Belief Scheduler for sequential data analysis based on model failure detection in TBM framework applied to activity recognition in athletics videos

E. Ramasso$^a$, C. Panagiotakis$^b$, M. Rombaut$^a$, D. Pellerin$^a$

$^a$GIPSA-lab, Image and Signal Department (DIS), 961 Rue de la Houille Blanche, BP46, 38402 Saint Martin d’Hères, France.

$^b$Computer Science Department, University of Crete, P.O. Box 2208, Heraklion, Greece.

Abstract

In this paper, a generic architecture based on the Transferable Belief Model (TBM) is proposed for on line and automatic human action and activity recognition in videos of athletics sports with moving camera. The first contribution of this paper is a tool called Belief Scheduler which makes belief on actions smoother, separates actions states (true or false) and synchronizes actions in order to infer activity described as a sequence of actions. The scheduler is based on a temporal evidential filter proposed in a previous paper. The second contribution of this paper concerns a quality criterion which allows to assess the performance of actions and activities recognition and to infer activity using the scheduler. This criterion is based on the conflict between a state model of evolution and observations and it is quantified using TBM conjunctive rule. In order to assess the robustness of the system, we have analyzed 69 real athletics video sequences (coming from TV and DVD) acquired by a moving camera under several view angles. Videos are of low quality without any constraint on the environment conditions (indoor, outdoor, other moving people). The goal is to automatically recognize actions,
running, jumping, falling and standing up, and activities, high jump, pole vault, triple jump and long jump of one athlete that the camera tracks. The Belief Scheduler is compared to Hidden Markov Models for video classification.

Key words: Sequence recognition, Belief Scheduler, Belief state machine, Transferable Belief Model, Temporal Evidential Filter, Conflict, Videos analysis, Human motion analysis.

1. Introduction

1.1. Context

Human motion analysis is an important topic of interest in the Computer Vision and Video Processing communities. Research in this domain is motivated by the diversity of applications such as automatic surveillance [1], video indexing and retrieval [2, 3], human-computer interaction [4] and biometrics [5].

The analysis of human motions generally consists in human detection, tracking and activity understanding [6, 7]. Detection and tracking aim at locating human limbs. Activity understanding is a high level task aiming at recognizing human actions and using sequences of actions to recognize activities and sometimes interaction [8, 9]. Hidden Markov Models (HMM), initially proposed for speech processing [10] are the most common methods used for human action and activity recognition. As most of approaches in human motion analysis, they rely on Probability Theory. Several drawbacks inherent to these usual methods can be cited [4, 8]. First, intensive learning of models is necessary using large and representative databases representing
actions and activities. In these models, adding new information is difficult and generally it is necessary to learn again the models. Moreover, it is difficult to interpret the models and therefore, a user can hardly understand actions and activities models since the systems generally appear as “black boxes”. This is a problem for human-computer interaction and database management. This explains also the very little number of work on pattern (action and activities) discovery. Lastly, actions and activities of humans can generally not be separated: in the state-of-the-art, one model is built for each activity and a log-likelihood is computed but, information on actions within activities are not available or not reliable except if models of actions are also built.

These last years, new efficient tools for pattern recognition have been proposed in the framework of belief functions [11, 12, 13] based on Dempster-Shafer’s theory of evidence and on Transferable Belief Model [14, 15, 16, 17].

1.2. Using Transferable Belief Model

As an alternative to Probability Theory for temporal sequence modeling and recognition, we explore the use of Transferable Belief Model (TBM) in particular for human motion analysis and more precisely for action and activity recognition. The TBM, proposed by Smets and Kennes [16], allows to represent the belief of an agent (a sensor, a human, or another source of belief) about the observed system. The TBM is attractive because it is close to human-reasoning by means of logical rules extended to belief functions as well as implication rules [18]. Possibility, probability and TBM are three different models of knowledge representation (uncertainty, imprecision, vagueness, and so on) [19, 20], they are different but not concurrent.
The TBM relies on belief functions [20], a more general measure which allows to encode and combine a variety of knowledge, wider than probability measures. The TBM was initially developed in Artificial Intelligence [15, 16] and applications of the TBM in the context of human motion analysis from video is just in its infancy partially because the TBM is a recent theory compared to Probability Theory.

1.3. Contributions and paper overview

The main contribution of this paper is a tool called Belief Scheduler [21] which allows to manage temporal aspects of belief functions and to recognize states (actions) and sequences of states (activities) on-line in TBM framework. The belief Scheduler is applied on real videos of athletics jumps acquired by a moving camera and it is compared to HMM for sequences classification. The videos are characterized by the variety of environment conditions involving other moving people and occlusions.

The proposed architecture for human action and activities recognition is presented in Section 2. TBM background and fusion are described Section 3. Then, in Section 3.5 are recalled the main parts of the Temporal Evidence Filter [22, 23] which are required to present the Belief Scheduler. Section 4 focuses on Belief Scheduler. In Section 5, we performed several tests focusing on on-line human action and activity recognition in real videos with moving camera. The goal is to recognize actions, running, falling, jumping and standing up, and activities, high jump, pole vault, triple jump and long jump, of one athlete. The results show the system performance in real conditions. We also study a classification task comparing TBM-based Belief Scheduler to probability-based HMM.
2. System overview

2.1. Architecture

Human action and activity recognition requires several operations that can be represented as in the architecture presented in Fig. 1. The proposed architecture is built such as to be generic enough to add new features and new actions. It is based on two levels of processing. The low level provides relevant features concerning actions that are extracted from the video stream. They are generally application dependent and the choice of these features is a recurrent problem: they must give enough information about the goal to reach in term of decision their number must be low in order to make the system efficient in term of time consumption.

The high level part starts from the conversion of the features values into beliefs on symbols describing actions. A semantic is assigned to each action according to the features. The weighted opinions, available at each frame of a video and provided by the features conversion, are combined. The combination is independently performed frame by frame in the Transferable Belief Model framework. The resulting belief may be frame-inconsistent due to contradictory features at a given frame, or temporal-inconsistent i.e. the belief varies abruptly in a given interval of frames. Disturbed belief is not adapted for interpretation, notably in case of monitoring where it is better to have smooth results to be understandable and to take reliable decision. This accounts for the proposition of a Temporal Evidence Filter (TEF) [23]. Then, in order to infer activity, sequences of actions are recognized. A process

\footnote{The Temporal Evidential Filter was previously named Temporal Belief Filter (TBF).}
called Belief Scheduler relying on the TEF allows to infer activity. At each frame, the output of the whole system is a belief on the realization of each action in each activity. A quality criterion is also proposed to assess the confidence of actions and activities recognition.

2.2. Features extraction

In this paper, the choice of features is based on three a priori assumptions:
• The human is tracked by the cameraman. This assumption is well satisfied when the human is the center of interest. For instance, in athletics jumps, athletes are tracked to satisfy telespectators, trainers and sponsors,

• The trajectories of particular human body points (human’s head, center of gravity and end of legs) give information on actions. However, the system is generic enough to add new information concerning other features,

• A single human is moving. The case of multiple humans is more complex and not considered in this paper. Note that in [24] we proposed an algorithm for people counting that was applied to athletics video analysis and a recognition rate of more than 95% was obtained in the discrimination of single vs. multiple athletes.

These assumptions are not strong compared to the ones generally assumed [25] such as fixed view point, calibrated camera and video quality. The proposed features are general and suffices to recognize a lot of actions and activities. More complicated actions and activities probably demand higher level features such as arm and knee points tracking that can be detected using a modification of the proposed scheme. However, higher level tracking requires better quality of videos such in [26] for 18 human points detection under good quality of videos.

Robust shape/motion features are automatically extracted at each frame of the video. Three of them are computed by a camera motion estimator [27] which are the horizontal \( P_{hm} \) and vertical \( P_{vm} \) motions and divergence
The dominant motion image is obtained from the camera motion estimation, where the intensity of a pixel depends on its membership to the dominant motion that is assumed to be the motion of the background. Fig. 2(b) depicts dominant motion for images corresponding to running, jumping, falling and standing up actions in a high jump sequence. The silhouette of the athlete is in black meaning he does not belong to the dominant motion (foreground).

