Expert Systems with Applications 40 (2013) 5240–5259

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

DS-DPSO: A dual surrogate approach for intelligent watermarking of bi-tonal document image streams

Eduardo Vellasques *, Robert Sabourin, Eric Granger

Laboratoire d'imagerie, de vision et d'intelligence artificielle, École de Technologie Supérieure, Université du Québec, Montreal, Canada

ARTICLE INFO

Keywords:
Intelligent Watermarking
Evolutionary Computation
Particle Swarm Optimization
Surrogate-based Optimization
Gaussian Mixture Models

ABSTRACT

Intelligent watermarking (IW) techniques employ population-based evolutionary computing in order to optimize embedding parameters that trade-off between watermark robustness and image quality for digital watermarking systems. Recent advances indicate that it is possible to decrease the computational burden of IW techniques in scenarios involving long heterogeneous streams of bi-tonal document images by recalling embedding parameters (solutions) from a memory based on Gaussian Mixture Model (GMM) representation of optimization problems. This representation can provide ready-to-use solutions for similar optimization problem instances, avoiding the need for a costly re-optimization process. In this paper, a dual surrogate dynamic Particle Swarm Optimization (DS-DPSO) approach is proposed which employs a memory of GMMs in regression mode in order to decrease the cost of re-optimization for heterogeneous bi-tonal image streams. This approach is applied within a four level search for near-optimal solutions, with increasing computational burden and precision. Following previous research, the first two levels use GMM re-sampling to recall solutions for recurring problems, allowing to manage streams of heterogeneous images. Then, if embedding parameters of an image require a significant adaptation, the third level is activated. This optimization level relies on an off-line surrogate, using Gaussian Mixture Regression (GMR), in order to replace costly fitness evaluations during optimization. The final level also performs optimization, but GMR is employed as a costlier on-line surrogate in a worst-case scenario and provides a safeguard to the IW system. Experimental validation were performed on the OULU image data set, featuring heterogeneous image streams with a varying levels of attacks. In this scenario, the DS-DPSO approach has been shown to provide comparable level of watermarking performance with a 93% decline in computational cost compared to full re-optimization. Indeed, when significant parameter adaptation is required, fitness evaluations may be replaced with GMR.

1. Introduction

The decreasing costs of data transmission and storage provided numerous opportunities for sharing multimedia documents like images. This has led to the creation of a digital economy with new services that are available 24 h a day, 7 days a week, around the globe. Individuals and businesses depend more and more on sharing important documents which raises serious privacy concerns. Enforcing the security of document images is an important issue. Cryptography can solve part of this issue. However, specially with multimedia documents like images, the protection allowed by cryptography vanishes as the data has been decrypted. Digital watermarking (Cox et al., 2002) which consists of embedding image-related secret data through the manipulation of pixel values in an imperceptible manner, allows another layer of protection. Most importantly, the protection mechanism provided by digital watermarking follows the image even when it is inadvertently distributed or tampered. Enforcing the security of bi-tonal document images poses an additional challenge as bi-tonal images have lower embedding capacity and the manipulation of bi-tonal pixels is more prone to result in visual artifacts.

Digital watermarking has become an active area of research in recent years. Because of its nature, watermarking systems are subject to attacks by hackers (Voloshynovskiy et al., 2001). Robustness against attacks always comes at the cost of degradation on imperceptibility (Cox et al., 1996). Many watermarking techniques allow adjusting the trade-off between robustness and quality through the manipulation of embedding parameters. The optimal trade-off and the corresponding values vary from one image to another. To make matters worse, security requirements also vary across applications.

In a typical use case, a robust watermark is added to a document image and then a fragile watermark is added to the top of it. The robust watermark can contain, for example, information

* Corresponding author.
E-mail addresses: evellasques@livia.etsmtl.ca (E. Vellasques), robert.sabourin@etsmtl.ca (R. Sabourin), eric.granger@etsmtl.ca (E. Granger).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.03.021

about the ownership of that document. For this reason it must survive intentional and unintentional image processing operations. The fragile watermark by its way, is easily destroyed which allows detecting that tampering has occurred. Therefore, compared to other security mechanisms such as cryptography, digital watermarking allows for a concealed type of protection. And more important, it will always follow the image (the protection provided by cryptography vanishes when the image is decrypted). The main limitation of digital watermarking is that the robustness against attacks and the corresponding image quality trade-off are defined through heuristic parameters. A poor choice of parameter can result in a robust watermark that can be easily removed. Therefore, these parameters must be adjusted for different sets of attacks, according to the embedding capacity of each image.

Manual adjustment of digital watermarking parameters is infeasible in practical applications. Evolutionary computing (EC) techniques such as Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995) and Genetic Algorithms (Holland, 1992) have been employed in order to find embedding parameters that optimize the trade-off between image quality and watermark robustness for each image and set of attacks (Vellasques et al., 2010). In EC, a population of candidate solutions is evolved through a certain number of generations, and guided by an objective function. In intelligent watermarking (IW), objective functions are usually a combination of image quality and watermark robustness. The fitness of each candidate solution is evaluated at each generation. Each fitness evaluation requires one or more embedding, detection and attack (image processing) operations which is prohibitively expensive in industrial applications.

Recent efforts to decrease the computational cost of IW techniques for streams of document images is promising. In the Dynamic PSO (DPSO) system proposed in Vellasques et al. (2011), IW of homogeneous streams of bi-tonal document images (or problems) was formulated as a special type of dynamic optimization problem (DOP[1]). In this special formulation of DOP, a stream of document images corresponds to a stream of recurring optimization problems. A change detection mechanism assigns case-based solutions of previously-seen problem instances (associated with previous document images) to new similar problem instances (associated with new images). This significantly reduced the number of costly re-optimization operations, allowing for a significant decrease in computational burden. In the DPSO system proposed in Vellasques et al. (in press), Gaussian mixture modeling (GMM) of optimization history was employed in order to model previous optimization problems.

In that approach, which is based on the combined use of PSO and GMM, a memory of GMMs is incrementally built using fitness (phenotypic) and parameter (genotypic) information of all intermediary solutions found during each case of optimization. This memory is split in two levels: Short Term Memory (STM) and Long Term Memory (LTM). For every new image, solutions are sampled from each GMM in the memory and re-evaluated. The distributions of sampled and re-evaluated fitness values are compared with the use of a statistical test and if they are considered to be similar, the best re-evaluated solution is employed directly for the new image avoiding a costly re-optimization operation. Otherwise, re-optimization is triggered. After that, a GMM is trained using fitness and parameter values of all intermediary solutions (from all generations). The new GMM plus the global best solution will form a probe. The STM probe is replaced with the new probe. For the LTM, the new GMM is merged with the most similar GMM in the memory if their distance is below a threshold estimated on the last cases of update. Otherwise it is simply added to the LTM, replacing the least recalled probe if a memory limit has been reached. The main motivation in using a GMM is that it allows modeling multimodal fitness landscapes. Moreover, new models can be easily merged into existing models, which provides an incremental learning capability to the system. This is crucial in scenarios involving long streams of heterogeneous document images as the memory needs to adapt automatically to variations in the stream. Readers interested in understanding better how is GMM employed in the optimization of long streams of document images are referred to our previous article (Vellasques et al., in press). This approach allowed for a significant decrease in the cost for IW of **heterogeneous** streams of document images compared to the case-based approach. In both approaches, when a new optimization problem is similar to a previously-solved one, solutions in memory corresponding to that previous problem should be recalled, avoiding a costly re-optimization process.

The basic assumption behind that approach is that after a learning phase, most new problem instances will result in recall rather than re-optimization operations. However, a significant variation in the stream of optimization problems such as that caused by a new attack, will result in an increase in the number of re-optimization operations. The time complexity of re-optimization is orders of magnitude higher than that of recall. Each attempt of recalling a memory element has a time complexity comparable to a single iteration in the optimization phase, and optimization generally requires generally 50 plus iterations. Decreasing this cost is an important issue. It has been demonstrated in literature that optimization strategies based on the use of an associative memory (Yang and Yao, 2008) outperform other dynamic optimization strategies in cyclic/recurrent problems. These techniques rely on storage of high performance solutions, as well as information about their fitness landscape using a density estimate. The most common approach to associative memory optimization is to inject memory solutions in the initial population, in a strategy named memory-based immigrants (Wang et al., 2007).

One limitation of approaches based on associative memory is that for a case of re-optimization, the density estimates will only provide an initial set of candidate solutions. After that, these solutions are evolved with the use of EC and the knowledge of previous problems provided by that estimate is not explored during the optimization process whatsoever. It has been observed in the Estimation of Distribution Algorithms (EDA) literature that probabilistic models can be employed in order to guide the optimization process (Pelikan et al., 2002). A second limitation is that although memory-based immigrants can reduce the number of generations needed for convergence, still, each generation involves re-evaluating the fitness of each solution.

In surrogate-based optimization, costly fitness evaluation operations are replaced by a regression model. Sampling, model update and optimization are applied in an iterative manner. The advantage of such approach is that most of the fitness evaluations required for optimizations are performed using a regression model at a fraction of the cost of an exact fitness evaluation. There are two schools of thought: a first one that sees a surrogate as an oracle that will replace the objective function (Queipo et al., 2005) and a second one that sees a surrogate as a compact database employed in order to forecast good solutions during optimization, accelerating convergence (Parno et al., in press). Both provide different ways of addressing the trade-off between model precision and fitness evaluation cost. The first one favors decreasing fitness evaluation over precision and is preferred in situations where the model provides a precise representation of the fitness landscape and/or the computational cost of fitness evaluation is too high. The second one favors precision over decreasing fitness evaluations and is preferred in situations where the model does not provide a precise representation of the fitness landscape and/or the cost of optimization is not too

[1] In a DOP, the optimum location and/or fitness value change over time.

high. Surrogate-based optimization involves a mixed use of exact and predicted fitness values which leads to a trade-off between increase in model precision and decrease in computational cost – a more precise model makes possible relying less on expensive exact fitness evaluations but improving model precision involves probing the exact fitness landscape which is computationally expensive.

It is important distinguishing between EDA and surrogate-based optimization. In each generation of EDA, solutions are sampled from a probabilistic model, re-evaluated and the best solutions are employed in order to update the model. In contrast, surrogate-based optimization builds a sampling plan in the parameter space. Then, numerical simulations are performed at the sampled locations, followed by model update and optimization. However, optimization is based on fitness values predicted by the model. This is the main advantage of surrogate-base optimization compared to EDA. In EDA the model guides the search process while in surrogate-based optimization, the model provides a mean of replacing expensive exact fitness evaluations with cheaper approximated fitness values obtained through regression.

In this paper, a novel approach called Dual Surrogate Dynamic PSO (DS-DPSO) is proposed in which models of previous optimization are employed as surrogates in order to decrease the computational burden associated with full re-optimization for a hybrid GMM/DPSO system (proposed in Vellasques et al. (2012)). This system performs four different levels of search for solutions with increasing computational burden and precision. As in previous research, levels 1 and 2 first attempt to recall ready-to-use solutions from a memory of GMMs. If embedding parameters require a significant adaptation, the optimization modules are activated in levels 3 and 4. Whenever re-optimization is triggered, an attempt to optimize the embedding parameters using the surrogate as an oracle is performed level 3. This allows for a substantial decrease in the number of fitness evaluations as the optimization process is performed mostly on a GMM regression model. If it fails, a new attempt is performed, this time using the surrogate as a database in order to accelerate convergence in level 4. This second optimization stage relies mostly on exact fitness evaluations (the surrogate is employed on a best-case basis) and, for this reason, provides a safeguard to the whole system for situations where the surrogate model recovered from memory at level 3 does not correspond to the new problem. The main advantage of this DS-DPSO strategy is that it tackles the precision/cost trade-off by relying on a memory of previous surrogates and employing two different surrogate-based optimization strategies in sequence – level 3 with smaller cost and smaller precision (but which should provide good results for situations where the model shares some similarity with the new problem), and level 4 with higher cost and precision which provides a safeguard to the whole system and allows building new surrogates (for cases of novel problem instances).

This research attempts to exploit a memory of GMMs (learned for a stream of reference training images) to decrease the computational cost of full re-optimization. In addition, incorporating knowledge about a new optimization problem into the model of a previous problem is considered in order to produce a model with a better fit to the new problem. It is assumed that whenever re-optimization is triggered, the best fit model should provide a good starting point for building a surrogate for the new problem. Regardless, it should decrease the computational burden of optimization by accelerating convergence. The proposed approach is validated empirically with the use of heterogeneous streams of bi-tonal document images. The University of Oulu's MediaTeam (Sauvola and Kauniskangas, 1999) dataset and samples of the Computer Vision and Image Understanding (CVIU) Journal are employed for this purpose. For each simulation, watermarking performance and number of fitness evaluations are reported.

A formulation of optimization problems in digital watermarking is provided in Section 2. A literature review of surrogate-based optimization is presented in Section 3. The proposed method named DS-DPSO is presented in Section 4. The experimental methodology and simulation results are presented and discussed in Sections 5 and 6.

2. Particle swarm optimization of embedding parameters

In the given formulation of IW, a stream of document images corresponds to a stream of optimization problems, where some problems may reappear over time. A PSO-based system proposed by the authors in a previous research (Vellasques et al., 2011) allows for the optimization of embedding parameters of multiple watermarks with different levels of robustness into a given bi-tonal image of a stream. During embedding, (1) the bi-tonal images is partitioned in blocks of equal size, (2) a flippability analysis is performed in order to evaluate the visual impact of flipping each pixel, and (3) the pixels are shuffled in order to distribute flippable pixels evenly across the image. Then, (4) each message bit is embedded on each block by manipulating the quantized number of black pixels on that block and finally, and (5) the image is deshuffled. Detection involves partitioning the image, shuffling using the same key used on embedding, and counting the quantized number of black pixels on each block in order to detect each message bit. Four different parameters can be adjusted in order to modify the trade-off between watermark robustness and image quality for a given image during embedding, namely: quantization step size (Q), size of the window employed in the flippability analysis (W), block width (B) and shuffling key index (S). Readers are referred to Vellasques et al. (2011) for more information on the bi-tonal watermarking system.

In our formulation of the optimization problem for digital watermarking, two watermarks – a fragile one with $Q_F = 2$ and a robust one with $Q_R = Q_F + \Delta Q$ – are embedded into the cover image **Co** where ΔQ is the difference between the robust and fragile quantization step sizes. The fitness function comprises robustness and quality metrics, aggregated with the use of the Chebyshev technique Collette and Siarry (2008):

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

where α_s is the scaling factor of the quality measurement *DRDM* (Distance Reciprocal Distortion Measure Lu et al., 2004), BCR_R (Bit Correct Ratio Areef et al., 2005; Pan et al., 2004 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 $\omega_i = \frac{1}{3},\ \forall\ i,\ r_i$ is the reference point of objective i. It is important to note that (unlike the BCR_F and *DRDM*) BCR_R is computed after an attack has been applied. 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.

