Dominant Color Extraction based on Dynamic Clustering by Multi-Dimensional Particle Swarm Optimization

Serkan Kiranyaz, Stefan Uhlmann, and Moncef Gabbouj
Department of Signal Processing, Tampere University of Technology, Tampere, Finland
{serkan.kiranyaz, stefan.uhlmann, moncef.gabbouj}@tut.fi

***Abstract*—Color is the major source of information widely used in image analysis and content-based retrieval. Extracting dominant colors that are prominent in a visual scenery is of utter importance since human visual system primarily uses them for perception. In this paper we address dominant color extraction as a dynamic clustering problem and use techniques based on Particle Swarm Optimization (PSO) for finding optimal (number of) dominant colors in a given color space, distance metric and a proper validity index function. The first technique, so-called Multi-Dimensional (MD) PSO, re-forms the native structure of swarm particles in such a way that they can make inter-dimensional passes with a dedicated dimensional PSO process. Therefore, in a multidimensional search space where the optimum dimension is unknown, swarm particles can seek both positional and dimensional optima. Nevertheless, MD PSO is still susceptible to premature convergences due to lack of divergence. To address this problem we then present Fractional Global Best Formation (FGBF) technique, which basically collects all promising dimensional components and fractionally creates an artificial global-best particle (aGB) that has the potential to be a better "guide" than the PSO's native gbest particle. We finally propose an efficient color distance metric, which uses a fuzzy model for computing color (dis-) similarities over HSV (or HSL) color space. The comparative evaluations against MPEG-7 dominant color descriptor show the superiority of the proposed technique.**

Keywords-dominant color, particle swarm optimization, dynamic clustering

I. INTRODUCTION

Dominant color (DC) extraction is basically a dynamic color quantization process, which seeks for such prominent color centers that minimize the quantization error. In that, the study of human color perception and similarity measurement in the color domain become crucial and there is a wealth of research performed in this field. For example in [24], Broek et. al. focused on the utilization of color categorization (called as focal colors) for CBIR purposes and introduced a new color matching method, which takes human cognitive capabilities into account. They have exploited the fact that humans tend to think and perceive colors only in 11 basic categories. In [16], Mojsilovic et. al. performed a series of psychophysical experiments analyzing how humans perceive and measure similarity in the domain of color patterns. One observation worth mentioning here is that the human eye cannot perceive a large number of colors at the same time, nor it is able to distinguish similar (close) colors well. Based on this, they showed that at the coarsest level of judgment, human visual system (HVS) primarily uses dominant colors (i.e. few prominent colors in the scenery) to judge similarity.

The usual approach for DC extraction is to perform clustering in color domain. The most popular clustering method, which is also used for DC descriptor [17] in MPEG-7, is K-means, [9]. However, given a validity index, clustering is a multi-modal problem especially in high dimensions, which contains many sub-optimum solutions such as over- and under-clustering. Therefore, well-known deterministic methods such as K-means, Max-Min [9], FCM [9], SOM [9], etc. are susceptible to get trapped to the closest local minimum since they are nothing but greedy descent methods, which start from a random point in the solution space and perform a localized search. This fact eventually turns the focus on stochastic Evolutionary Algorithms (EAs) [1] such as Genetic Algorithms (GAs) [7], Genetic Programming (GP) [13], Evolution Strategies (ES), [2] and Evolutionary Programming (EP) [6], all of which are motivated by the natural evolution process and thus make use of evolutionary operators. The common point of all is that EAs are in population based nature and can perform a globalized search. So they may avoid becoming trapped in a local optimum and find the optimum solution; however, this is never guaranteed.

Conceptually speaking, Particle Swarm Optimization (PSO) [1], which has obvious ties with the EA family, lies somewhere in between GA and EP. Yet unlike GA, PSO has no complicated evolutionary operators such as crossover, selection and mutation. In a PSO process, a swarm of particles (or agents), each of which represent a potential solution to an optimization problem, navigate through the search space. Particles are initially distributed randomly over the search space and the goal is to converge to the global optimum of a function or a system. Several researchers have shown that PSO show better clustering performance than the aforementioned techniques [1], [18]-[20]; however, when the problem is in multi-modal nature PSO may also become trapped in local optima [21] due to the premature convergence problem especially when the search space is of high dimensions [23]. Furthermore, PSO has so far been applied to simple clustering problems [1], [18]-[20], where the data space is limited and usually in low dimensions and the number of clusters (hence the solution space dimension) is kept reasonably low (e.g. <10). Moreover, all clustering methods mentioned earlier are in static nature, that is, the number of clusters has to be specified a priori. This is also true for PSO since in its basic form it can only be applied to a search space with a fixed dimension. Particularly for

dominant color extraction, the optimal (true) number of DCs in an image is unknown and should thus be determined within the (PSO) process.

In this paper, we shall address data clustering as an optimization problem and present techniques for finding optimal (number of) clusters in a multi-dimensional data or feature space. The first technique, the so-called Fractional Global Best Formation (FGBF), is designed to alleviate the premature convergence problem by collecting all promising components from each particle and fractionally create an artificial Global Best (GB) particle, the aGB, which may guide the swarm better than the swarm's native gbest particle [1] in such a way that the swarm can converge to the global optimum (or near-optimum) solution even in high dimensions and usually in earlier stages. In order to achieve a dynamic clustering where the optimum number of clusters is also determined within the process, we shall then present the so-called Multi-Dimensional Particle Swarm Optimization (MD PSO) method, which extends the native structure of PSO particles in such a way that they can make inter-dimensional passes with a dedicated dimensional PSO process. Therefore, in a multidimensional search space where the optimum dimension is unknown, swarm particles can seek for both positional and dimensional optima. In recent works, [11] and [12], both techniques have been successfully applied for optimization over dynamic environments and for automatic design of artificial neural networks, respectively. In this work we use them for finding the optimal (number of) dominant colors in an image with respect to a (clustering) validity index function and a distance metric obtained from the proposed fuzzy model over HSV (or HSL) color space.