A human detection/tracking algorithm provides human’s head, center of gravity and end of legs position from which the variation of the center of gravity (\(P_{vcg}\)), the angle between horizon and human axis (\(P_{swing}\)) are derived. The feature vector is thus \(O_t = [P_{hm} \ P_{vm} \ P_{div} \ P_{vcg} \ P_{swing}]\). This algorithm can easily be updated using 4 points instead of 3 [28].

<table>
<thead>
<tr>
<th>camera motion (affine motion features)</th>
<th>tracking (coordinates)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(P_{hm}) horizontal motion</td>
<td>(P_{vcg}) variation of center of gravity</td>
</tr>
<tr>
<td>(P_{vm}) vertical motion</td>
<td>(P_{swing}) angle between horizon and human axis</td>
</tr>
<tr>
<td>(P_{div}) divergence</td>
<td></td>
</tr>
</tbody>
</table>

The numerical features provided by the camera motion estimation and the human points detection and tracking are gathered in Table 1. They represent the outputs of the low level part of the proposed architecture (Fig. 1). An example is given for a high jump in figure 11.
3. Models of actions and fusion process

This section presents methods that link numerical features $O_t$ to belief on actions using Transferable Belief Model (TBM) framework which allows to take imprecision, uncertainty, inconsistency and reliability of the features into account. These imperfections may have been caused by lack of texture in images, brightening variations and other moving objects which disturb both camera motion estimation and human detection/tracking algorithm. Features are modeled using two different evidential methods that will be compared in experiments: the model of likelihood (MLGBT) and the model of distance (EDC).
3.1. Formal definition of belief assignment

A belief is expressed by a belief function to explicitly model imprecision as well uncertainty. Formally, the basic belief assignment (BBA) $m_k^\Omega$, provided by information source $k$ and defined on the power set $2^\Omega$ of the frame of discernment (FoD) $\Omega$, is:

$$m_k^\Omega : 2^\Omega \rightarrow [0, 1]$$

such that $\sum_{X \subseteq \Omega} m_k^\Omega(X) = 1$. When $m_k^\Omega(X) > 0$, $X$ is called focal set and a value $m_k^\Omega(X)$ expresses a confidence in proposition $X \subseteq \Omega$ but does not imply any additional claims regarding subsets of $X$ [20] (it is a fundamental difference with Probability Theory) and this property allows to explicitly model doubt between the hypotheses.

In the case of action recognition, the FoD for each action $A_j \in \{\text{running, jumping, falling, standing up}\}$ is binary (either true or false) and denoted as:

$$\Omega_j^t = \{T_j^t, F_j^t\}$$

where time $t$ is explicit since we consider that belief actions evolves along time. The power set $2^{\Omega_j^t} = \{\{T_j^t\}, \{F_j^t\}, \{T_j^t, F_j^t\}, \emptyset\}$ gathers the subsets of the FoD (called propositions). Note that, for sake of simplicity, braces around the propositions are not written. The belief mass on subset $\{T_j^t, F_j^t\}$ can be interpreted as the weight on the logical proposition $T_j^t \cup F_j^t$ meaning that state of action $A_j$ at $t$ is imprecise (either true: $T_j^t$, OR false: $F_j^t$). When $m_k^{\Omega_j^t}(\emptyset) = 0$ and $m_k^{\Omega_j^t}(T_j^t \cup F_j^t) = 0$ then the BBA is said Bayesian since only
singletons are focal sets.

3.2. Problem statement

The goal is to define a belief function on $2^{\Omega_j}$ concerning action $A_j$ related to observed features $O_t$ at time $t$. Obtaining a belief function from features can be stated as a problem of pattern recognition [11], i.e. we need to build a mapping from the feature space $\mathbb{R}^F$ to action space $\Omega_j^t$. This mapping is traditionally called classifier $C$ of type III [29] defined as:

$$C : \mathbb{R}^F \rightarrow \Omega_j^t$$

where $F$ is the dimension of $O_t$. The mapping can be obtained automatically using:

- The model of likelihood (MLGBT$^2$) which consists in applying the Generalized Bayesian Theorem (GBT) [30] on likelihoods conditional to action states [31, 32, 11]. This method needs to estimate some probability densities.

- The model of distance (EDC$^3$) of Denœux et al. [33, 34]). This method is interesting when the models of classes are not known and/or difficult to be obtained.

$^2$Stands for Model of Likelihood based on Generalized Bayesian Theorem.
$^3$Stands for Evidential Distance-based Classifier.
In the sequel, we will also refer to $\Omega^t_s = \{\text{Run, Jmp, Fal, Stu}\}$ (standing for running, jumping, falling, standing up respectively) which is the set of the four actions.

3.3. Model of likelihood (MLGBT)

We first estimate conditional probability densities of $O_t$ given each action $A_j$. The densities can, for example, be modeled by Gaussian mixtures where means and variances are estimated using an Expectation-Maximization algorithm. For each action, a learning set corresponding to 30% of the database is used (with a 3-fold cross-validation). The number of Gaussians is set using the method proposed in [35] based on Minimum Description Length: 10, 4, 4 and 8 components are used for running, jumping, falling and standing up action respectively.

After that, given an unknown feature vector $O_t$ at $t$, a likelihood $P(O_t|A_j)$ is generated for each action $A_j$. Then, as proposed by Smets et al. [30, 31, 32], these likelihoods are supposed to represent plausibilities of observations conditional to states, i.e. $p^{RF}[A_j](O_t)$, defined in the feature space $\mathbb{R}^F$. They are used in the Generalized Bayesian Theorem in order to compute the posterior belief mass $m^{\Omega^t_s}[O_t](S^t)$ as follows [32]:

$$m^{\Omega^t_s}[O_t](S^t) = \prod_{A_j \in S^t} P^{RF}[A_j](O_t) \cdot \prod_{A_j \notin S^t} \left(1 - P^{RF}[A_j](O_t)\right)$$  \hspace{1cm} (4)

where $m^{\Omega^t_s}$ is a BBA defined on the set of actions $\Omega^t_s = \{\text{Run, Jmp, Fal, Stu}\}$. The BBA $m^{\Omega^t_s}$ on each action can be derived from $m^{\Omega^t_s}[O_t]$ using a coarsening
process that provides:

\[
\begin{align*}
    m^\Omega_s[O_i](T^s_j) & \leftarrow m^\Omega_s[O_i](A^s_j) \\
    m^\Omega_s[O_i](T^s_j \cup F^s_j) & \leftarrow \sum_{A^s_j \subseteq B_j} m^\Omega_s[O_i](B_j) \\
    m^\Omega_s[O_i](\emptyset') & \leftarrow m^\Omega_s[O_i](\emptyset')
\end{align*}
\]  

(5)

where \(A^s_j \in \Omega^s_s\) (a singleton) and \(B_j\) is an element of the power set of \(\Omega^s_s\). Figure 7(a) depicts some results (output of the modeling process).

3.4. The model of distance (EDC)

A learning set with \(N\) instances is available as \(L = \{O_n, m^\Omega_n\} \) where \(n \in \{1, 2, \ldots N\}\) is a sample index\(^4\). Each sample \(e_n\) is composed of observations \(O_n\) labeled by a belief function \(m^\Omega_n\) defined on the set of actions \(\Omega_s = \{\text{Run, Jump, Fall, Stay}\}\). When the class of \(e_n\) is known then the belief function is categorical \(m^\Omega_n(A_j) = 1, A_j \in \Omega_s\) whereas when the class is unknown then \(m^\Omega_n(\Omega_s) = 1\). In this application, the labeling using belief function is interesting in order to model the transitions between actions which are intrinsically imprecise, for example \(m^\Omega_n(\{\text{Run, Jump}\}) = 1\) in a transition between running and jumping.

