International Journal of Computational Intelligence Systems, Vol.2, No. 1 (March, 2009), 27-38

Feature Weighting and Retrieval Methods for Dynamic Texture Motion Features

Ashfaqur Rahman[*]
Department of Computer Science, American International University Bangladesh
Dhaka 1213, Bangladesh

Manzur Murshed[†]
Gippsland School of Information Technology, Monash University
Churhill, Vic 3842, Australia

Received: 18-06-2008
Revised: 11-02-2009

Abstract

Feature weighing methods are commonly used to find the relative significance among a set of features that are effectively used by the retrieval methods to search image sequences efficiently from large databases. As evidenced in the current literature, dynamic textures (image sequences with regular motion patterns) can be effectively modelled by a set of spatial and temporal motion distribution features like motion co-occurrence matrix. The aim of this paper is to develop effective feature weighting and retrieval methods for a set of dynamic textures while characterized by motion co-occurrence matrices.

Keywords: dynamic texture, feature weighting, video retrieval.

1. Introduction

Dynamic textures are image sequences on natural scenarios possessing regular motion patterns. The motion assembly by a flock of flying birds, water streams, fluttering leaves, and waving flags are some of the most common examples of dynamic textures. Multimedia objects e.g., image sequences, are presented in terms of their features for the purpose of indexing and retrieval. A good feature has the property that it is similar among the same class of objects and dissimilar among different classes of objects. Feature weighing methods are commonly used to find the relative significance among a set of features.

Retrieval methods, on the other hand are used to search image sequences efficiently from large databases by using appropriate feature set. The growing popularity of the Internet, the introduction of new consumer products for digital image and video creation, and the emergence of digital standards for television broadcasting have resulted in a greater demand for efficient retrieval of multimedia data. Feature weighting and retrieval methods while applied on first order features deserve some special consideration.

Motion Co-occurrence Matrix (MCM), a commonly used fist order feature for analysing dynamic textures[3,7,11], is computed from some motion related measures along a clique from time-space domain (Section 2.1). The conventional feature weighting approach considers the features altogether and computes feature weights using a heuristic search as otherwise an exhaustive search is highly time consuming under such a scenario for large feature sets. Considering domain

[*]Dr. Ashfaqur Rahman, Assistant Professor, Department of Computer Science, American International University Bangladesh (AIUB). *Mail*: Campus 4, 55/B, Road 21, Kemal Ataturk Avenue,Banani, Dhaka 1213, Bangladesh. *Ph*: +8801713408082 (M) *Email*: ashfaqur@aiub.edu
[†] Dr. Manzur Murshed, Associate Professor, Gippsland School of Information Technology, Monash University, Churchill, Vic 3842, Australia. *Ph*: +61351226467 (O) *Email*: manzur.murshed@infotech.monash.edu.au.

information, however, MCMs can be organized into a hierarchy and weights can be computed at each level in the hierarchy. Such an approach facilitates using an exhaustive search at levels with a smaller number of features, and thus improving the overall classification accuracy. We thus propose in this paper a hierarchical approach for feature weighting that provides a mechanism to control motion measure specific MCM, clique and domain weights at different levels. Note that the conventional feature weighting approach is unable to control MCM weights domain wise and leaves room for some classification accuracy improvement as observed in the result section (Table 1).

Fast retrieval of image sequences from large databases is commonly performed using hierarchical retrieval methods. To perform hierarchical retrieval it is necessary to group the image sequences beforehand, based on some similarity measures. This phase is known as clustering. The goal is to group similar videos into a hierarchy of clusters and compute the cluster centres so that during retrieval, the query image sequence can be compared with the cluster centres instead of all the image sequences in the database. The cluster centre is commonly computed using the average of feature vectors of a set of videos in the cluster. Such averaging of MCMs however represents a mixed dynamic texture and commonly used distance measures between single and mixed textures suffer from the volumetric problem[1] where the MCM of the mix can represent none of the component textures. So, MCM averaging is not suitable to represent a cluster for retrieving dynamic textures based on their MCM. We thus propose a new cluster centre computation procedure and develop a Multi Feature Centre Retrieval (MFCR) method. As evidenced from experimental results, our proposed MFCR method provides better retrieval performances over the conventional hierarchical retrieval methods.

This paper is organized as follows. Some background and related works are briefed in Section 3. Some contemporary and our proposed feature weighting approaches are elaborated in Section 4. Our proposed retrieval method is elaborated in Section 5 along with some other retrieval methods applicable on MCM. Comparative study on experimental results of feature weighting and retrieval methods is presented in Section 6. Finally Section 7 concludes the paper.

2. Background and Related Works

MCM is the most commonly used first order motion distribution statistics in all the dynamic texture characterization techniques and for the sake of clarity of future discussion we first define MCM in this Section. There exist a number of motion based dynamic texture characterization techniques[1–11]. MCM as a first order feature is used in an explicit manner in Optimal Time Space Ratio technique[3,11]. The functionality of OTSR is thus also elaborated in this Section.

2.1. *Motion Co-occurrence Matrix*

Let a sequence of motion frames be represented by a function $M_t(x,y)$ such that (x,y) points to the spatial location at t-th motion frame. A MCM Γ is a tabulation of how often different pair of motion measures (u,v) occur over the frame sequence such that u is associated with a pixel $M_t(x,y)$ and its neighbour $M_{t+\eta_t}(x+\eta_x, y+\eta_y)$ is associated with v, where $\eta_x, \eta_y \in \{-1,0,1\}$ and $\eta_t \in \{-1,0\}$. The neighbours (Fig 1) are called temporal neighbours for $\eta_t = -1$ and spatial neighbours for $\eta_t = 0$. Let the term "clique" represent a unique neighbour direction identified by a triple (η_x, η_y, η_t). The co-occurrence matrix corresponding to clique (η_x, η_y, η_t) is denoted by $\Gamma_{(\eta_x,\eta_y,\eta_t)}$.

