Gesture Recognition by Learning Local Motion Signatures

Anonymous CVPR submission

Paper ID 1834

Abstract

*This paper overviews a new gesture recognition framework based on learning local motion signatures (LMSs) introduced by [1]. After the generation of these LMSs computed on one individual by tracking Histograms of Oriented Gradient (HOG) [3] descriptor, we learn a codebook of video-words (*i.e. *clusters of LMSs) using k-means algorithm on a learning gesture video database. Then the video-words are compacted to a codebook of code-words by the Maximization of Mutual Information (MMI) algorithm. At the final step, we compare the LMSs generated for a new gesture w.r.t. the learned codebook via the k-nearest neighbors (k-NN) algorithm and a novel voting strategy. Our main contribution is the handling of the N to N mapping between code-words and gesture labels with the proposed voting strategy. Experiments have been carried out on two public gesture databases: KTH [15] and IXMAS [18]. Results show that the proposed method outperforms recent state-of-the-art methods.*

1. Introduction

Gesture recognition from video sequences is one of the most important challenges in computer vision and behavior understanding since it enables to interact with some human machine interfaces (HMI) or to monitor complex human activities.

In this paper we overview a new learning-classification framework for gesture recognition using local motion signatures [1] as a gesture representation. First, we compute for each detected individual in the scene a set of features (*i.e*. corner points). For each feature, we associate a 2D descriptor (*i.e*. Histograms of Oriented Gradients (HOG) [3]), which is tracked over time to build a reliable local motion signature. Thus a gesture is represented as a set of local motion signatures. Second, we learn the local motion signatures for a given set of gestures by clustering them into local motion patterns (*i.e*. clusters). Last, we classify the gesture of a person in a new video by extracting the person local motion signatures and voting for the most likely gesture w.r.t. learned local motion patterns. The approach has been validated on two public gesture databases: KTH [15] and IXMAS [18] and results demonstrate an improvement over recent state-of-the-art methods.

The remaining of this paper is structured into six parts. The next section overviews the State-of-the-art in gesture recognition. Section 3 summarizes the building process of local motion signatures. Section 4 presents the learning stage and section 5 details the classification stage. Results are described and discussed in section 6. Finally, section 7 concludes this paper by overviewing the contributions and exposing future work.

2. Previous Work

In this section, we focus on overviewing the state-of-the-art of motion model based gesture recognition algorithms which contains two main categories: (1) global motion based methods and (2) local motion based methods. For global motion based methods, [19] have proposed to encode an action by an "action sketch" extracted from a silhouette motion volume obtained by stacking a sequence of tracked 2D silhouettes. The "action sketch" is composed of a collection of differential geometric properties (*e.g*. peak surface, pit surface, ridge surface) of the silhouette motion volume. For recognizing an action, the authors use a learning approach based on a distance and epipolar geometrical transformation for viewpoint changes. [10] propose to recognize gestures via maximum likelihood estimation with Hidden Markov Models and a global HOG descriptor computed over the whole body. The authors extend their method in [9] by reducing the global descriptor size with principal component analysis. [5] extract space-time saliency, space-time orientations and weighted moments from the silhouette motion volume. Gesture classification is performed using nearest neighbors algorithm and Euclidean distance. Recently, [2] introduce action signatures. An action signature is a 1D sequence of angles (forming a trajectory) which are extracted from a 2D map of adjusted orientation of the gradients of the motion-history image. A similarity measure is used for clustering and classification.

As these methods are using global motion, they strongly depend on the segmentation quality of the silhouette which influences the robustness of the classification. Furthermore, local motion, which can help to discriminate similar gestures, can easily get lost with a noisy video sequence or with repetitive self-occlusions.

Local motion based methods overcome these limits by considering sparse and local spatio-temporal descriptors more robust to short occlusions and to noise. For instance, [16] propose a 3-D (2D + time) SIFT descriptor and apply it to action recognition using the bag of word paradigm. [15] propose to use Support Vector Machine classifier with local space-time interest points for gesture categorization. [11] introduce local motion histograms and use an Adaboost framework for learning action models. More recently, [8] apply Support Vector Machine learning on correlogram and spatial temporal pyramid extracted from a set of video-word clusters of 3D interest points.

These methods are generally not robust enough since the temporal local windows (with short size and fixed spatial position) do not model the exact local motion but arbitrarily several slices of that motion instead.