For a given observed feature vector \(O_t\), we need to assess the BBA \(m^\Omega_t[O_t]\) that reflects the type of action. Using the distance model of Denœux [33], the BBA is given by the conjunctive combination of the \(K\) nearest neighborhoods’ BBA (neighbors of \(O_t\)) determined by the Euclidean distance. For that, let \(L_k = \{O_k, m^\Omega_k\} \in L\) the set of \(K\) nearest neighborhoods. The BBA \(m^\Omega_k[O_k]\)

\(^4\)We do not use \(t\) here but \(n\) since time is not important for the modeling process. Time will be explicitly taken into account during sequence recognition.
for sample $e_k$ in $L_k$ is then obtained by:

$$
\begin{align*}
    m^\Omega_{k,s}[O_k](\{A_j\}) &= \zeta \cdot \phi_q(D(O_k, O_t)) \\
    m^\Omega_{k,s}[O_k](\Omega_s) &= 1 - \zeta \cdot \phi_q(D(O_k, O_t)) \\
    m^\Omega_{k,s}[O_k](B) &= 0, \quad B \in 2^{\Omega_s}\setminus\{\Omega_s, \{A_j\}\}
\end{align*}
$$

(6)

where $\phi_q(D(O_k, O_t)) = \exp(-\gamma_q \cdot D(O_k, O_t))$, the function $D(O_k, O_t)$ is the Euclidean distance between $O_k$ and $O_t$, $\gamma_q \geq 0$ and $\zeta$ is such that $0 < \zeta < 1$.

The $K$ BBA for the $K$ nearest neighborhoods are then conjunctively combined in order to estimate $m^\Omega_{t,s}[O_t]$:

$$
    m^\Omega_{t,s}[O_t] = m^\Omega_{1,s}[O_1] \oplus \cdots \oplus m^\Omega_{k,s}[O_k] \oplus \cdots \oplus m^\Omega_{K,s}[O_K]
$$

(7)

where $\oplus$ is Dempster’s conjunctive rule of combination which is computed in two steps (given for example the first two neighborhoods). First compute Smets’ conjunctive rule of combination (CRC) by:

$$
    m^\Omega_s'[O_t] = m^\Omega_s'[O_1] \odot \cdots \odot m^\Omega_s'[O_k] \odot \cdots \odot m^\Omega_s'[O_K]
$$

(8)

where

$$
    (m^\Omega_1 \odot m^\Omega_2)(D) = \sum_{B \cap C = D} m^\Omega_1(B) \cdot m^\Omega_2(C)
$$

(9)

and then normalize the CRC:

$$
\begin{align*}
    m^\Omega_s(B) &= \frac{m^\Omega_s(B)}{1 - m^\Omega_s(\emptyset)}, \quad \forall B \subseteq \Omega_s, B \neq \emptyset \\
    m^\Omega_s(\emptyset) &= 0
\end{align*}
$$

(10)
We have set $K = 5$ and $\zeta = 0.99$ using heuristics attached to the application through several tests. The value of $\gamma_q$ is optimized using a gradient-based method proposed in [33]$^5$. The learning set represents 30% of the whole dataset (as for the MLGBT model).

As previously, the BBA $m^{\Omega_t}$ is defined on the set of actions (running, jumping, falling, standing up) and the BBA $m^{\Omega_l}$ on each action can be derived from $m^{\Omega_l}[O_t]$ using the coarsening process described in Eq. 5. Figure 7(b) depicts some results (output of the modeling process).

3.5. Temporal Evidence Filter for action state filtering

The Temporal Evidence Filter (TEF) proposed in [22, 23] makes belief on actions temporally consistent (the resulting belief has no conflict and made smooth) and separates action states (assumed to be true or false). The TEF works on-line on each action independently taking as input the BBA obtained from features fusion and the previous TEF output. The system is described in Figure 3. In this section, the 8 main points of the TEF process are recalled [23].

The TEF uses a model of belief evolution $M \in \{T, F\}$, one for each state ($T$ for $T^t_j$ and $F$ for $F^t_j$). Only one model is applied at each time $t$ and each model assumes that the BBA of the current TEF output $m^{\Omega_t}$ at frame $t$ is close to the previous one $m^{\Omega_{t-1}}$ (this is a common hypothesis in filtering, in particular for our application since human motions are continuous). A model of evolution can be viewed as an equivalent to conditional probability tables but in the TBM context.

1-Prediction: The model of evolution is used to predict the current state of each action $\hat{m}_{M}^{T}$ (at time $t$) by combining the BBA of the current model of belief evolution and the previous TEF output $m_{t-1}^{T}$ resulting in two possible BBA [23]: either $\hat{m}_{T}^{O_{t}}$ if the current model is $T$ or $\hat{m}_{F}^{O_{t}}$ if the current model is $F$. These BBAs are given by:

$$\begin{align*}
\hat{m}_{T}^{O_{t}}(T_{t}) & = \gamma_{T} \cdot m_{t-1}^{O_{t}}(T_{t-1}) \\
\hat{m}_{T}^{O_{t}}(O_{t}) & = \gamma_{T} \cdot m_{t-1}^{O_{t}}(O_{t-1}) + 1 - \gamma_{T} \\
\hat{m}_{F}^{O_{t}}(F_{t}) & = \gamma_{F} \cdot m_{t-1}^{O_{t}}(F_{t-1}) \\
\hat{m}_{F}^{O_{t}}(O_{t}) & = \gamma_{F} \cdot m_{t-1}^{O_{t}}(O_{t-1}) + 1 - \gamma_{F}
\end{align*}$$

A method has been proposed in [36] in order to estimate $\gamma_{M}$ that we can not describe in this paper due to limited space. In this paper we have always

---

6 The automatic method used to compute $\gamma_{T}$ (resp. $\gamma_{F}$) is supervised and consists in computing the mean of belief mass on hypothesis $T_{t}$ (resp. $F_{t}$) on the cartesian product $\Omega_{t} \times \Omega_{t-1}$ on intervals of frames where the state is true (resp. false).
set these parameters to 0.9.

2-Fusion of prediction and measure: $m^{\Omega_j}_t = \hat{m}^{\Omega_j}_t \circ \check{m}^{\Omega_j}_t$ combines the available information, where the operator $\circ$ is the conjunctive rule of combination defined in equation 9.

3-Conflict: $\epsilon_t = m^{\Omega_j}(\emptyset)$ quantifies the contradiction between model of belief evolution and data. The higher the conflict, the higher the necessity to change the current model. We thus introduce the concept of unlikelihood in order to give a semantic to the conflict value.

4-Cusum: $CS(t) = \lambda \times CS(t - 1) + \epsilon_t$ builds the cumulative sum of conflict along time, and $\lambda \in [0, 1]$ is a fader coefficient to cope with low/high variation of conflict (smoothing).

5-Decision on model change: when the cumulative sum is too high, i.e. if $CS(t) > T_s$ (stop threshold) at frame $t_s$, the model is changed. The new model is applied from $t_s$ and belief on interval of frames $[t_s - W, t_s]$ is compelled to be vacuous (i.e. $m^{\Omega_j}(\Omega_t^f) = 1$) to emphasize action states transition ($W = 3$ is a window size representing transition size). The threshold $T_s$ can be estimated using a method proposed in [36] that we can not describe in this paper due to limited space.

6-TEF output: if current conflict $\epsilon_t$ is low, then the output is the fusion result of prediction and observations. It conflict is too high, then we keep the prediction (cautious approach). Formally: $m^{\Omega_j}_t = \hat{m}^{\Omega_j}_t \circ \check{m}^{\Omega_j}_t$ if $\epsilon_t \leq \delta_\emptyset$ and $\check{m}^{\Omega_j}_t$ otherwise where $\delta_\emptyset$ is a threshold reflecting a tolerance to the conflict.

---

7The automatic method used to compute $T_s$ is supervised and consists in applying the TEF with a constrained false model on a reference belief function that evolves with time and with $T_s \approx \infty$ (i.e. unreachable). Then, given the ground truth which provides the start frame of the true state, the stop threshold is chosen close to $T_s \approx CS(t)$. 

17
adaptively computed using the mean of conflict over a window (size $N = 5$) of a few frames: \( \delta_\emptyset = 1/N \cdot \sum_{t_i=(t-N-1)}^t \epsilon_{t_i} \).