2.2. *OTSR Technique*

A dynamic texture has one temporal dimension and two spatial dimensions and in OTSR three instances of MCM namely temporal co-occurrence matrix $\Gamma_{(0,0,-1)}$ and spatial co-occurrence matrices $\Gamma_{(1,0,0)}$ and $\Gamma_{(0,1,0)}$ is computed. Use of first order motion features like MCM in OTSR facilitates the use of block motion leading to real time dynamic texture recognition. Distance between dynamic textures is computed along each of these three cliques using KL- divergence and are fused into a single distance measure using a weighted Euclidian summation at an optimal weight. Assignment of explicit weights to time and space domains facilitates the use of any feature weighting technique to obtain the best classification results at optimal weight and hence the technique is named Optimal Time-Space Ratio (OTSR) technique. As first order features are directly used in OTSR, it provides a good platform for the application of our proposed feature weighting and retrieval methods and the reported experimental results in this paper are applied on an extended OTSR model as detailed next.

3. Feature Set of Extended OTSR Technique

Only a minimal set of three MCMs are considered in OTSR. In order to obtain more generalized feature weighting and retrieval methods, we need to extend the basic feature set of OTSR technique by incorporating

more MCMs. Consequently, some modifications to the parameter notations of the basic OTSR technique are made below.

Assuming that each image processing element in the motion frame sequence M is associated with a motion vector $\vec{\mathbf{v}} = (v_x, v_y)$, let $\zeta(\vec{\mathbf{v}})$ be a vector-to-scalar mapping function. In the extended OTSR technique, the following two vector-to-scalar mapping functions are considered

$$\zeta_m(\vec{\mathbf{v}}) = \sqrt{v_x^2 + v_y^2} \text{ and } \zeta_d(\vec{\mathbf{v}}) = \arctan(v_y / v_x) \quad (1)$$

whereas in the basic OTSR technique we used only $\zeta_m(\vec{\mathbf{v}})$. Both magnitude and direction of motion vectors are quantized into motion measures using a linear quantization method. In order to distinguish MCMs corresponding to magnitude and direction, we denote the MCM corresponding to clique η by Γ_η^ζ where $\zeta \in \{\zeta_m, \zeta_d\}$.

We also consider higher number of cliques in the extended OTSR technique. MCMs are computed for temporal cliques $\chi_t = \{(0,0,-1), (-1,1,-1), (0,1,-1), (1,1,-1), (1,0,-1)\}$ and spatial cliques $\chi_s = \{(-1,1,0), (0,1,0), (1,1,0), (1,0,0)\}$ (Fig 1). Thus in the extended OTSR technique, each dynamic texture is identified by a total of $5 \times 2 = 10$ temporal and $4 \times 2 = 8$ spatial MCMs. Note that the remaining cliques are ignored as their respective MCMs are identical to the MCMs of the cliques considered.

4. Feature Weighting Methods

The two most common groups of feature weighting methods are the filter and wrapper methods. Filter methods[16,27,29] determine the relevance of features to describe the data using statistical techniques and do not use feedback from subsequent learning or classification algorithms. Conversely, wrapper methods[14,17,26] use feedback that incorporate a Maximum Likelihood (ML) approach where weight assignment is optimized using classification accuracy as a performance measure. It is in this context that in this paper we focus on wrapper methods for feature weighting.

Finding optimal weights of a large set of features using an exhaustive search is a highly time consuming process. Heuristic search algorithms are common in the literature for feature weighting under such circumstances, and Genetic Algorithm (GA) is the most commonly used method because it performs a randomized search for optimal feature weights and it is not susceptible to becoming stuck in local minima. We thus explore the GA for feature weighting in this paper.

The conventional Flat Feature Weighting (FFW) approach considers the entire feature set for computing weights without factoring in domain information and thus is able to use only heuristic search methods like GA. We thus develop a Hierarchical Feature Weighting (HFW) approach, which provides a mechanism to control MCM, clique, and domain weights at different levels. We have used both exhaustive search and GA based randomized search for feature weighting at different levels in the hierarchy. We first elaborate GA feature weighting method in this section followed by the proposed HFW approach.

4.1. *Feature Weighting Using Genetic Algorithm*

GA[14] performs the optimal weight search using two operators – crossover and mutation. The crossover operator has the effect of merging solutions whilst preserving the already successful feature weights. The mutation operator makes sure that GA does not get stuck to local minima. GA encodes feature weights in a string of binary digits. Each binary digit is known as a *gene* and the string of genes is known as a *chromosome*. A group of b consecutive genes in a chromosome is decoded to calculate the feature weight. For a total of n features the chromosome length is nb.

In order to find the optimal feature weights, GA starts with a large set (*population*) of randomly generated chromosomes. The fitness of each chromosome in the current population is evaluated from the classification accuracy of a dataset with feature weights decoded from corresponding chromosome. A global variable stores the feature weights corresponding to the chromosome with maximum fitness. A set of chromosomes, whose fitness is better than the remaining chromosomes in the current population, are used to fill in a mating pool for subsequent crossover and mutation operations. The crossover operator randomly selects two parent chromosomes from the mating pool and interchanges a portion of the gene stream between them to generate new child chromosomes (Fig 2). A mutation operator is then applied on the new chromosomes where every gene (bit) in a chromosome is altered with a certain probability. These newly generated chromosomes constitute the GA population for the next generation. During the evolution of GA, the classification accuracy of the underlying data set is used as a performance measure of the weight sets. GA stops when there is no improvement of average fitness of the population for a consecutive number of generations.