To go beyond the state of the art, we propose a novel gesture learning-classification framework based on learning local motion signatures which are built thanks to tracking local HOG descriptors over sufficiently long period of time. The proposed gesture representation combines the advantages of global and local gesture motion approaches in order to improve the recognition quality.

3. Local Motion Signatures Generation

Our gesture representation is a set of Local Motion Signatures (LMSs) which are generated through two steps based on [1]: (1) People Detection/Feature Selection and (2) HOG Descriptor Generation/Tracking.

3.1. People Detection and Feature Selection

People detection is performed by background subtraction to determine moving regions followed by a morphological dilation. Then a people classifier is applied to determine bounding boxes around single individuals. The people bounding boxes define a mask for feature point extraction. This step not only limits the search space for feature points but also separates distinct moving regions: corresponding to different individuals. This enables to apply the gesture recognition process to different people until they overlap each other.

Feature selection is then performed for each detected person using Shi-Thomasi corner detector [17] or Features from Accelerated Segment Test (FAST) corner detector [14]. Then corner points are sorted in decreasing order according to the corner strength. After that, we select the most significant corners by ensuring a minimum distance among them. Thus, feature points enable us to localize points where HOG descriptors can be computed since they usually correspond to locations where motion can be easily discernable.

3.2. HOG Descriptor Generation and Tracking

For each feature point, we compute a local HOG descriptor [3] from a descriptor block composed of 3×3 cells; each of them having a pixel size of 5×5: Therefore, the local HOG descriptor is a vector concatening the nine cell histograms of the descriptor block.

Local motion signatures are built by tracking HOG descriptors. Let us suppose that we have detected a HOG descriptor d_{t-1} in the frame f_{t-1}, we are now interested to determine the descriptor d_t in the frame f_t which can be identified to d_{t-1}. The basic idea is to minimize a quadratic error function $\mathcal{E}(d_t, d_{t-1})$ in a neighborhood $\mathcal{V}_{f_t}$ in the frame f_t corresponding to the predicted position of d_{t-1} obtained by an extended Kalman filter. In the case when several descriptors $(d_t^1, d_t^2, ..., d_t^k)$ in this neighborhood satisfy the minimum of the error function, we compute the visual evidence (intensity difference in gray-scale) between each descriptor and the descriptor of the previous frame to track d_{t-1}. The tracker will choose the descriptor that has the nearest visual evidence to d_{t-1}. For each tracked HOG descriptor, we define the temporal HOG descriptor as the vector obtained by the concatenation of the final descriptor estimate $\hat{d}$ and the positions of the descriptor during the tracking process. A local motion signature (LMS) is built from a temporal HOG descripor by computing the angle trajectory and then applying Principal Component Analysis (PCA) to select the three first principal axes (*c.f.* [1]).

4. Gesture Learning

4.1. K-means Clustering

For learning gestures, we assume that the training dataset is built with videos, each of them containing one and only one gesture instance. For each training video sequence, LMSs are extracted and annotated with the corresponding gesture label. Then, we apply the k-means algorithm in order to group these LMSs into clusters called video-words which are the local motion patterns. The similarity measure used for comparing two different LMSs in k-means is the Euclidean distance. Indeed we have carried out different experiments with different distances and found out that the euclidean distance gives the best results.

Thus, given as input the set S of annotated local motion signatures generated with all the training videos, the k-means algorithm outputs k video-words $S_i, i = 1, 2, ..., k$. The value of k is empirically chosen so that is large enough to describe correctly the set of the m gestures to learn $(k > 3 \times m)$. The lower bound for choosing of k is justified by analyzing the videos illustrating gestures [13]: gestures are usually composed of three units of coherent motion (*i.e.* pre-stroke, stroke and post-stroke) and in our representation, these units of motion correspond to local motion patterns. Thanks to the cluster membership map provided by the k-means algorithm, we annotate each video-word with the gesture labels associated with the LMSs of this cluster.

4.2. Maximization of Mutual Information

Once we have obtained the video-words, an optional step is to reduce their dimensionality by compacting them into code-words. To achieve this goal, we propose to apply Maximization of Mutual Information (MMI) algorithm [8] on the clusters generated by the k-means algorithm. Let C be the centroids of the generated clusters $C \in \mathcal{C} = [\mu_1..\mu_k]$. Let G be the gesture labels $G \in \mathcal{G} = [g_1..g_m]$. With the cluster membership map $A : S \rightarrow \mathcal{C}$, we define the conditional probability distributions $P(C|G)$ and $P(G|C)$ by equations 1 and 2.

