Belief Scheduler based on model failure detection in the TBM framework. Application to human activity recognition.

Emmanuel Ramasso, Costas Panagiotakis, Michèle Rombaut, Denis Pellerin

Belief Scheduler based on model failure detection in the TBM framework. Application to human activity recognition

E. Ramasso[a], C. Panagiotakis[b], M. Rombaut[c], D. Pellerin[c]

[a] *FEMTO-ST Institute, UFC / ENSMM / UTBM, Automatic Control and Micro-Mechatronic Systems Department, 25000 Besançon, France*
[b] *Computer Science Department, University of Crete, P.O. Box 2208, Heraklion, Greece.*
[c] *GIPSA-lab, Image and Signal Department (DIS), 961 Rue de la Houille Blanche, BP46, 38402 Saint Martin d'Hères, France.*

Abstract

A tool called Belief Scheduler is proposed for state sequence recognition in the Transferable Belief Model (TBM) framework. This tool makes noisy temporal belief functions smoother using a Temporal Evidential Filter (TEF). The Belief Scheduler makes belief on states smoother, separates the states (assumed to be *true* or *false*) and synchronizes them in order to infer the sequence. A criterion is also provided to assess the appropriateness between observed belief functions and a given sequence model. This criterion is based on the conflict information appearing explicitly in the TBM when combining observed belief functions with predictions. The Belief Scheduler is part of a generic architecture developed for on-line and automatic human action and activity recognition in videos of athletics taken with a moving camera. In experiments, the system is assessed on a database composed of 69 real athletics video sequences. The goal is to automatically recognize *running*, *jumping*, *falling* and *standing-up* actions as well as *high jump*, *pole vault*, *triple jump* and *long jump* activities of an athlete. A comparison with Hidden Markov

Models for video classification is also provided.

Key words: Sequence recognition, Belief finite state machine, Transferable Belief Model, Temporal Evidential Filter, Conflict, 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 of **human *detection*, *tracking* [6]** and *activity understanding* [7] where *detection* and *tracking* aim at locating human limbs while *activity understanding* is a higher level task aiming at recognizing human *actions* and using *ordered sequences of actions* to recognize *activities* [8].

Hidden Markov Models (HMM), initially proposed for speech processing [9] is the most common method used for human action and activity recognition. Like most approaches in human motion analysis, and more generally in sequence recognition, HMM rely on Probability Theory. Several drawbacks inherent to these usual methods can be mentioned [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 implies re-estimating the model parameters. Moreover, it is difficult to interpret the models and therefore, a user can barely understand action and activity models since the systems gen-

erally appear as "black boxes". Lastly, actions and activities of humans can generally not be separated. Indeed, in the state of the art, one model is built for each activity and a log-likelihood is computed for the sequence. However, information on actions within activities is not available or not reliable.

1.2. Using the Transferable Belief Model for sequence recognition

Possibility, probability and belief functions are three alternative measures of uncertainty used for knowledge representation [10]. A belief function is a general measure that can encode and combine a *variety of knowledge* wider than *probability measures* and was the basis of Dempster-Shafer's theory of evidence and of the *Transferable Belief Model* (TBM) [11, 12, 13]. Recently, new tools were proposed for pattern recognition that showed the efficiency of approaches based on belief functions [14, 15, 16, 17]. In this paper, we consider the general and sound framework of the TBM proposed by Smets and Kennes [12] as an alternative to probability methods for temporal sequence modeling and recognition.

The TBM applications in the context of state sequence recognition and in human motion analysis from video is just in its infancy, partially because the TBM is a recent theory compared to Probability Theory. Human motion analysis based on the TBM was pioneered by Hammal et al. [18, 19] and Girondel et al. [20]. However the authors focus on static recognition of human expressions and postures and thus the dynamic aspects of human motion were not modeled.

One of the first tools used for the analysis of state sequence in the TBM was proposed by Rombaut et al. [21] in 1999. The authors developed a generalization of a Petri Net to belief functions based on the Generalized

Bayesian Theorem (GBT) [22]. This Belief Petri Net is, however, not robust to noise because links between states at successive times are given by an evolving and sparse transition matrix depending on sensor measurements. Moreover, no classification criterion was proposed.

The second tool is the generalized HMM proposed in 2000 by Mohamad and Gader [23] where the generalization is narrowed down to possibility measures and thereby their framework is not able to manage belief functions. One advantage of their framework is the possibility of managing dependent observations by using fuzzy operators but the authors used the product, thereby assuming statistical independence.

The third tool is the generalized Kalman filter [24] proposed by Smets and Ristic in 2004 for joint tracking and classification in the TBM framework. The Kalman equations in the tracking step are quite similar to the probabilistic version but the TBM showed better results for the classification step on a military problem using implication rules. The first problem with the generalized Kalman filters for the application concerned on human motion analysis is that they rely on linear dynamic systems that must be identified. However, human motions can be highly non-linear and depend on the camera view-point and thus are not known in advance, except in specific situations. Moreover, as presented in Section 2, five features are extracted and twenty actions are detected in four types of jumps, thus the number of parameters can be high. In [6], the authors propose an alternative to a Kalman Filter using particle filter. The second problem is that Kalman filters are used when the states are continuous while we are interested in detecting human actions which are discrete. Moreover, in the classification step, the implication rules

used in [24] also require parameters that can be given by experts in some applications. Actually, HMM are preferred to Kalman filters for human motion analysis [1, 2, 4, 7] because HMM are suitable for discrete states and one can use any type of distribution, while a Kalman filter assumes every distribution to be Gaussian.

1.3. Contributions and paper overview

The goal of the system is to determine the most likely activity defined as a sequence of actions. An activity can be described by a graph where each node corresponds to an action. At anytime, using the features extracted from the videos, the system determines what the current action is. **The transition is made when the current action becomes false and the next action becomes true. All the other actions of the graph are false.** The extracted features are noisy and cannot be directly used for activity recognition. The temporal belief functions associated with the actions are made smoother by the Temporal Evidential Filter. The main contribution of this paper is a tool called *Belief Scheduler* [25], developed in the TBM framework, which recognizes states (representing actions) and sequences of states (representing activities) in an *on-line* manner. This tool is a deterministic state machine where transitions between states are controlled by additional parameters (experiments showed that only two parameters are really sensitive). An original inference criterion based on conflict is also proposed for sequence classification. The other contribution is the design of a generic architecture for human action and activity recognition based on the TBM. Lastly, we propose several experiments and a comparison with HMM on athletic videos.

The paper is organized as follows. Section 2 presents the components of the architecture for human motion analysis. Background on the TBM and presentation of the Temporal Evidential Filter [26] (used in the Belief Scheduler) are given in Section 3. The Belief Scheduler is then described in Section 4. Finally, Section 5 provides results of experiments on human action and activity recognition.

2. Architecture for human motion analysis

Human action and activity recognition requires several steps that can be represented as in the architecture presented in Fig. 1. The proposed architecture is built so as to be generic enough to add new features and new actions. The *low level* part provides relevant features concerning actions that are extracted from the video stream. The *high level* part starts with the conversion of the feature values into beliefs on actions which are then filtered by the *Temporal Evidential Filter* (TEF) [26] to make action detection more reliable. Then, in order to infer activity, sequences of actions are recognized using the *Belief Scheduler*. A quality criterion is computed on-line to assess the confidence of actions and activities.

Robust shape/motion features are automatically extracted each time from the video using a camera motion estimator and a tracking algorithm. The camera motion estimator [27] provides horizontal (P_{hm}) and vertical (P_{vm}) motions as well as divergence (P_{div}). The dominant motion image is obtained from the camera motion estimation where the intensity of a pixel depends on its membership of the dominant motion that is assumed to be the motion of the background. Fig. 2(b) depicts dominant motion for images corresponding

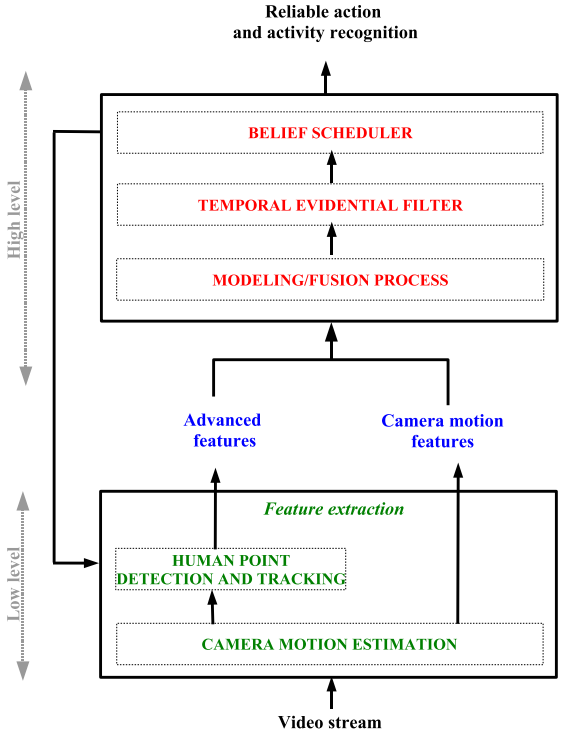
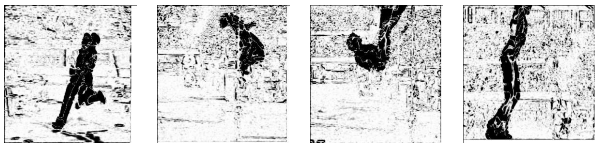


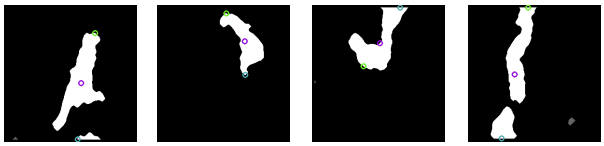
Figure 1: System architecture for human motion analysis.

to *running*, *jumping*, *falling* and *standing-up* actions in a *high jump* sequence. The second source of features is a human detection/tracking algorithm which provides human head, center of gravity and end of legs position (Fig. 2(c)) from the dominant motion images. The variation of the center of gravity (P_{vcg}), the angle between horizon and human axis (P_{swing}) are then computed. The feature vector is denoted $\mathbf{O}_t = [P_{\text{hm}} \;\; P_{\text{vm}} \;\; P_{\text{div}} \;\; P_{\text{vcg}} \;\; P_{\text{swing}}]$.

(a) Original video sequence.



(b) Dominant motion images.



(c) Human point detection and tracking.

Figure 2: Original video sequence (a), dominant motion images (b) and human point detection and tracking results (c) for a *high jump* (with *running*, *jumping*, *falling* and *standing-up* actions).