The rest of the paper is organized as follows. Section II surveys related work on PSO whilst presenting a brief overview on MD PSO, FGBF and data clustering. The proposed dominant color extraction technique is presented in Section III. Section IV provides the experiments conducted over a real image database and discusses the results. Finally, Section V concludes the paper.

II. Related Work

A. DC Descriptors

In order to solve the problems of static quantization in color histograms, various DC descriptors, e.g. [1], [2], [5], [17] and [17], have been developed using dynamic quantization with respect to image color content. DCs, if extracted properly according to the aforementioned color perception rules, can indeed represent the prominent colors in any image. They have a global representation, which is compact and accurate and they are also computationally efficient. MPEG-7 DC descriptor (DCD) is adopted as in [3] where the method is designed with respect to HVS color perceptual rules. For instance, HVS is more sensitive to the changes in smooth regions than in detailed regions. Thus colors are quantized more coarsely in the detailed regions while smooth regions have more importance. To exploit this fact, a smoothness weight (*w(p)*) is assigned to each pixel (*p*) based on the variance in a local window. Afterwards, the *General Lloyd Algorithm* (*GLA*, also referred to as *Linde-Buzo-Gray* and it is equivalent to the well-known *K-means* clustering method [9]) is used for color quantization. For a color cluster C_i, its centroid c_i is calculated by

$$c_i = \frac{\sum w(p)x(p)}{\sum w(p)}, \qquad x(p) \in C_i \tag{1}$$

and the initial clusters for *GLA* is determined by using a weighted distortion measure, defined as

$$D_i = \sum w(p)\|x(p) - c_i\|^2, \qquad x(p) \in C_i \tag{2}$$

This is used to determine which clusters to split until either a maximum number of clusters (DCs), $N_{DC}^{\max}$, is achieved or a maximum allowed distortion criteria, ε_D, is met. Hence, pixels with smaller weights (detailed sections) are assigned fewer clusters so that the number of color clusters in the detailed regions where the likelihood of outliers' presence is high, is therefore suppressed. As the final step, an agglomerative clustering (AC) is performed on the cluster centroids to further merge similar color clusters so that there is only one cluster (DC) hosting all similar color components in the image. A similarity threshold T_S is assigned to the maximum color distance possible between two similar colors in a certain color domain (CIE-Luv, CIE_Lab, etc.). Another merging criterion is the color area, that is, any cluster should have a minimum amount of coverage area, T_A, so as to be assigned as a DC; otherwise, it will be merged with the closest color cluster since it is just an outlier. Another important issue is the choice of the color space since a proper color clustering scheme for DC extraction tightly relies on the metric. Therefore, a perceptually uniform color space should be used and the most common ones are CIE-Luv and CIE-Lab, which are designed such that color distances perceived by HVS are also equal in L_2 (Euclidean) distance in these spaces. For CIE-Luv, a typical value for T_S is between 10 and 25, T_A is between 1-5% [17] and $\varepsilon_D < 0.05$. Based on the earlier remarks, $N_{DC}^{\max}$ can be conveniently set to 8.

B. Basic PSO Algorithm

In the basic PSO method, (*bPSO*), a swarm of particles flies through an *N*-dimensional search space where the position of each particle represents a potential solution to the optimization problem. Each particle keeps track of its position in the search space and its best solution so far achieved. This is the personal best value (the so-called *pbest* in [10]) and the PSO process also keeps track of the global best solution so far achieved by the swarm with its particle index (the so called *gbest* in [10]). So during their journey with discrete time iterations, the velocity of each agent in the next iteration is computed by the best position of the swarm (position of the particle *gbest* as the *social* component), the best personal position of the particle (*pbest* as the *cognitive* component), and its current velocity (the *memory* term). Both *social* and

cognitive components contribute randomly to the position of the agent in the next iteration.

Each particle a in the swarm, $\xi = \{x_1,..,x_a,..,x_S\}$, is represented by the following characteristics:

$x_{a,j}(t)$: j^{th} dimensional component of the position of particle a, at time t

$v_{a,j}(t)$: j^{th} dimensional component of the velocity of particle a, at time t

$y_{a,j}(t)$: j^{th} dimensional component of the personal best (*pbest*) position of particle a, at time t

$\hat{y}_j(t)$: j^{th} dimensional component of the global best position of swarm, at time t

Let f denote the fitness function to be optimized. Without loss of generality assume that the objective is to find the minimum of f in N dimensional space. Then the personal best of particle a can be updated in iteration t+1 as,

$$y_{a,j}(t+1) = \begin{cases} y_{a,j}(t) & \text{if } f(x_a(t+1)) > f(y_a(t)) \\ x_{a,j}(t+1) & \text{else} \end{cases} \forall j \in [1,N] \quad (3)$$

Since *gbest* is the index of the GB particle, then $\hat{y}(t) = y_{gbest}(t) = \min(y_1(t),..,y_S(t))$. Then for each iteration in a PSO process, positional updates are performed for each particle, $a \in [1,S]$ and along each dimensional component, $j \in [1,N]$, as follows:

$$\begin{aligned} v_{a,j}(t+1) &= w(t)v_{a,j}(t) + \\ &c_1 r_{1,j}(t)\left(y_{a,j}(t) - x_{a,j}(t)\right) + c_2 r_{2,j}(t)\left(\hat{y}_j(t) - x_{a,j}(t)\right) \quad (4) \\ x_{a,j}(t+1) &= x_{a,j}(t) + v_{a,j}(t+1) \end{aligned}$$

