text{iter}) = [K_1, K_2, \ldots, K_N]^T$. At step 4a, the velocity and the updated position of the particle q, at iteration iter $+ 1$ are computed through Eqs. (1.20) and (1.21) :

$$\mathbf{v}_q^{\mathbf{k}}(\text{iter} + 1) = W\ \mathbf{v}_q^{\mathbf{k}}(\text{iter}) + \gamma_{1q}\ r_{1q}(\mathbf{p}_q^{\mathbf{k}} - \mathbf{x}_q^{\mathbf{k}}(\text{iter}))) + \gamma_{2q}\ r_{2q}(\mathbf{g}^{\mathbf{k}} - \mathbf{x}_q^{\mathbf{k}}(\text{iter}))) \tag{1.20}$$

At step 4b, the position at iteration $iter + 1$ is computed as the summation of the position at iteration $iter$ and the velocity at iteration $iter$.

$$\mathbf{x}_q^{\mathbf{k}}(\text{iter} + 1) = \mathbf{x}_q^{\mathbf{k}}(\text{iter}) + \mathbf{v}_q^{\mathbf{k}}(\text{iter} + 1) \tag{1.21}$$

In Eqs. (1.20) and (1.21), the arithmetic operations are computed component-wise. For a given particle q, the velocity is influenced by two contributions : the cognitive one and the social one. The cognitive contribution is due to personal best location of the particle over all iterations. It is denoted by $\mathbf{p}_q^{\mathbf{k}}$ in Eq. (1.20). Social contribution represents global best location over all particles in the swarm and all iterations. It is denoted by $\mathbf{g}^{\mathbf{k}}$ in Eq. (1.20).
In Eq. (1.20), W is the inertia weight, γ_{1q} and γ_{2q} are the acceleration constants encouraging a local and a global search respectively.

1.4.1.2 Highlights of the Grey Wolf Optimization

Grey wolf optimizer (GWO) mimics the behavior of a herd of wolves [92]. Four types of grey wolves are employed to simulate the leadership hierarchy : three leaders α, β, δ, and the ω wolves. Searching and attacking prey are implemented. They account for exploration or global search, and exploitation or local search respectively. The GWO methods assumes first that the position of the prey, that is, the global minimum, is known, or that its estimate is known. Further, how to get this estimate across the iterations with be explained. In the following, our notations differ slightly from those in [92] in order for us to preserve the coherence of our notations throughout the manuscript.
For instance, in the seminal work about GWO, the position vector of a grey wolf at any iteration iter of the algorithm was denoted by $\overrightarrow{\mathbf{X}}$. In this manuscript, the position of the Q wolves in the herd is denoted by $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$, $q = 1, \ldots, Q$. The position vector of the prey (global solution) is denoted by $\mathbf{g}^{\mathbf{k}}$. Firstly, let's present the principles of the algorithm : the encircling behavior is mathematically modeled as follows :

$$\mathbf{d} = |\mathbf{c} \cdot \mathbf{g}^{\mathbf{k}} - \mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})| \tag{1.22}$$

$$\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}+1) = \mathbf{g}^{\mathbf{k}} - \mathbf{b} \cdot \mathbf{d} \tag{1.23}$$

where the vectors $\mathbf{b}$ and $\mathbf{c}$ are calculated as $\mathbf{b} = 2\mathbf{a} \cdot \mathbf{r}_1 - \mathbf{a}$ and $\mathbf{c} = 2 \cdot \mathbf{r}_2$. In these expressions, the components of vector $\mathbf{a}$ are all equal to a, a scalar value which is a key parameter in the algorithm that we call phase distinguisher. The value of a decreases from 2 to 0 across the iterations. Vectors $\mathbf{r}_1$, $\mathbf{r}_2$ have random components between 0 and 1.
During the hunt, the wolves firstly diverge from each other to search for prey. Secondly, they converge to attack prey. This is mathematically modeled through the deterministic vector $\mathbf{a}$. When $a > 1$, the search agents are obliged to diverge from the prey : this is the exploration phase. Conversely, when $a \leq 1$, the search agents are obliged to attack towards the prey : this is the exploitation phase. We notice that a key parameter to balance the exploration and the exploitation phases is the deterministic vector $\mathbf{a}$.

Of course, the position of the global minimum is not known. Hence, to define the position of this wolf at iteration $\mathrm{iter}+1$, the locations of the three best solutions (denoted by α, β, and δ) obtained so far are taken into account.

The steps of GWO are described in algorithm 2.

Algorithm 2 Pseudo-code : Grey Wolf Optimization for multiple parameter estimation

Inputs : fitness function, small factor ϵ set by the user, to stop the algorithm, lower and upper bounds for each parameter.

1. Set iteration number $\mathrm{iter}=1$, create an initial herd composed of Q wolves with all required parameter values $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$, $q=1,\ldots,Q$. This initial population takes the form of a matrix with Q rows and N columns.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}))$ of each wolf $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$, $q=1,\ldots,Q$.
3. Sort the wolves through their fitness value and update the α, β, and δ wolves which hold respectively the first, second and third best fitness value. Store their position in vectors $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, and $\mathbf{x}_\delta^{\mathbf{k}}$ respectively.
4. Repeat steps for each wolf $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$, $q=1,\ldots,Q$:
   (a) Compute the contributions $\mathbf{y}^\alpha$, $\mathbf{y}^\beta$, and $\mathbf{y}^\delta$ of wolves α, β, and δ respectively to the displacement of the q^{th} wolf.
   (b) Compute the update position $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}+1)$ of the q^{th} wolf :

$$\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}+1)=\frac{1}{3}(\mathbf{y}^\alpha+\mathbf{y}^\beta+\mathbf{y}^\delta) \tag{1.24}$$

5. If $\mathrm{iter}<\mathrm{T_{max}}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}))>\epsilon$, increase iter, and go to step 2.

Output : estimated parameter values $\hat{K_1},\hat{K_2},\ldots,\hat{K_N}$

Here are some details for GWO algorithm :
In step 4b the updated position for any wolf q is computed from Eq. (1.24) as the equal contribution of the α , the β, and the δ wolves. These contributions, mentioned in step 4a, are computed as follows :

$$\mathbf{y}^\alpha=\mathbf{x}_\alpha^{\mathbf{k}}-\mathbf{b}_1\cdot\mathbf{d}_\alpha,\ \mathbf{y}^\beta=\mathbf{x}_\beta^{\mathbf{k}}-\mathbf{b}_2\cdot\mathbf{d}_\beta,\ \mathbf{y}^\delta=\mathbf{x}_\delta^{\mathbf{k}}-\mathbf{b}_3\cdot\mathbf{d}_\delta \tag{1.25}$$

with :

$$\mathbf{d}_\alpha=|\mathbf{c}_1\cdot\mathbf{x}_\alpha^{\mathbf{k}}-\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})|,\ \mathbf{d}_\beta=|\mathbf{c}_2\cdot\mathbf{x}_\beta^{\mathbf{k}}-\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})|,\ \mathbf{d}_\delta=|\mathbf{c}_3\cdot\mathbf{x}_\delta^{\mathbf{k}}-\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})| \tag{1.26}$$

In the seminal version of GWO [92], a is regularly decreased from 2 to 0 : $a=2(1-\frac{\mathrm{iter}}{T_{max}})$, where iter is the iteration index, and T_{max} is the maximum number of iterations. The

exploration step lasts until $a = 1$, the exploitation step lasts from $a = 1$ to $a = 0$. In [95], a modified GWO (mGWO) is implemented with an exponential model for $a : a = 2(1 - \frac{\text{iter}^2}{\text{T}_{\text{max}}{}^2})$. This permits to emphasize the exploration phase, that is, to encourage a global search, at the expense of the exploitation phase.

1.4.1.3 Highlights of the Tree-Seed Algorithm

Proposed by Kiran in [61], the Tree-Seed Algorithm (TSA) mimics the relationship between trees and their seeds. In this bio-inspired algorithm, the particles searching the best fitness are symbolized by the trees and the searching space is symbolized by the soil in which the trees will grow.
At each iteration, each one of the trees will launch seeds which will grow new trees at a new position in the searching space. This new position is chosen using one of two equations. Eq. (1.27a) uses the current best position found while Eq. (1.27b) uses a random tree's position among the population.

$$S_{k,j} = T_{i,j} + \alpha_{i,j} \times (B_j - Tr, j) \tag{1.27a}$$
$$S_{k,j} = T_{i,j} + \alpha_{i,j} \times (T_{i,j} - Tr, j) \tag{1.27b}$$

with $S_{i,j}$ the value of the jth parameter of the kth seed that is produced, $T_{i,j}$ the value of the jth parameter of the ith tree, B_j the jth parameter of the best position found until now and Tr, j the value of the jth parameter of the rth tree selected randomly and $r \neq i$. The parameter $\alpha_{i,j}$ is a factor randomly selected from the range $[-1, 1]$.
Which equation to use is randomly chosen but is strongly dependent of the parameter ST, set at the initialization of the TSA.

The initial population of tree set at the begining of the algorithm is chosen using the Eq. (1.28). And after each iteration, if

$$T_{i,j} = L_{j,min} + r_{i,j} \left(H_{j,max} - L_{j,min}\right) \tag{1.28}$$

with $L_{j,min}$ and $H_{j,max}$ respectively the lower and higher bound of the search space for the jth parameter and $r_{i,j}$ a random value between 0 and 1.

The steps of the TSA is described in Algorithm 3.

Algorithm 3 Pseudo-code : Tree-Seed Algorithm for multiple parameter estimation

Inputs : fitness function f maximum number of iterations T_{max}, ST parameter, lower and upper bounds for each parameter.

1. Set iteration number iter $= 1$, create an initial population of N trees using Eq. (1.28).
2. Evaluate fitness function value $f(T_i)$ of each tree T_i, with $i = 1, \ldots, N$, and select the tree with best fitness B.
3. Repeat steps for each tree T_i, $i = 1, \ldots, N$:
   (a) Decide the number of seeds K produced by T_i
   (b) Repeat steps for each seed S_k, $k = 1, \ldots, K$
   For each parameter $S_{k,j}$
      i. Get a random number $rand$ with $0 \leq rand \leq 1$
      ii. Compute the position $S_{k,j}$
      if $rand < ST$ use Eq. (1.27a)
      else if $rand > ST$ use Eq. (1.27b)
   (c) Compute the fitness $f(S_k)$ for each seed and select the seed with the best fitness $B(S)$
   If $f(B(S))$ is better than $f(T_i)$, then $T_i = B(S)$.
4. Select the tree with the best fitness $B(T)$.
   If $f(B(T))$ is better than $f(B)$, then $B = B(T)$
5. If iter $<$ $\mathrm{T_{max}}$, increase iter, and go to step 2.

Output : Estimated best solution B

A low value of ST will emphasize the probability of a global search, using Eq. (1.27b) whereas a high value of ST will emphasize the probability of a local search, using Eq. (1.27a).

1.4.2 Bio-inspired Optimization in discrete space

The methods which aim at estimating the best integer values which minimize some linear criterion are called integer linear programming (ILP). When only part of the values are enforced to be integer, one refers to mixed-integer linear programming (MILP).

A major discrete optimization method is ant colony optimization (ACO). ACO is a nature-inspired optimization algorithm [30] motivated by the natural collective behavior of real-world ant colonies. Artificial ants used in ACO are stochastic solution construction procedures that probabilistically build a solution by iteratively adding solution components to partial so-

lutions [30] by taking into account heuristic information about the problem instance being solved, if available, and (artificial) pheromone trails which change dynamically at runtime to reflect the agents' acquired search experience.
The other swarm intelligence methods, which are originally dedicated to continuous search spaces, have been firstly adapted to binary problems. Binary optimization is a subfield of discrete optimization, in the sense that the research space is restricted to the set $\{0, 1\}$ for all parameters. In [57] a binary version of PSO (BPSO) is proposed. In [13] a mimetic binary particle swarm optimization scheme is introduced based on hybrid local and global searches in BPSO. In [38], the first binary version of Cuckoo search was proposed, while very last advances about TSA also concern a binary version, proposed in [25].
In [33] a binary GWO is proposed and used to select an optimal feature subset for classification purposes. In [80], the first discrete version of GWO which is not restricted to binary search spaces is proposed, to address a combinatorial problem. A multi-objective mixed integer programming model is formulated. These advances are driven by an application to the job scheduling of machines. One of the contributions in [80] consists in turning the computation of the influence of the leaders into a decision process : one leader is selected at random, between the three leaders, at each iteration. Job permutation is performed with a probability $\frac{1}{3}$ for each leader at any iteration. Machine assignment is performed with a probability which depends on the iteration index.
In [79], to further enhance the exploration, the authors embed a genetic operator, namely permutation, into their discrete GWO algorithm. The proposed method is called hybrid multi-objective grey wolf optimizer because GWO and GA are mixed together. In [63], an improved discrete cuckoo optimization algorithm is proposed for a flowshop scheduling problem.

1.4.2.1 Highlights of the Ant Colony Optimization

Proposed in 1996 in [29], the Ant Colony Optimization (ACO) method, originally called the Ant Search (AS) system, is a discrete bio-inspired optimization method designed for combinatorial optimization problems, such as the traveling salesman problem (TSP), using the pheromone phenomenon used by ants in the real world. The goal of a combinatorial optimization problem is to find the best itinerary among several nodes that should be followed in order to optimize the output, without passing through the same node twice.
In ACO, each particle, symbolized here by an ant, will be composed of 2 elements of information : the node where it currently is in and its constructed solution, $i.e.$ the nodes where it has already been and the arrangement it has followed. This constructed solution is updated as the algorithm runs.

At initialization, each ant k is assigned to a randomly chosen node i. Each ant will then move to another node j using the probability given by the Eq. (1.29).

$$p_{ij}^{k}(\text{iter}) = \frac{[\tau_{ij}(\text{iter})]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in \mathcal{N}_i^k} [\tau_{ij}(\text{iter})]^{\alpha} \cdot [\eta_{il}]^{\beta}} \quad \text{if } j \in \mathcal{N}_i^k \tag{1.29}$$

with $\eta_{ij} = 1/d_{ij}$ is an a-priori available heuristic information, α and β are two parameters determining the influence of pheromone trail τ_{ij} and heuristic information η_{ij} and $\mathcal{N}_i^k$ is the allowed neighborhood of the ant k, *i.e.* the nodes around the current node i where the ant k has not been yet.

After each ant has traveled through n nodes, the pheromone trail is updated using Eq. (1.30).

$$\tau_{ij}(\text{iter}+1) = (1-\rho) \cdot \tau_{ij}(\text{iter}) + \sum_{k=1}^{Q} \Delta\tau_{ij}^{k}(\text{iter}) \quad \forall (i,j) \tag{1.30}$$

with $0 < \rho \leq 1$ is the pheromone trail evaporation trail parameter, Q is the number of ants and $\Delta\tau_{ij}^{k}(\text{iter})$ is the amount of pheromone deposed by the ant k and is defined by Eq. (1.31).

$$\Delta\tau_{ij}^{k}(\text{iter}) = \begin{cases} 1/L^{k}(\text{iter}) & \text{if arc } (i,j) \text{ is used by ant } k \\ 0 & \text{otherwise} \end{cases} \tag{1.31}$$

where $L^{k}(\text{iter})$ is the length of the kth ant's complete track.

Eq. (1.31) is made so that the shorter an ant's track, the greater the amount of pheromone deposed. As such, the shortest path will be chosen more easily on the next iterations.
At the end of the ACO algorithm, the complete trail with the highest quantity of pheromones is considered as the best solution.

1.4.2.2 Highlights of the Multi-Objective Discrete GWO

The discrete GWO proposed in [80], a multi-objective discrete grey wolf optimizer (MODGWO), is proposed to solve a combinatorial multi-objective optimization problem. The main idea is to turn the computation of a centroid -which is not valid in a discrete search space- into a selection process. Therefore, contrary to the update rule presented in Eq. (1.24), no weighted summation is used but instead a selection is performed to shift a given wolf in the search space.
In [80], a job assignment problem with several stages is considered : with n jobs and m stages, finding the best job sequence involves the estimation of n parameters, and finding the

number of machines to process the operation of job j $(1 \geq j \geq n)$ at the stage i $(1 \geq i \geq m)$ involves the estimation of n m parameters : in total $n + n$ m parameters. So two types of parameters are involved : job index in the sequence, and machine assignment. Each one has its own update rule. In [80], problems with up to 100 jobs and 5 stages are studied. For sake of clarity, we considered the simplified case where $n = 1$ and $m = 1$. The first update rule is valid for the first parameter (job index), denoted by K_1. The second update rule is valid for the second parameter (machine assignment), denoted by K_2. In this subsection, the notations K_1^α, K_1^β, K_1^δ, K_2^α, K_2^β, K_2^δ hold for the first and second components respectively of the leader vectors $\mathbf{x}_\alpha^\mathbf{k}$, $\mathbf{x}_\beta^\mathbf{k}$, $\mathbf{x}_\delta^\mathbf{k}$.
The first update rule is as follows : Let $rand$ be a random real number between 0 and 1.

$$K_1 = \begin{cases} K_1^\alpha & \text{if} \quad rand \leq \frac{1}{3} \\ K_1^\beta & \text{if} \quad rand > \frac{1}{3} \quad \text{and} \quad rand \leq \frac{2}{3} \\ K_1^\delta & \text{if} \quad rand > \frac{2}{3} \quad \text{and} \quad rand \leq 1 \end{cases} \tag{1.32}$$

The second update rule is as follows :

Let $rand$ be a random real number between 0 and 1.

$$K_2 = \begin{cases} K_2^\alpha,\ K_2^\beta, \text{ or } K_2^\delta & \text{if} \quad rand \leq a \\ K_2' & \text{if} \quad rand > a \quad \text{and} \quad rand \leq 1 \end{cases} \tag{1.33}$$

In Eq. (1.33), $a = 1 - \frac{\text{iter}}{\text{T}_{\max}}$ at a given iteration iter, and K_2' is the second component of any wolf in the herd, taken at random.

We notice that, in Eqs. (1.32) and (1.33), each leader may be selected with the same probability as the others. The first component of the current wolf is directly replaced by the first component of the selected leader. In Eq. (1.33), as parameter a decreases from 1 to 0 across the iterations, the probability for one of the leaders to be chosen decreases and the probability for the random component K_2' to be chosen increases.

1.5 Application of bio-inspired optimization methods to image processing

The physics-based WDO method has been applied to solve an issue of satellite image thresholding [15]. As for swarm intelligence algorithms, they have been used essentially for classification issues, and more precisely for the estimation of the parameters of SVM (support vector machines) classifiers [71, 126, 76]. In [21] a 'firefly-SVM' algorithm is proposed to train all parameters of the SVM simultaneously. In [32], GWO estimates the parameters of an SVM classifier : 3-fold validation is performed for color image classification. GWO has

also been applied to the training of Multi-Layer Perceptron [90]. Bio-inspired optimization is undoubtedly of great interest for the processing of the images provided by optical remote sensing intruments [127, 111]. Indeed, their huge dimentionality and complex data structure yield nonlinear optimization problems [127]. For instance, in [87], GWO is adapted to band selection in hyperspectral data. Binary versions of GWO have been developped for image processing applications : feature selection [33], and classification of cervix lesion images [107].

Continuous bio-inspired optimization methods have also been used for multispectral and hyperspectral image processing problems.
Generally speaking, an image is a multidimensional array, whose values are accessed via indices : we need two indices to access the values of a 2-dimensional (2-D) image, and 3 indices to access the values of a 3-dimensional (3-D) image. Such a multidimensional array is called 'tensor' [98].
A seminal work dedicated to tensor denoising consisted in adapting Wiener filtering in a tensor framework, yielding the Multiway Wiener Filtering (MWF) [97], a subspace-based method requiring the estimation of the dimension of the signal subspace, also called rank, along each mode. The following notations are adopted : $\mathcal{X}$ is the noise-free tensor, $\mathcal{R}$ is the noised tensor and $\hat{\mathcal{X}}$ is the estimated tensor. In this paper, modes are indexed by i, and we consider third-order tensors : multispectral images of size $I_1 \times I_2 \times I_3$. For $i = 1, 2$ or 3, I_i is the size of the multispectral image along the i^{th} mode : I_1 is the number of rows, I_2 the number of columns, and I_3 the number of bands. Impaired multispectral images are expressed as : $\mathcal{R} = \mathcal{X} + \mathcal{N}$, where tensor $\mathcal{N}$ stands for additive independent and identically distributed zero-mean Gaussian noise.
A signal subspace value for mode i is denoted by K_i. We denote by $\hat{\mathcal{X}}(K_1, K_2, K_3)$ the estimate provided by Multiway Wiener Filtering applied to $\mathcal{R}$ with rank values K_1, K_2, K_3. Details about MWF can be found in [98, 97]
In [72], MWF has been inserted into a wavelet framework to denoise images while preserving details. The advantage of this method is to preserve small features while denoising efficiently, but requires the knowledge of numerous rank values. In [128], a least squares (LS) criterion is minimized with PSO to estimate these rank values.
To the best of our knowledge, and due to their novelty, the discrete versions of GWO which have been applied to solve an image processing issue are restricted to binary. Moreover, no mixed bio-inspired optimization method has been developed to solve any image processing issue.

1.6 Conclusion

In this Chapter, a broad overview of the state-of-the-art in 2 completely distinct fields has been done :
• the audience measurement in computer vision, which is composed of face detection, object tracking and face recognition,
• bio-inspired optimization methods, and more particularly the swarm optimization methods.
Concerning audience measurement, it is possible to anticipate which methods are the best adapted for an audience measurement tool which must have real-time performance while running on an embedded system. For the face detection part, the Viola-Jones algorithm, described in subsection 1.1.1, appears as the best one to use. Indeed, it has a low computational complexity solely depending of the image size [55] and has proven its worth for real-time detection [116]. However, a work is needed about the handling of the false detections. For the object tracking, the sparse version of the Lucas-Kanade Optical flow, described in subsection 1.2.2, appears as the most fitting to the studied problem as it is fast and easy to compute. Moreover, the object to track in the target application, $i.e.$ faces, will have an unknown and non repetitive trajectory, therefore not compatible with the Kalman filters. Finally, in order to perform gender recognition, image classification by SVM seems to be the best choice. Indeed, in the industrial context of this application, the classifier must be set in stone prior being run on the systems. Indeed, an online method, $i.e.$ which can continuously trained itself, such as the CNN, is not compatible with a low computational load whereas a unmovable method such as the SVM is fitting. However, a special attention must be paid on the SVM training in order to avoid a low accuracy and a loss of time in repetitive trainings.
Regarding swarm optimization methods from the bio-inspired methods family, it appears from this review that all the state-of-the-art method are either for fully continuous problems or for fully discrete (or binary) problems. Bio-inspired optimization methods designed for mixed problems, $i.e.$ with both continuous and discrete variables, do not exist yet.
Such a method is proposed in this thesis and detailed next in Chapter 2.

CHAPTER 2

Mixed Grey Wolf Optimizer

2.1 Proposed Mixed grey wolf optimizer : theory

The general goal of this study is to propose a bio-inspired optimization algorithm able to tackle problems with both discrete and continuous variables. In the rest of this report, those kind of problems will be named as **mixed problems**. Moreover, it is relevant for the research field to have a tool able to adapt itself to the encountered problem. Thus, the proposed algorithm should be able not only to handle mixed problems but also fully discrete problems or fully continuous problems. As it is more robust than the basic PSO regarding the local minimum avoidance and regarding its computation speed, the GWO has been chosen as the base of the proposed algorithm. Moreover, it has already been adapted with success to a discrete version non binary, although it has been designed for a specific application [79, 80].

The discrete process is applied to a subset of the expected parameters, taking their values in discrete search spaces, while the continuous process is applied independently to the other parameters, taking their values in continuous search spaces.

Firstly, in subsection 2.1.1 we propose a discrete bio-inspired optimization algorithm, that we will call 'improved discrete GWO' (**IDGWO**). Though inspired by GWO because it involves several leaders, its continuous counterpart is not exactly the classical GWO as described previously in section 1.4.1.2. That is why we explain in subsection 2.1.2 what is the exact continuous counterpart of the proposed discrete GWO. Finally, we combine these two algorithms in a mixed grey wolf optimizer presented in subsection 2.1.3, which handles both discrete and continuous parameters.

2.1.1 Improved discrete GWO

As explained previously in section 1.4.2, the existing discrete bio-inspired methods are usually designed for binary problems or for specific problems [79, 80]. Therefore, as they are not applicable in a global manner, it is relevant to develop a discrete version of GWO which won't aim for a specific application but for being used in a a wider range of applications.

The idea behind the improved discrete grey wolf optimizer is that search spaces are defined which are discrete, and that not only vectors with values are defined for each wolf but also vectors with indexes. Still, a search agent or 'wolf' at iteration iter is denoted by $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$ and contains N components which are candidate values of expected parameters.

We notice that a value located at the component i with index h_i in $\mathbf{d}_i^{ind}$ is denoted by $K_i^{h_i}$. For instance, for the vector of values $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}) = \left[K_1^2, \ldots, K_N^5\right]^T$, the associated vector of indexes is $\mathbf{h}_q(\mathrm{iter}) = [2, \ldots, 5]^T$.
Algorithm 4 details the overall process of the proposed improved discrete GWO while in Algorithm 5, we detail the wolf update rule for one wolf index $q \in [1, \ldots, Q]^T$, at any iteration iter.

Algorithm 4 Pseudo-code : Improved Discrete Grey Wolf Optimization for multiple parameter estimation

Inputs : fitness function, number N of expected parameters, small factor ϵ set by the user, to stop the algorithm, maximum number of iterations T_{max}.
For each parameter indexed by $i = 1, \ldots, N$: the search space $\mathbf{d}_i^{ind}$ with H_i possible values.

1. Set iteration number iter $= 1$.
   Create an initial set of index vectors $\mathbf{h}_q(\text{iter})$, $q = 1, \ldots, Q$. For each index i between 1 and N, a component $h_i(\text{iter})$ of $\mathbf{h}_q(\text{iter})$ is an integer value between 1 and H_i.
   Create an initial herd composed of Q wolves $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$ with the N required parameter values. This initial population takes the form of a matrix with Q rows and N columns. For each index q, $x_i(\text{iter}) = \mathbf{d}_i^{ind}(h_i(\text{iter}))$.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Sort the wolves through their fitness value and update the wolves which hold the first, second and third best fitness value : store the corresponding vectors of indexes $\mathbf{h}_\alpha$, $\mathbf{h}_\beta$, $\mathbf{h}_\delta$ and vectors of values $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, $\mathbf{x}_\delta^{\mathbf{k}}$.
4. If $a > 1$, select two wolves $\rho 1$ and $\rho 2$, randomly among the herd of Q wolves, with $\rho 1 \neq \rho 2$. Store the corresponding vectors of indexes $\mathbf{h}_{\rho 1}$, $\mathbf{h}_{\rho 2}$ and vectors of values $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, $\mathbf{x}_{\rho 2}^{\mathbf{k}}$.
   Else if $a \leq 1$, go to step 5.
5. Repeat steps for each wolf q, $q = 1, \ldots, Q$, with vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ and vector of indexes $\mathbf{h}_q(\text{iter})$:
   Apply Algorithm 5.
6. Exchange the current population with the new one, obtained at step 5
7. If iter $<$ T$_{\max}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, increase iter, and go to step 2.

Output : estimated parameter values $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$ contained in $\mathbf{x}_\alpha^{\mathbf{k}}$.

Algorithm 5 Pseudo-code : Update rule for Improved Discrete Grey Wolf Optimization for multiple parameter estimation

Inputs : Vector of indexes $\mathbf{h}_q(\text{iter})$, vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$.

1. Select leader :
   (a) Let $\mathbf{x}_{\text{l}}^{\mathbf{k}}$ denote the vector of values for the selected leader, chosen randomly among $\mathbf{x}_{\alpha}^{\mathbf{k}}$, $\mathbf{x}_{\beta}^{\mathbf{k}}$, $\mathbf{x}_{\delta}^{\mathbf{k}}$, $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, and $\mathbf{x}_{\rho 2}^{\mathbf{k}}$. This selection process is detailed in Eqs. (2.1) to (2.2).
   (b) Store the vector of indexes corresponding to $\mathbf{x}_{\text{l}}^{\mathbf{k}}$ in a vector denoted by $\mathbf{h}_{\text{l}}$.
2. Update wolf q :
   for each index $i = 1, \ldots, N$
   (a) Compute the updated component $h_i(\text{iter}+1)$.
   (b) Compute the updated component $x_i(\text{iter}+1)$.

   This update process is detailed in Eqs. (2.3), (2.5), (2.6)
3. Store the N components obtained at steps 2a and 2b :
   (a) Get the updated vector of indexes :
   $\mathbf{h}_q(\text{iter}+1) = [h_1(\text{iter}+1), \ldots, h_i(\text{iter}+1), \ldots, h_N(\text{iter}+1)]^T$.
   (b) Get the updated vector of values :
   $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1) = [x_1(\text{iter}+1), \ldots, x_i(\text{iter}+1), \ldots, x_N(\text{iter}+1)]^T$.

Output : $\mathbf{h}_q(\text{iter}+1)$, $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$

Here are some details about the update rule of the proposed improved discrete GWO algorithm :

At step 1, if $a > 1$, the leader is selected randomly among the α, β, δ, $\rho 1$, and $\rho 2$ wolves :

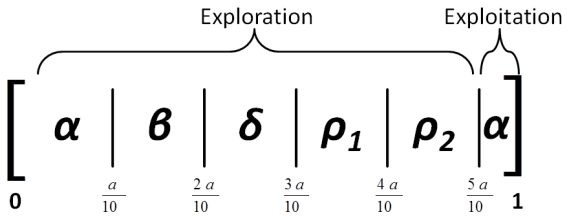
$$\mathbf{x}_{\text{l}}^{\mathbf{k}} = \begin{cases} \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{if} \quad r \leq \frac{a}{10} \\ \mathbf{x}_{\beta}^{\mathbf{k}} & \text{if} \quad r > \frac{a}{10} \quad \text{and} \quad r \leq \frac{2a}{10} \\ \mathbf{x}_{\delta}^{\mathbf{k}} & \text{if} \quad r > \frac{2a}{10} \quad \text{and} \quad r \leq \frac{3a}{10} \\ \mathbf{x}_{\rho 1}^{\mathbf{k}} & \text{if} \quad r > \frac{3a}{10} \quad \text{and} \quad r \leq \frac{4a}{10} \\ \mathbf{x}_{\rho 2}^{\mathbf{k}} & \text{if} \quad r > \frac{4a}{10} \quad \text{and} \quad r \leq \frac{5a}{10} \\ \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{if} \quad r > \frac{5a}{10} \end{cases} \tag{2.1}$$

where r is a random value in $\mathbb{R}$, between 0 and 1.

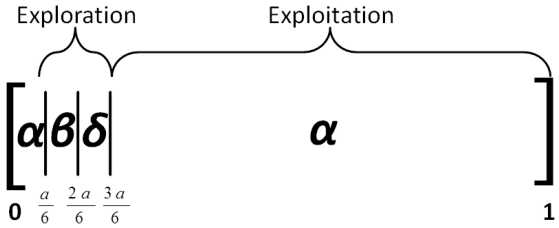
if $a \leq 1$, the leader is selected randomly among the α, β, and δ wolves :

$$\mathbf{x}_{\mathrm{l}}^{\mathbf{k}} = \begin{cases} \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{if} \quad r \leq \frac{a}{6} \\ \mathbf{x}_{\beta}^{\mathbf{k}} & \text{if} \quad r > \frac{a}{6} \quad \text{and} \quad r \leq \frac{2a}{6} \\ \mathbf{x}_{\delta}^{\mathbf{k}} & \text{if} \quad r > \frac{2a}{6} \quad \text{and} \quad r \leq \frac{3a}{6} \\ \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{if} \quad r > \frac{3a}{6} \end{cases} \tag{2.2}$$

where r is a random value in $\mathbb{R}$, between 0 and 1.
As the parameter a is decreasing from 2 to 0 across the iterations, it is more and more probable for α to be chosen as the leader. The random wolves may be selected during the first part of the process, when $a > 1$, and cannot be selected during the second part of the process, when $a \leq 1$. This is coherent with the paradigm of the original Grey Wolf Optimizer [92], where exploration is emphasized when $a > 1$, and exploitation is emphasized when $a \leq 1$.



(a) Interval of possible values of r : possibilities of leader selection with $a \simeq 2$



(b) Interval of possible values of r : possibilities of leader selection with $a \simeq 0$

Figure 2.1 — Evolution of the possibilities of leader selection according to the decrease of a

The Figure 2.1 illustrates how the separation between the exploration phase and the exploitation phase is done in the proposed algorithm. At the beginning of the process, when $a \simeq 2$, each leader may be chosen with the same probability, which permits to 'explore' the search space. When a decreases towards 0, the last possibility in Eqs. (2.1) and (2.2) may be selected with a higher probability, which permits to 'exploit' the corresponding promising location in the search space. Indeed, a wolf will then move according to the leader α, which is the leader with the best fitness.
The leader $\mathbf{x}_{\mathrm{l}}^{\mathbf{k}}$ will rule the displacement of wolf $\mathbf{x}_{q}^{\mathbf{k}}(\mathrm{iter})$ as explained below.
At step 2a, each component of the vector of indexes $\mathbf{h}_{q}(\mathrm{iter})$ is updated as follows :

For each index i, $i = 1, \ldots, N$:

$$h_i(\text{iter}+1) = (h_i(\text{iter}) + \Delta \; sgn(h_i^1 - h_i(\text{iter}))) \; mod \; H_i \tag{2.3}$$

where :

— $sgn(\cdot)$ denotes the sign function, which is such that $sgn(z) = -1$ if $z < 0$, $sgn(z) = 0$ if $z = 0$, and $sgn(z) = 1$ if $z > 0$ for any real value z ;

— mod denotes the 'Modulo' operator, defined as follows : whatever the real values $u \in \mathbb{R}_+$ and $v \in \mathbb{R}_+^*$:

$$u \; mod \; v = \begin{cases} u - v\lfloor u/v \rfloor & \text{if} \quad u \neq v \\ v & \text{if} \quad u = v, \text{ or } u = 0 \end{cases} \tag{2.4}$$

where $\lfloor \cdot \rfloor$ denotes integer part.

— Δ is computed as follows :

$$\Delta = \begin{cases} 1 & \text{if} \quad \phi \leq \frac{a}{2\Omega(H_i)} \\ 2 & \text{if} \quad \phi > \frac{a}{2\Omega(H_i)} \quad \text{and} \quad \phi \leq \frac{2a}{2\Omega(H_i)} \\ \vdots & \\ n_i & \text{if} \quad \phi > \frac{(n_i-1)a}{2\Omega(H_i)} \quad \text{and} \quad \phi \leq \frac{n_i a}{2\Omega(H_i)} \\ \vdots & \\ \Omega(H_i) - 1 & \text{if} \quad \phi > \frac{(\Omega(H_i)-2)a}{2\Omega(H_i)} \quad \text{and} \quad \phi \leq \frac{(\Omega(H_i)-1)a}{2\Omega(H_i)} \\ \Omega(H_i) & \text{if} \quad \phi > \frac{(\Omega(H_i)-1)a}{2\Omega(H_i)} \quad \text{and} \quad \phi \leq \frac{\Omega(H_i)a}{2\Omega(H_i)} \\ 1 & \text{if} \quad \phi > \frac{\Omega(H_i)a}{2\Omega(H_i)} \end{cases} \tag{2.5}$$

where ϕ is a random value in $\mathbb{R}$, between 0 and 1 and :

$$\Omega(H_i) = \begin{cases} \frac{H_i}{2} & \text{if } H_i \text{ is even} \\ \frac{H_i+1}{2} & \text{if } H_i \text{ is odd} \end{cases}$$

As the parameter a is decreasing from 2 to 0 across the iterations, it is more and more probable for the value 1 to be chosen, and less and less probable for the larger values such as 2 and 4 to be chosen. This means that, at the beginning of the process, the wolf which is currently modified may be displaced by 2 or 4 components in the direction of the leader with a rather high probability, and that, at the end of the process, it is highly probable that it will be displaced by only 1 component in the direction of the leader.
This permits to explore the research space with large displacement values at the beginning of the optimization process, and to perform exploitation, with rather small displacements towards the selected leader at the end of the optimization process.

Then, at step 3a the updated values $h_i(\text{iter}+1)$, $i = 1, \ldots, N$, computed in Eq. (2.6) are stored in vector $\mathbf{h}_q(\text{iter}+1)$.

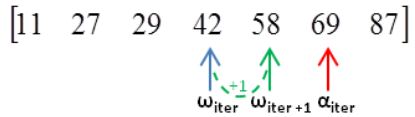
At step 2b, each component of the updated vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$ is computed as follows :

$$x_i(\text{iter}+1) = \mathbf{d}_i^{val}(h_i(\text{iter}+1)) \tag{2.6}$$

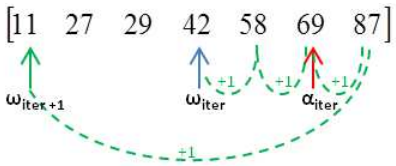
Then, at step 3b the updated values $x_i(\text{iter}+1)$, $i = 1, \ldots, N$, computed in Eq. (2.6) are stored in vector $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$.

When each vector, or 'search agent' $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$, $q = 1, \ldots, Q$, has been computed at step 5, the whole population of wolves is updated at step 6, and one may increase the iteration index or terminate the algorithm. At step 7, vector $\mathbf{x}_\alpha^{\mathbf{k}}$ contains the estimated parameters $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$.

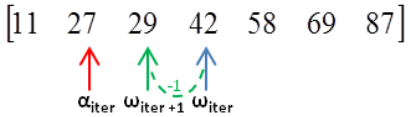
The Figure 2.2 illustrates a simple example with 3 possible updates for a wolf ω. In this example, the α leader has been selected as the leader that ω must follow. Moreover, there is only one parameter to search with $\mathbf{d}^{val} = [11, 27, 29, 42, 58, 69, 87]^T$ and $H = 7$.



(a) Case 1 : $\Delta = 1$



(b) Case 2 : $\Delta = 4$



(c) Case 3 : $\Delta = 1$

Figure 2.2 — Examplification of a wolf update with the proposed discrete GWO

Table 2.1 presents, in cases 2.2a, 2.2b, and 2.2c, the index $h(iter)$, the updated index $h(iter+1)$ and the updated value $x(iter+1) = \mathbf{d}^{val}(h(iter+1))$.

This improved discrete GWO is the most innovative part of the mixed GWO we propose. In subsection 2.1.2, we present its continuous counterpart.

Case \ Wolf update	h^l	Δ	$sgn(h^l - h(iter))$	$h(iter)$	$h(iter+1)$	$x(iter+1)$
2.2a	6	1	$+1$	4	$(4+1)\ mod\ 7 = 5$	58
2.2b	6	4	$+1$	4	$(4+4)\ mod\ 7 = 1$	11
2.2c	2	1	-1	4	$(4-1)\ mod\ 7 = 3$	29

Table 2.1 — Updated index values for cases illustrated in Fig. 2.2

2.1.2 Global Continuous Grey Wolf Optimizer

In Algorithm 6, we detail the continuous version of wolf leader selection and wolf update for one parameter index $i \in [1, \ldots, N]^T$ and one wolf index $q \in [1, \ldots, Q]^T$, at any iteration index iter. To preserve the coherence with the proposed improved discrete grey wolf optimizer, we introduce two random wolves which may also contribute the displacement of any wolf. As these random wolves help the algorithm exploring the search space in avoiding local minima, we call this version of GWO the Global Continuous Grey Wolf Optimizer.

Algorithm 6 Pseudo-code : Global Continuous Grey Wolf Optimization for multiple parameter estimation

Inputs : $x_i(\text{iter})$, i^{th} component of a given wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ at iteration iter ; leaders α, β, δ.