$$P(C = \mu_i | G = g_i) = \frac{Card(Label^{-1}(g_i) \cap A^{-1}(\mu_i))}{Card(Label^{-1}(g_i))} \tag{1}$$

$$P(G = g_i | C = \mu_i) = \frac{Card(Label^{-1}(g_i) \cap A^{-1}(\mu_i))}{Card(A^{-1}(\mu_i))} \tag{2}$$

Where the function $Label$ is the map between feature LMSs and gesture labels; $Card(.)$ is the cardinal operator. By taking as definition of the marginal distributions of C and G the formulas 4 and 5, we can verify that these definitions (*i.e.* equations 1 and 2) match the conditional probability definition (*c.f.* equation 3).

$$\begin{aligned} P(G = g_i | C = \mu_i) &= \frac{P(G = g_i, C = \mu_i)}{P(C = \mu_i)} \\ &= \frac{P(C = \mu_i | G = g_i)\ P(G = g_i)}{P(C = \mu_i)} \end{aligned} \tag{3}$$

$$P(C = \mu_i) = \frac{Card(A^{-1}(\mu_i))}{Card(S)} \tag{4}$$

$$P(G = g_i) = \frac{Card(Label^{-1}(g_i))}{Card(S)} \tag{5}$$

Thus, we can deduce the joint distribution of C and G from equation 3 which gives:

$$P(G = g_i, C = \mu_i) = \frac{Card(Label^{-1}(g_i) \cap A^{-1}(\mu_i))}{Card(S)} \tag{6}$$

Hence, the mutual information between C and G which measures how much information from C is contained in G is:

$$MI(C, G) = \sum_{\mu_i \in \mathcal{C}, g_i \in \mathcal{G}} P(C = \mu_i, G = g_i) \log \frac{P(C = \mu_i, G = g_i)}{P(C = \mu_i) P(G = g_i)} \tag{7}$$

The goal of MMI algorithm is to reduce incrementally the size of the video-words $\mathcal{C}$ in order to obtain a compact set of code-words $\hat{\mathcal{C}}$ by keeping the value of $MI(\hat{C}, G)$ as high as possible and the value of $MI(\hat{C}, C)$ (which measures the compactness of $\hat{\mathcal{C}}$ with respect to $\mathcal{C}$) as low as possible. At each step of the algorithm, the pair of video-words that gives the minimum loss of mutual information when merged, is chosen. The merge is actually done if and only if the loss of mutual information (*c.f.* formula 8) generated by the merge of this optimal pair is not larger than a predefined threshold ϵ or if the minimal number of clusters is reached. Before the optimization process, the set $\mathcal{C}$ corresponds to video-words and after that process, the optimal set $\hat{\mathcal{C}}$ corresponds to code-words.

So, the trade-off between the compactness of the optimal set and the discrimination criterion (maximum of mutual information) when merging video-words μ_i and μ_j can be solved by equation 8(*c.f.* Liu & Shah 2008 [8]).

$$\Delta MI(\mu_i, \mu_j) = \sum_{k \in \{i,j\}} P(C = \mu_k) D_{KL}(P(G|C = \mu_k) || Q(G|C = \mu)) \tag{8}$$

Where $D_{KL}(.||.)$ is the Kullback-Leibler divergence (*c.f.* formula 9), $Q(G = g|C = \mu)$ is defined by equation 10 and μ is the resulting merged video-word.

$$D_{KL}(P(.|y)||Q(.|z)) = \sum_x P(x|y) \log(\frac{P(x|y)}{Q(x|z)}) \tag{9}$$

$$Q(G = g | C = \mu) = \frac{P(C = \mu_i) P(G = g | C = \mu_i)}{P(C = \mu_i) + P(C = \mu_j)} + \frac{P(C = \mu_j) P(G = g | C = \mu_j)}{P(C = \mu_i) + P(C = \mu_j)} \tag{10}$$

The non-recursive version of the MMI algorithm is described hereafter (Algorithm 1). Note that $\otimes$ is the merging operator which is applied to two video-words.

Compared to the code-words of [8], our code-words already integrate the spatio-temporal structural information which is not the case of the formers. Indeed, our code-word is a compact information of local motion signature clusters which can caractherize directly gestures.