4.2. *Hierarchical Feature Weighting (HFW)*

Weight of a feature computed by any feature weighting approach is applied to the corresponding feature

distance between two textures. We thus elaborate the distance measure first followed by the hierarchical feature weighting approach.

The primary dissimilarity between two textures i and j is computed in terms of motion magnitude MCM and motion direction MCM along each clique η using KL divergence (a commonly used distance metric for dynamic textures[3,7,12]). Let the distance along clique η for magnitude MCM and direction MCM be denoted by $D_\eta^{\zeta m}(i,j)$ and $D_\eta^{\zeta d}(i,j)$ respectively. These distances are fused into a single distance measure for each clique η as

$$DistClq_\eta(i,j) = \alpha_\eta^m D_\eta^{\zeta m}(i,j) + \alpha_\eta^d D_\eta^{\zeta d}(i,j), \quad (2)$$

where $0 \le \alpha_\eta^m, \alpha_\eta^d \le 1$ are the weights of the magnitude and direction measures respectively for clique η and $\alpha_\eta^m + \alpha_\eta^d = 1$.

The optimized clique distances are then fused into two domain distance measures, one for each of the two domains as

$$DistDom_t(i,j) = \sum_{\forall \eta \in \chi_t} \lambda_\eta DistClq_\eta^{OPT}(i,j) \quad (3)$$

and

$$DistDom_s(i,j) = \sum_{\forall \eta \in \chi_s} \lambda_\eta DistClq_\eta^{OPT}(i,j) \quad (4)$$

where $0 \le \lambda_\eta \le 1$ is the weight of clique η such that $\sum_{\forall \eta \in \chi_t} \lambda_\eta = 1$ and $\sum_{\forall \eta \in \chi_s} \lambda_\eta = 1$.

Finally, the optimized domain distances are fused into a single distance measure as

$$\nabla(i,j) = \omega_t DistDom_t^{OPT}(i,j) + \omega_s DistDom_s^{OPT}(i,j) \quad (5)$$

where $0 \le \omega_t, \omega_s, \le 1$ are the weights of temporal and spatial domains respectively and $\omega_s + \omega_t = 1$. It is clearly evident from the above hierarchical distance measure that explicit clique and domain weights are maintained at different levels to provide a mechanism to control corresponding weights.

In order to compute feature weights, three levels of weighting are involved, using hierarchical distance measure. We use the wrapper method to optimize the distance measure at each level, i.e., the weight ratio is searched for which the dataset is classified with the highest accuracy. At the first stage, α_η^m and α_η^d are optimized for each clique η where distance between two textures is computed using (2). We use an exhaustive search up to two decimal points at this stage to find the optimal weights between magnitude and direction MCMs corresponding to each clique. Although two weights are involved, due to the $\alpha_\eta^m + \alpha_\eta^d = 1$ constraint, the degree of freedom of this optimization process is reduced to $2-1=1$, hence justifying the exhaustive search. At the second stage, λ_η's are optimized for each domain using the distance measures in (3) and (4). As the degree of freedom of the optimization process at temporal and spatial domains are $|\chi_t|-1=4$ and $|\chi_s|-1=3$ respectively, an exhaustive search is no longer feasible, hence we opt to use the randomized heuristic GA search technique. Finally domain weights ω_t and ω_s are optimized where the distance between two textures is computed using (5). At this level again we use an exhaustive search up to two decimal points, as the degree of freedom is 1.

The HFW approach has the following benefits over the FFW approach: (i) As the search space is divided into smaller groups in HFW, we are able to use the exhaustive weight search at top and bottom level that FFW fails to use because of huge search space. Exhaustive search is better than randomized search and as a result HFW achieves better classification accuracy than FFW approach; (ii) for the same reason HFW is faster than FFW. If weighting needs to be adjusted adaptively, HFW is clearly a better alternative to FFW; and (iii) HFW provides a mechanism to control weights among cliques as well as domains that FFW fails to provide. HFW clearly portrays relative significance among a set of cliques within a domain to make the selection process of a minimal set of features easy.

5. Retrieval Methods

Time efficient retrieval of visual documents from large databases has become an important research area in recent times because of the proliferation of video and image data in digital form. The most commonly used video retrieval scheme in contemporary visual document retrieval systems[18,19,22,24,28] is Query By Example[13] (QBE), where given a query video, similar examples are retrieved by comparing the low level features of the query video with the corresponding pre-computed features of the videos in the database. Such exhaustive search for similar videos in the database is highly time consuming and thus restricts the use of QBE for many practical applications. One approach to solve this problem is to create an indexing scheme by grouping similar videos beforehand into similar clusters

so that at the time of the query, only the relevant set of clusters are examined. These time efficient retrieval methods are commonly known as *hierarchical* retrieval methods. In this section we elaborate two existing such retrieval methods, HREC (Hierarchical Retrieval using Exhaustive Centre[7]) and HRRC (Hierarchical Retrieval using Representative Centre[15]).

As mentioned in Section 1, to perform hierarchical retrieval similar videos are grouped into a hierarchy of clusters beforehand and the cluster centres are computed so that during retrieval, the query video is guided to the proper cluster by comparing its features with those of the cluster centres. The basic hierarchical clustering algorithm used in all retrieval methods is presented in Fig 3. Hierarchical clustering starts by placing all the videos in the database into distinct clusters. These clusters are iteratively merged into new clusters based on some similarity measure $DClust$ and placed higher in the hierarchy (Fig 4). The merging process ends when only the root cluster is left.