where w is the inertia weight,[22] and c_1, c_2 are the acceleration constants which are usually set to 1.49 or 2. $r_{1,j} \sim U(0,1)$ and $r_{2,j} \sim U(0,1)$ are random variables with a uniform distribution. Recall from the earlier discussion that the first term in the summation is the *memory* term, which represents the contribution of previous velocity, the second term is the *cognitive* component, which represents the particle's own experience and the third term is the *social* component through which the particle is "guided" by the *gbest* particle towards the GB solution so far obtained. Although the use of inertia weight, *w,* was later added by Shi and Eberhart [22], into the velocity update equation, it is widely accepted as the basic form of PSO algorithm. A larger value of w favors exploration while a small inertia weight favors exploitation. As originally introduced, w is often linearly decreased from a high value (e.g. 0.9) to a low value (e.g. 0.4) during the iterations of a PSO run, which updates the positions of the particles using (4). Depending on the problem to be optimized, PSO iterations can be repeated until a specified number of iterations, say *IterNo,* is exceeded, velocity updates become zero, or the desired fitness score is achieved (i.e. $f < \varepsilon_C$ where f is the fitness function and ε_C is the cut-off error).

C. MD PSO and FGBF

Instead of operating at a fixed dimension N, the MD PSO algorithm is designed to seek both positional and dimensional optima within a dimension range, ($D_{\min} \le N \le D_{\max}$). In order to accomplish this, each particle has two sets of components, each of which has been subjected to two independent and consecutive processes. The first one is a regular positional PSO, i.e. the traditional velocity updates and following positional moves in N dimensional search (solution) space. The second one is a dimensional PSO, which allows the particle to navigate through dimensions. Accordingly, each particle keeps track of its last position, velocity and personal best position (*pbest*) in a particular dimension so that when it re-visits the same dimension at a later time, it can perform its regular "positional" fly using this information. The dimensional PSO process of each particle may then move the particle to another dimension where it will remember its positional status and keep "flying" within the positional PSO process in this dimension, and so on. The swarm, on the other hand, keeps track of the *gbest* particles in all dimensions, each of which respectively indicates the best (global) position so far achieved and can thus be used in the regular velocity update equation for that dimension. Similarly the dimensional PSO process of each particle uses its personal best dimension in which the personal best fitness score has so far been achieved. Finally, the swarm keeps track of the global best dimension, *dbest*, among all the personal best dimensions. The *gbest* particle in *dbest* dimension represents the optimum solution (and the optimum dimension).

In a MD PSO process and at time (iteration) t, each particle a in the swarm, $\xi = \{x_1,..,x_a,..,x_S\}$, is represented by the following characteristics:

$xx_{a,j}^{xd_a(t)}(t)$: j^{th} component (dimension) of the position of particle a, in dimension $xd_a(t)$

$vx_{a,j}^{xd_a(t)}(t)$: j^{th} component (dimension) of the velocity of particle a, in dimension $xd_a(t)$

$xy_{a,j}^{xd_a(t)}(t)$: j^{th} component (dimension) of the personal best (*pbest*) position of particle a, in dimension $xd_a(t)$

gbest(d) : Global best particle index in dimension d

$x\hat{y}_j^d(t)$: j^{th} component (dimension) of the global best position of swarm, in dimension d

$xd_a(t)$**:** Dimension component of particle a

$vd_a(t)$**:** Velocity component of dimension of particle a

$x\tilde{d}_a(t)$**:** Personal best dimension component of particle a

Figure 1 shows sample MD PSO and *bPSO* particles with index a. The *bPSO* particle that is at a (fixed) dimension, N=5, contains only positional components whereas MD PSO particle contains both positional and dimensional components respectively. In the figure the dimension range for the MD PSO is given between 2 and 9; therefore the particle contains 8 sets of positional components (one for each dimension). In this example, the current dimension where the particle a resides is 2 ($xd_a(t) = 2$) whereas its personal best dimension is 3 ($x\tilde{d}_a(t) = 3$). Therefore, at time *t,* a positional PSO update is first performed over the positional elements, $xx_a^2(t)$ and then

the particle may move to another dimension by the dimensional PSO.

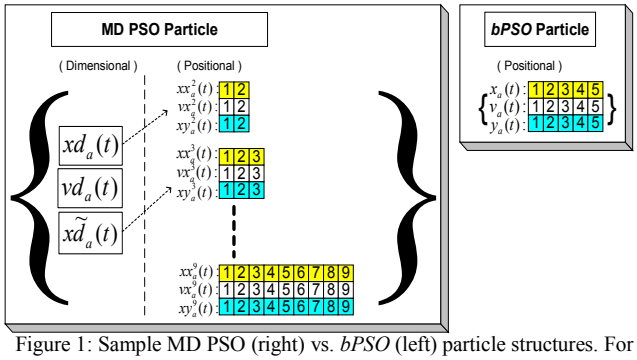

Figure 1: Sample MD PSO (right) vs. *bPSO* (left) particle structures. For MD PSO $[D_{\min}=2,\ D_{\max}=9]$ and at the current time *t*, $xd_a(t)=2$ and $\widetilde{xd}_a(t)=3$. For *bPSO N*=5.

Once the MD PSO process terminates, the optimum solution will be $\hat{xy}^{dbest}$ at the optimum dimension, *dbest*, achieved by the particle *gbest(dbest)* and finally the best (fitness) score achieved will naturally be $f(\hat{xy}^{dbest})$. Due to space limitations the pseudo-code of the MD PSO technique is skipped in this paper.

