Applied Soft Computing 13 (2013) 3130–3148

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Fast intelligent watermarking of heterogeneous image streams through mixture modeling of PSO populations

Eduardo Vellasques*, Robert Sabourin, Eric Granger

École de Technologie Supérieure, Université du Québec, Montreal, Canada

ARTICLE INFO

Article history:
Received 31 December 2011
Received in revised form 2 May 2012
Accepted 14 August 2012
Available online 8 September 2012

Keywords:
Dynamic particle swarm optimization (DPSO)
Memory-based optimization
Digital watermarking
Intelligent watermarking
Gaussian mixture modeling (GMM)

ABSTRACT

In intelligent watermarking (IW), evolutionary computing (EC) is employed in order to automatically set the embedding parameters of digital watermarking systems for each image. However, the computational complexity of EC techniques makes IW unfeasible for large scale applications involving heterogeneous images. In this paper, we propose a dynamic particle swarm optimization (DPSO) technique which relies on a memory of Gaussian mixture models (GMMs) of solutions in the optimization space. This technique is employed in the optimization of embedding parameters of a multi-level (robust/fragile) bi-tonal watermarking system in high data rate applications. A compact density representation of previously-found DPSO solutions is created through GMM in the optimization space, and stored in memory. Solutions are re-sampled from this memory, re-evaluated for new images and have their distribution of fitness values compared with that stored in the memory. When the distributions are similar, memory solutions are employed in a straightforward manner, avoiding costly re-optimization operations. A specialized memory management mechanism allows to maintain and adapt GMM distributions over time, as the image stream changes. This memory of GMMs allows an accurate representation of the topology of a stream of optimization problems. Consequently, new cases of optimization can be matched against previous cases more precisely (when compared with a memory of static solutions), leading to considerable decrease in computational burden. Simulation results on heterogeneous streams of images indicate that compared to full re-optimization for each document image, the proposed approach allows to decrease the computational requirement linked to EC by up to 97.7% with little impact on the accuracy for detecting watermarks. Comparable results were obtained for homogeneous streams of document images.

1. Introduction

Enforcing the security of digital images has become a critical issue over the last decade. Advances in communications and computing allow easy transmission and manipulation of digital images which limits the efficiency of traditional security methods like cryptography since when the image has been decrypted there is no mean of enforcing its integrity and authenticity. Digital watermarking [1] allows an additional level of security by embedding image related information in a covert manner through a manipulation of pixel values. The embedding process is subject to a trade-off between the robustness against intentional and unintentional image processing operations (attacks) and the imperceptibility of the embedded watermark (image quality) [2]. The embedding of multiple watermarks with different levels of robustness [3] allows enforcing image authenticity and integrity at the same time, which is a crucial issue in applications involving document images.

The trade-off between robustness and quality can be adjusted through manipulation of embedding parameters. In intelligent watermarking (IW), evolutionary computing (EC) algorithms such as genetic algorithms (GA) [4], particle swarm optimization (PSO) [5] are employed in order to automatically find the embedding parameters that result in an optimal trade-off for a given image [6]. A population of candidate embedding parameters is evolved through time using a combination of robustness and quality metrics as objective function [7–20]. But this process is not feasible in a large scale scenario due to the high computational cost of EC [10].

In [21,22], the IW of **homogeneous** streams of bi-tonal document images was formulated as a special case of *dynamic* optimization problem (DOP[1]), where a stream of images corresponds to a stream of optimization problems (states) and some states may occur repeatedly [24]. Then, selected solutions found

* Corresponding author. Tel.: +1 514 396 8800x7675.
E-mail addresses: evellasques@livia.etsmtl.ca (E. Vellasques), robert.sabourin@etsmtl.ca (R. Sabourin), eric.granger@etsmtl.ca (E. Granger).

[1] In a DOP the optima change over time and might be followed by a period of stasis [23].

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.08.040

at the end of optimization were stored in an archive and recalled for similar problems. One limitation with such approach is that it assumes an homogeneous stream of document images, which is not always the case with real world applications. Selected solutions do provide an accurate representation of such stream of optimization problems, which makes it unfit for applications involving **heterogeneous** streams of document images. In this paper, a novel IW technique is proposed for the fast intelligent watermarking of **heterogeneous** streams of document images. A memory consisting of Gaussian mixture models (GMMs) of all solutions in the optimization space (optimization history) plus their respective global bests is incrementally built, and for every image, solutions are sampled from this memory and re-evaluated for the new image. If both distributions of fitness values are similar, memory solutions are employed directly. Otherwise, the respective optimization problem is considered to be novel and a costlier DPSO operation is performed. After that, the memory is updated with the GMM of the optimization history of the new problem. Such approach results in a more precise representation of the topology of the stream of optimization problems. For this reason, it allows better recalling previously seen problems and is preferred in a scenario involving heterogeneous streams of document images. The research problem addressed in this paper is how to use knowledge of past optimization problems in order to obtain a precise representation of a stream of optimization problems. The hypothesis on which this approach is based is that through time, density estimates of solutions found during optimization provide a compact but yet precise representation of the optimization problems presented to the intelligent watermarking system up to that point. The two main research questions addressed in this paper are (1) how to build a compact representation of a stream of optimization problems in an incremental manner and (2) how to employ such representation in order to detect new cases of optimization.

The idea of using density estimates of solutions in the optimization space is not new. Estimation of Density Algorithms (EDA) [25] rely on iteratively estimating density of **genotypic** data of **high evaluating** solutions. Differently than in EDA, our approach relies on both, **genotypic** and **phenotypic** data of **all** solutions from the optimization history in order to build a more general representation of the optimization problem. Moreover, in our approach the model is employed in order to match new problems with previously seen problems and to provide ready-to-use solutions. The research presented in this paper follows the research presented in a previous paper [22]. However, in the previous research we formulated IW of homogeneous streams of document images as the optimization of a stream of recurring problems and proposed a DPSO technique based on a memory of static solution. It was observed that such memory lacked precision to tackle IW of heterogeneous streams of document images which led to a degradation in computational burden of that approach in such scenario. In this paper, we focused on obtaining a precise representation of the underlying optimization problems in order to allow a better match between new and previous cases of optimization. Memory precision is an important element in our initial formulation of intelligent watermarking and has been neglected in our first paper. Therefore, this strategy of incrementally building a compact yet precise model of a stream of optimization problems is the main contribution of this research and is to the best of our knowledge, novel.

The proposed approach is evaluated in the optimization of the embedding parameters of a multi-level (robust/fragile) bi-tonal watermarking system [3,26] for both heterogeneous and homogeneous image streams, with and without cropping and salt & pepper (which are removal attacks [27]). The standard approach in the bi-tonal watermarking literature is to test watermark robustness against tampering attacks like cropping, manual removal/modification of connected components like characters

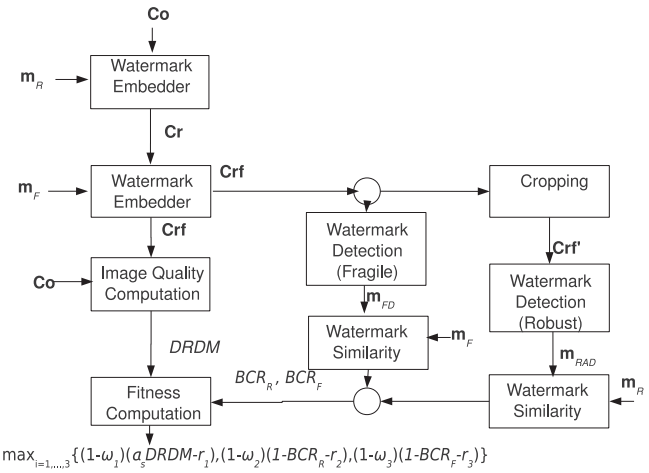

Fig. 1. Fitness evaluation module.

[28–30,26,31,3,32]. Other removal attacks like Stirmark [33], image enhancement, JPEG compression, noise filtering either require gray-scale images or knowledge about the features present in the bi-tonal image [34] and were not considered in our research. Resistance against geometric attacks can be easily tackled with the use of reference marks [3] and is also outside the scope of this paper. Experimental results demonstrate that the proposed approach has a good memorization capability but at the same time, is flexible enough to adapt to variations in the stream of optimization problems.

Our optimization problem formulation of intelligent watermarking is presented in Section 2. A brief literature review of related techniques is presented in Section 3. The new approach proposed in this paper, based on Gaussian mixture modeling for density estimation of solutions in the optimization space, and on adaptive memory management mechanisms is described in Section 4. Finally, Section 5 provides simulation results and discussion.

2. Optimization problem formulation of intelligent watermarking

The problem addressed in this article is the optimization of embedding parameters of a bi-tonal watermarking system, aimed at a high throughput adaptive watermarking of heterogeneous streams of document images. In this formulation, a stream of images is seen as a stream of optimization problems. Two possible actions can occur when an image from that stream is to be watermarked: (1) an existing solution (set of embedding parameters) is recalled from the memory; (2) optimization is triggered in order to find a new solution. If optimization is triggered, a population (swarm) of candidate solutions (particles) is evolved through several generations using Dynamic PSO (DPSO). At each generation, each solution has its fitness evaluated in a given watermarking task. The fitness function of the proposed technique is depicted in Fig. 1.

The PSO algorithm employed on full optimization is the same described in [22]. The fitness function was slightly modified. Firstly, the Conventional Weighted Aggregation (CWA) mechanism was replaced by Chebyshev Weighted Aggregation which is more robust to anomalies in the trade-off between the various fitness functions in a multi-objective optimization problem. In the Chebyshev approach, fitness values are aggregated according to their distances from reference points, under which the values of these fitnesses are considered good [35]. Secondly, the robustness of the fragile watermark was added to the aggregated function in order to minimize interference of the robust watermark as observed in [22]. Thirdly, BCR^{-1} was replaced by $1 - BCR$. Therefore, the fitness function will

Table 1
Range of embedding parameter values considered for PSO algorithm in this paper.

Embedding parameter	Particle encoding
Block size (B): $\{2, 3, 4, \ldots, B_B\}$	$x_{i,1} : \{1, 3, 4, \ldots, B_B - 1\}$
Difference between Q for the robust (Q_R) and fragile (Q_F) watermarks (ΔQ): $\{2, 4, 6, \ldots, 150\}$	$x_{i,2} : \{1, 2, \ldots, 75\}$
SNDM window size (W): $\{3, 5, 7, 9\}$	$x_{i,3} : \{1, 2, 3, 4\}$
Shuffling seed index (S): $\{0, 1, 2, \ldots, 15\}$	$x_{i,4} : \{0, 1, 2, \ldots, 15\}$

be defined as:

$$F(\mathbf{x}) = \max_{i=1,\ldots,3} \{(1-\omega_1)(\alpha_s DRDM - r_1), (1-\omega_2)(1-BCR_R - r_2), (1-\omega_3)(1-BCR_F - r_3)\} \quad (1)$$

where α_s is the scaling factor of the quality measurement *DRDM* (Distance Reciprocal Distortion Measure [36]), BCR_R (Bit Correct Ratio [7,17] between embedded and detected watermark) is the robustness measurement of the robust watermark, BCR_F is the robustness measurement of the fragile watermark, ω_i is the weight of the ith objective with $\omega_i = 1/3$, $\forall_i$, r_i is the reference point of objective i. The fitness function is depicted in Fig. 1 where **Co** is the cover image, $\mathbf{m}_R$ and $\mathbf{m}_F$ are the robust and fragile watermarks, respectively, **Cr** is the robust watermarked image, **Crf** is the image that has been watermarked with both, the robust and the fragile watermarks (multi-level watermarked image), **Crf′** is the multi-level watermarked/attacked image, $\mathbf{m}_{RAD}$ is the robust watermark that has been detected from the multi-level watermarked/attacked image, $\mathbf{m}_{FD}$ is the fragile watermark that has been detected from the multi-level watermarked image.

The bi-tonal method of Wu and Liu [3] (relying on the pixel flippability analysis technique of Muharemagic [26]) is employed as the baseline watermarking method in exactly the same manner as in [22]. This method allows the embedding of multiple watermarks in a same image with different levels of robustness where robustness is defined by a quantization step size parameter Q.

The particle encoding employed in this system can be seen in Table 1. Basically, the block size has lower bound of 2×2 and upper bound of $B_B \times B_B$ with $B_B = \max_B\{B^2 \times \max\{|\mathbf{m}_R|, |\mathbf{m}_F|\} \leq |\mathbf{Co}|\}$ pixels where B is the block width in pixels, $|\mathbf{m}_R|$, $|\mathbf{m}_F|$ and $|\mathbf{Co}|$ is the size of the robust watermark, fragile watermark and cover images, respectively. The remaining bounds, ΔQ, SNDM (Structural Neighborhood Distortion Measure [26]) window size and number of shuffling seeds were defined based on the literature [26]. Finally, $x_{i,j}$ is the jth parameter encoded in the ith particle.

3. Related work

3.1. Dynamic particle swarm optimization (DPSO) of recurrent problems

Particle swarm optimization (PSO) [5] relies on heuristics found on bird flocks and fish schooling in order to tackle the optimization of non-linear, noisy optimization problems. The underlying principle is that a population (swarm) of candidate solutions (particles) can tackle such type of optimization problem in a parallel manner with each particle performing its search guided by the best position found by itself and its best neighbor. The canonical PSO cannot tackle dynamic optimization when the optima changes due to issues like outdated memory, lack of a change detection mechanism and diversity loss [37,38]. One possible strategy to tackle this problem is to restart optimization whenever a change has been identified. However, the computational burden of such approach is prohibitive, specially in practical applications. But numerous practical applications, including intelligent watermarking of stream of document images, involve recurrent problems, that reappear through time, in a cyclical manner. It has been demonstrated in the literature that the best strategy to tackle such time of problem is to keep a memory of previous solutions to be recalled for future similar problems, in an approach named memory-based optimization [24]. It has also been demonstrated that depending on the level of similarity between previous and new problems, it is possible to employ the solutions directly in the new problem, without any need of re-optimization [22].

According to Yang and Yao [24], solutions can be stored in a memory either by an implicit or an explicit memory mechanism. In an implicit memory mechanism, redundant genotype representation (i.e. diploidy-based GA) is employed in order to preserve knowledge about the environment for future similar problems. In an explicit mechanism, precise representation of solutions is employed but an extra storage space is necessary to preserve these solutions for future similar problems. There are three major concerns in memory-based optimization systems that rely on an explicit mechanism: (1) what to store in the memory; (2) how to organize and update the memory; (3) how to retrieve solutions from the memory. Regarding what to store, there are two known approaches: direct memory scheme, where good solutions are stored and reused when the environment changes; associative memory scheme, where what is stored is information that associates good solutions with their environment (in most cases, a density estimate of the parameter space). The memory organization, by its way, can be based on a local mechanism (individual oriented) or on a global mechanism (population oriented). Regarding the memory update, since most real world applications assume limited memory, the basic approach is to select a solution stored in the memory to be removed (a review of removal strategies can be found in [39]) or updated by the newest solution.

An external memory requires an appropriate memory retrieval mechanism. There are two main memory retrieval strategies [40] – memory-based resetting and memory-based immigrants. In the first strategy, when a change is detected (change detection is usually achieved by re-evaluating memory solutions on the new environment), all solutions in the memory are re-evaluated and the best one is chosen as the new global best solution if it is better than the old one. In the memory-based immigrants strategy, all the solutions in the memory are re-evaluated and injected into the population.