3. Models of actions

At anytime, only the truth of the current action and the next action is addressed. At this time, the other actions have no influence on activity recognition. The focus on these two actions can be done early by directly modeling their truth from the extracted features (*graph theory approach*), or can be done late after a global fusion process on all the actions (*fusion theory approach*). Following previous works [21], we have chosen the *early focus* for its efficiency.

We present below two evidential methods ("likelihood" and "distance" models) that link numerical features $\mathbf{O}_t$ to belief on actions.

3.1. Basic belief assignment

As in the graph theory, the Frame of Discernment (FoD) for each action $A_k \in \{$*running, jumping, falling, standing-up*$\}$ is binary (either true or false) and denoted as $\Omega_k^t = \{T_k^t, F_k^t\}$ where time t is explicit, since we consider that belief actions evolve over time. The power set $2^{\Omega_k^t} = \{\{T_k^t\}, \{F_k^t\}, \{T_k^t, F_k^t\}, \emptyset_k^t\}$ gathers the subsets of the FoD (called propositions). For the sake of simplicity, braces around the propositions are not written. The belief mass on subset $\{T_k^t, F_k^t\}$ can be interpreted as the weight of the logical proposition $T_k^t \cup F_k^t$, meaning that state of action A_k at t is imprecise (either true: T_k^t, OR false: F_k^t).

The goal is to define the belief functions $m^{\Omega_k^t}$ on $2^{\Omega_k^t}$ concerning the actions A_k related to observed features $\mathbf{O}_t$ at time t. Obtaining a belief function from features can be stated as a problem of pattern recognition [14], i.e. we need to build a mapping from the feature space $\mathbb{R}^F$ to action space Ω_k^t. The mapping can be obtained automatically using:

- **The model of likelihood** (MLGBT[1]) which consists in applying the Generalized Bayesian Theorem (GBT) [22] to likelihood conditional of action states [28, 29, 14].
- **The model of distance** (EDC[2]) of Denœux et al. [30, 31]. This method is interesting when the models of classes are not known and/or difficult to obtain.

In order to define the *basic belief assignment* (BBA) directly

[1]Stands for Model of Likelihood based on Generalized Bayesian Theorem.
[2]Stands for Evidential Distance-based Classifier.

on Ω_k^t, it is necessary to build a learning set composed of two sets of features: one where action A_k is true and one where action A_k is false. Feature intervals where action A_k is true are easy to find (using the ground truth) but the problem is to choose intervals where action A_k is false. That is why we choose to model knowledge by a (BBA) named $m^{\Omega_s^t}$ on the FoD $\Omega_s^t = \{Run, Jmp, Fal, Stu\}$ (standing for *running*, *jumping*, *falling*, *standing-up* respectively) which is the set of the four actions. Then the BBA $m^{\Omega_k^t}$ on the two actions concerned (current and next) is computed by a coarsening process, seen as a focus process.

3.2. Model of likelihood (MLGBT)

We first estimate conditional probability densities of observed features $\mathbf{O}_t$ given each action A_k. For example, the densities can 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 [32] based on Minimum Description Length: 10, 4, 4 and 8 components are used for *running*, *jumping*, *falling* and *standing-up* actions respectively.

Afterward, given an unknown feature vector $\mathbf{O}_t$ at t, a likelihood $P(\mathbf{O}_t|A_k)$ is generated for each action A_k. Then, as proposed by Smets et al. [22, 28, 29], these likelihoods are supposed to represent plausibilities of observations conditional to states, i.e. $pl^{\mathbb{R}^F}[A_k](\mathbf{O}_t)$, defined in the feature space $\mathbb{R}^F$. They are used in the Generalized Bayesian Theorem in order to compute the pos-

terior belief mass $m^{\Omega_s^t}[\mathbf{O}_t](S^t)$ of $S_t \subseteq \Omega_s^t$ as follows [29]:

$$m^{\Omega_s^t}[\mathbf{O}_t](S^t) = \prod_{A_k \in S^t} pl^{\mathbb{R}^F}[A_k](\mathbf{O}_t) \cdot \prod_{A_k \notin S^t} \Big(1 - pl^{\mathbb{R}^F}[A_k](\mathbf{O}_t)\Big) \qquad (1)$$

where $m^{\Omega_s^t}$ is a BBA defined on the set of actions $\Omega_s^t = \{Run, Jmp, Fal, Stu\}$.

3.3. The model of distance (EDC)

A learning set with D samples is available as $\mathcal{L} = \{\mathbf{O}_d, m_d^{\Omega_s}\}$ where $d \in \{1, 2 \dots D\}$ is a sample index[3]. Each sample e_d is made up of observations $\mathbf{O}_d$ labeled by a belief function $m_d^{\Omega_s}$ defined on the set of actions $\Omega_s = \{Run, Jmp, Fal, Stu\}$. When the class of e_d is known then the belief function is *categorical* ($m_d^{\Omega_s}(A_k) = 1, A_k \in \Omega_s$) whereas if the class is unknown then $m_d^{\Omega_s}(\Omega_s) = 1$.

For a given observed feature vector $\mathbf{O}_t$, we need to assess the BBA $m^{\Omega_s}[\mathbf{O}_t]$ that reflects the type of action (this BBA is identical to $m^{\Omega_s^t}$ but the superscript t is not used for the sake of simplicity). Using the Denœux's distance model [30], the BBA is given by the conjunctive combination of the BBA of the $\mathcal{K}$ nearest neighborhoods $\mathbf{O}_t$ determined by the Euclidean distance. For that, let $\{\mathbf{O}_j, m_j^{\Omega_s}\} \in \mathcal{L}$ the subset of the $\mathcal{K}$ nearest neighborhoods. The BBA $m_j^{\Omega_s}[\mathbf{O}_j]$ for sample e_j in this subset is then obtained by:

$$\begin{aligned} m_j^{\Omega_s}[\mathbf{O}_j](\{A_k\}) &= \zeta \cdot \phi_q(dist(\mathbf{O}_j, \mathbf{O}_t)) \\ m_j^{\Omega_s}[\mathbf{O}_j](\Omega_s) &= 1 - \zeta \cdot \phi_q(dist(\mathbf{O}_j, \mathbf{O}_t)) \\ m_j^{\Omega_s}[\mathbf{O}_j](B) &= 0, \quad B \in 2^{\Omega_s} \backslash \{\Omega_s, \{A_k\}\} \end{aligned} \qquad (2)$$

[3] We do not use t here but d since time is not important for the modeling process. Time will be explicitly taken into account during sequence recognition.

where $A_k \in \Omega_s^t$, $\phi_q(dist(\mathbf{O}_j, \mathbf{O}_t)) = \exp(-\gamma_q \cdot dist(\mathbf{O}_j, \mathbf{O}_t))$, the function $dist(\mathbf{O}_j, \mathbf{O}_t)$ is the Euclidean distance between $\mathbf{O}_j$ and $\mathbf{O}_t$, $\gamma_q \geq 0$ and ζ is such that $0 < \zeta < 1$. The $\mathcal{K}$ BBA are then conjunctively combined Smets' conjunctive rule of combination (CRC) to estimate $m^{\Omega_s}[\mathbf{O}_t]$:

$$m_*^{\Omega_s}[\mathbf{O}_t] = m_1^{\Omega_s}[\mathbf{O}_1] \textcircled{\cap} \cdots \textcircled{\cap} m_j^{\Omega_s}[\mathbf{O}_j] \textcircled{\cap} \cdots \textcircled{\cap} m_{\mathcal{K}}^{\Omega_s}[\mathbf{O}_{\mathcal{K}}] \tag{3}$$

where the CRC $\textcircled{\cap}$ is defined by:

$$(m_1^{\Omega_s} \textcircled{\cap} m_2^{\Omega_s})(D) = \sum_{B \cap C = D} m_1^{\Omega_s}(B) \cdot m_2^{\Omega_s}(C) \tag{4}$$

At this step, the mass $m^{\Omega_s}(\emptyset)$ on the empty set can be different than zero. That can correspond to a transition between two actions where they seems together true. Then we normalize the CRC as follows: $m^{\Omega_s}(B) = \frac{m_*^{\Omega_s}(B)}{1 - m_*^{\Omega_s}(\emptyset)}, \forall B \subseteq \Omega_s, B \neq \emptyset$. We have set $\mathcal{K} = 5$ and $\zeta = 0.99$ using heuristics attached to the application through several tests. The value of γ_q is optimized using a gradient-based method proposed in [31][4]. The learning set represents 30% of the whole dataset (as for the MLGBT model).

3.4. Coarsening process

In both modeling methods (MLGBT and EDC), the FoD is Ω_s^t. Because we have chosen to focus early on the current action and the next action, the BBA $m^{\Omega_s^t}$ is then coarsened onto one $m^{\Omega_k^t}$ for

[4]Matlab code available at `http://www.hds.utc.fr/~tdenoeux/`.

these actions. The coarsening process is then:

$$\begin{aligned} m^{\Omega_k^t}(T_k^t) &\leftarrow m^{\Omega_s^t}(A_k) \\ m^{\Omega_k^t}(F_k^t) &\leftarrow \sum_{\substack{A_k \cap B_k = \emptyset \\ B_k \subseteq \Omega_s^t}} m^{\Omega_s^t}(B_k) \\ m^{\Omega_k^t}(T_k^t \cup F_k^t) &\leftarrow \sum_{\substack{A_k \cap B_k \neq \emptyset \\ B_k \neq A_k, B_k \subseteq \Omega_s^t}} m^{\Omega_s^t}(B_k) \end{aligned} \tag{5}$$

where A_k is the current and next actions and B_k is the other actions known as false. Fig. 9 depicts some results (output of the modeling process).

Another alternative could be to directly carry out a coarsening process from Ω_s^t to the frame of discernment $\Omega_{i,i+1}^t$ where A_i is the current action and A_{i+1} is the next action. The effect of that alternative is similar to the previous one.

3.5. Temporal Evidential Filter for action state filtering

Because the features extracted from the videos are noisy, not perfectly reliable and conflicting, it is necessary to filter the BBA $m^{\Omega_k^t}$ obtained previously. The Temporal Evidential Filter (TEF) proposed in [26] makes belief on actions temporally consistent (the resulting belief has no conflict and is made smooth). On the other hand, this filter is used to detect when the states (false or true) of actions change. This filter is relatively easy to work out because the BBA $m^{\Omega_k^t}$ concerned are binary. That is not the case of $m^{\Omega_s^t}$.

The TEF works on-line on each action A_k independently taking as input the BBA obtained from feature fusion and the previous TEF output (Fig. 3).

In this section, the eight steps of the TEF process are recalled [26].