Algorithm 1 Maximization of Mutual Information (MMI) Algorithm

Require: $\mathcal{C}, \mathcal{G}, C, G$ {inputs}
Ensure: $\hat{\mathcal{C}}, \hat{C}$ {outputs}
$\hat{\mathcal{C}} \leftarrow \mathcal{C}$
$minimalLoss \leftarrow 0$
while $minimalLoss < \epsilon \;\&\; Card(\hat{\mathcal{C}}) > Card(\mathcal{G})$ **do**
  $minimalLoss \leftarrow \infty$
  for all $\mu_i, \mu_j \in \hat{\mathcal{C}} / \mu_i \neq \mu_j$ **do**
    Compute $\Delta MI(\mu_i, \mu_j)$
    if $\Delta MI(\mu_i, \mu_j) < minimalLoss$ **then**
      $minimalLoss \leftarrow \Delta MI(\mu_i, \mu_j)$
      $merge_i \leftarrow \mu_i$
      $merge_j \leftarrow \mu_j$
    end if
  end for
  if $minimalLoss < \epsilon \& [Card(\hat{\mathcal{C}}) - 1] > Card(\mathcal{G})$ **then**
    $\hat{\mathcal{C}} \leftarrow \hat{\mathcal{C}} - \{merge_i, merge_j\}$
    $\hat{\mathcal{C}} \leftarrow \hat{\mathcal{C}} \cup \{merge_i \otimes merge_j\}$
    Compute the new conditional density $\hat{C}$
    $C \leftarrow \hat{C}$
  end if
end while

5. Gesture Classification

5.1. Offline Recognition

The k-nearest neighbor algorithm is one of the most common classifier in the literature. The main idea behind this algorithm is to select the k-nearest neighbors (*i.e.* code-words) of an input LMSs and then assign it to the gesture label that casts a majority vote. In order to obtain always a majority vote, the "k" parameter is usually an odd number to prevent tie cases. The main advantage of this algorithm is that it is an universal approximator and can model any many-to-one mapping very well. The drawbacks consist of the lack of robustness for high dimension spaces and low computational complexity with huge training data-set. In order to adapt this algorithm to our training data-set, we must cope with the many-to-many mapping between code-words and gesture labels. A suitable solution is to make a voting mechanism which transforms this mapping into a many-to-one mapping.

Let $\mathcal{T} = \{(c, g)/c \in \mathcal{C} \& g \in \mathcal{G} \& g \in Label^{-1}(c)\}$ our final learned database with cardinal N. The likelihood $L(c|g)$ of a particular cluster c given a gesture g is defined by equation 11.

$$L(c|g) = P(G = g | C = c) \tag{11}$$

We define the likelihood **measure** of a gesture g according to k observed clusters $c'_i,\ i \in [1..k]$ by:

$$L(g|c'_1,\ \ldots, c'_k) = \frac{\sum_{i=1}^{k} L(c'_i|g)}{\sum_{h \in \mathcal{G}} \sum_{i=1}^{k} L(c'_i|h)} \tag{12}$$

Note that this likelihood measure satisfies the equation 13.

$$\sum_{g \in \mathcal{G}} L(g|c'_1,\ \ldots, c'_k) = 1 \tag{13}$$

During the classification process, testing a video sample generates several LMSs $lms_i,\ i \in [1..M]$. Each descriptor casts votes for k nearest code-words. If we note $L(g|lms_i)$ the likelihood measure of a gesture g according to the k nearest code-words from lms_i, then the gesture associated to the sample is defined by equation 14 and its recognition likelihood RL is defined by equation 15.

$$g_{recognized} = \arg\max_{g \in \mathcal{G}} \sum_{i=1}^{M} L(g|lms_i) \tag{14}$$

$$RL(g_{recognized}) = \frac{\sum_{i=1}^{M} L(g_{recognized}|lms_i)}{M} \tag{15}$$

When ties (*i.e.* several gestures with the same likelihood) occur, the classifier is unable to classify the new input. Then, the new input is fed to the learner which prompts the user for the gesture label. Two cases can be distinguished:

- The new gesture has been already learned: The user decides which gesture wins the vote and the learned clusters are updated according to this choice.
- The new gesture has not been learned: The user gives the appropriate gesture label and existing clusters are updated and eventually new clusters are created for the new gesture label.

Algorithm 2 describes the modified version of the k-nearest neighbors for our learning-classification framework. This version of the algorithm supposes that each test sequence contains one and only one gesture.