The approach proposed in this paper is based on an associative memory. Since it has been already demonstrated in the literature that an associative memory allows associating previous solutions with corresponding new cases of optimization, we evolve this idea a little further and employ the associative memory as a mean of modeling an stream of optimization problems. That is, more than associating solutions with new cases of optimization, the proposed approach allows classifying new cases of optimization based on previously learned problems.

3.2. Pattern classification

Pattern classification [41] deals with assigning category labels to new patterns based on previously learned pattern/label assignments. Novelty detection (or one-class classification [42]) comprises the identification of patterns that were not available during a training (learning) phase. The main objective of a novelty detection system is to detect whether a new pattern is part of the data that the classifier was trained on or not [43]. A novelty detection system can be either off-line [44] (when the model is created once and not updated at all) or on-line [45] (when the model is updated as new data arrives). In the proposed scenario, a cyclic DOP also requires detecting if a new problem corresponds to a previous (training) problem. And as in novelty detection, the complete

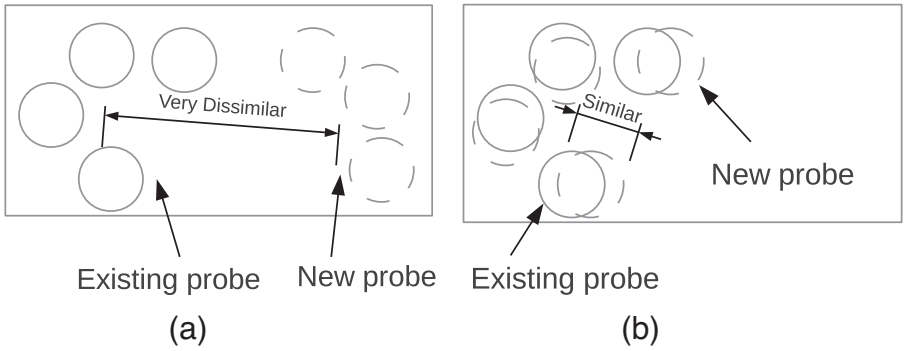

Fig. 2. Two possible scenarios involving memory update (existing probe is represented by solid circle while new probe is represented by dashed circle). (a) New probe is not similar to existing probe (new concept). (b) New probe is similar to existing probe (existing concept).

representation of a problem is not available due to computational constraints. That is, a memory must provide means of storing and recalling optimization problem concepts in an incremental manner rather than simply associating stored solutions with new problems (as in the memory-based optimization approaches found in the literature).

Markou and Singh [43] pointed the main issues related to novelty detection. Five of these issues are crucial in the envisioned scenario. The first is the principle of robustness and trade-off which means that the novelty detection approach must maximize the exclusion of novel patterns while minimizing the exclusion of known patterns. The second is the principle of parameter minimization which means that a novelty detection method must minimize the number of user-set parameters (mainly when we consider that in the envisioned application the data modeling technique must be closely integrated with the DPSO approach with minimal human intervention). The third is the principle of generalization which implies that the system should be able to generalize without confusing generalized information as novel. The fourth is the principle of adaptability which means that knowledge of novel samples must be integrated into the model. The fifth is the principle of computational complexity, which means that the computational complexity of a novelty detection should be as less as possible (also a very important issue in the given application, specially regarding detection, which should not be more expensive than re-optimizing).

It can be said that in the proposed application, the fourth and fifth principles are closely related. Retraining the model from scratch when novel optimization problem is detected would require storing all patterns (optimization history) seen so far, resulting in an ever increasing memory cost. Therefore, in the given scenario the model must be updated using only solutions from the new problem which can be seen as an incremental learning strategy. As defined by Jain et al. [46], in incremental learning, the learner has access only to a limited number of examples (patterns). In each step, an hypothesis can be built upon these examples and a former hypothesis in a way that (1) none of the intermediate hypotheses a learner explicates contradicts the data processed so far and (2) each intermediate hypothesis is maintained as long as it is consistent with the data seen. Gennari et al. [47] studied the use of incremental learning in building hierarchical models of concepts (concept formation). They observed that initial non-representative data may lead a learning system astray. The use of GMM in such case is very common [48,49] specially because it allows adaptability at a low computational cost when compared with other approaches such as neural networks [50].

From a memory-based optimization point of view, a new concept must (1) represent novelty when compared with existing concepts; (2) provide a precise manner of probing the fitness landscape. The basic memory unit in the proposed approach is a **probe** and it contains a density estimate of solutions plus the global best solution, both created after the optimization of a single image. When a new probe is created after a round of optimization, it should only be inserted if there is no similar probe in the memory. Otherwise it should be merged with the most similar probe in order to enforce (1). That is, a good memory management mechanism should keep the dissimilarity between new probes and probes in the memory consistently high. Put differently, inserts should occur when a new probe provides new information about the stream of optimization problems. Fig. 2 illustrates the two possible scenarios concerning a memory update.

By enforcing (1), memory redundancy is expected to be mitigated since the insert of new probes is constrained by a dissimilarity measure. In such case, memory elements are expected to resemble more Fig. 2a than Fig. 2b. That is, the memory is expected to be more diverse. This leads to a better usage of computational resources since the number of memory elements (probes) necessary to represent a given concept is minimized. Moreover, since the main objective of memory in the proposed system is to provide means of sampling the fitness landscape of unseen optimization problems, this increase in memory diversity should lead to an increased coverage of the sampled space (greater sampling diversity), enforcing (2). This means that during the optimization of a stream of images, as images are fed into the system, the amount of new information should decrease gradually as memorization takes place. Consequently the number of re-optimizations should gradually decrease after this memorization phase is complete. This allows for example, creating a memory on a laboratory environment (training mode) and then deliver this memory in a production environment.

4. Fast intelligent watermarking using Gaussian modeling of PSO populations

Fig. 3 depicts a new memory-based IW system that integrates density estimation in order to minimize memory size. Given an image $\boldsymbol{Co}_i$ picked from a stream of $|\boldsymbol{Co}|$ images (see **1** in Fig. 3), an attempt to recall the Short Term Memory (STM) – represented as $S\mathfrak{M}$ and comprising a mixture model of solutions Θ_S obtained during the optimization of a single image $\boldsymbol{Co_S}$ and the global best solution for that image $\boldsymbol{p}_{g,S}$ – is performed first (see **2** in Fig. 3). During a STM recall, a **set of solutions** (defined as $\boldsymbol{X}_{S,S}$) and their respective **fitness values** are sampled from Θ_S (including the global best, $\boldsymbol{p}_{g,S}$ stored in the STM). It is important to note that apart from $\boldsymbol{p}_{g,S}$, the position ($\boldsymbol{X}_{S,S}$) and fitness values ($\boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_S)$) of sentry solutions are an approximation of the positions and fitness values obtained during the optimization of $\boldsymbol{Co}_S$. The sentry solutions are re-evaluated for $\boldsymbol{Co}_i$ resulting in another set of fitness values $\boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_i)$. The Kolmogorov–Smirnov (KS) statistical test [51] is employed

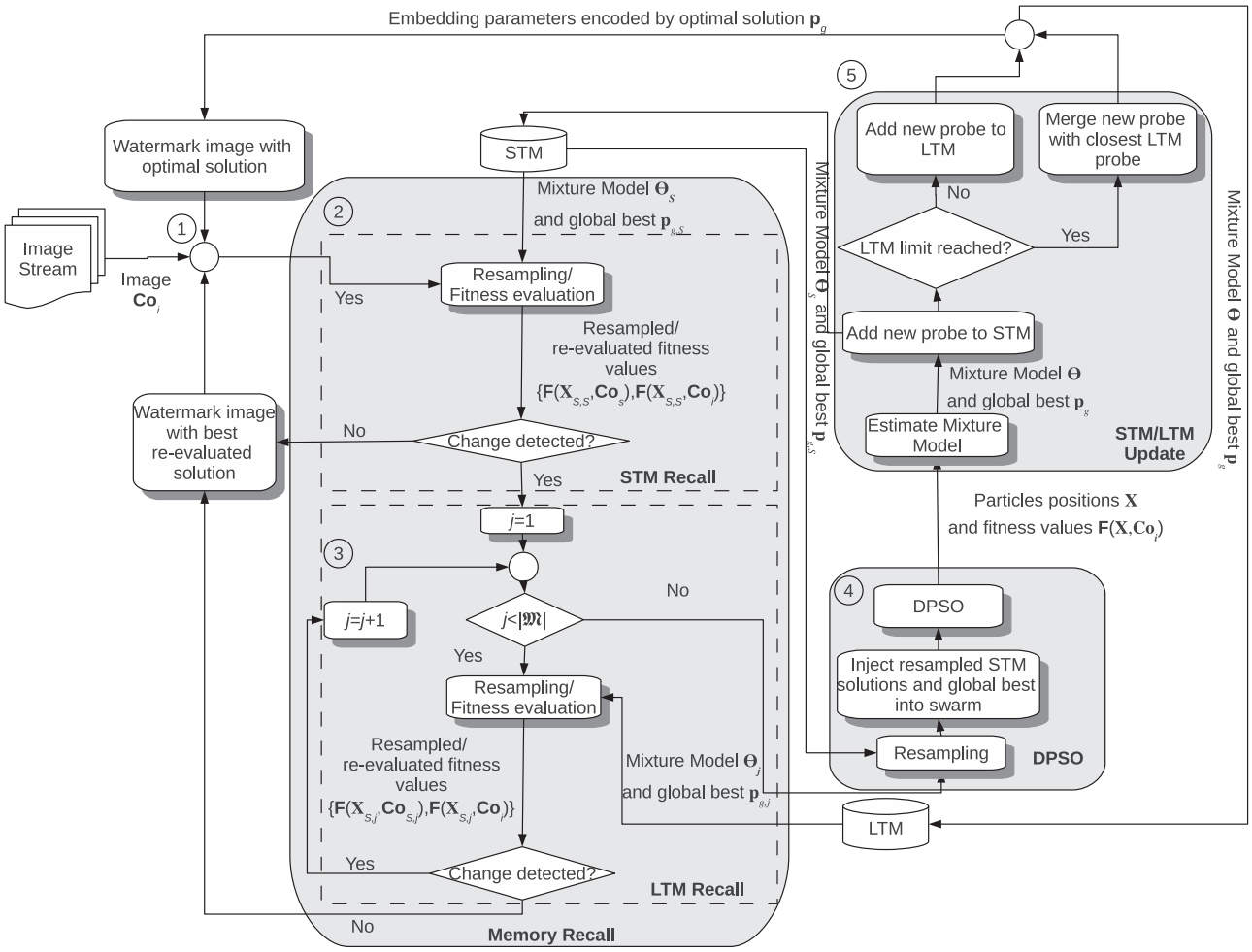

Fig. 3. Flowchart diagram representing the proposed method for fast intelligent watermarking of heterogeneous bi-tonal image streams using Gaussian mixture modeling of PSO populations (anchor points are employed in order to guide the reader).

in order to measure the similarity between the distribution of $\boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_S)$ and $\boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_i)$. If $KS(\boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_S), \boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_i))$ is smaller than a critical value D_α for a confidence level α, the watermarking parameters corresponding to the solution which resulted in the smallest $\boldsymbol{F}(\boldsymbol{X}_{S,S}, \boldsymbol{Co}_i)$ are employed right away for $\boldsymbol{Co}_i$, avoiding a costly optimization operation.

Otherwise (see **3** in Fig. 3), the same process is repeated for each mixture model Θ_j and global best $\boldsymbol{p}_{g,j}$ in the Long Term Memory (LTM) – represented as $\mathfrak{M}$ and comprising $|\mathfrak{M}|$ mixture models of solutions $(\{\Theta_1, \ldots, \Theta_{|\mathfrak{M}|}\})$ obtained during the optimization of several different images and their respective global best solutions $(\{\boldsymbol{p}_{g,1}, \ldots, \boldsymbol{p}_{g,|\mathfrak{M}|}\})$ – being the LTM probes sorted in reverse order of their number of successful recalls.

If a LTM probe $\mathfrak{M}_j$ results in a successful recall, the watermarking parameters corresponding to the solution which resulted in the smallest fitness value in $\boldsymbol{Co}_i$ are employed right away for that image. If no probe in the LTM resulted in successful recall, the Dynamic PSO (DPSO) technique described in [22] is employed in order optimize the embedding parameters for $\boldsymbol{Co}_i$ (see **4** in Fig. 3). A certain number of solutions re-sampled from the STM plus its respective global best are injected into the swarm, providing a starting point for optimization. After that, in the memory update (see **5** in Fig. 3), the optimization history (**position** and **fitness** of all solutions during all iterations) is employed in order to estimate a mixture model (Θ) of the fitness landscape. This mixture model plus the global best solution $(\boldsymbol{p}_g)$ obtained during optimization will form a probe to be added to the STM replacing previous probe. This probe is also either merged or inserted into the LTM based on the similarity between its mixture model and the mixture models of LTM probes. In the case of an insert, an older probe might be deleted to give room for the new one if memory limit has been reached.

The first level of memory allows for a fast recall in situations where a block of similar images (e. g. pages of a same document) appears. The second level allows for recall of solutions in situations where the fitness landscape associated with the image being watermarked is not similar to that of the last optimized image but still is similar to that of an image that had been processed before. Resampling of GMMs is expected to result in more diverse solutions which can cover a more significant region of the fitness landscape than would be possible with static solutions as the later tend to be concentrated in narrow regions of the fitness landscape (in the surroundings of previous optima). The rest of this section describes how the memory management approach addresses the three major concerns in memory-based optimization systems: (1) what to store in the memory; (2) how to organize and update the memory; (3) how to retrieve solutions from the memory. The memory update and retrieval algorithms are explained with details later in this section.

4.1. What to store?

In the proposed approach, a model of an optimization problem (which provides a more compact and precise representation than

selected individual solutions) is estimated through unsupervised learning techniques [52] based on the positions and fitness values of solutions in the optimization space. Because of the stream of optimization problems formulation of dynamic optimization, the distribution of these solutions is expected to be multi-modal. In such case, a finite mixture model is a powerful tool for estimating the distribution of these solutions. A mixture model consists of a linear combination of a limited (finite) number of models

$$p(\boldsymbol{x}|\Theta) = \sum_{j=1}^{K} \alpha_j p(\boldsymbol{x}|\theta_j) \tag{2}$$

where $p(\boldsymbol{x}|\Theta)$ is the probability density function (pdf) of a continuous random vector $\boldsymbol{x}$ given a mixture model Θ, K is the number of mixtures, α_j and θ_j are the mixing weights and parameters of the jth model (with $0 < \alpha_j \leq 1$ and $\sum_{j=1}^{K} \alpha_j = 1$). The mixture model parameters $\Theta = \{(\alpha_1, \theta_1), \ldots, (\alpha_K, \theta_K)\}$ are estimated using observed training data. The common approach is to employ a Gaussian distribution to represent each element ($\theta_j = \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}$) where $\boldsymbol{\mu}_j$ is the mean vector and $\boldsymbol{\Sigma}_j$ is the covariance matrix. A mixture containing Gaussian elements is known as a Gaussian mixture model (GMM).

The approach proposed in this paper builds a mixture model comprising both, the **parameter** and **fitness** space. Since it was observed that local best data results in density estimates that are over-fit to a specific problem, the approach employs current particle's position instead of local best data. We propose employing particle positions and fitness values rather than local best positions and fitness values in order to estimate the model as they provide a more general model of a given optimization problem. Every time re-optimization is triggered, historical particle position data (all generations of an optimization task) will be employed as a training dataset. Since the problem itself is dynamic, during an update, the LTM needs to adapt to new changes in the data but as well be capable of "forgetting" or pruning unnecessary information.