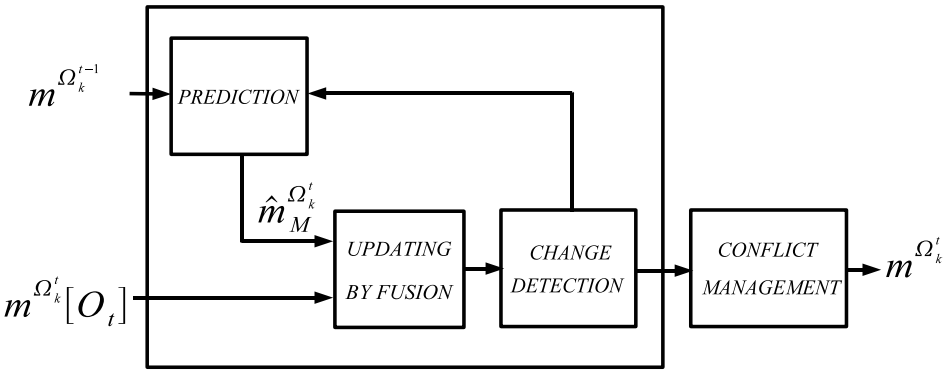


Figure 3: The *Temporal Evidential Filter* principle.

The TEF uses a model of belief evolution $\mathcal{M} \in \{\mathcal{T}, \mathcal{F}\}$, one for each state ($\mathcal{T}$ for T_k^t and $\mathcal{F}$ for F_k^t). Only one model is applied at each time t and each model assumes that the BBA of the current TEF output $m^{\Omega_k^t}$ at time t is close to the previous one $m^{\Omega_k^{t-1}}$ (this is a common hypothesis in filtering, in particular for our application since human motions are continuous).

1-Prediction: The model of evolution is used to predict the current state of each action $\hat{m}_{\mathcal{M}}^{\Omega_k^t}$ (at time t) by combining the BBA of the current model of belief evolution and the previous TEF output $m^{\Omega_k^{t-1}}$ resulting in two possible BBA [26]: either $\hat{m}_{\mathcal{T}}^{\Omega_k^t}$ if the current model is $\mathcal{T}$ or $\hat{m}_{\mathcal{F}}^{\Omega_k^t}$ if the current model is $\mathcal{F}$. These BBAs are given by:

$$\left\{ \begin{array}{lcl} \hat{m}_{\mathcal{T}}^{\Omega_k^t}(T_k^t) & = & \gamma_{\mathcal{T}} \cdot m^{\Omega_k^{t-1}}(T_k^{t-1}) \\ \hat{m}_{\mathcal{T}}^{\Omega_k^t}(\Omega_k^t) & = & \gamma_{\mathcal{T}} \cdot m^{\Omega_k^{t-1}}(\Omega_k^{t-1}) + 1 - \gamma_{\mathcal{T}} \end{array} \right. \tag{6}$$

$$\left\{ \begin{array}{lcl} \hat{m}_{\mathcal{F}}^{\Omega_k^t}(F_k^t) & = & \gamma_{\mathcal{F}} \cdot m^{\Omega_k^{t-1}}(F_k^{t-1}) \\ \hat{m}_{\mathcal{F}}^{\Omega_k^t}(\Omega_k^t) & = & \gamma_{\mathcal{F}} \cdot m^{\Omega_k^{t-1}}(\Omega_k^{t-1}) + 1 - \gamma_{\mathcal{F}} \end{array} \right. \tag{7}$$

In this paper we always have set parameters $\gamma_{\mathcal{T}}$ and $\gamma_{\mathcal{F}}$ to 0.9. It is important to note that the masses sum to one because of the redistribution rule proposed in step 6 that compels the mass at $t-1$ to be a simple belief function.

2-Fusion of prediction and measure: $\hat{m}_{\mathcal{M}}^{\Omega_k^t} \textcircled{\cap} m^{\Omega_k^t}[\mathbf{O}_t]$ combines the available information (prediction and observation), where the operator $\textcircled{\cap}$ is the conjunctive rule of combination defined in equation 4.

3-Conflict: $\epsilon_k^t = \left(\hat{m}_{\mathcal{M}}^{\Omega_k^t} \textcircled{\cap} m^{\Omega_k^t}[\mathbf{O}_t]\right)(\emptyset_k^t)$ quantifies the contradiction between model of belief evolution and data. The higher the conflict, the higher the necessity to change the current model (true or false). We thus introduce the concept of *unlikelihood* in order to give a semantic to the conflict value.

4-Cusum: $\mathbf{CS}_k(t) = \lambda \times \mathbf{CS}_k(t-1) + \epsilon_k^t$ builds the cumulative sum of conflict along time where $\lambda \in [0,1]$ 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 $\mathbf{CS}_k(t) > \mathcal{T}_s^k$ (stop threshold) at time t_s, the model is changed. The other model is applied from t_s and belief on interval of times $[t_s - \mathcal{W}, t_s]$ is compelled to be vacuous (i.e. $m^{\Omega_k^t}(\Omega_k^t) = 1$) to emphasize action state transition ($\mathcal{W} = 3$ is one window size representing transition size).

The threshold $\mathcal{T}_s^k$ can be easily estimated in four steps. These steps are described in the following (and each step is pictorially described in Fig. 4):

a) The ground truth is in the form of an interval of times where the action is really true. For instance, on Fig. 4, the ground truth appears as a bold black line on the time axis between time 48 and 61. The vertical dashed line represents the true beginning of the action. From the $\mathbf{O}_t$ vector, the temporal belief functions are computed and represented in

the first plot of Fig. 4. The blue, red and green curves represent the evolution of $m^{\Omega_k^t}[\mathbf{O}_t](T_k^t)$, $m^{\Omega_k^t}[\mathbf{O}_t](F_k^t)$ and $m^{\Omega_k^t}[\mathbf{O}_t](T_k^t \cup F_k^t)$ respectively.

b) First, we set the value of $\mathcal{T}_s$ to a unreachable value (infinity for example) and we apply the filter. Initially, the current model is the false one ($\mathcal{F}$). We thus obtain the second plot on Fig. 4. As expected, the belief on $m^{\Omega_k^t}[\mathbf{O}_t](T_k^t)$ is always zero (blue curve) due to the unreachable value of the stop threshold (no model change is possible and $\mathcal{F}$ is always the current one).

c) The CUSUM is represented in the third plot of Fig. 4. We choose a time in the ground truth where the CUSUM is high, for instance at $t = 52$ we have $CS^k(52) = 2$. This time should obviously be chosen so as to be close enough to the beginning of the true action. So choosing $\mathcal{T}_s^k = 2$ in this example could allow the proper detection of the action.

d) We set $\mathcal{T}_s^k = 2$ and apply the filter with this new threshold. This leads to the fourth plot on Fig. 4 where the action is correctly detected (a change correctly occurs from model $\mathcal{F}$ to model $\mathcal{T}$ model).

This estimation technique (which does not take the sequence into account) enables a rough value of the stop threshold to be estimated, that can then be refined by experiments.

6-TEF output: if the current conflict ϵ_k^t is low then the output is the fusion result of prediction and observations, otherwise we maintain the prediction (cautious approach). Formally: $m^{\Omega_k^t} = \hat{m}_{\mathcal{M}}^{\Omega_k^t} \textcircled{\cap} m^{\Omega_k^t}[\mathbf{O}_t]$ if $\epsilon_k^t \leq \delta_\emptyset$ and $\hat{m}_{\mathcal{M}}^{\Omega_k^t}$ otherwise where $\delta_\emptyset$ is a threshold reflecting a tolerance to the conflict

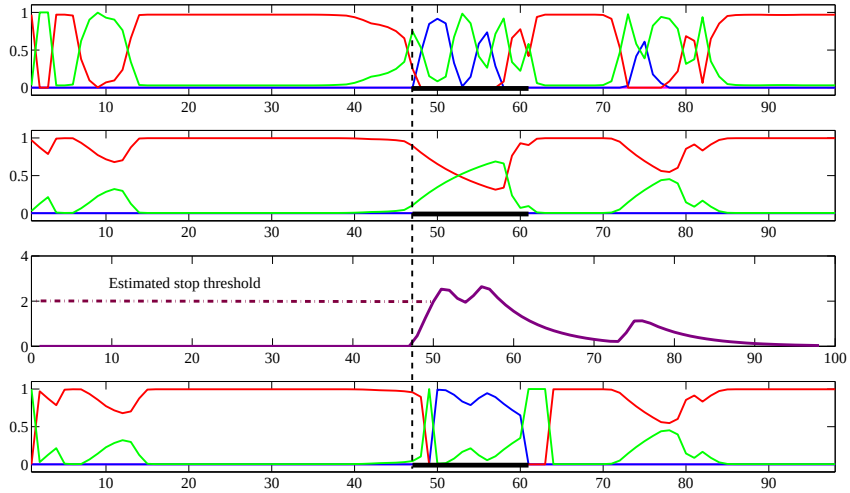


Figure 4: Estimation of $\mathcal{T}_s$. Explanations are given in the text (in the fifth step of section 3.5. The blue, green and red curves are respectively the evolution of belief on T_k^t (i.e. action A_k is true), on F_k^t (i.e. action A_k is false) and on $T_k^t \cup F_k^t$ (i.e. action A_k is true or false). Bold and black lines on the time axis represent ground truth for this video.

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

In order to remain coherent with the model of evolution that is used, the belief mass is modified as follows: if the model used is $\mathcal{T}$ then the belief on the empty set ($m^{\Omega_k^t}(\emptyset_k^t)$) and the belief on F_k^t ($m^{\Omega_k^t}(F_k^t)$) are transfered onto T_k^t and Ω_k^t respectively. The redistribution rule when the model is "$\mathcal{T}$: the state is true" is given by:

$$\begin{aligned} m^{\Omega_k^t}(T_k^t) &\leftarrow m^{\Omega_k^t}(T_k^t) + m^{\Omega_k^t}(\emptyset_k^t) \\ m^{\Omega_k^t}(\Omega_k^t) &\leftarrow m^{\Omega_k^t}(\Omega_k^t) + m^{\Omega_k^t}(F_k^t) \\ m^{\Omega_k^t}(\emptyset_k^t) &\leftarrow m^{\Omega_k^t}(F_k^t) = 0 \end{aligned} \tag{8}$$

A similar redistribution rule is used for the case "$\mathcal{F}$: the state is false"

replacing T_k^t by F_k^t. This redistribution is empirical and suitable for the TEF but one can also use other rules defined for instance in [33, 34].