A diversity preserving PSO (Kapp et al., in press) has been employed in order to optimize the embedding parameters based on the fitness function described above (Vellasques et al., 2011). Therefore, the fitness landscape comprises five dimensions: four in the parameter space ($\{x_1,x_2,x_3,x_4\} = \{\Delta Q,W,B,S\}$) and one in the fitness space ($f(\boldsymbol{x}),\ \forall\ \{\boldsymbol{x} \in \mathbb{R}^4\}$). Readers are referred to Vellasques

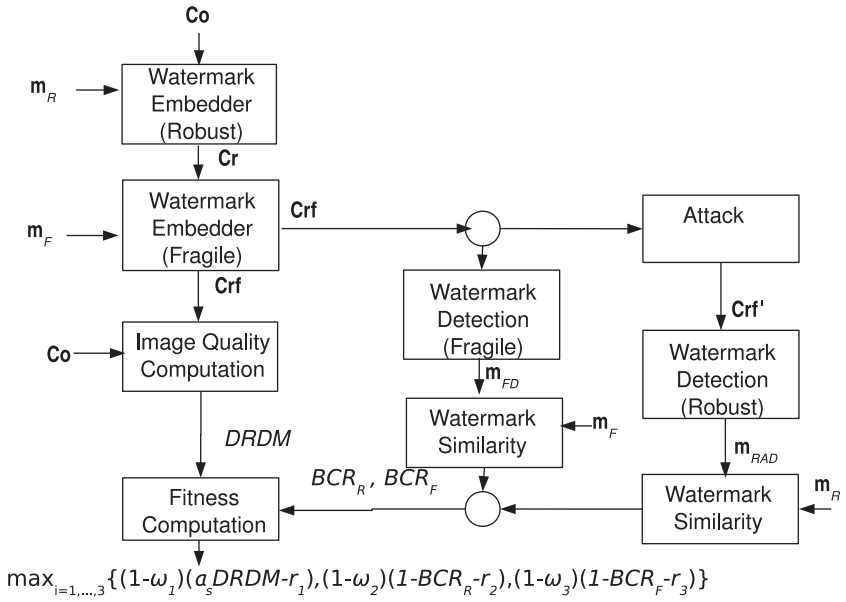


Fig. 1. Fitness evaluation module.

et al. (2011), Vellasques et al. (2012) for more information about the optimization problem formulation of digital watermarking.

One of the limitations regarding the use of EC on digital watermarking is that each fitness evaluation requires multiple embedding, detection and attack operations which are very costly. In the given application, a fitness evaluation involves numerous embedding operations, each of which has time complexity $O(|\mathbf{Co}| \cdot log(|\mathbf{Co}|))$ where $|\mathbf{Co}|$ is the number of pixels on cover image **Co** (with a magnitude of 10^6) while performing regression on a GMM has a time complexity of $O(K \times d)$ where K is the number of components in the GMM (usually between 5 and 20 for this case) and d is the dimension of the parameter space which is 4. Therefore, the cost of optimizing this bi-tonal watermarking system can be significantly reduced by replacing part of the exact fitness evaluations with regression models.

3. Surrogate-based optimization

Surrogate (or model-based optimization) allows tackling the computational burden of fitness evaluation in complex real-world problems. As stated by Queipo et al. (2005), a surrogate model can be seen as a non-linear inverse problem in which one aims to determine a continuous function $f(\boldsymbol{x}),\ \forall\ \{\boldsymbol{x} \in \mathbb{R}^d\}$ of a set of design variables from a limited amount of available data $\boldsymbol{f} = \{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)\}$ where $\boldsymbol{x}_i$ is a design variable, d is the dimensionality of the parameter space and N is the number of data points. During optimization, costly calls to $f(\boldsymbol{x})$ are partially replaced by a predicted value $f_P(\boldsymbol{x},\Theta)$

$$f_P(\boldsymbol{x}, \Theta) = \hat{f}(\boldsymbol{x}, \Theta) - \rho_c \varepsilon(\boldsymbol{x}, \Theta) \tag{2}$$

of $f(\boldsymbol{x})$ (Torczon and Trosset, 1998) where $\hat{f}(\boldsymbol{x}, \Theta)$ is an approximation to $f(\boldsymbol{x})$ based on model Θ, ρ_c is a constant that dictates how much emphasis will be put in exploring unknown regions of the model and $\varepsilon(\boldsymbol{x})$ is the prediction error. The basic approach to surrogate modeling assumes that Θ is unknown and then iteratively selects a set of design variables using stratified sampling – also known as design of experiments (DOE) to perform numerical simulations using this set and update the model. Once a given stop criterion has been achieved, the model is validated against $f(\boldsymbol{x})$. Once a good representation of $f(\boldsymbol{x})$ has been obtained, optimization is first performed using $f_P(\boldsymbol{x},\Theta)$, one or more solutions are re-evaluated on $f(\boldsymbol{x})$ and then, either Θ is refined using new data points or the process is halted in the case that a convergence criterion has been met.

A surrogate model can be either global or local (Dennis and Torczon, 1995). A local model provides a detailed approximation of a specific region of the fitness landscape while a global model provides a general approximation of the whole optimization problem. There are four different strategies to build a surrogate (Praveen and Duvigneau, 2007): (1) data-fitting models, where the approximation is constructed using available data; (2) variable convergence model, where the approximation is based on the numerical solution of a partial differential equation (PDE); (3) variable resolution models where the search space is discretized with the use of a hierarchy of grids; (4) variable fidelity models, where a hierarchy of physical models is employed in order to approximate the fitness function.

Most of the techniques found in the literature rely on data-fitting models. The advantage of such type of approach is that it uses pattern recognition methods such as radial basis functions, clustering, multilayer perceptron, polynomial fitting, Gaussian processes, support vector machines (Shi and Rasheed, 2008) which can be inferred even when domain knowledge is ill-defined (such as IW of stream of document images). Data-fitting approaches can be either off-line or on-line (Praveen and Duvigneau, 2007). An off-line surrogate is first trained with a set of data points that have been evaluated in the exact fitness function, is assumed to be an accurate representation of the exact function and is indicated in situations where computational burden is more important than precision. In contrast, an on-line surrogate is trained incrementally, closely integrated into the optimization method and is indicated in situations where precision is more important than computational burden.

One of the main issues with surrogate-based optimization is that it is generally difficult to obtain a model with sufficient approximation accuracy due to the lack of data and/or high dimensionality which leads to models with high approximation errors that commonly result in false optima during the optimization phase (Jin et al., 2002). This is an important issue for on-line surrogates since the construction of a good surrogate requires an experimental design which is space-filling in order to capture the essential trends of the fitness landscape. Yet the goal of optimization is to generate points which lead to improvements in the fitness function (El-Beltagy et al., 1999). A surrogate model is

therefore subject to a trade-off between decrease in computational burden and model fidelity. This issue can be partially alleviated with the use of evolution control, data selection (Jin et al., 2002), combined local and global models (Zhou et al., 2007), archive of solutions (case-based surrogate) (Fonseca et al., 2009) and incremental stratified sampling (Yan and Minsker, 2010).

Using evolution control, part of the solutions obtained through surrogate optimization are validated against the exact (but costly) fitness function. It provides a mean of avoiding false convergence (Gräning et al., 2005). Since the model provides an approximation of the real problem, it is expected to contain false optima (as observed in El-Beltagy et al. (1999), in the course of model update, false optima are likely to appear and disappear). Evolution control is subject to a trade-off between false convergence avoidance and computational burden – employing more solutions decreases the risk of false convergence but at a higher computational burden.

In data selection, the samples that will be employed to update the model are selected in order to improve the cost/benefit of model update in situations where such cost is high. Combining global and local models allows using a coarser level of fidelity to tackle exploration and a finer one to tackle exploitation in EC. The use of case-based models in EC can result in poor generalization and imply in high computational burden (in the long term) when compared to density estimates. Incremental stratified sampling allows a better space-filling of the fitness landscape since it accounts for previously sampled realizations. However, it has been demonstrated in Vellasques et al. (2012) that it is possible to obtain a good space-filling of the fitness landscape by combining a diversity preserving mechanism in EC with a model that is incrementally trained over a representative set of optimization problems.

4. A dual-surrogate DPSO approach for fast intelligent watermarking

4.1. System overview

Fig. 2 illustrates the DS-DPSO approach. It has four levels with increasing computational cost and fidelity and at each level, an attempt to solve the watermarking problem for current cover image ($\boldsymbol{Co}_i$) is made. If it fails, a next attempt is made, in an upper level with an increased fidelity but at higher computational cost. The first two levels comprise memory recall. The third and fourth are the off-line and on-line optimization levels, respectively. The assumption is that in a situation involving recurrent problems, after a training phase, the memory will be precise enough thus most of the images should result either in a recall to the Short Term Memory (STM) or to the Long Term Memory (LTM). If recall fails, but the memory still provides a good approximation to the given problem, an attempt to optimize the parameters using an off-line surrogate is attempted. If this attempt also fails, then the costlier on-line surrogate is activated.

Fig. 3 depicts the two recall levels. The STM (represented as $\mathfrak{M}_S$) contains the best solution ($\boldsymbol{p}_{g,S}$) plus a GMM approximation (Θ_S) of the fitness landscape of a single image ($\boldsymbol{Co}_S$). This combination of global best and GMM is called a **probe**. The LTM (represented as $\mathfrak{M}$) contains $|\mathfrak{M}|$ probes. Each probe contains a mixture model (Θ_i) obtained during the optimization of several different images and a global best solution ($\boldsymbol{p}_{g,i}$). LTM probes are sorted in reverse order of their number of successful recalls. It is important to mention that **phenotypic** and **genotypic** data of all solutions found during the optimization of a given image are employed in order to train a GMM.

During STM recall, $\boldsymbol{p}_{g,S}$ plus a set of solutions re-sampled from Θ_S are re-evaluated on image $\boldsymbol{Co}_i$. Then, the similarity between the distribution of the sampled and re-evaluated fitness values is computed with the use of the Kolmogorov–Smirnov (KS) statistical test (Anonyms, 2010). If the KS value is below a critical value for a confidence level α_{Crit}, it means that both distributions are similar and the best re-evaluated solutions is employed directly. Otherwise, a change is considered to have occurred in the landscape (compared to that of Θ_S) and level 2 is activated. In level 2, the same process is repeated for each LTM probe until either a case of KS value below the critical value has occurred or all probes have been tested.

Fig. 4 depicts the first optimization level (off-line surrogate). The underlying principle is that for some failed recall attempts, the best GMM (the one that resulted in the smallest KS value during recall attempts) will already provide a good approximation of the fitness landscape and the re-evaluated fitness values (necessary during the recall) allow improving its fidelity for the new problem (the proposed approach is based on data-fitting model). Therefore, the DPSO technique described in Kapp et al. (in press) is employed in order to optimize the embedding parameters, but using the best GMM as surrogate most of the

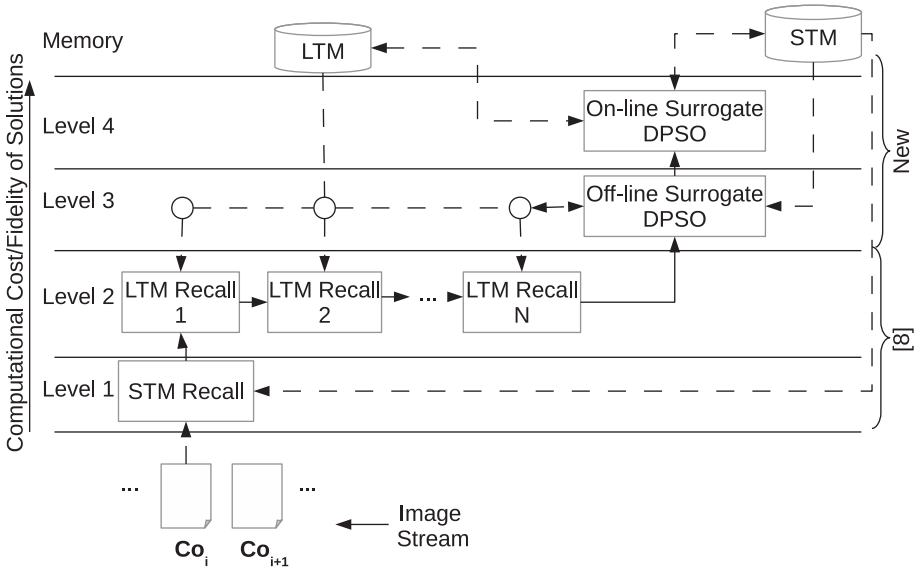


Fig. 2. Overview of the proposed DS-DPSO technique for intelligent watermarking of document image streams.

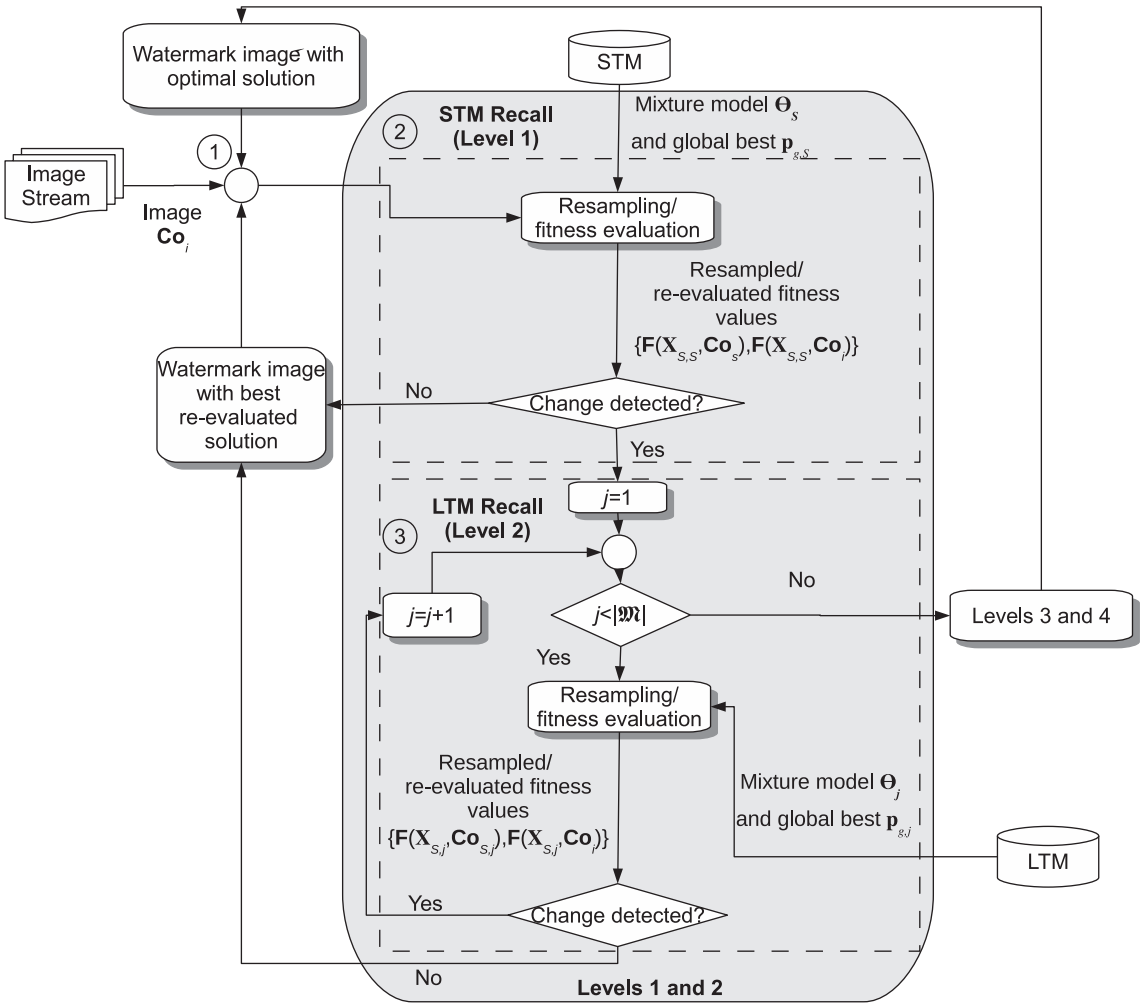


Fig. 3. Flowchart diagram detailing the recall modules. Anchor points are employed in order to guide the reader. For each image in a stream of document images (step 1), an attempt to recall the STM is performed first (step 2) followed by an attempt to recall LTM, if necessary (step 3).

time (which has a smaller computational burden than an on-line surrogate approach). The GMM is employed in regression mode, an approach named Gaussian Mixture Regression (GMR) (Sung, 2004).