4.2. How to organize and update?

In the proposed memory scheme there are two levels of update – STM and LTM. After re-optimization, position and fitness data of all particles for all iterations is employed in order to estimate a mixture model Θ (Eq. (2)) of the fitness landscape. This model plus the global best will comprise a new probe to be added to the STM and LTM. The standard approach in the literature to estimate mixture parameters is to employ Expectation-Maximization (EM). In EM, Θ is estimated by gradually applying the E-step followed by the M-step until convergence is met. Convergence is attained when the log likelihood has stabilized over some dataset. A limitation regarding the use of standard EM in practical applications is the initialization of mixture components [53]. The main problem is that EM is unable to move components across low likelihood regions. EM is also unable to escape from situations where two or more components are similar, sharing the same data points. Another limitation is defining the appropriate number of components in a mixture. Usually when there are much more components than the necessary and the covariance matrices are unconstrained, some of the α_j's may approach zero and the corresponding covariance matrix may become arbitrarily close to singular.

Figueiredo and Jain [53] initialize the mixture with a large number of components, where each component is centered at a randomly picked data point. As the parameters are updated (1) components lacking enough data points to estimate their covariance matrices have their corresponding α's set to zero (component annihilation); (2) the number of components is gradually decreased until a lower boundary is achieved and then, the number that resulted in the best performance is chosen. They also proposed the following (log-likelihood) convergence criterion based on the Minimum Message Length (MML) which avoids local minima when two or more components are similar:

$$\mathfrak{L}(\Theta, \boldsymbol{x}) = \frac{N}{2} \sum_{\alpha_j > 0} \log\left(\frac{n\alpha_j}{12}\right) + \frac{k_{nz}}{2} \log\frac{n}{12} + \frac{k_{nz}(N+1)}{2} - \log p(\boldsymbol{x}|\Theta) \tag{3}$$

where k_{nz} is the number of components with $\alpha_j > 0$, n is the number of data points and N is the number of parameters (variables) in a given mixture (which is a function of d, the number of dimensions of X):

$$N = d + \frac{d(d+1)}{2} \tag{4}$$

Then, the E-step and M-step are applied iteratively. In the E-step, the posterior probability is computed [54]:

$$w_{ij}^{(t)} = \frac{\alpha_j p(\boldsymbol{x}_i|\theta_j)}{\sum_{k=1}^{K} \alpha_k p(\boldsymbol{x}_i|\theta_k)} \tag{5}$$

In the M-step the model parameters are updated. The following α update annihilates components lacking enough data points:

$$\alpha_j^{(t+1)} = \frac{max\{0, (\sum_{i=1}^{n} w_{i,j}) - N/2\}}{\sum_{k=1}^{K} max\{0, (\sum_{i=1}^{n} w_{i,k}) - N/2\}} \tag{6}$$

The remaining mixture parameters are updated as:

$$\boldsymbol{\mu}_j^{(t+1)} = \frac{\sum_{i=1}^{n} w_{i,j}^{(t)} \boldsymbol{x}_i}{w_{i,j}^{(t)}} \tag{7}$$

$$\boldsymbol{\Sigma}_j^{(t+1)} = \frac{\sum_{i=1}^{n} w_{i,j}^{(t)} (\boldsymbol{x}_i - \boldsymbol{\mu}_j^{(t+1)})(\boldsymbol{x}_i - \boldsymbol{\mu}_j^{(t+1)})^T}{w_{i,j}^{(t)}} \tag{8}$$

where d is the number of dimensions of $\boldsymbol{x}$.

4.2.1. Memory management operators – insert, merge and delete

In the given scenario, a memory update mechanism must address two fundamental issues of memory management. The first is what to do when a new probe is created. More specifically in which conditions should a new probe be merged with an existing probe and in which conditions should it be plainly inserted? The second is, in such situation, what to do when the memory is full? Should the new probe be merged with an existing probe even though they are not similar? Should an existing probe be deleted to make room for the new probe?

In order to mitigate these issues, we propose a selective memory update mechanism. In this mechanism, when the memory is due to be updated with a new probe, the $C2$ distance metric [55] (which provides a good trade-off between computational burden and precision) will determine if the new probe will be either added to the LTM (insert operation) or merged with an existing probe. The distance between two mixtures Θ and Θ' (or $C2(\Theta, \Theta')$) is defined as:

$$\Phi_{i,j} = (\boldsymbol{\Sigma}_i^{-1} + \boldsymbol{\Sigma}_j'^{-1})^{-1} \tag{9}$$

$$\eta_{i,j} = \boldsymbol{\mu}_i^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu}'_j) + \boldsymbol{\mu}_j^T \boldsymbol{\Sigma}'^{-1}_j (\boldsymbol{\mu}'_j - \boldsymbol{\mu}'_i) \tag{10}$$

$$C2(\Theta, \Theta') = -\log\left[\frac{2\sum_{i,j}\alpha_i\alpha'_j\sqrt{|\Phi_{i,j}|/(e^{\eta_{i,j}}|\boldsymbol{\Sigma}_i||\boldsymbol{\Sigma}'_j|)}}{\sum_{i,j}\alpha_i\alpha_j\sqrt{|\Phi_{i,j}|/(e^{\eta_{i,j}}|\boldsymbol{\Sigma}_i||\boldsymbol{\Sigma}_j|)} + \sum_{i,j}\alpha'_i\alpha'_j\sqrt{|\Phi_{i,j}|/(e^{\eta_{i,j}}|\boldsymbol{\Sigma}'_i||\boldsymbol{\Sigma}'_j|)}}\right] \tag{11}$$

If the distance is smaller than a given threshold, the new probe is merged with the closest probe in LTM. Otherwise an insert operation is performed. In such case, whenever the memory is full the probe with smallest number of successful recalls is deleted in order to give room for the new probe. Instead of using a fixed threshold we propose using an adaptive threshold, computed based on the minimum distance between new probes and probes on the LTM for the T previous updates (μ_δ^t). An insert occurs if $C2 - \mu_\delta^t$ is greater than the standard deviation for the same time-frame (σ_δ^t). Otherwise a merge operation is performed.

In what regards merging two mixtures, the basic approach consists of considering both mixtures as one ($p(x|\Theta) \cup p(x|\Theta')$) and then merge their components iteratively. A survey of techniques to merge components in a mixture of Gaussians can be found in [56]. Basically there are two main families of techniques: modality-based and those based on misclassification probability. In modality-based clustering, the components are assumed to be unimodal and then merging is performed until all mixture components are unimodal but any further merging would result in a component that is no longer unimodal. In misclassification probability approach, the notion of a cluster is not based on gaps between the densities but on how well two components (despite not being clearly separated) classify a sample generated from one of them. Split of mixture components [54,57] can also be employed in order to avoid situations where a single component is fit over multimodal data. However, it has been demonstrated in [56] that a series of distance-based merge operations is already enough in tackling multi-modality of mixture components.

We propose the use of Hennig [56] technique which is based on misclassification probability and resorts to the use of a Bhattacharyya distance. Differently than other techniques based on misclassification probability, Hennig's approach does not require the use of historical data. The Bhattacharyya distance is defined as:

$$\overline{\boldsymbol{\Sigma}} = \frac{1}{2}(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2) \tag{12}$$

$$d_B(\Theta_1, \Theta_2) = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \overline{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) + \frac{1}{2} \log \left(\frac{|(1/2)(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)|}{\sqrt{|\boldsymbol{\Sigma}_1||\boldsymbol{\Sigma}_2|}} \right) \tag{13}$$

This method works as follows. Given a tuning constant $d^* < 1$, compute the Bhattacharyya distance between all pairs of components (d_B). If $e^{-d_B} < d^*$ for all components stop merging and let the mixture as is. Otherwise, merge the two components with maximum distance and repeat the whole process. The merged component parameters $\{\alpha_M, \boldsymbol{\mu}_M, \boldsymbol{\Sigma}_M\} = \{\alpha_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1\} + \{\alpha_2, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2\}$ are defined as [57,54]:

$$\alpha_M = \alpha_1 + \alpha_2 \tag{14}$$

$$\boldsymbol{\mu}_M = \frac{\alpha_1 \boldsymbol{\mu}_1 + \alpha_2 \boldsymbol{\mu}_2}{\alpha_1 + \alpha_2} \tag{15}$$

$$\boldsymbol{\Sigma}_M = \frac{\alpha_1 \boldsymbol{\Sigma}_1 + \alpha_2 \boldsymbol{\Sigma}_2}{\alpha_1 + \alpha_2} \tag{16}$$

We propose merging the two components with minimum distance instead as it should result in smaller (more incremental) variations in the mixture components.

After the merge, if the number of mixture components is still higher than a given limit, unmerged components from the older mixture are deleted (purge). We propose the following purge approach: (1) compute Bhattacharyya distance between new/merged and old unmerged components; (2) delete the old unmerged component with the highest distance; (3) go to 1 until memory limit has been achieved.

The memory update mechanism is summarized in Algorithm 1. After optimization is over, the parameters of the new mixture (Θ_N) are estimated using position and fitness values of all particles found during the whole optimization process (step 1). This mixture along with the global best solution ($\boldsymbol{p}_g$) form a probe, to be added to the STM, replacing previous STM probe (step 2). After that, if the length of $\boldsymbol{\delta}$ (which contains the last n minimum $C2$ distances between new probes and probes in the LTM) is smaller than T (step 3), its mean and standard deviation ($\mu_{\boldsymbol{\delta}}^t$ and $\sigma_{\boldsymbol{\delta}}^t$) are set to user defined values ($\mu_{\boldsymbol{\delta}}^0$ and $\sigma_{\boldsymbol{\delta}}^0$, steps 4 and 5). Otherwise, they are computed based on $\boldsymbol{\delta}$ (steps 7 and 8). Then, the minimum $C2$ distance between new probe and probes in the LTM is added to $\boldsymbol{\delta}$ (steps 10 and 11). If the difference between the minimum $C2$ distance and $\mu_{\boldsymbol{\delta}}^t$ is greater than $\sigma_{\boldsymbol{\delta}}^t$ (step 12), the new probe is added to the LTM, noticing that the LTM probe with smallest number of recalls must be deleted if memory limit has been reached (steps 13–16). Otherwise the new probe is merged with the most similar probe in the LTM and mixture elements are purged if mixture size limit has been reached (steps 18 and 19). Finally, if the limit of vector $\boldsymbol{\delta}$ has been reached, its first (oldest) element is deleted (steps 21–23).

Algorithm 1. Memory update mechanism.

Inputs:
k_{max} – maximum number of components with $\alpha_j > 0$.
$\mathfrak{M}_S$ – Short Term Memory.
$\mathfrak{M} = \{\mathfrak{M}_1, \ldots, \mathfrak{M}_{|\mathfrak{M}|}\}$ – Long Term Memory.
$\mathfrak{D}$ – optimization history (set of all particle positions and fitness values for new image).
$L_{\mathfrak{M}}$ – maximum number of probes in LTM.
$\boldsymbol{\delta}$ – last T minimum $C2$ distances between a new probe and probes in the LTM.
$|\boldsymbol{\delta}|$ – number of elements in $\boldsymbol{\delta}$.
T – maximum size of $\boldsymbol{\delta}$.
$\mu_{\boldsymbol{\delta}}^0, \sigma_{\boldsymbol{\delta}}^0$ – initial mean and standard deviation of $\boldsymbol{\delta}$.

Output:
Updated memory.

1: Estimate Θ_N using $\mathfrak{D}$ [53].
2: Add Θ_N and $\boldsymbol{p}_g$ to $\mathfrak{M}_S$.
3: **if** $|\boldsymbol{\delta}| < T$ **then**
4: $\quad \mu_{\boldsymbol{\delta}}^t \leftarrow \mu_{\boldsymbol{\delta}}^0$
5: $\quad \sigma_{\boldsymbol{\delta}}^t \leftarrow \sigma_{\boldsymbol{\delta}}^0$
6: **else**
7: $\quad \mu_{\boldsymbol{\delta}}^t \leftarrow \frac{1}{|\boldsymbol{\delta}|} \sum_{i=1}^{|\boldsymbol{\delta}|} \delta_i$
8: $\quad \sigma_{\boldsymbol{\delta}}^t \leftarrow \sqrt{\frac{\sum_{i=1}^{n} (\delta_i - \mu_{\boldsymbol{\delta}}^t)^2}{|\boldsymbol{\delta}|}}$
9: **end if**
10: $i^* \leftarrow \underset{i}{argmin}\{C2(\Theta_N, \Theta_i)\}, \ \forall_{\Theta_i \in \mathfrak{M}}$
11: $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} \cup C2(\Theta_N, \Theta_{i^*})$
12: **if** $C2(\Theta_N, \Theta_{i^*}) - \mu_{\boldsymbol{\delta}}^t > \sigma_{\boldsymbol{\delta}}^t$ **then**
13: $\quad$ **if** $|\mathfrak{M}| = L_{\mathfrak{M}}$ **then**
14: $\quad\quad$ Remove LTM probe with smallest number of successful recalls.
15: $\quad$ **end if**
16: $\quad$ Add Θ_N and $\boldsymbol{p}_g$ to $\mathfrak{M}$
17: **else**
18: $\quad$ *Merge*(Θ_{i^*}, Θ_N) (Section 4.2.1)
19: $\quad$ Purge merged mixture in case number of elements exceed k_{max}.
20: **end if**
21: **if** $|\boldsymbol{\delta}| > T$ **then**
22: $\quad$ Remove δ_1.
23: **end if**

4.3. How to retrieve solutions?

In the proposed memory retrieval technique, an attempt to recall the STM is first made. If it succeeds, the best solution is employed immediately as the embedding parameter for that image. Otherwise, recall of probes in the LTM is attempted. If no probe can be successfully recalled, STM provides solutions to be injected into the swarm for a new round of optimization.

Since the proposed technique relies on the use of a GMM of particle positions (rather than selected particles as in the case-based technique [22]), recall requires sampling solutions from the GMM. Sampling N_s solutions from a mixture of Gaussians can be attained through a linear combination between a random vector and the eigen-decomposition of the covariance matrix, centered at the mean vector:

$$\boldsymbol{X}_s = \boldsymbol{\mu}_j + \boldsymbol{\Lambda}_j^{1/2}\boldsymbol{U}_j\boldsymbol{R}_s \tag{17}$$

where $\boldsymbol{X}_s$ is a sampled solution, s is the index of a solution sampled for the component j in the mixture ($\lfloor(N_s\alpha_j)+0.5\rfloor$ solutions are sampled per component), $\boldsymbol{\Lambda}_j$ and $\boldsymbol{U}_j$ are the eigen-decomposition of $\boldsymbol{\Sigma}_j$ ($\boldsymbol{\Sigma}_j = \boldsymbol{U}_j\boldsymbol{\Lambda}_j\boldsymbol{U}_j^{-1}$) and $\boldsymbol{R}_s$ is a vector with the same length as $\boldsymbol{\mu}_j$ whose elements are sampled from a normal distribution $N(0, \mathbf{I})$, being $\mathbf{I}$ the identity matrix.