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

$$LQ_i^{t_s:t}[\mathcal{M}](T_k^t) = \left(1 - \frac{1}{t - t_s}\right) \times LQ_i^{t_s:(t-1)}[\mathcal{M}](T_k^t) + \frac{m^{\Omega_k^t}(T_k^t)}{t - t_s} \cdot (1 - \epsilon_k^t) \quad (9)$$

for each action A_k within each activity S_i. This criterion represents a sliding weighted average (thus computed on-line) which uses past events and innovation. It uses conflict to weigh the current belief on T_k^t: the lower the conflict, the higher the confidence (or the plausibility) in the hypothesis "the true state is T_k^t". The weighted sum generates a smooth evolution of the criterion over time.

8-Transition and false alarm detection: Let say that at t_0, an action A_k in a sequence S_i is true and thus filtered by the model $\mathcal{T}$. When the stop threshold is reached at a given time t_1, we compare the Local Quality criterion $LQ_i^{t_s:t}[\mathcal{M}](T_k^t)$ (of action A_k in sequence S_i) with a threshold δ_{FA}. The threshold is the minimal quality value required to make a model change valid. Thus, if the criterion is higher than δ_{FA}, then the model change is declared to be valid. Otherwise, a false alarm occurs. In the latter case, the TEF is run again on the interval of time $[t_0, t_1]$ with a model compelled to be false (i.e. model $\mathcal{F}$) and with the CUSUM detector shunted (i.e. it does not take into account the stop threshold on this interval).

4. Belief Scheduler

Activity recognition is done when the K understandable actions A_k of the corresponding sequence have been true in the correct order. At any time, only the current action and next action states are taken into account. In the early focus process presented in this paper, the knowledge about these actions is given directly by the active models $\mathcal{T}$ and $\mathcal{F}$, and by the BBAs $m^{\Omega_k^t}$.

The method called *Belief Scheduler* [25] proposed for activity recognition based on the TBM is a state machine which exploits the results of the TEF to synchronize actions. It is built on the classical rules of such a machine: only the current action is assumed to be true at the given time and the other $(K-1)$ actions are thus false. Therefore, only one action uses the model $\mathcal{T}$ (in its associated TEF) whereas the other $(K-1)$ actions use the model $\mathcal{F}$ (in their associated TEF). The transition is passed when the current action becomes false and the next action becomes true.

The models $\mathcal{F}$ or $\mathcal{T}$ are considered as *resources* to which actions attempt to access. To access a model, an action has to ask for it and the *Belief Scheduler* manages this access. Ideally, the actions are synchronized (in this case, a simple state machine can be used) but, in real cases they can be either *overlapping* or *unconnected* as is represented in Fig. 5. Using particular rules, the *Belief Scheduler* overcomes these problems.

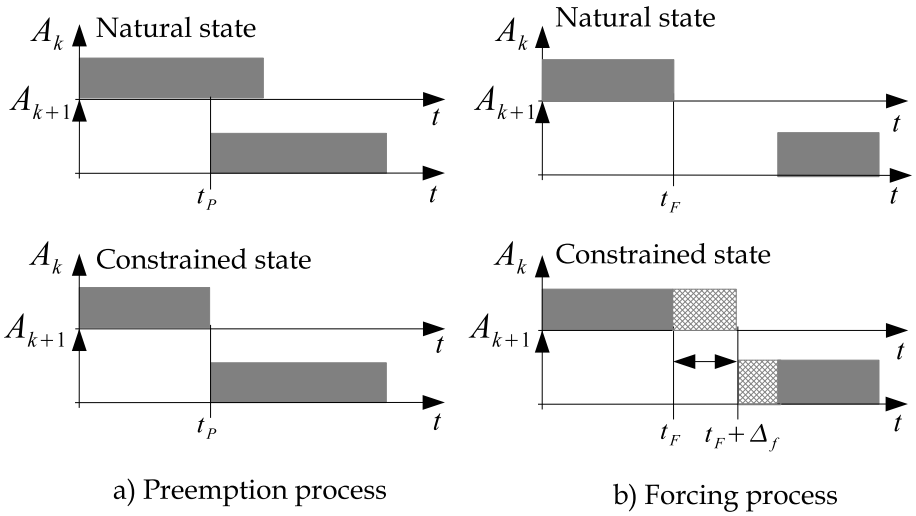


Figure 5: Due to data imperfection, overlapping (a) and unconnection (b) generally appear between current action A_k and the next action A_{k+1}.

4.1. Description

In the sequel, we call *natural* state the belief provided by the fusion process without filtering or scheduling. We call *constrained* state the belief provided by the scheduling process (it is constrained by the sequence).

4.1.1. PREEMPTION *process*

This process manages *overlapped* actions (Fig. 5.a). At $t = t_P$, 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 time $t = t_P$ when the CUSUM $CS_{k+1}(t)$ of the next action A_{k+1} is greater than its stop threshold $\mathcal{T}_s^{k+1}$:

$$\begin{array}{l} \text{if } CS_{k+1}(t) > \mathcal{T}_s^{k+1} \text{ and } CS_k(t) < \mathcal{T}_s^k \\ \text{then PREEMPTION and } t_P = t \text{ (current time)} \end{array} \tag{10}$$

In this case, the *natural* state of A_{k+1} is temporarily true (*true state*) from time t_P and the *constrained* state of A_k is temporarily false (*false state*) until *validation* (see Fig. 5). The validation is enabled when the quality of the action A_{k+1} recognition (which asks for PREEMPTION) is satisfactory (Section 4.1.3 focuses on this process). Information at $t = t_P$ concerning actions (cusum, belief ....), i.e. the *context*, is stored. This allows us to restore the context in case the PREEMPTION is not enabled. Note that, at the beginning of scheduling, all actions are in the *false* state. An artificial initial true state action is added to 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 *disconnected* actions (Fig. 5.b). At $t = t_F$, the current action A_k is false as well as the next action A_{k+1}. This process occurs at time t_F when the CUSUM $CS_k(t)$ of the current action A_k is greater than its stop threshold $\mathcal{T}_s^k$:

$$\begin{array}{l} \text{if } CS_k(t) > \mathcal{T}_s^k \text{ and } CS_{k+1}(t) < \mathcal{T}_s^{k+1} \\ \text{then FORCING and } t_F = t \text{ (current time)} \end{array} \tag{11}$$

If the two successive actions are disconnected with a gap smaller than a fixed threshold Δ_F, the *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 Δ_F. In this case, the action requiring a FORCING, e.g. *constrained* state of A_k, keeps on being true until the time "$t_F + \Delta_F$". At this time, the *constrained* state of A_{k+1} is forced to be true and *constrained* state of A_k becomes false (Fig. 5).

4.1.3. False alarm detection

If actions A_{k+1} and A_{k+2} are too unconnected and if A_{k+1} had previously preempted A_k, then A_{k+1} can be interpreted as a false alarm (see Fig. 6). It appears when an action becomes true instead of staying false. This false alarm procedure is applied to *validate* a PREEMPTION.

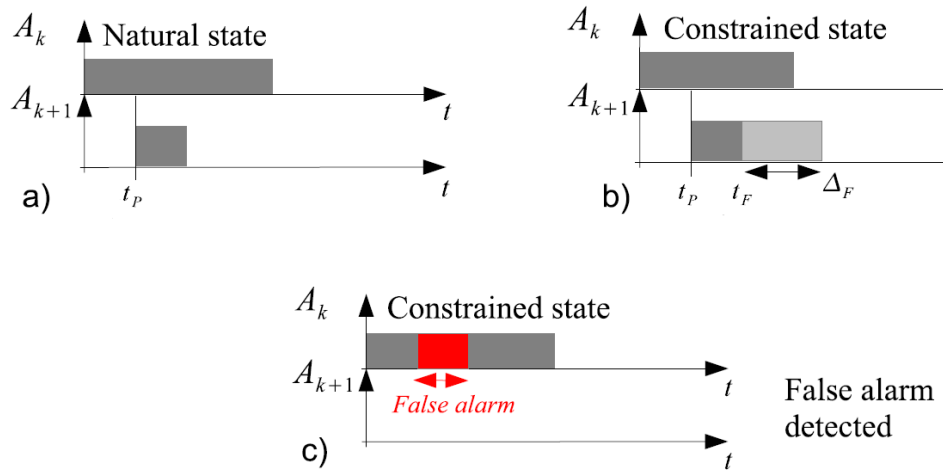


Figure 6: False alarm processing. a) Natural states of A_k and A_{k+1}. b) A_{k+1} is forced to be in a *true* state. c) A_{k+2} does not become true and the quality of A_{k+1} is bad, thus A_{k+1} is forced to be *false*.

In order to decide whether action A_{k+1} is a false alarm or not, we assess the *recognition performance* of this action. The criterion chosen is the *Local Quality recognition performance* $LQ_i^{t_P:t_F+\Delta_F}[\mathcal{T}](T_k^t)$ (action k in sequence i) defined in equation 9 and computed on interval of times $[t_P, t_F + \Delta_F]$ (the bounds are the time of PREEMPTION and of FORCING). As in Section 3.5, the following rule is applied to make an action valid or not:

$$\text{if } LQ_i^{t_P:t_F+\Delta_F}[\mathcal{T}](T_k^t) < \delta_{\text{FA}} \text{ then } A \text{ is a FALSE ALARM}$$

where δ_{FA} is a crisp threshold corresponding to a severity degree on the quality. When a false alarm is detected, the context of actions at time t_P (such as values of the CUSUM) is restored and the previous action (true before PREEMPTION), e.g. A_k, becomes true again. If $LQ_i^{t_P:t_F+\Delta_F}[\mathcal{T}](T_k^t) > \delta_{\text{FA}}$ then the quality is sufficient and therefore A_{k+2} becomes true and A_{k+1} and A_{k+2} are both validated.

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. One approach is to assign a score to each potential activity. 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* that is computed from the Local Quality recognition performance criterion. For that, each $LQ_i^{t_s:t}[\mathcal{T}](T_k^t)$ (only for model *true* and the *true* hypothesis), for all actions A_k in a particular activity S_i (composed of K_i actions) is aggregated into a *Global Quality recognition performance criterion* GQ_i^t to represent the confidence in activity S_i from time t_s (a given start time) to t (the current time). The aggregation is simply the

arithmetic mean:

$$GQ_i^t = \frac{1}{K_i} \sum_{n \in \{1..K_i\}} LQ_n^{t_s:t}[\mathcal{T}](T_k^t) \tag{12}$$

In order to find the best activity S_*^t at the current time t, we maximize GQ_i^t over all possible sequences. Then, a threshold is applied to decide whether the recognition is satisfactory. Formally:

$$S_*^t = \text{argmax}_{\,i}\, GQ_i^t > \theta \tag{13}$$

where θ 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 δ_{FA}.