The surrogate is initialized with the mixture model that resulted in the smallest KS value and updated with re-evaluated solutions obtained during recall. Then, while the stopping criterion has not been reached (for all cases of re-optimization, we propose stopping optimization if global best has not improved for a certain number of generations Zielinski and Laur, 2007), an iteration is performed on the surrogate function (swarm $\mathbf{X}_A$), swarm solutions are re-evaluated in the exact function in order to avoid false optima (evolution control) and these solutions are employed in order to update the model. After that, if the surrogate improved the best recalled solution, the most similar LTM probe is updated with the surrogate and the surrogate solution that resulted in the best fitness in the exact function is employed on $\boldsymbol{Co}_i$. Otherwise, level 4 is activated.

Fig. 5 depicts level 4. Here, DPSO will be performed using primarily the exact fitness function, at a higher computational burden than that of the third level, but still with the possibility of a decreased computational burden compared to full optimization (depending on how fast convergence occurs). Solutions are re-sampled from the probe that resulted in the smallest KS value and injected into the exact function swarm ($\mathbf{X}_B$). Optimization is performed on the surrogate function until a stop criterion has been reached. Then, the best solution is re-evaluated on the exact fitness function and injected into swarm $\mathbf{X}_B$ if it improves a corresponding neighbor. After that, an iteration is performed on the exact fitness function (swarm $\mathbf{X}_B$). When a stop criterion has been reached, a new mixture model is estimated, the best solution and mixture model are added to the STM, replacing the previous STM probe. If the new probe is similar to a given LTM probe, it is merged with that probe. Otherwise it is inserted (the probe with smallest number of successful recalls is deleted if memory limit has been reached).

Such approach allows tackling the optimization of recurrent problems as a machine learning problem. The surrogate might be de-activated in a training environment, where the constraints on computational cost are less severe in order to obtain a high fidelity representation of a given dynamic optimization problem. This should result in a model with good space filling properties, specially because the DPSO technique employed has a diversity preserving capability (Clerc, 2006). Then, the surrogate can be re-activated and the models obtained during training can be employed in order to perform large scale dynamic optimization of recurrent problems in a production (test) environment where computational burden constraints are more severe. This should minimize the issues of high approximation errors and false optima, specially early in optimization, since the on-line surrogate provides a safeguard for the whole system.

The STM/LTM recall (Fig. 3) and update mechanisms (memory update box in Figs. 4 and 5) are described with details in

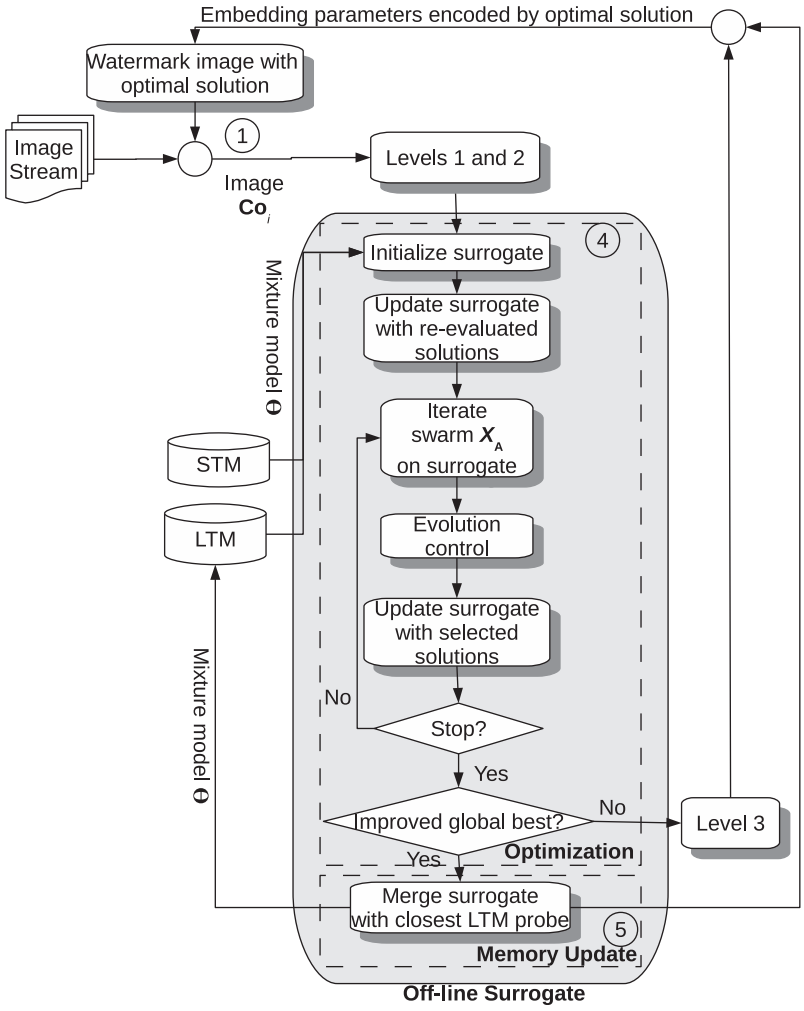


Fig. 4. Flowchart diagram detailing level 3. Anchor points are employed in order to guide the reader. Whenever levels 1 and 2 fail, optimization is performed primarily on the surrogate (step 4). After that, the LTM is updated with the GMM employed on optimization (step 5).

Vellasques et al. (2012). Next, we present in details the key elements of the proposed approach, namely on-line update of GMMs, Gaussian Mixture Regression (GMR), evolution control. Then, we present the off-line and on-line surrogate DPSO modules and how both are integrated.

4.2. STM and LTM recall

For every new image $\boldsymbol{Co}_i$ (see 1 in Fig. 3), an attempt to recall the STM probe is conducted at first. If it fails, the same process is conducted for each LTM probe until either all probes have been tested or a successful recall has occurred.

During a STM recall (see 2 in Fig. 3) solutions are re-sampled from the STM mixture model (Θ_S) and re-evaluated in the new image (along with the global best $\boldsymbol{p}_{g,S}$). The distribution of the sampled set of fitness values $\boldsymbol{F}(\boldsymbol{X}_{S,S},\boldsymbol{Co}_S)$ is compared against the distribution of re-evaluated fitness values $\boldsymbol{F}(\boldsymbol{X}_{S,S},\boldsymbol{Co}_i)$ with the use of the Kolmogorov–Smirnov statistical test (KS). If the KS value between both distributions is smaller than a critical value for a confidence level α_{Crit}, the watermarking parameters encoded by the best recalled solution 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 Θ_j and global best $\boldsymbol{p}_{g,j}$ in the LTM – re-sampling, re-evaluating the re-sampled and global best solutions on the new image and comparing re-sampled fitness values $\boldsymbol{F}(\boldsymbol{X}_{S,j},\boldsymbol{Co}_S)$ against the re-evaluated values $\boldsymbol{F}(\boldsymbol{X}_{S,j},\boldsymbol{Co}_S)$ using the KS test. This process is repeated until a case of KS value smaller than the critical value occurs or all probes have been tested. The STM/LTM recall is described more carefully in Vellasques et al. (2012).

STM/LTM recall (levels 1 and 2) is expected to be enough in most of the cases (specially for stable problem streams). However, when optimization is triggered too often (in situations involving high variability in the problem stream) the cost of re-optimization becomes a serious issue since a single re-optimization operation is several times more expensive than a recall. Next we propose a strategy to decrease this cost based on knowledge obtained on previous cases of re-optimization.

4.3. Off-line/on-line surrogate PSO

Rather than focusing on the level of detail provided by the model (global or local) we will focus in the fidelity/computational burden trade-off. The reason different levels of model fidelity are employed in the literature is that it is assumed that a model has to be trained from scratch for each new problem. Therefore, the exploration/exploitation has to be addressed at the same time. However, we formulate surrogate-based optimization as a pattern recognition problem: a set of surrogates is built during a training phase and then matched against new problems during a test phase.

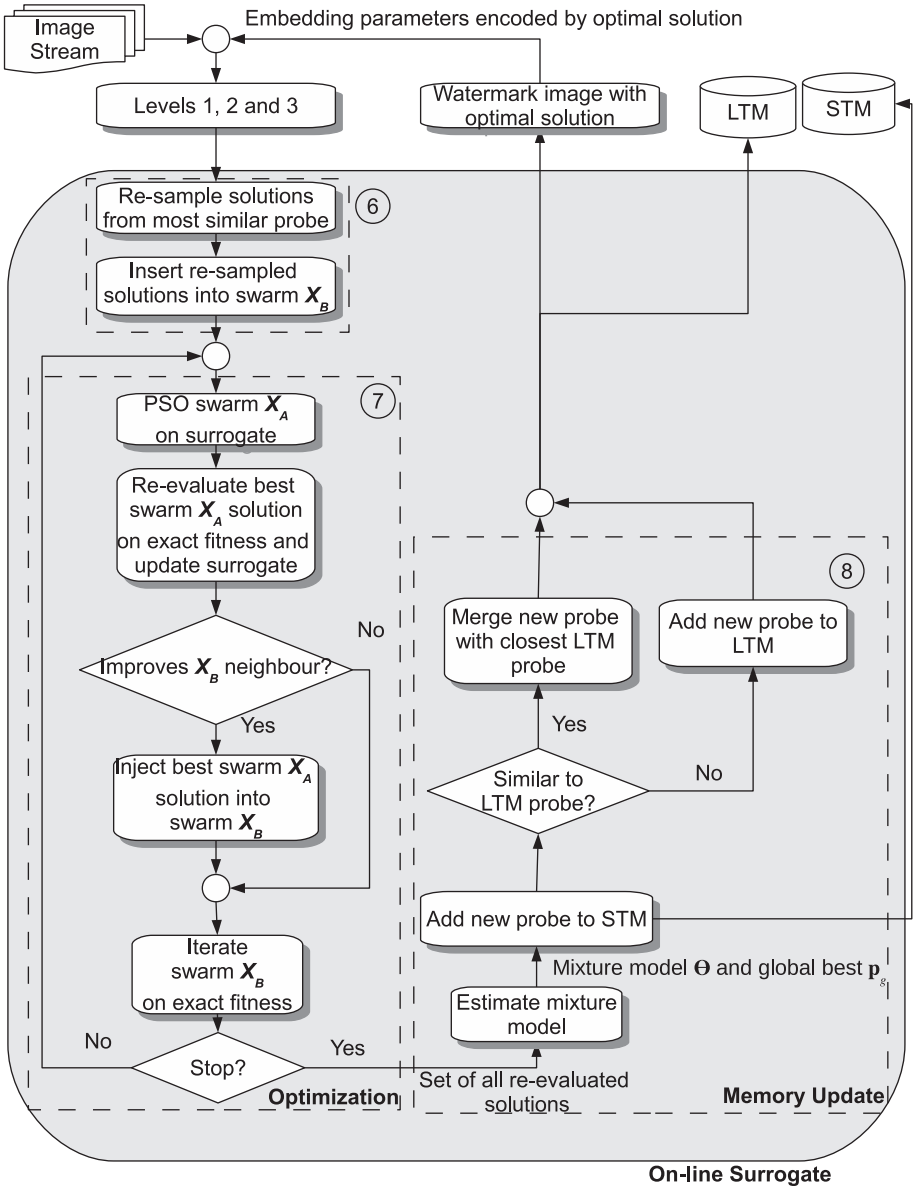


Fig. 5. Flowchart diagram detailing level 4. Anchor points are employed in order to guide the reader. Whenever level 3 fails, solutions are re-sampled from the most similar probe (step 6) and then, optimization is performed using two different swarms, one for the the exact fitness and another one for the surrogate (step 7). After that, the memory is updated using the optimization history of the swarm employed to optimize the exact fitness (step 8).

Since the matching is based on a limited set of sentry points, we propose a multi-level optimization approach where the fidelity is increased as the solution proposed by a preceding level is rejected (at the cost of a higher computational burden).

The underlying assumption behind the dual surrogate mechanism is that whenever a model is a good representation of the new problem but did not result in a successful recall, a near optimal solution can be found through a fine search. Model update requirements in such case is minimal. Otherwise, a full search is required at the cost of a more expensive model update, involving a greater number of exact fitness evaluations.

The recall mechanism will provide the starting surrogate model and a set of fitness values for an initial update. Moreover, the optimal solution ($\boldsymbol{X}_{S,o}$) is initialized with the best recalled solution. The GMM that resulted in the smallest KS value during recall (updated with the re-evaluated solutions) will be chosen as the initial surrogate. We also inject the best recalled solutions of that probe into both, the surrogate and exact fitness swarms (as proposed by Kapp et al. (in press)). Since the GMM has been trained using all solutions found during the optimization of a given image, it should be considerably more precise than a model built using a few sampled points.

Three aspects are crucial in the surrogate optimization levels: updating GMMs with new data in an on-line manner, performing regression on GMMs and validating the evolution of the off-line surrogate against the exact fitness function.