During the retrieval tree building process, the *cluster centre* is computed for each newly formed cluster. A cluster centre is a hypothetical video that in general is represented by a feature vector computed over the feature vectors of the videos in that cluster. During retrieval, similarity between the feature vector of the query video and that of a cluster centre is computed using some similarity measure $DQuery$. The search process starts at the root cluster by comparing the query video with its two children. Based on similarity, the appropriate cluster is chosen from these two for searching in the next level. The process continues until the search reaches a leaf video node.

HREC, HRRC and our proposed MFCR method differs in defining $DClust$, cluster centre and $DQuery$, as detailed next.

5.1. *The HREC Method*

During the retrieval tree building process, the distance $DClust$ between two clusters C_k and C_l, is defined in terms of the distances of the videos as

$$DClust(C_k, C_l) = \max_{(i,j)\in E_k \times E_l} \nabla(i,j) \qquad (6)$$

where $\nabla(i,j)$ is the distance between videos i and j. The cluster centre for cluster C_k is represented by a hypothetical video whose feature vector is obtained by averaging the feature vectors of all the videos in E_k. During retrieval, the distance between a query video i and a cluster C_k is computed as

$$DQuery(i, C_k) = \nabla(i, c_k) \qquad (7)$$

where c_k is the hypothetical video representing cluster C_k.

5.2. *The HRRC Method*

During the retrieval tree building process, the distance $DClust$ between two clusters C_k and C_l, is defined in terms of the distances of the videos as

$$DClust(C_k, C_l) = \underset{(i,j)\in E_k \times E_l}{avg} \nabla(i,j) \qquad (8)$$

where $\nabla(i,j)$ is the distance between videos i and j. The cluster centre for cluster C_k is represented by a hypothetical video whose feature vector is obtained by averaging the feature vectors of a set of the R_k representative videos in E_k that are selected as

$$R_k = \left\{ \forall_{l\in E_k} l \;\middle|\; \underset{\forall_{j\neq l\in E_k}}{avg} \nabla(l,j) \le R\text{-}th \min_{\forall_{i\in E_k}} \underset{\forall_{j\neq i\in E_k}}{avg} \nabla(i,j) \right\} \qquad (9)$$

.

During retrieval, the distance between a query video i and a cluster C_k is computed using (7).

5.3. *The MFCR Method*

The feature vector computation process for cluster centre in existing HREC and HRRC methods is not suitable when each feature in the vector is a MCM. Average of a set of MCMs actually represents a mixed texture[1], and distance computation between single and mixed textures in terms of MCM suffers *volumetric* problem. Thus during retrieval the distance between the feature vector of a query video and that of the cluster centre is faulty when MCM is used as feature. We thus propose a new approach in the MFCR method to compute the cluster centre.

In the proposed MFCR method the distance $DClust$ between two clusters C_k and C_l is computed using (8). The cluster centre c_k for cluster C_k is associated with a set of R_k representative videos instead of a hypothetical mixed video. The R_k representative videos for cluster C_k are selected using (9). During retrieval, the distance between a query video i and a cluster C_k is computed as

$$DQuery(i, C_k) = \min_{\forall_{j\in R_k}} \nabla(i,j). \qquad (10)$$

As distance is computed between feature vectors of single videos in (10), the *volumetric* problem is resolved

in the proposed MFCR method. A cluster centre c_k is associated with a set of R_k feature vectors and hence we use the name Multi Feature Centre Retrieval method. For the sake of simplicity we have used the same number of representatives for all the clusters. Thus considering R representative videos, a cluster C_k containing N_k videos has $\min(R, N_k)$ videos associated with the cluster centre c_k.

With an increased number of representative videos, the probability of a query video belonging to the class of any representative video will be high, leading to higher retrieval accuracy, although at the cost of higher search time (as more distances need to be computed). Thus our proposed MFCR provides a user-controlled threshold (number of cluster representatives R) to control between retrieval accuracy and search time. In terms of storage overhead, MFCR is better than HREC and HRRC. In MFCR, when clusters are created we need to store only references to the representative videos in the database for cluster centre as we use their features directly whereas in both HREC and HRRC methods additional storage is required for the feature vector for the cluster centre that are generated by averaging multiple feature vectors.

6. Experimental Results

We conducted a set of feature weighting and retrieval experiments using the features of Extended OTSR technique. We used the dynamic texture dataset consisting of 124 video clips of Szummer image sequences[20,25]. Motion vectors were computed using block motion computation method[21] with block size 16×16 and search window of length $d=\pm7$ pixels. Magnitude of motion vectors were quantized into 21 motion measures such that motion measure i covered the range $\max(0,i-0.5)\le 4\times \textit{vector length} < \min(i+0.5,4\times7\sqrt{2})$ where $7\sqrt{2}$ is the maximum possible vector length with ±7 maximum displacement and $i=0,1,\ldots,40$. Motion directions were quantized into 26 motion measures such that motion measure i covered the range $\max(0,i-0.5)\le 4\times \textit{vector angle} < \min(i+0.5,4\times2\pi)$ where 2π is the maximum possible vector angle and $i=0,1,\ldots,25$.

For feature weighting experiments using both the FFW and HFW approach, we used the half of the dataset for learning feature weights and the full dataset for evaluation. In order to conduct the classification we used 1–NN classification method. For the FFW approach we used GA as the underlying feature weighting method. For the GA weighting method we considered 10,000 chromosomes in the GA population. The crossover operation is performed by dividing a chromosome into two equal halves and interchanging gene streams between two right corresponding halves of two different chromosomes. For mutation, a gene is altered if the corresponding random number generated using uniform distribution is less than 0.1. We used the same GA setup for the second level weighting in the HFW approach.

