Recognizing and localizing individual activities through graph matching

Anh-Phuong Ta Christian Wolf Guillaume Lavoué Atilla Baskurt
Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69621, France
{anh-phuong.ta,christian.wolf,glavoue,atilla.baskurt}@liris.cnrs.fr

Abstract

In this paper we tackle the problem of detecting individual human actions in video sequences. While the most successful methods are based on local features, which proved that they can deal with changes in background, scale and illumination, most existing methods have two main shortcomings: first, they are mainly based on the individual power of spatio-temporal interest points (STIP), and therefore ignore the spatio-temporal relationships between them. Second, these methods mainly focus on direct classification techniques to classify the human activities, as opposed to detection and localization. In order to overcome these limitations, we propose a new approach, which is based on a graph matching algorithm for activity recognition. In contrast to most previous methods which classify entire video sequences, we design a video matching method from two sets of ST-points for human activity recognition. First, points are extracted, and a hyper graphs are constructed from them, i.e. graphs with edges involving more than 2 nodes (3 in our case). The activity recognition problem is then transformed into a problem of finding instances of model graphs in the scene graph. By matching local features instead of classifying entire sequences, our method is able to detect multiple different activities which occur simultaneously in a video sequence. Experiments on two standard datasets demonstrate that our method is comparable to the existing techniques on classification, and that it can, additionally, detect and localize activities.

1. Introduction

Human action recognition has been an active research area in recent years due to its wide number of applications which include video-surveillance but also annotation and retrieval, human computer interaction etc. At this time, building a robust activity recognition system still remains a very challenging task, because of the variations in actions classes, different possible viewpoints, as well as illumination changes, moving cameras, complex dynamic backgrounds and occlusions.

Based on the features used for recognition, existing action recognition methods can be broadly divided into two categories: local approaches [4, 14, 16, 18] and holistic approaches [12, 23, 22] and some methods which do not neatly fall into these categories, e.g. Sun et al. [20] combine local and holistic features. Most of the holistic-based approaches rely on pre-processing of input data such as background subtraction or tracking. The local-based approaches overcome some limitations by exploiting robust descriptors extracted from interest points. Most of these methods are based on bag-of-words models (BoW), which have been very successful for text analysis, information retrieval and image classification. Inspired by this, a number of works have shown very good results for human action recognition [4, 14, 16]. However, they discard the spatio-temporal layout of the local features which may be almost as important as the features themselves.

To overcome the limitations of the BoW models, efforts have been made to exploit information from the spatial and temporal distribution of interest points [13, 26]. These extensions, however, still suffer from some of the inherent problems involved in classification: they do not allow to localize activities, and they require selecting the optimal number of codewords for codebook formation as well as fine-tuning of parameters.

As a response, matching techniques have been introduced recently, e.g [9, 19, 15]. Shechtman and Irani [19] define a motion consistency measure to match space-time volumes directly. However, the distance between pair of videos is computed by exhaustively comparing patches extracted from every space-time point. Ke et al. [9] combine a part-based shape and flow matching framework from [19] for event detection in crowded videos. Recently, Ryoo and Aggarwal [15] have presented a histogram-based match kernel for video matching. Among the methods mentioned above, our approach is most closely related to the work of Ryoo and Aggarwal [15], who perform video matching from two sets of ST-points. Our method differs from their work in two main points. First, the authors in [15] define

a set of logical predicates for taking into account the pairwise relationships among points. Instead, we take into account higher-order relationships between points through a graph-matching technique. It should be noted that such logical predicates are difficult to extend to higher relationships. Second, their method needs to train a codebook from the training sets, while our method does not require any learning.