5.2. Online Recognition

Now, we are interested to adapt this algorithm for on-line recognition where several gestures can occur in a video sequence. For that purpose, we cannot wait for all local motion signatures to be computed in order to estimate the likelihood of gesture recognition. So we derive a recursive equation from equation 14 by considering that local motion

Algorithm 2 k-nearest neighbors - offline version

Require: $\mathcal{T}$ {The training data-set}
$lms_i,\ i >= 1$ {The generated local motion signatures from the test sequence}
Ensure: $g_{recognized}, recognitionLikelihood(g_{recognized})$

$M \leftarrow 1$
while an lms_i is generated **do**
  execute the usual k-nearest neighbors for lms_i
  $g^M_{recognized} \leftarrow \arg\max_{g \in \mathcal{G}} Likelihood^M(g)$
  Compute $RL(g_{recognized})$
  $M \leftarrow M + 1$
end while

signatures $lms_i,\ i \in [1..M]$ are indexed by their chronological order of computation which gives the equation 16.

$$g^M_{recognized} = \arg\max_{g \in \mathcal{G}} Likelihood^M(g) \quad (16)$$

Where $Likelihood^1(g) = L(g|lms_1)$ and for $M > 1$, $Likelihood^M(g)$ verifies the recursion defined by equation 17.

$$\begin{aligned} Likelihood^M(g) =& Likelihood^{M-1}(g) \\ &+ \frac{1}{M}(L(g|lms_M) - Likelihood^{M-1}(g)) \end{aligned} \quad (17)$$

In addition, we must integrate the time duration of a gesture in the learning-classification process to decide when to stop the recognition process and to start a new one. We assume that the duration of any gesture is ruled by a duration of life law (*i.e.* poisson law). So, the samples (*i.e.* videos) of the training data-set for a given gesture are instances of a random variable with exponential distribution. We know that if p is the number of instances of a gesture in the training data set and $\ell_i,\ i \in [1..s]$ are the duration of these samples, then a $100(1-\alpha)\%$ exact confidence interval for the mean duration $\frac{1}{\lambda}$ is given by equation 18.

$$\frac{1}{\hat{\lambda}} \frac{2s}{\chi^2_{2s;\alpha/2}} < \frac{1}{\lambda} < \frac{1}{\hat{\lambda}} \frac{2s}{\chi^2_{2s;1-\alpha/2}} \quad (18)$$

Where $\frac{1}{\hat{\lambda}}$ is defined by equation 19 and $\chi^2_{k;x}$ is the value of the chi squared distribution with k degrees of freedom that gives x cumulative probability.

$$\frac{1}{\hat{\lambda}} = \frac{\sum_{i=1}^{p} \ell_i}{p} \quad (19)$$

Then, we can consider that a gesture is recognized if and only if its duration is in the confidence interval and we have reached a local maximum of likelihood. So the on-line version of the k-nearest neighbors can be described by algorithm 3. To use this online-version, we can compute a sliding window algorithm [6] which detects the prestroke phase of gestures, calls the on-line classifier and solves the issue of overlapping prestrokes.

Algorithm 3 k-nearest neighbors - on-line version

Require: $\mathcal{T}$ {The training data-set}
$lms_i,\ i >= 1$ {The generated local motion signatures from the test sequence}
Ensure: $g_{recognized}, recognitionLikelihood(g_{recognized})$

$M \leftarrow 1$
$duration \leftarrow 0$
$previousLikelihood \leftarrow 0$
repeat
  $duration \leftarrow duration + 1$
  save the previous likelihood if any in $previousLikelihood$
  while a lms_i is generated **do**
    execute the usual k-nearest neighbors for lms_i
    $g^M_{recognized} \leftarrow \arg\max_{g \in \mathcal{G}} Likelihood^M(g)$
    Compute $RL(g^M_{recognized})$
    $M \leftarrow M + 1$
  end while
until $duration \in confidenceInterval(g_{recognized})$ & $previousLikelihood > RL(g^M_{recognized})$

6. Experiments and Results

6.1. Gesture Recognition on KTH Database

The KTH database [15] contains 600 videos illustrating six actions/gestures: (1) walking, (2) jogging, (3) running, (4) hand waving, (5) hand clapping and (6) boxing. Each action/gesture is performed many times by 25 actors for four different scenarios. Thus, there are $4 \times 6 = 24$ videos per actor. The database is split into three independent data-sets: (1) a training data-set (8 actors), (2) a validation data-set for tuning parameters (8 actors) and (3) a testing data-set for evaluation (9 actors). All videos from this database were taken over homogeneous background thanks to a static camera with 25fps frame rate. The spatial resolution of each video is 160x120 pixels. We train our algorithm on the KTH training data-set and test it on the corresponding test data-set. All the parameters of the framework have been tuned using the validation data-set. Since a gesture is composed of three motion patterns (*c.f.* section 4), we have tested all the values of k between 18 and 57 for the k-means clustering algorithm. We realized that the best classification results is when $k = 27$. Finally, the best value of the k parame-

Table 1. Confusion matrix for the classification on KTH database using Shi-Tomasi (upper values) and FAST corner points (lower values).