1. Compute the contributions y_i^{α}, y_i^{β}, and y_i^{δ} of wolves α, β, and δ respectively to the displacement of the q^{th} wolf.
   This computation is detailed in Eqs. (2.9) and (2.10) below.
2. if $a > 1$ go to step 3, else if $a \leq 1$ go to step 4
3. Compute the contributions $y_i^{\rho 1}$ and $y_i^{\rho 2}$ of wolves $\rho 1$ and $\rho 2$, respectively to the displacement of the q^{th} wolf.
   This computation is detailed in Eqs. (2.9) and (2.10) below.
4. Compute the update position at the i^{th} of the q^{th} wolf :
   if $a > 1$:
   $$x_i(\text{iter}+1) = \frac{1}{5}(y_i^{\alpha} + y_i^{\beta} + y_i^{\delta} + y_i^{\rho 1} + y_i^{\rho 2}) \tag{2.7}$$
   else if $a \leq 1$:
   $$x_i(\text{iter}+1) = \frac{1}{3}(y_i^{\alpha} + y_i^{\beta} + y_i^{\delta}) \tag{2.8}$$

Output : $x_i(\text{iter}+1)$

This updated position mentioned at step 4 of Algorithm 6 is computed as the equal contribution of the leaders α, β and δ. If $a > 1$, then the contributions of two random wolves $\rho 1$ and $\rho 2$ are also used. These contributions are computed as follows :

$$\begin{cases} y_i^{\alpha} = x_i^{\alpha} - b_1 \cdot d_i^{\alpha}, \\ y_i^{\beta} = x_i^{\beta} - b_2 \cdot d_i^{\beta}, \\ y_i^{\delta} = x_i^{\delta} - b_3 \cdot d_i^{\delta}, \\ y_i^{\rho 1} = x_i^{\rho 1} - b_4 \cdot d_i^{\rho 1}, \\ y_i^{\rho 2} = x_i^{\rho 2} - b_5 \cdot d_i^{\rho 2} \end{cases} \tag{2.9}$$

with :

$$\begin{cases} d_i^{\alpha} = |c_1 \cdot x_i^{\alpha} - x_i(\text{iter})|, \\ d_i^{\beta} = |c_2 \cdot x_i^{\beta} - x_i(\text{iter})|, \\ d_i^{\delta} = |c_3 \cdot x_i^{\delta} - x_i(\text{iter})|, \\ d_i^{\rho 1} = |c_4 \cdot x_i^{\rho 1} - x_i(\text{iter})|, \\ d_i^{\rho 2} = |c_5 \cdot x_i^{\rho 2} - x_i(\text{iter})| \end{cases} \tag{2.10}$$

where the scalars b and c are calculated as in the vanilla GWO : $b = 2ar_1 - a$ and $c = 2r_2$. In these expressions, r_1 and r_2 are random scalars between 0 and 1.

2.1.3 Extension to a mixed grey wolf optimizer

In a same problem, the expected parameters do not necessarily belong to the same type of searching space. Some of these parameters will be continuous parameter while the others will be discrete parameters and though, they may be interdependent and should be estimated simultaneously. We will refer to such problems as mixed problems. Combining the improved discrete and global continuous GWO methods proposed in subsections 2.1.1 and 2.1.2 respectively, we propose a mixed GWO method. The following notations will be specifically used for the mixed GWO, in addition to the notations presented in the Nomenclature at the beginning of this manuscript :

— Assuming that the i^{th} parameter K_i takes its values in a continuous search space, its minimum acceptable value is denoted by K_i^{min} and its maximum acceptable value is denoted by K_i^{max} ;

— the interval of acceptable values for the parameter K_i in a continuous search space is denoted by $\mathbf{d}_i^{val} = \left[K_i^{min}; K_i^{max}\right]^T$.

Algorithm 7 describes the proposed mixed grey wolf optimization while the Figures A.0 and A.1 , available in the Appendix, describe its flowchart.

Algorithm 7 Pseudo-code : Mixed Grey Wolf Optimization for multiple parameter estimation

Inputs : fitness function, small factor ϵ set by the user, to stop the algorithm.

1. Set iteration number iter $= 1$, create an initial herd composed of Q wolves with all required parameter values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$. This initial population takes the form of a matrix with Q rows and N columns.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Sort the wolves through their fitness value and update the α, β, and δ wolves which hold respectively the first, second and third best fitness value. Store their position in vectors $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, and $\mathbf{x}_\delta^{\mathbf{k}}$ respectively. For the discrete parameters, store the corresponding vectors of indexes of their discrete components $\mathbf{h}_\alpha$, $\mathbf{h}_\beta$ and $\mathbf{h}_\delta$.
4. If $a > 1$, select two wolves $\rho 1$ and $\rho 2$, randomly among the herd of Q wolves, with $\rho 1 \neq \rho 2$. Store the vectors of values $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, $\mathbf{x}_{\rho 2}^{\mathbf{k}}$ and, for the discrete parameters, the corresponding vectors of indexes $\mathbf{h}_{\rho 1}$, $\mathbf{h}_{\rho 2}$.
   Else if $a \leq 1$, go to step 5.
5. Repeat steps for each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$:
   For each component $x_i(\text{iter})$ with $i = 1, \ldots, N$:
   (a) if the i^{th} parameter K_i takes its values in a continuous search space then go to step 5b,
   else if K_i takes its values in a discrete search space then go to step 5c.
   (b) Apply the continuous versions of wolf update and displacement, proposed in Algorithm 6.
   Skip steps 5c to 5e.
   (c) select the leader $\mathbf{x}_1^{\mathbf{k}}$, and the displacement magnitude Δ which will be used for the discrete parameters. For this, refer to Eqs. (2.1), (2.2), (2.3), and (2.5).
   (d) apply step 2a of algorithm 4 to get $h_i(\text{iter} + 1)$ as in Eq. (2.3)
   (e) apply step 2b of algorithm 4 to get $x_i(\text{iter} + 1)$ as in Eq. (2.6)
6. Exchange the current population with the new one, obtained at step 5
7. If iter $< \text{T}_{\max}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, increase iter, and go to step 2.

Output : estimated parameter values $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$

Here are some remarks about Algorithm 7, the mixed GWO algorithm :

The components of each search agent may be updated with continuous update rules (see step 5b) or with discrete update rules (see step 5d), whether they belong to continuous or discrete search spaces. However, at a given iteration iter, a given wolf q characterized by the vector of values $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$ where continuous and discrete parameters are mixed yields one common fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}))$ (see step 2).

Further in the paper, we will distinguish between two versions of our mixed grey wolf optimizer, depending on the expression of the parameter a. In the mixed GWO (denoted by mixedGWO) the parameter a is expressed as follows :

$$a = 2(1 - \frac{\mathrm{iter}^{\eta}}{{T_{max}}^{\eta}}) \tag{2.11}$$

In a second version, that we call adaptive mixed GWO (denoted by amixedGWO) and inspired by the adaptive GWO presented in section 3.2, parameter a is such that :

$$a = \begin{cases} 2(1 - \frac{\mathrm{iter}^{\eta}}{{T_{max}}^{\eta}}) & \text{if} \quad \mathrm{iter} \leq \mathrm{T_{max}}/2 \\ 2(1 - \frac{\mathrm{iter}^{\frac{1}{\eta}}}{{T_{max}}^{\frac{1}{\eta}}}) & \text{if} \quad \mathrm{iter} > \mathrm{T_{max}}/2 \end{cases} \tag{2.12}$$

An elevated value of η encourages exploration during the first phase, from $\mathrm{iter} = 1$ to $\mathrm{iter} = \mathrm{T_{max}}/2$; and exploitation during the second phase, from $\mathrm{iter} = \mathrm{T_{max}}/2 + 1$ to $\mathrm{iter} = \mathrm{T_{max}}$.

2.1.4 Exemplification of the mixed Grey Wolf Optimizer

Let's exemplify the mixed GWO method on a mixed problem involving $N = 3$ parameters K_1, K_2 and K_3. K_1 and K_2 are continuous parameters with their respective search space being $[1; 8]$ and $[2; 50]$. K_3 is a discrete parameter with $\mathbf{d}_3^{val} = [11, 27, 29, 42, 58, 69, 87]^T$ and $H_3 = 7$. In the rest of the examples, the position of the wolf q, that is $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$, will be presented as follows : $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}) = \begin{bmatrix} K1 \\ K2 \\ K3 \end{bmatrix}$. In the following, the wolves will be denoted by α, β, δ, $\rho 1$, $\rho 2$ and ω.

Let the GWO algorithm at iteration t, with $T_{max} = 100$ and $\eta = 1$.

The leaders are currently as follow :

$$\alpha = \begin{bmatrix} 1.5 \\ 32.7 \\ 27 \end{bmatrix}, \beta = \begin{bmatrix} 3.2 \\ 4.0 \\ 11 \end{bmatrix}, \delta = \begin{bmatrix} 7.9 \\ 24.0 \\ 27 \end{bmatrix}, \rho 1 = \begin{bmatrix} 1.5 \\ 32.5 \\ 29 \end{bmatrix} \text{ and } \rho 2 = \begin{bmatrix} 2.8 \\ 47.9 \\ 69 \end{bmatrix}$$

The evolution of the wolf ω from the iteration iter to $\mathrm{iter}+1$ will be studied. At the iteration iter, the wolf ω is as follows :

$$\omega = \begin{bmatrix} 4.3 \\ 49.0 \\ 27 \end{bmatrix}$$

In the sections 2.1.4.1 and 2.1.4.2, the evolution of the wolf ω is studied in two different cases, respectively at $\mathrm{iter} = 20$ and $\mathrm{iter} = 75$.

2.1.4.1 Example 1 : at iteration index 20

In this example, the GWO algorithm is at the iteration $\mathrm{iter} = 20$, therefore $a = 2(1 - \frac{20}{100}) = 1.6$.
As $a > 1$, the algorithm is in the exploration phase, *i.e.* the five leaders will be used to update the position of the wolf ω.

Two values r and ϕ comprised between 0 and 1 are selected randomly. We got $r = 0.19$ and $\phi = 0.38$. Therefore, for the update of the discrete components, the wolf ω will be updated according to the leader β with a movement coefficient of $\Delta = 2$.

The update will start with the computation of the new component $K1$, which is a continuous component.
Two values per leader ($r_{1,l}$ and $r_{2,l}$) are randomly selected between 0 and 1 in order to compute y_1^{l}, the leaders' contributions for the component K_1.

The following values and the resulting contributions are obtained :

$$\begin{cases} r_{1,\alpha} = 0.47,\ r_{2,\alpha} = 0.39 \implies y_1^{\alpha} = 1.80048 \\ r_{1,\beta} = 0.48,\ r_{2,\beta} = 0.59 \implies y_1^{\beta} = 3.233536 \\ r_{1,\delta} = 0.80,\ r_{2,\delta} = 0.97 \implies y_1^{\delta} = -2.68496 \\ r_{1,\rho 1} = 0.62,\ r_{2,\rho 1} = 0.71 \implies y_1^{\rho 1} = 0.66672 \\ r_{1,\rho 2} = 0.24,\ r_{2,\rho 2} = 0.56 \implies y_1^{\rho 2} = 3.768448 \end{cases}$$

Resulting in the computation of the new value of the component $K1$ of the wolf ω, at $\mathrm{iter}+1$:

$$x_1(\text{iter}+1) = \frac{y_1^{\alpha}+y_1^{\beta}+y_1^{\delta}+y_1^{\rho 1}+y_1^{\rho 2}}{5} = 1.3568448$$

Then, the algorithm has to update the component $K2$ of the wolf ω. The same calculation as for the component $K1$ will be used but with new values of $r_{1,l}$ and $r_{2,l}$.

The following values and the resulting contributions are obtained :

$$\begin{cases} r_{1,\alpha} = 0.66,\ r_{2,\alpha} = 0.33 \implies y_2^{\alpha} = 23.851616 \\ r_{1,\beta} = 0.83,\ r_{2,\beta} = 0.24 \implies y_2^{\beta} = 1.48672 \\ r_{1,\delta} = 0.24,\ r_{2,\delta} = 0.83 \implies y_2^{\delta} = 53.56928 \\ r_{1,\rho 1} = 0.24,\ r_{2,\rho 1} = 0.29 \implies y_2^{\rho 1} = 44.6056 \\ r_{1,\rho 2} = 0.36,\ r_{2,\rho 2} = 0.57 \implies y_2^{\rho 2} = 70.437088 \end{cases}$$

Resulting in the computation of the new value of the component $K2$ of the wolf ω :

$$x_2(\text{iter}+1) = \frac{y_2^{\alpha}+y_2^{\beta}+y_2^{\delta}+y_2^{\rho 1}+y_2^{\rho 2}}{5} = 38.7900608$$

Finally, the component $K3$ has to be updated. As it is a discrete component, the method described in Section 2.1.1 will be used.

Prior to the update of the components of the wolf ω, the leading leader and the movement coefficient have been selected, in this case the selected leader is β and the movement coefficient is $\Delta = 2$ (see Eq. (2.5)).

In the current state, $h_3^{\omega}(\text{iter}) = 2$ and $h_3^{\beta}(\text{iter}) = 1$ therefore, the new value of the component $K3$ of the wolf ω become :

$$h_3^{\omega}(\text{iter}+1) = (2 + \Delta sgn(1-2))\ mod\ 7 = 7 \implies \omega_3(\text{iter}+1) = 87$$

At the end of this iteration, the position of the wolf ω become :

$$\omega = \begin{bmatrix} 1.3568448 \\ 38.7900608 \\ 87 \end{bmatrix}$$

2.1.4.2 Example 2 : at iteration index 75

In this example, the GWO algorithm is at the iteration $\text{iter} = 75$, therefore $a = 2(1 - \frac{75}{100}) = 0.5$.
As $a \leq 1$, the algorithm is in the exploitation phase, *i.e.* only the leaders α, β and δ will be used to update the position of the wolf ω.

Two values r and ϕ comprised between 0 and 1 are selected randomly. We got $r = 0.86$ and $\phi = 0.22$. Therefore, for the update of the discrete components, the wolf ω will be updated according to the leader α with a movement coefficient of $\Delta = 4$ (see Eq. (2.5)).

The update will start with the computation of the new component $K1$, which is a continuous component.
Two values per leader ($r_{1,l}$ and $r_{2,l}$) are randomly selected between 0 and 1 in order to compute y_1^{l}, the leaders' contributions for the component $K1$.

The following values and the resulting contributions are obtained :

$$\begin{cases} r_{1,\alpha} = 0.47,\ r_{2,\alpha} = 0.39 \implies y_1^{\alpha} = 1.5939 \\ r_{1,\beta} = 0.48,\ r_{2,\beta} = 0.59 \implies y_1^{\beta} = 3.21048 \\ r_{1,\delta} = 0.80,\ r_{2,\delta} = 0.97 \implies y_1^{\delta} = 4.5922 \end{cases}$$

Resulting in the computation of the new value of the component K_1 of the wolf ω :

$$x_1(\text{iter}+1) = \frac{y_1^{\alpha}+y_1^{\beta}+y_1^{\delta}}{3} = 3.13219333$$

Then, the algorithm has to update the component $K2$ of the wolf ω. The same calculation as for the component $K1$ will be used but with new values of $r_{1,l}$ and $r_{2,l}$.

The following values and the resulting contributions are obtained :

$$\begin{cases} r_{1,\alpha} = 0.66,\ r_{2,\alpha} = 0.33 \implies y_2^{\alpha} = 29.93488 \\ r_{1,\beta} = 0.83,\ r_{2,\beta} = 0.24 \implies y_2^{\beta} = 3.2146 \\ r_{1,\delta} = 0.24,\ r_{2,\delta} = 0.83 \implies y_2^{\delta} = 33.2404 \end{cases}$$

Resulting in the computation of the new value of the component $K2$ of the wolf ω :

$$x_2(\text{iter}+1) = \frac{y_2^{\alpha}+y_2^{\beta}+y_2^{\delta}}{3} = 22.12996$$

Finally, the component $K3$ has to be updated. As it is a discrete component, the method described in Section 2.1.1 will be used.

Prior to the update of the components of the wolf ω, the leading leader and the movement coefficient have been selected, in this case the selected leader is α and the movement coefficient is $\Delta = 4$.

In the current state, $h_3^{\omega}(\text{iter}) = 2$ and $h_3^{\alpha} = 2$ therefore, the new value of the component K_3 of the wolf ω becomes :

$$h_3^{\omega}(\text{iter}+1) = (2 + \Delta sgn(2-2)) \; mod \; 7 = 2 \implies x_3(\text{iter}+1) = 27$$

At the end of this iteration, the position of the wolf ω becomes :

$$\omega = \begin{bmatrix} 3.13219333 \\ 22.12996 \\ 27 \end{bmatrix}$$

2.2 Performance evaluation on synthetic data

In this section, we evaluate the performances of the proposed methods and compare it to the bio-inspired optimization methods from the state-of-the-art, when applied to the minimization of various unimodal and multi-modal test functions. Unimodal functions possess only one global minimum and no local minima, whereas multi-modal functions exhibit several local minima.

2.2.1 Experimental conditions

In this subsection, we present the experimental conditions which, unless specified, are common to the whole section 2.2. Unless specified, the results are computed over $M = 30$ independents runs, $Q = 30$ search agents and $\mathrm{T}_{\max} = 3000$ iterations for each algorithm. These experimental conditions are the same as in [95].

We have implemented the GWO, mGWO, and MODGWO (see [92] and Algorithm 2, and [80] and Section 1.4.2.2) comparative methods so that they can be compared with the proposed mixedGWO and amixedGWO in the same conditions. We present results obtained by GWO, mGWO, MODGWO, and of course the proposed mixed GWO (mixedGWO) and adaptive mixed GWO (amixedGWO). In the amixedGWO, the value for η is empirically set to $\eta = 3$, after performing a comparative evaluation with several values between 1 and 10. This value holds for the rest of the paper.

In section 2.2, programs were written in $C++$, and executed on a PC running Windows, with a 3.4GHz i5 core and 16GB RAM.

The performance metrics are the following :

We consider, for the m^{th} run, $f(\mathbf{x}_\alpha^\mathbf{k})_m$, the fitness value obtained at iteration $\mathrm{T}_{\max}$ for the best wolf, namely α :

• The statistical mean is the average (Avg.) of fitness values acquired from running an optimization algorithm for different M runs. The average performance of a given stochastic optimizer is formulated in Eq. (2.13) :

$$Avg = \frac{1}{M}\sum_{m=1}^{M} f(\mathbf{x}_\alpha^\mathbf{k})_m \tag{2.13}$$

• The standard deviation (Std.) is a representation for the variation of the obtained best solutions found for running a stochastic optimizer for M different runs. Std is used as an indicator for optimizer stability and robustness. If Std. is small, this means that the optimizer converges always towards the same solution. Conversely, if Std. is large, the results obtained are much more random and the optimizer is less reliable. The standard deviation is formulated in Eq. (2.14) :

$$Std = \sqrt{\frac{1}{M}\sum_{m=1}^{M}(f(\mathbf{x}_\alpha^\mathbf{k})_m - Avg)^2} \tag{2.14}$$

• The median (Med.) is the value separating the $\frac{M}{2}$ higher half from the $\frac{M}{2}$ lower half of values obtained for $f(\mathbf{x}_\alpha^\mathbf{k})$.

2.2.2 Benchmark functions

In this subsection, the benchmarks functions used for the rest of the section are presented.

Tests have been run on 20 benchmarks functions in order to compare the performance on a fully continuous problem of the proposed mixedGWO algorithm and its adaptive version amixedGWO to the vanilla GWO method and its modified version mGWO proposed by Mittal *et. al.* in [95].

The benchmarks functions have been taken from [95] and can be divided in 3 categories : the Unimodal functions, the Multi-modal functions and the fixed dimension Multi-modal functions. The functions' expressions are presented on Table 2.2, Table 2.3 and Table 2.4

and their meshes are available in the Appendix on the Figure A.-1. The unimodal functions exhibit only one minimum, which is the expected global minimum, while the multi-modal functions exhibit several minima, among them the expected global minimum.

Doubting from the expressions of the functions F12, F13 and F19 presented in [95], we do not present results related to these functions. For instance, we do not know the value of the constant used in function F19.

Function	Dim	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]$	0
$F_4(x) = max\{\lvert x_i \rvert,\ 1 \leq i \leq n\}$	30	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + \left(x_i - 1 \right)^2 \right]$	30	$[-30, 30]$	0
$F_6(x) = \sum_{i=1}^{n} \left(x_i + 0.5 \right)^2$	30	$[-100, 100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random(0, 1)$	30	$[-1.28, 1.28]$	0

Table 2.2 — Unimodal benchmark functions

Function	Dim	Range	f_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	$[-500, 500]$	$-418.9829 \times n$
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10 \right]$	30	$[-5.12, 5.12]$	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^n} \right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos\left(2\pi x_i\right) \right) + 20 + e$	30	$[-32, 32]$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]$	0

Table 2.3 — Multi-modal benchmark functions

Function	Dim	Range	f_{min}
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	2	$[-65, 65]$	1
$F_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x1\left(b_i^2 + b_i x_2\right)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	$[-5, 5]$	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + x_2^4$	2	$[-5, 5]$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 5]$	0.397887
$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right]$ $\left[30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	$[-2, 2]$	3
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	6	$[0, 1]$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.5363

Table 2.4 — Fixed dimension multi-modal benchmark functions

2.2.3 Numerical results on synthetic data

In this subsection, the numerical results of our methods and the state-of-the-art are presented. The subsections 2.2.3.1, 2.2.3.2 and 2.2.3.3 are respectively the results on the continuous functions, the mixed functions and the discrete functions.

2.2.3.1 Results on continuous functions

The results we have computed are available in Table 2.5 for the unimodal functions, Table 2.6 for the multi-modal functions and Table 2.7 for the fixed dimension multi-modal functions. Although the average found minimum are close to each other, it can be seen in the Tables 2.5 and 2.6 that the vanilla GWO method and the mGWO method remain slightly better than our method for the Unimodal functions and the Multi-modal functions. However, it can be seen in Table 2.7 that the proposed mixedGWO and amixedGWO methods outperform the state-of-the-art methods for the fixed dimension multi-modal functions.

F		GWO	mGWO	mixedGWO	amixedGWO
F_1	*Avg.*	$1.08e-205$	$\mathbf{3.03e-263}$	$1.17e-177$	$2.50e-115$
	Std.	0	0	0	$6.53e-115$
	Rank	2	1	3	4
F_2	*Avg.*	$1.15e-118$	$\mathbf{1.30e-152}$	$2.80e-104$	$6.55e-66$
	Std.	$3.62e-118$	$2.79e-152$	$7.11e-104$	$6.78e-66$
	Rank	2	1	3	4
F_3	*Avg.*	$6.23e-41$	$\mathbf{8.60e-53}$	$2.83e-41$	$4.24e-31$
	Std.	$3.24e-40$	$4.69e-52$	$1.52e-40$	$2.31e-30$
	Rank	3	1	2	4
F_4	*Avg.*	$1.69e-40$	$\mathbf{2.97e-58}$	$3.38e-38$	$3.17e-22$
	Std.	$7.18e-40$	$7.15e-58$	$8.43e-38$	$5.92e-22$
	Rank	2	1	3	4
F_5	*Avg.*	**26.336**	26.375	27.248	26.556
	Std.	0.843	0.743	0.959	0.674
	Rank	1	2	4	3
F_6	*Avg.*	0.487	0.495	1.916	0.834
	Std.	0.282	0.238	0.524	0.382
	Rank	1	2	4	3
F_7	*Avg.*	$3.06e-04$	$\mathbf{2.09e-04}$	$3.10e-04$	$2.27e-03$
	Std.	$1.8e-04$	$1.33e-04$	$1.75e-04$	$8.78e-04$
	Rank	2	1	3	4
Average Rank		1.86	1.29	3.14	3.71
Overall Rank		2	1	3	4

Table 2.5 — Results of unimodal continuous benchmark functions with $T_{max} = 3000$

F		GWO	mGWO	mixedGWO	amixedGWO
F_8	*Avg.*	-6311.745	$\mathbf{-6378.536}$	-5970.654	-5783.265
	Std.	1053.778	603.356	835.292	555.451
	Rank	2	1	3	4
F_9	*Avg.*	**0**	**0**	**0**	**0**
	Std.	0	0	0	0
	Rank	1	1	1	1
F_{10}	*Avg.*	$5.77e-15$	$\mathbf{4.47e-15}$	$6.84e-15$	$7.31e-15$
	Std.	$1.81e-15$	$1.23e-15$	$1.45e-05$	$9.01e-16$
	Rank	2	1	3	4
F_{11}	*Avg.*	$1.12e-03$	**0**	$4.51e-03$	$7.01e-03$
	Std.	$4.58e-03$	0	$8.52e-03$	$7.95e-03$
	Rank	2	1	3	4
Average Rank		1.75	1	2.5	3.25
Overall Rank		2	1	3	4

Table 2.6 — Results of multi-modal benchmark continuous functions with $T_{max} = 3000$

F		GWO	mGWO	mixedGWO	amixedGWO
F_{14}	*Avg.*	4.1333240	2.9999936	4.9999841	**2.8362586**
	Std.	4.6068296	3.8506485	5.0854562	3.6191509
	Rank	3	2	4	1
F_{15}	*Avg.*	$3.09e-03$	$3.68e-03$	$4.40e-03$	**$9.90e-04$**
	Std.	$6.90e-03$	$7.59e-03$	$8.12e-03$	$3.66e-03$
	Rank	2	3	4	1
F_{16}	*Avg.*	-1.03162845222	-1.03162845227	-1.03162845241	**-1.03162845331**
	Std.	$3.19e-10$	$4.82e-10$	$2.02e-09$	$2.04e-09$
	Rank	4	3	2	1
F_{17}	*Avg.*	0.3979427	0.3979661	0.3979500	**0.3979117**
	Std.	$6.52e-05$	$8.18e-05$	$5.81e-05$	$2.59e-05$
	Rank	2	4	3	1
F_{18}	*Avg.*	5.7000011	**3.0000007**	**3.0000007**	3.0000010
	Std.	14.7885089	$9.36e-07$	$7.40e-07$	$1.02e-06$
	Rank	4	1	1	3
F_{20}	*Avg.*	-3.2563368	-3.2635423	**-3.2980638**	-3.2780500
	Std.	0.0635625	0.0645893	0.0486790	0.0600625
	Rank	4	3	1	2
F_{21}	*Avg.*	-9.8163635	-9.2886724	-9.9847094	**-9.9847752**
	Std.	1.2818425	1.9692828	0.9224400	0.9224418
	Rank	3	4	2	1
F_{22}	*Avg.*	-10.2257553	-10.2257344	-10.4028815	**-10.4029327**
	Std.	0.9704291	0.9704252	$4.02e-05$	$5.98e-06$
	Rank	3	4	2	1
F_{23}	*Avg.*	-10.1758610	-10.0856251	-10.5363495	**-10.5364004**
	Std.	1.3720325	1.7518213	$6.00e-05$	$7.40e-06$
	Rank	3	4	2	1
Average Rank		3.11	3.11	2.33	1.33
Overall Rank		3	3	2	1

Table 2.7 — Results of fixed dimension multi-modal benchmark continuous functions $T_{max} = 3000$

The Figure 2.3 showcases the convergence plot of the four studied methods on some of the fixed dimension multi-modal functions, *e.g.* $F20$ to $F23$. Starting the 1500th iteration, the proposed method amixedGWO's curve remains constant for the functions $F21$, $F22$ and $F23$, that is because the variations are on a scale inferior to 10^{-4}, therefore unseeable on the curves.

In all of the cases, either the mixedGWO or the amixedGWO converges faster than the state-of-the-art methods.

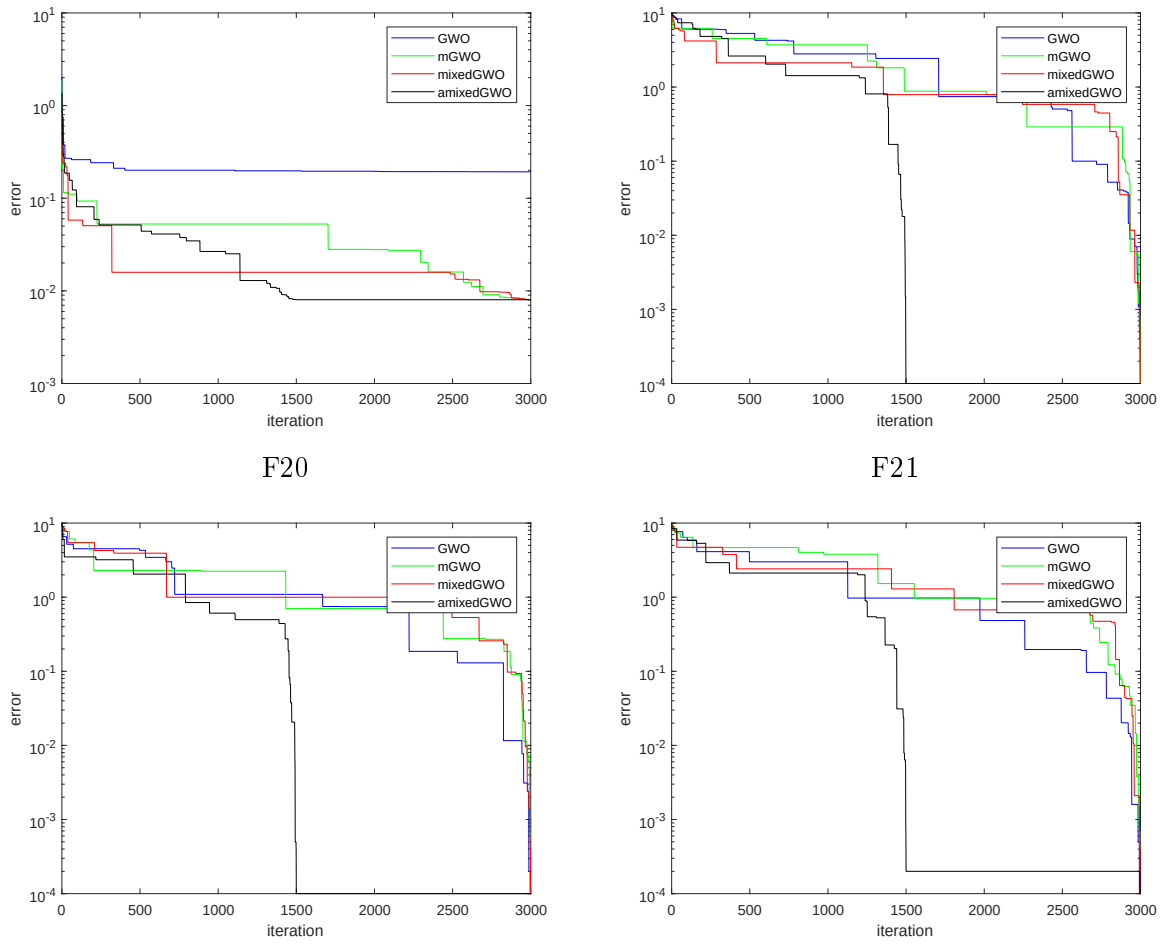


Figure 2.3 — Convergence plot of the functions 'F20', 'F21', 'F22' and 'F23'

To statistically validate the results obtained on Table 2.7 with the fixed-dimension multi-modal benchmark functions, the Wilcoxon test is used. The obtained p-value p for the continuous functions and the discrete functions are displayed on Table 2.8. The level of significance is set as $p_s = 0.05$. A $(+)$ means that the first compared method is better than the second, a $(-)$ means that the second compared method is better than the first and a $(=)$ means that there is no significant difference between the two compared methods.

Function	mixedGWO vs GWO	mixedGWO vs mGWO	amixedGWO vs GWO	amixedGWO vs mGWO
F_{14}	0.057 (=)	0.277 (=)	0.154 (=)	0.631 (=)
F_{15}	0.936 (=)	0.612 (=)	0.485 (=)	0.467 (=)
F_{16}	$1.94e-08$ (+)	$4.11e-03$ (+)	$4.20e-11$ (+)	$9.04e-07$ (+)
F_{17}	0.046 (−)	0.644 (=)	0.0345 (+)	0.406 (=)
F_{18}	0.45 (=)	0.096 (=)	0.0120 (+)	$2.41e-05$ (−)
F_{20}	0.21 (=)	0.959 (=)	0.164 (=)	0.020 (+)
F_{21}	$3.06e-08$ (+)	0.018 (+)	$3.13e-09$ (+)	$3.06e-08$ (+)
F_{22}	$1.43e-09$ (+)	$2.34e-03$ (+)	$2.48e-12$ (+)	$1.43e-09$ (+)
F_{23}	$2.16e-11$ (+)	0.0243 (+)	$4.84e-13$ (+)	$3.14e-07$ (+)
$+/=/-$	4/4/1	4/5/0	6/3/0	5/3/1

Table 2.8 — Results of the wilcoxon test on the fixed-dimension multi-modal functions in continuous search space

According to the p-values obtained with the Wilcoxon test, it appears that the proposed method is usually better than the GWO and the mGWO methods on the fixed dimension multi-modal benchmark functions. Indeed it is significantly better in 19 cases out of 36 and only significantly worse in 2 cases.

The proposed method amixedGWO has also been tested on the CEC2014 benchmark functions [70], except the hybrid functions, and compared to state-of-the-art methods such as PSO, GWO, ABC, TSA and GA. The average residual obtained are available on Table 2.9. The tests have been done in the same conditions as previously, that is $Q = 30$ and $T_{max} = 3000$.

These results follow the logic of what has been noted previously : the proposed mixedGWO is not the best method on strictly continuous unimodal and multi-modal problems. However, the proposed method is not meant for these kinds of problems but rather for the fixed-dimension multi-modal ones and more particularly for the mixed problems. Moreover, it has been noted that, in these conditions, the proposed method is at least twice faster than the compared methods, for the same number of iterations and for the same number of search agents.

Function	amixedGWO	PSO	GWO	ABC	TSA	GA
F_1	3070.2	462.8	1534.9	434.4	**45.6**	152.4
F_2	396.6	514.7	1126.2	63	**25.5**	508.5
F_3	542.5681	342.5489	684.1074	96.8305	**21.4919**	162.4
F_4	$1.00e-04$	**0**	$4.6e-03$	**0**	**0**	$1.14e-13$
F_5	$9.6e-03$	**0**	$6.84e-01$	**0**	**0**	$1.7e-13$
F_6	$3.8e-03$	**0**	$8.2e-03$	**0**	**0**	$2.27e-13$
F_7	$4.2e-03$	$4.8e-03$	$9.5e-03$	$3.5e-03$	**0**	$8.4e-03$
F_8	$1.33e-01$	**0**	$7.18e-02$	**0**	**0**	$2.27e-13$
F_9	$1.99e-01$	**0**	$9.96e-02$	**0**	**0**	$3.41e-13$
F_{10}	49.2603	$1.69e-01$	30.30	$1.02e-01$	**0**	$6.24e-3$
F_{11}	4.0561	$2.39e-01$	4.0174	$9.89e-02$	**4.8e−02**	3.5807
F_{12}	$4.96e-01$	$1.12e-01$	$3.73e-01$	$4.49e-01$	$9.39e-02$	**5.12e−13**
F_{13}	$3.38e-02$	$3.81e-02$	$4.18e-02$	$4.78e-02$	**1.41e−02**	$2.75e-02$
F_{14}	$7.5e-03$	$1.32e-02$	$1.14e-02$	$2.49e-02$	**3.5e−03**	$1.67e-02$
F_{15}	$2.24e-02$	**0**	$1.04e-02$	$2.00e-04$	**0**	$1.38e-02$
F_{16}	$6.9e-03$	$1.09e-02$	$1.10e-02$	$1.15e-02$	**0**	$1.94e-02$
F_{23}	99.0327	33.6887	112.9178	39.3915	**7.56**	160.4773
F_{24}	80.9750	89.8178	69.7263	57.1990	**19.6872**	100.6363
F_{25}	$1.60e-02$	$1e-04$	$4.23e-02$	$4.2e-03$	**0**	47.9145
F_{26}	$2.47e-01$	$3.81e-01$	$1.53e-01$	$1.09e-01$	**0**	1.1451
F_{27}	$2.08e-01$	$5.37e-02$	$4.18e-01$	**1.02e−01**	$5.5e-03$	$5.96e-01$
F_{28}	260.5198	28.8892	297.3808	107.5971	**9.4956**	176.3242

Table 2.9 — Average residual errors on the CEC2014 functions

2.2.3.2 Results on discrete functions

The Unimodal functions from F_1 to F_6 will be used to test the proposed method on discrete functions.

To be as close as possible to the multidimensional image denoising application described further in this thesis, there will only be 3 variables to be close to the multidimensional image processing issue considered further. The searching space used are the same as in the study on mixed functions.

The two proposed methods mixedGWO and amixedGWO are compared with the MODGWO method, which is presented in Subsection 1.4.2.2. The results obtained for each algorithm are presented in Table 2.10. For each method, in addition to the average result and the standard deviation, the medium value obtained is also indicated.

The Ant Colony Optimization method has not be used for the comparison. Indeed, as pointed out in Subsection 1.4.2.1, it is designed for combinatorial optimization problems such as Traveling Salesman Problem (TSP). In the benchmark problem that are used in this section and the studied problem studied in Section 3.3, it is possible to have a solution $\{\hat{K}_1, \hat{K}_2, \hat{K}_3\}$ with $\hat{K}_1 = \hat{K}_2$. A solution which can not be obtained with the ACO algorithm. Conversely, the proposed mixedGWO is not compatible with a combinatorial optimization problem. Indeed, a problem such as TSP implies that the searching space will shrink through the iteration, which is a case not taken in account in the proposed method, e.g. in a TSP problem composed of 4 cities, the proposed method would find a solution such as $\hat{\mathbf{K}} = [3, 3, 3, 3]^T$.

The results shown in Table 2.10 confirm that the proposed methods outperform the state-of-the-art method on discrete problems. Moreover, as indicated by the medium value, the optimal minimum has been found on all the benchmark functions on at least 50% of the runs, even on the function F_5, which was not the case on the continuous and mixed functions.

F	step	MODGWO			mixedGWO			amixedGWO		
		Avg.	Std.	Med.	Avg.	Std.	Med.	Avg.	Std.	Med.
F_1	1	8.100	5.880	6	**0**	**0**	**0**	**0**	**0**	**0**
F_2	0.5	0.183	0.278	**0**	**0**	**0**	**0**	**0**	**0**	**0**
F_3	1	7.100	4.428	5	0.067	0.254	**0**	**0**	**0**	**0**
F_4	1	2	0.695	2	0.067	0.254	**0**	**0**	**0**	**0**
F_5	0.5	23.217	17.849	18	4.650	8.394	**0**	**2.583**	**5.173**	**0**
F_6	0.5	8.692	6.060	7.5	**0**	**0**	**0**	**0**	**0**	**0**

Table 2.10 — Results of unimodal benchmark functions in discrete search space with $T_{max} = 3000$

The Table A.1, available in the Appendix, showcases the results obtained with the MODGWO methods with much more iteration. From these results, it is shown that the MODGWO methods need an elevated number of iterations in order to converge and approach the optimal minimum still without reaching it.

To statistically validate the results obtained on Table 2.10 with the discrete benchmark functions, the Wilcoxon test is used. The obtained p-value for the discrete benchmarks functions are displayed on Table 2.11. The level of significance is set as $p = 0.05$. A $(+)$ means that the first compared method is better than the second, a $(-)$ means that the second compared method is better than the first and a $(=)$ means that there is no significant difference between the two compared methods.

Function	mixedGWO vs HMOGWO	amixedGWO vs HMOGWO
F_1	$4.12e-12(+)$	$4.12e-12(+)$
F_2	$1.09e-02(+)$	$1.09e-02(+)$
F_3	$9.92e-13(+)$	$1.96e-12(+)$
F_4	$4.62e-12(+)$	$2.90e-13(+)$
F_5	$5.46e-11(+)$	$8.05e-10(+)$
F_6	$1.18e-12(+)$	$1.18e-12(+)$
$+/=/-$	6/0/0	6/0/0

Table 2.11 — Results of the wilcoxon test on discrete functions

According to the p-values obtained with the Wilcoxon test, it appears that the differences in the average values between the HMOGWO method and the proposed methods are significant. However, the two proposed methods mixedGWO and amixedGWO seems to not hold significant difference between them.

2.2.3.3 Results on mixed functions

The Unimodal functions from F_1 to F_6 will be used to test the mixedGWO method on mixed functions.