Fractional GB formation (FGBF) is designed to avoid premature convergence by providing a significant diversity obtained from a proper *fusion* of the swarm's best components (the individual dimension(s) of the current position of each particle in the swarm). At each iteration in a *bPSO* process, an artificial GB particle (*aGB*) is (fractionally) formed by selecting the most promising (or simply the best) particle (dimensional) components from the entire swarm. Therefore, especially during the initial steps, the FGBF can most of the time be a better alternative than the native *gbest* particle since it has the advantage of assessing each dimension of every particle in the swarm individually, and forming the *aGB* particle fractionally by using the most promising (or simply the best) components among them. This process naturally uses the available diversity among individual dimensional components and thus it can prevent the swarm from trapping in local optima. Suppose for a swarm ξ, FGBF is performed in a PSO process at a (fixed) dimension *N*. Recall from the earlier discussion that in a particular iteration, *t*, each PSO particle, *a*, has the following components: position ($x_{a,j}(t)$), velocity ($v_{a,j}(t)$) and the personal best position ($y_{a,j}(t)$), $j\in[1,N]$). *aGB* particle, first of all, does not use a velocity term, since instead of velocity updates, the *aGB* particle is fractionally (re-) created from the dimensions of some swarm particles. Consequently, $y_{aGB}(t)$ is set to the best of $x_{aGB}(t)$ and $y_{aGB}(t-1)$. As a result, the FGBF process creates one *aGB* particle providing a (potential) GB solution ($y_{aGB}(t)$). Let $f(a,j)$ be the dimensional fitness score of the j^{th} component of particle *a*.

FGBF has its generalized form when applied with MD PSO where there is one *gbest* particle per (potential) dimension of the solution space. For this purpose, recall from the earlier discussion that in a particular iteration, *t*, each MD PSO particle, *a*, has the following components: position ($xx_{a,j}^{xd_a(t)}(t)$), velocity ($vx_{a,j}^{xd_a(t)}(t)$) and the personal best position ($xy_{a,j}^{xd_a(t)}(t)$) for each potential dimensions in solution space (i.e. $xd_a(t)\in[D_{\min}, D_{\max}]$ and $j\in[1, xd_a(t)]$) and their respective counterparts in the dimensional PSO process (i.e. $xd_a(t)$, $vd_a(t)$ and $\widetilde{xd}_a(t)$). *aGB* particle does not need dimensional components where a single positional component with the maximum dimension $D_{\max}$ is created to cover all dimensions in the range, $\forall d\in[D_{\min}, D_{\max}]$, and as explained earlier, there is no need for the velocity term either, since *aGB* particle is fractionally (re-) created from the dimensions of some swarm particles. Furthermore, the aforementioned competitive selection ensures that $xy_{aGB}^{d}(t), \forall d\in[D_{\min}, D_{\max}]$ is set to the best of the $xx_{aGB}^{d}(t)$ and $xy_{aGB}^{d}(t-1)$. As a result, the FGBF process creates one *aGB* particle providing (potential) GB solutions ($xy_{aGB}^{d}(t)$) for all dimensions in the given range (i.e. $\forall d\in[D_{\min}, D_{\max}]$). Due to space limitations the pseudo-code of the FGBF over MD PSO is skipped.

D. Data Clustering

As the process of identifying natural groupings in a multidimensional data based on some distance metric (e.g. *Euclidean*), data clustering can be divided into two main categories: hierarchical and partitional [9]. Each category then has a wealth of sub-categories and different algorithmic approaches for finding the clusters. Clustering can also be performed in two different modes: hard (or crisp) and fuzzy. In the former mode, the clusters are disjoint, non-overlapping and any data point belongs to a single cluster whereas in the latter case it can belong to all the clusters with some degree of membership [1]. *K-means* [9] is a well known and widely used clustering method, which first assigns each data point to one of the *K* cluster *centroids* and then updates them to the *mean* of their associated points. As a hard clustering method, *K-means* suffers from the following drawbacks:

- The number of clusters *K*, needs to be set in advance.
- The performance of the method depends on the initial (random) centroid positions as the method converges to the closest local optima.
- The method is also dependent on the data distribution.

A hard clustering technique based on the *bPSO* was first introduced by Omran et al. in [18] and this work showed that the *bPSO* can outperform *K-means*, FCM, KHM and some other state-of-the-art clustering methods in any (evaluation) criteria. This is indeed an expected outcome due to the PSO's aforementioned ability to cope up with the local optima by maintaining a guided random search operation through the swarm particles. In clustering, similar to other PSO applications, each particle represents a potential solution at a

particular time t, i.e. the particle a in the swarm, $\xi = \{x_1,..,x_a,..,x_S\}$, is formed as $x_a(t) = \{c_{a,1},..,c_{a,j},..,c_{a,K}\} \Rightarrow x_{a,j}(t) = c_{a,j}$ where $c_{a,j}$ is the j^{th} (potential) cluster centroid in N dimensional data space and K is the number of clusters fixed in advance. Note that the data space dimension, N, is now different than the solution space dimension, K. Furthermore, the fitness (validity index) function, f that is to be optimized, is formed with respect to two widely used criteria in clustering:

- *Compactness*: Data items in one cluster should be similar or close to each other in N dimensional space and different or far away from the others when belonging to different clusters.
- *Separation*: Clusters and their respective centroids should be distinct and well-separated from each other.

The fitness functions for clustering are then formed as a regularization function fusing both *Compactness* and *Separation* criteria and in this problem domain they are known as clustering validity indices. A throughout survey about them can be found in [8]. Most of them presented promising results; however, none of them can guarantee the "optimum" number of clusters in every clustering scheme. Especially for the aforementioned PSO-based clustering in [18], the clustering scheme further depends on weight coefficients and may, therefore, result in over- or under-clustering particularly in complex data distributions.

III. The Proposed DC Extraction Technique

In this section, we shall first present the proposed dynamic clustering method using MD PSO and FGBF. The distance metric based on a fuzzy model over HSV color space will be introduced next.