The memory retrieval mechanism will basically bind the whole system together and is depicted in Algorithm 2. The best recalled solution $\boldsymbol{X}_o$ is initialized with null (step 1). After that, a given number of solutions are sampled from the STM mixture and best solution (steps 2 and 3). The fitness values of these sampled solutions are re-evaluated for the new image and if the KS statistic between these values and the sampled fitness values is smaller than a critical value (step 4), the best recalled solution is set with the solution that resulted in the smallest fitness value for the new image (step 5). Otherwise, the LTM probes are sorted in reverse order of their success counter (step 7) and the same process (re-sampling, followed by re-evaluation and KS test) is repeated for each probe in the LTM (steps 8–16). It is important to observe that in the event of a successful LTM recall, the success counter of that LTM probe is incremented (step 12) and the best recalled solution is set with the recalled solution that resulted in the smallest fitness for the new image (step 13). If the best recalled solution is null (step 18), the top STM re-sampled solutions are injected into the swarm and re-optimization is triggered (step 19). Otherwise, the embedding parameters encoded by the best recalled solution are employed in the watermarking of the new image (step 21).

Algorithm 2. Memory retrieval mechanism.

Inputs:
$\boldsymbol{Co}$ – cover image.
$\mathfrak{M}_S$ – Short Term Memory.
$\mathfrak{M} = \{\mathfrak{M}_1, \ldots, \mathfrak{M}_{|\mathfrak{M}|}\}$ – Long Term Memory.
N_i – amount of injected solutions (%).
D_α – critical value for KS-test.

Output:
Watermarked image (based on parameters encoded by optimal solution $\boldsymbol{X}_o$).

1: $\boldsymbol{X}_o \leftarrow \emptyset$
2: $\boldsymbol{X}_{S,S} \leftarrow Sample(N_s, \mathfrak{M}_S)$
3: $\boldsymbol{X}_{S,S} \leftarrow \boldsymbol{X}_S \cup \boldsymbol{p}_{g,S}$
4: **if** $KS(\boldsymbol{F}(\boldsymbol{X}_{S,S}, \mathbf{Co}_S), \boldsymbol{F}(\boldsymbol{X}_{S,S}, \mathbf{Co})) \leq D_\alpha$ **then**
5: Set $\boldsymbol{X}_o$ with solution which resulted in smallest $\boldsymbol{F}(\boldsymbol{X}_{S,S}, \mathbf{Co})$.
6: **else**
7: Sort $\mathfrak{M}$ by *Count* (in reverse order).
8: **for** $i \in [1, |\mathfrak{M}|]$**do**
9: $\boldsymbol{X}_{S,i} \leftarrow Sample(N_s, \mathfrak{M}_i)$
10: $\boldsymbol{X}_{S,i} \leftarrow \boldsymbol{X}_{S,i} \cup \boldsymbol{p}_{g,i}$
11: **if** $KS(\boldsymbol{F}(\boldsymbol{X}_{S,i}, \mathbf{Co}_i), \boldsymbol{F}(\boldsymbol{X}_{S,i}, \mathbf{Co})) \leq D_\alpha$ **then**
12: $Count_i \leftarrow Count_i + 1$
13: Set $\boldsymbol{X}_o$ with solution which resulted in smallest $\boldsymbol{F}(\boldsymbol{X}_{S,i}, \mathbf{Co})$.
14: Exit for.
15: **end if**
16: **end for**
17: **end if**
18: **if** $\boldsymbol{X}_o = \emptyset$ **then**
19: Inject the N_i best solutions in $\boldsymbol{X}_{S,S}$ into the swarm (replacing its N_i worst solutions), re-optimize and update memory (Algorithm 1).
20: **else**
21: Use $\boldsymbol{X}_o$ as optimal embedding parameter.
22: **end if**

The proposed memory management scheme (insert/update) is illustrated using five different bi-modal sets of 2D Gaussian points. For simplicity, all sets of points have the same covariance matrix and only their mean vectors vary. Each bi-modal set of points will simulate the behavior of particles positions during the optimization of a 2D problem. In this example the memory size is limited to three probes. Fig. 4a shows the five bi-modal sets of points. From $t = 0$ to $t = 2$, memory update consists of insert operations (Fig. 4b). Memory limit is reached at $t = 3$ leading to an insert followed by a delete (Fig. 4c). At $t = 4$, one of the components appears close to a previously seen component and both components are merged (Fig. 4d). It is worth noticing that in all cases, the knowledge about a new scenario is acquired without completely "forgetting" previous knowledge.

5. Simulation results

5.1. Experimental protocol

5.1.1. Databases

The two watermarks to be employed in all experiments for all databases are same defined in [22], namely, the 26×36 BancTec logo (Fig. 5a) as robust watermark and the 36×26 Université du Québec logo (Fig. 5b) as fragile watermark.

Since the main objective of the proposed method is to tackle high throughput adaptive watermarking in heterogeneous streams of document images, the database of document images of the University of Oulu's MediaTeam [58] (OULU-1999) is employed in order to validate the performance of the proposed technique in such task (scenario A). This database is considerably heterogeneous, scanned at 300 dpi with 24-bit color encoding. Since this database is not bi-tonal, it was binarized using the same protocol as in [22]. However, it was observed that some of the images contained very large uniform regions (with only white pixels). These images lack the capacity necessary to embed the watermarks described above. Thus, a reject rule was applied: all images with less than 1872 flippable pixels were discarded (pixels with SNDM equal to 0). This is the minimum number of flippable pixels in order to embed the 936-bit robust watermark presented above with a quantization step size ($Q = 4$) which is the minimum level of robustness necessary for multi-level embedding. With this rule, 15 of the 512 images from the OULU-1999 database were excluded. The second objective of the proposed method is to allow learning the different categories of problems found throughout the stream of optimization problems. To validate this, two separate sets of images – training and testing – are required. For this reason, the OULU-1999 database was split in two subsets. The training (memorization) subset contains 100 images chosen randomly from OULU-1999 and is named OULU-1999-TRAIN. The remaining 397 images compose the testing (generalization) subset which is named OULU-1999-TEST. Since the images on this database are from 19 different categories (Table 2), there is a lot of variation in the size and number of flippable pixels among these images.

Although the proposed technique was devised to tackle intelligent watermarking of heterogeneous image streams, in a real life scenario it needs to adapt to watermarking of homogeneous image streams as well. To validate this, the proposed technique will be also evaluated in two different (training and testing) homogeneous image streams, namely TITI-61 and CVIU-113-3-4 [22] (scenario B). Finally, the performance on an unconstrained (homogeneous/heterogeneous) stream (scenario C) will be validated. For this purpose, the OULU-1999-TEST and CVIU-113-3-4 streams were concatenated and the images were shuffled in order to create a larger stream named SHUFFLE, to assess how does the proposed approach scales as the length of the stream grows. A larger learning

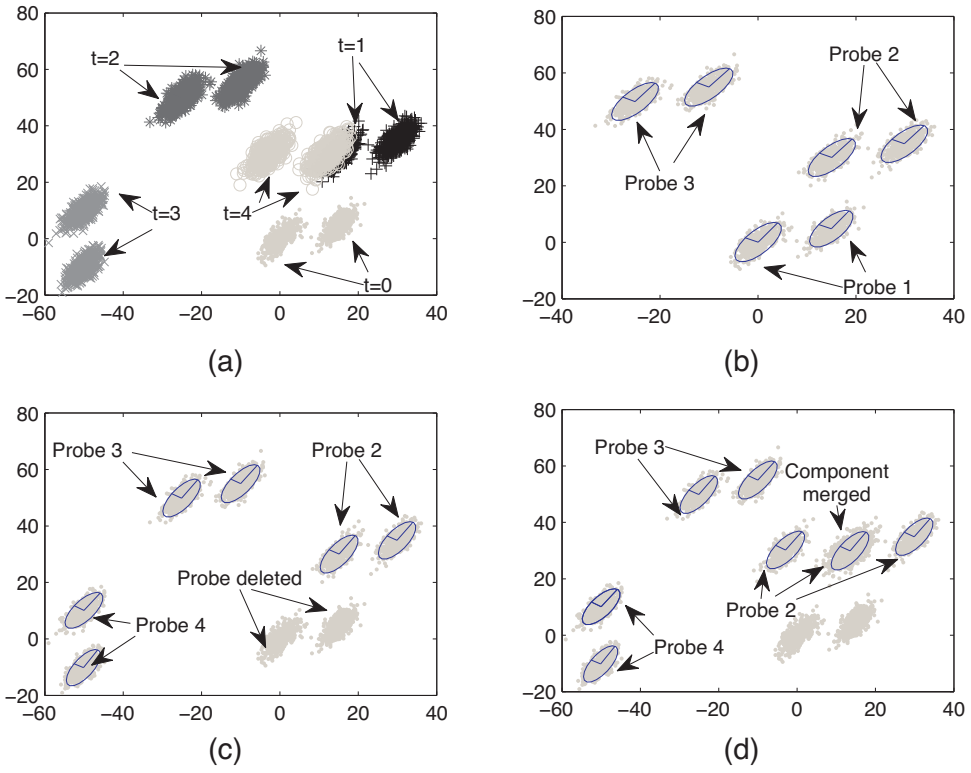

Fig. 4. Illustration of memory update technique. (a) Bi-modal Gaussian points. (b) Three probes added between $t = 0$ and $t = 2$. (c) New probe at $t = 3$ is inserted while that of $t = 0$ is deleted. (d) Merging of probe obtained at $t = 4$ with that of $t = 1$. One of the components of the new probe was overlapped with another one of the old probe and both were merged.

stream was also created by concatenating TITI-61 and OULU-1999-TRAIN streams.

5.1.2. Methodology

The memory management mechanism should mitigate redundancy in the LTM. Therefore, a sensitivity analysis will be conducted in a first moment in order to find out how do the distance between probes and sampled particles diversity relate. The current method will be applied to the OULU-1999-TRAIN database but forcing re-optimization for each image and without using any memory management technique. The purpose of this experiment is to build a large memory (containing 100 probes) and then assess the distance between these probes in order to set an initial distance threshold for the proposed technique. As each probe is inserted in the LTM, the C2 distance [55] between this probe and the probes already in the memory will be computed. Then 2000 solutions will be sampled uniformly from all probes and the normalized mean of the pairwise distance among individuals in the population D^N_{PW} [59] will be computed for the sampled solutions:

$$D^N_{PW} = \frac{(2/|X|(|X|-1))\sum_{i=2}^{|X|}\sum_{j=1}^{i-1}\sqrt{\sum_{k=1}^{d}(x_{i,k}-x_{j,k})^2}}{NMDF} \quad (18)$$

where $|X|$ is the population size, $x_{i,k}$ is the kth parameter encoded by the ith individual, d is the landscape dimensionality and $NMDF$ is the normalization (factor) with maximum diversity so far. This metric reflects quite well the population diversity.

Considering the number of probes in LTM is $|\mathfrak{M}|$, this involves sampling $2000/|\mathfrak{M}|$ from each probe. A plot of the minimum distance between the new probe and the probes already in the memory (min_{C2}) versus the diversity of the sampled population should show how does limiting the number of insert operations based on a distance threshold impacts sampling diversity.

We propose a novel metric based on the same principle of D^N_{PW} but tailored to measure the diversity of the LTM, namely the normalized pairwise distance between probes:

$$D^N_{PWM} = \frac{(2/|\mathfrak{M}|(|\mathfrak{M}|-1))\sum_{i=2}^{|\mathfrak{M}|}\sum_{j=1}^{i-1}C2(\Theta_i,\Theta_j)}{NMDF_{C2}} \quad (19)$$

where $NMDF_{C2}$ is the normalization (factor) with maximum diversity so far (applied to the $C2$ metric). This metric will show the amount of inter-probe diversity while D^N_{PW} will show the amount of intra-probe diversity.

The proposed management strategy should allow the memory to quickly adapt to an abrupt change in the stream of optimization problems. First we have to define what an abrupt change is. In this specific scenario an abrupt change is a change in the stream of optimization problems that requires re-optimization to be triggered. Since defining when re-optimization should be triggered is subjective, we propose the use of Kullback–Leibler (KL) [60]

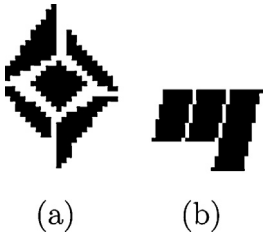

Fig. 5. Bi-tonal logos used as watermarks. (a) 26×36 BancTec logo. (b) 36×26 Université du Québec logo.

Table 2
OULU-1999 database structure.

Category	OULU-1999-TRAIN					OULU-1999-TEST				
	#	# Pixels				#	# Pixels			
		Regular		Flippable			Regular		Flippable	
		Min	Max	Min	Max		Min	Max	Min	Max
Addresslist	0	0	0	0	0	6	2.2×10^6	6.6×10^6	3.7×10^5	2×10^6
Advertisement	5	4.9×10^6	7.9×10^6	8.1×10^5	2.6×10^6	19	1.1×10^6	8.1×10^6	1.5×10^5	2.5×10^6
Article	51	1.8×10^6	7.9×10^6	2.5×10^5	3.0×10^6	180	2.0×10^6	15.7×10^6	2.4×10^5	3.0×10^6
Businesscards	1	6.2×10^5	6.2×10^5	9.8×10^4	9.8×10^4	10	5.3×10^5	1.1×10^6	7.8×10^4	3.4×10^5
Check	0	0	0	0	0	3	3.4×10^5	1.4×10^6	1.3×10^5	1.9×10^5
Color segmentation	1	2.5×10^6	2.5×10^6	7.9×10^5	7.9×10^5	7	1.5×10^6	7.3×10^6	4.5×10^5	3.3×10^6
Correspondence	6	2.0×10^6	5.2×10^6	2.1×10^5	1.1×10^6	18	1.1×10^6	4.9×10^6	1.4×10^5	8.2×10^5
Dictionary	1	2.8×10^6	2.8×10^6	3.3×10^5	3.3×10^5	9	1.6×10^6	3.3×10^6	2.3×10^5	6.6×10^5
Form	9	7.3×10^5	5.5×10^6	1.2×10^5	1.1×10^6	14	4.5×10^5	3.9×10^6	7.6×10^4	7.5×10^5
Line drawing	0	0	0	0	0	10	1.5×10^6	7.1×10^6	1.3×10^5	1.1×10^6
Manual	6	3.0×10^6	4.1×10^6	2.8×10^5	8.7×10^5	29	2.4×10^6	4.1×10^6	2.6×10^5	8.6×10^5
Math	4	3.2×10^6	3.9×10^6	2.0×10^5	3.1×10^5	13	3.2×10^6	3.9×10^6	1.8×10^5	3.8×10^5
Music	0	0	0	0	0	4	3.9×10^5	2.1×10^6	8.8×10^4	4.0×10^5
Newsletter	4	7.6×10^6	7.9×10^6	1.3×10^6	1.7×10^6	37	1.5×10^6	7.9×10^6	1.3×10^5	2.2×10^6
Outline	4	1.6×10^6	4.1×10^6	2.5×10^5	9.1×10^5	13	3.1×10^6	5.2×10^6	3.2×10^5	1.0×10^6
Phonebook	4	7.9×10^6	8.1×10^6	2.3×10^3	2.4×10^3	3	7.9×10^6	8.1×10^6	1.4×10^6	2.2×10^6
Program listing	2	3.8×10^6	7.0×10^6	6.6×10^5	1.3×10^6	10	3.6×10^6	7.3×10^6	3.9×10^5	2.0×10^6
Street map	0	0	0	0	0	5	1.8×10^6	1.1×10^7	3.5×10^5	6.2×10^6
Terrainmap	2	7.0×10^6	1.0×10^7	2.6×10^6	6.0×10^6	7	2.9×10^6	1.1×10^7	1.2×10^6	6.2×10^6
Total	100					397				

divergence measure between the cumulative sets of particles of two consequent optimization problems in order to precisely verify this variation. The KL divergence is a measure of information gain between two distributions. A cumulative set of particles at instant t (or $\boldsymbol{X}_{C,t}$) is the set of all particles seen in all generations of all problem instances up to t. The KL divergence between cumulative sets of particles at instants t and $t-1$ is defined as $D_k(\boldsymbol{X}_{C,t-1} || \boldsymbol{X}_{C,t})$. The method proposed in [60] is non-parametric and depends on a k-nearest neighborhood estimate (that is, depends on a neighborhood size parameter). This parameter was set to 10 in our experiments as seen in [60].