In order to remain coherent with the model of evolution that is used, we modify the belief mass as follows: if the model used is $T$ then the belief on the emptyset ($m^{\Omega_t}(\emptyset_t)$) and the belief on $F^j_t$ ($m^{\Omega_t}(F^j_t)$) are transfered onto $T^t_j$ and $\Omega^t_j$ respectively:

**Redistribution when the model is “$T$: the state is true”**

\[
\begin{align*}
m^{\Omega_t}(T^1_t) & \leftarrow m^{\Omega_t}(T^1_t) + m^{\Omega_t}(\emptyset_t) \\
m^{\Omega_t}(\Omega^t_t) & \leftarrow m^{\Omega_t}(\Omega^t_t) + m^{\Omega_t}(T^1_t) \\
m^{\Omega_t}(\emptyset_t) & = m^{\Omega_t}(F^1_t) = 0
\end{align*}
\]

**Redistribution when the model is “$F$: the state is false”**

\[
\begin{align*}
m^{\Omega_t}(F^j_t) & \leftarrow m^{\Omega_t}(F^j_t) + m^{\Omega_t}(\emptyset_t) \\
m^{\Omega_t}(\Omega^t_t) & \leftarrow m^{\Omega_t}(\Omega^t_t) + m^{\Omega_t}(T^1_t) \\
m^{\Omega_t}(\emptyset_t) & = m^{\Omega_t}(T^1_t) = 0
\end{align*}
\]

**7-Local Quality criterion:** It reflects how we can be confident in an action. This criterion is said “local” because concerns only one action within a sequence. Given a model of evolution ($\mathcal{M}$), we compute the quantity:

\[
\begin{align*}
LQ^{t,2}_i[A_j][\mathcal{M}](T^j_t) & = \left(1 - \frac{1}{t - t_s}\right) \times \\
LQ^{t,2(t-1)}_i[A_j][\mathcal{M}](T^j_t) & + \frac{m^{\Omega^t_j}_t(T^j_t)}{t - t_s} \cdot (1 - \epsilon_t)
\end{align*}
\]

for each action $A_j$ within each activity $S_i$. This criterion is computed on-line and embeds past events and innovation. It uses conflict to weigh the current
belief on $T_j$.

**8-Transition and false alarm detection:** when the stop threshold is reached for an action $A_i$ in sequence $S_j$, we compare its Local Quality criterion $LQ_i^{t_j}([\mathcal{M}](T_j^t))$ with a threshold $\delta_{FA}$ which is the minimal quality value required to validate a model change. If the criterion is lower, we declare the model change. When a false alarm occurs with the model $\mathcal{T}$ on a given interval of frames, then the TEF is run again on this interval but the model is compelled to be false $\mathcal{F}$ (it does not take into account the stop CUSUM threshold on this interval).

4. **Belief Scheduler**

Describing an activity by means of state machines is a natural and human-like way of thinking. In this section, a method called *Belief Scheduler* [21] is proposed for activity recognition based on TBM. It considers an activity $S_n$ as a sequence of $K$ understandable actions $A_k$. This method is a state machine which exploits the results of the *Temporal Evidence Filter* to synchronize actions.

It is considered that only one action can be true at the same frame, the others $K-1$ actions are false. By using the *Temporal Evidence Filter*, it means that, at a frame $f$, 1 action uses the model $\mathcal{T}$ whereas $K-1$ actions use the model $\mathcal{F}$. The models are considered as a kind of resources to which actions, a kind of *threads*\(^8\), attempt to access. To access a model, either $\mathcal{F}$ or $\mathcal{T}$, an action has to ask for it and the *Belief Scheduler* manages this access.

\(^8\)In [37], the concept of *thread* for human motion analysis has been introduced but our interpretation is here different.
Ideally, actions are synchronized (in this case, a simple state machine can be used) but, in real cases they can be either overlapped or unconnected such as represented in Figure 4. The *Belief Scheduler* allows to overcome these problems.

![Diagram](image)

Figure 4: Due to data imperfection, overlapping (a) and unconnection (b) generally appear between actions.

4.1. Description

The *Belief Scheduler* relies on the TEF with a little change in notations: the CUSUM of an action $A_k$ at each frame $t$ is denoted $CS^k(t)$ and the stop threshold is $T_s^k$. Moreover, in the sequel, we call natural state the belief provided by the fusion process without filtering nor scheduling. We call constrained state the belief provided by the scheduling process. The state of an activity is said constrained because of temporal constraints included in the sequence (one action follows another action).
4.1.1. PREEMPTION process

This process manages overlapped actions as presented in Figure 4.a. At \( t = f_P \), \( A_k \) is still true while \( A_{k+1} \) becomes true thus, two actions are true at the same time: it is said that \( A_{k+1} \) wants to preempt \( A_k \). This process occurs at frame \( t = f_P \) when the cusum \( CS^{k+1}(t) \) of the next action \( A_{k+1} \) is greater than its stop threshold \( T^{k+1}_s \):

\[
\begin{align*}
\text{if } CS^{k+1}(t) &> T^{k+1}_s \text{ and } CS^k(t) < T^k_s \\
\text{then PREEMPTION and } f_P &= t
\end{align*}
\]

As depicted in this case, the constrained as the natural state of \( A_{k+1} \) is temporarily true (true state) from frame \( f_P \) and the constrained state of \( A_k \) is temporarily false (false state) until validation (see Figure 4). The validation is enabled when the quality of the action \( A_{k+1} \) recognition (which asks for PREEMPTION) is satisfying (Section 4.1.3 focuses on this process). Information at \( t = f_P \) concerning actions (cusum, belief . . . .), i.e. the context, is stored. This allows to restore the context in case the PREEMPTION would not be enabled.

Belief Scheduler initialization: At the beginning of scheduling, all actions are in the false state. An artificial initial true state action is added in the sequence (first state) that allows the Belief Scheduler to wait for a PREEMPTION of the first action.

4.1.2. FORCING process

This process manages unconnected actions as presented in Figure 4.b. At \( t = f_F \), \( A_k \) is false as well as \( A_{k+1} \). This process occurs at frame \( f_F \) when the cusum \( CS^k(t) \) of the current action \( A_k \) is greater than its stop threshold
$T^k_s$:

\[
\text{if } CS^k(t) > T^k_s \text{ and } CS^{k+1}(t) < T^{k+1}_s \\
\text{then FORCING and } f_F = f
\] (17)

If the two successive actions are a little unconnected, i.e. with a small gap less than a fixed $\Delta_F$, constrained state of $A_k$ is forced to the true state until $A_{k+1}$ becomes true. However, sometimes, the gap between successive actions can be large, i.e. with a size greater than the value $\Delta_F$. The action requiring a FORCING, e.g. constrained state of $A_k$, keeps on being true until the frame “$f_F + \Delta_F$”. At this time, constrained state of $A_{k+1}$ is forced to be true and constrained state of $A_k$ becomes false (see Figure 4).

4.1.3. False alarm detection

If actions $A_{k+1}$ and $A_{k+2}$ are too much unconnected and if $A_{k+1}$ had previously preempted $A_k$, then $A_{k+1}$ can be interpreted as a false alarm (see Figure 5). It appears when an action becomes true instead of staying false. This false alarm procedure is applied to validate a PREEMPTION.

In order to decide whether action $A_{k+1}$ is or not a false alarm, it is required to assess the recognition performance of this action. The chosen criterion is the Local Quality recognition performance $LQ_{i}^{f_{P}:f_{F}+\Delta_{F}}[T](T_{j}^{i})$ defined in equation (15) and computed on interval of frames $[f_P, f_F + \Delta_F]$ (the bounds use the frames of PREEMPTION and of FORCING). The computation of $LQ_{i}^{f_{P}:f_{F}+\Delta_{F}}[T](T_{j}^{i})$ embeds (1) an artificial conflict on $[f_F, f_F + \Delta_F]$ because the system forces the action to remain in a true state although this action is actually false and (2) a natural conflict on $[f_P, f_F]$ coming from the
data. The following rule is therefore applied:

\[
\text{if } LQ_i^{f_P,f_F+\Delta_F}[T](T_j^t) < \delta_{FA} \text{ then } A \text{ is a FALSE ALARM}
\]

where \( \delta_{FA} \) is a crisp threshold corresponding to a severity degree on the quality. When a false alarm is declared, the context of actions at frame \( f_P \), such as the value of the CUSUM, is restored and the previous action (before PREEMPTION), e.g. \( A_k \), becomes true again. If \( LQ_i^{f_P,f_F+\Delta_F}[T](T_j^t) > \delta_{FA} \) then the quality is sufficient and therefore \( A_{k+2} \) becomes true and \( A_{k+1} \) and \( A_{k+2} \) are both validated.