A. Dynamic Clustering by MD PSO and FGBF

Based on the earlier discussion it is obvious that the clustering problem requires the determination of the solution space dimension (i.e. number of clusters, K) and an effective mechanism to avoid local optima traps (both dimensionally and spatially) particularly in complex clustering schemes in high dimensions (e.g. $K > 10$). The former requirement justifies the use of the proposed MD PSO technique while the latter calls for FGBF. At time t, the particle a in the swarm, $\xi = \{x_1,..,x_a,..,x_S\}$, has the positional component formed as,

$$xx_a^{xd_a(t)}(t) = \{c_{a,1},..,c_{a,j},..,c_{a,xd_a(t)}\} \Rightarrow xx_{a,j}^{xd_a(t)}(t) = c_{a,j}$$

meaning that it represents a potential solution (i.e. the cluster centroids) for the $xd_a(t)$ number of clusters whilst j^{th} component being the j^{th} cluster centroid. Apart from the regular limits such as (spatial) velocity, $V_{\max}$, dimensional velocity, $VD_{\max}$ and dimension range $D_{\min} \le xd_a(t) \le D_{\max}$, the N dimensional data space is also limited with some practical spatial range, i.e. $X_{\min} < xx_a^{xd_a(t)}(t) < X_{\max}$. In case this range is exceeded even for a single dimension j, $xx_{a,j}^{xd_a(t)}(t)$, then all positional components of the particle for the respective dimension $xd_a(t)$ are initialized randomly within the range and this further contributes to the overall diversity. The following validity index is used to obtain computational simplicity with minimal or no parameter dependency,

$$f(xx_a^{xd_a(t)}, Z) = Q_e(xx_a^{xd_a(t)})(xd_a(t))^{\alpha} \quad where$$

$$Q_e(xx_a^{xd_a(t)}) = \frac{1}{xd_a(t)} \sum_{j=1}^{xd_a(t)} \frac{\sum_{\forall z_p \in xx_{a,j}^{xd_a(t)}} \left\| xx_{a,j}^{xd_a(t)} - z_p \right\|}{\left\| xx_a^{xd_a(t)} \right\|} \quad (5)$$

where Q_e is the quantization error (or the average intra-cluster distance) as the *Compactness* term and $(xd_a(t))^{\alpha}$ is the *Separation* term, by simply penalizing higher cluster numbers with an exponential, $\alpha > 0$. Using $\alpha = 1$, the validity index yields the simplest form (i.e. only the nominator of Q_e) and becomes entirely parameter-free.

On the other hand, (hard) clustering has some constraints. Let $C_j = \{xx_{a,j}^{xd_a(t)}(t)\} = \{c_{a,j}\}$ be the set of data points assigned to a (potential) cluster centroid $xx_{a,j}^{xd_a(t)}(t)$ for a particle a at time t. The partitions C_j, $\forall j \in [1, xd_a(t)]$ should maintain the following constraints:

- Each data point should be assigned to one cluster set, i.e. $\bigcup_{j=1}^{xd_a(t)} C_j = Z$
- Each cluster should contain at least one data point, i.e. $C_j \neq \{\phi\}$, $\forall j \in [1, xd_a(t)]$
- Two clusters should have no common data points, i.e. $C_i \cap C_j = \{\phi\}$, $i \neq j$ *and* $\forall i, j \in [1, xd_a(t)]$

In order to satisfy the 1st and 3rd (hard) clustering constraints, before computing the clustering fitness score via the validity index function in (5), all data points are first assigned to the *closest* centroid. Yet there is no guarantee for the fulfillment of the 2nd constraint since $xx_a^{xd_a(t)}(t)$ is set (updated) by the internal dynamics of the MD PSO process and hence any dimensional component (i.e. a potential cluster candidate), $xx_{a,j}^{xd_a(t)}(t)$, can be in an abundant position (i.e. no closest data point exists). To avoid this, a high penalty is set for the fitness score of the particle, i.e. $f(xx_a^{xd_a(t)}, Z) \approx \infty$, *if* $\{xx_{a,j}^{xd_a(t)}\} = \{\phi\}$ for any j.

The major outlines so far given are sufficient for the standalone application of the MD PSO technique for a dynamic clustering application; however, the FGBF operation presents further difficulties since for the *aGB* creation the selection of the best or the most promising dimensions (i.e. the cluster centroids) among all dimensions of swarm particles is not straightforward. Furthermore, data points assigned to the j^{th} dimension of a particle *a,* ($\forall z_p \in xx_{a,j}^{xd_a(t)}(t)$), also depend on the distribution of the other dimensions (centroids), i.e. the "closest" data points are assigned to the j^{th} centroid only because the other centroids happen to be at a farther location. Inserting this particular

dimension (centroid) into another particle (say *aGB*, in case selected), might create an entirely different assignment (or cluster) including the possibility of having no data points assigned to it and thus violating the 2nd clustering constraint. To avoid this problem, a new approach is adopted. At each iteration, a subset among all dimensions of swarm particles is first formed by verifying the following: a dimension of any particle is selected into this subset if and only if there is at least one data point that is closest to it. Henceforth, the creation of the *aGB* particle within this verified subset ensures that the 2nd clustering constraint will (always) be satisfied. Figure 2 illustrates the formation of the subset on a sample data distribution with 4 clusters. Note that in the figure, all dimensions of the entire swarm particles are shown as '+' but the red ones belonging to the subset have at least one (or more) data points closest whereas the blue ones have none and hence they are discarded.

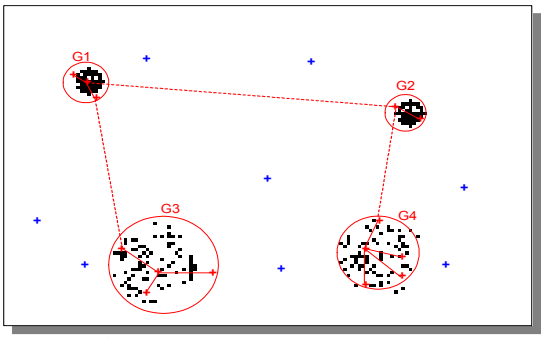

Figure 2: The formation of the centroid subset in a sample clustering example. The black dots represent data points over 2D space and each colored '+' represents one centroid (dimension) of a swarm particle.