4.3.1. On-line update of GMMs

Model update (see "Update surrogate with re-evaluated solutions", "Update surrogate with selected solutions" blocks in Fig. 4 and "Re-evaluate best swarm $\boldsymbol{X}_A$ solution on exact fitness and update surrogate" block in Fig. 5) is an essential issue in surrogate-based optimization. The baseline intelligent watermarking system already relies on GMM modeling of all solutions found through all generations (optimization history) in order to model a fitness landscape. A GMM is a powerful statistical modeling technique which consists of a linear combination of a finite number of Gaussian models

$$p(\boldsymbol{x}|\Theta) = \sum_{j=1}^{K} \alpha_j \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{3}$$

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 is the mixing weights, parameters of the jth model (with $0 < \alpha_j \leqslant 1$ and $\sum_{j=1}^{K} \alpha_j = 1$) and $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ is a multivariate Gaussian probability density function (pdf) with mean vector $\boldsymbol{\mu}_j$ and covariance matrix $\boldsymbol{\Sigma}_j$.

In the baseline approach, a GMM is first estimated in batch mode with optimization history data using Expectation Maximization (EM) (Figueiredo and Jain, 2000). Then, this new GMM is either inserted or employed in order to update an existing GMM in the LTM according to a distance metric (Sfikas et al., 2005). During update, components of the the new and existing GMMs are merged based on their Bhattacharyya distance (Hennig, 2010).

Since the proposed approach relies on GMMs obtained for a training stream of images in order to predict fitness values for a different stream of images, it is crucial to adapt a GMM using new data. An intuitive approach would be to use the same strategy employed in the baseline system (train a new GMM using new data and then merge with the existing GMM). However, the number of data points needed to estimate a covariance matrix is $N_d = d + d(d+1)/2$ which means it grows quadratically with dimension d (Figueiredo and Jain, 2000) making unfeasible the applicability of such approach for a small quantity of data. Engel and Heinen (2010) tackle this problem by starting with an initial uniform covariance matrix $\boldsymbol{\Sigma}^0 = \sigma_{ini}^2 \boldsymbol{I}$ where σ_{ini} is the width of the initial covariance matrix and $\boldsymbol{I}$ is an identity matrix and incrementally adding new components or updating existing ones based on a novelty criterion. Such approach assumes an untrained GMM and is justified in situations where a new GMM has to be trained from scratch in an on-line fashion. However, there are several practical limitations on training a GMM using a small quantity of data such as initialization of mixture components, escaping from situations where two or more components share the same data points, defining the appropriate number of components (Figueiredo and Jain, 2000).

Since we rely on an initial GMM trained in batch mode using a technique that can tackle the issues above, we can rely on this initial model and then adjust its components using new data. There are two strategies to do that. The first is to rely on some sort of statistics for each component about the previous update in order to adjust the components using the new datum (Yamanishi et al., 2000; Zhang and Scordilis, 2008). The second is to rely on a learning factor which is gradually decreased (Stauffer and Grimson, 2000). We will employ the second approach (slightly adapted to our specific problem) since the first assumes a fixed number of components and our baseline memory management mechanism employs pruning in order to adjust the number of components according to new data which would result in loss of such statistic. Given a new datum $\boldsymbol{x}_t$ at time t, we first find the index of the component that best fits $\boldsymbol{x}_t$:

$$j^* = \text{argmax}_j\{\mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\} \tag{4}$$

and then update the mixture weights of the components:

$$\alpha_j^t = \begin{cases} (1-\gamma)\alpha_j^{t-1} + \gamma, & \text{if } j = j^* \\ (1-\gamma)\alpha_j^{t-1}, & \text{otherwise} \end{cases} \tag{5}$$

where γ is the learning rate. The mean and covariance matrix of the best fit component are updated in a similar manner:

$$\boldsymbol{\mu}_{j^*}^t = (1-\rho)\boldsymbol{\mu}_{j^*}^{t-1} + \rho \boldsymbol{x}_t \tag{6}$$

$$\boldsymbol{\Sigma}_{j^*}^t = (1-\rho)\boldsymbol{\Sigma}_{j^*}^{t-1} + \rho\left(\boldsymbol{x}_t - \boldsymbol{\mu}_{j^*}^t\right)^T\left(\boldsymbol{x}_t - \boldsymbol{\mu}_{j^*}^t\right) \tag{7}$$

where

$$\rho = \alpha_{j^*}^{t-1}\mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_{j^*}^{t-1}, \boldsymbol{\Sigma}_{j^*}^{t-1}\right) \tag{8}$$

4.3.2. Gaussian mixture regression (GMR)

In the proposed approach, GMR (see "Iterate swarm $\boldsymbol{X}_A$ on surrogate" block in Fig. 4 and "PSO swarm $\boldsymbol{X}_A$ on surrogate" block in Fig. 5) allows employing the knowledge of previous cases of optimization to decrease the computational burden of re-optimization. The main motivation for relying on GMMs in order to model the fitness landscape of a stream of optimization problems is that it combines the memorization ability of non-parametric techniques with the compactness of parametric techniques. It has been observed in our previous research, that in this specific application it allows a very precise sampling of the fitness landscape.

By partitioning each density into disjoint sets of independent and dependent variables, Sung (2004) formulated that it is possible to use a GMM as a regression model:

$$\hat{\boldsymbol{f}}(\boldsymbol{x}, \Theta) = \sum_{j=1}^{K} P_j(\boldsymbol{\theta}_j|\boldsymbol{x})\boldsymbol{m}_j(\boldsymbol{x}) \tag{9}$$

$$\boldsymbol{\varepsilon}^2(\boldsymbol{x}, \Theta) = \sum_{j=1}^{K} P_j(\boldsymbol{\theta}_j|\boldsymbol{x})(\boldsymbol{m}_j(\boldsymbol{x})^2 + \boldsymbol{\sigma}_j^2) - \left(\sum_{j=1}^{K} P_j(\boldsymbol{\theta}_j|\boldsymbol{x})\boldsymbol{m}_j(\boldsymbol{x})\right)^2 \tag{10}$$

where:

$$\boldsymbol{m}_j(\boldsymbol{x}) = \boldsymbol{\mu}_{j,1} + \boldsymbol{\Sigma}_{j,21}\boldsymbol{\Sigma}_{j,11}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_{j,1}) \tag{11}$$

$$\boldsymbol{\sigma}_j^2 = \boldsymbol{\Sigma}_{j,22} - \boldsymbol{\Sigma}_{j,21}\boldsymbol{\Sigma}_{j,11}^{-1}\boldsymbol{\Sigma}_{j,12} \tag{12}$$

$$P_j(\boldsymbol{\theta}_j|\boldsymbol{x}) = \frac{\alpha_j \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{j,11}, \boldsymbol{\Sigma}_{j,11})}{\sum_{j=1}^{K} \alpha_j \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{j,1}, \boldsymbol{\Sigma}_{j,11})} \tag{13}$$

This approach provides a distribution of the predicted value with $\hat{\boldsymbol{f}}(\boldsymbol{x})$ as the mean and $\boldsymbol{\varepsilon}^2(\boldsymbol{x})$ as the covariance matrix. This makes GMR a very interesting approach for situations where a smooth approximation of a function is necessary like robotics (Calinon, 2009). Predicting fitness values using this technique is straightforward, for a given $\boldsymbol{x}$, we compute $f_P(\boldsymbol{x}) = \hat{f}(\boldsymbol{x}) + \varepsilon(\boldsymbol{x})$ using Eqs. (9) and (10). It is important noticing that in the given application, the predicted value and error are scalars (mean and variance) rather than a vector and a covariance matrix.

4.3.3. Evolution control

Avoiding convergence to false optima is one of the most important issues in harnessing the computational cost savings allowed by surrogate-based optimization. This is specially important for level 3 which relies mostly on surrogate fitness evaluation. For this reason, we propose the use of an evolution control mechanism (see "Evolution control" block in Fig. 4) for the off-line surrogate in order to mitigate this problem. Because of the space-filling nature of surrogate models, optima will consist many times of an interpolation of many different near optimal points. For this reason, model fidelity tends to be improved as the model is updated. However, this requires re-evaluating more fitness values resulting in an increase in computational burden.

As mentioned before, the model fidelity versus computational burden trade-off varies across different applications and can be adjusted with the use of evolution control. There are two main approaches to evolution control (Jin et al., 2000): (1) controlled individuals; (2) controlled generations. In controlled individuals, the actual and predicted fitness values of part of the individuals in the population are re-evaluated with the real fitness function. In controlled population, the whole population is re-evaluated at a certain time interval.

We propose using an individual-based approach as it has a smaller computational burden than generation-based approach. In our approach, for each generation, solutions in the surrogate swarm are ranked according to their surrogate fitness value and the N_{s1} best performing solutions are re-evaluated in the exact function $f(\boldsymbol{x})$. If both, the predicted and effective fitness value for the best re-evaluated solution is better than that of the best re-evaluated solution (optimal) found so far, then the optimal solution is replaced by the best re-evaluated solution for that generation. This can be seen as a pre-selection strategy (Gräning et al., 2005) as the parents of the next generation (attractors in PSO terms) are chosen among the best re-evaluated solutions.

4.3.4. Off-line surrogate PSO

In the off-line surrogate optimization, the PSO approach described in Vellasques et al. (2011) will be employed in order to optimize the embedding parameters, but using the surrogate as fitness function (approach described in Section 4.3.2). The surrogate is initialized with the best recalled mixture (see 4 in Fig. 4). The best recalled mixture is the one that resulted in the smallest KS value during STM/LTM recall. After that, the surrogate is updated using all the solutions re-sampled during recall and their re-evaluated fitness solution (based on the approach described in Section 4.3.1). At each generation, the velocity and position of surrogate swarm solutions ($\boldsymbol{X}_A$) are updated based on the surrogate fitness and the N_{s1} best solutions are re-evaluated in $\boldsymbol{Co}_i$. The model is updated using these solutions and their re-evaluated fitness. If the best re-evaluated fitness ($f(\boldsymbol{x}_{g,s1})$) improves the candidate optimal solution ($\boldsymbol{X}_{S,o}$) then the surrogate global best ($\boldsymbol{p}_{g^*,s1}$) is replaced with it. This process (optimization, re-evaluation, model update, best solution update) is repeated until no improvement in the best solution occurs for a given number of generations.

It is important to observe that in surrogated-based optimization, predicted improvements in $\boldsymbol{X}_{S,o}$ must correspond to actual improvements. That is, if an improvement has been predicted but not achieved (or the opposite), it means that the surrogate provides little knowledge about that specific region. Therefore we propose updating $\boldsymbol{X}_{S,o}$ only if an improvement has been predicted and achieved (Dennis and Torczon, 1995) (more specifically, if $\frac{f(\boldsymbol{X}_{S,o})-f(\boldsymbol{x}_{g,s1})}{\boldsymbol{X}_{S,o}-f_P(\boldsymbol{x}_{g,s1},\Theta)} > 0$). After the stop criterion has been reached (no improvement in $\boldsymbol{X}_{S,o}$ for a certain number of generations), if at least one case of improvement has occurred during the whole optimization process, the best re-evaluated solution found will be employed as is and the LTM is updated with the surrogate model. Otherwise, level 4 is activated.

Algorithm 1 summarizes the off-line surrogate level. For the sake of clarity, Eqs. (2), (9) and (10) described earlier define the surrogate fitness (GMR) while Eqs. (5)–(7) also described earlier define the GMM on-line update mechanism. The main loop is repeated until the stop criterion is reached. In the proposed approach, optimization stops if the global best has not improved for a certain number of generations.

The optimal solution ($\boldsymbol{x}_{o1}$) is initialized with the best recalled solution (line 1). Then, the surrogate model (Θ_b) is updated with all the re-sampled solutions ($\boldsymbol{X}_S$) and respective fitness values (line 2). After that, the swarm is iterated (velocity and position update) based on the surrogate fitness (line 4). The best N_{s1} solutions are re-evaluated on image $\boldsymbol{Co}$ (line 5). If the best re-evaluated solution improves the optimal solution (line 6), the optimal solution (line 7) and the surrogate swarm global best (line 8) are updated with the best re-evaluated solution. Next, the surrogate model is updated with the best N_{s1} re-evaluated solutions (line 10). Lines 4 to 10 are repeated until the stop criterion has been reached. Next, if at least one improvement occurred in the optimal solution (line 12), the LTM is updated (either merge or insert) with the surrogate (line 13) and the optimal solutions is employed on $\boldsymbol{Co}$ avoiding the costlier level 4.

Algorithm 1. Off-line surrogate optimization

Inputs:
$\boldsymbol{Co}$ – cover image.
Θ_b – surrogate model (mixture model which resulted in best KS value during recall).
$\boldsymbol{X}_S$ – set of all solutions sampled during recall.
N_{s1} – number of solutions for evolution control.
Definitions:
$\boldsymbol{X}_{S,o}$ – best recalled solution.
$f_P(\boldsymbol{x},\Theta)$ – surrogate fitness (Eqs. (2), (9) and (10).
$\boldsymbol{x}_{g,s1}$ – best re-evaluated solution for current generation.
$\boldsymbol{p}_{g^*,s1}$ – surrogate swarm global best.
Output:
$\boldsymbol{x}_{o1}$ – optimal solution.
1: $\boldsymbol{x}_{o1} \leftarrow \boldsymbol{X}_{S,o}$
2: Update Θ_b with $\boldsymbol{X}_S$ (Eqs. (5)–(7)).
3: **repeat**
4: Iterate swarm (update particles velocity and position) based on $f_P(\boldsymbol{x},\Theta)$.
5: Re-evaluate the best N_{s1} solutions on $\boldsymbol{Co}$.
6: **if** $\frac{f(\boldsymbol{x}_{o1})-f(\boldsymbol{x}_{g,s1})}{f(\boldsymbol{x}_{o1})-f_P(\boldsymbol{x}_{g,s1},\Theta_b)} > 0$ **then**
7: $\boldsymbol{x}_{o1} \leftarrow \boldsymbol{x}_{g,s1}$
8: $\boldsymbol{p}_{g^*,s1} \leftarrow \boldsymbol{x}_{g,s1}$
9: **end if**
10: Update Θ_b with the best N_{s1} solutions and respective re-evaluated fitness values.
11: **until** Stop criterion has been reached
12: **if** $f(\boldsymbol{x}_{o1}) < f(\boldsymbol{X}_{S,o})$ **then**
13: Update LTM with Θ_b.
14: **end if**

4.3.5. On-line surrogate PSO

The on-line surrogate technique is based on the approach of Parno et al. (in press). Two populations ($\boldsymbol{X}_A$ and $\boldsymbol{X}_B$) are employed, one for the surrogate fitness function and another one for the exact fitness. The $\boldsymbol{X}_B$ population is partially initialized with solutions sampled from the same mixture employed in the surrogate initialization (see 6 in Fig. 5). Optimization is performed first using population $\boldsymbol{X}_A$ on the surrogate fitness function (see 7 in Fig. 5). The best solution from $\boldsymbol{X}_A$ ($\boldsymbol{p}_{g,s2}$) is re-evaluated in the current image. If it improves the neighborhood best of population $\boldsymbol{X}_B$, that neighborhood best is replaced with $\boldsymbol{p}_{g,s2}$. The surrogate model is updated using the re-evaluated solution. Next, an iteration is performed using population $\boldsymbol{X}_B$ on the exact fitness. This process (optimization on $\boldsymbol{X}_A$, re-evaluation on exact fitness, injecting the re-evaluated solution on $\boldsymbol{X}_B$, iteration on $\boldsymbol{X}_B$) is repeated until a stop criterion has been reached.