For retrieval experiments, the retrieval tree was built up using all the videos in the dataset. For retrieval purpose all the videos in the database were supplied as a query. We used a 1–NN classifier to evaluate the results of the retrieval methods. The feature vector of each video consists of 18 MCMs. To calculate the distance between videos we used hierarchical distance measure ∇ defined in (5) where the MCMs were weighted using the weights computed by the HFW approach.

6.1. *Feature Weighting Experiments*

Feature weights obtained by FFW and HFW are presented in Fig 5 and Fig 6 respectively. The performances of the feature weighting approaches are presented in Table 1. It can be observed from Table 1 that with feature weighting, classification accuracy improves by 9.1% and 10% for FFW and HFW approaches compared to that when no weighting in used. It is also observed that the HFW approach achieves better classification accuracy than the FFW approach. This can be attributed to the fact that two exhaustive search steps are used in HFW whereas FFW uses a randomized search. Note that the accuracy improvement is only 0.83%. This is because an exhaustive search is used to find weights between only two features at bottom level and two domains at top level compared to a significant number of cliques that are weighted at Step 2 in HFW that makes the influence more biased towards GA. However the influence of GA in FFW is more than that in the HFW approach.

It is evidenced from Fig 5 and Fig 6 that the HFW approach provides a clear picture of MCM, clique and domain weights at different levels that FFW fails to provide. It can be observed from bottom level (Step 1) weights of HFW that on an average magnitude MCMs are getting more weights than direction MCMs and thus direction MCMs can be ignored for practical applications. In OTSR, clique triplet $\{(1,0,0),(0,1,0),(0,0,-1)\}$ was selected over $\{(-1,1,0),(1,1,0),(0,0,-1)\}$. It can also be observed from the middle level (Step 2) of HWF that the former clique triplet is better than the later. Finally the time–space weight ratio obtained at the top level (Step 3) of HFW portrays the importance of spatial domain over temporal

that also agrees with the fact that an image sequence has two spatial and one temporal dimension.

6.2. *Retrieval Experiments*

In Fig 7 we report the retrieval accuracy results obtained using our proposed MFCR method for different number of cluster representatives using 1–NN classifiers. An important observation can be summarized from Fig 7. With an increased number of cluster representatives, classification accuracy gets better in general. This is because, with an increased number of representative videos, the probability of a query video belonging to the class of any representative video will be high, leading to a higher retrieval accuracy.

Although the accuracy of MFCR increases with an increase in the number of representatives per cluster, this is not without cost. Fig 8 shows the average retrieval time computed over different dynamic texture classes. It can be observed that retrieval time increases with the increased number of representatives per cluster. This is because with a higher number of representatives more distances need to be computed, thus increasing retrieval time. Thus the number of cluster representatives can act as a good threshold to control between accuracy and retrieval time. An increased number of cluster representatives provide better classification accuracy at the cost of retrieval time.

In Table 2 a comparison of the proposed MFCR method with the HREC and the HRRC methods is provided. For MFCR we have used $\lceil \log_2 124 \rceil = 7$ representatives per centre where 124 is the total number of videos in the database. It can be observed that the MFCR method outperforms the other two methods. The reason for this better accuracy level can be explained form from the fact that HREC and HRRC perform feature vector averaging, which is not suitable for features like MCM because of the *volumetric* problem. Thus query videos are misled to wrong clusters, resulting in poor retrieval accuracy. Our proposed MFCR method eliminates the volumetric problem and results in better retrieval accuracy.

7. Conclusions

In this paper we have developed effective feature weighting and retrieval methods to apply on MCMs. We have proposed a hierarchical approach for weighting a set of motion co-occurrence features to exploit the inherent domain influence by dividing the search space in such a way that the degree of freedom at the top and bottom level is low enough to use an exhaustive search. Consequently, better classification results have been achieved using this approach compared to the conventional flat feature weighting approach. We have modified the conventional hierarchical retrieval process so that the motion co-occurrence features can be used. More precisely, we have replaced the feature computation process of cluster centres at each level by associating a set of feature vectors instead of a single average feature vector, as such average features neither represent the cluster nor are suitable for computing similarity between motion distribution features. As evidenced from the experimental results, our proposed method provides better retrieval results over the conventional hierarchical retrieval methods.

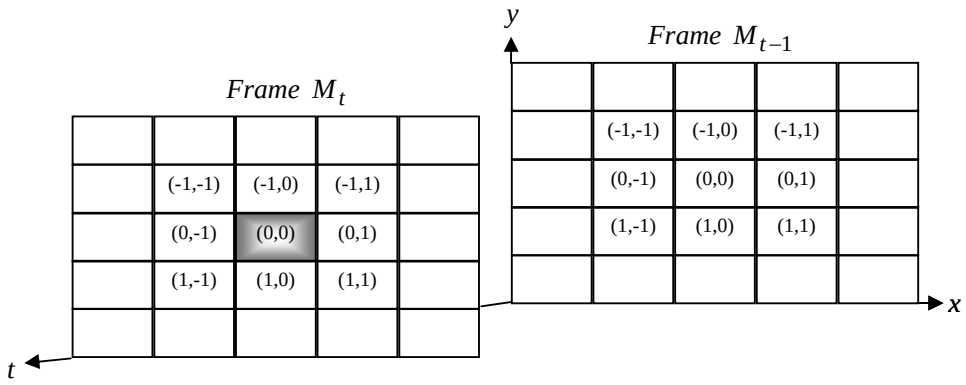

Fig 1: Spatiotemporal neighbourhood of point $M_t(0,0)$. The temporal and spatial neighbouring points are in frames M_{t-1} and M_t.