Once the subset centroids are selected, then the most promising $D_{\max}$ centroids are selected from the subset in such a way that each dimensional component of the *aGB* particle with K dimensions can represent one of the true clusters, i.e., being in a close vicinity of its centroid. To accomplish this, only such $D_{\max}$ dimensions that fulfill the two clustering criteria, *Compactness* and *Separation* are selected and then stored in $a[j]$. To achieve well-separated clusters and to avoid the selection of more than one centroid representing the same cluster, *spatially* close centroids are first grouped using a Minimum Spanning Tree (MST) [15] and then a certain number of centroid groups, say $d \in [D_{\min}, D_{\max}]$, can be obtained simply by breaking (d-1) longest MST branches. From each group, one centroid, which provides the highest *Compactness* score (i.e. minimum dimensional fitness score, $f(a,j)$) is then selected and inserted into $a[j]$ as the j^{th} dimensional component. During the computation of the validity index $f(xx_a^{xd_a(t)}, Z)$ in (5), $f(a,j)$ can simply be set as the j^{th} term of the summation in Q_e expression, such as,

$$f(a,j) = \frac{\sum_{\forall z_p \in xx_{a,j}^{xd_a(t)}} \left\| xx_{a,j}^{xd_a(t)} - z_p \right\|}{\left\| xx_a^{xd_a(t)} \right\|} \qquad (6)$$

In Figure 2, a sample MST is formed using 14 subset centroids as the nodes and 13 branches are shown as the red lines connecting the closest nodes (in a minimum span). Breaking the 3 longest branches (shown as the dashed lines) thus reveals the 4 groups (G1,..,G4) among which one centroid yielding the minimum $f(a,j)$ can then be selected as an individual dimension of the *aGB* particle with 4 dimensional components (i.e. d=K=4, $xx_{aGB,j}^{K}(t), \forall j \in [1,K]$).

B. *Fuzzy Distance Model over HSV-HSL Color Domains*

Humans tend to think and describe the color the way they perceive it. Therefore, in order to achieve a color (dis-) similarity metric suitable to HVS, HSV (or HSL), which is a perceptual color space and provides means of modeling color in a way HVS does, is used in the proposed technique for extracting the dominant colors. Note that in a typical image with 24 bit RGB representation, there exists several thousands of color components, most of which cannot be perceived by HVS. Therefore, to reduce the computational complexity of RGB to HSV color transformation and to speed up the dynamic clustering process via MD PSO and FGBF, a pre-processing step, which creates a limited color palette in RGB color domain, is first performed. In this way such a massive, yet unperceivable amount of colors can be reduced to reasonable number, i.e. 256<N<512. For this we used the Median Cut method [14] because it is fast (O(N)) and an extensive set of experiments show that for such N it can result such an image that can hardly be distinguished from the original. Only the RGB color components in the color palette are then transformed into HSV (or HSL) color space over which the proposed dynamic clustering technique is applied to extract the dominant colors.

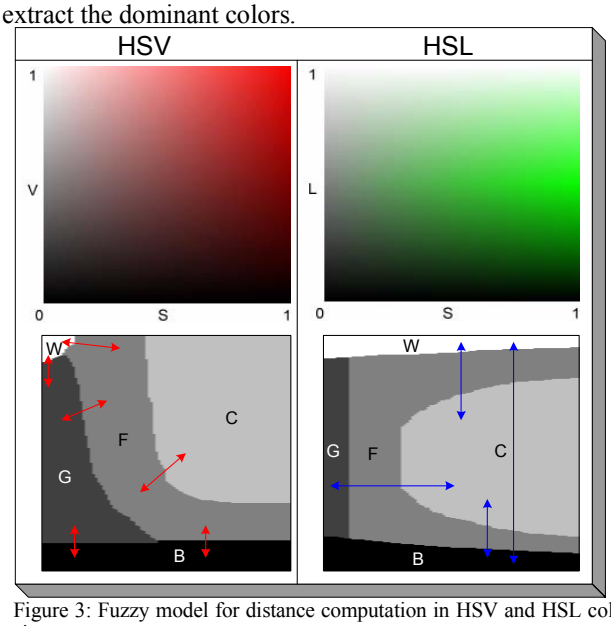

Figure 3: Fuzzy model for distance computation in HSV and HSL color domains.

Let $c_1 = \{h_1, s_1, v_1\}$ and $c_2 = \{h_2, s_2, v_2\}$ be two colors in HSV domain. Assume for the sake of simplicity that hue is in between 0 to 360 degrees and both s and v is unit normalized. The unit normalized *Euclidean* distance between the two can be defined as in Eq. (7).

$$\|c_1 - c_2\|^2 = (v_1 - v_2)^2 + (s_1 \cos(h_1) - s_2 \cos(h_2))^2 + (s_1 \sin(h_1) - s_2 \sin(h_2))^2 \quad (7)$$

Instead of using this typical distance metric for all color pairs, we adopt a perceptual approach in order to improve the discrimination among different colors. Recall from the earlier discussion that humans can recognize and distinguish 8 to 12 colors. For instance in [24], they have exploited the fact that humans tend to think and perceive colors only in 11 basic categories. Therefore, above a certain hue difference between two colors, it is obvious that they become entirely different for HVS, e.g. yellow and green are as different as yellow and blue or cyan or black or purple, so on. So if the hue difference is above certain threshold, a maximum difference should be introduced (i.e. 1.0) despite of the fact that Eq. (7) can only result in fractional distances. For a practical case, we have selected a conservative threshold by considering minimum number of colors as 8, therefore the perceptual threshold is, $\Delta H = 360/8 = 45$ degrees.