In this paper, we advocate for the advantages of graph matching methods and their capability of exploiting relationships between local primitives. The main contribution of our paper is the introduction of a graph-based matching method for detecting and localizing multiple actions in video sequences. Graph matching techniques have been studied intensively in the field of pattern recognition [2, 5, 11, 21, 25], but no method has yet been given for recognizing human activities — a straightforward application of these techniques to video recognition is difficult. It is widely known that exact (sub) graph matching is NP-complete [21], as is sub graph isomorphism [24]. Approximate solutions have been proposed for various applications. For instance, let N_1, N_2 be the number of vertices (nodes) in graphs G_1 and G_2 , respectively. Optimally assigning each node from G_1 to one of the nodes in G_2 is of complexity $O(N_1 \cdot N_2)$ if only the unary measurements (e.g., SIFT descriptors) associated with each node are used, i.e. for each node in G_1 we assign the node G_2 having minimum feature distance. However, this is highly suboptimal. If neighborhood relationships are taken into account, i.e., coherence of distances and/or angles associated to the edges in the graphs, the complex interactions between assignment variables make the complexity exponential: there are $N_1^{N_2}$ possible assignments over the whole set of nodes in G_1, where each assignment takes $O(N_1^2)$ to check. Although fast approximative algorithms do exist, e.g. with graph cuts [21], the problem remains very difficult.

As pointed out in [25], the vertex correspondance problem can be transformed into an edge correspondence problem. In this approach, each edge in graph G_1 is assigned an edge in graph G_2 according to a minimal distance which involves, both, the feature distances of the 4 involved nodes as well as edge compatibility, e.g. a comparison of the lengths of the model edge and the assigned edge. In [11], the resulting optimization problem is solved approximately with a spectral method, which relaxes the discrete assignemnt variables into continuous ones and then solves it numerically by removing some constraints during the optimization procedure itself. This principle can be naturally extended to higher order interactions. Zass and Shashua [25] present a hyper-graph[1] matching method, which is of complexity $O(|N_1|.|N_2|.z^{(2d-1)})$, where z is the closest hyper-edges per vertex and d is the order of hyper-edges considered (d=2 for pairs). This method is still very time consuming, because in a real application z could not be far from $min(|N_1|, |N_2|)$. Very recently, Duchenne et al. [5] generalized the spectral matching method from [11] by using a tensor-based algorithm for high order graph matching, which is of complexity $O(n^3 + n^d \log(n))$ where $n = max(|N_1|, |N_2|)$. A modified version of this triangle based algorithm (i.e., d=3) is the basis of our work on activity recognition in video sequences.

Exploiting the full potential of a graph based representation, our method offers several important advantages over other activity detection methods for videos:

- The proposed method can not only classify but also detect and localize activities, which occur simultaneously in the same video sequence[2].
- It does not require any parameter tuning, training, foreground/background segmentation, or any motion estimation or tracking.
- By verifying the spatio-temporal constraints in a significant way, our method needs only a small number of features points (i.e., the *important* ones) extracted from two given videos to perform matching. In constrast, the conventional BoW methods need to collect dense points from the videos to perform classification.

Besides these advantages, our method features several contributions compared to the original graph matching method introduced by Duchenne et al. [5] for object detection:

- By significantly reducing the number of hyper-edges in the graph, our method is of much lower complexity compared to the original one [5] (c.f section 3.1).
- We incorporate both triangle geometries and their orientations to find potential corresponding triangles (c.f section 3.2), which speeds up the convergence by eliminating incompatible triangles.
- We benefit from the features calculated at each ST-point to initialize the algorithm (c.f section 3.3), which reduces the number of false alarms and speeds up convergence.
- Most of methods on graph matching consider the score returned by the objective function as a detection criteria. However, this is not optimal, as one cannot distinguish the case in which several vertices in the scene graph are matched to the same vertex in the model graph. We propose to interpret the projection of the

[1] A hyper-graph is a generalization of a graph, where an edge can connect any number of vertices, and hyper-edge is an arbitrary number of nodes [25].

[2] Several methods can be adapted for detecting multiple actions by using sliding windows in both space and time dimensions, e.g. from [14, 16].

set of vertices of the first graph onto the second one to compute a second score, called the detected score, which, along with the matching score, is used for detection (c.f section 3.4).