	W.	J.	R.	B.	H.C.	H.W.
W.	**0.95 / 0.97**	0.03 / 0.03	0.02 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00
J.	0.03 / 0.02	**0.85 / 0.91**	**0.10 / 0.07**	0.02 / 0.00	0.00 / 0.00	0.00 / 0.00
R.	0.05 / 0.03	0.07 / 0.05	**0.88 / 0.92**	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00
B.	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	**0.95 / 0.97**	0.03 / 0.02	0.02 / 0.01
H.C.	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.05 / 0.03	**0.88 / 0.92**	0.07 / 0.05
H.W.	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.02 / 0.01	0.01 / 0.00	**0.97 / 0.99**

Table 2. Comparison of different results of the KTH database.

Method	Variant	Precision
Our method	Shi-Tomasi	91.33%
	FAST	**94.67%**
Liu and Shah [8]	SVM VWCs	91.31%
	VWC Correl.	94.16%
Luo et al. [11]		85.10%
Kim et al. [7]		95.33%

Table 3. Precision, recall and F-score of the proposed method on KTH database.

	Precision	Recall	F-score
with Shi-Tomasi	91.33%	99.07%	95.04%
with FAST	94.67%	99.78%	**97.15%**

ter of the k-nearest neighbor algorithm is $k = 5$. Results are illustrated by the confusion matrix 1 and are compared to the state of the art methods in table 2. We obtain better or slightly better results than recent methods. We also find out that FAST corners outperform Shi-Tomasi corners which is consistent with results in [14]. Note that even if [7] obtain slightly better results, their results are not comparable to ours since they use a different experimental protocol (Leave-one-out cross-validation) which includes more learning videos and enables to train better code-words. Table 3 shows the performance metrics (*i.e.* precision, recall and F-score) of the proposed framework. It has a high sensitivity which means that there is few false negatives. However, the precision can be improved.

6.2. Gesture Recognition on IXMAS Database

The IXMAS database [18] contains 468 action clips for 13 gestures and each of them is performed three times by 12 actors. Each video clip has a spatial resolution of 390x291 pixels, a frame-rate of 23fps and it is captured by five cameras from different points of view (*i.e.* five video sequences for each clip). The gestures of the database are : (1) check watch, (2) cross arms, (3) scratch head, (4) sit down, (5) get up, (6) turn around, (7) walk, (8) wave, (9) punch, (10) kick, (11) point, (12) pick up and (13) throw. For this gesture database, we adopt a leave-one-out cross-validation scheme. Since each action is captured from five points of view, we have selected $k = 197$ for the k-means clustering algorithm. As said in section 4, the three phases of a gesture (*i.e.* prestroke, stroke and poststroke) can generate different 2D motion patterns from different points of view. So for each action, we can expect $3 \times 5 = 15$ motion patterns. Due to the fact that the IXMAS database contains 13 gestures, the expectation grows to $13 \times 15 = 195$ motion patterns. For the learning phase, we use two learning procedures: (1) without MMI and (2) with MMI. For the classification phase, we use the same value of the k parameter (*i.e.* 5) as for KTH database. The classification is carried out independently for each gesture video corresponding to the remaining actor (*i.e.* discarded by the leave-one-out rule). So, for each actor (1 out of 12), each gesture (1 out of 13) and at a particular step of the cross-validation, there are $5 \times 3 = 15$ video sequences (5 views and 3 manners) to be classified versus $11 \times 5 \times 3 = 165$ video sequences used for learning. In addition, we choose to carry out this experiment with the FAST corner detector only since it gives better results on KTH database. The confusion matrix for this experiment is given in table 4 and table 6 presents the performance metrics. We compare the results of our method to those of [18] in table 5. Note that the results with the compacted learned database (*i.e.* using the MMI algorithm) are slightly better (7%) than the ones with the non-compacted version. Unsurprisingly gestures with large motion (*e.g.* sit down, get up, turn around, walk) are much better recognized than gestures with small motion (*e.g.* scratch head, wave, point) which besides, share some common motion patterns. The mean processing time of the offline k-NN classifier for the IXMAS database is 35 seconds per gesture which is quite reasonnable knowing that a gesture is in mean depicted by 80 frames.