For those colors that have hue differences less than ΔH and therefore, show some degree of (hue) similarity, we apply a fuzzy color model for further discrimination. As shown in Figure 3, for a fixed hue, red for HSV and green for HSL, a typical saturation (S) vs. Value (V) or Lightness (L) plot can be partitioned into 5 regions. White (W), black (B), Gray (G), Color (C) and Fuzzy (F), which is a transition area among them. As shown in the HSV map on the left with red arrows, for those consecutive regions, which have a common boundary in between, the typical distance metric as in Eq. (7) can conveniently be used except between W - G and B - G since they have no color component and hence the distance can only be computed over V (or L) components. This is the reason why there is no need for drawing the precise boundaries of F (even if possible) because, say between two colors, one in C and one in F, or both in C or both in F, the same distance metric shall anyway be applied (in Eq. (7)). The boundaries are only important to distinguish areas such as C, W and B where there is no similarity in between. Therefore, as shown in the HSL map on the right with blue arrows, if two colors despite of the fact that they have similar hues (i.e. $\Delta H < 45$), happen to be in such regions, maximum distance (1.0) shall be applied rather than computing Eq. (7).

IV. Experimental Results

We have made comparative evaluations against MPEG-7 DCD over a sample database with 110 images, which are selected from *Corel* database in such a way that the prominent colors (DCs) can be selected by ground-truth.

In order to demonstrate the strict parameter dependency of MPEG-7 DCD, we have varied only 2 parameters, T_A and T_S whilst keeping the other two parameters fixed, i.e. $N_{DC}^{\max} = 25$, and $\varepsilon_D = 0.01$. Three experiments with $T_A = 1\%, T_S = 15$, $T_A = 1\%, T_S = 25$ and $T_A = 5\%, T_S = 25$ are performed and DC number (per image) plot can be seen in Figure 4. It is evident from the figure that number of DCs is strictly dependent to the parameters used and can vary significantly, e.g. between 2 to 25 even for a particular image.

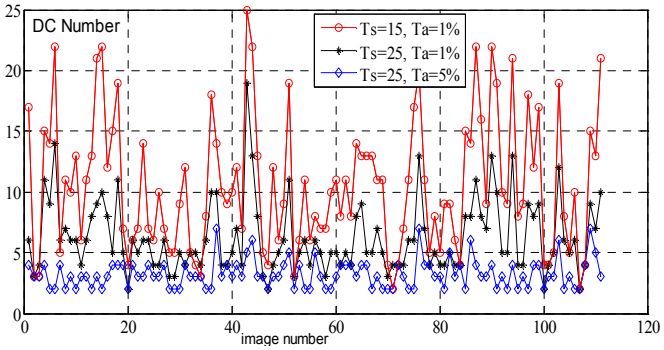

Figure 4: Number of DC plot from three different MPEG-7 DCDs over the sample database.

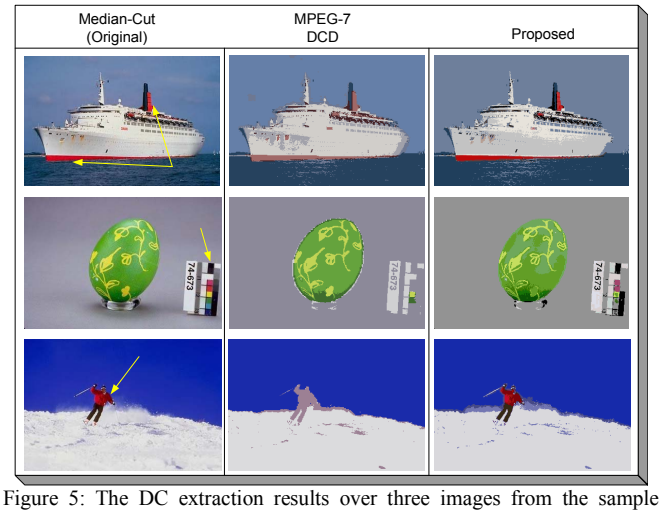

Figure 5: The DC extraction results over three images from the sample database.

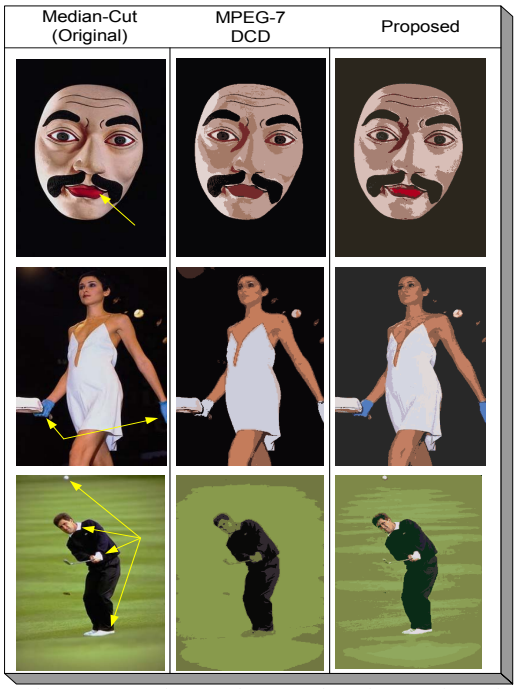

Figure 6: The DC extraction results over three images from the sample database.

The best MPEG-7 DCD is obtained using the first set of parameters, $T_A = 1\%$, $T_S = 15$, where highest number of DCs extracted. Figure 5 and Figure 6 shows some visual examples from the sample database. In both figures, the first column shows the output of the Median-Cut algorithm with 256 (maximum) colors, which is almost identical to the original image. The second and third columns show the back-projected images using the DCs extracted from MPEG-7 DCD with those best parameters and from the proposed technique, respectively. Note that the parts, where DC centroids cannot be accurately localized (the first row in both figures) or missed completely (second and third rows) by MPEG-7 DCD, are pointed with arrows.

V. CONCLUSIONS

In this paper, we first presented two efficient techniques, MD PSO and FGBF, as a cure to common drawbacks of the family of PSO methods such as *a priori* knowledge of the search space dimension and pre-mature convergence to local optima. A novel dynamic clustering technique based on MD PSO with FGBF is then proposed and applied for extracting *true* number of dominant colors in an image. In order to improve the discrimination among different colors, a fuzzy model over HSV (or HSL) color space is then proposed to achieve such a distance metric that reflects HVS perception of color difference.