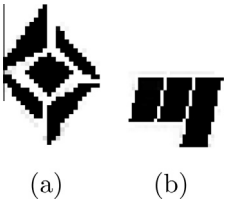
This approach allows avoiding the extra cost of stratified sampling since (1) the initial model is expected to provide some knowledge about the new problem; (2) surrogate in level 4 is more like an insurance policy for the previous levels (in the worst case, the surrogate will provide no improvement and the performance will be equivalent to that of completely reseting the swarm for each new image). However, as observed in Parno et al. (in press), such approach generally results in a speed up in convergence time compared to full optimization. The reason is that it relies primarily on exact fitness evaluations, which should compensate any false optimum found in the surrogate fitness. Thus, evolution control is not an issue in level 4.

After optimization if finished, the best solution is employed for the given image and all solutions found during the course of the optimization of the exact function are employed in order to train a GMM (see 8 in Fig. 5). The resulting GMM and best solution will form a probe that will replace the current STM probe. The LTM update works as follows: the GMM of the new probe is either merged with the GMM of the most similar probe in the LTM or inserted based on a C2 distance (Sfikas et al., 2005) threshold (computed over the last T cases of re-optimization). The mean value of the smallest C2 distance for each update operation (μ_{δ}^{t}) is computed for the T last cases of re-optimization. An insert occurs if $C2 - \mu_{\delta}^{t}$ is greater than the standard deviation (σ_{δ}^{t}) for the same time-frame. Otherwise a merge operation is performed. The LTM update procedure is described more carefully in Vellasques et al. (2012).

Algorithm 2 summarizes level 4. Initially, N_i solutions are re-sampled from the surrogate model (Θ_b) and injected into the exact fitness swarm ($\boldsymbol{X}_B$, line 1). The optimal solution ($\boldsymbol{x}_{o2}$) is initialized with the best recalled solution (line 2). Then, the solutions in the surrogate swarm ($\boldsymbol{X}_A$) are initialized randomly (line 4) and $\boldsymbol{X}_A$ is optimized based on the surrogate function (line 5) until a stop criterion has been reached (global best did not improve for a certain number of iterations). Next, the surrogate global best ($\boldsymbol{p}_{g^*,s2}$) is re-evaluated on $\boldsymbol{Co}$ (line 6) and the surrogate model is updated with the re-evaluated $\boldsymbol{p}_{g^*,s2}$ (line 7). After that, the corresponding best neighbor in $\boldsymbol{X}_B$ is updated with $\boldsymbol{p}_{g^*,s2}$ accordingly (lines 8–11). Next, $\boldsymbol{X}_B$ is iterated based on the exact fitness function (line 12) and the optimal solution is updated with the best of generation ($\boldsymbol{X}_{B,g}$) accordingly (lines 13–15). The procedure between lines 4 and 15 is repeated until the stop criterion has been reached ($\boldsymbol{x}_{B,g}$ did not improve $\boldsymbol{x}_{o2}$ for a certain number of generations). Finally, a new GMM is created using genotypic and phenotypic data from all re-evaluated solutions (including recall) and the STM/LTM memory is updated (line 17). It is important to mention that despite their similarity, Algorithms 1 and 2 are based in two fundamentally different strategies in what concerns the use of surrogates. The former employs the surrogate as a replacement (oracle) for the exact fitness function while the latter employs the surrogate as a database that should help to accelerate convergence in the exact fitness optimization.

Algorithm 2. On-line surrogate optimization

Inputs:
$\boldsymbol{Co}$ – cover image.
Θ_b – mixture model which resulted in best KS value during recall.
N_i – amount of injected solutions.
Definitions:
$\boldsymbol{p}_g$ – exact fitness swarm neighborhood best.
$\boldsymbol{X}_A$ – surrogate population.
$\boldsymbol{X}_B$ – exact function population.
$\boldsymbol{x}_{B,g}$ – best of generation ($\boldsymbol{X}_B$).
$\boldsymbol{p}_{g^*,s2}$ – surrogate swarm global best.
$\boldsymbol{X}_{\boldsymbol{x},k}$ – k nearest neighbors of $\boldsymbol{x}$ in $\boldsymbol{X}_B$.
Output:
$\boldsymbol{x}_{o2}$ – optimal solution from $\boldsymbol{X}_B$.
1: Re-sample N_i solutions from Θ_b and inject into $\boldsymbol{X}_B$.
2: $\boldsymbol{x}_{o2} \leftarrow \boldsymbol{X}_{S,o}$
3: **repeat**
4: Re-randomize $\boldsymbol{X}_A$.
5: Optimize $\boldsymbol{X}_A$ based on Θ_b.
6: Re-evaluate $\boldsymbol{p}_{g^*,s2}$ on $\boldsymbol{Co}$.
7: Update Θ_b with $\boldsymbol{p}_{g^*,s2}$.
8: $\boldsymbol{p}_g \leftarrow min_{f(x)}\{\boldsymbol{X}_{\boldsymbol{p}_{g^*,s2},k}\}$
9: **if** $f(\boldsymbol{p}_{g^*,s2}) < \boldsymbol{p}_g$ **then**
10: $\boldsymbol{p}_g \leftarrow \boldsymbol{p}_{g^*,s2}$
11: **end if**
12: Iterate $\boldsymbol{X}_B$ (update particles velocity and position) based on $\boldsymbol{Co}$.
13: **if** $f(\boldsymbol{x}_{B,g}) < f(\boldsymbol{x}_{o2})$ **then**
14: $\boldsymbol{x}_{o2} \leftarrow \boldsymbol{x}_{B,g}$
15: **end if**
16: **until** Stopping criterion (on $\boldsymbol{X}_B$) has been reached
17: Generate new GMM using phenotypic and genotypic data from all re-evaluated solutions from all levels (including optimization history of level 4) and update STM and LTM with new GMM and $\boldsymbol{p}_{g^*,s2}$.



(a) (b)

Fig. 6. Bi-tonal logos used as watermarks: (a) BancTec, and (b) Université du Québec.

(a) (b) (c) (d)

Fig. 7. Examples of document images from OULU-1999-TRAIN: (a) image 1, (b) image 2, (c) image 5, and (d) image 6.

5. Experimental methodology

For proof-of-concept simulations, two watermarks are employed in all experiments for all databases as in Vellasques et al. (2011), Vellasques et al. (2012): the 26 × 36 resolution BancTec logo (Fig. 6a) as robust watermark and the 36 × 26 resolution Université du Québec logo (Fig. 6b) as fragile watermark.

The experiments were conducted using the University of Oulu's MediaTeam (Sauvola and Kauniskangas, 1999) (OULU-1999) document image database, which is considerably heterogeneous. The same protocol was followed as in Vellasques et al. (2012): the images were binarized and 15 of the 512 images were discarded because they have less than 1872 flippable pixels (Muharemagic, 2004; Wu and Liu, 2004) which is the minimum required to embed the watermarks presented above. Then, the 497 images were randomly split into a training set containing 100 images (OULU-1999-TRAIN), and test set, containing 397 images (OULU-1999-TEST). Fig. 7 shows some examples from the OULU-1999-TRAIN database. Two more homogeneous databases: TITI-61 and CVIU-113–3-4 containing respectively 61 and 342 binarized pages from issues 113(3) and 113(4) of the Computer Vision and Image Understanding journal as described in Vellasques et al. (2011) were employed. A database – named SHUFFLE – comprising images from both Oulu and CVIU databases, but with their positions shuffled was also employed.

The proposed approach was evaluated for optimization of embedding parameters for a bi-tonal watermarking system by considering four main situations: (1) no attack; (2) cropping 1%; (3) cropping 2%; (4) salt and pepper with intensity 0.02.

The technique described in Vellasques et al. (2012) was applied to OULU-1999-TRAIN, TITI-61 data and a combination of both. Simulations were conducted based on the four situations described above in order to create the memories for the DS-DPSO simulations. These were conducted on the OULU-1999-TEST, CVIU-113–3-4 and SHUFFLE streams in order to validate four different cases. These cases will be useful in answering four research questions:

1. Can a surrogate learned on previous streams of optimization help to decrease the computational cost of optimization in new streams with significant variations across the problems (cases I and III)?
2. Will such learned memory of surrogates have a negative impact in performance compared to the approach proposed in Vellasques et al. (2012) for more stable streams of optimization problems (case II)?
3. Does the proposed technique scale to other types of attacks (case IV)?

Addressing these questions is an important issue in what regards applying the proposed approach to real world applications. A practical intelligent watermarking system must handle streams of hundreds/thousands of document images in feasible time (hours rather than days). It must also adapt to variations in document structure as well as to variations in the types of attacks applied on attack modeling without the need of human intervention. These questions (and their respective cases) are related since question 2 implies that question 1 is true while question 3 implies that the other two questions are also true.

5.1. Case I – adaptation performance

Tackling adaptation in scenarios involving significant variations in the stream of optimization problems is the motivation behind the proposed approach. In order to validate adaptability, the memory of OULU-1999-TRAIN is employed with no attack as a starting point for OULU-1999-TRAIN with cropping of 2%. Next, the resulting memory for OULU-1999-TRAIN is employed with salt and pepper 0.02. Finally, the resulting memory is employed as starting point in four separate scenarios for OULU-1999-TEST: (I) no attack, (II) cropping of 2%, (III) salt and pepper with intensity of 0.02, (IV) randomly chosen attacks (no attack, cropping 2%, salt and pepper 0.02) and (IVa) same as IV but without the use of a memory of previous cases of optimization (to validate the impact of previous knowledge in such challenging scenario).

5.2. Case II – comparison to previous DPSO approach (Vellasques et al., 2012):

In order to evaluate the performance of the proposed approach in a more stable scenario, simulations are performed using no attack and cropping of 1% on all streams. Simulations with and without the use of a previous memory are performed for the test streams in order to assess the impact of a previous memory in the performance of the proposed approach.

5.3. Case III – memorization capacity

In the memorization experiment, the memory management mechanism is de-activated first (all re-optimizations result in a LTM insert operation) in order to avoid any possible bias caused by the merge operators (memory management is resumed in the test phase). Then, a memory is created by applying the proposed technique with both, level 3 and surrogate of level 4 de-activated to OULU-1999-TRAIN with cropping of 1%. A more restrictive confidence level (α_{Crit}) of 0.8 during training is proposed for this particular case in order to obtain high fidelity probes (we propose a less restrictive confidence level of 0.95 for all the other simulations). Then, a probe from OULU-1999-TRAIN is chosen and has its performance evaluated for the off-line and on-line surrogate mechanisms (on OULU-1999-TRAIN as well) in two situations: (1) for cases where the selected probe resulted in a successful recall; (2) for cases where re-optimization was triggered. The motivation for this experiment is to understand the impact of previous knowledge in the computational cost of a given optimization task and also to understand at what point previous knowledge can be helpful when the new problem is knowingly different from any previous problem.

5.4. Case IV – management of different attacks

To validate how well the proposed approach can tackle other attacks, we created two other memories using OULU-1999-TRAIN: one using cropping of 2% and another one using salt and pepper with 0.02 intensity. Then, these memories are employed in OULU-1999-TEST for the same two attacks. We also evaluated the performance of the proposed approach without the use of a previous memory on both, the OULU-1999-TRAIN and OULU-1999-TEST streams.

5.5. Parameters values

In the first two levels, 19 solutions are re-sampled and are re-evaluated along with the global best for change detection. DPSO parameters for levels 3 and 4 are set as in Vellasques et al. (2011). 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.

The number of solutions employed in the evolution control for level 3 (N_{s1}) was set to 6 which corresponds to 30% of the population. The constant ρ_c defines the trade-off between exploitation and exploration for the surrogate and was set to 1. The LTM size

Table 1
Parameters employed in most of the simulations.

Parameter	Description	Value
α_{Crit}	Confidence level	0.95
γ^0	Initial learning rate	0.02
d_γ	Learning rate decay	0.99
$\|X\|$	Population size	20
N_{s1}	Evolution control population size	6
ρ_c	Surrogate exploration/exploitation trade-off	1
c_1	Acceleration constant 1	2.05
c_2	Acceleration constant 2	2.05
T	Number of previous re-optimizations to compute the insert/update threshold	10
χ	Constriction factor	0.7298

Table 2
Average computational cost performance. *AFPI* is the average number of fitness evaluations per image. Values in parentheses are standard deviation. 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	Full PSO		GMM-based			DS-DPSO		
		AFPI	F_{Evals}	AFPI	F_{Evals}	DFE (%)	AFPI	F_{Evals}	DFE (%)
Cropping 2%	OULU-1999-TRAIN	860 (335)	86040	135 (297)	13500	84.3	99 (298)	9903	88.5
S&P 0.02		893 (354)	89280	184 (417)	18360	79.4	160 (309)	15990	82.1
No attack (I)	OULU-1999-TEST	1007 (341)	399840	112 (278)	44600	88.9	85 (171)	33649	91.6
Cropping 2% (II)		828 (309)	328900	72 (187)	28400	91.4	59 (166)	23283	92.9
S&P 0.02 (III)		978 (379)	388220	123 (300)	49020	87.4	79 (218)	31366	91.9
Random (IV)		951 (344)	377640	138 (307)	54880	85.5	123 (244)	48709	87.1
Random (IVa)		951 (344)	377640	221 (410)	87720	76.3	149 (287)	58979	84.1

Table 3
Average watermarking performance, where the mean μ and standard deviation σ of each metric are presented as $\mu(\sigma)$.