*Queuing* - When several actions perform consecutive PREEMPTION, a validation must be performed to ensure that they are not false alarms. They are stored in a FIFO queue to wait for their validation. The number of actions in the queue is limited, e.g. two actions, so when the queue is full then the
oldest queued action is validated.

4.2. Activity inference

The problem is to determine which activity (sequence of actions) is the best one at a given sequence. It is a problem of evaluation which is relevant when trying to assign a score to each potential activity, as in case of competing systems [10]. For example, in Hidden Markov Models, inference is performed using the forward-backward algorithm which provides a log-likelihood for each activity. In this paper, we propose a criterion for on-line inference within the Belief Scheduler and computed from the Local Quality recognition performance criterion. For that, each $LQ^{f_s:f}(T)(T^t_j)$ (only for model true and state true), for all actions $A_j$ in a particular activity $S_i$ (composed of $K_i$ actions) is aggregated into a Global Quality recognition performance criterion $GQ^t_i$ to represent the confidence in activity $S_i$ from frame $f_s$ (a given start frame) to $t$ (the current frame). The aggregation is simply the arithmetic mean:

$$GQ^t_i = \frac{1}{K_i} \sum_{n \in \{1..K_i\}} LQ^{f_s:f}(T)(T^t_j)$$ \hspace{1cm} (18)

In order to find the best activity $S^t_i$ at the current frame $t$, we maximize $GQ^t_i$ over all possible sequences. Then, a threshold is applied to decide whether the recognition is satisfying. Formally:

$$S^t_i = \text{argmax} \ i, GQ^t_i > \theta$$ \hspace{1cm} (19)
where \( \theta \) is a degree of severity on activity recognition quality which can be used for a class of rejections (if all activities are not well recognized). Its value can be the same as the false alarm threshold \( \delta_{FA} \).

5. Experiments

The system is tested for action and activity recognition in athletics jumps. The database\(^9\) is composed of 69 videos acquired with a moving camera and several unknown view angles. There are 26 pole vaults, 15 high jumps, 12 triple jumps and 16 long jumps equivalent to about 12620 frames (see table 2 for details). The database is characterized by its heterogeneity (Figure 6) with a panel of view angles as well as environments and athletes (out/indoor, male, female, other moving people). Codes are written in C/C++ and Mat-lab and, using a PC with a 2.8GHz Xeon processor and 1 GB RAM, tracking and motion estimation parts evolve at about 10 frames/s, the fusion, the Temporal Evidence Filter and the Belief Scheduler parts at real time.

5.1. Settings

The proposed system is used to recognize actions, running, falling, jumping and standing up, and activities (actions sequence) high jump, pole vault, triple jump and long jump. The three first actions are described by a four states belief scheduler (running \(\rightarrow\) jumping \(\rightarrow\) falling \(\rightarrow\) standing up) while

\(^9\)Some videos and results are available on authors website: \url{www.lis.inpg.fr/pages_perso/ramasso/index.htm} (Demos link) and \url{www.csd.uoc.gr/~cpanag/DEMOS/actionActivityRecognition.htm} but videos used can not be distributed because of copyrights. Codes of TBM are based on TBMlab toolbox of Smets available at \url{http://iridia.ulb.ac.be/~psmets}.
Figure 6: Heterogeneous database used for testing.

Triple jumps are described by an eight states scheduler (running → jumping → falling → jumping → falling → jumping → falling → standing up).

The parameters of the Temporal Evidence Filter and the Belief Scheduler have been tuned using 5-fold cross validations: 1) we have selected 80% of the database (distinct of the database used for EDC and MLGBT models estimation), 2) made estimation of the parameters such as to maximize

<table>
<thead>
<tr>
<th>Jumps/Actions</th>
<th>( N_v )</th>
<th>running</th>
<th>jumping</th>
<th>falling</th>
<th>standing up</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>high jumps</td>
<td>15</td>
<td>1000</td>
<td>500</td>
<td>400</td>
<td>200</td>
<td>2100</td>
</tr>
<tr>
<td>long jumps</td>
<td>16</td>
<td>1400</td>
<td>400</td>
<td>550</td>
<td>470</td>
<td>2820</td>
</tr>
<tr>
<td>pole vaults</td>
<td>26</td>
<td>2000</td>
<td>1400</td>
<td>1000</td>
<td>700</td>
<td>5100</td>
</tr>
<tr>
<td>triple jumps</td>
<td>12</td>
<td>1200</td>
<td>400</td>
<td>600</td>
<td>400</td>
<td>2600</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>69</strong></td>
<td><strong>5600</strong></td>
<td><strong>2700</strong></td>
<td><strong>2550</strong></td>
<td><strong>1770</strong></td>
<td><strong>12620</strong></td>
</tr>
</tbody>
</table>

Table 2: Description of the database where \( N_v \) corresponds to the number of videos.
recognition performance and 3) tested on the remaining 20%. We have done it 5 times and computed the average of the criteria used in evaluation. The best set of parameters obtained are given in table 3.

5.2. Tests and evaluation protocol

The goal is to assess 1) the modeling using MLGBT and EDC methods (before scheduling) and 2) the performance of the belief scheduler (after filtering by the TEF and scheduling). In the sequel we will refer to the scheduling process by the acronym BSS (standing for Belief State Scheduler) for simplicity.

For quantitative evaluation, an action is said true if its pignistic probability (BetP) [38] defined by BetP(T_j) = \frac{1}{1-m(\emptyset)}(m(T_A) + \frac{m(T_A \cup F_A)}{2}) is greater than 0.5 (since an action can be true or false), where m is the belief mass provided by the output of the modeling process or by the scheduler. We then compare these decisions with the ground truth (the database was manually annotated). Recall (R) and precision (P) criteria are used [39]. They are computed as $R = \frac{C \cap R}{C}$ and $P = \frac{C \cap R}{R}$, where $C$ is the set of correct frames obtained by expert annotations, $R$ is the set of retrieved frames provided by the recognition module using the BetP-based criterion, and $C \cap R$ is the number of correctly retrieved frames. In order to assess the method by only one criterion, the $F_1$-measure defined as $F_1 = \frac{2 \times R \times P}{R + P}$ combines $R$ and $P$ (here, the same importance is given to both components). This measure tends to the minimum between $R$ and $P$.

Figures 7-8-9-10 are examples of, respectively, action detection, scheduling, and local and global qualities evolution for a high jump video using modeling based on MLGBT (top figures) and EDC (bottom figures). These
Table 3: TEF filter and scheduler parameters settings.
figures are commented below.

5.3. Illustration of the belief scheduler

Beliefs for each action are provided by the model of distance (EDC) [33]. An example of belief is depicted in figure 7 (before scheduling) and in figure 8 (after scheduling). Using only 30% of the learning set yields little complexity, uses a few memory but generates noise beliefs. The scheduler and the filter allow to smooth them and ensure good recognition performance ($GQ = 74\%$). Figure 11 provides the features measured on the video sequence and from which beliefs are computed.

In order to analyze the scheduler behavior, let consider two consecutive actions, e.g. running and jumping, that correspond to the two first lines of figures 7 (input) and 8 (output). We consider the case of EDC-modeling (for MLGBT the same reasoning can be applied).

The scheduler starts by filtering belief on running using model $\mathcal{T}$ (natural true state) and use the model $\mathcal{F}$ for all the other three actions (natural or constrained false state). Then at time $t \approx 100$, running becomes false and forces jumping action to be true. The natural state of running is false and the filter on running uses naturally the model $\mathcal{F}$ while jumping action is constrained to be true and the filter on this action uses the model $\mathcal{T}$. At time $t \approx 130$, falling action makes a preemption on jumping. Then at $t \approx 155$, standing up makes a preemption on falling and since the quality of falling is sufficient ($GQ \approx 0.95$, third figure on the left of fig. 9 where GQ stands for Global Quality recognition performance), standing-up is allowed to use the model $\mathcal{T}$ (natural true state) while the others use model $\mathcal{F}$. Finally at $t \approx 184$, the sequence ends and the global quality reaches $\approx 75\%$. 
Figure 7: Beliefs obtained by the model of likelihood (MLGBT) and the model of distance (EDC) from features observed on the current video (figure 11).
Figure 8: Beliefs of figure 7 after filtering and scheduling.
Figure 9: Local quality criterion for each action after filtering and scheduling.
Figure 10: Global quality criterion during scheduling, computed from the local ones (fig. 10) at each time.
Sometimes activity recognition fails due to the following main reasons:

- Bad camera view angle. When the athlete motion is perpendicular to the image plane (e.g., front view), the recognition of jumping and
falling action is difficult since the angle ($P_{swing}$) remains stable (does not mean anything) in this case due to perspective projection.