5. Experiments

This part concerns the testing of the action/activity recognition architecture. The goal is to assess 1) the modeling using MLGBT (Model of Likelihood based on Generalized Bayesian Theorem) and EDC methods (Evidential Distance-based Classifier) before scheduling, and 2) the performance of the belief scheduler (BS) after filtering by the TEF (Temporal Evidential Filter) and scheduling. Because the Hidden Markov Models (HMM) are a reference in such applications, we have compared the results of the proposed approach to HMM approach.

5.1. Settings

The system was tested for action and activity recognition in athletics jumps. The database[5] 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 images (with 5600 images for *running*, 2700 for *jumping*, 2550 for *falling* and 1770 for *standing-up*). The database is characterized by its heterogeneity (Fig. 7) with a panel of view angles as well as environments and athletes (out/indoor, male, female, other moving people).

Figure 7: Heterogeneous database used for testing.

The proposed system was used to recognize actions, *running*, *falling*, *jumping* and *standing-up*, and activities (action sequence) *high jump*, *pole*

[5]Some videos and results are available on the author's website: `http://www.femto-st.fr/~emmanuel.ramasso/actionActivityRecognition.htm` and `www.csd.uoc.gr/~cpanag/DEMOS/actionActivityRecognition.htm`. Some codes for the TBM operations can be found in the TBMlab toolbox of Smets available at `http://iridia.ulb.ac.be/~psmets`.

vault, *triple jump* and *long jump*. The first three activities were described by a four-state belief scheduler (*running* → *jumping* → *falling* → *standing-up*) while triple jumps were described by a eight-state scheduler (*running* → *jumping* → *falling* → *jumping* → *falling* → *jumping* → *falling* → *standing-up*). The parameters of the TEF and the *Belief Scheduler* were tuned using 5-fold cross validations: 1) we selected 80% of the database, 2) made an estimation of the parameters so as to maximize recognition performance and 3) tested on the remaining 20%. We did it 5 times and computed the average of the performance. The best set of parameters is given in Tab. 1.

	Activity							
	Highjump		Longjump		Polevault		Triplejump	
	$\mathcal{T}_s$	Δ_F	$\mathcal{T}_s$	Δ_F	$\mathcal{T}_s$	Δ_F	$\mathcal{T}_s$	Δ_F
Running	3.1	10	3.1	5	3.1	10	1.7	2
Jumping	3.1	15	4.1	5	3.9	30	1.7	2
Falling	3.1	5	4.1	15	4.5	30	1.7	2
Standing-up	2.1	15	3.1	10	4.1	10	1.7	2

Table 1: TEF and scheduler parameter settings for $\mathcal{T}_s$ and Δ_F. The other parameters ($\lambda = 0.9$, $\gamma_{\mathcal{T}} = 0.9$, $\gamma_{\mathcal{F}} = 0.9$, $\mathcal{W} = 3$ and $\delta_{FA} = 50\%$) are set at the same value for all actions and all activities.

5.2. Tests and evaluation protocol

For quantitative evaluation, an action is said to be true if its pignistic probability (BetP) [35] defined by $\text{BetP}(T_k^t) = \frac{1}{(1-m(\emptyset))}(m(T_A) + \frac{m(T_A \cup F_A)}{2})$ 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 compared these decisions with the ground truth (the database was manually annotated). Recall ($\mathcal{R}$) and precision ($\mathcal{P}$) criteria were used [36]. They were computed as $\mathcal{R} = \frac{C \cap R}{C}$ and $\mathcal{P} = \frac{C \cap R}{R}$, where C is the set of

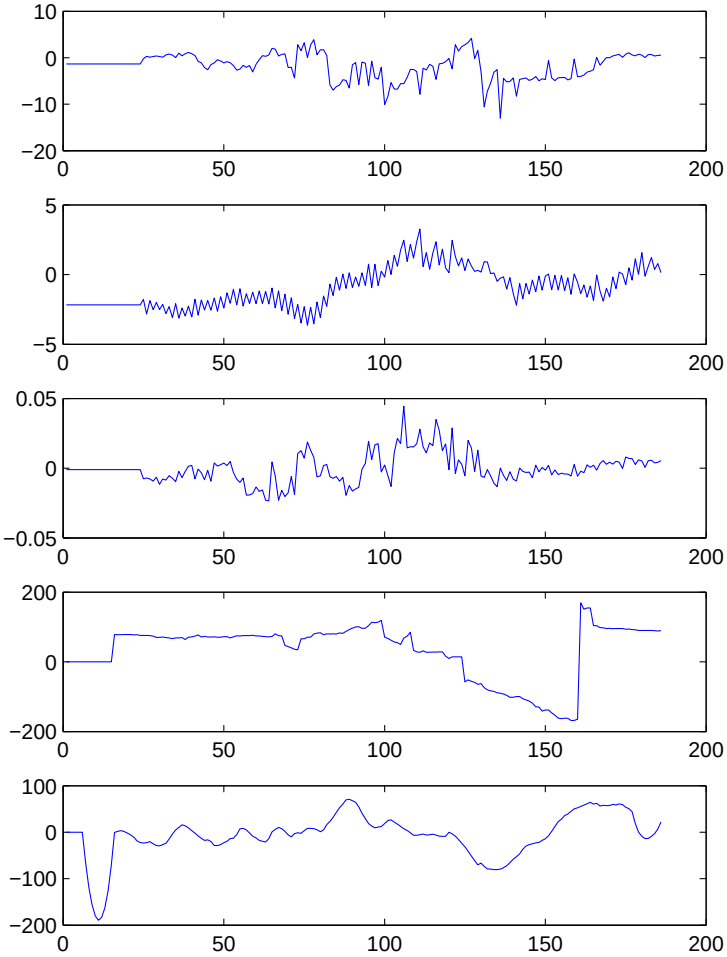
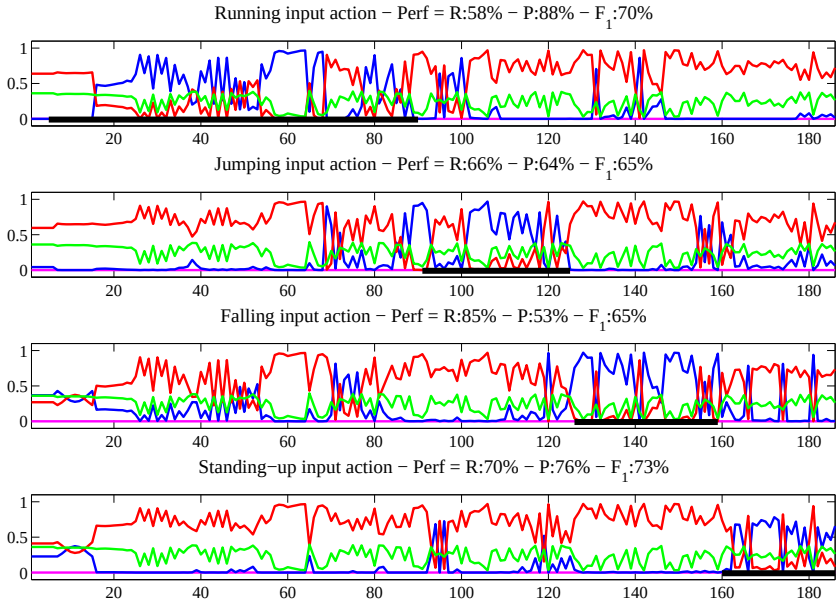
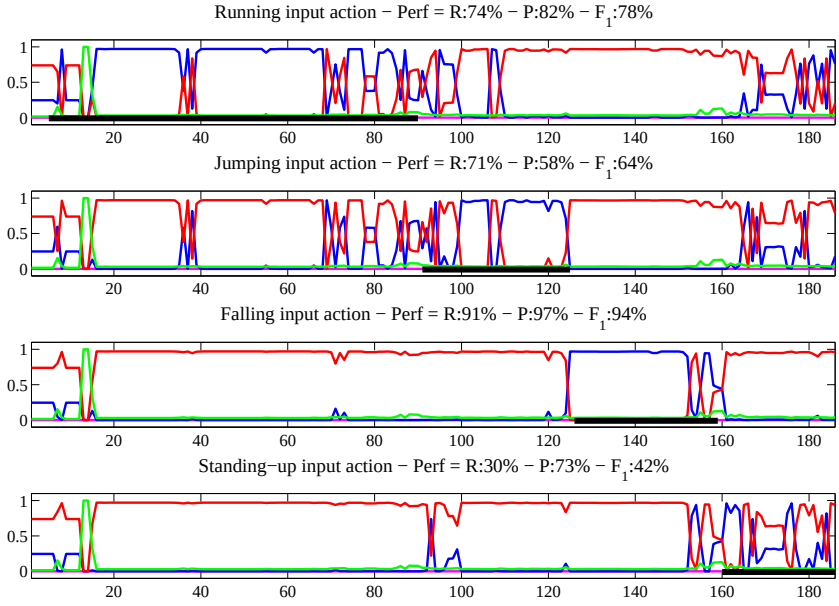


Figure 8: Features observed on the video sequence and used to compute beliefs of Fig. 9. From top to bottom: horizontal motion (P_{hm}, in pixels by image), vertical motion (P_{vm}, in pixel by image), zoom (P_{div}), angle (P_{swing}, in degree) and vertical variation of center of gravity (P_{vcg}, in pixel by image).



(a) MLGBT



(b) EDC

Figure 9: Beliefs obtained by the model of likelihood (MLGBT) and the model of distance (EDC) from features observed on the current video (Fig. 8). As on Fig. 4 the blue, green and red curves are respectively the evolution of belief on T_k^t (i.e. action A_k is true), on F_k^t (i.e. action A_k is false) and on $T_k^t \cup F_k^t$ (i.e. action A_k is true or false). Bold and black lines on the time axis represent ground truth for this video. The symbols R, P and F_1 are recall, precision and F_1-measure for the detection.