The number of previous updates T employed to compute the adaptive threshold will be set to 10. The mean and standard deviation of the minimum distance obtained in the memory fill up experiments with no attack (which are 361.7 and 172.3, respectively) will be employed as an initial minimum distance threshold in the memory update. These values were obtained by simply measuring the minimum $C2$ distance during inserts for the memory fill up experiments (which resulted in 99 $C2$ values) and then, computing their mean and standard deviation.

In order to measure the impact in the computational cost we will analyze how does the number of fitness evaluations behave in different scenarios. One of the metrics that will be employed to this end is the average number of fitness evaluations per image (*AFPI*). A second metric to be employed is the cumulative number of fitness evaluations (F_{Evals}) which is the total number of fitness evaluations required to optimize the whole image stream. A third is the decrease in the number of fitness evaluations (*DFE*), computed as:

$$DFE = 1 - \frac{F_{Evals,M}}{F_{Evals,F}} \tag{20}$$

where $F_{Evals,M}$ is the cumulative number of fitness evaluations for the memory based approach and $F_{Evals,F}$ is the cumulative number of fitness evaluations for full optimization. For each experiment, the mean and standard variation of *AFPI*, the F_{Evals} and the *DFE* is presented.

The reference points for the Chebyshev Weighted Aggregation were set to $r_1 = r_2 = r_3 = 0.01$ based on sensitivity analysis using the OULU-1999-TRAIN dataset. The scaling factor of the *DRDM* (α_r) was set to 0.53 based on the largest *DRDM* value found for all fitness evaluations during the full optimization of all images of the OULU-1999-TRAIN dataset. These parameters have been used in the test streams to validate their generalization performance.

The confidence level (α) of the KS statistic will be set to 0.95, which corresponds to a coefficient $c_\alpha = 1.36$ and a critical value (D_α) of 0.43 in order to allow a comparison with the results reported in [22]. The LTM size is limited to 20 probes. All the simulations were performed first with no attack and then with cropping of 1%.

DPSO parameters are set as in [22]. Constants c_1 and c_2 are set to 2.05 while χ is set to 0.7298. Population size is set to 20 particles and optimization halts if the global best has not improved for 20 iterations. The neighborhood size of the L-Best topology is set to 3.

5.2. Overview

In terms of computational burden, the GMM-based approach outperformed the case-based approach for the heterogeneous streams and underperformed for some of the homogeneous streams (Table 3).

However, the watermarking performance of the GMM-based approach is equivalent to that of the case-based approach for the heterogeneous streams but at a smaller computational burden (Table 4). Moreover, there was a significant improvement in watermarking performance for the homogeneous streams (mainly due to the modified fitness function). It is important to observe that mainly for the cropping 1%, the worsening in computational cost is largely offset by the improvement in watermarking performance.

Fig. 6 summarizes the computational and memory burden results.

5.3. Scenario A – optimization of heterogeneous streams of bi-tonal images using memory-based DPSO versus full PSO:

5.3.1. LTM fill up

In the first experiment, performed on the OULU-1999-TRAIN stream, the memory limit was removed and re-optimization was forced on each image transition. This led to the creation of 100 probes. Fig. 7 shows the normalized pairwise distance between

Table 3
Computational cost performance. *AFPI* is the average number of fitness evaluations per image where the mean μ and standard deviation σ are presented as $\mu(\sigma)$. F_{Evals} is the cumulative number of fitness evaluations required to optimize the whole stream and *DFE* is the decrease in the number of fitness evaluations compared to full optimization. An asterisk (∗) indicates results extracted from [22].

Attack	Database	Learning	Full PSO		Case-based			GMM-based		
			AFPI	F_{Evals}	*AFPI*	F_{Evals}	*DFE* (%)	*AFPI*	F_{Evals}	*DFE* (%)
No attack	OULU-1999-TRAIN	No	925 (286)	92,520	564 (630)	56,380	39.1	66 (194)	6580	92.9
No attack	OULU-1999-TEST	No	1007 (341)	399,840	270 (551)	107,060	73.2	59 (188)	23,280	94.2
No attack	OULU-1999-TEST	Yes	1007 (341)	399,840	464 (842)	184,180	53.9	42 (133)	16,700	95.8
No attack	TITI-61	No	844 (226)*	51,460*	46 (134)*	2760*	94.6*	84 (224)	5140	92.6
No attack	CVIU-113-3-4	No	882 (251)*	301,580*	32 (103)*	10,720*	96.4*	76 (233)	26,000	91.4
No attack	CVIU-113-3-4	Yes	882 (251)*	301,580*	31 (83)*	10,560*	96.5*	49 (157)	16,600	95.4
No attack	SHUFFLE	No	1026 (345)	758,500	273 (571)	201,640	73.4	66 (189)	48,840	93.6
No attack	SHUFFLE	Yes	1026 (345)	758,500	259 (613)	191,240	74.8	54 (179)	40,220	94.7
Cropping 1%	OULU-1999-TRAIN	No	887 (340)	88,740	351 (455)	35,100	60.5	179 (363)	17,860	79.9
Cropping 1%	OULU-1999-TEST	No	860 (310)	341,520	177 (351)	70,300	79.4	83 (212)	32,920	90.4
Cropping 1%	OULU-1999-TEST	Yes	860 (310)	341,520	148 (301)	58,940	82.7	67 (205)	26,760	92.2
Cropping 1%	TITI-61	No	911 (237)*	55,580*	66 (200)*	3960*	92.9*	52 (178)	3200	94.8
Cropping 1%	CVIU-113-3-4	No	872 (251)*	298,100*	26 (36)*	8740*	97.1*	50 (166)	16,980	94.5
Cropping 1%	CVIU-113-3-4	Yes	872 (251)*	298,100*	25 (10)*	8480*	97.2*	21 (4)	7120	97.7
Cropping 1%	SHUFFLE	No	887 (320)	798,100	151 (292)	111,420	86	67 (194)	49,780	93.8
Cropping 1%	SHUFFLE	Yes	887 (320)	798,100	128 (252)	94,780	88.1	49 (136)	36,300	95.5

probes (D^N_{PWM}) for both, no attack and cropping 1%. It is possible to observe that in both cases, inter-probe diversity decreases steeply until image 11 for the cropping 1% case and image 12 for the no attack case. After that, for the no attack case it rebounds sharply until image 20 and then becomes stable. For the cropping 1% it rebounds softly and becomes stable.

It is interesting to observe that the sampling diversity has a similar behavior (Fig. 8). If a probe brings new knowledge to the LTM, the sampling diversity should increase. However, it follows a downward trend as new probes are added indiscriminately which means that in most cases, the new probes do not imply in new knowledge about the fitness landscape (the sampled solutions are just probing already probed areas).

In Fig. 9 it is possible to observe that the minimum distance between new probes and probes already in the memory behaves in a similar manner. Although the minimum distance itself is less stable than the LTM diversity, its moving average ($mov\_avg(min_{C2})$) follows a steep downward trend for the first 11–12 images and then becomes stable. It is worth noticing that a steep variation in the minimum distance is associated with a steep change in the LTM diversity. For example, for the no attack case, the D^N_{PWM} decreases steeply between images 1 and 12 and then increases gradually until image 20. Nearly at the same time-frame, $mov\_avg(min_{C2})$ follows a similar trend. It is slightly slower because of the window size chosen. A smaller window size would give less importance to the min_{C2} of previous probes and make it follow more rapidly the trend of D^N_{PWM}. The same phenomenon can be observed for the cropping 1% case.

The Kullback–Leibler (KL) divergence [60] between the cumulative sets of particles at instants t and $t-1$ (Fig. 10) behaves similarly. It is possible to see here that from an information theoretical standpoint, the particles of a given optimization problem provide new information about the stream of optimization problems until around image 30 (for both no attack and cropping 1%). After that, except for small disturbances like for image 60 in the no attack case, swarm solutions do not bring new knowledge about the stream of optimization problems. Most importantly, the KL divergence follows a trend similar to that of the moving average of the minimum *C*2 distances seen in Fig. 9. Therefore, the proposed strategy of only performing an insert operation if distance between the new probe and probes already in the memory is above a certain threshold should maximize the amount of new information brought by each new probe.

5.3.2. Adaptive memory management

The GMM-based technique resulted in less re-optimizations when compared with the case-based approach for all experiments involving heterogeneous image streams which consequently led to a bigger decrease in the number of fitness evaluations when compared to full optimization. It is also important to mention that the use of a training sequence resulted in a further decrease in computational burden for the OULU-1999-TEST stream in both cases (with and without attack). Despite the decrease in computational burden, the watermarking performance of the GMM-based technique is comparable to that of the case-based technique. The reason is that the solutions sampled from the GMM are less biased to a particular optimization problem than the case-based solutions.

The same was observed for the cropping 1% case. The proposed GMM-based memory scheme resulted in considerably less re-optimizations than the case-based memory scheme for the three heterogeneous streams with an equivalent watermarking performance. For this reason, the number of fitness evaluations decreased significantly when compared to full optimization.

An analysis of LTM dynamics for the OULU-1999-TRAIN stream shows that the proposed memory management scheme resulted in a more diverse memory than that obtained in the memory fill-up experiment (Fig. 11). What is interesting here is that for the no attack case, re-optimization was triggered 28 times. However, it resulted in an insert for only 5 of these cases. For the remaining 23 cases, a merge took part. A similar situation occurred for the cropping 1% case. Re-optimization was triggered 21 times but the number of inserts was 4 (with 17 merges).

At the same time, the sampled solutions have more diversity than when insert is used indiscriminately (Fig. 12). It is possible to observe also that the two plots in Fig. 12 are more stable than those of Fig. 8. This means that the sampling obtained by the use of the proposed memory scheme not only improves diversity but is also more consistent. This shows that this strategy of limiting insert operations to cases where the distance between new probes and probes in the memory is above an historic average helps to improve the diversity of the sampled solutions.

The plot of minimum C2 distance between new probes and probes in the memory (Fig. 13) gives another perspective about the memory dynamics. In this plot, a min_{C2} of zero means that the memory was not updated (that is, re-optimization was not triggered). It is possible to observe that insert operations have in general a min_{C2} that is many times greater than that of merge operations. It

Table 4
Watermarking performance. Here, † is the *DRDM*, ‡ is the *BCR* robust, § is the *BCR* fragile. For all values, the mean μ and standard deviation σ per image are presented in the following form: $\mu(\sigma)$. *DRDM* is presented with two decimal points and *BCR* is presented in percentage (%) with one decimal point. An asterisk (*) indicates results extracted from [22].

Attack	Database	Learning	Full PSO			Case-based			GMM-based		
			†	‡	§	†	‡	§	†	‡	§
No attack	OULU-1999-TRAIN	No	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0)
No attack	OULU-1999-TEST	No	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0)
No attack	OULU-1999-TEST	Yes	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0)	0 (0)	99.9 (0.4)	99.9 (0.7)
No attack	TITI-61	No	0 (0)*	99.9 (0.5)*	99.7 (0.6)*	0 (0)*	99.8 (0.9)*	99.6 (1.2)*	0 (0)	100 (0)	100 (0)
No attack	CVIU-113-3-4	No	0 (0)*	99.5 (3.6)*	99.3 (3)*	0 (0)*	99.5 (3.3)*	99.6 (2.7)*	0 (0)	100 (0)	100 (0)
No attack	CVIU-113-3-4	Yes	0 (0)*	99.5 (3.6)*	99.3 (3)*	0 (0)*	99.4 (3.3)*	99.2 (2.8)*	0 (0)	100 (0)	100 (0)
No attack	SHUFFLE	No	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0.1)	0 (0)	100 (0)	100 (0)
No attack	SHUFFLE	Yes	0 (0)	100 (0)	100 (0)	0 (0)	100 (0)	100 (0.1)	0 (0)	100 (0)	100 (0)
Cropping 1%	OULU-1999-TRAIN	No	0.03 (0.03)	98.4 (2.1)	99.7 (0.6)	0.03 (0.03)	97.9 (2.6)	99.6 (1)	0.03 (0.03)	97.1 (3.8)	99.4 (1)
Cropping 1%	OULU-1999-TEST	No	0.03 (0.04)	98.4 (2.2)	99.6 (0.6)	0.03 (0.03)	97.2 (3.6)	99 (1.6)	0.03 (0.03)	96.7 (4)	99.1 (1.5)
Cropping 1%	OULU-1999-TEST	Yes	0.03 (0.03)	98.4 (2.2)	99.6 (0.6)	0.03 (0.03)	97.5 (2.8)	99.3 (1.2)	0.03 (0.04)	97.5 (3.3)	99.4 (1.1)
Cropping 1%	TITI-61	No	0 (0)*	*92 (6.5)*	94 (4)*	0 (0)*	92.4 (6)*	94.8 (4.5)*	0.03 (0.03)	99 (1.8)	99.7 (0.04)
Cropping 1%	CVIU-113-3-4	No	0 (0)*	89.6 (7.1)*	92.5 (5.3)*	0 (0)*	86.6 (7.2)*	90 (5.9)*	0.04 (0.05)	98.3 (3)	99.5 (0.8)
Cropping 1%	CVIU-113-3-4	Yes	0 (0)*	89.6 (7.1)*	92.5 (5.3)*	0 (0)*	90.5 (6.4)*	93.4 (5.1)*	0.04 (0.06)	98.1 (0.03)	99.4 (1)
Cropping 1%	SHUFFLE	No	0.03 (0.04)	98.6 (2.2)	99.6 (0.5)	0.03 (0.04)	97.9 (3)	99.3 (1.1)	0.03 (0.04)	97.1 (4.4)	98.9 (1.8)
Cropping 1%	SHUFFLE	Yes	0.03 (0.04)	98.6 (2.2)	99.6 (0.5)	0.03 (0.04)	98 (2.8)	99.4 (1)	0.03 (0.04)	97.1 (4.3)	99.1 (1.4)

becomes clear as well that in both cases, for the first 30 images, the update frequency is high, which means that learning (memorization) is taking place, and then updates become less frequent. When we go back to the KL divergence plot in Fig. 10 it becomes clear that this memorization phase occurs when there is novelty in the stream of optimization problems.

5.3.3. Impact of choice of confidence level

In terms of memory size, the worst case scenario for the GMM-based technique results in a memory that is a fraction of the size obtained for the case-based approach (Fig. 14).