A multi-view experiment has been also carried out excluding the fifth view point (*i.e.* the top view) in order to compare the results with [8]. We learn gestures from three selected views and classify gestures with the remaining view. We repeat the experiment for all possible combinations. Only the version with MMI algorithm has been tested. Table 7 overviews the average precision for each experiment. We can notice an improvement w.r.t. [8]. we can see that the first and the last view points are more dependent on other view points hence they achieve better precision.

	C.W.	C.A.	S.H.	S.D.	G.U.	T.A.	Wl.	Wv.	Pu.	K.	Po.	P.U.	T.
C.W.	$\frac{0.87}{0.93}$	$\frac{0.05}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.05}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.01}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
C.A.	$\frac{0.10}{0.09}$	$\frac{0.75}{0.81}$	$\frac{0.05}{0.04}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.01}$	$\frac{0.03}{0.01}$	$\frac{0.00}{0.00}$	$\frac{0.04}{0.02}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
S.H.	$\frac{0.07}{0.06}$	$\frac{0.08}{0.07}$	$\frac{0.69}{0.73}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.03}$	$\frac{0.03}{0.02}$	$\frac{0.07}{0.06}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
S.D.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{1.00}{1.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
G.U.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{1.00}{1.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
T.A.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{1.00}{1.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
Wl.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.05}{0.03}$	$\frac{0.95}{0.97}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
Wv.	$\frac{0.05}{0.05}$	$\frac{0.03}{0.03}$	$\frac{0.12}{0.10}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.62}{0.67}$	$\frac{0.03}{0.03}$	$\frac{0.03}{0.03}$	$\frac{0.09}{0.07}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.02}$
Pu.	$\frac{0.04}{0.03}$	$\frac{0.04}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.02}{0.02}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.03}$	$\frac{0.75}{0.80}$	$\frac{0.07}{0.05}$	$\frac{0.00}{0.00}$	$\frac{0.01}{0.01}$	$\frac{0.04}{0.03}$
K.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.01}$	$\frac{0.97}{0.99}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
Po.	$\frac{0.03}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.06}{0.05}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.03}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.08}{0.07}$	$\frac{0.20}{0.16}$	$\frac{0.03}{0.03}$	$\frac{0.57}{0.63}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$
P.U.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.05}{0.05}$	$\frac{0.00}{0.00}$	$\frac{0.02}{0.00}$	$\frac{0.03}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.01}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.89}{0.95}$	$\frac{0.00}{0.00}$
T.	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.05}{0.05}$	$\frac{0.00}{0.00}$	$\frac{0.04}{0.03}$	$\frac{0.00}{0.00}$	$\frac{0.00}{0.00}$	$\frac{0.06}{0.05}$	$\frac{0.03}{0.01}$	$\frac{0.00}{0.00}$	$\frac{0.06}{0.05}$	$\frac{0.00}{0.00}$	$\frac{0.76}{0.81}$

Table 4. Confusion matrix for the classification on IXMAS database using FAST corner points (uppervalues without MMI and lowervalues with MMI).

Table 5. Comparison of different results of the IXMAS database.

Method	Variant	Precision
Our method	without MMI	83.23%
	with MMI	**90.57%**
Weinland et al. [18]		81.27%
Lv et al. [12]		80.60%
Liu & Shah [8]		82.80%

Table 6. Precision, recall and F-score of the proposed method on IXMAS database.

	Precision	Recall	F-score
without MMI	83.23%	87.46%	85.29%
with MMI	90.57%	85.72%	**88.07%**

Table 7. Multi-view results for IXMAS database: precision of the classification of each view.

Method	cam1	cam2	cam3	cam4
Our method	75.34%	67.11%	69.52%	74.95%
Liu & Shah [8]	72.29%	61.22%	64.27%	70.59%

6.3. Discussion

The value of the k parameter for both k-means and k-nearest neighbor algorithms is mainly dependent on the number of gestures to be recognized, the gesture database size (*i.e.* number of gestures, number of view points per gesture). Indeed, when we process a multiview database of n gestures all captured under m view points, the constraint on the parameter k of the k-means algorithm becomes $k > 3 \times n \times m$ since local motion patterns for each gesture phase (*i.e.* prestroke, stroke and poststroke) can be different for the view points. To avoid tuning parameter k, the Mean Shift clustering algorithm [4] and the SVM classifier can be used. Also, the time precedence constraints among local motion signatures is to be studied. However, when the different gestures to recognize are composed of different local motion patterns, this requirement is not necessary. Nonetheless, if we want to differentiate two very similar gestures sharing the same motion patterns but in different timeline order then it will be convenient to use it.