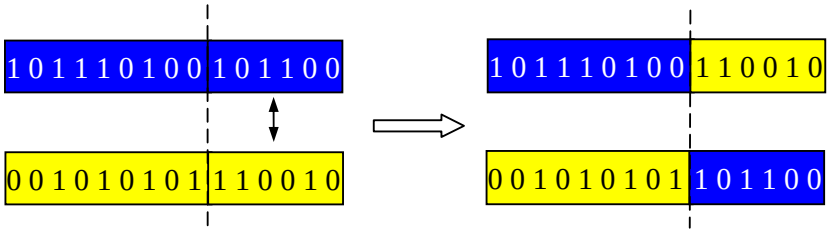

Fig 2: Crossover of binary chromosomes.

Algorithm $(C_1,\ldots,C_{2n-1})$ = Cluster($\{1,\ldots,n\}$)
Input: A video database containing n videos $1,\ldots,n$.
Output: A binary decision tree of $2n-1$ cluster nodes $C_1,\ldots,C_{2n-1}$ having C_{2n-1} as the root. The i-th cluster node is defined as $C_i=(E_i,LC_i,RC_i)$ where E_i is the set of videos in that cluster, and LC_i and RC_i are the left and right child node of that cluster.
// Forming n initial clusters each having a single video
1. FOR $i=1,\ldots,n$
2. $C_i=(E_i=\{i\},LC_i=\text{null},RC_i=\text{null})$;
// Initializing the cluster set, which will be reduced to just the root cluster of the hierarchy
3. $S=\{C_1,\ldots,C_n\}$;
// At each iteration, the videos of the closest two clusters, using distance measure DClust,
// will be used to create a new cluster with these two clusters as the children
4. FOR $i=n+1,\ldots,2n-1$
5. $(C_l,C_r)=\underset{(C_j,C_k\neq C_j)\in S\times S}{\text{argmin}}\ DClust(C_j,C_k)$;
6. $C_i=(E_i=E_l\cup E_r,LC_i=C_l,RC_i=C_r)$;
7. $S=(S\cup\{C_i\})-\{C_l,C_r\}$;

Fig 3: Cluster computation using hierarchical clustering algorithm.

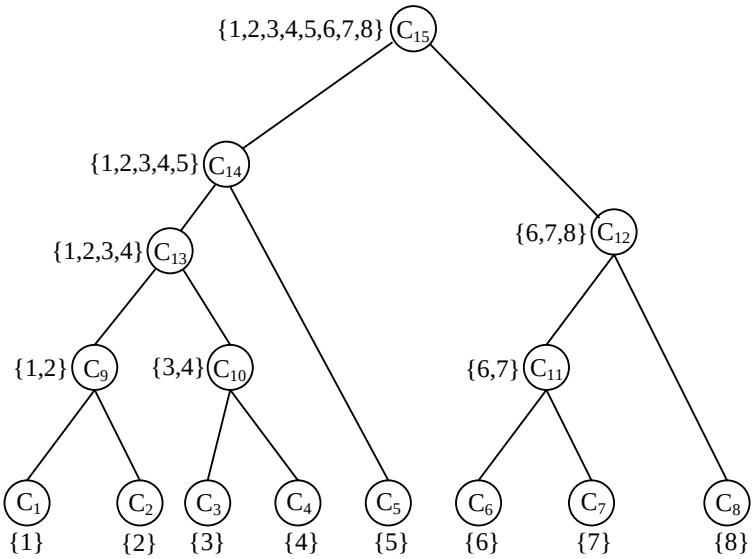

Fig 4: An example hierarchical clustering on eight videos. Each cluster is expressed by the set of videos in that cluster.

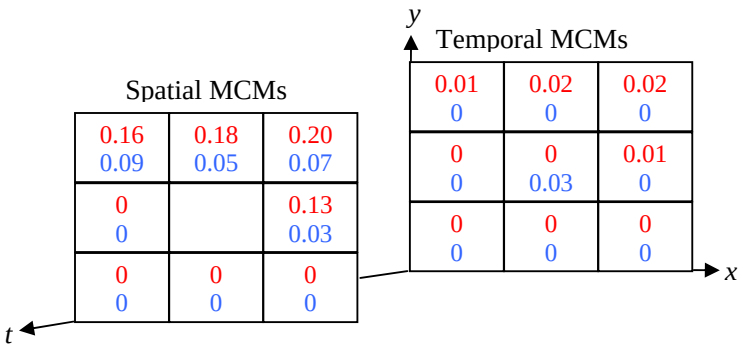

Fig 5: Feature weights obtained using the FFW approach using the GA feature weighting method. Along each clique, the weight of magnitude and direction MCMs are marked in red and blue respectively.