- Bad camera view position. When the camera is very far from the athlete, the computed silhouette can be perceived as noise.

- Bad quality of estimated silhouettes. When the quality of the video is too bad or when the moving athlete is too small, the camera motion estimator fails leading to bad moving objects localization.

In the sequel, we present action detection performance using:

a1) MLGBT modeling alone,

a2) MLGBT modeling coupled with BSS,

b1) EDC modeling alone,

b2) EDC modeling coupled with BSS.

These tests are used as follows:

1. Tests a1) and a2) allow to compare MLGBT with and without BSS (Section 5.4),

2. Tests b1) and b2) allow to compare EDC with and without BSS (Section 5.5),

3. Tests a2) and b2) allow to quantify BSS performance with two different modelings (Section 5.6).

Three sets of tables are presented:

1. One set of four-by-three tables: each four row concerns one type of jump and the three tables represent respectively EDC performance,
EDC+BSS performance and difference between both. So, there is one set of three tables for each jump and each table presents action detection performance (Section 5.4, tables 4, 5, 6 and 7),

2. One set of four-by-three tables as before but concerning MLGBT (Section 5.5, tables 8, 9, 10 and 11),

3. One set of four tables comparing MLGBT+BSS and EDC+BSS performances, with one table for each jump (Section 5.6, table 12).

Performance is assessed using recall (first column named R), precision (second column named P) and $F_1$ measure (third column named $F_1$). The reader may refer to the latter one ($F_1$) in each table for quick performance assessment.

5.4. Results of scheduling with MLGBT modeling

Tables 4, 5, 6 and 7 present recall, precision and $F_1$-measure of the detection of each action (running, jumping, falling and standing up) in each type of jump (high jump, pole vaut, long jump and triple jump) before (tables (a)) and after (tables (b)) the processing made by the belief scheduler (BSS) using MLGBT-based modeling. These tables demonstrate the contribution of the scheduler. The differences before and after applying the BSS are explicitly given in tables (c). When a difference is positive, that means the belief scheduler improves the criterion. This column shows that the BSS greatly improves performance in all jumps.

The results particularly illustrate that the BSS generally increases the recall rate (R) through filtering because of the stop threshold $T_s$ which allows to fill “gaps”. However, a counterpart is that precision (P) sometimes decreases. This is due to the same threshold $T_s$, because a high value makes
the filter wait before model change and thus the filter remains in the previous state a bit too much decreasing precision. Since recall improvement is quite high, the $F_1$ measure is globally improved.
Table 5: Recall ($R$), precision ($P$) and $F_1$-measure for four actions in pole vaults with (a) MLGBT without scheduler and (b) MLGBT with scheduler. Table (c) is the difference between MLGBT with and without scheduler.

<table>
<thead>
<tr>
<th></th>
<th>$R$</th>
<th>$P$</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>0.5968</td>
<td>0.9211</td>
<td>0.7243</td>
</tr>
<tr>
<td>Jumping</td>
<td>0.4336</td>
<td>0.8756</td>
<td>0.5800</td>
</tr>
<tr>
<td>Falling</td>
<td>0.3636</td>
<td>0.8999</td>
<td>0.5179</td>
</tr>
<tr>
<td>Standing-up</td>
<td>0.2321</td>
<td>0.9043</td>
<td>0.3693</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$R$</th>
<th>$P$</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>+0.2684</td>
<td>-0.0730</td>
<td>+0.1322</td>
</tr>
<tr>
<td>Jumping</td>
<td>+0.0935</td>
<td>+0.1046</td>
<td>+0.1056</td>
</tr>
<tr>
<td>Falling</td>
<td>+0.1065</td>
<td>-0.0677</td>
<td>+0.0828</td>
</tr>
<tr>
<td>Standing-up</td>
<td>+0.1131</td>
<td>-0.0182</td>
<td>+0.1275</td>
</tr>
</tbody>
</table>

Table 6: Recall ($R$), precision ($P$) and $F_1$-measure for four actions in long jumps with (a) MLGBT without scheduler and (b) MLGBT with scheduler. Table (c) is the difference between MLGBT with and without scheduler.

<table>
<thead>
<tr>
<th></th>
<th>$R$</th>
<th>$P$</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>0.5490</td>
<td>0.8851</td>
<td>0.6777</td>
</tr>
<tr>
<td>Jumping</td>
<td>0.1556</td>
<td>0.9619</td>
<td>0.2678</td>
</tr>
<tr>
<td>Falling</td>
<td>0.2214</td>
<td>0.9775</td>
<td>0.3610</td>
</tr>
<tr>
<td>Standing-up</td>
<td>0.2421</td>
<td>0.9623</td>
<td>0.3868</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$R$</th>
<th>$P$</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>+0.1471</td>
<td>-0.0456</td>
<td>+0.0834</td>
</tr>
<tr>
<td>Jumping</td>
<td>+0.0733</td>
<td>-0.1000</td>
<td>+0.0938</td>
</tr>
<tr>
<td>Falling</td>
<td>+0.1161</td>
<td>-0.1091</td>
<td>+0.1251</td>
</tr>
<tr>
<td>Standing-up</td>
<td>+0.1288</td>
<td>-0.0230</td>
<td>+0.1450</td>
</tr>
</tbody>
</table>
### Table 7: Recall (R), precision (P) and F<sub>1</sub>-measure for four actions in triple jumps with (a) MLGBT without scheduler and (b) MLGBT with scheduler. Table (c) is the difference between MLGBT with and without scheduler.

<table>
<thead>
<tr>
<th>Action</th>
<th>R</th>
<th>P</th>
<th>F&lt;sub&gt;1&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>0.3917</td>
<td>0.9165</td>
<td>0.5488</td>
</tr>
<tr>
<td>Jumping</td>
<td>0.3212</td>
<td>0.8476</td>
<td>0.4658</td>
</tr>
<tr>
<td>Falling</td>
<td>0.3569</td>
<td>0.8339</td>
<td>0.4956</td>
</tr>
<tr>
<td>Standing-up</td>
<td>0.2058</td>
<td>0.9350</td>
<td>0.3373</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Action</th>
<th>R</th>
<th>P</th>
<th>F&lt;sub&gt;1&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>0.5122</td>
<td>0.7272</td>
<td>0.6010</td>
</tr>
<tr>
<td>Jumping</td>
<td>0.3490</td>
<td>0.7694</td>
<td>0.4801</td>
</tr>
<tr>
<td>Falling</td>
<td>0.3945</td>
<td>0.7486</td>
<td>0.5167</td>
</tr>
<tr>
<td>Standing-up</td>
<td>0.3404</td>
<td>0.8576</td>
<td>0.4873</td>
</tr>
</tbody>
</table>

### Differences (c) MLGBT − MLGBT+BSS

<table>
<thead>
<tr>
<th>Action</th>
<th>R</th>
<th>P</th>
<th>F&lt;sub&gt;1&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>+0.1204</td>
<td>−0.1893</td>
<td>+0.0522</td>
</tr>
<tr>
<td>Jumping</td>
<td>+0.0278</td>
<td>−0.1582</td>
<td>+0.0143</td>
</tr>
<tr>
<td>Falling</td>
<td>+0.0177</td>
<td>−0.1953</td>
<td>+0.0210</td>
</tr>
<tr>
<td>Standing-up</td>
<td>+0.1346</td>
<td>−0.0774</td>
<td>+0.1500</td>
</tr>
</tbody>
</table>
5.5. Results of scheduling with EDC modeling

The same study as previously has been done using EDC modeling. Tables 8, 9, 10 and 11 present recall, precision and $F_1$-measure of the detection of each action in each type of jump before (tables (a)) and after (tables (b)) the processing by the belief scheduler (BSS) with EDC modeling.