To be as close as possible to the considered image processing application described further in this thesis, there will only be 6 variables instead of 30. Out of those 6 variables, 4 will be discrete and 2 continuous. The searching space is always in the same range as in the continuous case. For the discrete values, There will be a step of 1 in between each possible value for the functions F_1, F_3 and F_4 while the step will be of 0.5 for the functions F_2, F_5 and F_6.

Only the two proposed methods mixedGWO and amixedGWO will be tested here, because the comparative methods PSO, GWO, mGWO, and MOGWO can not tackle mixed problems. The results obtained for each algorithm are presented in Table 2.12. For each method, in addition to the average result and the standard deviation, the medium value obtained is also indicated.

The medium values obtained indicate that the proposed methods have found the optimal minimum value in at least 50% of the runs on the benchmark functions F_1 to F_4 and converge correctly on the function F_6. However, the proposed methods have difficulties to converge correctly on the function F_5, which is in accordance with the results obtained on the continuous benchmark functions in section 2.2.3.1.

F	step	mixedGWO			amixedGWO		
		Avg.	Std.	Med.	Avg.	Std.	Med.
F_1	1	**0**	**0**	**0**	**0**	**0**	**0**
F_2	0.5	**0**	**0**	**0**	**0**	**0**	**0**
F_3	1	0.100	0.257	**0**	**0**	**0**	**0**
F_4	1	0.033	0.182	**0**	**0**	**0**	**0**
F_5	0.5	6.351	8.749	**2.771**	**4.084**	**5.318**	**2.771**
F_6	0.5	$2.93e-13$	$9.18e-13$	$4.74e-14$	**6.39e − 14**	**1.00e − 13**	**1.61e − 14**

Table 2.12 — Results of unimodal benchmark functions in mixed search space with $T_{max} = 3000$

2.3 Conclusion

In this Chapter, a novel bio-inspired optimization method designed for mixed problems, *i.e.* problems which hold both continuous and discrete variables, has been presented and detailed. This method, based on the Grey Wolf Optimizer (GWO) algorithm, is named mixedGWO.

The mixedGWO and its adaptive version amixedGWO have been compared to the vanilla GWO and the mGWO on fully continuous problems, then compared to the MODGWO on fully discrete problems. Finally, the mixedGWO and amixedGWO have been tested on custom problems.

The proposed methods got good results on both the discrete and mixed problems (see Tables 2.10 and 2.12). However, it is not far better than the state-of-the-art methods on continuous problems, except on the fixed dimension multi-modal functions, as shown in Tables 2.5, 2.6 and 2.7. The results obtained on the CEC2014 benchmarks functions, which do not contain any fixed-dimension multi-modal function, share the same observation, as shown in Table 2.9.

As the "No Free Lunch" theorem states [118], it is impossible for a sole optimization algorithm to tackle all the existing problems. Therefore, it is normal that the proposed mixedGWO is not able to tackle every problems better than the other existing methods. However, it is a fact that the proposed mixedGWO is best when used on fixed-dimension multi-modal problems in the continuous space, which is the case for the main target applications for the proposed algorithm : image classification by SVM and joint denoising and unmixing of multi-spectral and hyper-spectral images. Therefore, it seems beforehand that the proposed mixedGWO should be well adapted for these 2 application.

The application of the mixedGWO on those two real problems is described in the remainder of this manuscript, in Chapter 3.

CHAPTER 3

Applications : The ISAM algorithm and joint denoising/unmixing of Multi-Spectral Images

3.1 The IntuiSense Audience Metrics (ISAM) algorithm

Audience Measurement is meant to collect data in order to profile customers, e.g. the number of customers, their gender (man/woman), their age or their presence time in the studied environment. Such a tool is useful either for marketing or anthropological studies. Developed during this thesis for the company IntuiSense, the IntuiSense Audience Metrics (ISAM) tool is an audience measurement tool which is based on computer vision and designed to work with a camera embedded at the top of a vending machine.

The ISAM tool aims to count and characterize the people appearing in the camera's field of view using face detection and face recognition algorithms. Existing algorithms are usually performed in an ideal environment and/or have their performances measured on ideal databases. Moreover, to do face recognition, it is highly recommended to use a high quality 3D camera in order to get optimal results. Although existing algorithms got good results, even on non ideal databases, the problem of computational cost is still mainly present. According to the state of the art about face detection/recognition and the industrial constraints, the following points can be foreseen as problems which have been taken into account for the ISAM tool development :

— The images' background and the global illumination are unknown and uncontrollable,
— The faces to detect are not exactly in front of the camera and rarely look directly at it,
— The low cost hardware (processor and camera) implies limited computational capacities,
— Real-time performances are targeted.

As of now, the ISAM algorithm is able to count the people in the camera field of view and to measure the amount of time each one of them spent in the field of view. The algorithm's workflow is divided in 2 parts : a detection part and a tracking part. The main workflow of the ISAM algorithm is shown on Figure 3.1. The algorithm works on a camera stream and works on each frame one after the other in an independent manner. The only information saved between each frame is the detections' position on the previous studied frame. This software has been fully developped in C++ and uses the computer vision library OpenCV [1].

The detection part and the tracking part are further detailed in section 3.1.1 and section 3.1.2 respectively.

In the rest of this section, *previous detections* will stand for the detection obtained prior to the studied frame t, and *current detections* will stand for the detection obtained on the studied frame t. Moreover, a detection position must not be seen only as coordinates but as a complete rectangle circling the detected face.

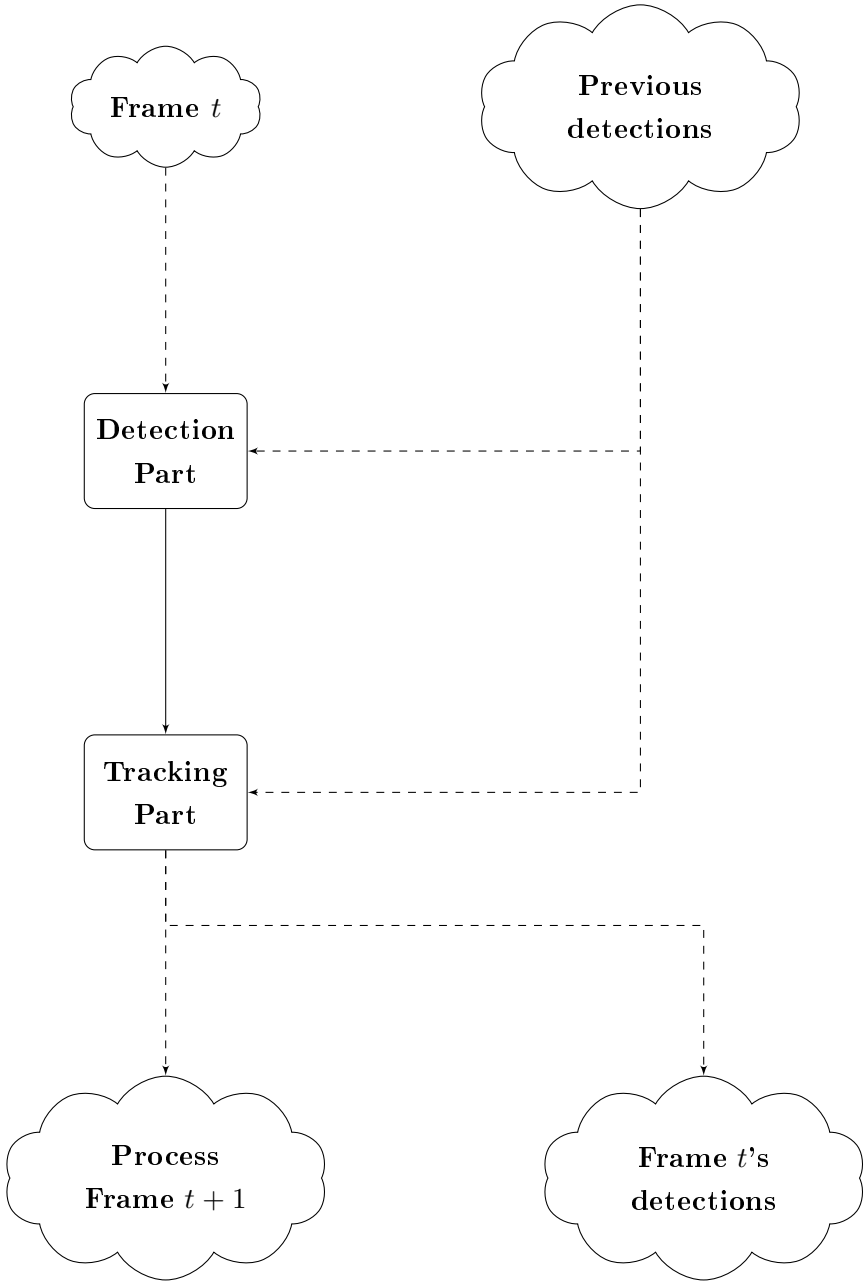


Figure 3.1 — The ISAM algorithm's main workflow

3.1.1 The ISAM algorithm : Detection part

The purpose of the detection part is to detect the faces present in the studied frame t, using the Viola-Jones method. Indeed, as said previously in section 1.1.1, the Viola-Jones algorithm has shown good results in real-time cases and seems adequate for an embedded context, as its complexity is mainly dependent on the studied image size [55]. The workflow

of the ISAM algorithm's detection part is shown of the Figure 3.2.

In order to reduce the algorithm's computational cost, the Viola-Jones detector will be applied on the whole frame only on one frame out of five. On the remaining four, the Viola-Jones detector will only be applied on regions of interest (ROI) selected according to the detections previously known. This step is illustrated on Figure 3.3. This reduction of the searching zone for the Viola-Jones detector permits to reduce its computational cost by a factor two.

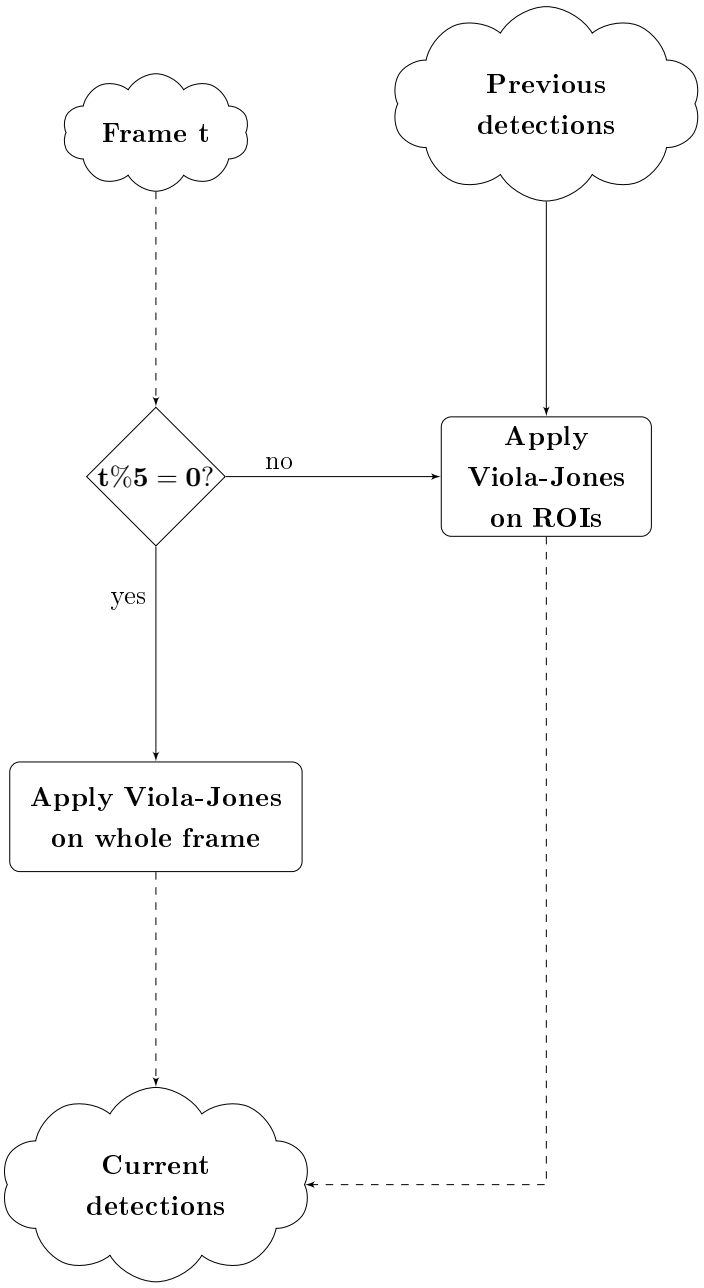


Figure 3.2 — The detection part's workflow

Figure 3.3 — Viola-Jones on ROIs

3.1.2 The ISAM algorithm : Tracking part

After a frame has passed the detection stage, it must go through the tracking one. As the first purpose of the ISAM software is to count people, it must be able to differentiate these people between them, in order to know when one of them leaves the field of view or if a new person comes in.
The method used is based on the Kanade-Lucas-Tomasi (KLT) tracker, based itself on the Lucas-Kanade (LK) Optical Flow explained in subsection 1.2.2 and the Kanade-Tomasi features method [112]. The KLT tracker is convenient for multi-object tracking. The proposed algorithm's flowchart is available on Figure 3.4.

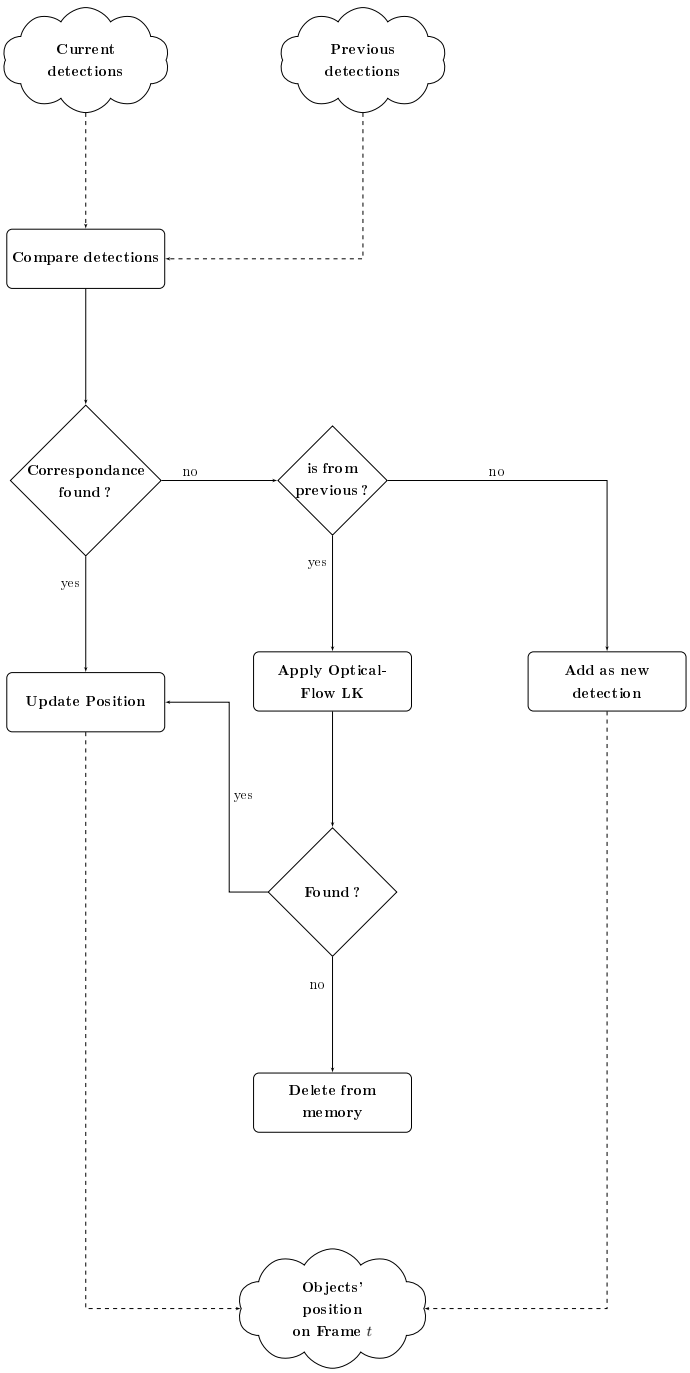


Figure 3.4 — The Tracking part's workflow

At each frame, the algorithm will first compare the position of the current and previous detections to each other. To do so, the area of the intersection S between two detections's position (one current and one previous) is computed. If this area S is high enough, *i.e.* at least 20% of the previous detection's surface, then the current detection's position is considered as the previous detection's new position. This step is illustrated on Figure 3.5.

Figure 3.5 — Spatial comparison between previous detection i and current detection j

After all the correspondences have been found, the algorithm has to handle the detections (previous as well as current) which do not have a correspondence.
In the case of the current detections without correspondence, the algorithm will consider them as new detections and will add them to the previous detections for the upcoming frame $t+1$.
In the case of the previous detections without correspondence, the Lucas-Kanade Optical Flow [81] will be used to retrieve the detection's new position. The keypoints to track are a regular grid paste on the previous detection. The idea of this grid of keypoints is similar to the one used in [51] except that a space of a length r is put between the right and left sides of the detection and the right and left sides of the grid, such as $r = \frac{w}{Y}$ with Y a constant integer and w the detection's width. This grid, even if less precise than most automatic keypoints detector such as SIFT, SURF or HOGs is well-adapted to this problematic because it does not need a lot of computational load and is fast to apply for each detection. Moreover, the metrics measured with it are good, as explained later in subsection 3.1.4.
Using the Optical flow to follow these keypoints' movement between frame $t-1$ and t, the algorithm is able to update the new position of the studied previous detection. However, if

the keypoints are not found in the frame t, then the detection is considered as disappeared and is therefore deleted from the memory.

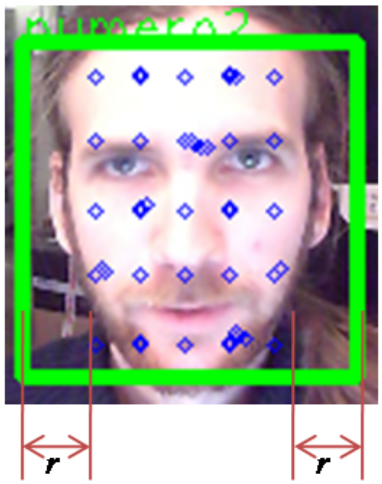


Figure 3.6 — Grid of Keypoints used for tracking in ISAM

3.1.3 Potential user detection

By knowing how many time a person has spent using the vending machine, flaws in the vending machine Human-Machine Interface (HMI) could be detected. An additionnal functionnality permitting to determine which person, among the detected people at frame t, has the highest probability of being the current user of the vending machine has been added to the ISAM tool. To do this, the Eq. (3.1) is used.

$$P(D = user) = A \times B \times C \tag{3.1}$$

with

$$A = \left(1 - \frac{D_x - \frac{w_{tot}}{2}}{\frac{w_{tot}}{2}}\right) \tag{3.2}$$

and

$$B = \left(1 - \frac{D_y - \left(h_{tot} - \frac{\bar{D_h}}{2}\right)}{h_{tot} - \frac{\bar{D_h}}{2}}\right) \tag{3.3}$$

and

$$C = \left(\frac{D_{area}}{Area_{max}}\right) \tag{3.4}$$

Where D is the studied detection, (D_x, D_y) are the detection's center coordinates, D_{area} is the detection's area, h_{tot} and w_{tot} are respectively the height and the width of the total image, $\bar{D_h}$ is the height of an average detection and $Area_{max}$ is the maximum area a

detection can have. The probability P is the most elevated when the detected face is the biggest, and at the bottom-center of the FOV. Thanks to this method, the software is able to determine which person in the video is the current user with as much precision as the human eye.

Some qualitative results of the potential user detection are available on Figure 3.7, where the green rectangles correspond to the person selected as the current user and the red rectangles as the other people detected. These results have been obtained directly from the vending machine's camera ((a) and (b)) and from the ChokePoint database [119] ((c) and (d)).

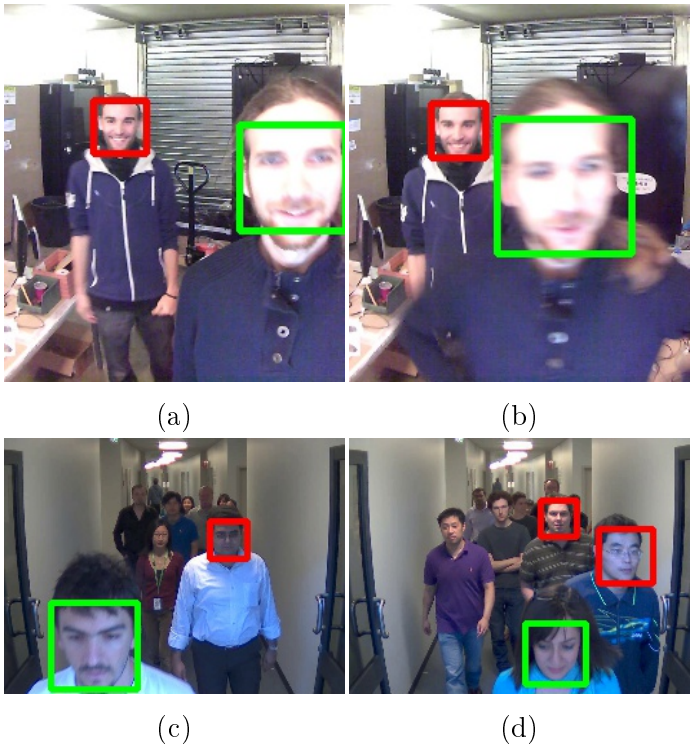


Figure 3.7 — User selection's qualitative results

3.1.4 ISAM's performances and future evolutions

To measure the accuracy of a multi-target detection and tracking algorithm on a video stream the Multi-Object Tracking Accuracy (MOTA) index [14] is used. This index, which goes from 0 to 100% is computed using Eq. (3.5).

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{3.5}$$

With m_t the number of objects missed, fp_t the number of false-positives and mme_t the number of mismatches all along the tested video stream.

The ISAM algorithm's performances have been measured using videos from the ChokePoint database [119]. These videos are interesting because the people in it behave almost like in front of the target vending machine :
• they move almost normally toward a camera which is put slighlty higher than their head,
• they do not watch towards the camera, as if they were not aware of it.
The $P2E\_S5\_C21$ and $P2L\_S5\_C21$ are especially challenging because the people do not appear one after another but altogether in a pack, which is a good way to measure the multi-target detection and tracking performances of the ISAM algorithm. The results obtained are available in Table 3.1.

Video's name	People counted	Real number of people	m_t	fp_t	mme_t	$MOTA\ index$
$P1E\_S1\_C1$	24	25	1	0	0	96.00%
$P1E\_S2\_C1$	20	25	5	0	0	80.00%
$P1L\_S3\_C3$	21	24	3	0	1	83.33%
$P2E\_S3\_C11$	19	24	5	0	0	79.17%
$P2E\_S5\_C21$	19	24	5	0	0	79.17%
$P2L\_S2\_C21$	23	25	2	0	0	92.00%
$P2L\_S4\_C21$	21	25	4	0	0	84.00%
$P2L\_S5\_C21$	23	25	2	0	0	92.00%

Table 3.1 — Results obtained on the videos from the ChokePoint database

The ISAM algorithm holds an average MOTA of 85.71% per tested video and a total MOTA of 85.79%. Moreover, when the ISAM algorithm is running on the target vending machine, which is running with a Windows 7 Embedded Operating System and equipped with a Celeron processor @2GHz and a RAM of 4GB. The algorithm only needs up to 30% of the maximum computational load and works with a camera frequency set at 15 frame per seconds (FPS).
At this point, the ISAM algorithm is able to count the people who pass through its field of view and to determine the time they spend in it with an accuracy of 85% while respecting a real-time functioning at 15 FPS. Some qualitative results of the ISAM algorithm, without the user selection part, obtained directly from the vending machine's camera stream and from the ChokePoint database are available on Figure 3.8.

Figure 3.8 — Qualitative results

In Figure 3.8, some of the faces are not detected. It can be due to :
• the people are too far from the camera, and therefore their faces are too small on the image ;
• the people appeared in the camera's field of view when the face detector was applied to the ROIs.
However, we notice that the proposed algorithm is not limited by the problem of false-positives.

3.2 Gender Classification optimized by Adaptive GWO

In this section, results on the application of the GWO method to optimize the training parameters of a SVM with a gaussian (RBF) kernel for gender recognition are presented. The training parameter of a SVM will greatly influence its performances [20], therefore, it is necessary to find the best ones.

The data to classify for this gender recognition purpose will be the LBP features of the faces extracted from the FERET database [103]. Some examples from the FERET database are showcased on Figure 3.9. The tests will be done with 200 training images and 500 testing images for each of the two genders (which makes 400 training images and 1000 testing images). The **fitness** of this problem, $i.e.$ the value we would like to minimize, will be the False Recognition Rate (FRR) which is computed by $FRR = \frac{nb_{fr}}{nb\_testing\_images}$ with nb_{fr} being the number of testing images recognized wrongly, $i.e.$ detected as man instead of woman and

vice-versa.

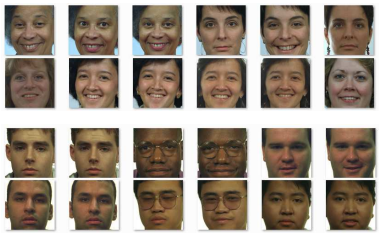


Figure 3.9 — Examples from the FERET database

In the case of a gaussian kernel SVM, there are 2 main parameters to determine, namely γ and C, which are both continuous parameters. To find these parameters in a non empirical manner is important because, as shown on the Table 3.2, a slight variation of only one of these parameters, can drastically change the resulting FRR. Therefore, an automatic method working directly on a continuous searching space is less prone than an empirical method to miss the possible optimal values.

C	γ	FRR
100	$1e-03$	50.00%
100	$1e-04$	18.50%
1	$1e-06$	26.10%
10	$1e-06$	13.60%

Table 3.2 — Empirical results on a RBF-SVM

As described previously in section 1.4.1.2, the GWO algorithm relies on the a parameter, which linearly decrease from 2 to 0. This parameter allows the algorithm's functioning to be split in two phases : the exploration phase (when $a > 1$) and exploitation phase (when $a \leq 1$). In the mGWO algorithm described in [95], the authors propose to change the update formula of a in order to decrease it without following a linear course and to emphasize more on the exploration phase. To do so, they added a power to the original formula a, becoming $a = 2(1 - \frac{\text{iter}^2}{T_{max}^2})$.

Following the idea of the mGWO, it should be possible to influence even more the repartition of the 2 phases. Therefore, a parameter η is added in the model so that :

$$a = 2(1 - \frac{\text{iter}^\eta}{{T_{max}}^\eta}) \tag{3.6}$$

With this parameter η, the amount of iterations spent on the exploration phase $t_{exploration}$ becomes :

$$t_{exploration} = \frac{T_{max}}{2^{\frac{1}{\eta}}} \tag{3.7}$$

Depending on the value of η, it is possible to freely decide if the GWO algorithm must emphasize exploration ($\eta > 1$), or exploitation phase ($\eta < 1$), or spend an equal amount of iterations on both phases ($\eta = 1$).

In order to exploit this idea of the η parameter, we propose an adaptive GWO (aGWO) algorithm, whose flowchart is described on Figure 3.10. The aGWO is simply the GWO algorithm run 3 times but each time with a different update rule for the a parameter. Moreover, between each run, the position of the best fitness found, $i.e.$ the position of the wolf α, will be the starting position of one wolf in the next run, all the other wolves being randomly placed in the searching space. The first run can be considered as a global research while the second and third runs are refined researches.

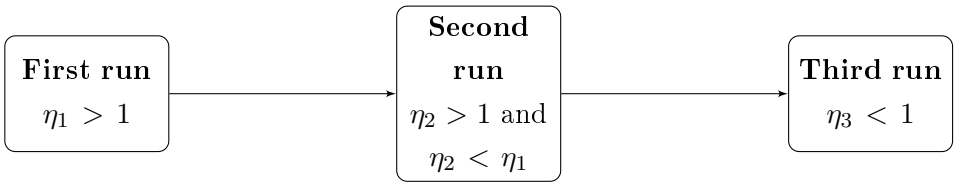


Figure 3.10 — aGWO flowchart

This configuration permits to have an alternation between exploration phases and exploitation phase. Therefore, it should be both more robust and more precise in cases where the classic GWO or the mGWO would not have enough time to converge to the best fitness.

The aGWO method has been tested on the problem described at the beginning of this section : the optimization of the training parameters of a SVM with a gaussian kernel (RBF) for gender recognition. In this test, 20 search agents and $\eta_1 = 8$, $\eta_2 = 2$ and $\eta_3 = 0.5$ were used. Furthermore, the aGWO was compared to the classic GWO , mGWO and PSO. In the aGWO case, each run will last 5 iterations ($i.e.$ 15 in total) while the state-of-the-art methods will last 15 iterations each.

Method	Best $(\gamma,\ C)$	$100 - FRR$	time (sec.)
PSO	$(6.7e-07,\ 600)$	91.4%	1970
GWO	$(1.6e-07,\ 800)$	91.5%	1970
mGWO	$(5.0e-08,\ 83)$	**91.9%**	1970
aGWO	$(4.9e-08,\ 82)$	**91.9%**	968

Table 3.3 — Estimated optimal parameters and required computational time

As shown in the Table 3.3, the classic GWO and the PSO methods are outperformed by the mGWO and the proposed aGWO. Moreover, the aGWO is twice faster than the mGWO for the same amount of iterations and the same resulting FRR.

From these results, three main pieces of information can be highlighted. Firstly, the GWO method showed good results on the studied problematic and therefore seems like a

good basis for our bio-inspired optimization for mixed problems. Secondly, working on the parameter η seems to be a good starting point. Indeed, mGWO and aGWO obtained the best results overall and aGWO even managed to outspeed the other methods by a factor of 2. Finally, the studied problem of the optimization of a SVM's training parameters for image classification seems to be a **fixed-dimension multi-modal** problem. Indeed, the number of parameter is fixed (*fixed-dimension*) and the problem seems to have several global minimum (*multi-modal*), as shown by the different results obtained by mGWO and aGWO in Table 3.3.

In this subsection, only the optimization of a SVM with a RBF kernel, $i.e.$ with only continuous parameters, has been studied. To go more in depth in this problem, it would be interesting to study other kernels such as the polynomial kernel. However, such a kernel require both continuous and discrete parameters, $i.e.$ a mixed problem. Therefore the proposed mixedGWO presented in section 2.1 and its adaptive version amixedGWO could be used to determine the optimal parameter of a SVM with a polynomial kernel and even determine which kernel would be the best one for a same classification problem.

3.3 Application to multidimensional image processing : joint denoising and unmixing of Multi-Spectral Images

3.3.1 Simultaneous denoising and unmixing issue

In this section, the mixed grey wolf optimizer described in Chapter 2 is used to perform a simultaneous denoising and unmixing of multispectral images.

3.3.1.1 Description of the proposed application

Hyperspectral and multispectral images are now currently used in remote sensing applications. In this context, devoted sensors have been developed, such as AVIRIS sensor (Airborne Visible/Infrared Imaging Spectrometer) [46], or ROSIS sensor (Reflective Optics System Imaging Spectrometer) [40]. A multispectral image can be obtained by selecting some important bands from the hyperspectral images obtained by these airborne sensors. Most of the multispectral aerial images are impaired by noise [73, 75] from solar radiation, or atmospheric scattering [58] for instance.
Denoising multispectral images is generally a preliminary step before higher level image processing operations such as target detection [73] or spectra unmixing [28]. One issue in this context consists in finding a good compromise between the efficiency of the image denoising process, and the accuracy of the forthcoming unmixing process. Let $\mathbf{y} \in \mathbb{R}^{I_3}$ be one spectrum

of the multispectral image $\mathcal{X}$. For this spectrum $\mathbf{y}$, supervised unmixing aims at estimating the contributions of the spectral signature of materials in the scene (called endmembers) for part of or all pixels in the scene. Note that 'supervised unmixing' means that the endmembers contained in the image have been estimated by an endmember extraction algorithm such as vertex component analysis [100].
An example of linear mixing model involving two endmembers is as follows :

$$\mathbf{y}(\lambda) = (1-\lambda)\mathbf{s1} + \lambda\mathbf{s2} + \mathbf{n} \tag{3.8}$$

where λ is a mixing coefficient, and $\mathbf{s1}$ and $\mathbf{s2}$ both $\in \mathbb{R}^{I_3}$ are the two endmembers. $\mathbf{n}$ is a noise vector, which follows a zero-mean Gaussian distribution.
The unmixing issue in the case of a linear mixing model is well-known and tackled by non-negative matrix factorization [102] in both supervised and unsupervised cases. We wish to exemplify the ability of our mixed GWO optimization method with non-linear mixing models. We assume that the endmember spectra are known, but we wish to determine, among two possible types of nonlinear mixing models, what is the model which fits the data.
The considered model for spectral mixture is denoted by $\mathbf{y}(f, \lambda_1, \lambda_2)$, where f is the mixing model, either f_0 of f_1.
The first model is the polynomial post-nonlinear mixing model [5] :

$$\mathbf{y}(f_0^{mix}, \lambda_1, \lambda_2) = f_0^{mix}(\lambda_1, \lambda_2) = \mathbf{g^{mix}}(\mathbf{s}(\lambda_1), \lambda_2) + \mathbf{n} \tag{3.9}$$

where $\mathbf{s} = [s_1, \ldots, s_{I_3}]^T$ follows a linear mixing model of the two endmembers $\mathbf{s1}$ and $\mathbf{s2}$:

$$\mathbf{s}(\lambda_1) = (1-\lambda_1)\mathbf{s1} + \lambda_1\mathbf{s2} \tag{3.10}$$

and $\mathbf{g^{mix}}$ is a second-order polynomial non linearity :

$$\begin{aligned} \mathbf{g^{mix}} : [0;1]^{I_3} &\to \mathbb{R}^{I_3} \\ \mathbf{s} &\mapsto \left[s_1 + \lambda_2 s_1^2, \ldots, s_{I_3} + \lambda_2 s_{I_3}^2\right]^T \end{aligned} \tag{3.11}$$

The second model is the generalized bilinear model [41] :

$$\mathbf{y}(f_1^{mix}, \lambda_1, \lambda_2) = f_1^{mix}(\lambda_1, \lambda_2) = (1-\lambda_1-\lambda_2)\mathbf{s1} + \lambda_1\mathbf{s2} + \lambda_2\mathbf{s1}\mathbf{s2} + \mathbf{n} \tag{3.12}$$

In Eqs. (3.9) and (3.12), λ_1 and λ_2 are in $[0;1]$. Moreover, $\mathbf{n}$ is an additive independent and identically distributed zero-mean Gaussian noise sequence. It is important to notice that both models reduce to a linear model if $\lambda_2 = 0$, such that they may be similar, particularly for small values of $\lambda_2 = 0$. In the polynomial post-nonlinear mixing model of Eq. (3.9), the non-linear terms come from second-order reflections. This model is also a simplified but reliable model for many types of non-linearities [5]. In the generalized bilinear model of Eq. (3.12), the non-linear terms come from multiple scattering of photons between the two components

s1 and **s2** [41].
By applying jointly the denoising and the supervised unmixing process to a multispectral image, the images could be specifically denoised in such a manner that the best possible unmixing results are attained. For this, it is of great importance to choose adequately a criterion to minimize.

3.3.1.2 Proposed criterion

With the amixedGWO method, we propose to minimize the following criterion :

$$J^{LS}(K_1, K_2, K_3, f, \lambda_1, \lambda_2) = \frac{1}{I_1 I_2 I_3}||\mathcal{X}_1 - \hat{\mathcal{X}}(K_1, K_2, K_3)||^2 + \frac{1}{I_3}||\mathbf{y}(f^{mix}, \lambda_1, \lambda_2) - \hat{\mathbf{y}}(K_1, K_2, K_3)||^2 \tag{3.13}$$

where tensor $\mathcal{X}_1$ is a gross estimate of $\mathcal{X}$, and $\hat{\mathcal{X}}(K_1, K_2, K_3)$ is the estimate provided by Multiway Wiener Filtering applied to $\mathcal{R}$ with rank values K_1, K_2, K_3. Vector $\mathbf{y}(f, \lambda_1, \lambda_2)$ is the spectrum model, where f, λ_1, and λ_2 should be estimated, and vector $\hat{\mathbf{y}}(K_1, K_2, K_3)$ is a spectrum, whose location is known, extracted from the tensor estimate $\hat{\mathcal{X}}(K_1, K_2, K_3)$. Generally, an anomaly detector such as RX [96] locates some spectra of interest before they can be unmixed.

3.3.2 Data description, evaluation criteria and image setup

In this subsection, the experimental conditions are presented. Unless specified, they are common to the whole section 3.3.

The performances of the mixedGWO and its adaptive version amixedGWO are evaluated on multispectral images. These images are extracted from the PaviaU scene. These data were collected by the ROSIS sensor over the urban area of Pavia University [46]. This image size is 610×340 pixels. The number of spectral bands is 103 in the wavelength range from about 420 to 850 nm.
In the experiments, the values of the expected image $\mathcal{X}$ and spectral mixture $\mathbf{y}$ are scaled between 0 and 1. The spectral mixture is generated with the reflectance of vegetation for **s1** and the reflectance of soil for **s2**. The resulting spectrum is placed at a specific location in the image, this location being known to extract the spectrum from the denoised image. The expected mixing parameters are set to : $f^{mix} = f_0^{mix} = 0$, $\lambda_1 = 0.15$, $\lambda_2 = 0.41$.
The denoising results obtained are evaluated in terms of SNR.
As a reminder, $SNR = 10\ log_{10}(\frac{||\mathcal{X}||^2}{||\mathcal{X} - \hat{\mathcal{X}}||^2})$.
The images are artificially impaired with white, identically distributed random noise. The input SNR is denoted by SNR_{in} and the value for each experiment is presented in dB. The

results are evaluated in terms of output SNR denoted by SNR_{out}. The unmixing results are evaluated in terms of reconstruction error RE between $\mathbf{y}$ and $\hat{\mathbf{y}}$: $RE = \sqrt{\frac{1}{I_3}||\mathbf{y} - \hat{\mathbf{y}}||^2}$.
Statistical results are obtained with images of size $32 \times 32 \times 4$, from $M = 10$ runs, each with a different random noise realization. For the RGB display of the multispectral images, only 3 representative bands have been selected. They are in the red ($690nm$), green ($550nm$), and blue ($450nm$) wavelength domains respectively. In section 3.3, programs were written in $Matlab^{®}$, and executed on a PC running Windows, with a 3GHz double core and 3GB RAM.

3.3.3 Parameter setting for the proposed and comparative optimization methods

We solve a problem of multispectral image denoising and unmixing, involving four discrete parameters and two continuous parameters.

To evaluate the performance of the proposed adaptive mixed GWO (amixedGWO), we compare it with famous meta-heuristic stochastic optimization methods : particle swarm optimization (PSO) [56], grey wolf optimization (GWO) [95], artificial bee colony (ABC) [54], tree seed algorithm (TSA) [61], genetic algorithm (GA) [45], and simulated annealing (SA) [77]. PSO, GWO, ABC, TSA, GA, and SA search continuous spaces. So, when integer values are expected, we round the result they provide.

The tested algorithms require a few parameters which are set once for all processed images : all methods are run with $\mathrm{T}_{\max} = 20$ iterations. All methods including TSA are run in such a way that the computational load they require when the image size is $32 \times 32 \times 4$ is the same, that is, 1.5 sec. for all methods except ABC, which requires 2.4 sec, and SA which requires 2.0 sec. For this, the number of agents is $Q = 12$ for amixedGWO, PSO, GWO, and GA ; and $Q = 6$ for TSA, SA and ABC.
We choose this relatively low number of agents and iterations, compared to the experiments in previous sections, because for this application the computational load required to compute the criterion value is much higher, in particular when the image contains more than 100 rows, columns or bands.

We remind that the first three expected parameters are the ranks K_1, K_2, K_3, the fourth is the type of mixing model f^{mix}, the fifth and sixth are the mixing coefficients λ_1 and λ_2. Table 3.4 presents the parameters used to define the search spaces for each unknown.