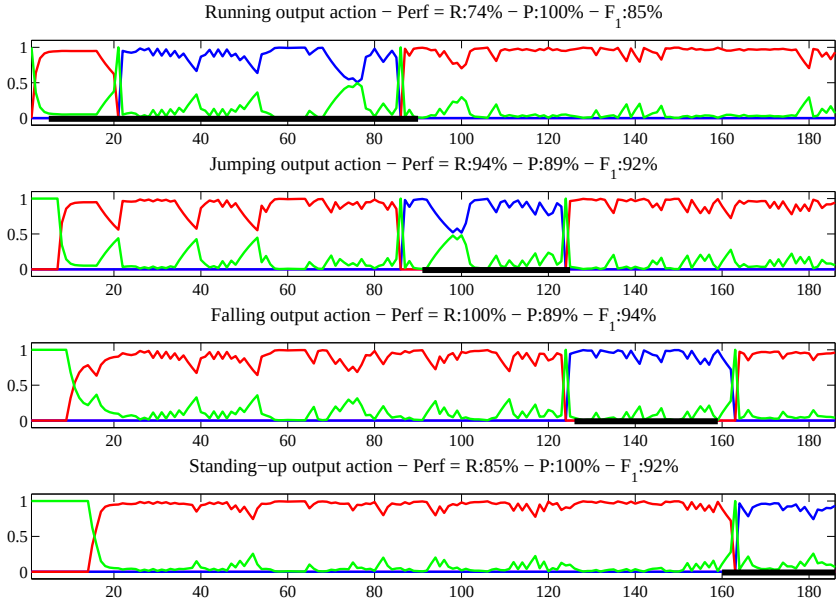
correct images obtained by expert annotations, R is the set of retrieved images provided by the recognition module using the BetP-based criterion, and $C \cap R$ is the number of correctly retrieved images. In order to assess the method by only one criterion, the $\mathcal{F}_1$-*measure* defined as $\mathcal{F}_1 = \frac{2 \times \mathcal{R} \times \mathcal{P}}{\mathcal{R}+\mathcal{P}}$ combines $\mathcal{R}$ and $\mathcal{P}$.

Fig. 8 provides the noisy features measured on the video sequence and from which beliefs are computed. Action detection (Fig. 9), scheduling (Fig. 10) and the GQ evolution (Fig. 11) are illustrated for a high jump video using MLGBT (top figures) and EDC (bottom figures). On Figures 9 and 10, blue curves, red curves and green curves represent respectively the evolution of the beliefs $m^{\Omega_k^t}(T_k^t)$ (action A_k is true), $m^{\Omega_k^t}(F_k^t)$ (action A_k is false) and $m^{\Omega_k^t}(T_k^t \cup F_k^t)$ (action A_k is true or false) all along time.

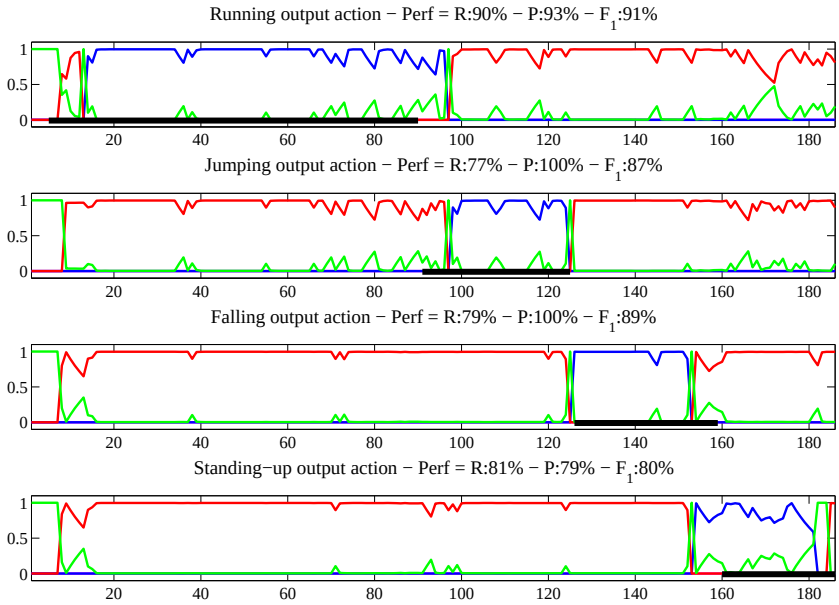
5.3. Illustration of the belief scheduler

Let us consider that the beliefs for each action are provided by the *model of distance* (EDC) [30]. An example of beliefs is depicted in Fig. 9 (before scheduling) and in Fig. 10 (after scheduling). One can clearly see the differences between both modeling methods: MLGBT provides much more noisy observations but the transitions are quite gradual while EDC provides less noisy observations but the transitions are much more abrupt. The ground truth is represented as a bold black line on the time axis. The goal of the BS is to filter these beliefs, separate actions and recognize activities. The scheduler and the filter make these beliefs smoother and ensure good recognition performance ($GQ = 74\%$).

In order to analyze the scheduler behavior, let us consider two consecutive actions, e.g. *running* and *jumping*, that correspond to the first two lines of



(a) MLGBT



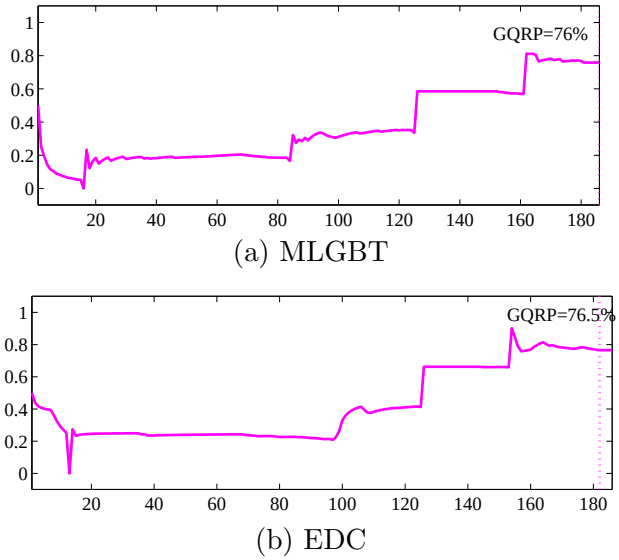
(b) EDC

Figure 10: Beliefs of Fig. 9 after filtering and scheduling. Meaning of color is the same as in Fig. 9. Note that the beliefs on T_k^t (action A_k is true) are generally well detected when compared with the ground truth (bold black lines on the time axis).

figures 9 (input) and 10 (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 uses the model $\mathcal{F}$ for each of the other three actions (natural or constrained false state). Then at time $t \approx 100$, *running* becomes false and *forces jumping* action to become 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 $t \approx 130$, the *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. 11 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 $\approx 75\%$.

We recall that MLGBT stands for "Model of Likelihood based on Generalized Bayesian Theorem", EDC stands for "Evidential Distance-based Classifier" and BS stands for "Belief Scheduler". In the sequel, we present action detection performance using: a1) MLGBT modeling alone, a2) MLGBT modeling coupled with BS, b1) EDC modeling alone and b2) EDC modeling coupled with BS. Tests a1) and a2) enable MLGBT to be compared with and without BS (Section 5.4), tests b1) and b2) enable EDC to be compared with and without BS (Section 5.5), tests a2) and b2) enable BS performance to be quantified with two different modelings (Section 5.6). Three sets of tables are then presented:

- The first set of 4-by-3 tables where four rows concern one type of



(a) MLGBT

(b) EDC

Figure 11: Global quality recognition performance criterion during scheduling.

jump and the three tables represent respectively EDC performance, EDC+BS performance and difference between both. Thus, there is one set of three tables for each jump and each table presents action detection performance (Section 5.4, Tables 2, 3, 4 and 5).

- The second set of 4-by-3 tables is similar to the previous but concerns the MLGBT (Section 5.5, Tables 6, 7, 8 and 9).

- The last set of four tables compares MLGBT+BS and EDC+BS performances with one table for each jump (Section 5.6, Tab. 10). 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 2, 3, 4 and 5 present recall, precision and F_1-measure of action detection in each activity using MLGBT before (Tables (a)) and after (Tables (b)) scheduling using the BS.

	R	P	F_1
Running	0.6246	0.7904	0.6978
Jumping	0.3989	0.7969	0.5317
Falling	0.3066	0.9601	0.4648
Standing-up	0.1657	0.8092	0.2750

(a) MLGBT

R	P	F_1
0.7919	0.8393	0.8149
0.5115	0.8231	0.6310
0.4064	0.8920	0.5584
0.3014	0.7283	0.4264

(b) MLGBT+BS

$\Delta(F_1)$
+0.1171
+0.0993
+0.0936
+0.1513

(c)Diff.

Table 2: Recall (R), precision (P) and F_1-measure for four actions in **high jumps** with *(a)* MLGBT without BS and *(b)* MLGBT with BS. Table *(c)* is the difference of detection (for the F_1-measure only) with and without the scheduler.

The BS performance is demonstrated on this dataset by greatly improving the detection in all jumps. The differences before and after applying the BS are explicitly given in Tables (c): if a difference is positive then it means that the belief scheduler improves the criterion.

	R	P	F_1
Running	0.5968	0.9211	0.7243
Jumping	0.4336	0.8756	0.5800
Falling	0.3636	0.8999	0.5179
Standing-up	0.2321	0.9043	0.3693

(a)MLGBT

R	P	F_1
0.8652	0.8481	0.8566
0.5271	0.9801	0.6856
0.4700	0.8322	0.6007
0.3452	0.8860	0.4968

(b)MLGBT+BS

$\Delta(F_1)$
+0.1322
+0.1056
+0.0828
+0.1275

(c)Diff.

Table 3: 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 of detection (for the F_1-measure only) with and without the scheduler.

The results illustrate in particular that the BS generally increases the recall rate (R) through filtering because of the stop threshold $\mathcal{T}_s$ that fills

"gaps".

	R	P	F_1	R	P	F_1	$\Delta(F_1)$
Running	0.5490	0.8851	0.6777	0.6961	0.8396	0.7611	**+0.0834**
Jumping	0.1556	0.9619	0.2678	0.4288	0.8619	0.5728	**+0.0938**
Falling	0.2214	0.9775	0.3610	0.4375	0.8684	0.5861	**+0.1251**
Standing-up	0.2421	0.9623	0.3868	0.3709	0.9393	0.5318	**+0.1450**
	(a)MLGBT			*(b)MLGBT+BS*			*(c)Diff.*

Table 4: 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 of detection (for the F_1-measure only) with and without the scheduler.

	R	P	F_1	R	P	F_1	$\Delta(F_1)$
Running	0.3917	0.9165	0.5488	0.5122	0.7272	0.6010	**+0.0522**
Jumping	0.3212	0.8476	0.4658	0.3490	0.7694	0.4801	**+0.0143**
Falling	0.3569	0.8339	0.4956	0.3945	0.7486	0.5167	**+0.0210**
Standing-up	0.2058	0.9350	0.3373	0.3404	0.8576	0.4873	**+0.1500**
	(a)MLGBT			*(b)MLGBT+BS*			*(c)Diff.*

Table 5: Recall (R), precision (P) and F_1-measure for four actions in **triple jumps** with *(a)* MLGBT without scheduler and *(b)* MLGBT with scheduler. Table *(c)* is the difference of detection (for the F_1-measure only) with and without the scheduler.

5.5. Results of scheduling with EDC modeling

The same study as previously was done using EDC modeling. Tables 6, 7, 8 and 9 present the performance of the detection of each action in each activity before (Tables (a)) and after (Tables (b)) scheduling based on EDC modeling.