The differences of performance of EDC and EDC+BSS are explicitly given in tables (c). This latter table shows that the BSS highly improves the results of action detection based on EDC (with the same comment concerning recall improvement). Improvements seem to be a bit less important than with MLGBT modeling, but this is due to a globally better performance of EDC modeling compared to MLGBT modeling. Indeed, the detection performance of EDC+BSS compared to MLGBT+BSS is in favor of the former except for some running actions. Running is better detected with MLGBT because this action is much more represented in the learning set since it generally takes about 50% of each jump as shown in table 2.

5.6. Comparison between EDC and MLGBT modeling with scheduling

Tables 12 present the differences of performance (in terms of recall, precision and $F_1$-measure) of the detection of each action (action names are not recalled for better readability) in each type of jump after scheduling (with BSS) between both EDC modeling and MLGBT modeling. When the difference is positive, EDC+BSS detection is better than MLGBT+BSS.

Better results are obtained with EDC than with MLGBT (since differences are mainly positive). Therefore, better results are obtained with a direct computation of belief functions (EDC) compared to the indirect com-
### Table 8: Recall ($R$), precision ($P$) and $F_1$-measure for four actions in high jumps with (a) EDC without scheduler and (b) EDC with scheduler. Table (c) is the difference between EDC with and without scheduler.

<table>
<thead>
<tr>
<th>Action</th>
<th>EDC</th>
<th>EDC + BSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>$R$: 0.7207</td>
<td>$R$: 0.7277</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.8653</td>
<td>$P$: 0.9760</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.7864</td>
<td>$F_1$: 0.8338</td>
</tr>
<tr>
<td>Jumping</td>
<td>$R$: 0.4857</td>
<td>$R$: 0.5706</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.7795</td>
<td>$P$: 0.8297</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.5985</td>
<td>$F_1$: 0.6762</td>
</tr>
<tr>
<td>Falling</td>
<td>$R$: 0.4877</td>
<td>$R$: 0.6218</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.7911</td>
<td>$P$: 0.7911</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.6034</td>
<td>$F_1$: 0.6963</td>
</tr>
<tr>
<td>Standing-up</td>
<td>$R$: 0.2606</td>
<td>$R$: 0.4294</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.7110</td>
<td>$P$: 0.8613</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.3814</td>
<td>$F_1$: 0.5731</td>
</tr>
</tbody>
</table>

### Table 9: Recall ($R$), precision ($P$) and $F_1$-measure for four actions in pole vaults with (a) EDC without scheduler and (b) EDC with scheduler. Table (c) is the difference between EDC with and without scheduler.

<table>
<thead>
<tr>
<th>Action</th>
<th>EDC</th>
<th>EDC + BSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>$R$: 0.5886</td>
<td>$R$: 0.6580</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.9112</td>
<td>$P$: 0.9136</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.7152</td>
<td>$F_1$: 0.7650</td>
</tr>
<tr>
<td>Jumping</td>
<td>$R$: 0.4699</td>
<td>$R$: 0.5824</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.8211</td>
<td>$P$: 0.9264</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.5977</td>
<td>$F_1$: 0.7152</td>
</tr>
<tr>
<td>Falling</td>
<td>$R$: 0.4337</td>
<td>$R$: 0.4519</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.8868</td>
<td>$P$: 0.9636</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.5825</td>
<td>$F_1$: 0.6152</td>
</tr>
<tr>
<td>Standing-up</td>
<td>$R$: 0.3519</td>
<td>$R$: 0.3907</td>
</tr>
<tr>
<td></td>
<td>$P$: 0.6322</td>
<td>$P$: 0.6843</td>
</tr>
<tr>
<td></td>
<td>$F_1$: 0.4522</td>
<td>$F_1$: 0.4974</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Action</th>
<th>EDC</th>
<th>EDC + BSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>$R$: +0.0094</td>
<td>$R$: +0.0094</td>
</tr>
<tr>
<td></td>
<td>$P$: +0.0576</td>
<td>$P$: +0.0576</td>
</tr>
<tr>
<td></td>
<td>$F_1$: +0.0498</td>
<td>$F_1$: +0.0498</td>
</tr>
<tr>
<td>Jumping</td>
<td>$R$: +0.1125</td>
<td>$R$: +0.1125</td>
</tr>
<tr>
<td></td>
<td>$P$: +0.1053</td>
<td>$P$: +0.1053</td>
</tr>
<tr>
<td></td>
<td>$F_1$: +0.1175</td>
<td>$F_1$: +0.1175</td>
</tr>
<tr>
<td>Falling</td>
<td>$R$: +0.0181</td>
<td>$R$: +0.0181</td>
</tr>
<tr>
<td></td>
<td>$P$: +0.0768</td>
<td>$P$: +0.0768</td>
</tr>
<tr>
<td></td>
<td>$F_1$: +0.0327</td>
<td>$F_1$: +0.0327</td>
</tr>
<tr>
<td>Standing-up</td>
<td>$R$: +0.0388</td>
<td>$R$: +0.0388</td>
</tr>
<tr>
<td></td>
<td>$P$: +0.0521</td>
<td>$P$: +0.0521</td>
</tr>
<tr>
<td></td>
<td>$F_1$: +0.0452</td>
<td>$F_1$: +0.0452</td>
</tr>
</tbody>
</table>
### Table 10: Recall ($R$), precision ($P$) and $F_1$-measure for four actions in long jumps with (a) EDC without scheduler and (b) EDC with scheduler. Table (c) is the difference between EDC with and without scheduler.

<table>
<thead>
<tr>
<th>Action</th>
<th>EDC</th>
<th>EDC+BSS</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>0.5912</td>
<td>0.5990</td>
<td>-0.0078</td>
</tr>
<tr>
<td>Jumping</td>
<td>0.4037</td>
<td>0.4510</td>
<td>+0.0473</td>
</tr>
<tr>
<td>Falling</td>
<td>0.4560</td>
<td>0.5040</td>
<td>+0.0480</td>
</tr>
<tr>
<td>Standing-up</td>
<td>0.3490</td>
<td>0.3963</td>
<td>+0.0472</td>
</tr>
</tbody>
</table>

### Table 11: Recall ($R$), precision ($P$) and $F_1$-measure for four actions in triple jumps with (a) EDC without scheduler and (b) EDC with scheduler. Table (c) is the difference between EDC with and without scheduler.

<table>
<thead>
<tr>
<th>Action</th>
<th>EDC</th>
<th>EDC+BSS</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>0.4253</td>
<td>0.4778</td>
<td>+0.0525</td>
</tr>
<tr>
<td>Jumping</td>
<td>0.3532</td>
<td>0.4875</td>
<td>+0.1343</td>
</tr>
<tr>
<td>Falling</td>
<td>0.4371</td>
<td>0.5251</td>
<td>+0.0880</td>
</tr>
<tr>
<td>Standing-up</td>
<td>0.3098</td>
<td>0.3862</td>
<td>+0.0764</td>
</tr>
</tbody>
</table>
The difference is highly significant for high jumps and triple jumps but less significant for long jumps and pole vaults. In the two last types of jumps, running action is better detected with MLGBT because, on the one hand, it is much more represented in the learning set and, in the other hand, MLGBT is a probabilistic method thus sensitive to frequent patterns. Running action generally takes about 50% of each jump (see table 2 which shows that long jumps and pole vaults present the higher number of running frames).