Fig. 15 shows the cumulative number of fitness evaluations for the case-based and GMM-based approaches with a confidence level of 0.8 (OULU-1999-TEST with learning, no attack). It is possible to observe that between images 137 and 240 the computational cost for the case-based memory approach is higher than that of full optimization while for the GMM-based approach it is practically stable after a learning phase that lasts until image 80. This illustrates the main limitation of case-based memory management strategy and the main advantage of GMM-based memory. It is important to observe that this result was obtained in a considerably small database. In a real world scenario, involving thousands or even millions of images, an ever growing memory would pose a serious issue to the performance of the case-based intelligent watermarking system.

The main reason for improved performance when compared with the case-based approach is that probe solutions in the case-based memory scheme are less diverse than those of the GMM-based memory. That is, case-based solutions only cover the near optimal region and for this reason are very sensitive to small variations in fitness values caused by a change of type II (basically, these solutions are over-fit to the images that generated them). However, the solutions sampled from the GMM have a more general coverage of the fitness landscape, mainly because they are generated from a density estimate of all solutions found during optimization and consequently, perform better in avoiding unnecessary re-optimizations than the case-based approach.

5.3.4. Memorization performance

In the first memorization experiment we picked a probe that resulted in re-optimization followed by a merge for OULU-1999-TRAIN with cropping of 1% (the probe of image 38) and performed multiple attempts to recall the new and merged probes in three situations: (1) new probe before merge; (2) old probe before merge; (3) merged probe. The first simulation should give an idea of the true acceptance rate of the proposed technique while the second simulation should give an idea of its true reject rate. The third simulation by its way should give an idea of at what point, incorporating new knowledge will improve the recall rate of a previous probe (adaptability).

In scenario (1), the newly created probe was recalled in all cases, which means a true acceptance rate of 100% (obviously, for this sample size, or put differently, a false reject rate smaller than 1%). In scenario (2), the old probe was accepted only 30 times of the cases, which means a true reject rate of 70%. Finally, in scenario (3), the merged probe resulted in an accept rate of 73%. That is, the merged probe has a better performance for image 38 than the old unmerged probe. At the same time, it is not as fit to the new image as the newly created (unmerged) probe which means it is less biased to a specific image.

In the second memorization experiment, the same stream (OULU-1999-TRAIN) with cropping of 1% was optimized twice, but using the memory of the first run as a starting point for the second run. The first run resulted in 17 re-optimizations while the second run resulted only in 10. This demonstrates that the proposed approach can memorize a stream of optimization problems

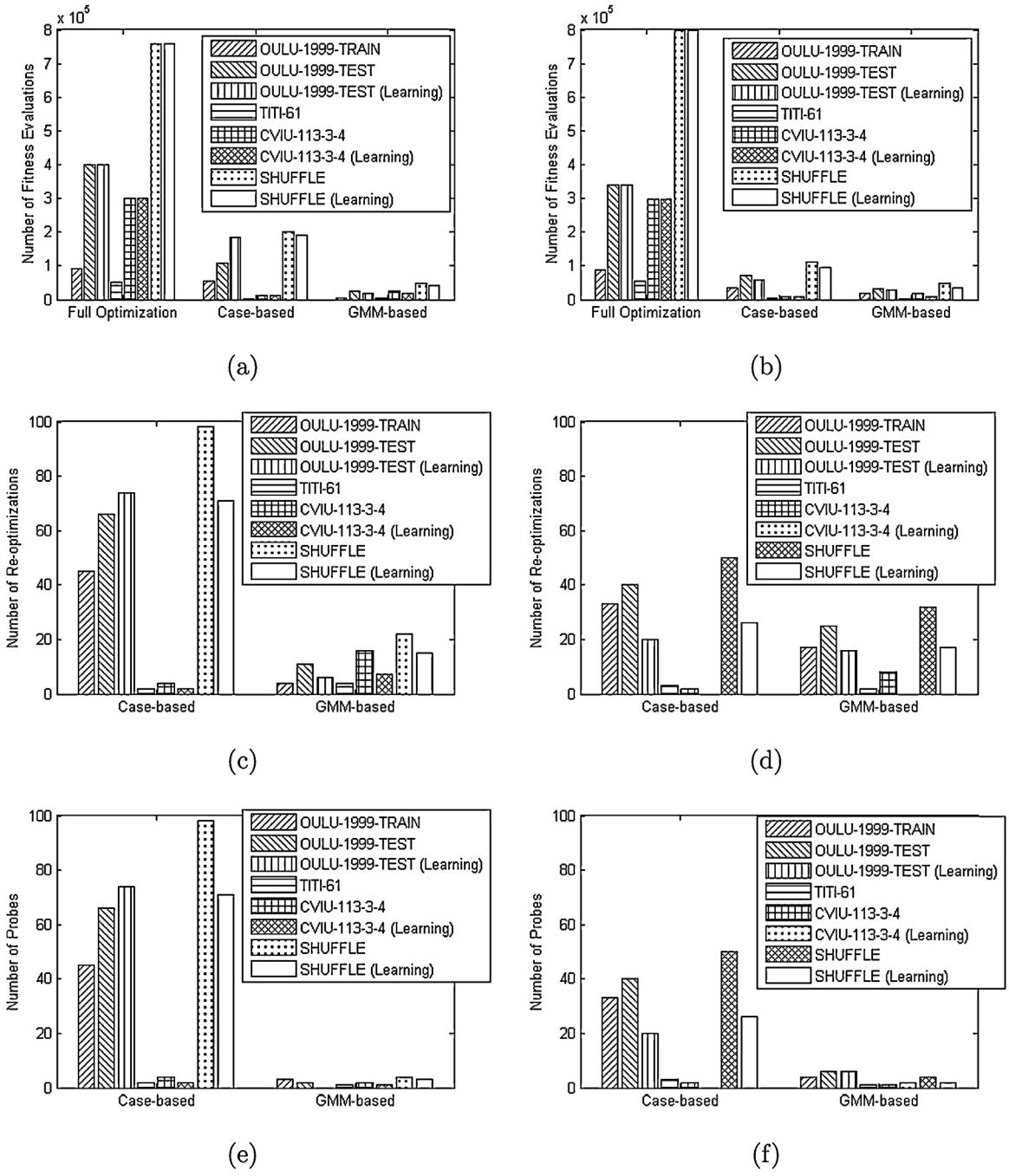

Fig. 6. Comparison of computational and memory burden for the different approaches. (a) Number of fitness evaluations, no attack. (b) Number of fitness evaluations, cropping 1%. (c) Number of re-optimizations, no attack. (d) Number of re-optimizations, cropping 1%. (d) Number of probes, no attack. (e) Number of probes, cropping 1%.

quite well. Then, the merge operator was de-activated and the same experiment was repeated. This time the second run resulted in 3 re-optimizations. It can be said that such increase in the number of re-optimizations for the merge operator was the result of the smaller bias of that approach. That is, the merge operator, as observed in the first memorization experiments, results in probes that are less tuned to specific images (more general).

5.3.5. Other attacks

It is possible to observe in Table 5 that the computational cost proposed approach is not considerably affected by an increase in the attack level or by a different removal attack such as salt & pepper (S&P).

Regarding the watermarking performance (Table 6), the behavior was similar to the cases of no attack and cropping of 1%: a slight variation when compared to full optimization, but largely offset by gains in computational burden.

5.3.6. Adaptation performance

Memory adaptability is another important aspect in the given scenario. It is reasonable to consider that in the course of its normal operation, the set of attacks an intelligent watermarking system must deal with is expected to change and that the memory should be capable to adapt to such change. In such case, the system must avoid recalling solutions that result in poor watermarking performance. To validate this, we performed a memory adaptation

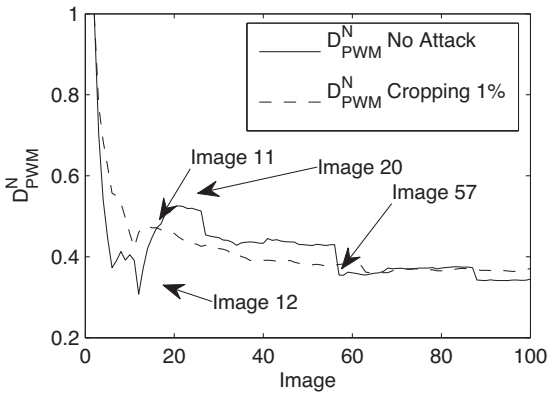

Fig. 7. LTM diversity (OULU-1999-TRAIN).

experiment (Fig. 16). In this experiment, the GMM-based approach was first applied to the OULU-1999-TRAIN stream with no attack. Then, using the resulting memory as a starting point, the same approach was applied to the same stream but with cropping of 2%. Next, the same procedure was repeated (also using the previous memory as a starting point) but now with salt & pepper 0.02. Finally, the proposed approach was applied to the OULU-1999-TEST database in four different scenarios: using the memory of previous case as a starting point but now with (I) no attack; (II) cropping 2%; (III) salt & pepper 0.02; (IV) randomly chosen attacks (salt & pepper 0.02, no attack, cropping 2%) for each image; (IVa) not using previous memory (no learning) with random attacks. In all cases the confidence level was set to 0.8, as adaptation requires a more restrictive confidence level.

It is interesting to observe that the results obtained in the adaptation experiments (Table 7) are similar to previously

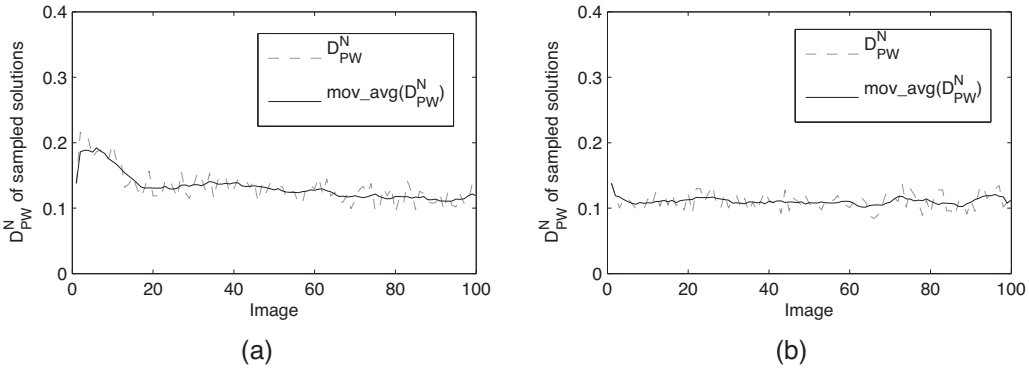

Fig. 8. Diversity of 2000 solutions sampled uniformly for all probes (D^N_{PW}) including moving average with window size 10 ($mov\_avg(D^N_{PW})$) for OULU-1999-TRAIN stream. (a) No attack. (b) Cropping 1%.

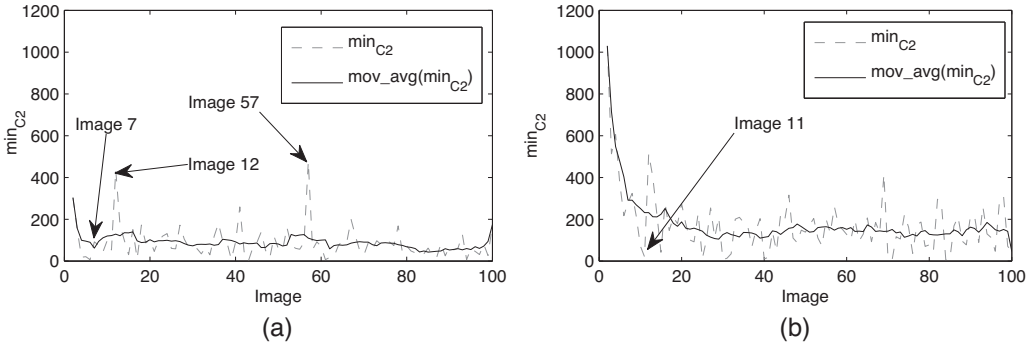

Fig. 9. Minimum $C2$ distances between new probes and probes already in the memory (min_{C2}) for OULU-1999-TRAIN stream. Moving average of min_{C2} with window size 10 ($mov\_avg(min_{C2})$) is also depicted. (a) No attack. (b) Cropping 1%.

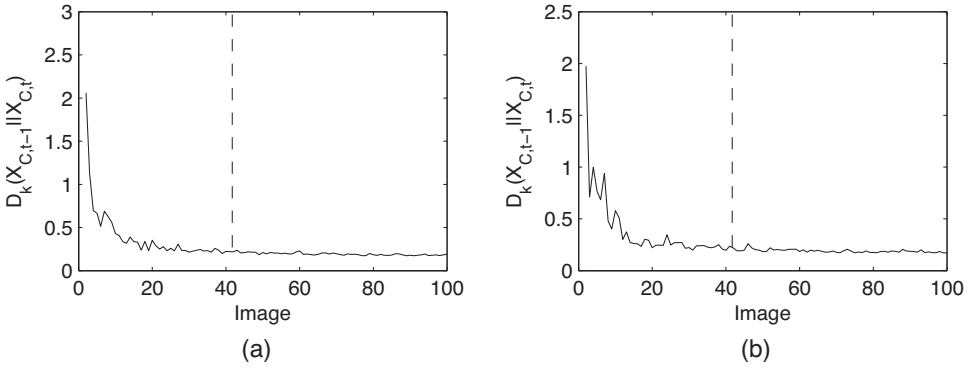

Fig. 10. Kullback–Leibler divergence between cumulative sets of particles at instants t and $t-1$. (a) No attack. (b) Cropping 1%.

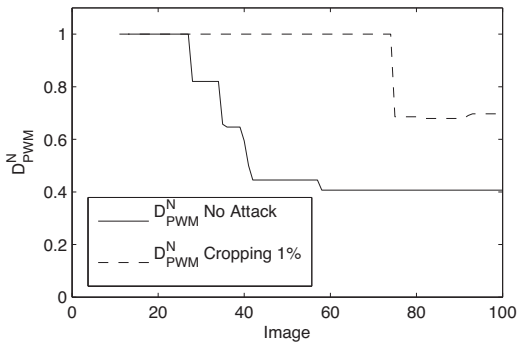

Fig. 11. LTM diversity (OULU-1999-TRAIN, with memory management).

presented results. The slight degradation in computational burden was mainly due to the more restrictive confidence level. For example, OULU-1999-TRAIN with no attack resulted in 92.9% decrease with confidence level 0.95 (Table 3) versus 84.8% with confidence level 0.8 (Table 7). However watermarking performance of both was very similar (Table 4). The same happened for the simulations involving cropping 2% and salt & pepper 0.02 (Tables 5 and 6). Regarding the OULU-1999-TEST stream, the computational performance of cases I, II, III and IV was close to that of no learning for the previous simulations (Tables 3 and 5) with an equivalent watermarking performance (Tables 4 and 6). It is worth noticing that in Table 7, for the random attacks, the use of a training sequence (IV) resulted in a considerable decrease in computational burden when compared to no training (IVa). It is also worth

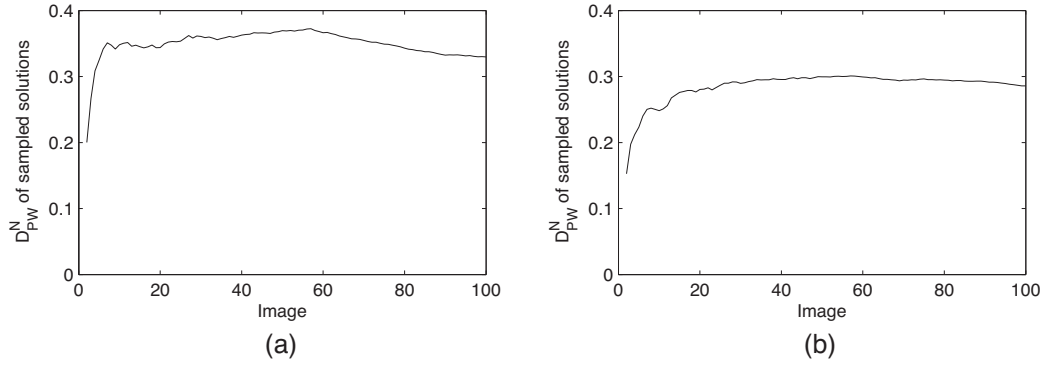

Fig. 12. Diversity of 2000 solutions sampled uniformly for all probes (D^N_{PW}) for OULU-1999-TRAIN stream (with memory management). (a) No attack. (b) Cropping 1%.