The differences of performance of EDC and EDC+BS are given explicitly in Tables (c). This latter table shows that the BS greatly improves the results of action detection. Improvements seem to be a slight less marked than with MLGBT modeling, but this is due to a globally better performance

	R	P	F_1
Running	0.7207	0.8653	0.7864
Jumping	0.4857	0.7795	0.5985
Falling	0.4877	0.7911	0.6034
Standing-up	0.2606	0.7110	0.3814

(a)EDC

R	P	F_1
0.7277	0.9760	0.8338
0.5706	0.8297	0.6762
0.6218	0.7911	0.6963
0.4294	0.8613	0.5731

(b)EDC+BS

$\Delta(F_1)$
+0.0474
+0.0777
+0.0929
+0.1917

(c)Diff.

Table 6: 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 of detection (for the F_1-measure only) with and without the scheduler.

	R	P	F_1
Running	0.5886	0.9112	0.7152
Jumping	0.4699	0.8211	0.5977
Falling	0.4337	0.8868	0.5825
Standing-up	0.3519	0.6322	0.4522

(a)EDC

R	P	F_1
0.6580	0.9136	0.7650
0.5824	0.9264	0.7152
0.4519	0.9636	0.6152
0.3907	0.6843	0.4974

(b)EDC+BS

$\Delta(F_1)$
+0.0498
+0.1175
+0.0327
+0.0452

(c)Diff.

Table 7: 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 of detection (for the F_1-measure only) with and without the scheduler.

	R	P	F_1
Running	0.5912	0.9017	0.7142
Jumping	0.4037	0.6548	0.4994
Falling	0.4560	0.7480	0.5665
Standing-up	0.3490	0.7525	0.4769

(a)EDC

R	P	F_1
0.5990	0.8330	0.6969
0.4510	0.7329	0.5583
0.5040	0.7769	0.6113
0.3963	0.7295	0.5136

(b)EDC+BS

$\Delta(F_1)$
−0.0173
+0.059
+0.0448
+0.0367

(c)Diff.

Table 8: 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 of detection (for the F_1-measure only) with and without the scheduler.

of EDC modeling compared to MLGBT modeling. Indeed, the detection performance of EDC+BS compared to MLGBT+BS is in favor of the former except for some *running* actions. *Running* action is better detected with MLGBT because this action is much more highly represented in the learning set since it generally takes about 50% of each jump.

	R	P	F_1
Running	0.4253	0.9376	0.5851
Jumping	0.3532	0.6542	0.4587
Falling	0.4371	0.7523	0.5529
Standing-up	0.3098	0.9133	0.4626

(a)EDC

R	P	F_1
0.4778	0.9332	0.6320
0.4875	0.6783	0.5672
0.5251	0.8282	0.6427
0.3862	0.8204	0.5251

(b)EDC+BS

$\Delta(F_1)$
+0.0468
+0.1085
+0.0898
+0.0624

(c)Diff.

Table 9: 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 of detection (for the F_1-measure only) with and without the scheduler.

5.6. Comparison between EDC and MLGBT modeling with scheduling

Table 10 presents the differences of performance of the detection of each action (action names are not recalled for better readability) in each activity after scheduling between both EDC and MLGBT modeling. When the difference is positive, EDC+BS detection *is better* than MLGBT+BS.

It can be observed that better results are obtained with EDC which is a method that directly computes belief functions (whereas, in MLGBT, beliefs are indirectly computed using a transformation of likelihoods into beliefs using the GBT). The difference is highly significant for high jumps and triple jumps but less significant for long jumps and pole vaults. In the last two types of jumps, *running* action is better detected with MLGBT because, on the one hand, it is much more highly represented in the learning set and, in the

R	P	F_1
+0.0712	−0.0259	**+0.0583**
+0.0590	+0.0066	**+0.0452**
+0.2154	−0.1009	**+0.1379**
+0.1280	+0.1329	**+0.1467**

(a)High jumps

R	P	F_1
−0.2073	+0.0655	**−0.0916**
+0.0553	−0.0538	**+0.0296**
−0.0181	+0.1314	**+0.0145**
+0.0455	−0.2017	**+0.0006**

(b)Pole vaults

R	P	F_1
−0.0971	−0.0065	**−0.0643**
+0.0222	−0.1090	**−0.0099**
+0.0665	−0.0915	**+0.0252**
+0.0254	−0.2098	**−0.0182**

(c)Long jumps

R	P	F_1
−0.0344	+0.2060	**+0.0310**
+0.0710	−0.0912	**+0.0871**
+0.1306	+0.0796	**+0.1260**
+0.0158	−0.0372	**+0.0378**

(d)Triple jumps

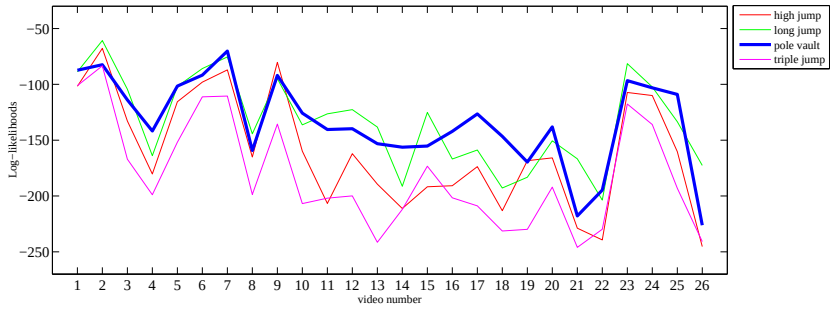
Table 10: Differences of detection between EDC+BS and MLGBT+BS for the four actions (one line per table) in each jump (one table per jump).

other hand, MLGBT is a probabilistic method, thus sensitive to frequent patterns.

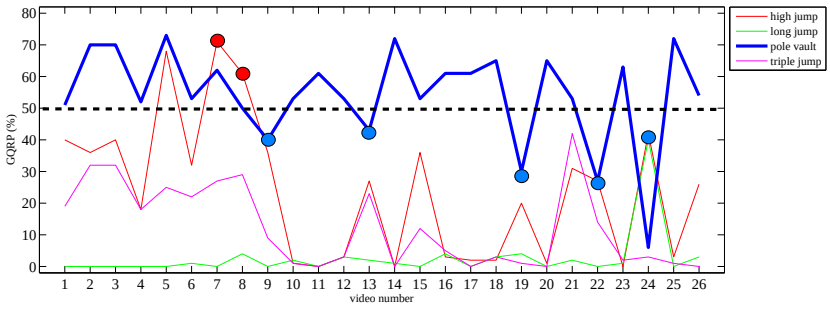
5.7. Comparing the Belief Scheduler and Hidden Markov Model for classification

As previously, four models of activities are built (for high jumps, long jumps, pole vaults and triple jumps). Previously, transition matrix and observation mixtures of Hidden Markov Model (HMM) were learned using the BNT toolbox [37]. Each state was modeled by a mixture of Gaussians (using settings in Section 3.2).

For the comparison, we used the same mixtures of Gaussians for both systems (and only MLGBT modeling). Likelihoods provided by the mixtures of Gaussians are transformed into belief functions using the Generalized Bayesian Theorem (Eq. 1). To assess both systems, we used the Viterbi



(a) Log-likelihoods in HMM.



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

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

algorithm for HMM [9] and the GQ criterion for the Belief Scheduler. The Viterbi algorithm was applied given each model of jumps providing four log-likelihoods, one for each sequence retrieved. The video was classified as a particular jump if the log-likelihood of this jump is the highest one. For the same video, we applied the Belief Scheduler and we chose the model that maximizes the Global Quality recognition performance criterion.

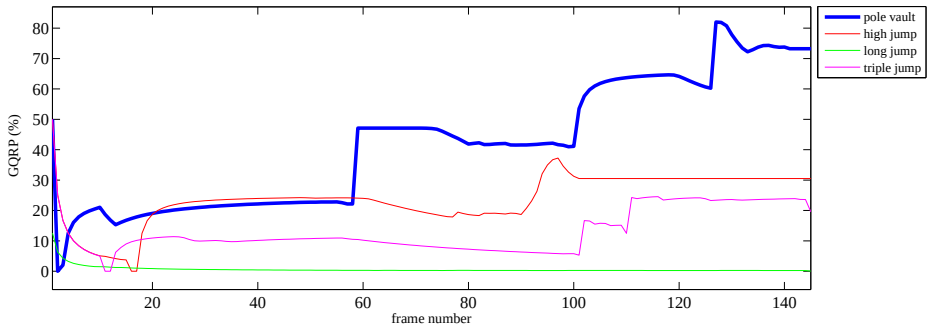


Figure 13: Evolution of a Global Quality recognition performance criterion over time in the Belief Scheduler for a pole vault video sequence (2nd video of Fig. 12). The bold curve represents the criterion evolution for the polevault model while the three other curves are highjump, triplejump and longjump.

The results are gathered in the confusion matrices of Tab. 11. The superiority 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 HMM. Bad results of HMM are explained by at least two factors: first, there is no class for rejections thus the decision is made without any other alternative. Secondly, there is a sensitivity to action and sequence length in the computation of log-likelihoods [38]. This sensitivity is represented in triple jump recognition. Indeed, *running* action in triple jumps is very long (generally two or

three times more than other videos) while other actions are very short (less than 10 frames). These differences make the state change difficult with the Viterbi decoder.

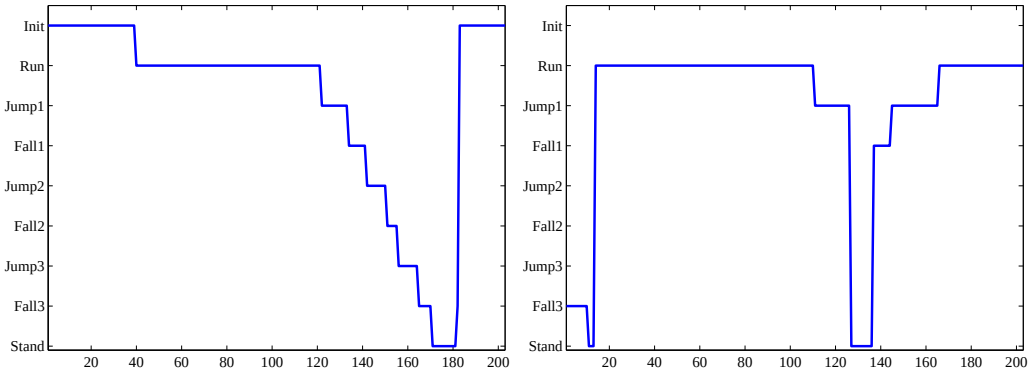
	Ground truth			
	pv	lj	tj	hj
pv	**19**	1	0	0
lj	0	**9**	0	0
tj	0	0	**9**	0
hj	2	2	0	**12**
rej	5	4	3	3

	Ground truth			
	pv	lj	tj	hj
pv	**14**	5	1	0
lj	11	**9**	0	9
tj	0	1	**9**	0
hj	1	1	2	**6**

Table 11: Classification results. Left: Belief Scheduler classification using the Global Quality recognition performance. Right: HMM classification using log-likelihood. Legends: *pv*, *lj*, *tj*, *hj* and "*rej*" stand for *pole vault*, *long jump*, *triple jump*, *high jump* and *class of rejections* respectively.