Attack	Database	Full PSO			GMM-based			DS-DPSO		
		DRDM	BCR robust	BCR fragile	DRDM	BCR robust	BCR fragile	DRDM	BCR robust	BCR fragile
Cropping 2%	OULU-1999-TRAIN	0.04 (0.05)	98.2 (2.7)	99.9 (0.4)	0.04 (0.05)	97.0 (3.6)	99.7 (1.0)	0.05 (0.05)	96.5 (4.9)	99.9 (0.5)
S&P 0.02		0.03 (0.03)	97.9 (2.6)	99.7 (0.5)	0.03 (0.04)	97.3 (3.6)	99.5 (1.2)	0.03 (0.03)	96.9 (4.6)	99.5 (1.0)
No attack (I)	OULU-1999-TEST	0.00 (0.00)	100 (0.0)	100 (0.0)	0.01 (0.02)	99.9 (0.1)	99.9 (0.1)	0.00 (0.02)	100 (0.0)	100 (0.0)
Cropping 2% (II)		0.04 (0.04)	98.0 (3.0)	99.8 (0.7)	0.04 (0.05)	93.3 (6.0)	99.1 (2.0)	0.05 (0.06)	95.8 (6.1)	99.7 (1.3)
S&P 0.02 (III)		0.03 (0.04)	98.0 (2.4)	99.6 (0.6)	0.04 (0.04)	97.1 (3.7)	99.3 (1.1)	0.03 (0.04)	97.4 (3.4)	99.4 (1.0)
Random (IV)		0.02 (0.03)	98.7 (2.3)	99.8 (0.6)	0.03 (0.04)	97.3 (4.3)	99.4 (1.4)	0.03 (0.05)	97.8 (4.1)	99.7 (0.6)
Random (IVa)		0.02 (0.03)	98.8 (2.1)	99.8 (0.6)	0.03 (0.04)	97.6 (3.7)	99.6 (1.0)	0.02 (0.03)	98.0 (3.5)	99.7 (0.8)

was limited to 20 probes. In all cases, the DPSO stops optimization if the global best has not improved for 20 generations. The number of previous cases of re-optimizations employed in order to compute the insert/update threshold (T) was set to 10. In level 4, surrogate-based DPSO is performed for each generation of exact fitness DPSO. The neighborhood size for the DPSO approach (and for the comparison in the on-line surrogate update) was set to 3. The learning rate of the GMM update technique (γ) was set to 0.02 at the beginning of each re-optimization and decreased for each sample ($\gamma^t = d_\gamma\, \gamma^{t-1}$) where $d_\gamma = 0.99$ is the learning rate decay. Table 1 summarizes the parameters employed in the simulations.

6. Simulation results

6.1. Case I – adaptation performance

The simulations involving adaptation over heterogeneous streams (Tables 2 and 3) show the main advantage of the proposed DS-DPSO approach. Since adaptation involves a more restrictive confidence level (0.8) which leads to more re-optimizations, the surrogate optimizers become more dominant than for homogeneous streams.

In the first transition (OULU-1999-TRAIN with no attack to OULU-1999-TRAIN with cropping 2%), the proposed approach allowed substantial decrease in computational burden. In the 8 times re-optimization was triggered, the off-line surrogate allowed an improvement in three cases, avoiding costly on-line optimization. For this reason, the total number of fitness values suffered decreased of 26.6% compared to the GMM-based approach (from 13500 to 9903).

The same improvement in computational performance was noticed for the second transition (to OULU-1999-TRAIN with salt and pepper 0.02). This time, off-line surrogate optimization was enough for 5 of the 14 cases of re-optimization. This led to a decrease of 12.9% in the number of fitness evaluations compared to the GMM-based approach (from 18360 to 15990). It is worth noticing that such decrease was made possible despite a higher number of re-optimizations for the DS-DPSO approach (14 versus 12). The same phenomenon was repeated for the OULU-1999-TEST with cropping of 2% (a decrease of 24.6%), salt and pepper 0.02 (a decrease of 36%).

In all cases, DS-DPSO had a smaller computational burden when compared to the previous approach while the watermarking performance was practically the same.

Table 4
Average cost performance. *AFPI* is the average number of fitness evaluations per image. Values in parentheses are standard deviation. 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		GMM-based			DS-DPSO		
			AFPI	F_{Evals}	AFPI	F_{Evals}	DFE (%)	AFPI	F_{Evals}	DFE (%)
No attack	OULU-1999-TRAIN	No	925 (286)	92520	66 (194)	6580	92.9	98 (243)	9809	89.4
	OULU-1999-TEST	No	1007 (341)	399840	59 (188)	23280	94.2	56 (177)	22153	94.5
	OULU-1999-TEST	Yes	1007 (341)	399840	42 (133)	16700	95.8	62 (204)	24489	93.9
	TITI-61	No	844 (226)*	51460*	84 (224)	5140	92.6	130 (336)	7910	88.7
	CVIU-113–3-4	No	882 (251)*	301580*	76 (233)	26000	91.4	75 (183)	22066	93.9
	CVIU-113–3-4	Yes	882 (251)*	301580*	49 (157)	16600	95.4	41 (121)	14090	96.1
	SHUFFLE	No	1026 (345)	758500	66 (189)	48840	93.6	51 (137)	37986	95
	SHUFFLE	Yes	1026 (345)	758500	54 (179)	40220	94.7	56 (130)	41129	94.6
Cropping 1%	OULU-1999-TRAIN	No	887 (340)	88740	179 (363)	17860	79.9	71 (205)	7139	92
	OULU-1999-TEST	No	860 (310)	341520	83 (212)	32920	90.4	66 (194)	26104	92.4
	OULU-1999-TEST	Yes	860 (310)	341520	67 (205)	26760	92.2	48 (87)	18890	94.5
	TITI-61	No	911 (237)*	55580*	52 (178)	3200	94.8	58 (189)	3553	94.2
	CVIU-113–3-4	No	872 (251)*	298100*	50 (166)	16980	94.5	58 (182)	19708	93.6
	CVIU-113–3-4	Yes	897 (310)	306860	21 (4)	7120	97.7	21 (4)	7080	97.7
	SHUFFLE	No	887 (320)	798100	67 (194)	49780	93.8	52 (138)	38155	95.2
	SHUFFLE	Yes	887 (320)	798100	49 (136)	36300	95.5	32 (52)	23690	97

* An asterisk indicates results extracted from Vellasques et al. (2011)

Table 5
Average watermarking performance, where the mean μ and standard deviation σ of each metric are presented as $\mu(\sigma)$.

Attack	Database	Learning	Full PSO			GMM-based			DS-DPSO		
			DRDM	BCR robust	BCR fragile	DRDM	BCR robust	BCR fragile	DRDM	BCR robust	BCR fragile
No attack	OULU-1999-TRAIN	No	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
	OULU-1999-TEST	No	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
	OULU-1999-TEST	Yes	0.00 (0.00)	100 (0.00)	100 (0.0)	0.00 (0.00)	99.9 (0.4)	99.9 (0.7)	0.00 (0.00)	100 (0.0)	100 (0.0)
	TITI-61	No	0.00 (0.00)*	99.9 (0.5)*	99.7 (0.6)*	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
	CVIU-113–3-4	No	0.00 (0.00)*	99.5 (3.6)*	99.3 (3.0)*	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
	CVIU-113–3-4	Yes	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
	SHUFFLE	No	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
	SHUFFLE	Yes	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)	0.00 (0.00)	100 (0.0)	100 (0.0)
Cropping 1%	OULU-1999-TRAIN	No	0.03 (0.03)	98.4 (2.1)	99.7 (0.6)	0.03 (0.03)	97.1 (3.8)	99.4 (1.0)	0.03 (0.04)	96.6 (4.6)	99.3 (1.2)
	OULU-1999-TEST	No	0.03 (0.04)	98.4 (2.2)	99.6 (0.6)	0.03 (0.03)	96.7 (4.0)	99.1 (1.5)	0.03 (0.05)	96.1 (4.9)	98.9 (1.7)
	OULU-1999-TEST	Yes	0.03 (0.03)	98.4 (2.2)	99.6 (0.6)	0.03 (0.04)	97.5 (3.3)	99.4 (1.1)	0.03 (0.04)	96.9 (4.0)	99.3 (1.0)
	TITI-61	No	0.00 (0.00)*	92.0 (6.5)*	94.0 (4.0)*	0.03 (0.03)	99.0 (1.8)	99.7 (0.4)	0.02 (0.03)	99.1 (1.6)	99.7 (0.4)
	CVIU-113–3-4	No	0.00 (0.00)*	89.6 (7.1)*	92.5 (5.3)*	0.04 (0.05)	98.3 (3.0)	99.5 (0.8)	0.03 (0.06)	98.3 (3.6)	99.4 (0.9)
	CVIU-113–3-4	Yes	0.02 (0.04)	98.8 (2.3)	99.6 (0.4)	0.04 (0.06)	98.1 (3.0)	99.4 (1.0)	0.04 (0.04)	98.0 (3.7)	99.4 (1.0)
	SHUFFLE	No	0.03 (0.04)	98.6 (2.2)	99.6 (0.5)	0.03 (0.04)	97.1 (4.4)	98.9 (1.8)	0.03 (0.05)	96.9 (4.8)	99.0 (1.5)
	SHUFFLE	Yes	0.03 (0.04)	98.6 (2.2)	99.6 (0.5)	0.03 (0.04)	97.1 (4.3)	99.1 (1.4)	0.03 (0.04)	97.6 (3.1)	99.3 (0.8)

* An asterisk indicates results extracted from Vellasques et al. (2011).

6.2. Case II – comparison to previous DPSO approach (Vellasques et al., 2012)

In terms of computational burden, the DS-DPSO approach resulted in improvement for most cases (Table 4). All this, with a comparable precision (Table 5).

Fig. 8a to h show the computational cost for the recall, off-line and on-line levels (no attack) compared to full optimization while Fig. 9a–h shows the same but for the cropping 1% simulations.

6.2.1. Heterogeneous streams

For the OULU-1999-TEST stream with no attack with training, re-optimization was triggered 14 times. The on-line surrogate was triggered in 12 of these cases which is twice the number of re-optimizations for the GMM-based approach. For this reason, there was an increase of 46.6% in the computational burden compared to the GMM-based approach. Yet, it is important to notice however that the levels 3 and 4 have a smaller computational burden than completely reseting the swarm (Fig. 8c).

For the SHUFFLE stream with no attack with training, re-optimization was triggered 18 times (versus 16 for the GMM-based approach) and the off-line surrogate replaced the more expensive on-line surrogate for three of these cases. The proposed approach was 2.3% costlier than the GMM-based approach. But again, it is worth noticing that in average, levels 3 and 4 are still less expensive than completely resetting the swarm (see Fig. 8h).

It is possible to observe that for the "no attack" case, the use of a training sequence was not helpful since, for both, OULU-1999-TEST and SHUFFLE streams, there was even a slight increase in the number of fitness evaluations when a training sequence was employed. It is also worth noticing that for the OULU-1999-TRAIN stream, the performance of the proposed approach was even worse than that of the GMM-based approach.

The OULU-1999-TEST with cropping 1% resulted in 8 re-optimizations (versus 16 for the GMM-based approach). The off-line surrogate was enough in 3 of these cases. Combined, the smaller number of re-optimizations and use of surrogates allowed a decrease of 29.4% in the number of fitness evaluations (from 26760 to 18890) compared to the GMM-based approach.

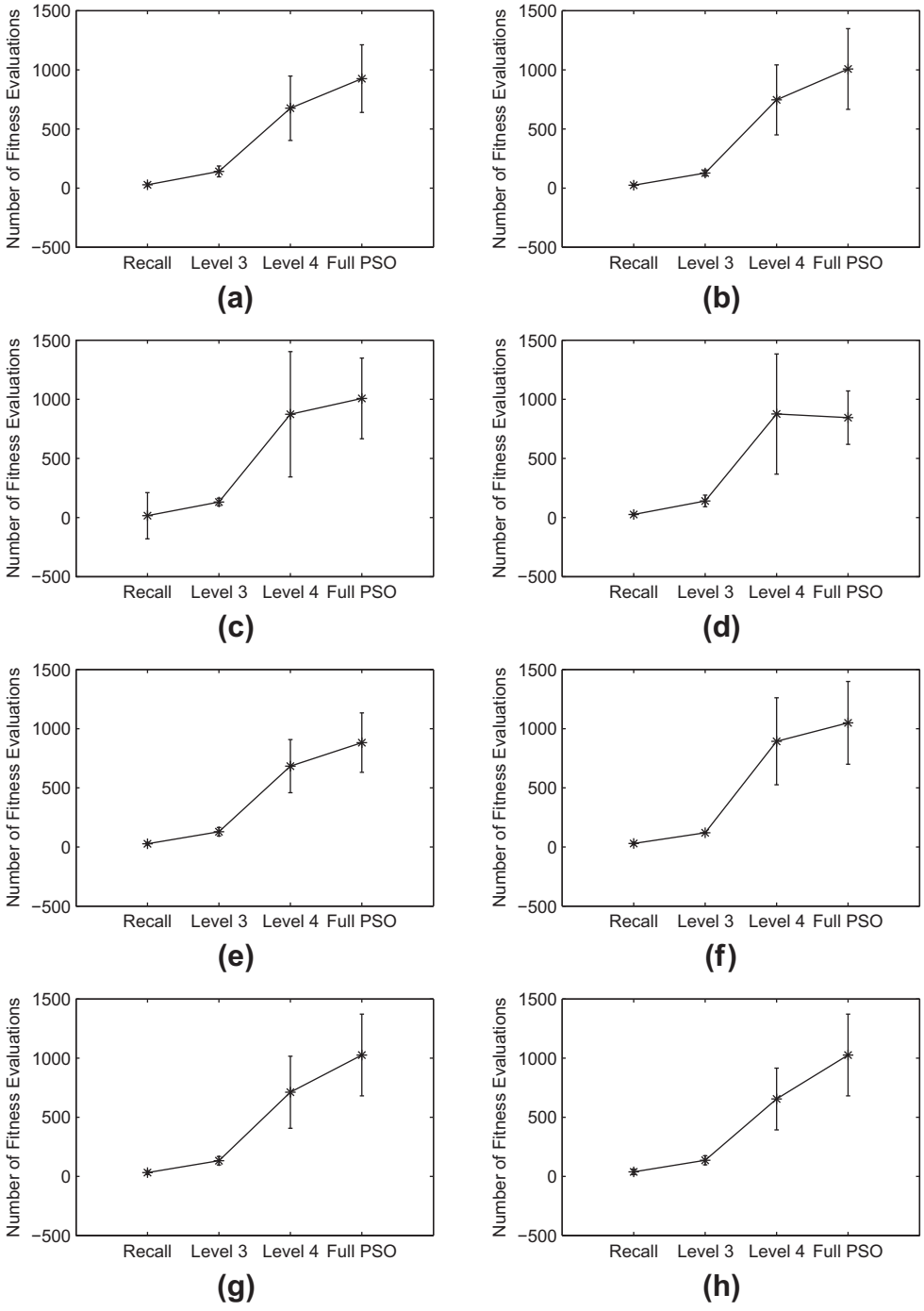


Fig. 8. Breakdown of computational cost for the "no attack" simulations (compared to full optimization): (a) OULU-1999-TRAIN, no training; (b) OULU-1999-TEST, no training; (c) OULU-1999-TEST, training; (d) TITI-61, no training; (e) CVIU-113–3-4, no training; (f) CVIU-113–3-4, training. (g) SHUFFLE, no training. (h) SHUFFLE, training.

The SHUFFLE stream with cropping 1% resulted in a single re-optimization (versus 17 for the GMM-based approach). This led to a decrease of 34.7% in the number of fitness evaluations between both techniques in the given scenario.

In the cropping 1% case, the use of a memory of previous solutions affected the computational cost positively. For the OULU-1999-TEST stream, the use of a training sequence led to a decrease of 27.6% in the number of fitness evaluations (from 26104 to 18890) while for the SHUFFLE stream the use of a training sequence led to a decrease of 37.9% (from 38155 to 23690). This time, the computational burden of the proposed approach for the OULU-1999-TRAIN stream was smaller than that of the previous approach.