The number H_i of values in the search spaces for the ranks K_i, $i = 1, \ldots, 3$ is either 8, or the size of the image I_i, $i = 1, \ldots, 3$ if $I_i \leq 16$. Note that there are only two possible values for the mixing model f^{mix}, and that the acceptable values for the mixing coefficients λ_1 and

Expected parameter index i \ Search space	H_i	$\mathbf{d}_i^{ind}$	$\mathbf{d}_i^{val}$
1,2,3	$\min(I_i, 8)$	$[1, 2, \ldots, I_i]^T$	$\left[1, \frac{I_i}{H_i}, 2\frac{I_i}{H_i}, \ldots, I_i\right]^T$
4	2	$[0, 1]^T$	$\left[f_0^{mix}, f_1^{mix}\right]^T$
5,6	•	•	$[0; 1]^T$

Table 3.4 — Search spaces for the optimization methods. Symbol • means irrelevant.

λ_2 are between 0 and 1.
For PSO, GWO, ABC, TSA, GA and SA, the search spaces are continuous, with bounds which, $\forall\ i = 1, \ldots, 6$, are the first and last values of $\mathbf{d}_i^{val}$, unless for $i = 4$, for which the lower bound is 0, and the upper bound is 1. For the first four parameters, that is, the rank values and the type of mixing model, all test values are rounded when the criterion is evaluated. In PSO, the acceleration constants γ_1 and γ_2 [56] are set to 2 and 3 respectively. Each run is performed with a random initialization of the search agents in the optimization algorithms. The ABC method is run with 4 onlooker bees. The TSA algorithm is run with $ST = 0.9$: a value close to 1 is recommended for lower dimensional optimization problem [61] and TSA should provide a fast convergence.
For GA the crossover probability is 0.9, and the mutation probability is 0.1. For SA we set a number of control points equal to 12, that is, the same value as the one used for Q in the other methods.

3.3.4 Experimental results

In this subsection we provide results obtained from a multispectral image extracted from PaviaU scene : statistical results are computed on a small image, and visual results are computed from a larger image.

Figure 3.11 shows the noise-free multispectral images which are used for the tests ; and Fig. 3.12 shows the two endmembers, and the expected spectrum, obtained with the parameters $f^{mix} = f_1^{mix} = 0$, $\lambda_1 = 0.15$, $\lambda_2 = 0.41$.

In subsubsection 3.3.4.1, we propose a school-case study in the ideal situation where $SNR_{in} = \infty$, in two cases, with two different versions of the reference $\mathcal{X}_1$. In this noise-free case we expect the optimization methods to yield a result tensor which is exactly the reference $\mathcal{X}_1$.

In subsubsection 3.3.4.2, we consider a realistic case : the image $\mathcal{R}$ is impaired with some finite input SNR value, and the reference image is obtained with a Wiener processing in

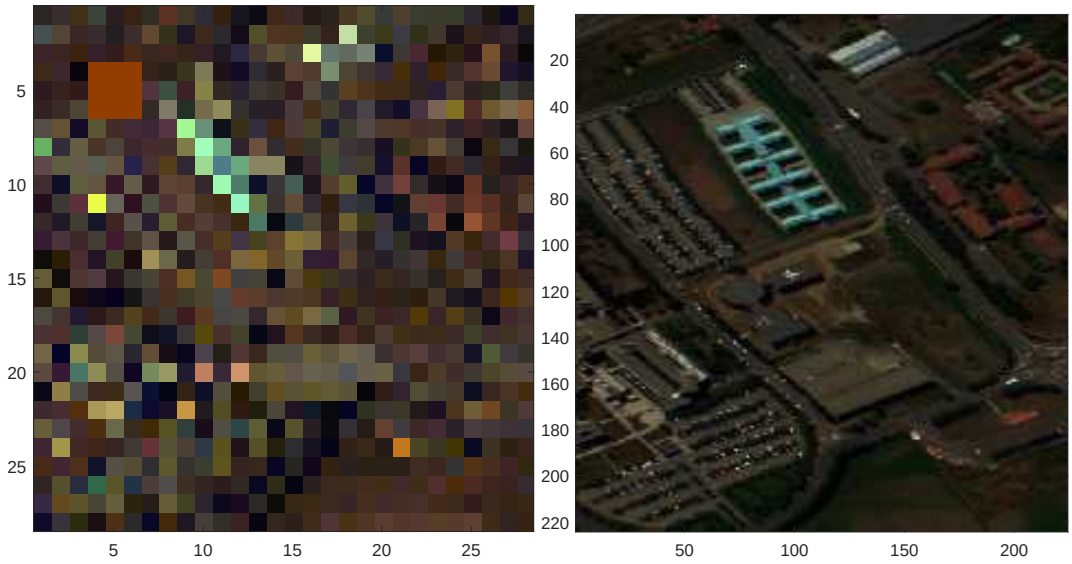


Figure 3.11 — PaviaU $32 \times 32 \times 4$ and PaviaU $256 \times 256 \times 103$: Noise-free images

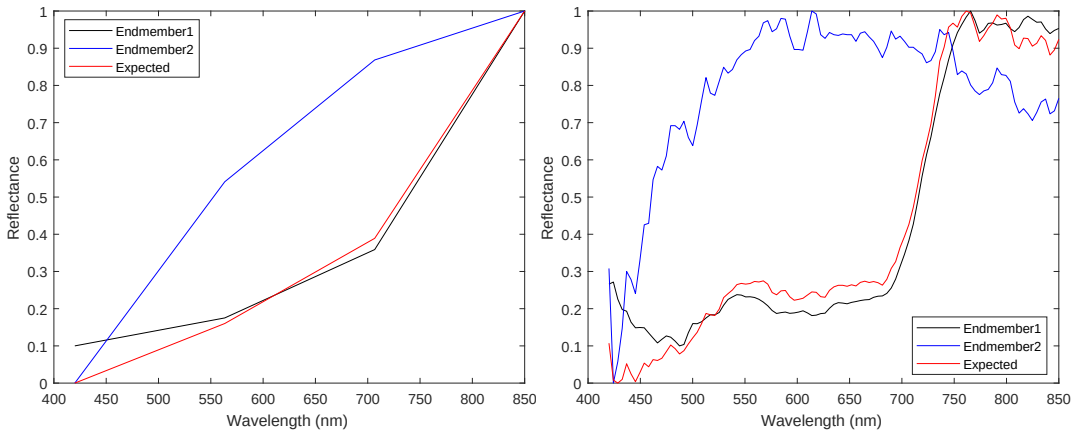


Figure 3.12 — PaviaU $32 \times 32 \times 4$ and PaviaU $256 \times 256 \times 103$: endmembers **s1** (black) and **s2** (blue), and expected spectrum **y** (red)

Fourier domain, which is a basic parameter-free method. This denoising method is applied band-by-band, for each spectral band of the processed image.

3.3.4.1 Convergence study on a school case

In this subsubsection, the input SNR is $SNR_{in} = \infty$, and the reference tensor $\mathcal{X}_1$ is either the noise-free image or the output of MWF when applied to $\mathcal{X}$ with a priori known rank values.

First case : the reference is the noise-free tensor

The reference tensor $\mathcal{X}_1$ is here the noise-free tensor, and the expected rank values are then $K_1 = I_1 = 32$, $K_2 = I_2 = 32$, and $K_3 = I_3 = 32$. Fig. 3.13 presents the mean convergence plot for this experiment, obtained with $M = 10$ runs. Table 3.5 presents statistical results obtained on the expected parameters, and Table 3.6 presents the mean reconstruction error RE over the M runs.

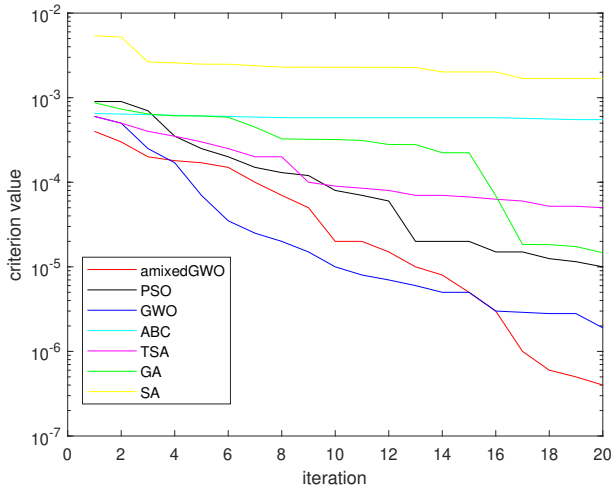


Figure 3.13 — Mean Convergence plot with $SNR_{in} = \infty$ and noise-free image as reference $\mathcal{X}_1$

Parameters		Expected Values	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
K_1	*Avg.*	32	32.0000	31.5858	31.9088	27.2380	32.0000	24.7749	15.0052
K_2	*Avg.*	32	31.1999	32.0000	31.9635	27.2451	30.1649	30.1359	17.4724
K_3	*Avg.*	4	3.900	3.9631	3.8694	3.1280	3.6024	2.6075	2.7257
f^{mix}	*Avg.*	0	0.2000	0.4023	0.2884	0.4048	0.4173	0.2926	0.5756
λ_1	*Avg.*	0.15	0.1587	0.1387	0.1390	0.1037	0.0539	0.2137	0.1575
λ_2	*Avg.*	0.41	0.5008	0.5352	0.4172	0.4683	0.3790	0.3836	0.3174

Table 3.5 — Estimated Parameters with $SNR_{in} = \infty$ and noise-free image as reference $\mathcal{X}_1$

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
∞		$1.85e-03$	$4.14e-03$	$1.77e-03$	$1.41e-02$	$7.68e-03$	$4.14e-02$	$5.90e-02$
	Rank	2	3	1	5	4	6	7

Table 3.6 — Spectrum reconstruction error RE with $SNR_{in} = \infty$ and noise-free image as reference $\mathcal{X}_1$

The convergence plot in Fig. 3.13, and the numerical results in Table 3.5 show that the proposed amixedGWO, PSO, and GWO behave best in this experiment. For instance, no mean estimated rank value differs from the expected one by more than 1. TSA, ABC, GA and SA underestimate rank values, which may have an influence on the estimation of the mixing parameters. It also can be noticed that the mean estimated values of mixing model f are elevated : we may face a multimodal optimization problem with two relative minima which are very close to each other. Still, our method surpasses the comparative methods in terms of convergence as shown in Fig. 3.13. Also, we can see from Table 3.6 that the reconstruction error are the smallest when the proposed amixedGWO or the comparative GWO methods are used.

Second case : the reference is the output of MWF

In this paragraph, the input SNR is still infinite, but the expected rank values are less than the image size. Indeed, the reference is the output of MWF applied to the noise-free tensor $\mathcal{X}$ with rank values which are as follows : $K_1 = 0.5 \cdot I_1 = 16$, $K_2 = 0.5 \cdot I_2 = 16$, and $K_3 = I_3 = 4$. We expect from the optimization methods that they retrieve these rank values.

Fig. 3.14 presents the mean convergence plot for this experiment, obtained with $M = 10$ runs. Table 3.7 presents statistical results obtained on the expected parameters, and Table 3.8 presents the mean reconstruction error RE over the M runs. We notice there exists a bias on the mixing model for all optimization methods, possibly due to two local minima with a very similar score. Indeed the RE values in Table 3.8 are similar for amixedGWO, GWO, TSA, and PSO, although, as we can see in Table 3.7, PSO yields an estimated value for the mixing model which is significantly different.

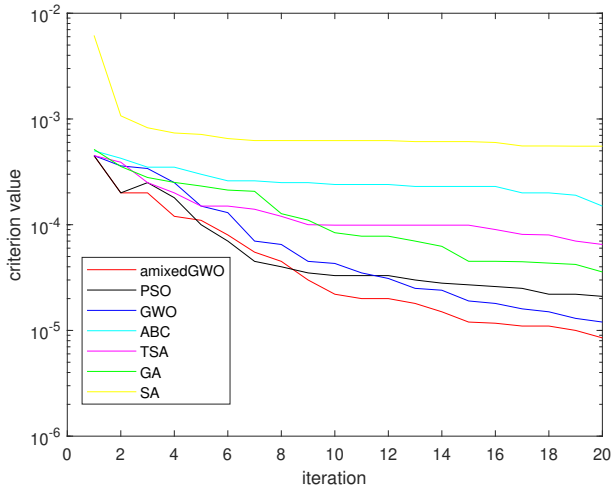


Figure 3.14 — Mean convergence plot with $SNR_{in} = \infty$ and $\hat{\mathcal{X}}(16, 16, 4)$ as reference $\mathcal{X}_1$

Parameters		Expected Values	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
K_1	*Avg.*	16	16.10	16.84	16.11	22.67	20.39	19.34	13.57
K_2	*Avg.*	16	15.90	15.77	16.05	19.85	19.57	20.05	22.02
K_3	*Avg.*	4	4	3.94	3.77	3.54	3.72	3.75	2.79
f^{mix}	*Avg.*	0	0.2	0.45	0.22	0.40	0.28	0.46	0.57
λ_1	*Avg.*	0.15	0.114	0.097	0.088	0.162	0.124	0.053	0.404
λ_2	*Avg.*	0.41	0.534	0.610	0.456	0.649	0.537	0.495	0.9999

Table 3.7 — Estimated Parameters with $SNR_{in} = \infty$ and $\hat{\mathcal{X}}(16, 16, 4)$ as reference $\mathcal{X}_1$

SNR_{in}	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
∞	$7.20e-03$	$8.90e-03$	$7.748e-03$	$1.017e-02$	$8.022e-03$	$1.35e-02$	$1.20e-01$
Rank	1	4	2	5	3	6	7

Table 3.8 — Spectrum reconstruction error RE with $SNR_{in} = \infty$ and $\hat{\mathcal{X}}(16, 16, 4)$ as reference $\mathcal{X}_1$

In Table 3.8, we can see that the reconstruction error is more elevated than in the previous case : through filtering with low rank values, the denoised spectrum is significantly different from the expected spectrum. As the optimization methods aim for the denoised spectrum, the RE value increases. The convergence plot in Fig. 3.14, and show that our method surpasses the comparative methods in terms of convergence.

3.3.4.2 Realistic case

In this subsection we consider a realistic case where the reference tensor is obtained with a parameter-free, simple method. This method is a Wiener filtering process applied band-by-band in Fourier domain. In this realistic case, the input SNR is less than ∞. We exemplify our method on a large image, with five different values of input SNR : $SNR_{in} = 0, 5, 10, 15,$ and 20 dB. For $SNR_{in} = 10$ dB we provide visual results for one run (see Figs. 3.15, 3.16, and 3.17), as well as the mean convergence curves obtained on $M = 3$ runs (in Fig. 3.20).

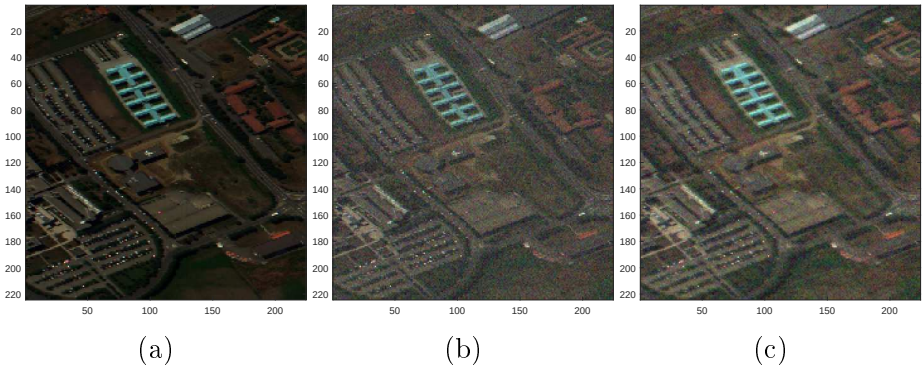


Figure 3.15 — PaviaU $256\times256\times103$: (a) Noise-free, (b) impaired 10 dB, and (c) reference images

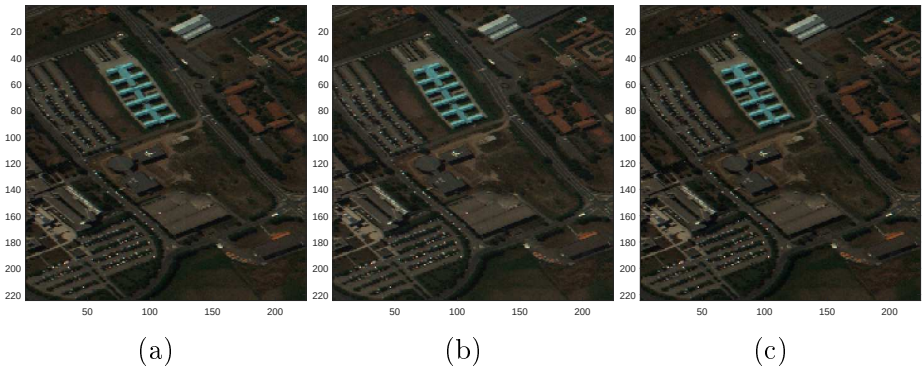


Figure 3.16 — PaviaU $256 \times 256 \times 103$. Denoised images obtained with (a) amixedGWO, (b) PSO, (c) GWO

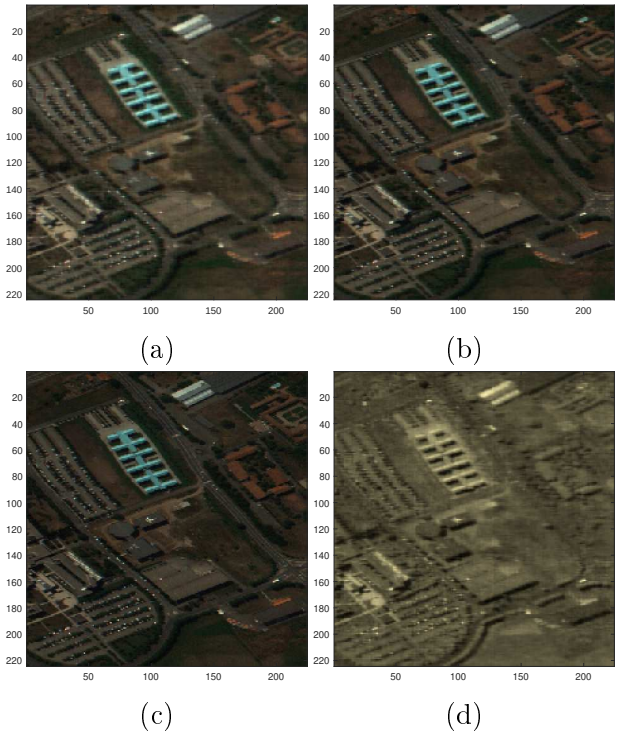


Figure 3.17 — PaviaU $256 \times 256 \times 103$. Denoised images obtained with (a) ABC, (b) TSA, (c) GA, (d) SA

Tables 3.9, 3.10 and 3.11 present respectively the output SNR, the reconstruction error values and the global best for all input SNR values. The overall rank for amixedGWO is 1 in all of these aspects, though the output SNR is smaller than for at least one comparative method, on input SNR values 5 and 15 dB. This means that, overall, amixedGWO may not yield the best rank values in terms of the sole denoising. However, a good behavior in terms of exploitation can explain the values of reconstruction error, which are the smallest for all input SNR value except 0 dB (see Table 3.10). Table 3.11 shows that the global best values are the smallest for amixedGWO except on input SNR values 5 and 15 dB. In these cases amixedGWO may yield a denoised spectrum which is the closest to the model spectrum, but at the expense of the first term of the criterion in Eq. (3.13).

From these results we can infer the following comments : thanks to the improved discrete optimization process used to estimate the rank values, the proposed amixedGWO seems to converge quickly towards rank values which minimize both terms of the criterion in Eq. (3.13), and has time to refine the estimation of the continous parameters (the mixing coefficients). A compromise must be found in some cases : the spectrum of interest in the denoised image may be the closest possible to the model spectrum, and not to the corresponding spectra in the reference image.

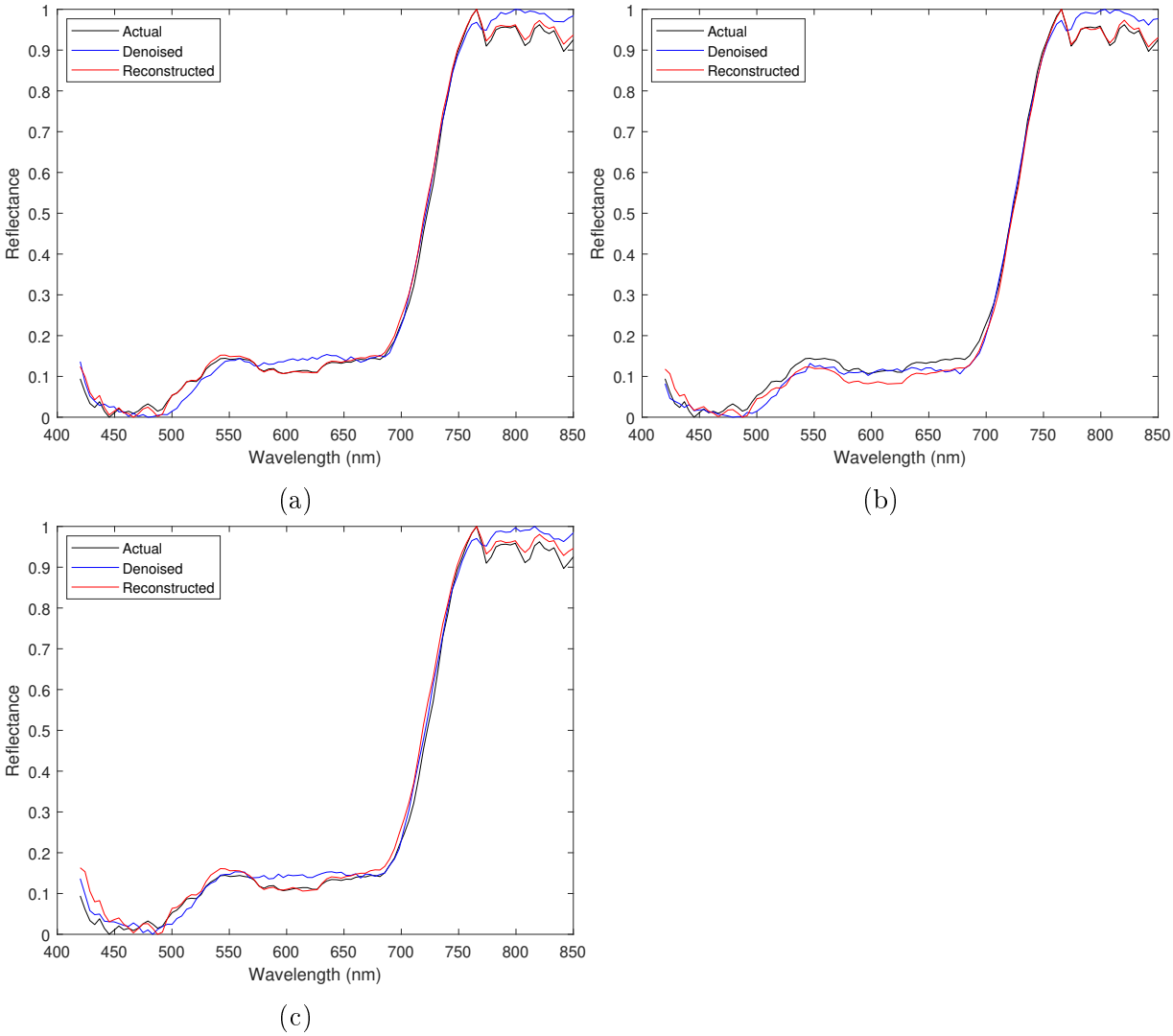


Figure 3.18 — PaviaU $256 \times 256 \times 103$. Actual, denoised, and reconstructed spectra obtained with : (a) amixedGWO, (b) PSO, (c) GWO

Here are details about one run, which yielded the images in Figs. 3.16 and 3.17, and the spectra in Figs. 3.18 and 3.19 :

In this case all methods except ABC and SA, yield approximately the same global best after convergence : $6.75\ 10^{-4}$ for amixedGWO, $8.63\ 10^{-4}$ for PSO, $6.87\ 10^{-4}$ for GWO, $1.97\ 10^{-3}$ for ABC, $8.61\ 10^{-4}$ for TSA, $7.60\ 10^{-4}$ for GA, $2.16\ 10^{-3}$ for SA.
The output SNR for the reference image is 12.79 dB. The output SNR values (in dB) are respectively 18.35 for amixedGWO, 14.88 for PSO, 17.83 for GWO, 8.64 for ABC, 16.26 for TSA, 18.91 for GA, and 8.54 for SA. So the denoised image with the best output SNR value is provided by GA, but as can be seen in Figs. 3.16 and 3.17, a significant difference only exists between the result obtained by ABC and SA and the other methods, but the other images are

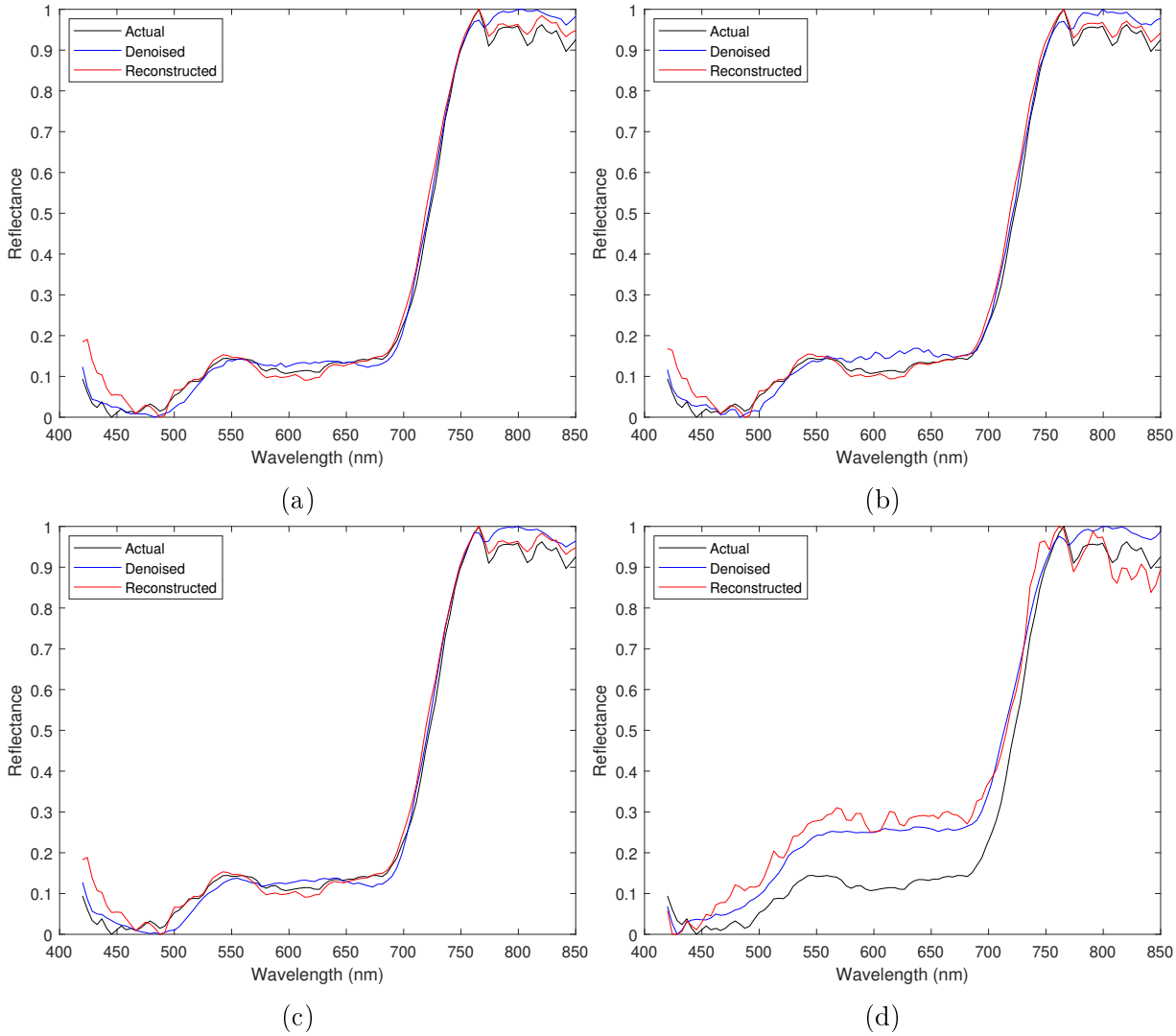


Figure 3.19 — PaviaU $256 \times 256 \times 103$. Actual, denoised, and reconstructed spectra obtained with :(a) ABC, (b) TSA, (c) GA, (d) SA

very similar.

The reconstruction error values are respectively $1.20\ 10^{-3}$ for amixedGWO, $1.94\ 10^{-3}$ for PSO, $2.47\ 10^{-3}$ for GWO, $2.92\ 10^{-3}$ for ABC, and $2.69\ 10^{-3}$ for TSA, $2.90\ 10^{-3}$ for GA, and $1.08\ 10^{-2}$ for SA.

The RE value obtained with GWO or TSA is higher than for amixedGWO, because one the one hand the RE is computed between the actual spectrum and the reconstructed one ; on the other hand the criterion which is minimized involves the difference between the denoised spectrum and the reconstructed spectrum.

This is confirmed by the spectra displayed in Fig. 3.18 : when amixedGWO is used, the reconstructed spectrum (in red) is the closest to the actual spectrum (in black), because the denoised spectrum (in blue) is the closest to the actual spectrum. Somehow, in this case, the

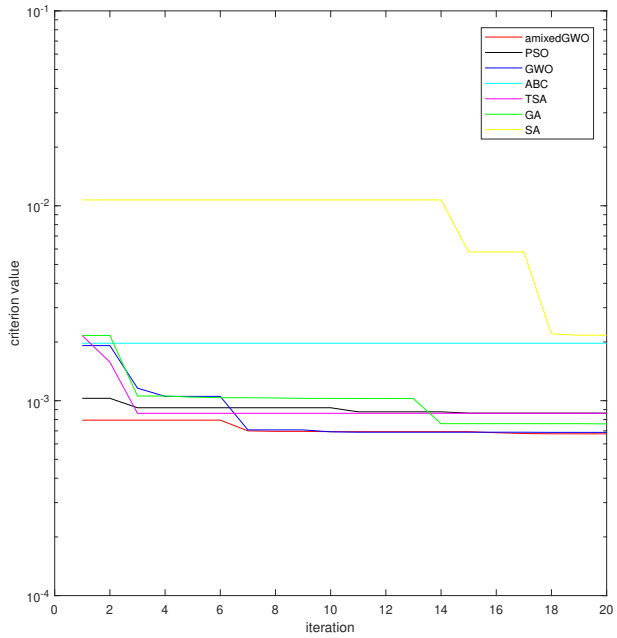


Figure 3.20 — Mean Convergence plot with SNR_{in} = 10 dB and Fourier Wiener as reference $\mathcal{X}_1$

SNR_{in}		Ref.	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		3.99	**9.333**	7.659	7.745	5.560	8.429	8.45	6.42
	Rank	8	1	4	5	6	3	2	7
5 dB		8.79	12.180	11.477	**13.258**	7.878	11.893	7.97	8.34
	Rank	5	2	4	1	7	3	8	6
10 dB		13.24	**17.777**	14.159	17.108	7.719	15.310	8.88	2.49
	Rank	5	1	4	2	7	3	6	8
15 dB		16.61	17.804	18.587	**19.733**	13.388	15.815	15.22	14.43
	Rank	4	3	2	1	8	5	6	7
20 dB		18.26	**22.664**	20.869	22.653	17.417	18.504	15.37	13.21
	Rank	5	1	3	2	6	4	7	8
	Avg. Rank	5.4	1.6	3.4	2.2	6.8	3.6	5.8	7.2
	Overall Rank	5	1	3	2	7	4	6	8

Table 3.9 — Results denoising SNR_{out} with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$

balance between denoising and unmixing is better when amixedGWO is used. Consequently, the denoised spectrum is the closest possible to the model. As concerns the estimated parameters, we could notice that the spatial ranks are elevated (between 255 or 256 for all methods except ABC and SA), and that the spectral rank is small (between 60 and 97 for all methods except ABC and SA) ; it can also be noticed that TSA and GA provided f_1 as spectrum model whereas f_0 was expected, yielding though small RE values. From this we infer that there may exist at least two close relative minima in the minimized criterion. That confirms we are

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		$2.81e-03$	$4.44e-03$	$3.51e-03$	$5.30e-03$	**$2.80e-03$**	$4.00e-03$	$2.53e-02$
	Rank	2	5	3	6	1	4	7
5 dB		**$2.34e-03$**	$4.41e-03$	$3.41e-03$	$5.45e-03$	$3.12e-03$	$5.63e-03$	$5.59e-03$
	Rank	1	4	3	5	2	7	6
10 dB		**$2.47e-03$**	$3.12e-03$	$3.02e-03$	$3.68e-03$	$2.69e-03$	$2.79e-03$	$5.17e-03$
	Rank	1	5	4	6	2	3	7
15 dB		**$1.89e-03$**	$2.82e-03$	$3.13e-03$	$4.40e-03$	$2.66e-03$	$2.83e-03$	$5.63e-03$
	Rank	1	3	5	6	2	3	7
20 dB		**$1.92e-03$**	$2.82e-03$	$2.09e-03$	$2.75e-03$	$2.59e-03$	$4.26e-03$	$4.28e-03$
	Rank	1	5	2	4	3	6	7
	Avg. Rank	1.2	4.4	3.4	5.4	2	4.6	6.8
	Overall Rank	1	4	3	6	2	5	7

Table 3.10 — Results spectrum reconstruction error RE with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		**$2.094e-03$**	$2.894e-03$	$2.699e-03$	$2.264e-03$	$2.433e-03$	$6.14e-03$	$1.74e-02$
	Rank	1	5	4	2	3	6	7
5 dB		$1.372e-03$	$1.556e-03$	**$1.166e-03$**	$3.540e-03$	$1.653e-03$	$3.81e-03$	$4.54e-03$
	Rank	2	3	1	5	4	6	7
10 dB		**$6.642e-04$**	$1.044e-03$	$7.137e-04$	$2.046e-03$	$7.620e-04$	$2.26e-03$	$7.88e-03$
	Rank	1	4	2	5	3	6	7
15 dB		$5.032e-04$	**$4.641e-04$**	$5.538e-04$	$9.461e-04$	$5.480e-04$	$1.08e-03$	$1.48e-03$
	Rank	2	1	4	5	3	6	7
20 dB		**$2.929e-04$**	$3.418e-04$	$2.931e-04$	$4.792e-04$	$3.919e-04$	$7.71e-04$	$1.90e-03$
	Rank	1	3	2	5	4	6	7
	Avg. Rank	1.4	3.2	2.6	4.4	3.4	6	7
	Overall Rank	1	3	2	5	4	6	7

Table 3.11 — Results Global best with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$

facing a fixed-dimension multimodal optimization problem.

3.4 Conclusion

In section 3.1, the principle Audience Measurement tool ISAM, designed for working on a vending machine using an embedded camera, has been detailed. As of now, this algorithm is able to detect the people appearing in the camera's field of view, to track them while they stay in this field of view, and to determine which of the detection is the most probable to be the current machine's user. Therefore, the ISAM algorithm is able to collect 4 different data elements : the number of people detected during a given period of time, the amount of time spent in the field of view by each person detected, whether or not a detected person

has been a potential user of the vending machine and, if it is the case, the amount of time spent in the field of view while being the potential user. This algorithm works at a real-time pace with a camera frequency of 15 FPS and with an accuracy of 85%. However, although the algorithm showed good results when it was tested in real-like situation, it would be necessary to test it in real situation. Indeed, in a real situation, all the software components of the vending machine are activated and the condition of its hardware are not necessarily in good shape. Moreover, as the Viola-Jones method is used to detect the faces, the ISAM algorithm is limited to front-face detection, *i.e.* the people who are looking in the direction of the vending machine, and cannot detect the people who are just passing through the field of view without giving a look to the machine. In the current context, this limit is welcomed because it would limit the algorithm to count only the people who could be interested in the machine's product, and thus counting the potential customers rather than counting all the people. Finally, the characterization part, such as gender or age recognition, is still missing from the ISAM tool.

In section 3.2, the impact of the bio-inspired optimization method Grey Wolf Optimizer (GWO) on gender classification by Support Vector Machine (SVM) has been studied. An adaptive version of the GWO algorithm was also applied, working in 3 steps. A global version of GWO, with $\eta > 1$ in order to emphasize on the exploration phase, was first used on a reduced image database to roughly determine the training parameters. This estimation was then refined by reducing the η parameter and increasing the image database size. Tested on the FERET database, the proposed aGWO achieved an accuracy of 91.9% while reducing the computing time by 2 compared to a classic method, such as GWO or PSO. The results obtained showed that the use of the GWO and a work around the η parameter is a good basis for the creation of a bio-inspired optimization method for mixed problems. Moreover, with a mixed optimization method, the work of image classification could be extended to the polynomial kernel. An algorithm able to select by itself the most suited kernel could also be conceivable. However, although the mixedGWO has been developed during this thesis, the gender recognition part has not been added to the ISAM tool yet for 2 reasons :

• the SVM training function is computationally heavy and slow to compute, even if its speed has been significantly increased with the adaptive GWO ;

• there is no image database compatible with the problematic, *i.e.* a database composed of images of people's faces taken slightly from above, who do not look directly at the camera, and images with non uniform lightning conditions.

In section 3.3, the robustness of the proposed amixedGWO, presented in Chapter 2, has been tested on a real-world application : simultaneous denoising and unmixing of multispectral images. The goal of this application was to find the best balance between the quality of the denoising and the quality of the spectrum unmixing. This application is a real case where some of the parameters, here the rank values and the type of mixing model, take their values in discrete search spaces, and some other parameters, here the mixing coefficients,

take their values in continuous search spaces. The proposed method has been compared to the state-of-the-art methods PSO, GWO, TSA, ABC, GA and SA on 3 specific cases : with a noise-free reference, with a reference computed by MWF and in a realistic case with a reference obtained with a basic parameter free method. In the 3 studied cases, the amixedGWO always found the correct expected rank values Besides, the amixedGWO was the one with the best reconstruction error, except in the noise-free case. It is inferred from the results obtained, that the criterion chosen was a multi-modal function of the parameters because a bias exists on the estimated mixing model for some of the tested algorithms and not for the others, with a small difference on the global best value which was reached. The limitation of the proposed strategy relies on the choice of the reference tensor : the convergence performance of amixedGWO was good, but the criterion was minimized with respect to a reference which may not be reliable. Indeed, it is impossible to compute a function to minimize without having a reference image.

Conclusion and evolution prospects

This thesis had two purposes. Firstly, we developed an audience measurement tool, using computer vision and image processing. Secondly, we proposed an optimization method designed for mixed problems, in order to tackle problems such as image classification with a SVM or the joint denoising and unmixing of multi-spectral and hyper-spectral images.

In Chapter 1, an overview of the state-of-the-art describes the three domains required for the development of an Audience Measurement tool using image processing and computer vision : the techniques for face detection, the techniques for object tracking and finally, object recognition, in particular the principles of classic image classification methods, such as the Conventional Neural Network (CNN) and the Support Vector Machine (SVM). From this review of the literature, it appears that the Viola-Jones detector, the Lucas-Kanade sparse Optical Flow and SVM seem like the ones which fits the most to the Audience Measurement problem in a real-time and embedded environment.
Also in Chapter 1, a review of bio-inspired optimization methods has been done, with a clear distinction between the methods designed for fully continuous problems and those designed for fully discrete problems. State-of-the-art algorithms, such as the Particle Swarm Optimization (PSO), the Grey Wolf Optimizer (GWO), the Tree-Seed Algorithm (TSA), the Ant Colony Optimization (ACO) and the multi-objective discrete GWO (MODGWO), were also detailed. From this review, we noticed that there is no bio-inspired optimization methods designed for mixed problems, although it could be useful for a SVM optimization, or for another image processing problematic which is the joint denoising and unmixing of multi-spectral and hyper-spectral images.