Fig. 12 presents the evolution of log-likelihoods for HMM and of the GQ criterion for the Belief Scheduler for 26 pole vaults videos analyzed by the four models (high jump, pole vault, triple jump and long jump). The GQ criterion (Fig. 12(b)) provides a more reliable decision than HMM's log-likelihoods (Fig. 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 Fig. 12(b) represents the threshold on quality (50%) which was used for adaptation (class of rejections). Big blue points in Fig. 12(b) represent rejection cases, whereas big red points concern recognition errors (decide high jumps instead of pole vaults). Interestingly, the system indicates that a specific model must be learnt for videos 9 and 10 (which were acquired with a distant view making the recognition difficult) and for video 19, 22 and 24 (for which the athlete motion was perpendicular to the image plane making

again the recognition difficult).



(a) States by Belief Scheduler. (b) States by HMM Viterbi.

Figure 14: Recognition of a triple jump (203 images) with some action discovery.

Fig. 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 useful 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). Fig. 14 describes results of action and activitie recognition for a triple jump described by eight states using the Viterbi decoder and the Belief Scheduler. Interpretation is clearly much easier using the latter.

6. Conclusion and future work

The generic architecture for sequence recognition applied to human motion analysis tested on real athletics videos shows the performance of the higher level part called Belief State Scheduler (BS) which carries out action (state) and activity (sequence) recognition. The BS finite state machine

made up of a Temporal Evidential Filter (TEF) and a set of constraints concerning state evolution makes the recognition of state sequence from noisy temporal belief functions possible. It generates both filtered belief functions and an inference criterion used for sequence classification. **The originality of this work lies in the proposal of a method for discrete state sequence recognition in the Transferable Belief Model framework and the early approach to focus on the current and next actions.** Compared to previous works, in particular we proposed a classification criterion based on conflict that appears between beliefs which are measured on the system and beliefs generated by a model of evolution. The chosen model of evolution is simple and consists of discounting of past beliefs. The number of thresholds is seven but five of them can easily be set heuristically while the other two are more sensitive and require cross-validation to assess their value.

The experiments on a first real dataset have shown good performance of human motion analysis architecture and in particular of the BS used for the detection of actions and the recognition of activities. This performance is obtained without adding explicit duration of actions or activities. We have proposed a thorough comparison of two modeling methods that generate beliefs from features: the Generalized Bayesian Theorem coupled with likelihood and the distance model. The latter seems better suited to 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 then transformed into a belief function. The comparison of the BS with probabilistic HMM proved the efficiency of the

approach proposed. This approach has also shown limitations in detecting actions in triple jumps. Actually, triple jumps are generally the longest and, above all, the noisiest activities (due to the poor quality of the videos coming from analogical TV). The challenge was thus to detect (in noisy data) *jumping* and *falling* actions which are very short (less than 10 frames, compared to more than 160 frames for *running*). Therefore, when action durations are short and, at the same time, beliefs contain a lot of noise then it is difficult to set the Belief Scheduler parameters in order to extract the sequence.

Some improvement can be made for instance by using the caution rules of combination instead of conjunctive rules as in equation 4. Experiments have also emphasized that the inference criterion of the BS can be used to create a *class of rejections*. This can improve the classification results but, above all, can point out new sequences. Since the classification criterion is bounded between 0% and 100%, it can be easily thresholded to create a class of rejections. This class is a first step toward adaptation since it gathers the cases for which the system of recognition could not take a decision. Work is under progress to pursue *pattern discovery* and *adaptation* which are promising in many applications.

References

[1] W. Hu, T. Tan, L. Wang, S. Maybank, A survey on visual surveillance of object motion and behaviors, IEEE Trans. on Systems, man and cybernetics C 34 (2004) 334–352.

[2] K. Messer, W. Christmas, E. Jaser, J. Kittler, B. Levienaise-Obadia,

D. Koubaroulis, A unified approach to the generation of semantic cues for sports video annotation, Signal Processing 85 (2005) 357–383.

[3] I. Ozer, W. Wolf, A hierarchical human detection system in uncompressed domains, IEEE Trans. on multimedia 4 (2002) 283–300.

[4] A. Jaimes, N. Sebe, Multimodal human computer interaction: A survey, in: IEEE Int. Workshop on Human Computer Interaction in conjunction with ICCV, Vol. 3766, Beijing, China, 2005, pp. 1–15.

[5] R. Green, L. Guan, Quantifying and recognizing human movement patterns from monocular video images - part II: Applications to biometrics, IEEE Trans. on Circuits and Systems for Video Technology 14 (2004) 191–198.

[6] R. Munoz-Salinas, R. Medina-Carnicer, F. Madrid-Cuevas, A. Carmona-Poyato, Multi-camera people tracking using evidential filters, International Journal of Approximate Reasoning 50 (2009) 732–749.

[7] T. Moeslund, A. Hilton, V. Kruger, A survey of advances in vision-based human motion capture and analysis, Computer Vision and Image Understanding 104 (2006) 90–126.

[8] M. Shah, Understanding human behavior from motion imagery, Machine Vision and Applications 14 (2003) 210–214.

[9] L. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. of the IEEE 77 (1989) 257–285.

[10] G. Klir, M. Wierman, Uncertainty-based information. Elements of generalized information theory, 2nd edition, Studies in fuzzyness and soft computing, Physica-Verlag, 1999.

[11] G. Shafer, A mathematical theory of Evidence, Princeton University Press, Princeton, NJ, 1976.

[12] P. Smets, R. Kennes, The Transferable Belief Model, Artificial Intelligence 66 (1994) 191–234.

[13] P. Smets, Advances in the Dempster-Shafer Theory of Evidence - What is Dempster-Shafer's model ?, Wiley, 1994, pp. 5–34.

[14] T. Denoeux, P. Smets, Classification using belief functions: The relationship between the case-based and model-based approaches, IEEE Trans. on Systems, Man and Cybernetics 36.

[15] T. Denoeux, M. Masson, EVCLUS: evidential clustering of proximity data, IEEE Trans. Systems, Man and Cybernetics B 34 (2004) 95–109.

[16] B. Quost, T. Denoeux, M. Masson, Pairwise classifier combination using belief functions, Pattern Recognition Letters 28 (2007) 644–653.

[17] M. Masson, T. Denoeux, ECM: An evidential version of the fuzzy C-means algorithm, Pattern Recognition 41 (2008) 1384–1397.

[18] Z. Hammal, A. Caplier, M. Rombaut, Belief theory applied to facial expressions classification, in: Int. Conf. on Advances in Pattern Recognition, Bath, United Kingdom, 2005.

[19] Z. Hammal, L. Couvreur, A. Caplier, M. Rombaut, Facial expression classification: An approach based on the fusion of facial deformations using the transferable belief model, International Journal of Approximate Reasoning (IJAR) 46 (3) (2007) 542–567.

[20] V. Girondel, A. Caplier, L. Bonnaud, M. Rombaut, Belief theory-based classifiers comparison for static human body postures recognition in video, Int. Jour. of Signal Processing 2 (2005) 29–33.

[21] M. Rombaut, I. Jarkass, T. Denoeux, State recognition in discrete dynamical systems using Petri nets and Evidence theory, in: Europ. Conf. on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 1999, pp. 352–361.

[22] P. Smets, Beliefs functions: The Disjunctive Rule of Combination and the Generalized Bayesian Theorem, Int. Jour. of Approximate Reasoning 9 (1993) 1–35.

[23] M. Mohamed, P. Gader, Generalized hidden Markov models - part i: Theoretical frameworks, IEEE Trans. on Fuzzy Systems 8 (2000) 67–81.

[24] P. Smets, B. Ristic, Kalman filters for tracking and classification and the Transferable Belief Model, in: Int. Conf. on Information Fusion, 2004.

[25] E. Ramasso, D. Pellerin, M. Rombaut, Belief Scheduling for the recognition of human action sequence, in: Int. Conf. on Information Fusion, Florence, Italia, 2006.

[26] E. Ramasso, M. Rombaut, D. Pellerin, State filtering and change detection using TBM conflict - application to human action recognition in

athletics videos, IEEE Trans. on Circuits and Systems for Video Technology 17 (7) (2007) 944–949.

[27] J. Odobez, P. Bouthemy, Robust multiresolution estimation of parametric motion models, Jour. of Visual Communication and Image Representation 6 (1995) 348–365.

[28] A. Appriou, Probabilités et incertitudes en fusion de données multisenseurs, Revue Scientifique et Technique de la Défense 11 (1991) 27,40.

[29] F. Delmotte, P. Smets, Target identification based on the Transferable Belief Model interpretation of Dempster-Shafer model, IEEE Trans. on Systems, Man and Cybernetics 34 (2004) 457–471.

[30] T. Denoeux, A k-nearest neighbor classification rule based on Dempster-Shafer theory, IEEE Trans. on Systems, Man and Cybernetics 5 (1995) 804–813.

[31] L. Zouhal, T. Denoeux, An evidence-theoretic K-NN rule with parameter optimization, IEEE Trans. on Systems, Man and Cybernetics 28 (1998) 262–271.

[32] M. Figueiredo, A. Jain, Unsupervised learning of finite mixture models, IEEE Trans. on Pattern Analysis and Machine Intelligence 24 (2002) .

[33] P. Smets, Analyzing the combination of conflicting belief functions, Information Fusion 8 (2005) 387–412.

[34] E. Lefèvre, O. Colot, P. Vannoorenberghe, Belief function combination and conflict management, Information Fusion 3 (2002) 149–162.

[35] P. Smets, Decision making in the TBM: The necessity of the pignistic transformation, Int. Jour. of Approximate Reasoning 38 (2005) 133–147.

[36] J. Makhoul, F. Kubala, R. Schwartz, R. Weischedel, Performances measures for information extraction, in: Proc. of DARPA Broadcast News Workshop, 1999.

[37] K. P. Murphy, Dynamic Bayesian networks: Representation, inference and learning, Ph.D. thesis, UC Berkeley (CSD) (2002).

[38] K. Yamazaki, On the likelihood function of hmms for a long data sequence, in: IEEE Int. Conf. on Machine Learning and Signal Processing, 2009, to appear.