6.2.2. Homogeneous streams

As observed in Tables 4 and 5, the proposed technique performance for the CVIU-113–3-4 stream with no attack resulted in a decrease of 15% in the number of fitness values (14090 versus 16600) compared to the GMM-based approach at an equivalent watermarking performance. Re-optimization was triggered 4 times (versus 7 for the GMM-based approach) and in all cases led to level 4 of the approach. For the cropping 1% case, optimization was not

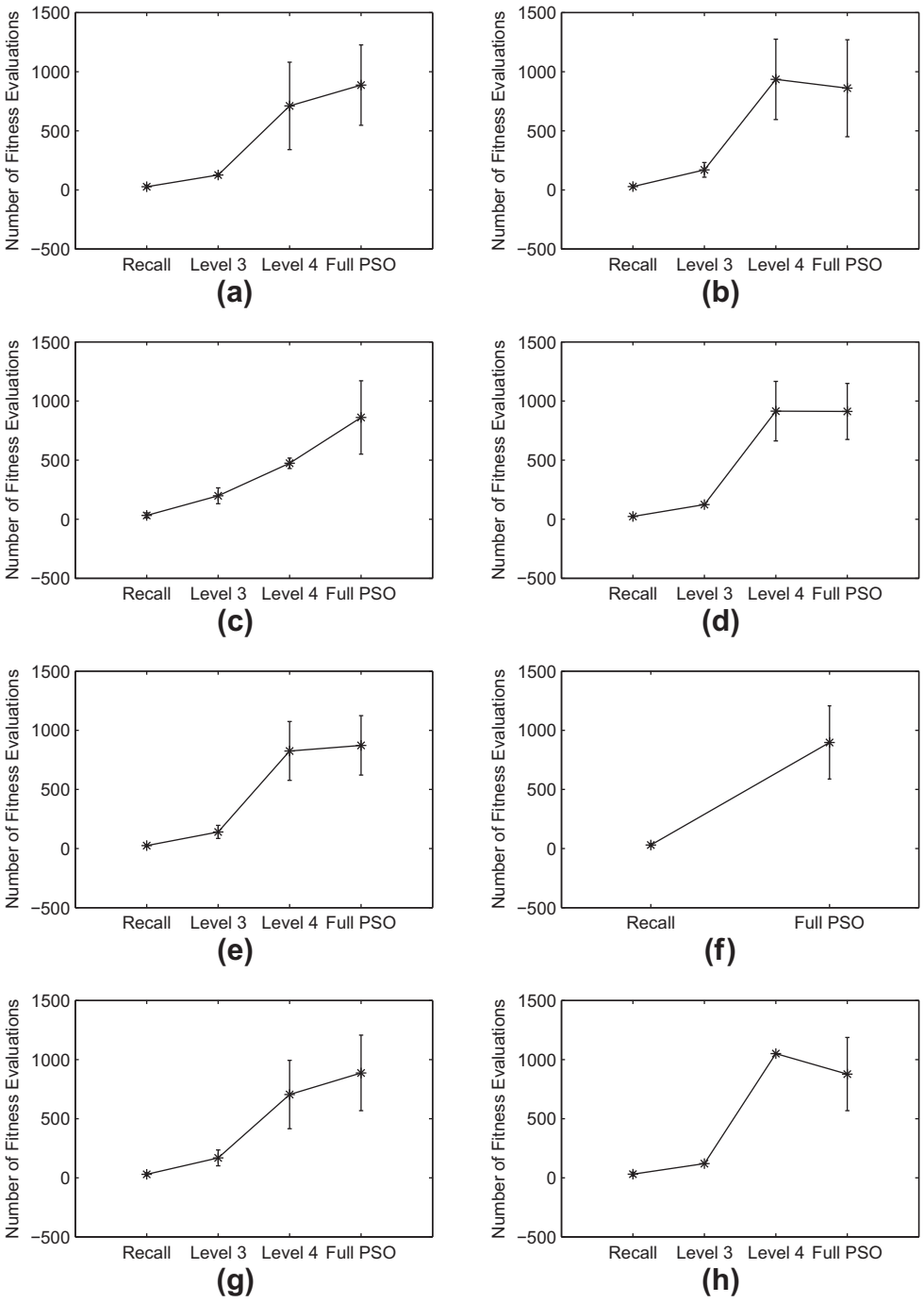


Fig. 9. Breakdown of computational cost for the cropping 1% simulations (compared to full optimization): (a) OULU-1999-TRAIN, no training; (b) OULU-1999-TEST, no training; (c) OULU-1999-TEST, training; (d) TITI-61, no training; (e) CVIU-113–3-4, no training; (f) CVIU-113–3-4, training; (g) SHUFFLE, no training; (h) SHUFFLE, training.

triggered at all (as for the GMM-based approach) therefore the computational burden performance of both approaches was nearly identical in this case.

For the no attack case, the use of a training sequence led to a decrease in the number of fitness evaluations for the proposed approach. As for the heterogeneous streams, the proposed approach performed worse than the previous approach for shorter streams.

6.3. Case III – memorization capacity

Re-optimization was triggered 21 times in training mode (OULU-1999-TRAIN). A probe was picked and tested against a set of 23 positive images (which resulted in successful recall for that probe) and a set of negative images (which resulted in re-optimization).

Table 6 shows the computational cost performance (exact fitness evaluations) for surrogate-based optimization versus no surrogate full optimization in both cases (positive and negative). It is possible to observe that the off-line surrogate resulted in a considerable decrease in the number of fitness evaluations. It is also worth noticing that for the on-line surrogate, although the fitness evaluations are performed primarily on the images, there was still a considerable decrease in the number of exact fitness evaluations which shows that the surrogate increases the convergence speed of the main population.

Table 6
Average computational cost performance (surrogate optimization). *AFPI* is the average number of fitness evaluations per image. Values in parentheses are standard deviation. 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	Type	No surrogate		Off-line			On-line		
		AFPI	F_{Evals}	AFPI	F_{Evals}	DFE (%)	AFPI	F_{Evals}	DFE	AFPI (%)
Cropping 1%	OULU-1999-TRAIN	Positives	1059 (362)	24,360	212 (33)	4878	80	955 (349)	17,191	29.4
		Negatives	941 (336)	16,940	212 (45)	3810	77.5	874 (376)	20,110	−18.7

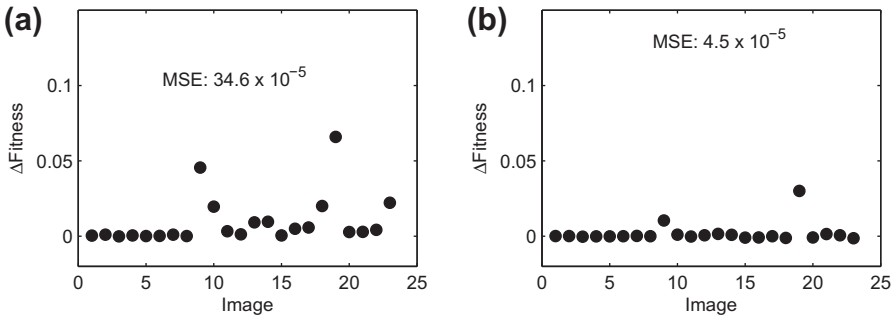


Fig. 10. Surrogate optimization performance for positive images: (a) off-line surrogate; (b) on-line surrogate.

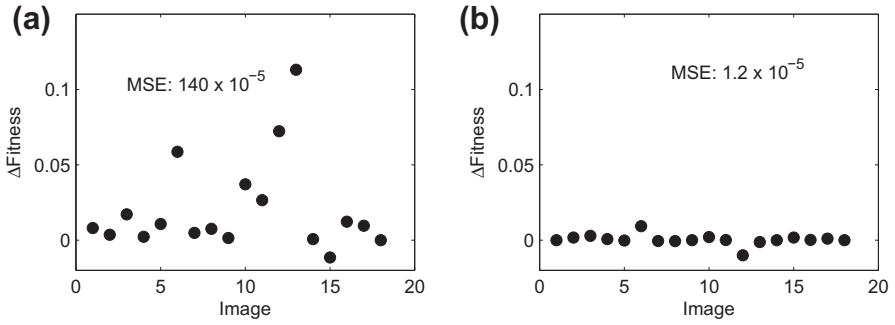


Fig. 11. Surrogate optimization performance for negative images: (a) off-line surrogate; (b) on-line surrogate.

Fig. 10 shows the difference between the fitness values (ΔFitness) of full optimization (no surrogate) and surrogate-based optimization (both, off-line and on-line), for each of the positive images. The off-line surrogate (Fig. 10a) resulted in a slight fitness degradation for a few images, but for most of them, the fitness values were quite similar to those obtained in full optimization. For the on-line surrogate instead (Fig. 10b), it was possible to observe even a slight improvement in the fitness values for some images.

Fig. 11 shows ΔFitness for the negative images. Here it is possible to observe a greater instability for the off-line surrogate (Fig. 11a) while the on-line surrogate (Fig. 11b) resulted in a similar performance to that observed for the positive images (as before, there was even an improvement in performance for some images). This demonstrates that as expected, the on-line surrogate is more sensitive to prior knowledge than the off-line surrogate. But it is also worth noticing that the fitness performance was quite good for some images of the off-line surrogate (despite being negative images). This justifies the dual surrogate approach.

It is important to observe that the Mean Squared Error (MSE) between the fitness values obtained in full optimization and in the proposed approach are negligible. However, for both subsets, the MSE obtained for the off-line surrogate is greater than that obtained for the on-line surrogate. It is also worth noticing a considerable deterioration in MSE between the positive and negative images. This justifies the use of an on-line surrogate as a safeguard.

6.4. Case IV – management of different attacks

The performance for cropping 2% and salt and pepper 0.02 (Tables 7 and 8) was compatible with that of cropping 1%.

In the case of cropping 2% (OULU-1999-TEST with learning), re-optimization was triggered twice. Only one of these cases required the on-line surrogate (level 4). The number of fitness evaluations suffered a decrease of 7.4% (from 19800 to 18339) when compared to the GMM-based approach.

For the salt and pepper 0.02 (OULU-1999-TEST with learning), re-optimization was triggered once. However, this single case was costlier than full reset (1365 versus 1000 fitness evaluations) as the off-line surrogate did not result in an improvement.

6.5. Discussion

Overall, the simulation results demonstrated that the off-line surrogate allows a considerable decrease in the number of fitness evaluations for the cases where re-optimization is triggered. The on-line surrogate operates as a safeguard for the whole system.

Table 7
Average computational cost performance. *AFPI* is the average number of fitness evaluations per image. Values in parentheses are standard deviation. 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		GMM-based			DS-DPSO		
			AFPI	F_{Evals}	AFPI	F_{Evals}	DFE (%)	AFPI	F_{Evals}	DFE (%)
Cropping 2%	OULU-1999-TRAIN	No	860 (335)	86040	72 (187)	7240	91.6	77 (204)	7684	91.1
	OULU-1999-TEST	No	828 (309)	328,900	64 (179)	25,560	92.2	77 (199)	30747	90.7
	OULU-1999-TEST	Yes	828 (309)	328,900	50 (150)	19,800	94	46 (84)	18,339	94.4
S&P 0.02	OULU-1999-TRAIN	No	893 (354)	89,280	163 (360)	16,320	81.7	121 (308)	12,055	86.5
	OULU-1999-TEST	No	978 (379)	388,220	92 (281)	36,360	90.6	59 (161)	23,327	94
	OULU-1999-TEST	Yes	978 (379)	388,220	42 (133)	16,560	95.7	33 (54)	13,183	96.6

Table 8
Average watermarking performance, where the mean μ and standard deviation σ of each metric are presented as $\mu(\sigma)$.

Attack	Database	Learning	Full PSO			GMM-based			DS-DPSO		
			DRDM	BCR robust	BCR fragile	DRDM	BCR robust	BCR fragile	DRDM	BCR robust	BCR fragile
Cropping 2%	OULU-1999-TRAIN	No	0.04 (0.05)	98.2 (2.7)	99.9 (0.4)	0.04 (0.06)	97.1 (3.8)	99.8 (0.6)	0.04 (0.05)	96.5 (4.6)	99.7 (0.9)
	OULU-1999-TEST	No	0.04 (0.04)	98.0 (3.0)	99.8 (0.7)	0.04 (0.04)	95.4 (5.7)	99.3 (2.0)	0.04 (0.05)	94.7 (7.4)	99.1 (2.0)
	OULU-1999-TEST	Yes	0.04 (0.04)	98.0 (3.0)	99.8 (0.7)	0.04 (0.05)	94.7 (6.4)	99.1 (1.9)	0.04 (0.06)	96.0 (5.2)	99.8 (0.8)
S&P 0.02	OULU-1999-TRAIN	No	0.03 (0.03)	97.9 (2.6)	99.7 (0.5)	0.03 (0.03)	97.1 (4.3)	99.3 (1.3)	0.03 (0.04)	97.2 (3.4)	99.5 (1.0)
	OULU-1999-TEST	No	0.03 (0.04)	98.0 (2.4)	99.6 (0.6)	0.03 (0.04)	97.2 (3.6)	99.4 (1.0)	0.03 (0.04)	96.7 (4.5)	99.3 (1.1)
	OULU-1999-TEST	Yes	0.03 (0.04)	98.0 (2.4)	99.6 (0.6)	0.03 (0.04)	97.1 (4.0)	99.2 (1.2)	0.03 (0.04)	96.3 (4.3)	99.1 (1.3)

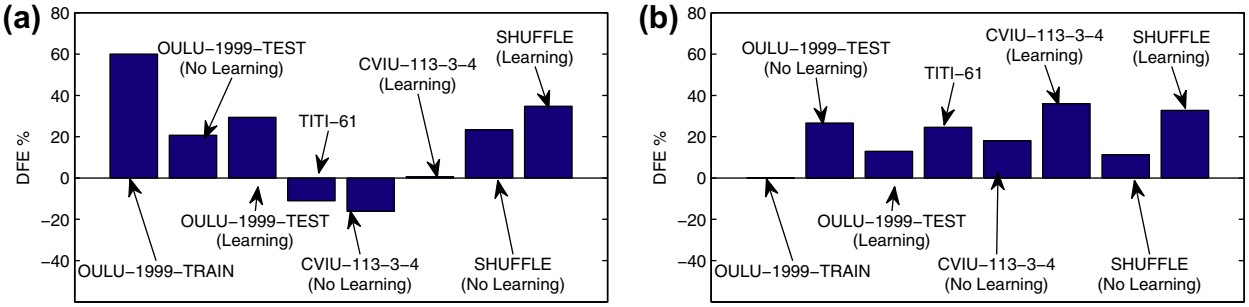


Fig. 12. Decrease in fitness evaluations (DFE) of DS-DPSO versus GMM-based approach: (a) cropping 1%, no adaptation; (b) adaptation simulations.