In Chapter 2, a variant of the GWO algorithm is presented in details. This method has been developed during this thesis to be able to tackle fully continuous problems, fully discrete problems and mixed problems. The proposed new GWO method, named mixedGWO,

was compared statistically to other versions of the GWO algorithm (GWO, mGWO and MODGWO) on fully continuous and fully discrete benchmark functions, as well as to PSO, TSA and GA on the CEC2014 functions. However, because there is not such methods in the state-of-the-art, it has not been compared to any other method on mixed benchmark functions, but the results obtained are self-satisfactory. From the results obtained on these benchmark functions and in comparison to the state-of-the-art, the proposed mixedGWO seems well adapted for the 2 target applications of this thesis.

The Chapter 3 has been divided in two distinguishable parts :

• In section 3.1, the IntuiSense Audience Measurement (ISAM) tool has been presented and detailed. This algorithm, which works thanks to a camera embedded in a vending machine, is able to detect and track people appearing in the camera field of view with an acquisition speed of 15 FPS and an accuracy of 85%. Moreover, the ISAM tool is also able to determine who among the detected people has the highest probability of being the current user of the vending machine. Four types of data are collected with the ISAM tool : the number of people detected during a given period of time, the amount of time spent in the field of view by each person detected, whether or not a detected person has been a potential user of the vending machine, and if it is the case, the amount of time spent in the field of view while being the potential user. In section 3.2, the impact of the bio-inspired optimization method Grey Wolf Optimizer (GWO) on gender classification by Support Vector Machine (SVM) has been studied. In this study, the adaptive version of the GWO algorithm (aGWO) was also tested and allowed to train a SVM with the same good recognition rate as the one trained with GWO, while reducing its computing time by a factor of 2. This study has permitted to validate the choice of GWO as a basis for the optimization method designed for mixed problems and its compatibility with the problem of image classification by SVM.
In the future, the people characterization part could be added to the ISAM tool by applying what has been done in section 3.2. However, it is not possible yet because of two important things are missing : time and data. The SVM training computation time is high (several days) and it must be done $Q \times T_{max}$ times, with Q the number of wolves and T_{max} the maximum number of iterations. Moreover, there is no training database adapted to the target problematic. Indeed, most of the existing databases, such as the FERET database used in section 3.2, are composed of human faces taken from the front in ideal conditions, $i.e.$ with an uniform lightning and with the people looking directly at the camera. Therefore, in the future, it should be a good idea to create a custom database of people faces taken in real-life conditions, $i.e.$ with unknown and uncontrollable lightning conditions, and people's faces taken from slightly above them without having them looking directly at the camera.

• In section 3.3, the performances of the proposed amixedGWO were measured on a real-world application : the joint denoising and unmixing of multi-spectral images. The goal of this aplication was to find the best balance between the quality of the image denoising and the quality of the spectra unmixing. In this section, the proposed amixedGWO was compared to the state-of-the-art methods PSO, GWO, TSA, ABC, GA and SA. The proposed amixedGWO outperformed the compared methods for the same amount of computational time spent per run. However, the proposed strategy relies heavily on the choice of the reference tensor : the convergence performance of amixedGWO was good, but the criterion was minimized with respect to a reference which may not be reliable. Indeed, it impossible to compute a function to minimize without having a reference image. Therefore, it could be interesting to implement an iterative version of this application, taking as an input reference the image obtained from the amixedGWO method at each iteration, in order to compensate the potential problem of a bad reference.

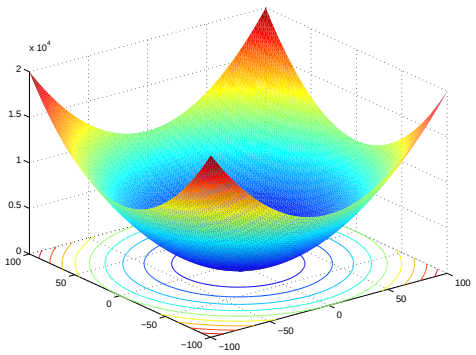
About the future evolution of the mixedGWO and amixedGWO methods, it could be interesting to study the comparative performances of various versions of GWO when the number of parameters to estimate is changing, and, for the discrete version of GWO, when the number of values in the search spaces is changing. An adaptive search space could be a solution. Moreover, it seems interesting to study the performance of the proposed amixedGWO algorithm on other applications, and to create its multi-objective version.
Finally, about bio-inspired optimization for mixed problems in general, it could be interesting to create a discrete version of the comparative methods such as ABC and TSA, which is not restricted to binary, and getting inspired by the formalism that we proposed with our mixed GWO. However, although they are well-performing methods, their high computational time needed could be a handicap.

APPENDIX

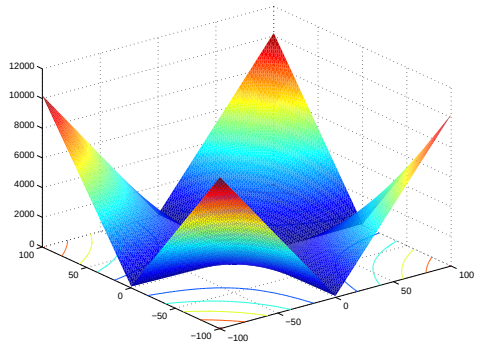
A Tables and Figures

F	step	$T_{max} = 15,000$			$T_{max} = 30,000$		
		Avg.	Std.	Med.	Avg.	Std.	Med.
F_1	1	2.73	1.64	2	1.20	0.98	1
F_2	0.5	0.08	0.19	0	0.10	0.20	0
F_3	1	2.60	1.81	2	1.40	0.97	1
F_4	1	1.13	0.68	1	1.17	0.53	1
F_5	0.5	7.93	6.42	7.5	4.53	3.77	4
F_6	0.5	2.74	2.04	2	1.72	1.34	1.25

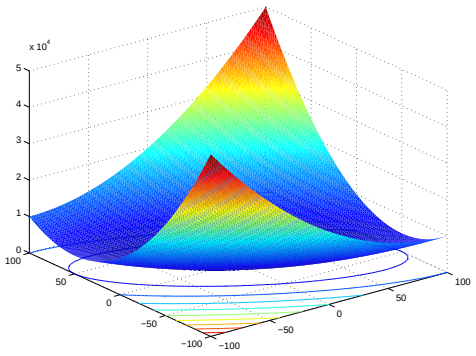
Table A.1 — Results of MODGWO on unimodal benchmark functions in discrete search space with $T_{max} = 15,000$ and $T_{max} = 30,000$



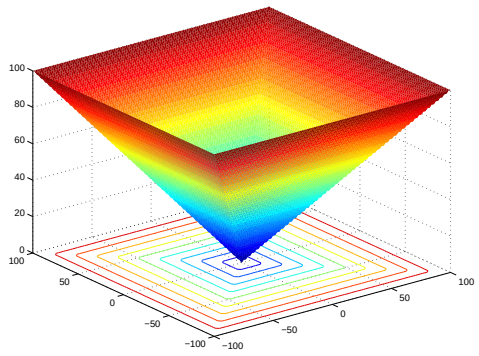
(a) F1



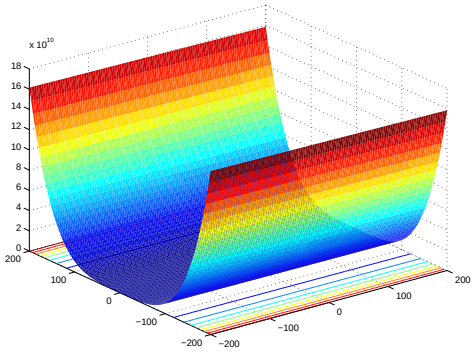
(b) F2



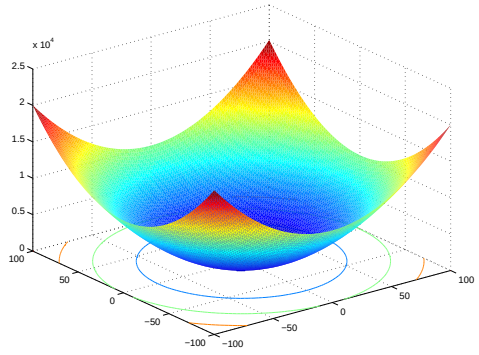
(c) F3



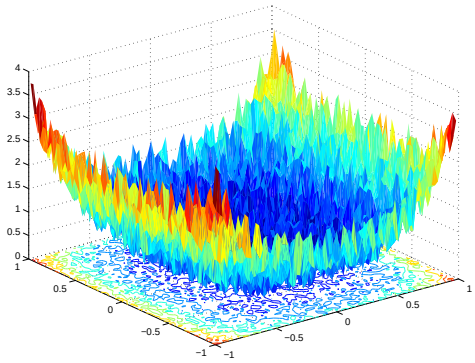
(d) F4



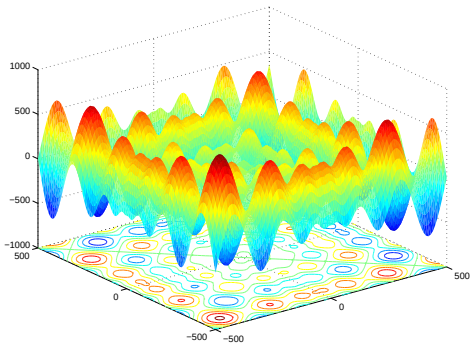
(e) F5



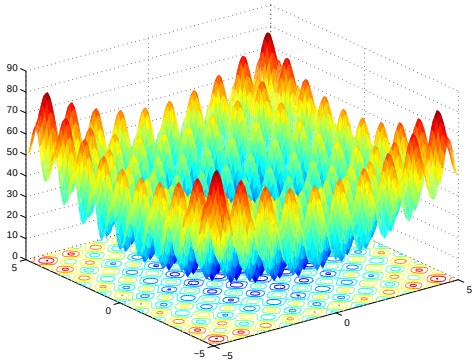
(f) F6



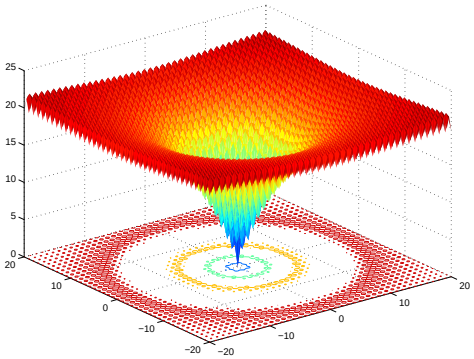
(g) F7



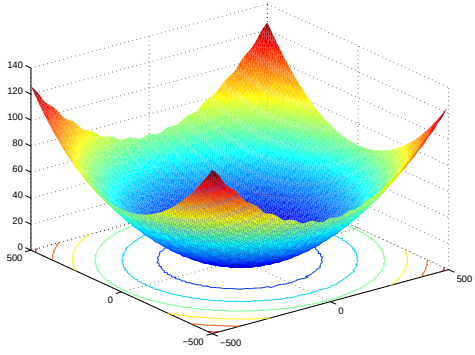
(h) F8



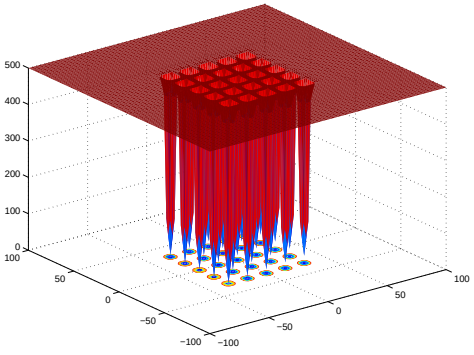
(i) F9



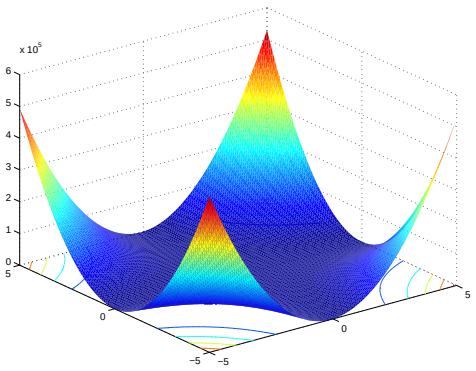
(j) F10



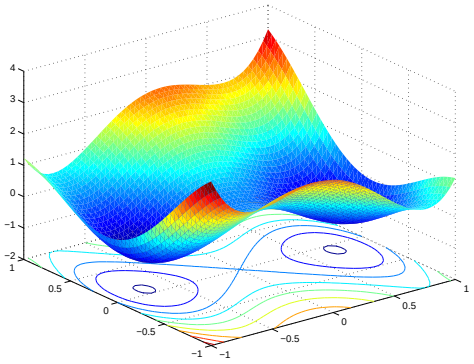
(k) F11



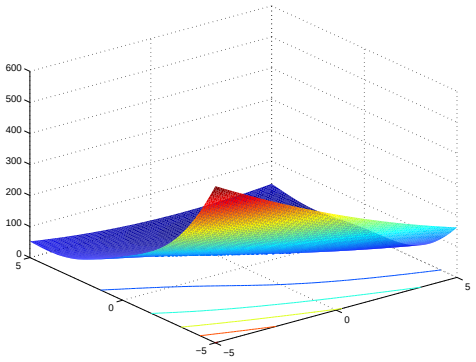
(l) F14



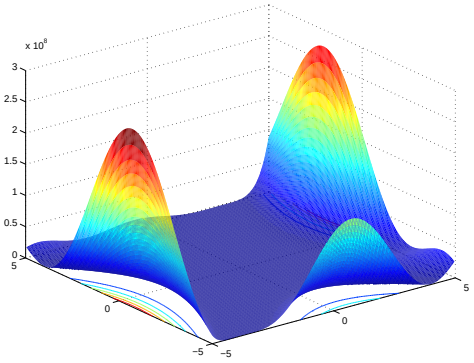
(m) F15



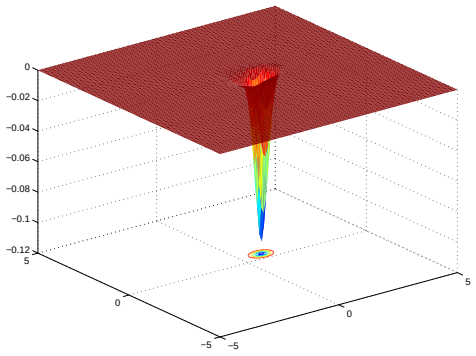
(n) F16



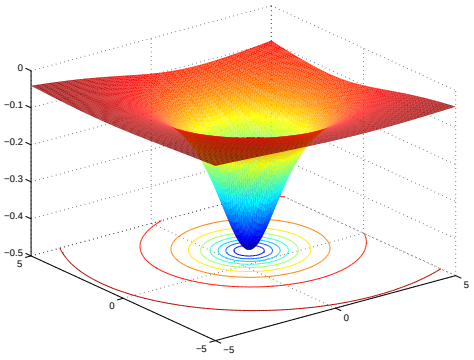
(o) F17



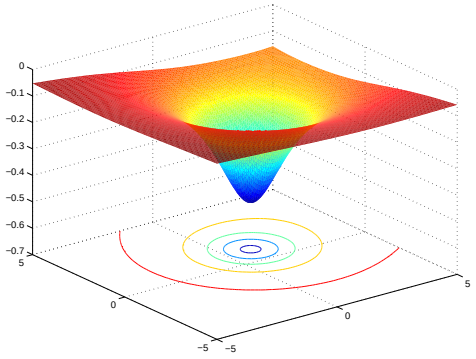
(p) F18



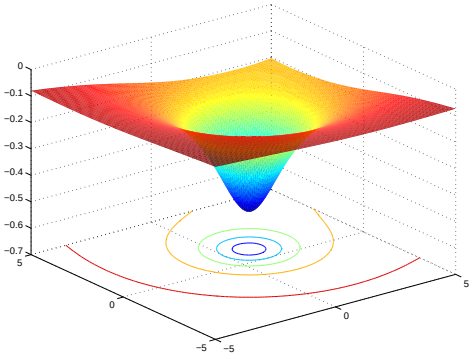
(q) F20



(r) F21



(s) F22



(t) F23

Figure A.-1 — Benchmark functions meshes

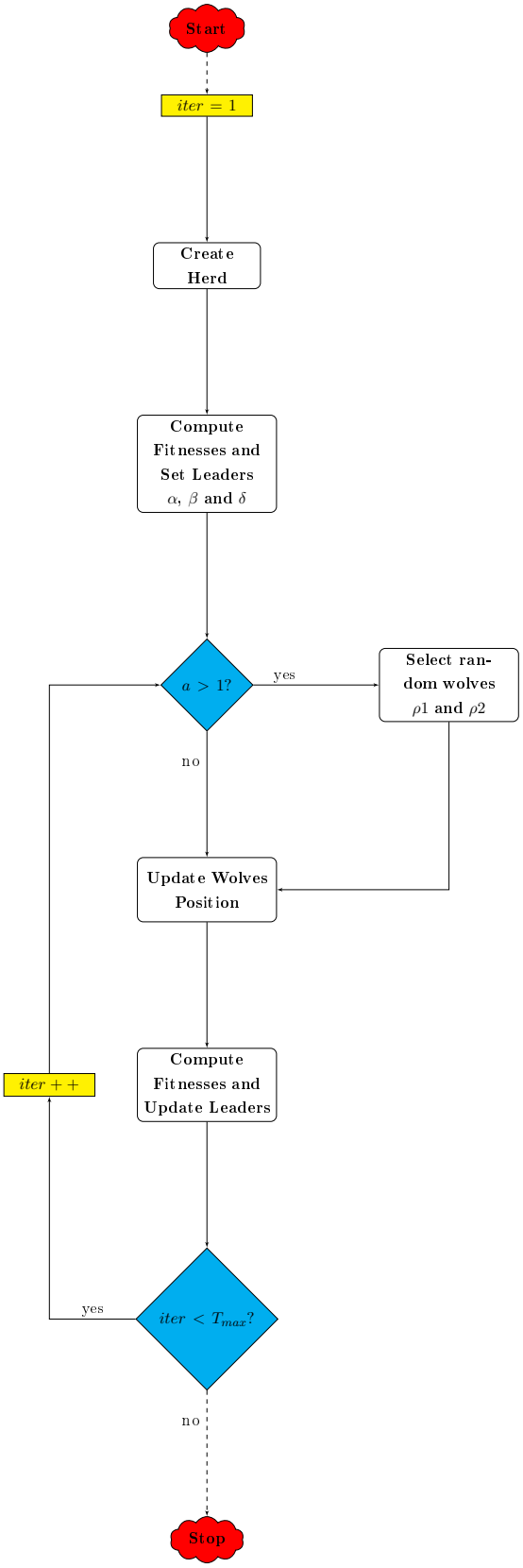


Figure A.0 — The mixedGWO's main flowchart

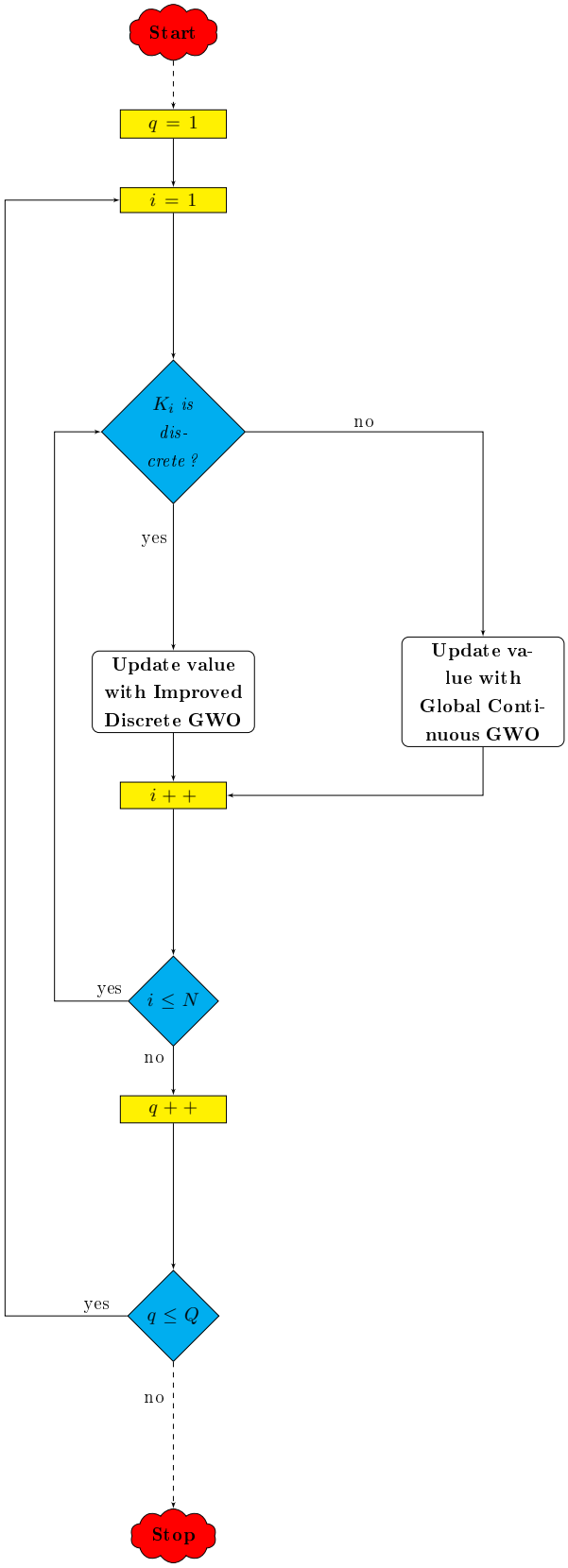


Figure A.1 — The mixedGWO's Update Wolves Step flowchart

APPENDIX B

Synthèse du manuscrit

B.1 Introduction générale

La Mesure d'Audience est l'acte de mesurer le nombre d'individus composant une audience. Ce terme est notamment utilisé pour les médias, quelque soit leur forme, et plus récemment pour les sites internet. La société IntuiSense, basée à Gemenos en France, est spécialisée dans le dévelopement d'interfaces pour distributeur automatique, et voudrait adapter le concept de la mesure d'audience sur ces distributeurs en utilisant de la vision assistée par ordinateur. Cela signifie qu'un distributeur devrait être capable, à l'aide d'une caméra embarquée, de détecter ses clients, les compter et les caractériser, $i.e.$ être capable de reconnaître son genre ou son âge. Un tel outil doit être capable de fonctionner à une cadence temp-réel et avoir une consommation de ressources mémoire aussi faible que possible tout en ayant un taux de précision suffisant. Une méthode classique et rapide utilisée pour la reconnaissance de genre est la classificatio d'image par Support Vector Machine (SVM). Cependant, la précision d'un SVM est extrêmement dépendante de ses paramètres d'entraînement et il n'existe pas de méthode globale pour définir ces paramètres pour un problème de classification donné, et sont, à l'heure actuelle, définis de manière empirique. Ces paramètres sont essentiellement continus, excepté dans le cas d'un kernel polynomial, où le degré du polyôme est une valeur discrète.

Habituellement, les travaux en traitement d'images s'effectuent sur des images composées des 3 couches principales Rouge, Vert et Bleu (RVB), qui correspondent respectivement aux longueurs d'onde de 690 nm, 550 nm et 450 nm dans le spectre de lumière. En imagerie multi-spectrale et hyper-spectrale, les images sont composées de plus de couches que les 3 couches basiques. De plus, ces couches ne sont pas limitées à la partie visible du spectre et peuvent être issues de l'Ultra-Violet ou de l'Infra-Rouge. Afin d'appliquer un démêlange de spectre sur une image multi-spectrale ou hyper-spectrale, l'image cible doit d'abord être débruitée. En revanche, cela ne doit pas être fait n'importe comment. En effet, une image trop fortement débruitée perdrait des informations nécessaires au démêlage de spectre tandis qu'une image pas assez débruitée risquerait de contenir trop d'informations. Il serait donc intéressant de pouvoir effectuer simultanément le débruitage et le démêlange de spectre afin de trouver le

meilleur équilibre entre ces deux étapes.
Il existe un verrou commun aux deux applications décrites précédemment : la reconnaissance de genre, et plus précisement la classification d'images par SVM, et le débruitage et démêlage simultané d'images multi-spectrales. Ces deux applications bénéficieraient de l'utilisation d'une méthode d'optimisation afin de déterminer leurs meilleurs paramètres.
Un problème d'optimisation consiste à maximiser ou minimiser une fonction, en choisissant de manière systématique des valeurs d'entrée issues d'un espace de recherche donné et en calculant la valeur de sortie de la fonction. Cependant, au fur et à mesure que les dimensions des problèmes augmentent, les méthodes d'optimisation classiques demandent une puissance de calcul de plus en plus élevée. Pour répondre à ce problème, de nouvelles méthodes moins gourmandes en ressources ont vu le jour, telles que les méthodes d'optimisation bio-inspirées [16]. Comme l'indique leur nom, les méthodes d'optimisation bio-inspirées trouvent leur inspiration dans le fonctionnement de Mère Nature. Le plus célèbre algorithme issu de la famille bio-inspirée est l'Algorithme Génétique (GA) [45], qui s'inspire de la théorie de l'évolution de darwiniste. Cette thèse se concentrera sur les méthodes d'optimisation de type "essaim", dont le représentant principale est l'algorithme du Particle Swarm Optimization (PSO) [56]. Un algorithme d'optimisation de type "essaim" va simuler le fonctionnement d'organismes fonctionnant les uns en fonction des autres. Par exemple, la méthode Ant Colony Optimization (ACO) simulera la manière dont les fourmis cherchent de la nourriture [29] tandis que la méthode du Grey Wolf Optimizer (GWO) simulera la méthode de chasse d'une horde de loups gris [92]. Cependant, les deux applications décrites précédemment sont composées de paramètres d'entrée appartenant à des espaces de valeur continus et discrets, la méthode d'optimisation à utiliser doit donc s'appliquer à des problèmes mixtes. Cependant, il n'existe pas de méthodes d'optimisation pour problèmes mixtes à ce jour.
Ce manuscrit de thèse a deux buts principaux :
• présenter un outil de Mesure d'Audience fonctionnant par traitement d'images et vision par ordinateur, et capable de fonctionner en temps réel avec une faible consommation processeur ;
• proposer une methode d'optimisation bio-inspirée capable d'optimiser des problèmes, soit totalement continus, soit totalement discrets, soit mixtes.

B.2 Corps du manuscrit

Dans cette section seront présentés les principaux enjeux et résultats correspondant à chaque chapitre de la thèse.

B.2.1 Résumé du chapitre 1 : Etude Bibliographique

B.2.1.1 La détection faciale en traitement d'images

En traitement d'images, la détection d'object consiste à pouvoir automatiquement determiner si un objet cible est présent ou non dans une image donnée et, si oui, de pouvoir donner sa position.

La méthode la plus célèbre, et qui est considérée comme la pierre angulaire dans le domaine de la détection d'objet en traitement d'image, est la méthode de Viola-Jones [123]. Proposée en 2001 par Paul Viola and Michael Jones dans [115], cette méthode est appliquée à la détection faciale et utilise des caractéristiques pseudo-Haar, Adaboost et une représentation appelée l'image Integral. Cette méthode présente l'avantage de pouvoir fonctionner en temps-réel et d'avoir une complexité algorithmique faible [55].

Avec la méthode de Viola-Jones, l'objet à détecter est décomposé en caractéristiques pseudo-Haar. La Figure B.1 montre les deux premières caractéristiques pseudo-Haar décrivant un visage humain.

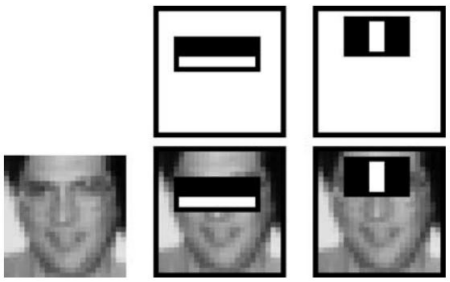


Figure B.1 — Les deux premières caractéristiques pseudo-Haar pour la detection de visages humain

Le détecteur de Viola-Jones en tant que tel est un classifieur en cascade créé avec AdaBoost et amélioré par l'utilisation d'image Integral [116]. Chacune des étapes de ce classifieur en cascade sont des sous-classifieurs, composés d'un ensemble de caractéristiques pseudo-Haar, et de leur seuil d'acceptation correspondant. Un schéma du fonctionnement d'un classifieur en cascade est disponible en Figure B.2.

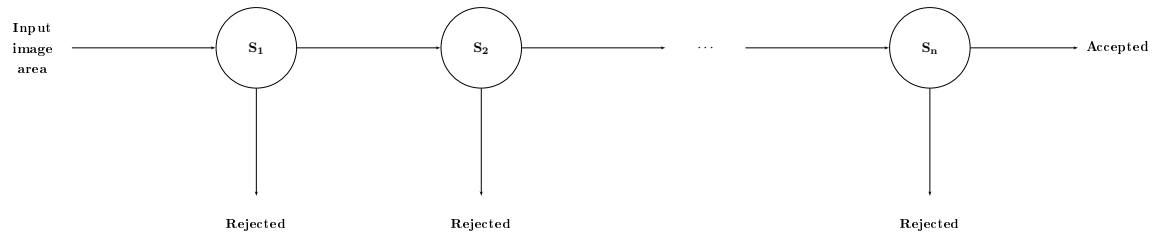


Figure B.2 — Fonctionnement d'un classifieur en cascade

Seules les zones de l'image étudiée ayant passées toutes les étapes du classifieur seront considérées comme étant l'objet recherché. Des exemples de résultats sont disponibles en Figure B.3.

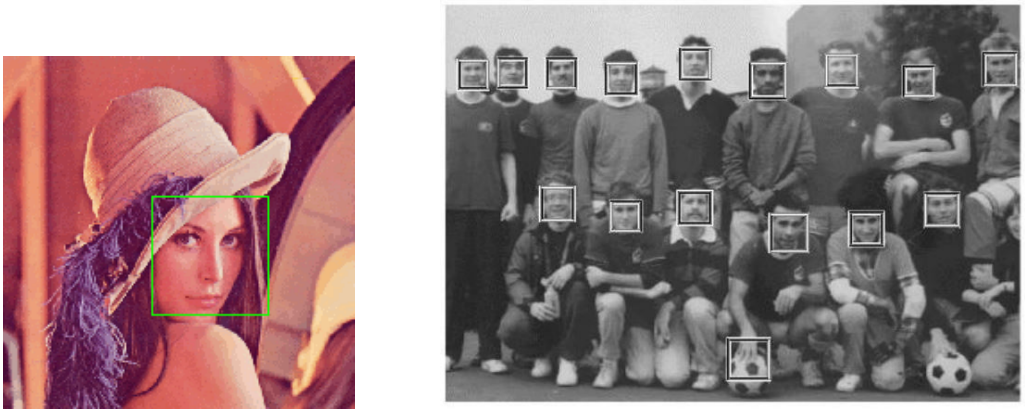


Figure B.3 — Exemples de résultats obtenus avec la méthode de Viola-Jones

B.2.1.2 Le suivi d'objet en traitement d'images

Le suivi d'objet en traitement d'images est le processus d'estimation de l'état, e.g. la position ou l'échelle, au cours du temps d'un objet en mouvement dans une vidéo ou dans une séquence d'images [125]. Il s'agit d'une problématique primordiale en vision par ordinateur qui possède une large gamme d'applications, telles que la vidéo surveillance, la navigation de robots ou la conduite de véhicules autonomes [120].
Dans un problème de suivi d'objet, connaître les composantes physiques de l'objet cible, *i.e.* sa couleur ou sa forme, peut-être utile. Dans [59], les informations colorimétriques de l'objet cible sont utilisées, tandis que dans [60], des informations de forme sont utilisées. Cependant, ce type d'information n'est pas toujours nécessaire et la méthode de suivi peut se contenter de connaître la position initiale de l'objet à suivre. Par exemple, la méthode de l'Optical Flow de Lucas-Kanade, proposée dans [81] et détaillée à la suite de ce paragraphe, nécessite simplement la position de point-clefs décrivant l'objet et suivra ces points d'une image à l'autre.

L'Optical Flow (ou Flux Optique) est la representation du mouvement, tel qu'il apparaît pour une caméra, d'objets de surfaces et de bords dans une scène causé par le mouvement relatif entre un observateur (un oeil ou une caméra) et la scène. L'idée de l'Optical FLow fut proposé par le psychologue James J. Gibson en 1950 pour décrire le stimuli visuel que subissent les animaux se mouvant dans le monde [39]. L'Optical Flow est souvent utilisé pour définir les mouvements entre deux trames (ou une séquence de trames) d'une vidéo, sans connaissance à-priori sur la contenance de ces trames. Typiquement, le mouvement en lui-même indiquera si quelque chose d'intéressant se produit. Le mouvement, caractérisé par Optical Flow, est utilisé en robotique pour du traitement d'images et du contrôle de navigation [12, 3].

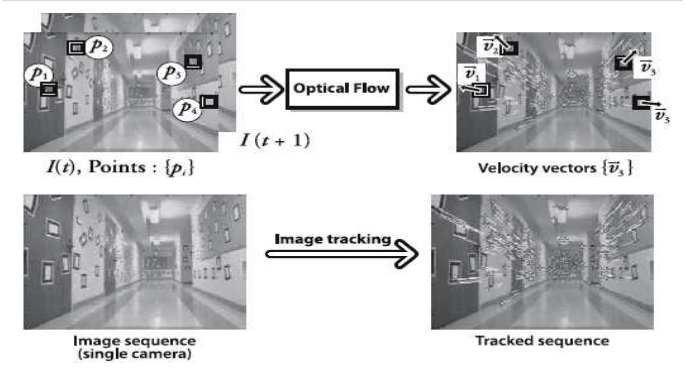


Figure B.4 — Optical Flow

Comme expliqué précédemment, l'Optical Flow permet de définir les mouvements entre deux trames (ou séquence de trames) dans une vidéo sans connaissance à-priori sur le contenu de ces frames. Un exemple de résultat obtenu grâce à l'Optical Flow est disponible en Figure B.4.

L'idée de base de l'algorithme de Lucas-Kanade (LK) se base sur trois hypothèses (voir Figure B.5) :

• *Luminosité constante* : Un pixel de l'image d'un objet dans la scène ne change pas (ou peu) d'apparence lors de ses potentiels mouvements d'une trame à l'autre. Dans le cas des images en nuances de gris, cela signifie que l'on considère que le niveau de gris d'un pixel donné de change pas alors quil est suivi d'une trame à l'autre.

• *Persistance temporelle ou "petits mouvements"* : Le mouvement d'un morceau surface évolue lentement au cours du temps. En pratique, cela signifie que l'incrémentation temporelle est suffisament élevée, comparée à l'échelle du mouvement dans une séquence vidéo, afin d'empêcher l'objet de bouger d'une trop grande distance d'une frame à l'autre.

•*Cohérence spatiale* : Les points voisins dans une scène appartiennent à la même surface, possèdent le même mouvement et se projettent vers les points proches dans le plan de l'image.

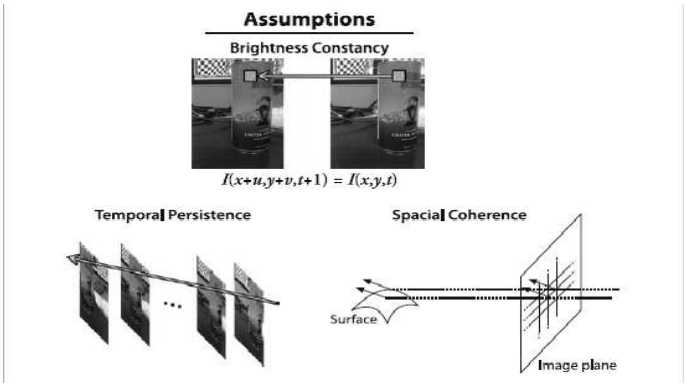


Figure B.5 — Hypothèses à propos de l'Optical Flow de Lucas-Kanade

Un potentiel problème pouvant apparaître lors de l'utilisation de LK est que, de par l'utilisation de petites fenêtres de recherche, les grands mouvements peuvent faire bouger les point-clefs suivis hors de cette petite fenêtre de recherche. Il devient donc impossible pour l'algorithme de trouver ces point-clefs. De plus, des mouvements larges ou incohérents sont souvent observés en pratique.
Ce problème a donc conduit au dévelopement d'un algorithme de Lucas-Kanade dit "pyramidal", qui suit un objet cible en commençant par le plus haut niveau d'une image pyramidale (précision des détails faible) et de descendre les étages au fur et a mesure (affiner la précision des détails). Ainsi, les hypothèses de mouvements sont respectées et il est possible de suivre des mouvements plus larges et plus rapides. Cette méthode de l'Optical flow de Lucas-Kanade "pyramidal" est illustrée en Figure B.6.
Dans [51], le calcul d'une erreur "aller-retour" est introduit à l'Optical Flow classique afin d'en améliorer la précision. Ce type d'Optical Flow est appelé le Median Flow.

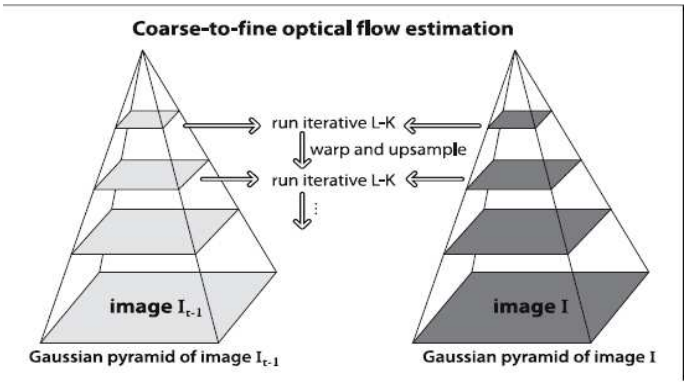


Figure B.6 — Optical Flow de Lucas-Kanade "pyramidal"

B.2.1.3 La reconnaissance d'objet en traitement d'images

En traitement d'images et vision par ordinateur, la reconnaissance d'objet est le fait de pouvoir automatiquement mettre une étiquette sur un objet, e.g. définir si un visage donné est un visage d'homme ou de femme. Pour pouvoir atteindre un tel objectif, il est nécessaire de passer par de la classification d'images. La classification d'images est habituellement faite à l'aide d'algorithmes de mahcine learning tels que les Réseaux de Neurones Conventionnels (CNN) [65] ou les Support Vector Machine (SVM) [69]. La reconnaissance d'objets peut aussi s'effectuer de manière plus simple en utilisant des distances (e.g. euclidienne ou de mahalanobis) [19] ou en utilisant un classifieur des "plus proches voisins" [2].

Pour pouvoir faire de la classification d'images, des images d'entraînement sont nécessaires, *i.e.* un ensemble d'images représentant l'objet que l'algorithme de classification utilisera afin "d'apprendre" l'objet. Le nombre d'images d'entraînement utilisé est un important facteur pour la reconnaissance d'objets, en effet un nombre suffisant d'images permettra au système de reconnaissance d'avoir un bon taux de précision tandis qu'un nombre trop faible d'images réduirai ce taux de précision [36]. De nombreuses bases de données d'images ont été créées dans le domaine de la reconnaissance et caractérisation faciale, telles que la FERET database [103], la YALE Face database [37], la ORL Database of Faces [108], la Labeled Face in the Wild database [49] et encore beaucoup d'autres.
De plus, un système de reconnaissance d'objet ne travaille pas directement sur une image brute : il est nécessaire d'en extraire la représentation de l'objet, *i.e.* des informations utiles issues des images, que ce soient des images d'entraînement ou des images d'entrée [109]. Une approche classique serait d'appliquer une méthode statistique telle que l'Analyse en Composante Principale (ACP) [113] ou une Analyse Linéaire Discriminante (ALD) [34] aux images. Une autre approche est d'utiliser des descripteurs pour décrire, soit la texture, soit la forme de l'objet étudié. Parmi les descripteurs les plus utilisés dans le domaine de la reconnaissance et de la caractérisation faciale on trouve : les Local Binary Patterns (LBP)

[2], les Histogram of Oriented Gradients (HOG) [74] et les descripteurs SIFT [35]. Dans le domaine de la reconnaissance de genre, afin d'améliorer le taux de bonne reconnaissance, ces descripteurs ont été combinés avec des Histogrammes de Couleur (CH) ou des caractéristiques de Gabor [109].
Dans ce résumé, seule la méthode du SVM sera expliquée en détail.