The experiments over MPEG-7 DCD have shown that it is strictly parameter dependent. Moreover, since it is entirely based on K-means clustering method, it can create artificial colors and/or misses some important DCs due to its convergence to local optima, such as over- and under-clustering. This may also cause severe degradations over color textures since the regular textural pattern cannot be preserved if the true DC centroids are missed or shifted. With a simple clustering validity index, we have successfully addressed these problems and a superior DC extraction is achieved with ground-truth DCs. The optimum number of DCs can slightly vary on some images, say one or few DCs less or more, but the number of DCs on such images is hardly definitive, rather subjective and thus on such occasions the dynamic clustering based on a stochastic optimization technique can converge to some near-optimal solutions. Note that the optimization is only achieved with respect to the validity index function used and therefore, using a more efficient validity index such as Dunn's index [4] might further improve the performance.

REFERENCES

[1] G. P. Babu, B. M. Mehtre, and M. S. Kankanhalli, "Color Indexing for Efficient Image Retrieval", *Multimedia Tools and Applications*, vol. 1, pp. 327-348, Nov. 1995.

[2] T. Back and F. Kursawe, "Evolutionary algorithms for fuzzy logic: a brief overview", In *Fuzzy Logic and Soft Computing*, World Scientific, pp. 3–10, Singapore, 1995.

[3] Y. Deng, C. Kenney, M. S. Moore, and B. S. Manjunath, "Peer Group Filtering and Perceptual Color Image Quantization", In *Proc. of IEEE Int. Symposium on Circuits and Systems, ISCAS*, vol. 4, pp. 21-24, 1999.

[4] J. C. Dunn, "Well separated clusters and optimal fuzzy partitions", *Journal of Cybernetics*, vol. 4, pp. 95–104, 1974.

[5] J. Fauqueur and N. Boujemaa, "Region-Based Image Retrieval: Fast Coarse Segmentation and Fine Color Description", in *Proc. of IEEE Int. Conf. on Image Processing (ICIP'2002)*, Rochester, USA, Sep. 2002.

[6] U.M. Fayyad, G.P. Shapire, P. Smyth and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining,* MIT Press, Cambridge, MA, 1996.

[7] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, pp. 1-25. MA, 1989.

[8] M. Halkidi, Y. Batistakis, M. Vazirgiannis, "On Cluster Validation Techniques", *Journal of Intelligent Information Systems*, vol. 17 no. 2, 3. pp. 107-145, 2001.

[9] A. K. Jain, M.N. Murthy and P.J. Flynn, "Data Clustering: A Review", *ACM Computing Reviews*, Nov 1999.

[10] J. Kennedy, R Eberhart., "Particle swarm optimization", in Proc. *of IEEE Int. Conf. On Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, 1995.

[11] S. Kiranyaz, J. Pulkkinen and M. Gabbouj, "Multi-dimensional Particle Swarm Optimization for Dynamic Environments", in *Proc. of 5th Int. Conf. On Innovations in Information Technology* (Innovations'08), Al Ain, UAE, Dec. 2008.

[12] S. Kiranyaz, T. Ince, A. Yildirim and M. Gabbouj, "Unsupervised Design of Artificial Neural Networks via Multi-Dimensional Particle Swarm Optimization", *in Proc. of Int. Conf. on Pattern Recognition, (ICPR 2008)*, Tampa, Florida, US, Dec. 2008.

[13] J. Koza, *Genetic Programming: On the Programming of Computers by means of Natural Selection,* MIT Press, Cambridge, Massachussetts, 1992.

[14] A. Kruger, "Median-cut color quantization," Dr. Dobb's Journal, pp. 46-54 and 91-92, Sept. 1994.

[15] J. R. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem", *Proc. of AMS,* 71*,* 1956.

[16] A. Mojsilovic, J. Kovacevic, J. Hu, R. J. Safranek, K. Ganapathy, "Matching and Retrieval based on the Vocabulary and Grammar of Color Patterns", *in IEEE Trans. on Image Processing*, vol. 9, no. 1, pp. 38-54, Jan. 2000.

[17] B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan, and A. Yamada, "Color and Texture Descriptors", *IEEE Trans. On Circuits and Systems for Video Technology*, vol. 11, pp. 703-715, Jun. 2001.

[18] M. Omran, A. Salman and A. P. Engelbrecht, "Image Classification using Particle Swarm Optimization", In *Conf. on Simulated Evolution and Learning*, vol. 1, pp. 370–374, 2002.

[19] M. G. Omran, A. Salman, and A.P. Engelbrecht, "Dynamic Clustering using Particle Swarm Optimization with Application in Image Segmentation", In *Pattern Analysis and Applications*, vol. 8, pp. 332-344, 2006.

[20] M. G. Omran, A. Salman, and A.P. Engelbrecht, *Particle Swarm Optimization for Pattern Recognition and Image Processing,* Springer Berlin, 2006.

[21] J. Riget and J. S. Vesterstrom, "A Diversity-Guided Particle Swarm Optimizer - The ARPSO", Technical report, Department of Computer Science, University of Aarhus, 2002.

[22] Y. Shi and R.C. Eberhart, "A Modified Particle Swarm Optimizer", In *Proc. of the IEEE Congress on Evolutionary Computation*, pp. 69-73, 1998.

[23] F. Van den Bergh, "An Analysis of Particle Swarm Optimizers", PhD thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.

[24] E. L. Van den Broek, P. M. F. Kisters, and L. G. Vuurpijl, "The utilization of human color categorization for content-based image retrieval", *in Proc. of Human Vision and Electronic Imaging IX,* pp. 351-362, San José, CA (SPIE, 5292), 2004.