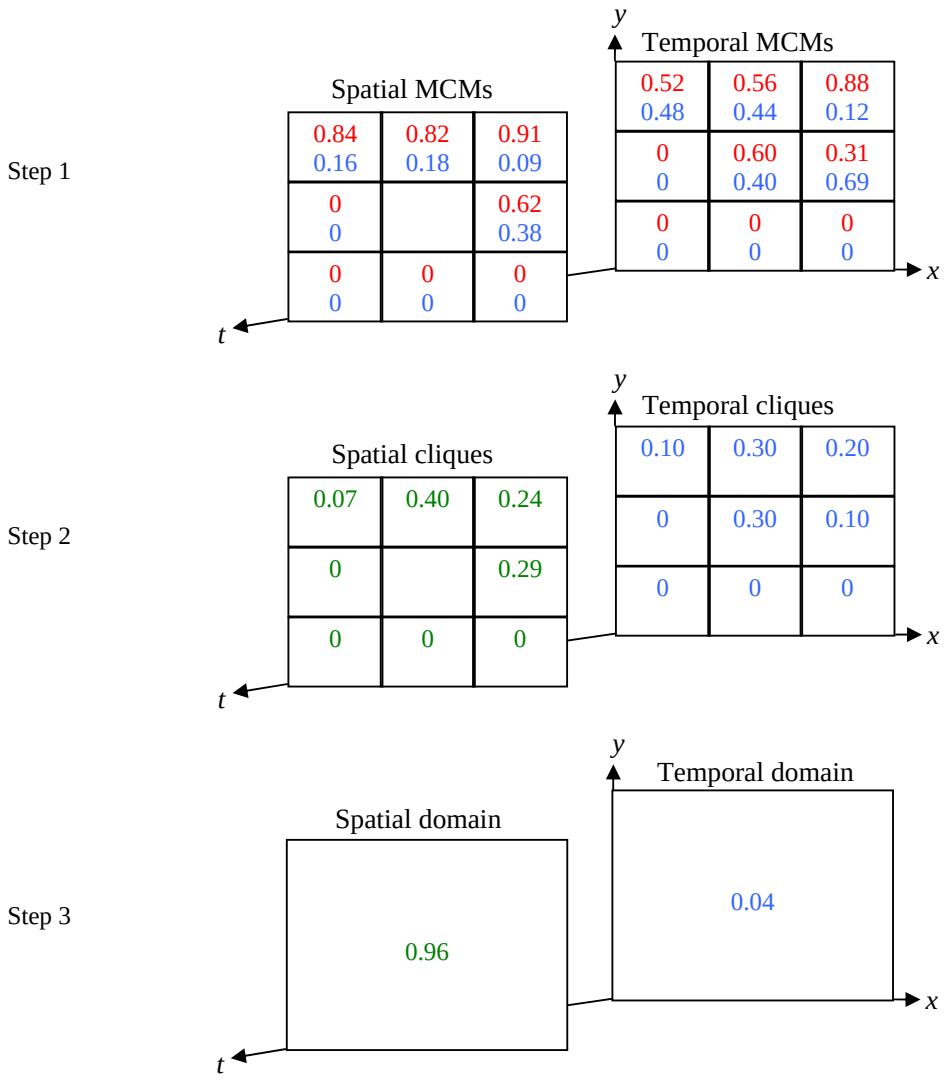

Fig 6: Feature weights obtained using the HFW approach — Step1: Along each clique weight of magnitude and direction MCMs are marked in red and blue respectively; Step2: Spatial and temporal clique weights are marked in green and blue respectively, and Step3: Spatial and temporal domain weight marked in green and blue respectively.

Table 1: Performance of different feature weighting approaches in terms of classification accuracy.

Feature weighting approach	Classification accuracy (%)
All features used with equal weight	87.30
FFW	95.24
HFW	96.03

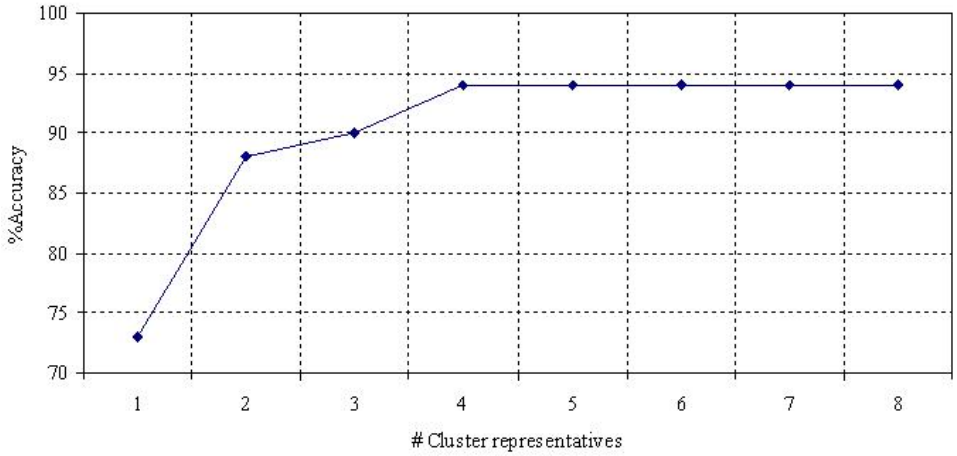

Fig 7: Retrieval accuracy of proposed MFCR method at a different number of cluster representatives. Results are reported using 1–NN classifier.

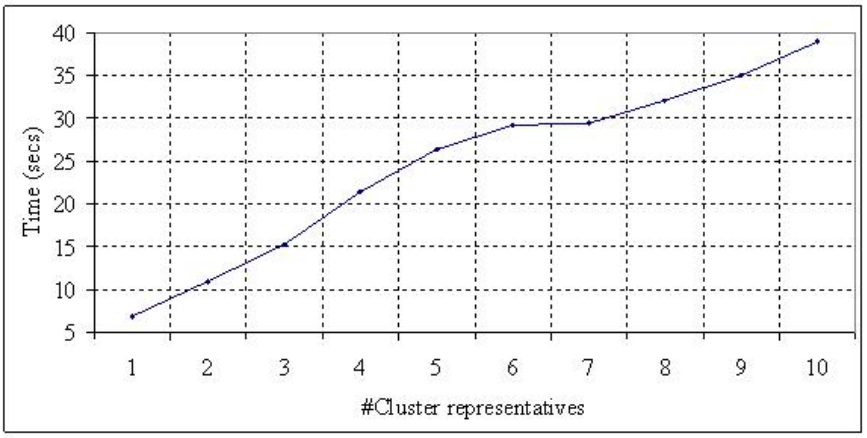

Fig 8: Average retrieval time of proposed MFCR method at a different number of cluster representatives.