7. Conclusion

A novel learning-classification framework using local motion signatures has been proposed for gesture recognition. Our main contribution is the gesture representation combining advantages of global and local gesture motion models. The local motion signatures can be considered as a local version of the global action signature proposed by [2] with the advantage of capturing also local motion. Compared to the global PCA-HOG descriptor proposed by [9] (one global HOG descriptor for each gesture/action), the proposed gesture/action representation consists of a set of local signatures which accounts more faithfully for local motion. Instead of computing a global HOG volume for a person already tracked, we use local HOGs tracked independently. Our method contrasts from common local motion methods by tracking salient HOG descriptors instead of computing arbitrary time-volume of HOG descriptors. We propose also a novel voting mechanism to deal with the many-to-many mapping between video-words and gesture labels. Results show an improvement w.r.t. recent state-of-the-art methods. As future work, we plan to validate the online version of the proposed classifier on real-world video databases like Homecare applications. The gesture representation can be enhanced to enable the detection of the frailty level of elderly people by analyzing the way people are sitting down or getting up from a chair (*e.g.* characterizing the gesture manner, the gesture speed). This protocol is already used by doctors for evaluating elderlies and the challenge is to automate this protocol.

References

[1] A. Anonymous. Tracking hog descriptors for building local motion signatures. *AnonymousConference*, 0, 200x. 1, 2

[2] S. Calderara, R. Cucchiara, and A. Prati. Action signature: A novel holistic representation for action recognition. In *International Conference on Advanced Video and Signal Based Surveillance*, pages 121–128, Washington, DC, USA, 2008. IEEE Computer Society Press. 1, 8

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, San Diego, CA, USA, June 20-25 2005. IEEE Computer Society Press. 1, 2

[4] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: A texture classification example. In *International Conference on Computer Vision*, volume 1, page 456, Los Alamitos, CA, USA, 2003. IEEE Computer Society Press. 7

[5] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007. 1

[6] D. Kim, J. Song, and D. Kim. Simultaneous gesture segmentation and recognition based on forward spotting accumulative hmms. *Pattern Recognition*, 40(11):3012–3026, 2007. 5

[7] T.-K. Kim, S.-F. Wong, and R. Cipolla. Tensor canonical correlation analysis for action classification. In *International Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007. 6

[8] J. Liu and M. Shah. Learning human actions via information maximization. In *International Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, Alaska, USA, June 24-26 2008. IEEE Computer Society Press. 2, 3, 6, 7

[9] W.-L. Lu and J. J. Little. Simultaneous tracking and action recognition using the pca-hog descriptor. In *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, pages 6–13, Quebec, Canada, June 2006. 1, 8

[10] W. L. Lu and J. J. Little. Tracking and recognizing actions at a distance. In *Proceedings of the ECCV Workshop on Computer Vision Based Analysis in Sport Environments (CVBASE '06)*, Graz, Austria, May 2006. 1

[11] Q. Luo, X. Kong, G. Zeng, and J. Fan. Human action detection via boosted local motion histograms. *Machine Vision and Applications*, November 2008. 2, 6

[12] F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *International Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society Press, June 18-23 2007. 7

[13] D. McNeill. *Hand and Mind: What Gestures Reveal about Thought*. University Of Chicago Press, 1992. ISBN: 9780226561325. 3

[14] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, Graz, Austria, May 7-13 2006. Springer. 2, 6

[15] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *International Conference on Pattern Recognition*, volume 3, pages 32–36, Cambridge, UK, August 23-26 2004. IEEE Computer Society Press. 1, 2, 5

[16] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *ACM International Conference on Multimedia*, pages 357–360, New York, NY, USA, 2007. ACM. 2

[17] J. Shi and C. Tomasi. Good features to track. In *International Conference on Computer Vision and Pattern Recognition*, pages 593–600, Seattle, WA, USA, June 1994. Springer. 2

[18] D. Weinland, E. Boyer, and R. Ronfard. Action recognition from arbitrary views using 3d examplars. In *International Conference on Computer Vision*, pages 1–7, Rio de Janeiro, Brazil, October 14-21 2007. IEEE Computer Society Press. 1, 6, 7

[19] A. Yilmaz and M. Shah. A differential geometric approach to representing the human actions. *Computer Vision and Image Understanding*, 109(3):335–351, 2008. 1