Since the objective of the on-line surrogate is to improve convergence speed of the population of the exact fitness function, it can be said that its efficiency is tied to how inefficient is the main population. For this reason, in some cases, when the fourth level was required, it implied in a larger computational burden than full reset. But it is important to remark that this cost also involves the cost of the previous three levels. Therefore, it can be said as a safeguard, the use of a surrogate is preferred to simply using full reset.

The adaptation on more heterogeneous streams simulations demonstrated the main advantage of the proposed approach. In situations involving substantial variability in the stream of optimization problems, the number of re-optimizations is expected to increase considerably. In such case, replacing costly full reset by a lighter surrogate-based optimization becomes crucial. The off-line surrogate was enough in numerous cases in such scenario, allowing even a more substantial decrease in computational burden compared to the more stable scenarios. This advantage can be seen more clearly in Fig. 12.

For cases of re-optimization (heterogeneous streams), the off-line surrogate successfully replaced the heavier on-line surrogate in numerous situations. Moreover, in many of the cases where it failed, the on-line surrogate gave a boost to the convergence speed of the main swarm, resulting in a further decrease (despite the last resort nature of the fourth level). It was observed that for the no attack case, the use of a memory of previous solutions is irrelevant in what regards decreasing the computational burden. The reason is that the memory in such case is a tool to aid adaptation to novel cases of optimization. Thus, it becomes less important in situations involving little adaptation as the no attack case.

In the memorization simulations, it was possible to observe in general that previous knowledge plays an important role in the performance of the off-line surrogate. It was also possible to observe that for the negative examples (which represent the exact situation in which the surrogate based-optimization is expected to work) the off-line surrogate still allows good watermarking performance for a small fraction of the computational burden of full optimization with no surrogate. And yet, the on-line surrogate works as a safety net for the whole system but also with a smaller computational burden than full optimization with no surrogate.

Finally, the simulations involving homogeneous streams showed one limitation of the proposed approach. The gain obtained by the surrogate-based approach is limited by the number of re-optimizations. And the number of re-optimizations depends on probe precision. Since solutions obtained during the course of optimization are employed in order to create a probe, probe precision varies depending on the amount of novelty brought by these solutions. Because of their smaller computational burden, memory recall operations are preferred over re-optimization. However, in a case of re-optimization, the use of a surrogate allows a considerable decrease in computational burden compared to full reset.

These experimental results support our strategy of employing two surrogates with different trade-offs between fidelity and computational burden rather than focusing on the detail level (global or local) of each surrogate. It also shows that the use of a memory of surrogates, trained with a separate set of images, contributes even further to the performance of the dual surrogate. Since it was observed that the use of a memory stream is irrelevant for small and stable streams, employing the previous approach is recommended in order to create a memory (using a training stream) and then, employing the proposed approach for larger streams, in situations requiring adaptability.

6.6. Guidelines for practical implementation of the proposed technique

The main objective behind both, the memory recall and the dual surrogate mechanisms is to decrease the computational cost of intelligent watermarking and thus, make it feasible in real world applications. Therefore the proposed approach already was conceived based on a practical application. In terms of scale, it takes between 40 min and 3:30 h to tackle intelligent watermarking on the streams described before using the proposed technique versus 3 to 30 h using full optimization (on a cluster containing 17 nodes with four 2.4 GHz CPUs and 7 GB of RAM each). Parallelization was based on a master–slave architecture. Here, a single master node sends tasks to the slave nodes. The master node is responsible for the optimization and memory management logic while the slave nodes are responsible for fitness evaluation (embedding/detecting the watermarks and fitness computation). The CPU time can be further decreased with the use of GPGPU (although such approach was not followed in this research) to tackle the image processing part of the system (watermarking, attacks), considering that a GPGPU-enabled graphics card is available in the slave nodes.

Another important issue is that the baseline watermarking technique employs a shuffling key in order to protect the watermark against unauthorized detection. Such approach is known as semi-blind watermarking where a shared secret (key) must be known at the embedder and detector. Therefore, in a distributed environment (for example, protecting document authenticity in a peer-to-peer network) the key employed on embedding must be transmitted to the detector along with the chosen embedding parameters. This is an open issue in the digital watermarking domain and the basic approach is to assume that a secure channel is available to that end. For document images, the cost associated with sending the key and embedding parameters is a fraction of that of sending the image and is therefore, irrelevant for most practical applications.

The main motivation for optimizing the embedding parameters of a digital watermarking system is to customize the robustness and quality trade-off for specific applications. For example, protecting medical documents shared over an open network implies in a different robustness/quality trade-off requirement compared to sharing historic documents over a local network. The different aspects that must be addressed for each specific application include: sets of attacks, expected watermark robustness, image quality requirements. One of the limitations of intelligent watermarking is that it can only make a watermark robust against attacks that can be tackled by manipulating heuristic parameters of the baseline watermarking system. For example, the bi-tonal watermarking system employed in this research allows increasing the robustness against removal attacks. A removal attack involves applying image processing operations such as cropping and filtering in order to destroy the embedded watermark. However, robustness against geometric attacks cannot be tackled in the same manner. A geometric attack impairs the synchronization between the original and the embedded watermark. Wu and Liu (2004) propose the use of synchronization marks in order to recover from geometric (rotation, scaling and translation) transforms.

7. Conclusion

In this paper, a multi-level intelligent watermarking system was proposed. This system is based in four levels. Each level increases the precision of the preceding level at the cost of higher computational burden. The first two levels, as defined in a previous research, comprise memory recall. These two levels allow matching new optimization problems to previously seen problems stored in a memory of GMMs and recalling ready-to-use solutions for similar problems. The other two levels (3 and 4) are optimization levels and are only activated when the recall modules fail (if embedding parameters require a significant adaptation). During optimization, the most adequate GMM is employed as a surrogate, which is initially updated with the fitness values obtained during recall. The third level performs exploitation and aims at optimizing problems where the optimum is near the surrogate optimum, but could not be found during recall. The fourth level works as a safety net for the whole system, but relies on a surrogate in order to boost convergence.

This approach of using a memory of previously learned surrogates, matched to the new problem using sampling and statistical test is novel and is one of the main contributions of our research. Moreover, this idea of focusing on the trade-off between cost and fidelity of the surrogate rather than on the detail level is also novel and is a secondary contribution of our research.

Experimental results demonstrate that when previous knowledge is available, the off-line surrogate is expected to result in a fitness performance comparable to that of full optimization (with no surrogate) but at a fraction of its cost. It was also demonstrated that in a real situation where the recall failed, it will allow avoiding a more costly on-line surrogate optimization. The on-line surrogate by its way, resulted in a fitness performance that is nearly identical to that of no surrogate (even for cases where recall failed) but with a computational burden that is usually cheaper than that of no surrogate. For this reason, the proposed approach allowed computational savings of up to 93% compared to full optimization in scenarios involving heterogeneous image streams with changing attacks.

These results validate our research hypothesis that whenever re-optimization is triggered, the best fit model should provide a good starting point for building a surrogate for the new problem and even if it cannot, it could still decrease the computational burden of optimization by speeding up convergence. It also demonstrates that knowledge of a new problem can be incorporated into knowledge of previous problems in order to make the model better fit to the new problem. Finally, the results demonstrate the main advantage of the proposed approach which is tackling intelligent watermarking in scenarios involving substantial variability in the stream of optimization problems.

References

NIST/SEMATECH, NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/ (March 2010).

Areef, T. E., Heniedy, H. S., & Mansour, O. M. O. (2005). Optimal transform domain watermark embedding via genetic algorithms, *ITI 3rd international conference on information and communications technology (ICICT)* (pp. 607–617).

Calinon, S. (2009). *Robot programming by demonstration: A probabilistic approach*. EPFL/CRC Press.

Clerc, M. (2006). *Particle swarm optimization*. London: ISTE Publishing Company.

Collette, Y., & Siarry, P. (2008). On the sensitivity of aggregative multiobjective optimization methods. *CIT, 16*(1), 1–13.

Cox, I. J., Kilian, J., Leighton, T., & Shamoon, T. (1996). A secure, robust watermark for multimedia. In: *Workshop on information hiding* (pp. 1–16).

Cox, I., Miller, M., & Bloom, J. (2002). *Digital watermarking*. Morgan Kaufmann Publishers.

Dennis, J. E., & Torczon, V. (1996). Managing approximation models in optimization. In *Multidisciplinary Design Optimization: State-of-the-Art, 1995* (pp. 330–347).

El-Beltagy, M., Nair, P. B., & Keane, A. J. (1999). Metamodeling techniques for evolutionary optimization of computationally expensive problems: Promises and limitations. In *GECCO '99: Proceedings of the 8th annual conference on Genetic and evolutionary computation* (pp. 196–203).

Engel, P. M., & Heinen, M. R. (2010). Incremental learning of multivariate gaussian mixture models. In *SBIA* (pp. 82–91).

Figueiredo, M. A. T., & Jain, A. K. (2000). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*, 381–396.

Fonseca, L., Barbosa, H., & Lemonge, A. (2009). A similarity-based surrogate model for enhanced performance in genetic algorithms. *OPSEARCH, 46*, 89–107.

Gräning, L., Jin, Y., & Sendhoff, B. (2005). Efficient evolutionary optimization using individual-based evolution control and neural networks: A comparative study. In *ESANN* (pp. 273–278).

Hennig, C. (2010). Methods for merging gaussian mixture components. *Advanced Data Analysis and Classification, 4*, 3–34.

Holland, J. H. (1992). *Adaptation in natural and artificial systems*. Cambridge, MA, USA: MIT Press.

Jin, Y., Olhofer, M., & Sendhoff, B. (2000). On evolutionary optimization with approximate fitness functions. In D. W. et al. (Ed.), *Proceedings of the genetic and evolutionary computation conference GECCO*, Morgan Kaufmann (pp. 786–793).

Jin, Y., Olhofer, M., & Sendhoff, B. (2002). A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation, 6*(5), 481–494.

Kapp, M. N., Sabourin, R., Maupin, P. (2012). A dynamic model selection strategy for support vector machine classifiers, Applied Soft Computing 12(8) (accepted for publication).

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In: *IEEE International Conference on Neural Networks, Perth, Australia*.

Lu, H., Kot, A. C., & Shi, Y. Q. (2004). Distance-reciprocal distortion measure for binary document images. *IEEE Signal Processing Letters, 11*(2), 228–231.

Muharemagic, E. (2004). Adaptive two-level watermarking for binary document images, Ph.D. thesis, Florida Atlantic University (December 2004).

Pan, J., Huang, H., & Jain, L. (2004). *Intelligent watermarking techniques*. World Scientific Co.. Ch. Genetic Watermarking on Spatial Domain..

Parno, M. D., Hemker, T., & Fowler, K. R. (2012). Applicability of surrogates to improve efficiency of particle swarm optimization for simulation-based problems *Engineering Optimization* 44(5) , doi:10.1080/0305215X.2011.598521.

Pelikan, M., Goldberg, D. E., & Lobo, F. G. (2002). A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications, 21*(1), 5–20.

Praveen, C., & Duvigneau, R. (2007). Metamodel-assisted particle swarm optimization and application to aerodynamic shape optimization, Tech. Rep. 6397, INRIA.

Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., & Tucker, P. K. (2005). Surrogate-based analysis and optimization. *Progress in Aerospace Sciences, 41*(1), 1–28.

Sauvola, J., & Kauniskangas, H. (1999). MediaTeam Document Database II, a CD-ROM collection of document images, University of Oulu, Finland.

Sfikas, G., Constantinopoulos, C., Likas, A., & Galatsanos, N. P. (2005). 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* (pp. 835–840). Springer-Verlag, Berlin, Heidelberg.

Shi, L., & Rasheed, K. (2008). ASAGA: an adaptive surrogate-assisted genetic algorithm. In *GECCO '08: Proceedings of the 2008 conference on Genetic and evolutionary computation* (pp. 1049–1056).

Stauffer, C., & Grimson, W. E. L. (2000). Learning patterns of activity using real-time tracking. *IEEE Trans. Pattern Anal. Mach. Intell., 22*(8), 747–757.

Sung, H. G. (2004). Gaussian mixture regression and classification, Ph.D. thesis, Rice University.

Torczon, V., & Trosset, M. W. (1998). Using approximations to accelerate engineering design optimization. In *Proceedings of the 7th AIAA/USAF/NASA/ISSMO multidisciplinary analysis and optimization symposium, Saint Louis, USA*.

Vellasques, E., Granger, E., & Sabourin, R. (2010). Handbook of Pattern Recognition and Computer Vision, 4th ed., World Scientific Review, 2010, Ch. Intelligent Watermarking Systems: A Survey (pp. 687–724).

Vellasques, E., Sabourin, R., & Granger, E. (in press). Fast intelligent watermarking of heterogeneous image streams through mixture modeling of PSO populations. *Applied Soft Computing* (doi: 10.1016/j.asoc.2012.08.040).

Vellasques, E., Sabourin, R., & Granger, E. (2011). A high throughput system for intelligent watermarking of bi-tonal images. *Applied Soft Computing, 11*(8), 5215–5229.

Vellasques, E., Sabourin, R., & Granger, E. (2012). Gaussian mixture modeling for dynamic particle swarm optimization of recurrent problems. In *GECCO'12: Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 73–80). ACM.

Voloshynovskiy, S., Pereira, S., Pun, T., Eggers, J., & Su, J. (2001). Attacks on digital watermarks: classification, estimation based attacks, and benchmarks. *IEEE Communications Magazine, 39*(8), 118–126.

Wang, H., Wang, D., & Yang, S. (2007). Triggered memory-based swarm optimization in dynamic environments. In: *EvoWorkshops* (pp. 637–646).

Wu, M., & Liu, B. (2004). Data hiding in binary image for authentication and annotation. *IEEE Transactions on Multimedia, 6*(4), 528–538.

Yamanishi, K., Takeuchi, J., Williams, G., & Milne, P. (2000). 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* (pp. 320–324). ACM.

Yan, S., & Minsker, B. (2010). Applying dynamic surrogate models in noisy genetic algorithms to optimize groundwater remediation designs. *Journal of Water Resources Planning and Management* 137 (3), 248–292.

Yang, S., & Yao, X. (2008). Population-based incremental learning with associative memory for dynamic environments. *IEEE Transactions on Evolutionary Computation, 12*(5), 542–561.

Zhang, Y., & Scordilis, M. S. (2008). Effective online unsupervised adaptation of gaussian mixture models and its application to speech classification. *Pattern Recognition Letters, 29*(6), 735–744.

Zhou, Z., Ong, Y. S., Nair, P. B., Keane, A. J., & Lum, K. Y. (2007). Combining global and local surrogate models to accelerate evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 37*(1), 66–76.

Zielinski, K., & Laur, R. (2007). Stopping criteria for a constrained single-objective particle swarm optimization algorithm. *Informatica, 31*(1), 51–59.