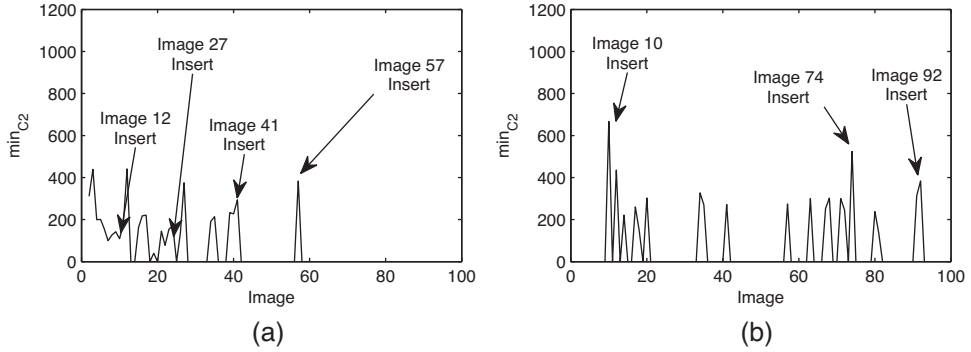

Fig. 13. Minimum *C*2 distance between new probes and probes already in the memory (min_{C2}) for OULU-1999-TRAIN stream (with memory management). (a) No attack. (b) Cropping 1%.

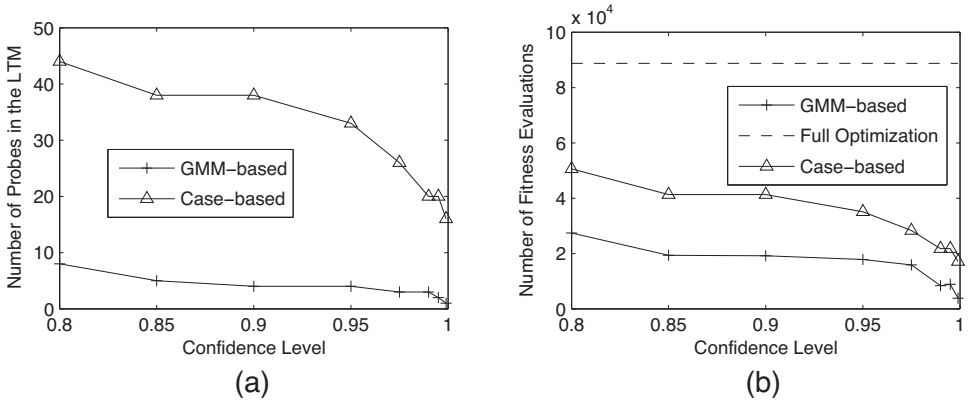

Fig. 14. Number of LTM probes produced by the case-based and GMM-based techniques as a function of confidence level for the OULU-1999-TRAIN with cropping of 1%. (a) LTM size. (b) Number of fitness evaluations.

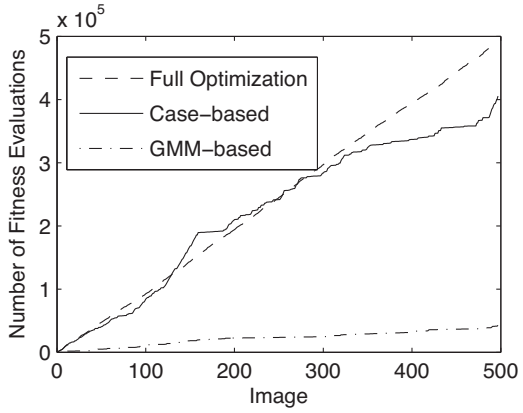

Fig. 15. Cumulative number of fitness evaluations for the case-based, GMM-based memory scheme and full optimization for OULU-1999-TEST (learning), no attack, confidence level of 0.8.

noticing that the OULU-1999-TEST simulations with learning resulted few inserted probes when compared to OULU-1999-TRAIN simulations. This demonstrates that even in such a challenging scenario involving changes in the set of attacks, the proposed approach can learn how to adapt to such changes.

5.4. Scenario B – optimization of homogeneous streams of bi-tonal images using memory-based DPSO versus full PSO

In general, for the homogeneous image streams, the computational burden performance of the GMM-based approach is slightly worse than what has been reported for the case-based approach in [22] as it required more re-optimizations. Yet, adjusted for the size of the image streams, the number of re-optimizations for the GMM-based approach in this scenario is consistent with that obtained for the heterogeneous image streams while for the case-based approach, there is a huge discrepancy between the performances for the heterogeneous and homogeneous streams. That is, since a case-based probe is over-fit to a particular optimization problem, it tends to perform better than the GMM-based approach when the stream of optimization problems is homogeneous. In the GMM-based approach by its way, a probe is less biased to a specific optimization problem and can cope better with variations in a more heterogeneous image stream. The watermarking performance (mainly watermark robustness) of the GMM-based approach is considerably better than that of the case-based approach.

5.5. Scenario C – optimization of unconstrained (homogeneous/heterogeneous) streams of bi-tonal images using memory-based DPSO versus full PSO

The behavior of the proposed technique when compared to case-based for scenario C was quite similar to that observed for scenario A. The proposed technique resulted in a decrease in computational burden at an equivalent watermarking performance. The use of a training sequence of images allowed a further decrease also with little impact on watermarking performance.

5.6. Discussion

The GMM-based approach was evaluated in three main scenarios – intelligent watermarking of homogeneous, heterogeneous image streams, and a mix of both, respectively. It is possible to observe through the simulation results that for the heterogeneous image streams, the proposed memory scheme results in less re-optimizations than the case-based scheme but at nearly the same watermarking performance. Both, the fidelity of the watermarked image and the detection rate of the robust and fragile watermarks are comparable to those of full optimization. The main reason is that by using particle history data, it is possible to sample a larger region of the fitness landscape but in a targeted manner. It can be said thus that the case-based mechanism is sensitive to the distribution of particles in the end of the optimization process. It was also observed that the proposed technique allows a significant decrease in computational burden when compared to full optimization in both, homogeneous and heterogeneous image streams. More specifically, the number of fitness evaluations per image was above 800 for the best scenario of Full Optimization which is unfeasible for practical applications as it involves more than 800 embedding and detection operations per image. This number was decreased to 67 in the worst case for the proposed approach with learning.

Table 5
Computational cost performance. *AFPI* is the average number of fitness evaluations per image where the mean μ and standard deviation σ are presented as $\mu(\sigma)$. F_{Evals} is the cumulative number of fitness evaluations required to optimize the whole stream and *DFE* is the decrease in the number of fitness evaluations compared to full optimization.

Attack	Database	Learning	Full PSO		Case-based			GMM-based		
			AFPI	F_{Evals}	*AFPI*	F_{Evals}	*DFE* (%)	*AFPI*	F_{Evals}	*DFE* (%)
Cropping 2%	OULU-1999-TRAIN	No	860 (335)	86,040	185 (382)	18,520	78.5	72 (187)	7240	91.6
Cropping 2%	OULU-1999-TEST	No	828 (309)	328,900	140 (342)	55,740	83.1	64 (179)	25,560	92.2
Cropping 2%	OULU-1999-TEST	Yes	828 (309)	328,900	113 (290)	44,940	86.3	50 (150)	19,800	94
S&P 0.02	OULU-1999-TRAIN	No	893 (354)	89,280	462 (507)	46,220	48.2	163 (360)	16,320	81.7
S&P 0.02	OULU-1999-TEST	No	978 (379)	388,220	253 (433)	100,580	74.1	92 (281)	36,360	90.6
S&P 0.02	OULU-1999-TEST	Yes	978 (379)	388,220	157 (321)	62,200	84	42 (133)	16,560	95.7

Table 6
Watermarking performance. Here, † is the *DRDM*, ‡ is the *BCR* robust, § is the *BCR* fragile. For all values, the mean μ and standard deviation σ per image are presented in the following form: $\mu(\sigma)$. *DRDM* is presented with two decimal points and *BCR* is presented in percentage (%) with one decimal point.

Attack	Database	Learning	Full PSO			Case-based			GMM-based		
			†	‡	§	†	‡	§	†	‡	§
Cropping 2%	OULU-1999-TRAIN	No	0.04 (0.05)	98.2 (2.7)	99.9 (0.4)	0.04 (0.05)	98 (3.1)	99.9 (0.5)	0.04 (0.06)	97.1 (3.8)	99.8 (0.6)
Cropping 2%	OULU-1999-TEST	No	0.04 (0.04)	98 (3)	99.8 (0.7)	0.03 (0.04)	97 (4.5)	99.6 (1.4)	0.04 (0.04)	95.4 (5.7)	99.3 (2)
Cropping 2%	OULU-1999-TEST	Yes	0.04 (0.04)	98 (3)	99.8 (0.7)	0.04 (0.05)	97.1 (4.4)	99.6 (1.2)	0.04 (0.05)	94.7 (6.4)	99.1 (1.9)
S&P 0.02	OULU-1999-TRAIN	No	0.03 (0.03)	97.9 (2.6)	99.7 (0.5)	0.03 (0.03)	97.9 (3.1)	99.7 (0.5)	0.03 (0.03)	97.1 (4.3)	99.3 (1.3)
S&P 0.02	OULU-1999-TEST	No	0.03 (0.04)	98 (2.4)	99.6 (0.6)	0.02 (0.03)	97.2 (3.3)	98.9 (1.4)	0.03 (0.04)	97.2 (3.6)	99.4 (1)
S&P 0.02	OULU-1999-TEST	Yes	0.03 (0.04)	98 (2.4)	99.6 (0.6)	0.03 (0.04)	97.1 (3.6)	99.4 (1.1)	0.03 (0.04)	97.1 (0.04)	99.2 (1.2)

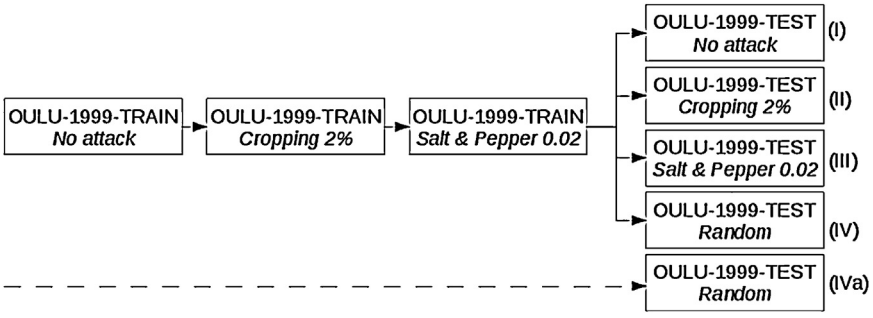

Fig. 16. Memory adaptation experiment.

For the heterogeneous scenario, a memory fill up experiment was performed and it showed that as new images are fed into the system, the amount of novelty brought by these images decreases considerably for the first third of the image stream (OULU-1999-TRAIN) and then stabilizes. Consequently, the lack of a proper memory management mechanism results in redundant probes which impair the computational performance of a unsuccessful recall (since all LTM probes need to be tested before re-optimization is triggered). At the same time, when insert operations are employed indiscriminately, the resulting memory becomes quite noneffective. Moreover, the probing capability of the memory is negatively affected as the diversity of sampling solutions decrease.

The adaptive memory management experiments involving heterogeneous streams showed that the proposed approach not only decreases the computational burden of intelligent watermarking (when compared to the case-based approach) but with practically no impact on watermarking performance. And more important than that, an analysis of memory dynamics showed that in the proposed mechanism, the memory space is used in a more effective manner as insert operations are employed sparingly. Moreover, it has been demonstrated that the frequency of memory update operations are in a par with the amount of novelty brought by the new problems. This is more in tune with the formulation of incremental learning seen in [46] as with this combination of merge and insert operations (1) none of the inserted probes will contradict the data processed up to that point and (2) through the use of a merge operator each intermediate hypothesis is maintained as long as it is consistent with the data seen. That is, insert only occurs when the new problem represents new knowledge to the memory. These experiments also showed that by maintaining the distance between LTM probes high, it is possible to improve the diversity of sampled solutions which allows a better probing capability. Analysis of memory dynamics showed that the proposed memory management mechanism helps to avoid inserting probes that do not bring novelty to the LTM. For example, both the pairwise distance between probes and the minimum distance between new probes and probes in the memory are increased considerably when the memory management scheme is employed. This shows that the proposed scheme minimizes redundancy in the LTM. The sampling diversity was also increased which means that despite smaller memory and computational burden, the proposed memory management scheme resulted in probes that cover a significant area of the fitness landscape.

Memorization experiments demonstrated that the GMM memory can learn considerably well the stream of optimization problems. First because density estimate of solutions in the optimization space offer a reliable approximation of the fitness landscape and second because the merge operator results in less biased probes that generalize well to new problems, as observed in the experiments involving multiple recalls for a same image. These experiments also demonstrated that the probe is subject to a trade-off between memorization and generalization (bias/variance trade-off). This trade-off can be modified when necessary (e.g. in an application involving more dynamism in the stream of document images) by adjusting the confidence level of the change detection mechanism. And yet, memorization can be further improved (when necessary) by de-activating the merge operator (not recommended for heterogeneous streams).

It was possible to observe in experiments with higher cropping intensity and salt & pepper attack that the results observed for the cropping 1% and no attack are applicable to other types of removal attacks. The conclusion to be drawn here is that as long as robustness against a given attack can be attained through optimization of embedding parameters and considering that the stream of images contains recurrent (similar images), the proposed GMM-based approach is expected to result in a smaller computational burden compared to full optimization, with an equivalent watermarking performance. The reason is that the use of GMM results in a precise approximation of the stream of optimization problems. The limitation of the proposed approach is that its watermarking performance is bounded by the watermarking performance of full optimization. For example, in the baseline watermarking system, robustness against geometric attacks cannot be attained through

Table 7
Adaptation performance. *DFE* is the decrease in the number of fitness evaluations compared to full optimization, † is the *DRDM*, ‡ is the *BCR* robust, § is the *BCR* fragile. For all values, the mean μ and standard deviation σ per image are presented in the following form: $\mu(\sigma)$. *DRDM* is presented with two decimal points and *BCR* is presented in percentage (%) with one decimal point.