Une Machine à Vecteur de Support (ou Support Vector Machine ou SVM) est un concept en informatique et statistique lié à un ensemble de techniques d'apprentissage supervisé destiné à analyser des données et reconnaître des objets. Lors d'une utilisation en classification, le SVM cherche un hyperplan optimal en tant que fonction de décision dans un espace à grandes dimensions [114, 101]. Pour un problème de reconnaissance d'objet bi-classe, si on suppose les classes linéairement séparables, le SVM choisit une frontière de décision linéaire qui minimise l'erreur générale. La frontière de décision choisie est représentée par un hyperplan dans l'espace et possède la plus grande marge possible entre les deux classes. Ici, la marge est définie comme la somme des distances entre les éléments de chaque classes les plus proches de l'hyperplan.

Dans le cas le plus simple, c'est à dire une classification bi-classes avec des classes linéairement séparables, les données d'entraînement comprennent l cas : $\{\mathbf{x}_i, y_i\}, i = 1, \cdots, l$, où $\mathbf{x} \in \mathbb{R}^N$ est un espace à N dimension et $y \in \{-1, +1\}$ est le label des classes. Les objets d'entraînement sont linéairement séparables si il existe un vecteur $\mathbf{w}$ et un scalaire b tels que :

$$y_i(\mathbf{w}^T \cdot \mathbf{x}_i) - b \geq 0$$

où $\mathbf{w}$ représente l'orientation d'un plan discriminant et b détermine l'offset de ce plan discriminant par rapport à l'origine. L'espace hypothétique peut être défini par l'ensemble de fonctions donné par :

$$f_{\mathbf{w}^T,b} = \text{sign}((\mathbf{w} \cdot \mathbf{x}_i) + b). \tag{B.1}$$

Le SVM trouve les hyperplans séparant les données pour lesquels la distance entre les classes est maximum en résolvant le problème d'optimisation contraint :

$$\min_{\mathbf{w}} \frac{1}{2} \parallel \mathbf{w} \parallel^2 .$$

Dans le cas des classes non séparables linéairement, le SVM recherche l'hyperplan maximisant la marge et minimisant une quantité proportionnelle au nombre d'erreurs de mauvaises classifications en introduisant une variable de relaxation $\varsigma_i \geq 0$. Le problème d'optimisation contraint pour des données non séparables devient alors :

$$\min_{\mathbf{w},\varsigma_1,\cdots,\varsigma_l} \left[\frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_{i=1}^{l} \varsigma_i \right] \tag{B.2}$$

où $0 < C < \infty$ est une constante positive permettant un compromis entre la marge et l'erreur de mauvaises classifications.

Dans le cas des surfaces de décision non-linéaires, un vecteur de caractéristique $\mathbf{x} \in \mathbb{R}^N$ est caractérisé dans un espace euclidien $\mathcal{F}$ de dimension plus élevée via une fonction non-linéaire $\varphi : \mathbb{R}^N \mapsto \mathcal{F}$ [101]. Le problème de la marge optimale dans $\mathcal{F}$ peut être obtenu en remplaçant $\mathbf{x}_i^T$ avec $\varphi(\mathbf{x}_i)^T$. Pour répondre à ce problème, Vapnik [114] proposa une manière de créer un SVM non-linéaire en appliquant une fonction de kernel pour maximiser la marge des hyperplans. Une fonction de kernel $\mathcal{K}$ est définie par :

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^T \cdot \varphi(\mathbf{x}_j) \tag{B.3}$$

et avec la fonction de kernel ci dessus, l'Eq. (B.1) devient :

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i \in \mathbb{S}} \zeta_i y_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b \right) \tag{B.4}$$

où ζ_i est un multiplicateur de Lagrange non nul, $\mathbb{S}$ est un sous-espace d'indices $\{1, 2, \cdots, l\}$ correspondant à ζ_i et définissant les fameux vecteurs de support (SVs).

En introduisant le kernel, les SVM gagnent en flexibilité dans le choix de la forme de l'hyperplan séparant les classes, la linéarité n'est pas nécessaire et les données ne nécessitent pas de toutes avoir la même forme fonctionnelle. Si les paramètres sont choisis de manière appropriée, comme par exemple le paramètre C, les SVM peuvent être suffisamment robustes même quand les données d'entraînement sont biaisés, *i.e.* ils sont plus robustes au bruit. Un SVM possède une solution unique, étant donnée que le problème d'optimalité est convexe [7]. Deux autres types de kernels sont aussi utilisés pour les SVM : le kernel gaussien (RBF) et le kernel polynomial, dont les expressions sont en Eq. (B.5) et Eq. (B.6) respectivement.

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{||\varphi(\mathbf{x}_i) - \varphi(\mathbf{x}_j)||^2}{2\gamma^2} \right) \tag{B.5}$$

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \left(\varphi(\mathbf{x}_i)^T \cdot \varphi(\mathbf{x}_j) + 1 \right)^d \tag{B.6}$$

B.2.1.4 L'optimisation bio-inspirée

Un problème d'optimisation consiste à maximiser ou minimiser une fonction, en choisissant de manière systématique des valeurs d'entrée issue d'un espace de recherche donné et en calculant la valeur de sortie de la fonction. Cependant, au fur et à mesure que les dimensions des problèmes augmentent, les méthodes d'optimisation classiques demande une puissance de calcul de plus en plus élevées. Pour répondre à ce problème, de nouvelles méthodes moins gourmande en ressources ont vu le jour, telle que les méthodes d'optimisation bio-inspirées

[16]. Les méthodes d'optimisation bio-inspirées existantes peuvent être séparées en deux catégories : les méthodes à espaces de recherche continus et les méthodes à espaces de recherche discrets.

Parmi les algorithmes évolutifs, le plus connu est l'algorithme génétique (GA). Proposé par Holland en 1992 [45], cet algorithme simule la théorie de l'évolution darwiniste. Une méthode fondatrice sur les méthodes d'optimisation de type "essaim" est le Particle Swarm Optimization (PSO) [56, 31], proposé par Kennedy et Eberhart, qui est lui-même inspiré de [106] où le terme de "essaim de particules" est utilisé pour décrire les membres d'une population ou d'un ensemble de tests. Le principe des essaims fut d'abord inspiré par le comportement des oiseaux, où un leader guide le reste de la flotte. Dans [92], une méthode utilisant trois leaders différent est proposée. Cette méthode, qui s'inspire du comportement des loups gris, se nomme le Grey Wolf Optimizer (GWO). L'algorithme du GWO simule la hiérarchie de commandement et la méthode de chasse d'une horde de loup gris dans la nature. Quatre types de loups (le alpha, le bêta, le delta et le oméga) sont utilisés pour simuler la hiérarchie de commandement. De plus, les trois grandes étapes de la chasse sont implémentées : la recherche de la proie, l'encerclement de la proie et l'attaque de la proie.
De nombreuses variantes pour améliorer les performances du GWO originel ont été proposées, telles qu'une version hybride de GWO+PSO [53] et un "modified GWO" (mGWO) [95] qui inclut une règle de mise à jour modifiée. Ces méthodes ne peuvent être utilisées que pour des problèmes dans des espaces de recherche strictement continus.

Une méthode d'optimisation pour espace de recherche discrets va chercher à estimer les meilleurs paramètres issus d'un espace de recherche discret afin de minimiser ou de maximiser une fonction.
Une célèbre méthode d'optimisation bio-inspirée pour problèmes discrets est la méthode du Ant Colony Optimization (ACO) [30]. Le ACO prend son inspiration dans le comportement des fourmis lorsqu'elles cherchent de la nourriture. Cet algorithme, très performant sur des problèmes de type Problème du Voyageur de Commerce, va chercher la combinaison optimale entre un ensemble de points dans un espace afin de réduire la distance de trajet.
Certaines méthodes d'optimisation bio-inspirée originellement créées pour des problèmes continus ont été adaptées pour des problèmes discrets. Par exemple, différentes versions du PSO adapté pour des problèmes binaires ont été proposées [57, 13]. Une version discrète du GWO, non limitée aux problèmes binaires, a également été proposée [80] en formulant un modèle de programmation entière multi-objectifs.

B.2.1.5 Conclusion du chapitre 1

Dans ce résumé du Chapitre 1, un bref état de l'art de deux domaines distincts a été fait :
• la mesure d'audience en traitement d'images et vision par ordinateur, qui se compose de la détection faciale, le suivi d'objets et la reconnaissance faciale ;
• les méthodes d'optimisation bio-inspirées, et plus particulièrement les méthodes d'optimisation à essaim de particules.
À propos de la mesure d'audience, les méthodes qui semblent les plus adaptées pour un outil fonctionnant en temps-réel sur un système embarquée se démarquent clairement. Pour la détection faciale, l'algorithme de Viola-Jones, expliqué en B.2.1.1 apparaît comme la plus pertinente, bien qu'elle soit une des plus ancienne méthode. En effet, elle possède une faible complexité, seulement dépendante de la taille de l'image cible [55], et a déjà prouvé sa capacité à fonctionner en temps-réel [116]. Cependant, un travail sera nécessaire afin de pallier aux fausses détections. Pour le suivi d'objet, l'Optical Flow de Lucas-Kanade, décrit en sous-section B.2.1.2, apparaît comme la méthode la plus adaptée. En effet, elle est rapide et simple à calculer, et les objets à suivre, des visages, possèderont des trajectoires inconnues et non-répétitives. Enfin, pour la classification d'images devant aboutir à la reconnaissance de genre, le SVM semble être le plus adapté. En effet, il s'agit d'avoir un système bi-classe (visage d'hommes et de femmes) qui doit fonctionner de manière offline, $i.e.$ qu'aucun apprentissage supplémentaire au cours du fonctionnement du système ne sera possible. C'est pourquoi le SVM semble plus adapté à la problématique visée que un Réseaux de Neurones Conventionnel. Cependant, une attention particulière doit être accordée à l'entraînement du SVM afin d'éviter une mauvaise précision et une perte de temps dûe à des entraînements répétitifs.
Dans le cas des méthodes d'optimisation bio-inspirée, il apparaît que toutes les méthodes de l'état de l'art sont, soit destinées à des problèmes exclusivement continus, soit destinées à des problèmes exclusivement discrets. Des méthodes d'optimisation bio-inspirées pour problèmes mixtes, $i.e.$ contenant des variables continues et des variables discrètes, n'existent pas encore.

B.2.2 Résumé du chapitre 2 : Le mixed Grey Wolf Optimizer

Le but de ce Chapitre est de proposer un algorithme d'optimisation bio-inspirée capable de résoudre des problèmes contenant des variables discrètes et continues. Dans ce rapport, ces problèmes sont appelés des **problèmes mixtes**. De plus, il est pertinent d'avoir un outil capable de s'adapter au type de problème rencontré. C'est pourquoi l'algorithme proposé devra être capable de gérer des problèmes entièrement continus et des problèmes entièrement discrets en plus des problèmes mixtes. Étant plus robuste que le PSO vis à vis de l'évitement des minima locaux et de sa vitesse de calcul, le GWO a été choisi comme base

pour l'algorithme proposé.
Le processus pour les variables discrètes est appliqué à un sous-ensemble de paramètres attendus, prenant leur valeur dans un espace de recherche discret, tandis que le processus pour les variables continues est appliqué, indépendamment, aux paramètres prenant leur valeur dans un espace de recherche continu.

Tout d'abord, en sous-section B.2.2.1, un algorithme d'optimisation discrète nommé Improved Discrete GWO (IDGWO) est proposé. Bien qu'inspiré du GWO classique, son pendant continu n'est pas exactement comme la version classique du GWO. Ce pendant continu, nommé Global Continuous GWO, est expliqué en sous-section B.2.2.2. Ces deux méthodes sont combinées pour créer un mixedGWO capable de gérr des problèmes continus, discrets et mixtes. Enfin, en sous-section B.2.2.3, le mixedGWO proposé est décrit plus en détails.

Pour plus d'informations sur les différentes notations qui seront utilisées dans ce Chapitre, veuillez vous référer à la Nomenclature disponible en début de manuscrit.

B.2.2.1 Improved discrete Grey Wolf Optimizer

Les méthodes d'optimisation bio-inspirées discrètes existantes sont souvent destinées à des problèmes binaires ou des problèmes spécifiques [79, 80]. N'étant donc pas applicables de manière globale, il est pertinent de développer une version discrète du GWO qui doit être applicable de manière globale.

L'idée du Improved Discrete GWO (IDGWO) est que les espaces de recherches discrets sont définis sous forme de vecteurs de valeurs, mais que leurs indices soient également sous forme de vecteurs. Une particule ou un "loup" à l'itération iter est défini par $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter})$ et contient N composantes qui sont les valeurs candidates des paramètres attendus.
Une valeur de la composante i avec un index h_i dans $\mathbf{d}_i^{ind}$ est définie par $K_i^{h_i}$. Par exemple, pour le vecteur de valeurs $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}) = \left[K_1^2, \ldots, K_N^5\right]^T$, le vecteur d'indices associé sera $\mathbf{h}_q(\mathrm{iter}) = [2, \ldots, 5]^T$.

L'Algorithme 8 détaille le processus global du IDGWO tandis que dans l'Algorithme 9, la loi de mise à jour d'un loup d'indice $q \in [1, \ldots, Q]^T$ à l'itération iter est détaillée.

Algorithm 8 Pseudo-code : Improved Discrete Grey Wolf Optimization pour l'estimation multi-paramètres

Entrées : fonction de fitness, nombre N de paramètres attendus, faible facteur ϵ défini par l'utilisateur pour arrêter l'algorithme, nombre maximum d'itérations T_{max}.
Pour chaque paramètre indexé par $i = 1, \ldots, N$: l'espace de recherche $\mathbf{d}_i^{ind}$ avec H_i valeurs possibles.

1. Initialiser la valeur d'itération iter $= 1$.
   Créer un ensemble initial de vecteurs d'indices $\mathbf{h}_q(\text{iter})$, $q = 1, \ldots, Q$. Pour chaque indice i compris entre 1 et N, un composant $h_i(\text{iter})$ de $\mathbf{h}_q(\text{iter})$ est une valeur entière comprise entre 1 et H_i.
   Création d'une horde initiale composée de Q loups $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$ avec les N paramètres attendus nécessaires. Cette population initiale se présente sous la forme d'une matrice avec Q lignes et N colonnes.
2. Calculer la valeur de la fonction de fitness $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ de chaque loup $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Trier les loups selon leur valeur de fitness et mettre à jour les 1ère, 2ème et 3ème meilleures valeurs de fitness : stocker les vecteurs d'indices $\mathbf{h}_\alpha$, $\mathbf{h}_\beta$, $\mathbf{h}_\delta$ et les vecteurs de valeurs $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, $\mathbf{x}_\delta^{\mathbf{k}}$ correspondants.
4. Si $a > 1$, sélectionner deux loups $\rho 1$ et $\rho 2$ aléatoirement parmi la horde de Q loups, avec $\rho 1 \neq \rho 2$. Stocker les vecteurs d'indices $\mathbf{h}_{\rho 1}$, $\mathbf{h}_{\rho 2}$ et les vecteurs de valeurs $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, $\mathbf{x}_{\rho 2}^{\mathbf{k}}$ correspondants.
   Sinon, si $a \leq 1$, aller à l'étape 5.
5. Répéter les étapes pour chaque loup q, $q = 1, \ldots, Q$, avec les vecteurs de valeur $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ et les vecteur d'indices $\mathbf{h}_q(\text{iter})$:
   Appliquer Algorithme 9.
6. Remplacer la population actuelle avec la nouvelle, obtenue à l'étape 5.
7. Si iter $<$ T$_{\max}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, incrémenter iter, et aller à l'étape 2.

Sortie : Valeurs des paramètres estimés $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$ contenue dans $\mathbf{x}_\alpha^{\mathbf{k}}$.

Algorithm 9 Pseudo-code : Règles de mise à jour du Improved Discrete Grey Wolf Optimization pour l'estimation multi-paramètres

Entrées : Vecteur d'indices $\mathbf{h}_q(\text{iter})$, vecteur de valeurs $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$.

1. Choix du leader :
   (a) Soit $\mathbf{x}_{\text{l}}^{\mathbf{k}}$ le vecteur de valeurs du leader choisi, choisi aléatoirement parmi $\mathbf{x}_{\alpha}^{\mathbf{k}}$, $\mathbf{x}_{\beta}^{\mathbf{k}}$, $\mathbf{x}_{\delta}^{\mathbf{k}}$, $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, et $\mathbf{x}_{\rho 2}^{\mathbf{k}}$. Le processus de selection du leader est détaillé dans les Eqs. (B.7) à (B.8).
   (b) Stocker le vecteur d'indices correspondant à $\mathbf{x}_{\text{l}}^{\mathbf{k}}$ dans un vecteur noté $\mathbf{h}_{\text{l}}$.
2. Mise à jour du loup q :
   Pour chaque indice $i = 1, \ldots, N$
   (a) Calculer le composant $h_i(\text{iter}+1)$.
   (b) Calculer le composant $x_i(\text{iter}+1)$.

   Ce processus est détaillé dans les Eqs. (B.9), (B.11) et (B.12)
3. Stocker les N composants obtenus aux étapes 2a et 2b :
   (a) Stocker le nouveau vecteur d'indices :
   $\mathbf{h}_q(\text{iter}+1) = [h_1(\text{iter}+1), \ldots, h_i(\text{iter}+1), \ldots, h_N(\text{iter}+1)]^T$.
   (b) Stocker le nouveau vecteur de valeurs :
   $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1) = [x_1(\text{iter}+1), \ldots, x_i(\text{iter}+1), \ldots, x_N(\text{iter}+1)]^T$.

Sortie : $\mathbf{h}_q(\text{iter}+1)$, $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$

Voici quelques détails sur la loi de mise à jour de l'IDGWO :

À l'étape 1, si $a > 1$, le leader est choisi aléatoirement parmi les loups α, β, δ, $\rho 1$ et $\rho 2$:

$$\mathbf{x}_{\text{l}}^{\mathbf{k}} = \begin{cases} \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{si} \quad r \leq \frac{a}{10} \\ \mathbf{x}_{\beta}^{\mathbf{k}} & \text{si} \quad r > \frac{a}{10} \quad \text{et} \quad r \leq \frac{2a}{10} \\ \mathbf{x}_{\delta}^{\mathbf{k}} & \text{si} \quad r > \frac{2a}{10} \quad \text{et} \quad r \leq \frac{3a}{10} \\ \mathbf{x}_{\rho 1}^{\mathbf{k}} & \text{si} \quad r > \frac{3a}{10} \quad \text{et} \quad r \leq \frac{4a}{10} \\ \mathbf{x}_{\rho 2}^{\mathbf{k}} & \text{si} \quad r > \frac{4a}{10} \quad \text{et} \quad r \leq \frac{5a}{10} \\ \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{si} \quad r > \frac{5a}{10} \end{cases} \tag{B.7}$$

avec r valeur aléatoire dans $\mathbb{R}$, comprise entre 0 et 1.

si $a \leq 1$, le leader est choisi aléatoirement parmi les loups α, β, et δ :

$$\mathbf{x}_{\mathrm{l}}^{\mathbf{k}} = \begin{cases} \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{si} \quad r \leq \frac{a}{6} \\ \mathbf{x}_{\beta}^{\mathbf{k}} & \text{si} \quad r > \frac{a}{6} \quad \text{et} \quad r \leq \frac{2a}{6} \\ \mathbf{x}_{\delta}^{\mathbf{k}} & \text{si} \quad r > \frac{2a}{6} \quad \text{et} \quad r \leq \frac{3a}{6} \\ \mathbf{x}_{\alpha}^{\mathbf{k}} & \text{si} \quad r > \frac{3a}{6} \end{cases} \tag{B.8}$$

avec r valeur aléatoire dans $\mathbb{R}$, comprise entre 0 et 1.

À l'étape 2a, chaque composant du vecteur d'indices $\mathbf{h}_q(\mathrm{iter})$ est mis à jour comme suit : Pour chaque indice i, $i = 1, \ldots, N$:

$$h_i(\mathrm{iter}+1) = (h_i(\mathrm{iter}) + \Delta \; sgn(h_i^{\mathrm{l}} - h_i(\mathrm{iter}))) \; mod \; H_i \tag{B.9}$$

avec :

— $sgn(\cdot)$ correspond à la fonction *signe*, de telle manière que $sgn(z) = -1$ si $z < 0$, $sgn(z) = 0$ si $z = 0$, et $sgn(z) = 1$ si $z > 0$ pour n'importe quelle valeur réelle z ;

— mod correspond à l'opérateur "Modulo", défini comme suit : pour n'importe quelle valeur réelle $u \in \mathbb{R}_+$ and $v \in \mathbb{R}_+^*$:

$$u \; mod \; v = \begin{cases} u - v\lfloor u/v \rfloor & \text{si} \quad u \neq v \\ v & \text{si} \quad u = v, \text{ or } u = 0 \end{cases} \tag{B.10}$$

où $\lfloor \cdot \rfloor$ correspond à la partie entière.

— Δ est calculé comme suit :

$$\Delta = \begin{cases} 1 & \text{si} \quad \phi \leq \frac{a}{2\Omega(H_i)} \\ 2 & \text{si} \quad \phi > \frac{a}{2\Omega(H_i)} \quad \text{et} \quad \phi \leq \frac{2a}{2\Omega(H_i)} \\ \vdots & \\ n_i & \text{si} \quad \phi > \frac{(n_i-1)a}{2\Omega(H_i)} \quad \text{et} \quad \phi \leq \frac{n_i a}{2\Omega(H_i)} \\ \vdots & \\ \Omega(H_i) - 1 & \text{si} \quad \phi > \frac{(\Omega(H_i)-2)a}{2\Omega(H_i)} \quad \text{et} \quad \phi \leq \frac{(\Omega(H_i)-1)a}{2\Omega(H_i)} \\ \Omega(H_i) & \text{si} \quad \phi > \frac{(\Omega(H_i)-1)a}{2\Omega(H_i)} \quad \text{et} \quad \phi \leq \frac{\Omega(H_i)a}{2\Omega(H_i)} \\ 1 & \text{si} \quad \phi > \frac{\Omega(H_i)a}{2\Omega(H_i)} \end{cases} \tag{B.11}$$

où ϕ est valeur aléatoire dans $\mathbb{R}$, comprise entre 0 et 1, et :

$$\Omega(H_i) = \begin{cases} \frac{H_i}{2} & \text{si } H_i \text{ pair} \\ \frac{H_i+1}{2} & \text{si } H_i \text{ impair} \end{cases}$$

Enfin, à l'étape 2b, chaque composante du vecteur de valeurs mis à jour $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}+1)$ est calculée suivant l'Eq. (B.12) puis stockée dans le vecteur $\mathbf{x}_q^{\mathbf{k}}(\mathrm{iter}+1)$.

$$x_i(\text{iter}+1) = \mathbf{d}_i^{val}(h_i(\text{iter}+1)) \tag{B.12}$$

B.2.2.2 Global Continuous Grey Wolf Optimizer

L'Algorithme 10 détaille la loi de mise à jour du pendant continu de l'IDGWO : le Global Continuous GWO (GCGWO), et plus particulièrement la mise à jour d'un paramètre d'indice $i \in [1, \dots, N]^T$ pour un loup d'indice $q \in [1, \dots, Q]^T$ à l'itération iter.

Algorithm 10 Pseudo-code : Global Continuous Grey Wolf Optimization pour l'estimation multi-paramètres

Entrées : $x_i(\text{iter})$, $i^{\text{ème}}$ composant d'un loup $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ à l'itération iter ; les leaders α, β, δ.

1. Calculer les contributions y_i^α, y_i^β, et y_i^δ respectivement des loups α, β, et δ par rapport au $q^{\text{ème}}$ loup.
   Ce calcul est détaillé dans les Eqs. (B.15) et (B.16).
2. si $a > 1$ aller à l'étape 3, sinon si $a \leq 1$ aller à l'étape 4
3. Calculer les contributions $y_i^{\rho 1}$ et $y_i^{\rho 2}$ respectivement des loups $\rho 1$ et $\rho 2$ par rapport au $q^{\text{ème}}$ loup.
   Ce calcul est détaillé dans les Eqs. (B.15) et (B.16).
4. Calculer la nouvelle position du $i^{\text{ème}}$ composant du $q^{\text{ème}}$ loup :
   si $a > 1$:
   $$x_i(\text{iter}+1) = \frac{1}{5}(y_i^\alpha + y_i^\beta + y_i^\delta + y_i^{\rho 1} + y_i^{\rho 2}) \tag{B.13}$$
   sinon si $a \leq 1$:
   $$x_i(\text{iter}+1) = \frac{1}{3}(y_i^\alpha + y_i^\beta + y_i^\delta) \tag{B.14}$$

Sortie : $x_i(\text{iter}+1)$

La nouvelle position mentionnée à l'étape 4 de l'Algorithme 10 est calculé comme la contribution équivalente des leaders α, β et δ. Si $a > 1$, les contributions des loups aléatoires $\rho 1$ et $\rho 2$ sont aussi utilisées. Ces contributions sont calculées comme suit :

$$\begin{cases} y_i^\alpha = x_i^\alpha - b_1 \cdot d_i^\alpha, \\ y_i^\beta = x_i^\beta - b_2 \cdot d_i^\beta, \\ y_i^\delta = x_i^\delta - b_3 \cdot d_i^\delta, \\ y_i^{\rho 1} = x_i^{\rho 1} - b_4 \cdot d_i^{\rho 1}, \\ y_i^{\rho 2} = x_i^{\rho 2} - b_5 \cdot d_i^{\rho 2} \end{cases} \tag{B.15}$$

avec :

$$\begin{cases} d_i^{\alpha} = |c_1 \cdot x_i^{\alpha} - x_i(\text{iter})|, \\ d_i^{\beta} = |c_2 \cdot x_i^{\beta} - x_i(\text{iter})|, \\ d_i^{\delta} = |c_3 \cdot x_i^{\delta} - x_i(\text{iter})|, \\ d_i^{\rho 1} = |c_4 \cdot x_i^{\rho 1} - x_i(\text{iter})|, \\ d_i^{\rho 2} = |c_5 \cdot x_i^{\rho 2} - x_i(\text{iter})| \end{cases} \tag{B.16}$$

où les scalaires b et c sont calculés de la même façon que dans le GWO classique : $b = 2ar_1 - a$ et $c = 2r_2$. Dans ces expressions, r_1 et r_2 sont des scalaires aléatoires compris entre 0 et 1.

B.2.2.3 Extension au mixedGWO

Dans un même problème, les paramètres à utiliser n'appartiennent pas forcément au même espace de recherche. De plus, certains de ces paramètres peuvent être continus tandis que les autres seront discrets. De plus, ces paramètres peuvent être interdépendants et doivent donc être estimés simultanément. De tels problèmes seront appelés des problèmes mixtes. L'association du IDGWO, expliqué en sous-section B.2.2.1, et du GCGWO, expliqué en sous-section B.2.2.2, permet de proposer un algorithme d'optimisation bio-inspiré capable de gérer des problèmes mixtes : le mixedGWO. Les notations suivantes seront utilisées spécifiquement pour le mixedGWO, en plus des notations présentées dans la nomenclature en début de manuscrit :

— En présumant que le $i^{\text{ème}}$ paramètre K_i soit une valeur issue d'un espace de recherche continu, sa valeur minimum possible est notée K_i^{min} et sa valeur maximum possible est notée K_i^{max} ;
— L'intervalle de valeur possible pour le paramètre K_i dans un espace de recherche continu est noté $\mathbf{d}_i^{val} = \left[K_i^{min}; K_i^{max}\right]^T$.

L'algorithme 11 décrit la méthode d'optimisation mixedGWO proposée tandis que les Figures A.0 et A.1 , disponibles dans en annexe, décrivent son algorigramme.

Algorithm 11 Pseudo-code : Mixed Grey Wolf Optimization pour l'estimation multi-paramètres

Entrées :fonction de fitness, nombre N de paramètres attendus, faible facteur ϵ défini par l'utilisateur pour arrêter l'algorithme, nombre maximum d'itérations T_{max}.

1. Initialiser la valeur d'itération iter $= 1$. Créer un ensemble initial de vecteurs d'indices $\mathbf{h}_q(\text{iter})$, $q = 1, \ldots, Q$. Pour chaque indice i compris entre 1 et N, un composant $h_i(\text{iter})$ de $\mathbf{h}_q(\text{iter})$ est une valeur entière comprise entre 1 et H_i.
   Création d'une horde initiale composée de Q loups $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$ avec les N paramètres attendus nécessaires. Cette population initiale se présente sous la forme d'un matrice avec Q lignes et N colonnes.
2. Calculer la valeur de la fonction de fitness $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ de chaque loup $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Trier les loups selon leur valeur de fitness et mettre à jour les loups α, β, et δ qui possèdent respectivement la meilleur, 2ème et 3ème valeurs de fitness. Stocker leur position dans les vecteurs $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, et $\mathbf{x}_\delta^{\mathbf{k}}$ respectivement. Dans le cas des paramètres discrets, stocker également les vecteurs d'indices correspondants $\mathbf{h}_\alpha$, $\mathbf{h}_\beta$ et $\mathbf{h}_\delta$.
4. Si $a > 1$, sélectionner deux loups $\rho 1$ et $\rho 2$, aléatoirement parmi la horde de Q loups, avec $\rho 1 \neq \rho 2$. Stocker les vecteurs de valeurs $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, $\mathbf{x}_{\rho 2}^{\mathbf{k}}$ et, pour les paramètres discrets, les vecteurs d'indices correspondants $\mathbf{h}_{\rho 1}$ et $\mathbf{h}_{\rho 2}$.
   Sinon si $a \leq 1$, aller à l'étape 5.
5. Répéter les étapes pour chaque loup $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$:
   Pour chaque composant $x_i(\text{iter})$ avec $i = 1, \ldots, N$:
   (a) si le $i^{\text{ème}}$ paramètre K_i est un paramètre continu, alors aller à l'étape 5b, sinon si K_i est un paramètre discret, alors aller à l'étape 5c.
   (b) Appliquer la loi de mise à jour continue proposée dans l'Algorithme 10. Sauter les étapes 5c à 5e.
   (c) Choisir le leader $\mathbf{x}_1^{\mathbf{k}}$, et le coefficient de déplacement Δ qui seront utilisés pour les paramètres discrets. Pour cela, se référer aux Eqs. (B.7), (B.8), (B.9), et (B.11).
   (d) Apliquer l'étape 2a de l'Algorithme 8 afin d'obtenir $h_i(\text{iter} + 1)$ comme indiqué dans l'Eq. (2.3)
   (e) Apliquer l'étape2b de l'Algorithme 8 afin d'obtenir $x_i(\text{iter} + 1)$ comme indiqué dans l'Eq. (B.12)
6. Remplacer la population actuelle avec la nouvelle, obtenue à l'étape 5
7. Si iter $<$ T_{max} ou $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, incrémenter iter et aller à l'étape 2.

Sortie : Valeurs des paramètres estimés $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$

Dans la suite de ce résumé, une distinction sera faite entre deux versions du mixedGWO, dépendantes de l'expression du paramètre a. Dans le mixedGWO simple, le paramètre a est calculé comme suit :

$$a = 2(1 - \frac{\text{iter}^{\eta}}{{T_{max}}^{\eta}}) \tag{B.17}$$

Dans une seconde version appelée adaptive mixed GWO (et notée amixedGWO), le paramètre a est calculé comme suit :

$$a = \begin{cases} 2(1 - \frac{\text{iter}^{\eta}}{{T_{max}}^{\eta}}) & \text{si} \quad \text{iter} \leq \text{T}_{\text{max}}/2 \\ 2(1 - \frac{\text{iter}^{\frac{1}{\eta}}}{{T_{max}}^{\frac{1}{\eta}}}) & \text{si} \quad \text{iter} > \text{T}_{\text{max}}/2 \end{cases} \tag{B.18}$$

Une valeur élevée de η permet une concentration sur l'exploration de l'espace de recherche pendant une première phase, de $\text{iter} = 1$ à $\text{iter} = \text{T}_{\text{max}}/2$; et une concentration sur l'exploitation pendant une seconde phase, de $\text{iter} = \text{T}_{\text{max}}/2 + 1$ à $\text{iter} = \text{T}_{\text{max}}$.

B.2.2.4 Conclusion du chapitre 2

Nous avons présenté et expliqué en détails une nouvelle méthode d'optimisation bio-inspirée imaginée pour des problèms mixtes, *i.e.* des problèmes contenant des variables discrètes et continues. Cette méthode, basée sur le Grey Wolf Optimizer (GWO), a été baptisé le mixedGWO. sL'algorithme du mixedGWO se divise en deux parties : une version discrète du GWO nommée Improved Discrete GWO (IDGWO) et une version modifiée du GWO classique, à destination des problèmes continus, nommée Global Continuous GWO (GCGWO).
Les performances du mixedGWO et de sa version adaptive le amixedGWO ont été testées et décrites plus en détail dans le corps de ce manuscrit dans la section 2.2. Ces deux algorithmes proposés ont été comparés au GWO classique et au mGWO sur des fonctions de référence continues puis au MODGWO sur des fonctions de référence discrètes. Enfin, le mixedGWO et le amixedGWO ont été testés sur des fonctions de référence mixtes. À partir des résultats obtenus, nous pouvons conclure que les méthodes proposées sont suffisamment performantes sur des problèmes discrets et mixtes. Cependant, les résultats obtenus sur des problèmes continus n'étaient pas forcément meilleurs que ceux obtenus par les fonctions de l'état-de-l'art, sauf dans le cas de fonctions multi-modales à dimensions fixes où les méthodes proposées étaient plus performantes que celles de l'état de l'art.
Les résultats obtenus sont commentés plus en détails dans les sections 2.2 et 2.3 du corps de ce manuscrit.

Comme il l'est défini dans le théorème du "No Free Lunch" [118], il est impossible pour un seul et unique algorithme d'optimisation de résoudre tous les problèmes existants. Il est donc normal que le mixedGWO proposé ne soit pas capable de résoudre absolument tous les problèmes possibles avec de meilleures performances que les méthodes existantes. Cependant, il apparaît que le mixedGWO est le plus performant dans les cas continus quand il est utilisé pour des problèmes multi-modaux à dimensions fixes, ce qui est le cas des applications visées par cet algorithme : la classification d'image par SVM et le débruitage et démêlange simultané d'images multi-spectrales. Aux premiers abords, le mixedGWO proposé semble donc adapté pour ces deux applications.

B.2.3 Résumé du chapitre 3 : Applications : l'algorithme ISAM et le débruitage et démêlange simultané d'images Multi-Spectrales

B.2.3.1 L'algorithme de mesure d'audience IntuiSense Audience Metrics (ISAM)

La Mesure d'Audience a pour but de collecter des données permettant de créer le profil d'une clientèle, e.g. le nombre de clients, leur genre (homme/femme), leur âge ou encore leur temps de présence dans l'environnement étudié. Un tel outil peut-être utile pour des études marketing ou anthropologiques. Développé durant cette thèse pour la société IntuiSense, l'outil IntuiSense Audience Metrics (ISAM) est un outil de mesure d'audience basé sur de la vision par ordinateur et conçu pour fonctionner à l'aide d'une caméra embarqué au sommet d'un distributeur automatique.

Le but de l'outil ISAM doit être de pouvoir compter et caractériser les personnes apparaissant dans le champ de vision d'une caméra, à l'aide d'algorithmes de détection faciale et de reconnaissance faciale. Cependant, en se référant à l'état de l'art sur la détection et la reconnaissance faciale et les contraintes industrielles, les points suivants peuvent être prévus comme des verrous qui doivent être pris en compte lors du développement de l'outil ISAM :

- Le fond des images et l'illumination globale seront inconnus et non-contrôlables ;
- Les visages à détecter ne seront pas exactement faces à la caméra et ne regarderont que rarement dans sa direction ;
- Le hardware bas-coût (le processeur et la caméra) implique une limite quant aux capacités de calcul ;
- Des performances temps-réel sont visées.

À l'heure d'aujourd'hui, l'algorithme ISAM est capable de compter les gens dans le champ de vision de la caméra et de mesurer le temps passé par chacun d'entre eux dans le champ de vision. Le fonctionnement de l'algorithme se divise en deux parties : une partie détection et une partie suivi. L'algorithme fonctionne directement sur le flux de la caméra et fonctionne

sur chaque trame l'une après l'autre de manière indépendante. Les seules informations stockées entre chaque trame sont les positions des détections sur la trame précédente. Ce programme a été entièrement développé en C++ et utilise la librairie open-source de vision par ordinateur OpenCV [1].

L'algorigramme de la partie détection est disponible en Figure B.7 tandis que la Figure B.8 détaille la partie suivi. Dans le reste de cette sous-section, *previous detection* symbolisera les détections obtenues avant la trame courante t, et *current detection* symbolisera les détections obtenues sur la trame courante t. De plus, la position d'une détection ne doit pas être vue comme une simple coordonnée mais comme un rectangle complet encadrant le visage détecté.

• **La Partie Détection** a pour but de détecter les visages présents dans la trame courante t, à l'aide de la méthode de Viola-Jones. L'algorigramme de la partie détection est disponible en Figure B.7.
Afin de réduire la charge processeur, le détecteur de Viola-Jones ne sera appliqué sur la trame complète que une fois sur cinq. Sur les quatre trames restantes, le détecteur de Viola-Jones ne sera appliqué que sur des régions d'intérêt (ROI) choisies en fonction des positions des *previous detections*. Cette réduction de la zone de travail du détecteur de Viola-Jones a permis de réduire sa charge processeur par un facteur 2.

• **La Partie Suivi** est l'étape par laquelle toute trame doit passer après la être passée par la partie détection. Le but primaire de l'outil ISAM étant de compter les personnes dans son champ de vision, il doit être capable de différencier ces personnes entre elles afin de pouvoir repérer quand l'une d'elles quitte sont champ de vision ou si une nouvelle personne y apparaît.
Cette méthode, dont l'algorigramme est disponible en Figure B.8, est basée sur la méthode de suivi KLT, elle-même basée sur l'optical flow de Lucas-Kanade détaillé en sous-section B.2.1.2.

À chaque trame, l'algorithme va commencer par comparer les positions des *current detections* à celles des *previous detections*. Pour cela, l'aire d'intersection S entre les positions de deux détections est calculée. Si cette aire S est suffisamment élevée, $i.e.$ au moins 20% de la surface de la *previous detection*, alors la position de la *current detection* est considérée comme la nouvelle position de cette *previous detections*.
Une fois que toutes les correspondances ont été trouvées, l'algorithme va gérer les détections qui n'ont pas de correspondance, qu'elles soient des *current detections* ou des *previous detections*.
Dans le cas d'une *current detection*, l'algorithme la considèrera comme une nouvelles personne et l'ajoutera à l'ensemble des *previous detections* pour la frame suivante $t + 1$.

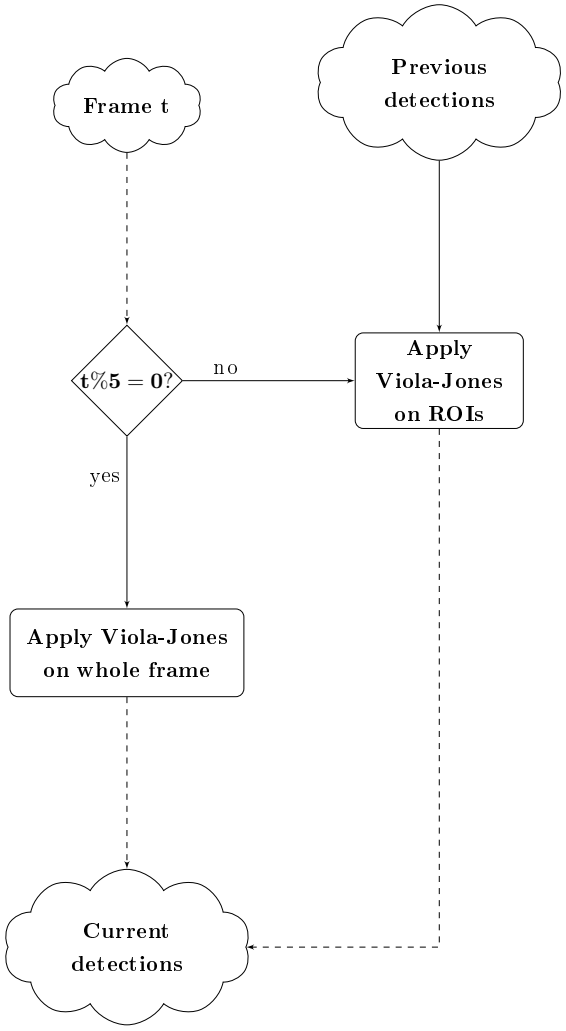


Figure B.7 — Partie détection de ISAM

Dans le cas d'une *previous detection*, l'Optical Flow de Lucas-Kanade (LK) [81] sera utilisé afin de trouver la nouvelle position de la détection. Si les points-clefs de la détection sont retrouvés par LK, alors sa position est mise à jour. En revanche, si les points-clefs ne sont pas retrouvés par LK, alors la personne est considérée comme ayant quitté le champ de vision et est effacée des *previous detections*.