<table>
<thead>
<tr>
<th></th>
<th>High jumps</th>
<th></th>
<th>Pole vaults</th>
<th></th>
<th>Long jumps</th>
<th></th>
<th>Triple jumps</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R$</td>
<td>+0.0712</td>
<td>$P$</td>
<td>-0.0259</td>
<td>$F_1$</td>
<td>+0.0583</td>
<td>$R$</td>
<td>-0.2073</td>
</tr>
<tr>
<td></td>
<td>+0.0590</td>
<td></td>
<td>+0.0066</td>
<td></td>
<td>+0.0452</td>
<td></td>
<td>+0.0553</td>
</tr>
<tr>
<td></td>
<td>+0.2154</td>
<td></td>
<td>-0.1009</td>
<td></td>
<td>+0.1379</td>
<td></td>
<td>-0.0181</td>
</tr>
<tr>
<td></td>
<td>+0.1280</td>
<td></td>
<td>+0.1329</td>
<td></td>
<td>+0.1467</td>
<td></td>
<td>+0.0455</td>
</tr>
</tbody>
</table>

Table 12: Differences between EDC+BSS performance and MLGBT+BSS performance (in terms of recall ($R$), precision ($P$) and $F_1$-measure) for the four actions (one line per table) in each jump (one table per jump).
5.7. Comparing TBM-based Belief Scheduler and probability-based Hidden Markov Models for a classification task

In this section we compare the proposed TBM-based Belief Scheduler to probability-based Hidden Markov Models for classification purpose.

Four models of jumps are built: high jumps, long jumps, pole vaults and triple jumps. Prior, transition matrix and observations mixtures of HMM have been learned using the BNT toolbox [40]. Each state is modeled by a mixture of Gaussians (conditionally independent on states). The number of components is set by the MDL-based criterion proposed in [35] and varies from 3 to 6 according the type of jump.

Each HMM and each scheduler is made of understandable states, the same states are chosen for both systems. High jump, pole vaults and long jumps sequences are described as the following sequence: running → jumping → falling → standing up. For triple jumps, the sequence is made of eight states: running → jumping → falling → jumping → falling → jumping → falling → standing up.

For the comparison and in order to show the interest of TBM theory associated with Belief Scheduler compared to Probability Theory associated with HMM for this task, we use the same mixtures of Gaussians for both systems (only MLGBT modeling). Likelihoods provided by the mixtures of Gaussians are transformed into belief functions using the Generalized Bayesian Theorem (Eq. 4) before processing by the Belief Scheduler.

To assess both systems, we use the Viterbi algorithm for HMM [10] and the GQ criterion for the Belief Scheduler. The Viterbi algorithm is applied given each model of jumps providing four log-likelihoods, one for each re-
trieved sequence. The video is classified as a particular jump if the log-
likelihood of this jump is the highest one. For the same video, we apply the
Belief Scheduler and we choose the model that maximizes the Global Quality
recognition performance criterion.

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>pv</td>
<td>lj</td>
</tr>
<tr>
<td>pv</td>
<td>19</td>
</tr>
<tr>
<td>lj</td>
<td>0</td>
</tr>
<tr>
<td>tj</td>
<td>0</td>
</tr>
<tr>
<td>hj</td>
<td>2</td>
</tr>
<tr>
<td>rej</td>
<td>5</td>
</tr>
</tbody>
</table>


The results are gathered in the confusion matrices of Table 13. The su-
periority of the Belief Scheduler is clearly demonstrated on this dataset. The
overall classification rate is 71% without rejections and 93% with rejections
for the Belief Scheduler whereas it is about 54% for Hidden Markov Mod-
els. Bad results of log-likelihood based classification on this dataset using
HMM are explained by at least two reasons: first, there is no class of re-
jections thus the decision is obligatory taken without any other alternative.
Secondly, there is certainly a sensitivity to actions and sequences length in
the computation of log-likelihoods. The action and sequence length plays
such a role that, for triple jumps, we have classified manually the videos (a
video of triple jump is well recognized if at least 2 jumping actions and 2
falling actions are recognized), because otherwise the detection rate for this
(a) Log-likelihoods in HMM.

(b) GQ criterion in Belief Scheduler (red points: errors, blue points: rejection).

Figure 12: Recognition criteria evolution for HMM (a) and Belief Scheduler (b) of the four jump models applied on 26 pole vault video sequences. The pole vault model (bold blue line) is generally better than other models.

type of video is almost 0% using log-likelihoods. This is due to the fact that triple jumps videos are generally longer than others (20 – 50%), while actions within sequences are very short (and difficult to catch) except running (longer and easier to detect). For the GQ criterion and Belief Scheduler case, triple jumps have been automatically classified. Note that, one can use more complex HMM with higher number of states, higher number of components in mixtures, explicit duration state and so on, yielding, may be, better results.
Figure 12 presents evolution of log-likelihoods for HMM and Global Quality recognition performance criterion for Belief Scheduler for 26 pole vaults videos analyzed by the four models (high jump, pole vault, triple jump and long jump). The GQ criterion (figure 12(b)) provides more reliable decision than HMM's log-likelihoods (figure 12(a)) since the relative difference between jumps is high, whereas log-likelihoods are sometimes very close (it is difficult to decide). The dotted line in figure 12(b) represents the threshold on quality (50%) which was used for adaptation (class of rejections). Blue and big points in figure 12(b) represents rejection cases, whereas red and big points concerns recognition errors (decide high jumps instead of pole vaults).

The two first videos are acquired by far view point, thus camera motion is very low or null and silhouette size very little therefore, features are disturbed. The last three ones correspond to cases where the athlete motion is perpendicular to the image plane. Interestingly, the system indicates that a specific model must be learn for these cases in order to improve the recognition. Figure 15 (resp. 22th and 24th pole vaults) presents samples of the
Figure 13 depicts the evolution of a Global Quality recognition performance criterion along time (it is an on-line criterion) for a pole vault. This curve is interesting for monitoring. The system indicates that the decision is “pole vault” with high quality (about 78%) and reliability (high gap with the second which is high jump).

Figure 14 describes results of actions (states) and activities (sequences) recognition for a triple jump described by eight states (a) the detected states by the Belief Scheduler, and (b) the Viterbi result when applied on the likelihoods. For both sets of images, the first action of the sequence is running located at the top while standing up is the last one and located at the bottom. All recognizers’ parameters are obviously the same as the one used for the classification task (Tab. 13).

database with front-back camera view for which the recognition fails.
Figure 15: (a) Athlete’s motion is parallel to image plane, so the projected human motion variation on the image plane is maximized (good camera view). (b) Athlete’s motion is perpendicular to the image plane (bad camera view) increasing the level of difficulty of action discrimination.

6. Conclusion and future work

We have proposed a state machine called Belief Scheduler that allows to find out state sequences from temporal belief functions in Transferable Belief Model framework. The Belief Scheduler relies on a temporal filter adapted for smoothing these evidential data. The transition from one state to another is controled by a criterion based on conflict between the current model of prediction and observations. This conflict reflects how unlikely is the model and informs on the need to make a transition. We have tested the system on a real problem concerning action and activity recognition in athletics videos. We have compared two evidential methods for action detection in order to generate the belief functions at each instant and we have shown the differences with HMM for a classification task.

Experiments have shown good performance of the Belief Scheduler for several actions in several types of jumps in terms of detection improvement. The first modeling method based on the Generalized Bayesian Theorem and
the Generalized Likelihood Principle [30, 32] and applied on mixtures of Gaussians is powerful to provide belief functions on states. However, the distance model seems more adapted for the application concerned in this paper. The difference comes from the fact that the distance model directly generates a belief function while the former generates a probabilistic result that is transformed into a belief function.

Experiments have also emphasized that the inference criterion of the Belief Scheduler enables one to create a class of rejections which improves the classification results and points out new sequences. The “Global Quality recognition performance” (GQ) criterion, proposed in this paper, is bounded between 0% and 100%. Therefore, a class of rejections can be created using a simple threshold (e.g. 50% for instance).

The class of rejections is a first step toward adaptation, since it gathers the cases for which the system of recognition could not take a decision. We plan to use the proposed inference criterion GQ for adaptation: when the best sequence has a GQ criterion less than the rejection threshold, the video is rejected and this is a mean to adapt the system. The features corresponding to the rejection cases can show strong similarities among themselves, such as many rejected videos corresponding to the cases where the athlete motion is perpendicular to the image plane. Work is under progress to pursue pattern discovery and adaptation, which are promising and fascinating in many applications.
References


[26] C. Panagiotakis, G. Tziritas, Recognition and tracking of the members of a moving human body, in: Int. Conf. on Articulated Motion and Deformable Objects (AMDO), Palma De Mallorca, Spain, 2004, pp. 86–98.


54