Attack	Database	Re-optimizations	Inserted probes	*DFE* (%)	†	‡	§
No attack	OULU-1999-TRAIN	13	3	84.8	0 (0)	100 (0)	100 (0)
Cropping 2%	OULU-1999-TRAIN	13	3	84.3	0.04 (0.05)	97 (3.6)	99.7 (1)
S&P 0.02	OULU-1999-TRAIN	12	1	79.4	0.03 (0.04)	97.3 (3.6)	99.5 (1.2)
No attack (I)	OULU-1999-TEST	20	1	88.9	0.01 (0.02)	99.9 (0.01)	99.9 (0.01)
Cropping 2% (II)	OULU-1999-TEST	15	2	91.4	0.04 (0.05)	93.3 (0.06)	99.1 (0.02)
S&P 0.02 (III)	OULU-1999-TEST	29	5	87.4	0.04 (0.04)	97.1 (3.7)	99.3 (1.1)
Random (IV)	OULU-1999-TEST	31	4	85.5	0.03 (0.04)	97.3 (4.3)	99.4 (1.4)
Random (IVa)	OULU-1999-TEST	65	8	76.3	0.03 (0.04)	97.6 (3.7)	99.6 (1)

manipulation of embedding parameters (instead, it is attained through the use of reference marks [3]). Therefore, the GMM-based approach also will not tackle robustness against such type of attack.

In the adaptation experiments, it was possible to observe that in applications involving high dynamism in the stream of problems (e.g. changing attacks), the proposed approach can adapt well, with a relatively small computational burden. The reason is that the memory of GMMs results in a more precise representation of the stream of optimization problems which allows a better change detection capability (as observed in the memorization experiments as well). These experiments also allow us to draw some guidelines regarding the choice of confidence level. In situations involving high variability (like changing attacks), a more restrictive confidence level is to be preferred. Otherwise, a more relaxed confidence level is preferred (since it should result in less re-optimizations).

It was possible to observe that the GMM-based approach is not only less expensive than the case-based approach (for the heterogeneous streams) but the gains in computational burden are more consistent, that is, are quite similar across different scenarios. Another advantage of the GMM-based approach is that it has a smaller memory footprint than the case-based approach. Not only because the mixture model offers a more compact data representation but also because in the GMM-based approach, the number of probes is considerably smaller than for the case-based approach. It is important to mention that although the LTM size is limited for the GMM-based approach, such limit was not achieved for the chosen confidence level. It is worth mentioning that the decrease in the number of fitness evaluations is proportional to the number of probes, the number of re-sampled particles, the frequency of recall and the number of fitness evaluations required in full optimization. Since the number of fitness evaluations required in full optimization varies across the images in a stream the possible boost obtained by replacing full optimization by memory recall is image-dependent. It is also important noticing that for a limited memory size, the number of fitness evaluations in full optimization tends to be considerably larger than that of a successful recall. Therefore, the impact of a case of re-optimization in the number of fitness evaluations tends to be exacerbated in small databases.

In general these experiments show that by estimating mixture models of swarm solutions and keeping a memory of these models with the use an appropriate memory management strategy it is possible to build a general model of a stream of optimization problems in an intelligent watermarking application using a set of learning images and then decrease significantly the cost of intelligent watermarking with little impact on watermarking performance. This general model is more adaptive than that created by the case-based approach and is thus more appropriate for applications where the stream of images to be optimized is heterogeneous.

6. Conclusion

In this paper an intelligent watermarking technique based on dynamic particle swarm optimization (DPSO) is proposed. The adaptive memory relies on sampled solutions from GMMs of previous optimization problems and their respective global best solutions in order to (1) compare how similar future optimization problems are to those previously seen and (2) provide alternative solutions in cases where the similarity between problems is small, avoiding re-optimization. Its memory management strategy aimed at tackling two main issues observed in previous experiments. The first was to avoid redundancy in the LTM while the second was to allow the memory to adapt quickly to new optimization problems.

Although the use of density models in evolutionary computing is not new, the use of models based on phenotypic and genotypic data of candidate solutions is novel. Moreover, while in the EDA literature most authors rely on high evaluation solutions in order to estimate these models, in the proposed approach we rely on all solutions in order to build a more comprehensive model of the fitness landscape. It was demonstrated empirically that this more comprehensive model allows a more precise match between previously seen and new optimization problems. Another contribution of the proposed technique was the inception of a management approach that allows the memory to incrementally learn new trends on the stream of optimization problems while limiting memory footprint.

Experimental results demonstrate that replacing memory solutions by density estimates of swarm solutions result not only in less memory burden but in a more precise probing mechanism which resulted in a decrease in the number of re-optimizations with little impact in watermarking performance. Since the proposed approach allows an incremental learning of optimization problems, the use of a learning stream of images allowed decreasing computational cost while improving precision altogether. In such case, a decrease of 97.7% in the number of fitness evaluations was obtained for heterogeneous image streams (when compared to full optimization) through the use of a learning stream of images. Such improvement in computational performance was higher than that of no learning. It was also possible to observe that the GMM memory allows a more precise representation of the fitness landscape. This results in better probing of the fitness landscape (compared to a memory of static solutions) which helps to avoid false positive errors (recalling wrong probes which would decrease the watermarking performance). Such memory makes possible for example, changing the attack employed on the DPSO module, without any further need of human intervention in what regards memory management.

As a future work we propose a deeper study on each of the main modules of the proposed technique and a comparison study with alternative approaches for these modules. We also propose validating the GMM-based approach using a larger image stream.

References

[1] I. Cox, M. Miller, J. Bloom, Digital Watermarking, Morgan Kaufmann Publishers, San Francisco, CA, USA, 2002.
[2] I.J. Cox, J. Kilian, T. Leighton, T. Shamoon, A secure, robust watermark for multimedia, in: Workshop on Information Hiding, 1996, pp. 1–16.
[3] M. Wu, B. Liu, Data hiding in binary image for authentication and annotation, IEEE Transactions on Multimedia 6 (4) (2004) 528–538.
[4] J.H. Holland, Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, USA, 1992.
[5] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks, Perth, Australia, 1995.
[6] E. Vellasques, E. Granger, R. Sabourin, Intelligent watermarking systems: a survey, in: Handbook of Pattern Recognition and Computer Vision, 4th ed., World Scientific Review, Singapore, 2010, pp. 687–724.
[7] T.E. Areef, H.S. Heniedy, O.M.O. Mansour, Optimal transform domain watermark embedding via genetic algorithms, in: ITI 3rd International Conference on Information and Communications Technology (ICICT), 2005, pp. 607–617.
[8] M. Arsalan, S.A. Malik, A. Khan, Intelligent threshold selection for reversible watermarking of medical images, in: GECCO '10: Proceedings of the Genetic and Evolutionary Computation Conference, ACM, Portland, OR, USA, 2010, pp. 1909–1914.
[9] M. Arsalan, S.A. Malik, A. Khan, Intelligent reversible watermarking in integer wavelet domain for medical images, Journal of Systems and Software 85 (4) (2012) 883–894.
[10] C. Chen, C. Lin, A GA-based nearly optimal image authentication approach, International Journal of Innovative Computing, Information and Control 3 (3) (2007) 631–640.
[11] R. Ji, H. Yao, S. Liu, L. Wang, Genetic algorithm based optimal block mapping method for LSB substitution, in: International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2006, pp. 215–218.
[12] A. Khan, A.M. Mirza, Genetic perceptual shaping: utilizing cover image and conceivable attack information during watermark embedding, Information Fusion 8 (4) (2007) 354–365.
[13] A. Khan, S.F. Tahir, A. Majid, T. Choi, Machine learning based adaptive watermark decoding in view of anticipated attack, Pattern Recognition 41 (8) (2008) 2594–2610.

[14] P. Kumsawat, K. Attakitmongcol, A. Srikaew, A new approach for optimization in image watermarking by using genetic algorithms, IEEE Transactions on Signal Processing 53 (12) (2005) 4707–4719.
[15] C. Shieh, H. Huang, F. Wang, J. Pan, Genetic watermarking based on transform-domain techniques, Pattern Recognition 37 (3) (2004) 555–565.
[16] F.Y. Shih, Y. Wu, Enhancement of image watermark retrieval based on genetic algorithms, Journal of Visual Communication and Image Representation 16 (2) (2004) 115–133.
[17] J. Pan, H. Huang, L. Jain, Genetic watermarking on spatial domain, in: Intelligent Watermarking Techniques, World Scientific Co., Singapore, 2004.
[18] I. Usman, A. Khan, BCH coding and intelligent watermark embedding: employing both frequency and strength selection, Applied Soft Computing 10 (1) (2010) 332–343.
[19] Z. Wei, H. Li, J. Dai, S. Wang, Image watermarking based on genetic algorithm, in: IEEE International Conference on Multimedia and Expo (ICME), 2006, pp. 1117–1120.
[20] Y. Wu, F.Y. Shih, Genetic algorithm based methodology for breaking the steganalytic systems, IEEE Transactions on Systems, Man and Cybernetics, Part B 36 (1) (2006) 24–31.
[21] E. Vellasques, R. Sabourin, E. Granger, Intelligent watermarking of document images as a dynamic optimization problem, in: International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2010.
[22] E. Vellasques, R. Sabourin, E. Granger, A high throughput system for intelligent watermarking of bi-tonal images, Applied Soft Computing 11 (8) (2011) 5215–5229.
[23] M. Farina, K. Deb, P. Amato, Dynamic multiobjective optimization problems: test cases, approximations, and applications, IEEE Transactions on Evolutionary Computation 8 (5) (2004) 425–442.
[24] S. Yang, X. Yao, Population-based incremental learning with associative memory for dynamic environments, IEEE Transactions on Evolutionary Computation 12 (5) (2008) 542–561.
[25] M. Pelikan, D.E. Goldberg, F.G. Lobo, A survey of optimization by building and using probabilistic models, Computational Optimization and Applications 21 (1) (2002) 5–20.
[26] E. Muharemagic, Adaptive two-level watermarking for binary document images, Ph.D. Thesis, Florida Atlantic University, December 2004.
[27] S. Voloshynovskiy, S. Pereira, T. Pun, J. Eggers, J. Su, Attacks on digital watermarks: classification, estimation based attacks, and benchmarks, IEEE Communications Magazine 39 (8) (2001) 118–126.
[28] I. Awan, S.A.M. Gilani, S.A. Shah, Utilization of maximum data hiding capacity in object-based text document authentication, in: International Conference on Intelligent Information Hiding and Multimedia (IIH-MSP), Washington, DC, USA, 2006, pp. 597–600.
[29] A. Ho, N. Puhan, P. Marziliano, A. Makur, Y. Guan, Perception based binary image watermarking, in: International Symposium on Circuits and Systems (ICAS), vol. 2, 2004, pp. 37–40.
[30] H. Lu, X. Shi, Y.Q. Shi, A.C. Kot, L. Chen, Watermark embedding in DC components of DCT for binary images, in: IEEE Workshop on Multimedia Signal Processing, 2002, pp. 300–303.
[31] H. Pan, Y. Chen, Y. Tseng, A secure data hiding scheme for two-color images, in: IEEE Symposium on Computers and Communication, 2000, pp. 750–755.
[32] H. Yang, A.C. Kot, Binary image authentication with tampering localization by embedding cryptographic signature and block identifier, IEEE Signal Processing Letters 13 (December (12)) (2006) 741–744.
[33] F. Petitcolas, R. Anderson, M. Kuhn, Attacks on copyright marking systems, in: Proceedings of the Second International Workshop on Information Hiding, Springer-Verlag, London, UK, 1998, pp. 218–238.
[34] S. Marchand-Maillet, Y.M. Sharaiha, Binary Digital Image Processing – A Discrete Approach, Academic Press, New York, NY, USA, 2000.
[35] Y. Collette, P. Siarry, On the sensitivity of aggregative multiobjective optimization methods, in: CIT, vol. 16, no. 1, 2008, pp. 1–13.
[36] H. Lu, A.C. Kot, Y.Q. Shi, Distance-reciprocal distortion measure for binary document images, IEEE Signal Processing Letters 11 (2) (2004) 228–231.
[37] T. Blackwell, Particle swarm optimization in dynamic environments, in: Evolutionary Computation in Dynamic and Uncertain Environments, Springer, Berlin, Germany, 2007.
[38] A. Carlisle, G. Dozier, Tracking changing extrema with adaptive particle swarm optimizer, in: Proceedings of the 5th Biannual World Automation Congress, 2002, vol. 13, 2002, pp. 265–270.
[39] J. Branke, Memory enhanced evolutionary algorithms for changing optimization problems, in: IEEE Congress on Evolutionary Computation (CEC), 1999, pp. 1875–1882.
[40] H. Wang, D. Wang, S. Yang, Triggered memory-based swarm optimization in dynamic environments, in: EvoWorkshops, 2007, pp. 637–646.
[41] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley-Interscience Publication, New York, NY, USA, 2000.
[42] D.M.J. Tax, R.P.W. Duin, Support vector data description, Machine Learning 54 (1) (2004) 45–66.
[43] M. Markou, S. Singh, Novelty detection: a review-part 1: statistical approaches, Signal Processing 83 (12) (2003) 2481–2497.
[44] N. Japkowicz, C. Myers, M. Gluck, A novelty detection approach to classification, in: Proceedings of the Fourteenth Joint Conference on Artificial Intelligence, 1995, pp. 518–523.
[45] J. Ma, S. Perkins, Online novelty detection on temporal sequences, in: KDD '03: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2003, pp. 613–618.
[46] S. Jain, S. Lange, S. Zilles, Towards a better understanding of incremental learning, Algorithmic Learning Theory 4264 (10) (2006) 169–183.
[47] J.H. Gennari, P. Langley, D. Fisher, Models of incremental concept formation, Journal of Artificial Intelligence 40 (1989) 11–61.
[48] J. Wu, X.S. Hua, B. Zhang, Tracking concept drifting with Gaussian mixture model, in: Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, 2005, pp. 1562–1570.
[49] K. Yamanishi, J. Takeuchi, G. Williams, P. Milne, On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms, in: KDD '00: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Boston, MA, USA, 2000, pp. 320–324.
[50] M. Markou, S. Singh, Novelty detection: a review-part 2: neural network based approaches, Signal Processing 83 (12) (2003) 2499–2521.
[51] NIST/SEMATECH e-Handbook of Statistical Methods, March 2010. http://www.itl.nist.gov/div898/handbook/
[52] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Computing Surveys 31 (3) (1999) 264–323.
[53] M.A.T. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2000) 381–396.
[54] K.B. Amd, I.E. Lagaris, Split-Merge Incremental Learning (SMILE) of mixture models, in: ICANN, 2007, pp. 291–300.
[55] G. Sfikas, C. Constantinopoulos, A. Likas, N.P. Galatsanos, An analytic distance metric for Gaussian mixture models with application in image retrieval, in: Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and their Applications – Volume Part II, ICANN'05, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 835–840.
[56] C. Hennig, Methods for merging Gaussian mixture components, Advanced Data Analysis and Classification 4 (2010) 3–34.
[57] N. Ueda, R. Nakano, Z. Ghahramani, G.E. Hinton, SMEM algorithm for mixture models, Neural Computation 12 (9) (2000) 2109–2128.
[58] J. Sauvola, H. Kauniskangas, Media Team Document Database II, a CD-ROM Collection of Document Images, University of Oulu, Finland, 1999.
[59] G. Corriveau, R. Guibault, A. Tahan, R. Sabourin, Review and study of genotypical diversity measures for real-coded representations, IEEE Transactions on Evolutionary Computation (2012), http://dx.doi.org/10.1109/TEVC.2011.2170075m in press.
[60] F. Pérez-Cruz, Kullback–Leibler divergence estimation of continuous distributions, in: IEEE International Symposium on Information Theory (ISIT), Toronto, Canada, 2008.