Table 2: Retrieval accuracy(%) of MFCR, HRRC, and HREC retrieval methods. Results are reported for each method using 1–NN classifier.

Retrieval method	Retrieval accuracy (%)
MFCR	93.65
HRRC	85.85
HREC	75.66

References

1. A. Salam, M. Islam, and P. Sarker, http://partha.bengalit.net/paper/monovision_aiub, Last accessed in July 2007.
2. A. Rahman and M. Murshed, "Multiple Temporal Texture Detection Using Feature Space Mapping," Proceedings ACM International Conference on Image and Video Retrieval (CIVR), pp. 417–424, 2007.
3. G. Zhao and M. Pietikainen, "Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, pp. 915–928, 2007.
4. A. Rahman and M. Murshed, "A Temporal Texture Characterization Technique Using Block-based Approximated Motion Measure," IEEE transaction on circuits and systems for video technology (TCSVT), vol 17, no 10, pp 1370–1382, 2007.
5. D. Chetverikov and S. Fazekas, "On motion periodicity of dynamic textures," British Machine Vision Conference (BMVC), 2006.
6. S. Fazekas, D. Chetverikov, "Analysis and performance evaluation of optical flow features for dynamic texture recognition," SP:IC(22), No. 7–8, pp. 680–691, 2007.
7. P. Bouthemy and R. Fablet, "Motion characterization from temporal cooccurrences of local motion-based measures for video indexing," International Conference on Pattern Recognition (ICPR), vol. 1, pp. 905–908, Brisbane, Australia, 1998.
8. R. Fablet, P. Bouthemy, and P. Perez, "Nonparametric motion characterization using casual probabilistic models for video indexing and retrieval," IEEE Transactions on Image Processing, vol. 11, pp. 393–407, 2002.
9. R. C. Nelson and R. Polana, "Recognition of motion using temporal texture," IEEE computer society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 129–134, 1992.
10. C. H. Peh and L. F. Cheong, "Synergizing spatial and temporal texture," IEEE Transactions on Image Processing, pp. 1179–1191, 2002.
11. R. Péteri and D. Chetverikov, "Dynamic texture recognition using normal flow and texture regularity," Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA), LNCS, 2005.
12. A. Rahman and M. Murshed, "Real-time temporal texture characterization using block-based motion co-occurrence statistics," IEEE International Conference on Image Processing (ICIP), pp: 1593–1596, 2004.
13. A. B. Chan and N. Vasconcelos, "Efficient Computation of the KL Divergence between Dynamic Textures," Technical Report SVCL-TR-2004-02, November 2004.
14. A. K. Jain, A. Vailaya, and W. Xiong, "Query by video clip," Multimedia Systems, pp. 369–384, 1999.
15. J. Jarmulak and S. Craw, "Genetic Algorithms for Feature Selection and Weighting," Workshop on Automating the Construction of Case Based Reasoners (IJCAI), pp. 28–33, 1999.
16. S. Krishnamachari and M. A. Mottaleb, "Hierarchical clustering algorithm for fast image retrieval," IS&T/SPIE Confeernec on Storage and Retrieval for Image and Video Databases VII, 1999.
17. H. K. Lee and S. I. Yoo, "A neural network-based image retrieval using nonlinear combination of heterogeneous features," Congress on Evolutionary Computation, vol. 1, pp. 667–674, July, 2000.
18. Y. Liu and J. R. Kender, "Sort-Merge Feature Selection for Video Data," SDM, 2003.
19. M. A. Mottaleb, N. Dimitrova, R. Desai, and J. Martino, "CONIVAS: CONtent-based image and video access system," ACM International Multimedia Conference, November, 1996.
20. W. Niblack, "The QBIC Project: querying images by content using color, texture and shape," Storage and Retrieval for Image and Video Databases I, vol. 1908, SPIE Proceedings, February, 1993.
21. R. Paget, "Texture synthesis and analysis," http://www.vision.ee.ethz.ch/ ~rpaget/links.htm., Last accessed in January, 2006.
22. Y. Q. Shi and H. Sun, "Image and video compression for multimedia engineering," CRC Press, 2000.
23. J. Smith and S. F. Chang, "A Fully Automated Con-tent-based Image Query System," ACM International Multimedia Conference, November, 1996.
24. S. Soatto, G. Doretto, and Y. N. Wu, "Dynamic textures," IEEE International Conference on Computer Vision (ICCV), vol. 2, pp. 439–446, Vancouver, BC, Canada, July 2001.
25. M. J. Swain and D. H. Ballard, "Color Indexing," International Journal of Computer Vision, vol. 7, pp. 11–32, 1991.
26. M. Szummer and R. W. Picard, "temporal texture modeling," IEEE International Conference on Image Processing (ICIP), pp. 823–826, Lausanne, Switzerland, 1996.
27. D. Wettschereck, D. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," Artificial Intelligence Review, pp. 273–314, 1997.
28. D. S. Yeung and X. Z. Wang, "Improving performance of similarity-based clustering by feature weight learning," IEEE transactions on pattern analysis and machine intelligence, vol. 24, pp. 556–561, April, 2002.
29. H. Zhang, Y. Gong, C. Y. Low, and S. W. Smoliar, "Image retrieval based on color features: an evaluation study," SPIE proceedings, vol. 2606, pp. 212–220, 1995.
30. E. C. C. Tsang, S. C. K. Shiu, X. Z. Wang, and M. Lam, "Clustering and classification of cases using learned global feature weights," IFSA World Congress and NAFIPS International Conference, vol. 5, pp. 2971–2976, 2001.