• **Les Performances de l'outil ISAM** ont été mesurées à l'aide de l'indice Multi-Object Tracking Accuracy [14], qui est calculé à l'aide l'Eq. (B.19).

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{B.19}$$

avec m_t le nombre d'objets non détecté, fp_t le nombre de faux-positifs et mme_t le nombre de mauvaises correspondances tout au long du flux vidéo.

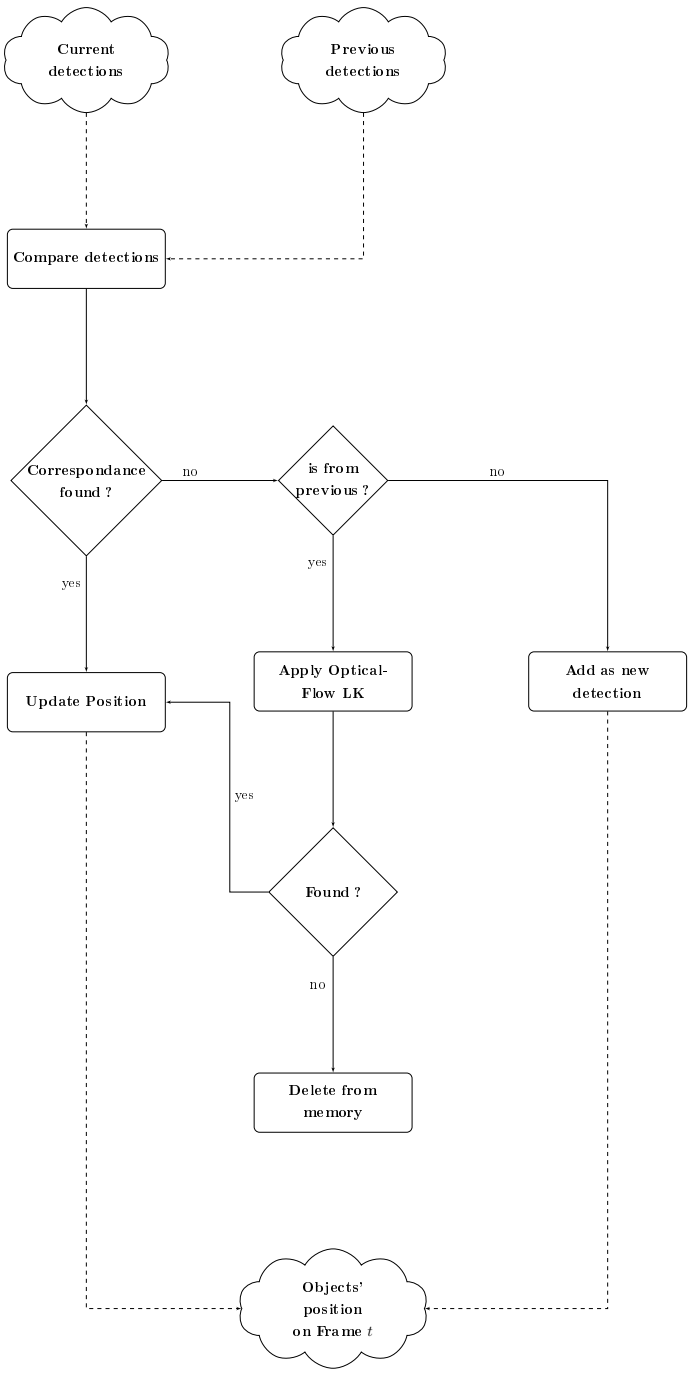


Figure B.8 — Partie suivi de ISAM

Les performances de l'outil ISAM ont été mesurées sur des vidéos issues de la ChokePoint database [119]. Ces vidéos sont intéressantes car le comportement des personnes filmées est assez similaire au comportement de personnes devant le distributeur automatique cible :

• elles bougent de manière quasi-naturelle en direction de la caméra, qui est positionnée

légèrement au-dessus de leur tête ;
• elles ne regardent pas directement en direction de la caméra, comme si elles n'étaient pas conscientes de sa présence.
Les résultats obtenus sont disponibles en Table B.1.

Video's name	People counted	Real number of people	m_t	fp_t	mme_t	$MOTA\ index$
$P1E\_S1\_C1$	24	25	1	0	0	96.00%
$P1E\_S2\_C1$	20	25	5	0	0	80.00%
$P1L\_S3\_C3$	21	24	3	0	1	83.33%
$P2E\_S3\_C11$	19	24	5	0	0	79.17%
$P2E\_S5\_C21$	19	24	5	0	0	79.17%
$P2L\_S2\_C21$	23	25	2	0	0	92.00%
$P2L\_S4\_C21$	21	25	4	0	0	84.00%
$P2L\_S5\_C21$	23	25	2	0	0	92.00%

Table B.1 — Résultats obtenus sur les vidéos de la ChokePoint database

L'algorithme ISAM obtient un MOTA moyen de 85.71% par vidéo testée et un MOTA total de 85.79%. De plus, quand l'algorithme ISAM est en fonctionnement sur le distributeur cible, qui possède un système d'explotatoin Windows 7 Embedded et un processeur Celeron @2GHz et 4GB de RAM, l'algorithme ne consomme que 30% des ressources processeur et fonctionne avec une cadence d'acquisition de 15fps. Quelques résultats qualitatifs supplémentaires sont visibles en Figure B.9.

Figure B.9 — Résultats qualitatifs de ISAM

B.2.3.2 Classification par genre optimisée par GWO Adaptif

Dans cette sous-section, seront présentés les résultats obtenus par l'application de la méthode d'optimisation GWO sur l'entraînement d'un SVM à kernel gaussien (RBF) pour de la reconnaissance de genre.
Les données à classifier pour la reconnaissance de genre étaient des caractéristiques LBPs de visages issus de la base de données FERET [103]. Les tests ont été effectués avec 200 images d'entraînement et 500 images de test pour chacun des deux genres (ce qui fait un total de 400 images d'entraînement et 1000 images de test). La **fitness** de ce problème, $i.e.$ la valeur à minimiser, sera le Taux de Mauvaises Reconnaissances (False Recognition Rate ou FRR) qui a été obtenu par $FRR = \frac{nb_{fr}}{nb\_testing\_images}$ avec nb_{fr} le nombre d'images de test mal reconnues, $i.e.$ reconnues comme un visage d'homme au lieu de femme et inversement.

Dans le cas d'un SVM à kernel gaussien, il y a 2 paramètres à déterminer : γ et C, qui sont tous deux des paramètres continus.
Comme expliqué précédemment, l'algorithme du GWO est dépendant d'un paramètre a qui décroit linéairement de 2 à 0. Ce paramètre permet au fonctionnement de l'agorithme d'être séparé en deux phases : la phase d'exploration (quand $a > 1$) et la phase d'exploitation (quand $a \leq 1$).
Suivant l'idée du mGWO proposé dans [95], un paramètre η permettant d'influer sur la répartition de ces 2 phases est inclus dans le modèle suivant :

$$a = 2(1 - \frac{\text{iter}^{\eta}}{{T_{max}}^{\eta}}) \tag{B.20}$$

Avec ce paramètre η, la quantité d'itérations passées durant la phase d'exploration $t_{exploration}$ devient :

$$t_{exploration} = \frac{T_{max}}{2^{\frac{1}{\eta}}} \tag{B.21}$$

En fonction de la valeur de η, il devient possible de choisir librement si le GWO doit privilégier l'exploration ($\eta > 1$), ou l'exploitation ($\eta < 1$), ou alors passer un nombre équivalent d'itérations sur chacune des deux phases ($\eta = 1$).

Pour exploiter au mieux ce paramètre η, nous proposons un GWO adaptatif (aGWO), dont l'algorigramme est disponible en Figure B.10. Le aGWO est simplement le GWO appliqué 3 fois mais à chaque fois avec une règle différente pour le calcul du paramètre a. De plus, entre chaque application, la position de la meilleur fitness trouvée, $i.e.$ la position du loup α, sera la position initiale d'un loup lors de l'application suivante, les autres loups étant positionnés aléatoirement dans l'espace de recherche. La première application peut

être considérée comme une recherche globale tandis que les deux applications suivantes sont des recherches affinées.

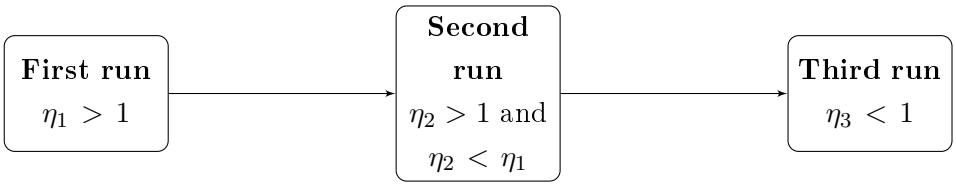


Figure B.10 — Algorigramme du aGWO

La méthode du aGWO a été testée sur le problème décrit précédemment : l'optimisation des paramètres d'entraînement d'un SVM à kernel RBF pour la reconnaissance de genre. Dans ce test, 20 agents (ou loups) ainsi que $\eta_1 = 8$, $\eta_2 = 2$ et $\eta_3 = 0.5$ ont été utilisés. De plus, le aGWO a été comparé au GWO classique, au mGWO et au PSO. Dans le cas du aGWO, chaque application duraient 5 itérations (15 en tout) tandis que les méthodes de l'état-de-l'art duraient 15 itérations chacune.

Method	Best (γ, C)	$100 - FRR$	time (sec.)
PSO	$(6.7e-07,\ 600)$	91.4%	1970
GWO	$(1.6e-07,\ 800)$	91.5%	1970
mGWO	$(5.0e-08,\ 83)$	**91.9**%	1970
aGWO	$(4.9e-08,\ 82)$	**91.9**%	968

Table B.2 — Estimations des paramètres optimaux et temps de calcul requis

Comme indiqué en Table B.2, le mGWO et le aGWO proposé étaient plus performants que le GWO classique et le PSO. De plus, le aGWO était deux fois plus rapide que le mGWO pour le même nombre d'itérations et la même précision.
Trois informations importantes peuvent être tirées de ces résultats. Tout d'abord, la méthode du GWO a montré de bons résultats sur le problème étudié et semble donc un bon point de départ pour une méthode d'optimisation bio-inspirée pour problèmes mixtes. Deuxièmement, l'inclusion du paramètre η semble une bonne idée. En effet, le mGWO et le aGWO ont obtenus le même FRR mais le aGWO est parvenu à être 2 fois plus rapide que les autres méthodes. Enfin, le problème étudié de l'optimisation des paramètres d'entraînement d'un SVM pour de la classification d'images semble être un problème **multi-modal à dimensions fixes**. En effet, le nombres de paramètres est fixé (*dimensions fixes*) et le problème semble avoir plusieurs minimum globaux (*multi-modal*), comme le montre les différents couples de résultats obtenus par mGWO et aGWO en Table B.2.

B.2.3.3 Application à l'imagerie Multi-Spectrale : débruitage et démêlange simultané d'images Multi-Spectrales

Dans cette sous-section, le mixedGWO proposé dans le Chapitre 2 est utilisé pour le débruitage et le démêlange simultané d'images multi-spectrales.

Description de l'application proposée

Les images multi-spectrales et hyper-spectrales sont aujourd'hui utilisées dans des applications de télédétection. Dans ce contexte, des capteurs dédiés ont été développés, tels que le capteur AVIRIS [46] ou le capteur ROSIS [40]. Une image multi-spectrale est obtenue en sélectionnant certaines bandes des images hyper-spectrales obtenues par ces capteurs aériens. La plupart des images multi-spectrales aériennes sont polluées par du bruit [73, 75] dû aux radiations solaires ou à de la diffusion atmosphérique [58], par exemple.
En imagerie multi-spectrale, le débruitage est souvent l'étape préliminaire avant de pouvoir effectuer des traitements de plus haut niveau tels que de la détection de cible [73] ou du démêlange de spectre [28]. Dans ce dernier cas, un des problèmes actuels est de trouver le meilleur compromis entre l'efficacité du processus de débruitage sur l'image étudiée et la précision de démêlange de spectres sur cette même image. Soit $\mathbf{y} \in \mathbb{R}^{I_3}$un spectre d'une image multi-spectrale $\mathcal{X}$. Pour ce spectre $\mathbf{y}$, un démêlange supervisé vise à estimer les contributions de la signature spectrale des matériaux présents dans l'image (appelés *endmembers*) pour une partie ou tous les pixels de l'image. Un démêlange supervisé signifie que les endmembers contenus dans l'image ont été estimés par un algorithme d'extraction d'endmembers tel que la Vertex Component Analysis [100].
Un modèle de mélange linéaire impliquant deux endmembers est tel que suit :

$$\mathbf{y}(\lambda) = (1-\lambda)\mathbf{s1} + \lambda\mathbf{s2} + \mathbf{n} \tag{B.22}$$

où λ est un coefficient de mélange, $\mathbf{s1}$ et $\mathbf{s2}$, dans $\mathbb{R}^{I_3}$, sont les deux endmembers et $\mathbf{n}$ est un vecteur de bruit.
Dans ce cas d'un modèle de mélange linéaire, le problème de démêlange, supervisé ou non, est simplement géré par une factorisation de matrice non-négative [102]. Nous voulons prouver les capacités de notre mixedGWO avec un modèle de mélange non-linéaire. Nous présumons que les spectres des endmembers sont connus, mais nous souhaitons déterminer, parmis les deux types de modèles non-linéaires possibles, quel modèles correspond le mieux aux données.
Le modèle considéré pour le mélange de spectres est noté $\mathbf{y}(f, \lambda_1, \lambda_2)$, où f est le modèle de mélange, valant f_0 ou f_1.

Le premier modèle est le modèle de mélange polynomial post-non-linéaire [5] :

$$\mathbf{y}(f_0^{mix}, \lambda_1, \lambda_2) = f_0^{mix}(\lambda_1, \lambda_2) = \mathbf{g^{mix}}(\mathbf{s}(\lambda_1), \lambda_2) + \mathbf{n} \tag{B.23}$$

où $\mathbf{s} = [s_1, \ldots, s_{I_3}]^T$ suit un modèle de mélange linéaire des deux endmembers $\mathbf{s1}$ et $\mathbf{s2}$:

$$\mathbf{s}(\lambda_1) = (1 - \lambda_1)\mathbf{s1} + \lambda_1 \mathbf{s2} \tag{B.24}$$

et $\mathbf{g^{mix}}$ est une non-linéarité polynomiale du second ordre :

$$\mathbf{g^{mix}} : [0; 1]^{I_3} \to \mathbb{R}^{I_3}$$

$$\mathbf{s} \mapsto \left[s_1 + \lambda_2 s_1^2, \ldots, s_{I_3} + \lambda_2 s_{I_3}^2\right]^T \tag{B.25}$$

Le second modèle est le modèle bilinéaire généralisé[41] :

$$\mathbf{y}(f_1^{mix}, \lambda_1, \lambda_2) = f_1^{mix}(\lambda_1, \lambda_2) = (1 - \lambda_1 - \lambda_2)\mathbf{s1} + \lambda_1 \mathbf{s2} + \lambda_2 \mathbf{s1}\mathbf{s2} + \mathbf{n} \tag{B.26}$$

Dans les Eqs. (B.23) et (B.26), λ_1 et λ_2 appartiennent à $[0; 1]$. Il est important de noter que les deux modèles se réduisent à un modèle linéaire si $\lambda_2 = 0$, ce qui signifie qu'ils peuvent être similaires, particulièrement dans le cas de λ_2 très petit.
En appliquant conjointement le débruitage et le démêlange supervisé sur une image multi-spectrale, les images pourraient être débruitées de manière à atteindre le meilleur résultat de démêlange possible. Pour cela, il est très important de choisir le critère à minimiser adéquat.

Critère proposé et critères d'évaluation

Avec la méthode proposée du amixedGWO, nous proposons de minimiser le critère suivant :

$$J^{LS}(K_1, K_2, K_3, f, \lambda_1, \lambda_2) = \frac{1}{I_1 I_2 I_3} ||\mathcal{X}_1 - \hat{\mathcal{X}}(K_1, K_2, K_3)||^2 + \frac{1}{I_3} ||\mathbf{y}(f^{mix}, \lambda_1, \lambda_2) - \hat{\mathbf{y}}(K_1, K_2, K_3)||^2 \tag{B.27}$$

où le tenseur $\mathcal{X}_1$ est une estimée grossière de $\mathcal{X}$, et $\hat{\mathcal{X}}(K_1, K_2, K_3)$ est l'estimée fourni par le Filtrage de Wiener Multi-directionnel (MWF) appliqué à $\mathcal{R}$ avec les valeurs de rang K_1, K_2, K_3. Le vecteur $\mathbf{y}(f, \lambda_1, \lambda_2)$ est le modèle spectral, où f, λ_1, et λ_2 doivent être estimés, et le vecteur $\hat{\mathbf{y}}(K_1, K_2, K_3)$ est un spectre, dont la position est connue, extrait de l'estimé du tenseur $\hat{\mathcal{X}}(K_1, K_2, K_3)$.

Les performances de la version adaptative du mixedGWO, le amixedGWO, ont été évaluées sur des images multi-spectrales issues de la base de données PaviaU scene [46]. Ces images possèdent 103 bandes spectrales de longueurs d'onde comprises entre $420nm$ et $850nm$.
Dans cette expérience, les valeurs de l'image attendue $\mathcal{X}$ et du mélange de spectre $\mathbf{y}$ ont été échelonnées entre 0 et 1. Le mélange de spectre a été généré avec la reflection de la végétation pour $\mathbf{s1}$ et la réfléction du sol pour $\mathbf{s2}$. Le spectre résultant a été positionné à un endroit spécifique de l'image. La connaissance de cette location était nécessaire afin de pouvoir extraire le spectre de l'image débruitée. Les paramètres de mélange attendus ont été fixés à : $f^{mix} = f_0^{mix} = 0$, $\lambda_1 = 0.15$, $\lambda_2 = 0.41$.
Les résultats de débruitage obtenus ont été évalués par leur valeur de SNR avec $SNR = 10\ log_{10}(\frac{||\mathcal{X}||^2}{||\mathcal{X}-\hat{\mathcal{X}}||^2})$.
Les images ont été bruitées artificiellement avec un bruit blanc réparti aléatoirement. La valeur de SNR originel est noté SNR_{in} et les valeurs de chaque expériences sont données en dB.
Les résultats de démêlange ont été évalués en terme d'erreur de reconstruction RE entre $\mathbf{y}$ et $\hat{\mathbf{y}}$ avec : $RE = \sqrt{\frac{1}{I_3}||\mathbf{y} - \hat{\mathbf{y}}||^2}$.
Enfin, les performances du amixedGWO ont été comparées aux méthodes de l'état de l'art : PSO [56], GWO [95], ABC [54], TSA [61], GA [45] et SA [77]. Ces méthodes étant originellement pour des espaces continus, un arrondi a été effectué sur les paramètres devant être des valeurs entières. Le nombre de particules de chacune de ces méthodes a été choisi de manière à ce que chacune nécessite exactement le même temps de calculs pour pouvoir effectuer $T_{max} = 20$ itérations.

Pour rappel, les paramètres recherchés étaient les paramètres de rang K_1, K_2 et K_3, le type de modèle de mélange f^{mix} et les coefficients de mélange λ_1 et λ_2. Ces paramètres et leurs espaces de recherches sont résumés en Table B.3.

Expected parameter index i \ Search space	H_i	$\mathbf{d}_i^{ind}$	$\mathbf{d}_i^{val}$
1,2,3	$\min(I_i, 8)$	$[1, 2, \ldots, I_i]^T$	$\left[1, \frac{I_i}{H_i}, 2\frac{I_i}{H_i}, \ldots, I_i\right]^T$
4	2	$[0, 1]^T$	$\left[f_0^{mix}, f_1^{mix}\right]^T$
5,6	•	•	$[0; 1]^T$

Table B.3 — Espace de recherche pour les méthodes d'optimisation. Le symbole • signifie "inutile"

Le nombre de valeurs H_i dans l'espace de recherche pour les rangs K_i, $i = 1, \ldots, 3$ est soit 8, soit la taille de l'image I_i, $i = 1, \ldots, 3$ si $I_i \leq 16$. Notons qu'il n'y a que deux valeurs possibles pour le modèle de mélange f^{mix}, et que les valeurs possibles pour les coefficients

de mélange λ_1 et λ_2 sont comprises entre 0 et 1.

Résultats expérimentaux

Les résultats fournis dans cette sous-section ont été obtenus sur des images extraites de la base de donnée PaviaU. Les résultats numériques ont été calculés sur une petite image tandis que les résultats visuels ont été calculés sur une grande image. La Figure B.11 montre les images utilisées sans bruit (noise-free), et la Figure B.12 montre les deux endmembers, et le spectre attendu, obtenus avec les paramètres $f^{mix} = f_1^{mix} = 0$, $\lambda_1 = 0.15$, $\lambda_2 = 0.41$.

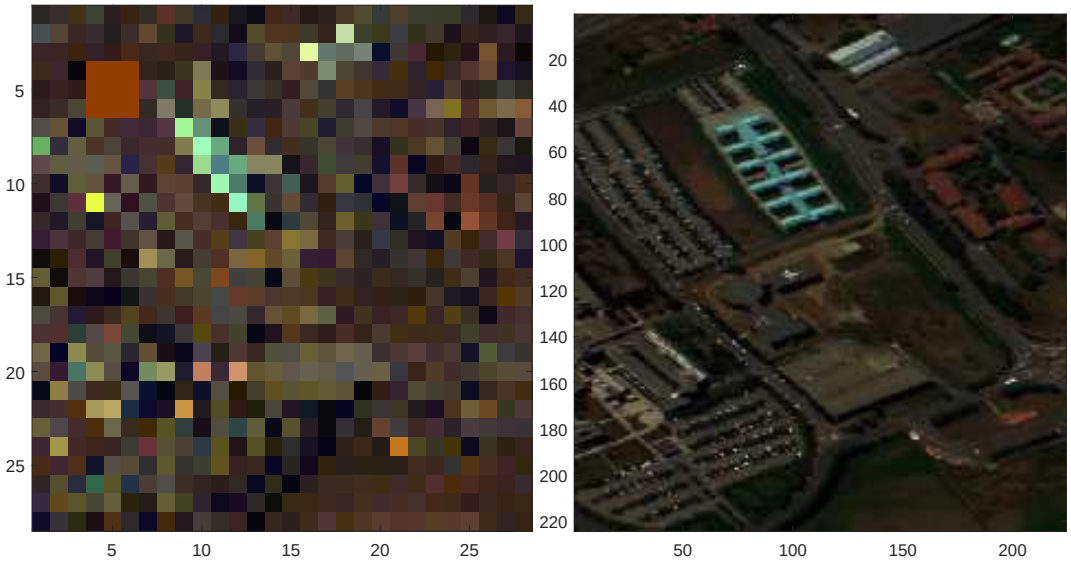


Figure B.11 — PaviaU $32 \times 32 \times 4$ et PaviaU $256 \times 256 \times 103$: images Noise-free

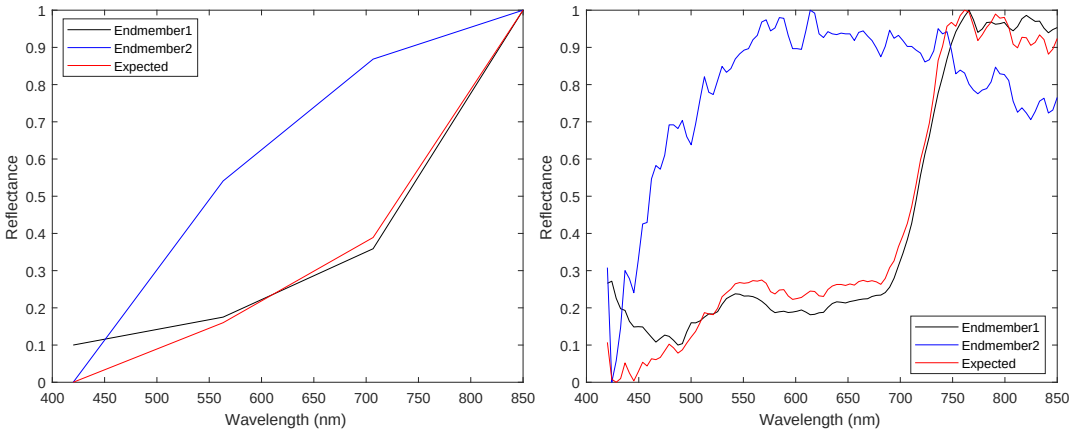
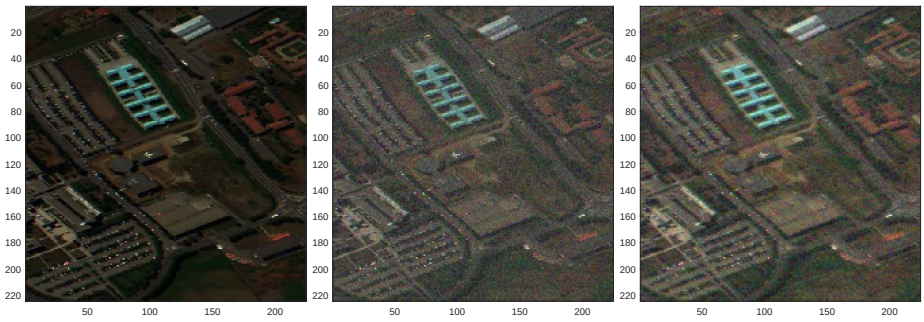


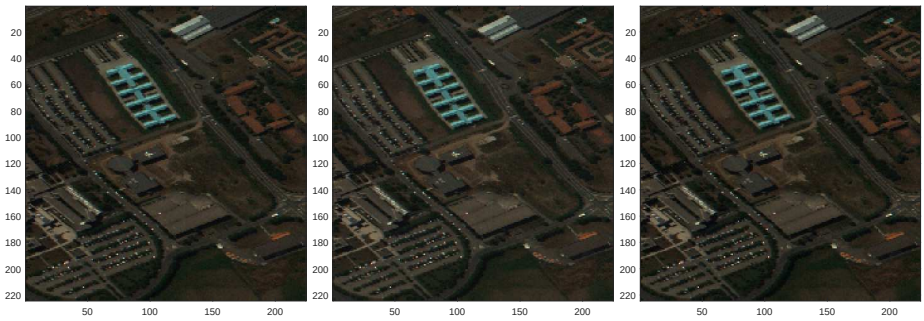
Figure B.12 — PaviaU $32 \times 32 \times 4$ and PaviaU $256 \times 256 \times 103$: endmembers **s1** (noir) et **s2** (bleu), et le spectre attendu **y** (rouge)

Dans la suite de cette sous-section, un cas réel a été étudié : l'image $\mathcal{R}$ a été bruitée avec une valeur de SNR finie, et l'image de référence a été obtenue avec un filtrage de Wiener appliqué dans le domaine de Fourier. Le débruitage a été appliqué bande par bande, *i.e.* pour chaque bande de l'image étudiée.
Nous avons appliqué notre méthode sur une grande image bruitée avec cinq valeurs de SNR différentes : $SNR_{in} = 0, 5, 10, 15$, and 20 dB. Pour $SNR_{in} = 10$, des résultats visuels sont donnés (voir Figures B.13, B.14 et B.15) ainsi que les spectres obtenus (voir Figures B.16 et B.17) et les courbes de convergences moyennes obtenues pour chaque méthode sur $M = 3$ runs (voir Figure B.18).



(a) (b) (c)

Figure B.13 — PaviaU $256 \times 256 \times 103$: (a) Noise-free, (b) bruité 10 dB, and (c) image de référence



(a) (b) (c)

Figure B.14 — PaviaU $256 \times 256 \times 103$. Images débruitées obtenue avec (a) amixedGWO, (b) PSO, (c) GWO

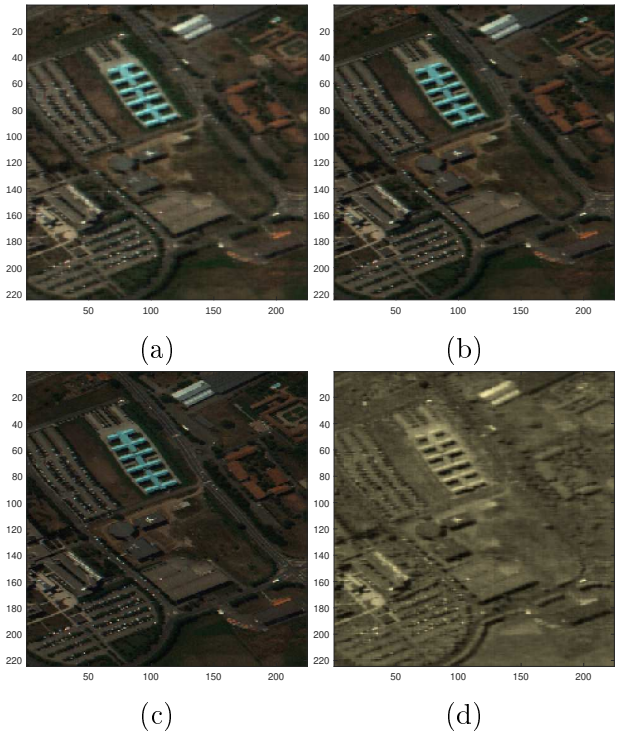


Figure B.15 — PaviaU $256 \times 256 \times 103$.Images débruitées obtenue avec (a) ABC, (b) TSA, (c) GA, (d) SA

Les Tables B.4, B.5 et B.6 présentent respectivement les valeurs de SNR de sortie, les valeurs des erreurs de reconstruction et les meilleures valeurs obtenues pour chaque valeur de SNR_{in}. Le rang général du amixedGWO était 1 sur tous ces points, bien que la valeur de SNR de sortie soit plus petite pour au moins une méthode comparée. Cela signifie que, dans l'ensemble, bien que le amixedGWO ne possède pas forcément la meilleure performance en terme de débruitage, il possède un bon comportement en terme d'exploitation, ce qui explique les valeurs d'erreur de reconstruction, qui étaient les plus petites pour tous les cas de figures exceptés $SNR_{in} = 0dB$ (voir Table B.5). La Table montre que les meilleures valeurs ont été obtenues par le amixedGWO excepté pour un SNR d'entrée de 5 et 15 dB. Dans ces cas, le amixedGWO possède un spectre débruité qui était le plus proche du spectre du modèle, mais au détriment du premier terme du critère de l'Eq. (B.27).

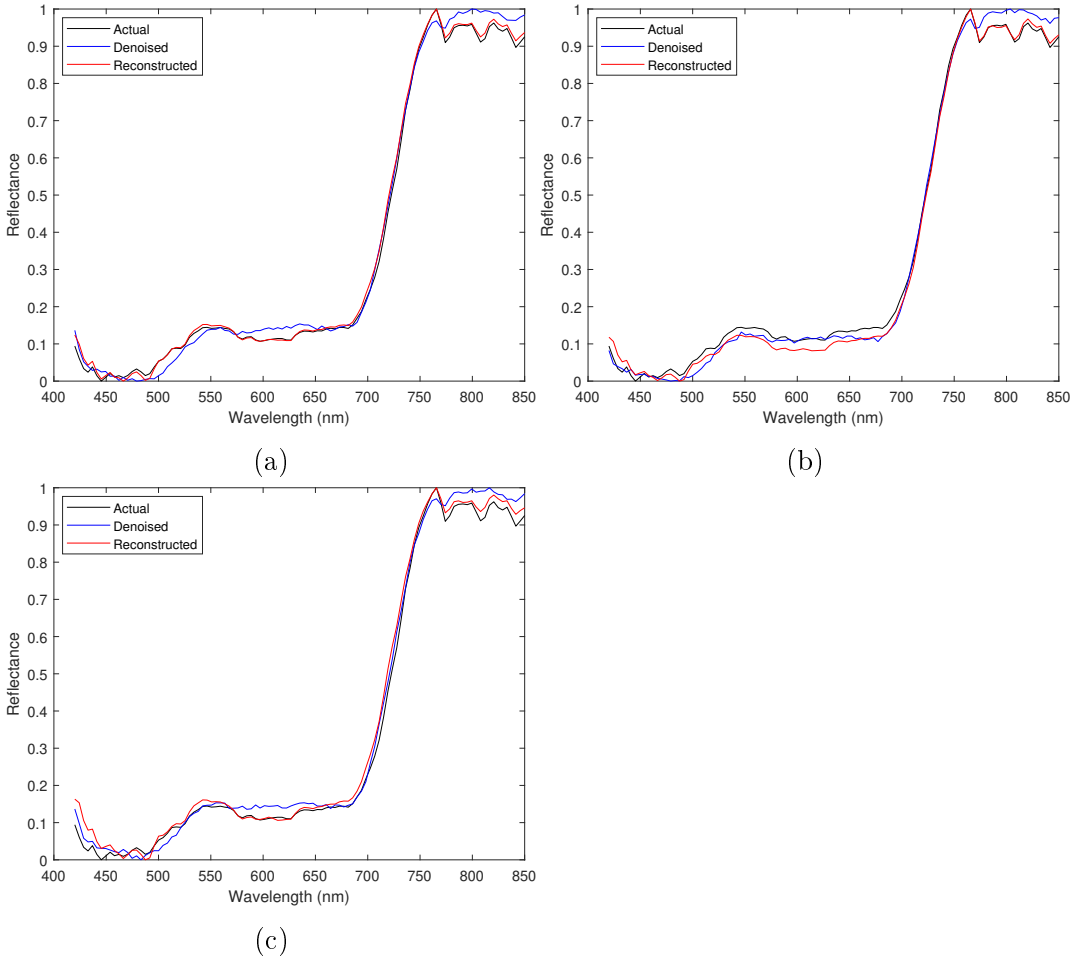


Figure B.16 — PaviaU $256 \times 256 \times 103$. Spectres réel, débruité, et reconstruit obtenus avec : (a) amixedGWO, (b) PSO, (c) GWO

B.2.3.4 Conclusion du chapitre 3

Dans la sous-section B.2.3.1, les principes de l'outil de Mesure d'Audience ISAM, conçu pour fonctionner sur un distributeur automatique à l'aide d'une caméra embarquée, ont été vu. Au jour d'aujourd'hui, cet algorithme est capable de détecter les personnes apparaissant dans le champ de vision de la caméra, de les suivre au cours de leurs mouvements et de déterminer qui, parmi les personnes présentes, a le plus de chance d'être l'utilisateur courant de la machine. L'algorithme ISAM est donc capable de récolter 4 types de données : le nombre de personnes détectées pendant une certaine période de temps, combien de temps chacune de ces personnes est restée dans le champ de vision de la caméra, si une personne a potentiellement été l'utilisateur du distributeur et, si oui, combien de temps a-t-elle été utilisateur. Cet algorithme fonctionne à un rythme temps-réel avec une cadence d'acquisition de 15fps et une précision de 85%. Cependant, la partie "reconnaissance de genre" manque toujours à l'outil ISAM.

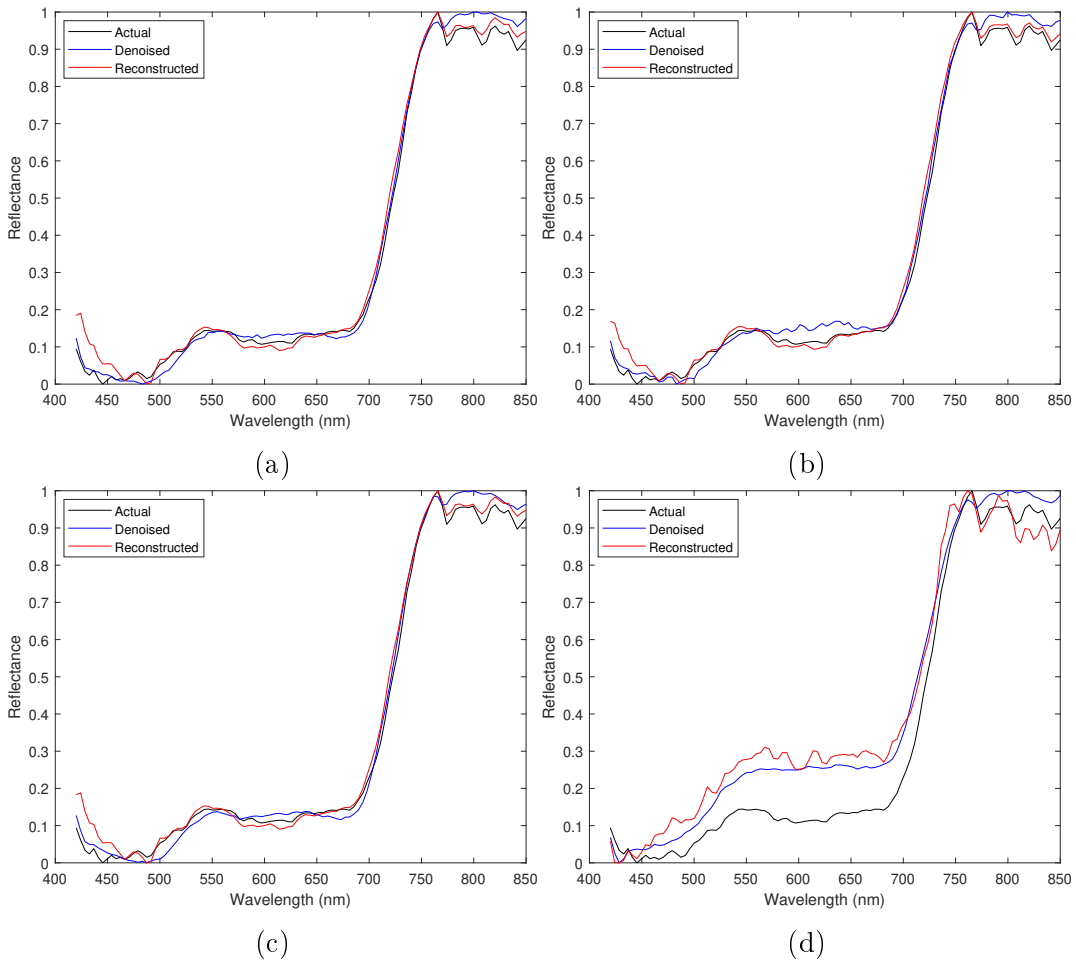


Figure B.17 — PaviaU $256 \times 256 \times 103$. Spectres réel, débruité, et reconstruit obtenus avec : (a) ABC, (b) TSA, (c) GA, (d) SA

Ensuite, en sous-section B.2.3.2, l'impact de l'application du GWO sur la classification par genre par Machine à Vecteurs de Support (SVM) a été étudié. Une version adaptative du GWO (aGWO) a également été apliquée, fonctionnant en 3 étapes. Une version global du GWO a d'abord été utilisée, avec $\eta > 1$ dans le but de se concentrer sur la phase d'exploration, afin de déterminer grossièrement les valeurs des paramètres d'entraînement. Cette estimation a ensuite été affinée en réduisant la valeur du paramètre η. Testé sur la base de données FERET, le aGWO proposé a atteint une précision de 91.9% tout en divisant le temps de calcul par 2 comparé au GWO classique ou au PSO. Cependant le aGWO n'est pas directement utilisable pour un SVM à kernel polynomial et n'est pas non plus capable de déterminer, de lui-même, quel kernel serait le plus adapté au problème de classification étudié. Pour cela, un algorithme d'optimisation pour problèmes mixtes, le mixedGWO, avait été proposé dans le Chapitre 2. Cependant, malgré la création du mixedGWO, la reconnaissance de genre n'a pas encore été ajouté à l'outil ISAM pour 2 raisons :

• La fonction d'entraînement d'un SVM est lourde et longue à calculer, même si sa vitesse a

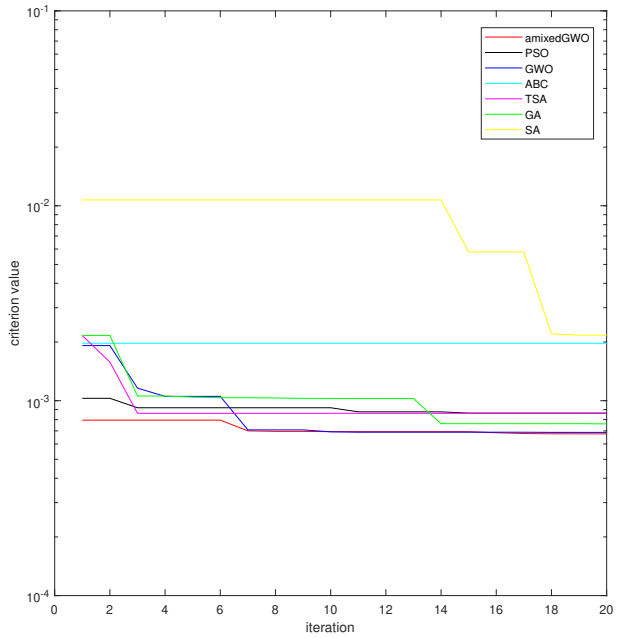


Figure B.18 — Courbes de convergence moyennes avec $SNR_{in} = 10$ dB et une référence Fourier Wiener $\mathcal{X}_1$

SNR_{in}		Ref.	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		3.99	**9.333**	7.659	7.745	5.560	8.429	8.45	6.42
	Rank	8	1	4	5	6	3	2	7
5 dB		8.79	12.180	11.477	**13.258**	7.878	11.893	7.97	8.34
	Rank	5	2	4	1	7	3	8	6
10 dB		13.24	**17.777**	14.159	17.108	7.719	15.310	8.88	2.49
	Rank	5	1	4	2	7	3	6	8
15 dB		16.61	17.804	18.587	**19.733**	13.388	15.815	15.22	14.43
	Rank	4	3	2	1	8	5	6	7
20 dB		18.26	**22.664**	20.869	22.653	17.417	18.504	15.37	13.21
	Rank	5	1	3	2	6	4	7	8
	Avg. Rank	5.4	1.6	3.4	2.2	6.8	3.6	5.8	7.2
	Overall Rank	5	1	3	2	7	4	6	8

Table B.4 — Results denoising SNR_{out} with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$

été significativement réduite avec le GWO adaptatif ;

• il n'existe pas de base de données d'images compatible avec la problématique, *i.e.* une base de données composée d'images de visages prises d'une caméra placée au dessus des gens, avec aucune personne ne regardant en direction de la caméra et avec une luminosité non uniforme et non maîtrisée.

Enfin, dans la sous-section B.2.3.3, la robustesse du amixedGWO proposé en Chapitre 2 a été testée sur une application réelle : le débruitage et démêlange simultanée d'images multi-

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		$2.81e-03$	$4.44e-03$	$3.51e-03$	$5.30e-03$	$\mathbf{2.80e-03}$	$4.00e-03$	$2.53e-02$
	Rank	2	5	3	6	1	4	7
5 dB		$\mathbf{2.34e-03}$	$4.41e-03$	$3.41e-03$	$5.45e-03$	$3.12e-03$	$5.63e-03$	$5.59e-03$
	Rank	1	4	3	5	2	7	6
10 dB		$\mathbf{2.47e-03}$	$3.12e-03$	$3.02e-03$	$3.68e-03$	$2.69e-03$	$2.79e-03$	$5.17e-03$
	Rank	1	5	4	6	2	3	7
15 dB		$\mathbf{1.89e-03}$	$2.82e-03$	$3.13e-03$	$4.40e-03$	$2.66e-03$	$2.83e-03$	$5.63e-03$
	Rank	1	3	5	6	2	3	7
20 dB		$\mathbf{1.92e-03}$	$2.82e-03$	$2.09e-03$	$2.75e-03$	$2.59e-03$	$4.26e-03$	$4.28e-03$
	Rank	1	5	2	4	3	6	7
	Avg. Rank	1.2	4.4	3.4	5.4	2	4.6	6.8
	Overall Rank	1	4	3	6	2	5	7

Table B.5 — Results spectrum reconstruction error RE with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		$\mathbf{2.094e-03}$	$2.894e-03$	$2.699e-03$	$2.264e-03$	$2.433e-03$	$6.14e-03$	$1.74e-02$
	Rank	1	5	4	2	3	6	7
5 dB		$1.372e-03$	$1.556e-03$	$\mathbf{1.166e-03}$	$3.540e-03$	$1.653e-03$	$3.81e-03$	$4.54e-03$
	Rank	2	3	1	5	4	6	7
10 dB		$\mathbf{6.642e-04}$	$1.044e-03$	$7.137e-04$	$2.046e-03$	$7.620e-04$	$2.26e-03$	$7.88e-03$
	Rank	1	4	2	5	3	6	7
15 dB		$5.032e-04$	$\mathbf{4.641e-04}$	$5.538e-04$	$9.461e-04$	$5.480e-04$	$1.08e-03$	$1.48e-03$
	Rank	2	1	4	5	3	6	7
20 dB		$\mathbf{2.929e-04}$	$3.418e-04$	$2.931e-04$	$4.792e-04$	$3.919e-04$	$7.71e-04$	$1.90e-03$
	Rank	1	3	2	5	4	6	7
	Avg. Rank	1.4	3.2	2.6	4.4	3.4	6	7
	Overall Rank	1	3	2	5	4	6	7

Table B.6 — Results Global best with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$

spectrales. Le but de cette application était de trouver le meilleur équilibre entre la qualité du débruitage et la qualité du démêlange de spectres. Dans cette application, certains paramètres étaient discrets, ici les valeurs de rang et le type du modèle de mélange, tandis que d'autres étaient des valeurs continues, ici les coefficients de mélange. La méthode proposée a été comparée aux méthodes de l'état-de-l'art PSO, GWO, ABC, TSA, GA et SA sur un cas réaliste. Le amixedGWO est parvenu à trouver les valeurs de rang attendus tout en obtenant l'erreur de reconstruction la plus faible. Cependant, une des limites de la stratégie proposée repose sur le choix du tenseur de référence : la capacité de convergence du amixedGWO était bonne, mais le critère était minimisé en fonction d'une référence qui est faillible. En effet, il est impossible de calculer une fonction à minimiser sans image de référence.

B.3 Conclusion Générale

Cette thèse possédait deux buts. Premièrement, nous avons développé un outil de mesure d'audience utilisant du traitement d'image et de la vision par ordinateur. Deuxièmement, nous proposons une méthode d'optimisation pour problèmes mixtes afin de répondre à des problèmes tels que la classification d'images par SVM et le débruitage et démêlage simultané d'images multi-spectrales.

Dans le Chapitre 1, un aperçu de l'état de l'art décrit les trois domaines nécessaires pour le développement d'un outil de Mesure d'Audience utilisant du traitement d'image et de la vision par ordinateur : les techniques de détection faciale, les techniques de suivi d'objet et la reconnaissance d'objet, avec une attention particulière sur les principes de la classification d'images, avec des outils tels que les Réseaux de Neurones Conventionnels (CNN) et les Machines à Vecteur de Support (SVM). De ce compte-rendu, le détecteur de Viola-Jones, l'Optical Flow de Lucas-Kanade et le SVM semblent être les méthodes les plus adaptées pour le problème de la Mesure d'Audience dans un environnement embarqué et temps-réel.
Toujours dans le Chapitre 1, une revue des méthodes d'optimisation bio-inspirées a été effectuée, avec une distinction claire entre les méthodes destinées à des problèmes continus et celles destinées à des problèmes discrets. Des algorithmes de l'état-de-l'art tels que le Particle Swarm Optimization (PSO), le Grey Wolf Optimizer (GWO), le Tree-Seed Algorithm (TSA), le Ant Colony Optimization (ACO) et le Multi-Objective Discrete GWO (MODGWO) ont également été détaillés. De cette revue, nous remarquons qu'il n'existe pas de méthode d'optimisation bio-inspirée à destination de problèmes mixtes. Pourtant, ce type de méthode serait utile pour l'optimisation d'un SVM ou pour une autre problématique de traitement d'images : la débruitage et démêlange simultané d'images multi-spectrales.

Dans le Chapitre 2, une variante de l'algorithme du GWO est présenté en détail. Cette méthode a été développée au cours de cette thèse dans le but de pouvoir résoudre des problèmes entièrement continus, des problèmes entièrement discrets et des problèmes mixtes. Cette nouvelle version du GWO, baptisée mixedGWO, a été comparée statistiquement à d'autres versions de l'algorithme du GWO (GWO, mGWO et MODGWO) sur des problèmes entièrement continus et des problèmes entièrement discrets. Le mixedGWO a également été comparé au PSO, au TSA et au GA sur les fonctions de référence CEC2014. Cependant, vu qu'aucune autre méthode de ce genre n'existe dans l'état-de-l'art, le mixedGWO n'a pas put être comparé à d'autres méthodes sur des fonctions de référence mixte, mais les résultats obtenus sur ces fonctions étaient satisfaisants en soi. De part les résultats obtenus sur ces fonctions de référence et en comparaison à l'état-de-l'art, le mixedGWO proposé semble adapté pour les deux applications cibles de cette thèse.

Le Chapitre 3 est divisé en deux parties distinctes :

• Dans la section 3.1, L'outil IntuiSense Audience Measurement (ISAM) est présenté et expliqué en détail. Cet algorithme, qui fonctionne à l'aide d'une caméra embarquée sur un distributeur automatique, est capable de détecter et suivre les personnes apparaissant dans le champ de vision de la caméra avec une vitesse d'acquisition de 15 FPS et une précision de 85%. De plus, l'outil ISAM est également capable de déterminer qui, parmi les personnes détectées, a la plus forte probabilité d'être l'utilisateur courant du distributeur. Quatre types de données peuvent donc être collectées par l'outil ISAM : le nombre de gens détectés au cours d'une période donnée, le temps passé par chaque personne détectée dans le champ de vision de la caméra, si une personne détectée a été l'utilisateur potentiel du distributeur et, si oui, combien de temps elle a été utilisateur potentiel. En section 3.2, l'impact de la méthode d'optimisation bio-inspirée Grey Wolf Optimizer (GWO) sur la classification par genre par Machine à Vecteur de Support (SVM) a été étudié. Dans cette étude, une version adaptative du GWO (aGWO) a également été testée. Cette version adaptative a permis à un SVM d'être entraîné avec la même précision que celui entraîné avec le GWO classique, tout en divisant le temps nécessaire par un facteur 2. Cette étude a permis de valider le choix du GWO comme base de la méthode d'optimisation à destination des problèmes mixtes, ainsi que sa compatibilité avec le problème de la classification d'images par SVM.
Dans le future, la caractérisation des personnes détectées pourra être ajoutée à l'outil ISAM en appliquant ce qui a été fait dans la section 3.2. Cependant, ce n'est pas encore possible en raison de deux éléments manquants : du temps et des données. En effet, l'entraînement d'un SVM demande beaucoup de temps (plusieurs jours) et cela doit être effectué $Q \times T_{max}$ fois, avec Q le nombre de loups et T_{max} le nombre maximum d'itérations. De plus, il n'existe pas de base de données d'entraînement adaptée à la problématique cible. En effet, la plupart des base de données existantes, telles que la base FERET utilisée en section 3.2, sont constituées d'images de visages humains prises dans des conditions idéales, $i.e.$ avec une luminosité uniforme et avec les personnes regardant directement en direction de la caméra. Dans le future, il serait donc pertinent de créer une base de données personnalisée, composée de visages de gens prises dans des conditions similaires à l'application, $i.e.$ avec une luminosité inconnue et incontrôlable, et des visages de gens pris légèrement du dessus, sans qu'ils ne regardent en direction de la caméra.
• En section 3.3, les performances du amixedGWO proposé ont été mesurées sur une application réelle : le débruitage et le démêlage simultané d'images multi-spectrales. Le but de cette application est de trouver le meilleur équilibre entre la qualité du débruitage d'image et la qualité du démêlange de spectre. Dans cette section, le amixedGWO proposé a été comparé aux méthodes de l'état de l'art PSO, GWO, ABC, GA et SA. Les performances de la méthode proposée étaient meilleures que celles des méthodes de l'état-de-l'art, pour un temps de calculs équivalent. Cependant, la stratégie proposé repose sur le choix d'un

tenseur de référence : la capacité de convergence du amixedGWO était bonne, mais le critère utilisé a été minimisé en fonction d'une référence qui n'était pas forcément fiable. Il pourrait donc être intéressant d'implémenter une version itérative de cette application, prenant en référence d'entrée l'image obtenue par le amixedGWO à chaque itération, dans le but de compenser le potentiel problème d'une mauvaise référence.

À propos des évolutions futures des méthodes mixedGWO et amixedGWO, il peut être intéressant d'étudier les performances des différentes versions de GWO si le nombre de paramètres à estimer change constamment, et, dans le cas de la version discrète du GWO, si le nombre d'éléments dans l'espace de recherche évolue. Un espace de recherche adaptatif pourrait être la solution. De plus, il semble intéressant d'étudier plus en détails les performances du amixedGWO proposé sur d'autres applications, et d'en créer une version multi-objectifs.
Enfin, à propos de l'optimisation bio-inspirée pour problèmes mixtes de manière globale, il pourrait être intéressant de créer une version discrète de certaines méthodes comparatives telles que le ABC ou le TSA. Cette version ne se limiterait pas aux problèmes binaires et serait inspirée du formalisme que nous proposons avec notre mixedGWO. Cependant, bien qu'elles soient très performantes, leur charge de calcul élevée risque d'être handicapant.

List of figures

List of tables

Bibliography

[1] *Opencv : Open-source computer vision library*. URL http://opencv.org/.

[2] T. Ahonen, A. Hadid et M. Pietikäinen, *Face recognition with local binary patterns*, dans *Computer Vision - ECCV 2004* (edité par T. Pajdla et J. Matas), (pp. 469–481), Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[3] K. R. T. Aires, A. M. Santana et A. A. D. Medeiros, *Optical flow using color information : Preliminary results*, dans *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, (pp. 1607–1611), ACM, New York, NY, USA, 2008.

[4] N. S. Altman, *An introduction to kernel and nearest-neighbor nonparametric regression*, The American Statistician, vol. 46, no. 3, pp. 175–185, 1992.

[5] Y. Altmann, A. Halimi, N. Dobigeon et J.-Y. Tourneret, *Supervised nonlinear spectral unmixing using a postnonlinear mixing model for hyperspectral imagery*, IEEE Transactions on Image Processing, vol. 21, no. 6, pp. 3017–3025, 2012.

[6] R. Archibald et G. Fann, *Feature selection and classification of hyperspectral images with support vector machines*, IEEE Transactions on Geoscience and Remote Sensing, vol. 4, no. 8, pp. 674–677, Oct. 2007.

[7] L. Auria et R. A. Moro, *Support vector machines (svm) as a technique for solvency analysis*, Available at SSRN : http ://ssrn.com/abstract=1424949 or http ://dx.doi.org/10.2139/ssrn.1424949 811, DIW Berlin Discussion Paper, 2008.

[8] I. B. Aydilek, *A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems*, Applied Soft Computing, vol. 66, pp. 232 – 249, 2018.

[9] A. Babalik, A. C. Cinar et M. S. Kiran, *A modification of tree-seed algorithm using deb's rules for constrained optimization*, Applied Soft Computing, vol. 63, pp. 289 – 305, 2018.

[10] A. Babalik, A. Ozkis, S. A. Uymaz et M. S. Kiran, *A multi-objective artificial algae algorithm*, Applied Soft Computing, vol. 68, pp. 377 – 395, 2018.

[11] H. Bay, T. Tuytelaars et L. V. Gool, *Surf : Speeded up robust features*, ECCV 2006. LNCS, (pp. 404–417), 2006.

[12] S. S. Beauchemin et J. L. Barron, *The computation of optical flow*, ACM Computing Surveys 27, (pp. 433–466), 1995.

[13] Z. Beheshti, S. M. Shamsuddin et S. Hasan, *Memetic binary particle swarm optimization for discrete optimization problems*, Information Sciences, vol. 299, pp. 58 – 84, 2015.

[14] K. Bernardin, A. Elbs et R. Stiefelhagen, *Multiple object tracking performance metrics and evaluation in a smart room environment*, dans *Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV*, vol. 90, (p. 91), Citeseer.

[15] A. K. Bhandari, V. K. Singh, A. Kumar et G. K. Singh, *Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using kapur's entropy*, Expert Systems with Applications, vol. 41, no. 7, pp. 3538 – 3560, 2014.

[16] S. Binitha et S. Sathya, *A survey of bio inspired optimization algorithms*, vol. 2, pp. 137–151, 01 2012.

[17] M. J. Black et P. Anandan, *A framework for the robust estimation of optical flow*, Fourth International Conference on Computer Vision, (pp. 231–236), 1993.

[18] M. J. Black et P. Anandan, *The robust estimation of multiple motions : Parametric and piecewise-smooth flow fields*, Computer Vision and Image Understanding 63, (pp. 75–104), 1996.

[19] N. Boughnim, J. Marot, C. Fossati et S. Bourennane, *Hand posture recognition using jointly optical flow and dimensionality reduction*, EURASIP Journal on Advances in Signal Processing, vol. 2013, no. 1, p. 167, Nov 2013.

[20] G. C. Cawley et N. L. C. Talbot, *On over-fitting in model selection and subsequent selection bias in performance evaluation*, Journal of Machine Learning Research, vol. 11, pp. 2079–2107, 2010.

[21] C.-F. Chao et M.-H. Horng, *The construction of support vector machine classifier using the firefly algorithm*, Intell. Neuroscience, vol. 2015, pp. 2 :2–2 :2, janvier 2015.

[22] C. Chen et S.-P. Chiang, *Detection of human faces in color images*, vol. 144, no. 6, pp. 384–388.

[23] K. Chen, T. Li et T. Cao, *Tribe-pso : A novel global optimization algorithm and its application in molecular docking*, Chemometrics and Intelligent Laboratory Systems, vol. 82, no. 1 - 2, pp. 248 – 259, 2006.

[24] G. G. Chrysos, E. Antonakos, P. Snape, A. Asthana et S. Zafeiriou, *A comprehensive performance evaluation of deformable face tracking "in-the-wild"*, International Journal of Computer Vision, vol. 126, no. 2, pp. 198–232, Apr 2018.

[25] A. C. Cinar et M. S. Kiran, *Similarity and logic gate-based tree-seed algorithms for binary optimization*, Computers and Industrial Engineering, vol. 115, pp. 631 – 646, 2018.

[26] M. Clerc, *Tribes-un exemple d'optimisation par essaim particulaire sans parametres de contrôle*, Optimisation par Essaim Particulaire (OEP 2003), Paris, France, vol. 64, 2003.

[27] Y. Cooren, M. Clerc et P. Siarry, *Performance evaluation of tribes, an adaptive particle swarm optimization algorithm*, Swarm Intelligence, vol. 3, no. 2, pp. 149–178, 2009.

[28] N. Dobigeon, Y. Altmann, N. Brun et S. Moussaoui, *Linear and nonlinear unmixing in hyperspectral imaging*, Data Handling in Science and Technology : Resolving Spectral Mixtures, (p. 41), 2016.

[29] M. Dorigo, V. Maniezzo et A. Colorni, *Ant system : optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 26, no. 1, pp. 29–41, Feb 1996.

[30] M. Dorigo et T. Stützle, *The ant colony optimization metaheuristic : Algorithms, applications, and advances*, dans *Handbook of metaheuristics*, (pp. 250–285), Springer, 2003.

[31] R. C. Eberhart, J. Kennedy *et al.*, *A new optimizer using particle swarm theory*, dans *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1, (pp. 39–43), New York, NY, 1995.

[32] E. Elhariri, N. El-Bendary, A. E. Hassanien et A. Abraham, *Grey wolf optimization for one-against-one multi-class support vector machines*, dans *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, (pp. 7–12), Nov 2015.

[33] E. Emary, H. M. Zawbaa et A. E. Hassanien, *Binary grey wolf optimization approaches for feature selection*, Neurocomputing, vol. 172, pp. 371–381, 2016.

[34] K. Etemad et R. Chellappa, *Discriminant analysis for recognition of human face images*, dans *Audio- and Video-based Biometric Person Authentication* (edité par J. Bigün, G. Chollet et G. Borgefors), (pp. 125–142), Springer Berlin Heidelberg, Berlin, Heidelberg, 1997.

[35] E. Fazl-Ersi, M. E. Mousa-Pasandi, R. Laganière et M. Awad, *Age and gender recognition using informative features of various types*, dans *2014 IEEE International Conference on Image Processing (ICIP)*, (pp. 5891–5895), Oct 2014.

[36] T. Gao, X. L. Feng, H. Lu et J. H. Zhai, *A novel face feature descriptor using adaptively weighted extended LBP pyramid*, vol. 124, no. 23, pp. 6286–6291.

[37] A. GEORGHIADES, P. BELHUMEUR et D. KRIEGMAN, *From few to many : Illumination cone models for face recognition under variable lighting and pose*, IEEE Trans. Pattern Anal. Mach. Intelligence, vol. 23, no. 6, pp. 643–660, 2001.

[38] A. GHERBOUDJ, A. LAYEB et S. CHIKHI, *Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm*, International Journal of Bio-Inspired Computation, vol. 4, no. 4, pp. 229–236, 2012.

[39] J. J. GIBSON, *The perception of the visual world.*, The perception of the visual world., Houghton Mifflin.

[40] R. O. GREEN, C. M. SARTURE, C. J. CHOVIT, J. A. FAUST, P. HAJEK et H. LAN NOVAK, *Aviris : A new approach to earth remote sensing*, Opt. Photon. News, vol. 6, no. 1, p. 30, Jan 1995.

[41] A. HALIMI, Y. ALTMANN, N. DOBIGEON et J.-Y. TOURNERET, *Nonlinear unmixing of hyperspectral images using a generalized bilinear model*, IEEE Transactions on Geoscience and Remote Sensing, vol. 49, no. 11, pp. 4153–4162, 2011.

[42] J. HAN et C. MORAGA, *The influence of the sigmoid function parameters on the speed of backpropagation learning*, dans *From Natural to Artificial Neural Computation* (edité par J. MIRA et F. SANDOVAL), (pp. 195–201), Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

[43] H.BAY, A.ESS, T. TUYTELAARS et L. V. GOOL, *Surf : Speeded-up robust features*, ComputerVision and Image Understanding (CVIU), vol. 110(3), pp. 346–359, 2008.

[44] A. A. HEIDARI et P. PAHLAVANI, *An efficient modified grey wolf optimizer with levy flight for optimization tasks*, Applied Soft Computing, vol. 60, no. Supplement C, pp. 115 – 134, 2017.

[45] J. H. HOLLAND, *Genetic algorithms*, Scientific american, vol. 267, no. 1, pp. 66–72, 1992.

[46] S. HOLZWARTH, A. MULLER, M. HABERMEYER, R. RICHTER, A. HAUSOLD, S. THIEMANN et P. STROBL, *Hysens-dais 7915/rosis imaging spectrometers at dlr*, dans *Proceedings of the 3rd EARSeL Workshop on Imaging Spectroscopy*, (pp. 3–14), 2003.

[47] B. HORN et B. SCHUNCK, *Determining optical flow*, Artificial Intelligence, vol. 17 (1-3), pp. 185–203, 1981.

[48] R.-L. HSU, M. ABDEL-MOTTALEB et A. K. JAIN, *Face detection in color images*, dans *Image Processing, 2001. Proceedings. 2001 International Conference on*, vol. 1, (pp. 1046–1049), IEEE, 2001.

[49] G. B. HUANG, M. RAMESH, T. BERG et E. LEARNED-MILLER, *Labeled faces in the wild : A database for studying face recognition in unconstrained environments*, Rapport technique 07-49, University of Massachusetts, Amherst, October 2007.

[50] Y. Huang et X. H. Zhuang, *Motion-partitioned adaptive block matching for video compression*, International Conference on Image Processing, vol. 1, p. 554, 1995.

[51] Z. Kalal, K. Mikolajczyk et J. Matas, *Forward-backward error : Automatic detection of tracking failures*, dans *Pattern Recognition (ICPR), 2010 20th International Conference on*, (pp. 2756–2759), IEEE, 2010.

[52] R. Kalman, *A new approach to linear filtering and prediciton problems*, vol. 82, pp. 35–45, 01 1960.

[53] V. K. Kamboj, *A novel hybrid pso–gwo approach for unit commitment problem*, Neural Computing and Applications, (pp. 1–13), 2015.

[54] D. Karaboga et B. Basturk, *A powerful and efficient algorithm for numerical function optimization : artificial bee colony (abc) algorithm*, Journal of Global Optimization, vol. 39, no. 3, pp. 459–471, Nov 2007.

[55] Z. T. Kardkovacs, Z. Paroczi, E. Varga, A. Siegler et P. Lucz, *Real-time traffic sign recognition system*, dans *Cognitive Infocommunications, 2011 2nd International Conference on*, (pp. 1–5), IEEE.

[56] J. Kennedy et R. Eberhart, *Particle swarm optimization*, dans *IEEE International Conference on Neural Networks*, (pp. 1942–1948), Perth, 1995.

[57] J. Kennedy et R. C. Eberhart, *A discrete binary version of the particle swarm algorithm*, dans *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 5, (pp. 4104–4108 vol.5), Oct 1997.

[58] J. Kerekes et J. Baum, *Full spectrum spectral imaging system analytical model*, IEEE Trans. on Geosc. and Remote Sensing, vol. 43, no. 3, pp. 571–580, March 2005.

[59] W. Ketchantang, S. Derrode, L. Martin et S. Bourennane, *Pearson-based mixture model for color object tracking*, Machine Vision and Applications, vol. 19, no. 5-6, pp. 457–466, 2008.

[60] W. Kim et J.-J. Lee, *Object tracking based on the modular active shape model*, Mechatronics, vol. 15, no. 3, pp. 371 – 402, 2005.

[61] M. S. Kiran, *Tsa : Tree-seed algorithm for continuous optimization*, Expert Systems with Applications, vol. 42, no. 19, pp. 6686 – 6698, 2015.

[62] M. Kohli et S. Arora, *Chaotic grey wolf optimization algorithm for constrained optimization problems*, Journal of Computational Design and Engineering, 2017.

[63] G. Komaki, E. Teymourian, V. Kayvanfar et Z. Booyavi, *Improved discrete cuckoo optimization algorithm for the three-stage assembly flowshop scheduling problem*, Computers and Industrial Engineering, vol. 105, pp. 158 – 173, 2017.

[64] A. Krizhevsky, I. Sutskever et G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, dans *Advances in Neural Information Processing Sys-*

tems 25 (edité par F. Pereira, C. J. C. Burges, L. Bottou et K. Q. Weinberger), (pp. 1097–1105), Curran Associates, Inc., 2012.

[65] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard et L. D. Jackel, *Handwritten digit recognition with a back-propagation network*, dans *Advances in Neural Information Processing Systems 2* (edité par D. S. Touretzky), (pp. 396–404), Morgan-Kaufmann.

[66] Y. A. LeCun, L. Bottou, G. B. Orr et K.-R. Müller, *Efficient BackProp*, (pp. 9–48), Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[67] J. Li, Y. Wang et T. Tan, *Video-based face recognition using earth mover's distance*, dans *Audio- and Video-Based Biometric Person Authentication* (edité par T. Kanade, A. Jain et N. K. Ratha), (pp. 229–238), Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[68] Z. Li et X. Huang, *Glowworm swarm optimization and its application to blind signal separation*, Mathematical Problems in Engineering, vol. Article ID 5481602, p. 8 pages, janvier 2016.

[69] H.-C. Lian et B.-L. Lu, *Multi-view gender classification using local binary patterns and support vector machines*, dans *Advances in Neural Networks-ISNN 2006*, (pp. 202–209), Springer.

[70] J. Liang, B. Y. Qu et P. Suganthan, *Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization*, 12 2013.

[71] S.-W. Lin, K.-C. Ying, S.-C. Chen et Z.-J. Lee, *Particle swarm optimization for parameter determination and feature selection of support vector machines*, Expert Systems with Applications, vol. 35, no. 4, pp. 1817–1824, novembre 2008.

[72] S.-W. Lin, K.-C. Ying, C.-Y. Lee et Z.-J. Lee, *An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection*, Applied Soft Computing, vol. 12, no. 10, pp. 3285 – 3290, 2012.

[73] T. Lin et S. Bourennane, *Hyperspectral image processing by jointly filtering wavelet component tensor*, IEEE Trans. Geosci. Remote Sens., vol. 51, no. 6, pp. 3529–3541, 2013.

[74] H. Liu, Y. Gao et C. Wang, *Gender identification in unconstrained scenarios using self-similarity of gradients features*, dans *2014 IEEE International Conference on Image Processing (ICIP)*, (pp. 5911–5915), Oct 2014.

[75] X. Liu, S. Bourennane et C. Fossati, *Nonwhite noise reduction in hyperspectral images.*, IEEE Geoscience and Remote Sensing Letters, vol. 9, no. 3, pp. 368–372, 2012.

[76] X. Liu et H. Fu, *PSO-Based Support Vector Machine with Cuckoo Search Technique for Clinical Disease Diagnoses*, The Scientific World Journal, vol. 2014, pp. 1–7, 2014.

[77] Z. Liu, Z. Liu, Z. Zhu, Y. Shen et J. Dong, *Simulated annealing for a multi-level nurse rostering problem in hemodialysis service*, Applied Soft Computing, vol. 64, pp. 148 – 160, 2018.

[78] D. G. Lowe, *Object recognition from local scale-invariant features*, dans *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2, (pp. 1150–1157), Ieee, 1999.

[79] C. Lu, L. Gao, X. Li et S. Xiao, *A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry*, Engineering Applications of Artificial Intelligence, vol. 57, pp. 61 – 79, 2017.

[80] C. Lu, S. Xiao, X. Li et L. Gao, *An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production*, Advances in Engineering Software, vol. 99, pp. 161 – 176, 2016.

[81] B. Lucas et T. Kanade, *An iterative image registration technique with an application to stereo vision*, Procs. of the 7th International Joint Conference on Artificial Intelligence, 1981.

[82] K.-F. Man, K. S. TANG et S. Kwong, *Genetic algorithms : concepts and designs*, Springer Science & Business Media, 2012.

[83] B. Martin, J. Marot et S. Bourennane, *Mixed grey wolf optimizer for the joint denoising and unmixing of multispectral images*, Applied Soft Computing. Under minor revision.

[84] B. Martin, J. Marot et S. Bourennane, *Real-time face detection and tracking for vending machine data management*, dans *26th Colloque GRETSI, Juan-Les-Pins, FRA, 5-8 septembre 2017*, GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2017.

[85] B. Martin, J. Marot et S. Bourennane, *Reconnaissance de genre optimisée par gwo adaptatif*, dans *26th Colloque GRETSI, Juan-Les-Pins, FRA, 5-8 septembre 2017*, GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2017.

[86] B. Martin, J. Marot et S. Bourennane, *Improved discrete grey wolf optimizer*, dans *2018 26th European Signal Processing Conference (EUSIPCO)*, Sept 2018.

[87] S. Medjahed, T. A. Saadi, A. Benyettou et M. Ouali, *Gray wolf optimizer for hyperspectral band selection*, Applied Soft Computing, vol. 40, pp. 178 – 186, 2016.

[88] F. Melgani et L. Bruzzone, *Classification of hyperspectral remote sensing images with support vector machines*, IEEE Transactions on Geoscience and Remote Sensing, vol. 42, no. 8, pp. 1778–1790, 2004.

[89] M. Minárová, D. Paternain, A. Jurio, J. Ruiz-Aranguren, Z. Takác et H. Bustince, *Modifying the gravitational search algorithm : A functional study*, Information Sciences, vol. 430-431, no. Supplement C, pp. 87 – 103, 2018.

[90] S. Mirjalili, *How effective is the grey wolf optimizer in training multi-layer perceptrons*, Applied Intelligence, vol. 43, no. 1, pp. 150–161, 2015.

[91] S. Mirjalili et A. H. Gandomi, *Chaotic gravitational constants for the gravitational search algorithm*, Applied Soft Computing, vol. 53, no. Supplement C, pp. 407 – 419, 2017.

[92] S. Mirjalili, S. M. Mirjalili et A. Lewis, *Grey wolf optimizer*, Advances in Engineering Software, vol. 69, pp. 46 – 61, 2014.

[93] S. Mirjalili, S. Saremi, S. M. Mirjalili et L. dos S. Coelho, *Multi-objective grey wolf optimizer : A novel algorithm for multi-criterion optimization*, Expert Systems with Applications, vol. 47, pp. 106 – 119, 2016.

[94] M. Mitchell, *An introduction to genetic algorithms*, MIT press, 1998.

[95] N. Mittal, U. Singh et B. Singh Sohi, *Modified grey wolf optimizer for global engineering optimization*, Applied Computational Intelligence and Soft Computing, vol. Article ID 7950348, p. 16 pages, 2016.

[96] J. M. Molero, E. M. Garzón, I. García et A. Plaza, *Analysis and optimizations of global and local versions of the rx algorithm for anomaly detection in hyperspectral data*, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 6, no. 2, pp. 801–814, 2013.

[97] D. Muti et S. Bourennane, *Multidimensional filtering based on a tensor approach*, Signal Proceesing Journal, Elsevier, vol. 85, no. 12, pp. 2338–2353, Dec. 2005.

[98] D. Muti, S. Bourennane et J. Marot, *Lower-rank tensor approximation and multiway filtering*, SIAM Journal on Matrix Analysis and Applications, vol. 30, no. 3, pp. 1172–1204, 2008.

[99] D. N., T. B. et S. C., *Human detection using oriented histograms of flow and appearance*, ECCV 2006. LNCS, vol. 3952, pp. 428–441, 2006.

[100] J. M. Nascimento et J. M. Dias, *Vertex component analysis : A fast algorithm to unmix hyperspectral data*, IEEE transactions on Geoscience and Remote Sensing, vol. 43, no. 4, pp. 898–910, 2005.

[101] M. Pal et G. M. Foody, *Feature selection for classification of hyperspectral data by svm*, IEEE Transactions on Geoscience and Remote Sensing, vol. 48, no. 5, pp. 2297 – 2307, 5 2010.

[102] V. P. Pauca, J. Piper et R. J. Plemmons, *Nonnegative matrix factorization for spectral data analysis*, Linear algebra and its applications, vol. 416, no. 1, pp. 29–47, 2006.

[103] P. J. PHILLIPS, H. MOON, S. A. RIZVI et P. J. RAUSS, *The feret evaluation methodology for face-recognition algorithms*, IEEE Transactions on pattern analysis and machine intelligence, vol. 22, no. 10, pp. 1090–1104, 2000.

[104] M. H. QAIS, H. M. HASANIEN et S. ALGHUWAINEM, *Augmented grey wolf optimizer for grid-connected pmsg-based wind energy conversion systems*, Applied Soft Computing, 2018.

[105] E. RASHEDI, H. NEZAMABADI-POUR et S. SARYAZDI, *Gsa : A gravitational search algorithm*, Information Sciences, vol. 179, no. 13, pp. 2232 – 2248, 2009. Special Section on High Order Fuzzy Sets.

[106] W. T. REEVES, *Particle systems, a technique for modeling a class of fuzzy objects*, ACM Transactions on Graphics (TOG), vol. 2, no. 2, pp. 91–108, 1983.

[107] A. SAHOO et S. CHANDRA, *Multi-objective grey wolf optimizer for improved cervix lesion classification*, Applied Soft Computing, vol. 52, no. Supplement C, pp. 64 – 80, 2017.

[108] F. S. SAMARIA et A. C. HARTER, *Parameterisation of a stochastic model for human face identification*, dans *Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*, (pp. 138–142), Dec 1994.

[109] V. SANTARCANGELO, G. M. FARINELLA et S. BATTIATO, *Gender recognition : Methods, datasets and results*, (pp. 1–6), IEEE.

[110] S. SHANTAIYA, K. VERMA et K. MEHTA, *Multiple object tracking using kalman filter and optical flow*, vol. 2, pp. 34–39, 01 2015.

[111] S. SURESH et S. LAL, *Multilevel thresholding based on chaotic darwinian particle swarm optimization for segmentation of satellite images*, Applied Soft Computing, vol. 55, pp. 503 – 522, 2017.

[112] C. TOMASI et T. KANADE, *Detection and tracking of point features*, Rapport technique, International Journal of Computer Vision, 1991.

[113] M. TURK et A. PENTLAND, *Eigenfaces for recognition*, J. Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, janvier 1991.

[114] V. N. VAPNIK, *The Nature of Statistical Learning Theory*, New York : Springer-Verlag, 1995.

[115] P. VIOLA et M. JONES, *Rapid object detection using a boosted cascade of simple features*, dans *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, (pp. I–511–I–518 vol.1), 2001.

[116] P. VIOLA et M. J. JONES, *Robust real-time face detection*, International Journal of Computer Vision, vol. 57, no. 2, pp. 137 – 154, 2004.

[117] C.-C. WANG et K.-C. WANG, *Hand posture recognition using adaboost with sift for human robot interaction*, dans *Recent progress in robotics : viable robotic service to human*, (pp. 317–329), Springer, 2007.

[118] D. H. WOLPERT et W. G. MACREADY, *No free lunch theorems for optimization*, IEEE transactions on evolutionary computation, vol. 1, no. 1, pp. 67–82, 1997.

[119] Y. WONG, S. CHEN, S. MAU, C. SANDERSON et B. C. LOVELL, *Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition*, dans *IEEE Biometrics Workshop, Computer Vision and Pattern Recognition (CVPR) Workshops*, (pp. 81–88), IEEE, June 2011.

[120] Y. XIANG, A. ALAHI et S. SAVARESE, *Learning to track : Online multi-object tracking by decision making*, dans *2015 IEEE International Conference on Computer Vision (ICCV)*, (pp. 4705–4713), Dec 2015.

[121] B. YANG, X. ZHANG, T. YU, H. SHU et Z. FANG, *Grouped grey wolf optimizer for maximum power point tracking of doubly-fed induction generator based wind turbine*, Energy Conversion and Management, vol. 133, pp. 427 – 443, 2017.

[122] X. S. YANG et S. DEB, *Cuckoo search via levy flights*, dans *2009 World Congress on Nature Biologically Inspired Computing (NaBIC)*, (pp. 210–214), Dec 2009.

[123] C. ZHANG et Z. ZHANG, *A survey of recent advances in face detection*, Technical report, Microsoft Research.

[124] X. ZHANG, Q. KANG, J. CHENG et X. WANG, *A novel hybrid algorithm based on biogeography-based optimization and grey wolf optimizer*, Applied Soft Computing, vol. 67, pp. 197 – 214, 2018.

[125] X. ZHANG, G.-S. XIA, Q. LU, W. SHEN et L. ZHANG, *Visual object tracking by correlation filters and online learning*, ISPRS Journal of Photogrammetry and Remote Sensing, vol. 140, pp. 77 – 89, 2018.

[126] Y. ZHANG et P. ZHANG, *Machine training and parameter settings with social emotional optimization algorithm for support vector machine*, Pattern Recognition Letters, vol. 54, pp. 36 – 42, 2015.

[127] Y. ZHONG, A. MA, Y. SOON ONG, Z. ZHU et L. ZHANG, *Computational intelligence in optical remote sensing image processing*, Applied Soft Computing, vol. 64, pp. 75 – 93, 2018.

[128] A. ZIDI, J. MAROT, S. BOURENNANE et K. SPINNLER, *Bio-inspired optimization algorithms for automatic estimation of multiple subspace dimensions in a tensor-wavelet denoising algorithm*, JRST, 2016.