The rest of this paper is organized as follows. After briefly summing up the Tensor-based algorithm in section 2, we introduce our adaptation and extension of this algorithm to video matching in section 3. In section 4, we discuss the computational complexity of the proposed method. The experimental results are presented in section 5. Finally we conclude and give some perspectives of this work.

2. Hyper-graph matching

In this section, we summarize the hyper-graph matching method introduced in [3] and [11] and refined in [5]. Let $G^m = (V^m, E^m, F^m)$ and $G^s = (V^s, E^s, F^s)$ be two hyper-graphs (the model and the scene graph, respectively) where hyper-edges correspond to a d-tuple of vertices. In our case, where d=3, E represents a set of triangles (our d-tuples), V a set of vertices, and F the set of their associated unary measurements (i.e an appearance feature). In the following we denote the number of nodes in both graphs as $N_1 = |V^m|$ and $N_2 = |V^s|$, respectively. A matching between G^m and G^s is equivalent to looking for an $N_1 \times N_2$ assignment matrix X such that X_{ij} is set to 1 when v_i^m is matched to v_j^s, and to 0 otherwise. Thus, the search space is the set X of assignment matrices:

$$X = \{X_{ij} \in \{0,1\} : \sum_i X_{ij} = 1\} \quad (1)$$

Note that we constrain each model node v_i^m to be matched to exactly one scene node v_j^s, but a scene node v_j^s may be matched to several model nodes. As in [5], the matching problem is formulated as the maximization of the following score on X:

$$score(X) = \sum_{i,i',j,j',k,k'} H_{i,i',j,j',k,k'} X_{i,i'} X_{j,j'} X_{k,k'} \quad (2)$$

where $H_{i,i',j,j',k,k'}$ is an energy potential estimating the compatibility of pairs of triplets $(i,j,k) \in V^m$ and $(i',j',k') \in V^s$. High values of H correspond to similar triplet pairs. Here, the product $X_{i,i'} X_{j,j'} X_{k,k'}$ will be equal to 1 if (i,j,k) are all matched to (i',j',k') (the original formulation of this problem in [3] and [11] involved pairs).

In [3, 11], the scoring function is maximized through a continuous optimization procedure which requires relaxing the values of X such that each element takes continuous values in the interval $[0,1]$ subject to the constraint that the norm of the column vectors of X is 1. Exploiting this constraint, and further requiring the elements of H to be non-negative, the maximum value of (2) can be calculated as the largest eigenvalue of X.

The basis of our method is the refined method proposed by Duchenne et al. [5], which improves upon [3] and [11] in three ways: (i) whereas the interactions in [3, 11] are pairs, [5] extends the order to triplets (triangles), which may boost the discriminative power of the method, especially since interactions between neighboring d-tuples are not taken into account in this class of hyper-graph techniques; (ii) the new organization into triplets in [5] allows to change the L_2 norm of the relaxation to the L_1 norm, which makes the de-relaxation of the continuous values into discrete assignement decisions more robust; The largest eigenvalue is calculated using an iterative power-iteration algorithm.

3. The proposed method

The main objective of our method is to measure the similarity of two videos through a graph-based matching technique. ST-interest points are first extracted from the video sequences, then the proximity graphs are constructed from them (see section 3.1). The activity recognition problem is now formulated as (sub) graph matching between a model graph and a (potentially larger) scene graph. We initialize the iterative matching process using appearance features associated with each node of the graphs (see section 3.3). We resort to the power-iteration algorithm introduced in [5] to maximize a new objective function designed for the application (see section 3.2). Finally, we propose a new way to verify the matching result in order to decide whether the scene graph contains an instance of the model graph, i.e., the two videos contain the same human activity (see section 3.4).

3.1. Constructing more expressive graphs

The complexity of the power-iteration method in [5] highly depends on the number of the d-tuples (triangles) in the two graphs constructed from the videos. The original algorithm constructed fully connected graphs, i.e. graphs with close to N_1^3 and N_2^3 nodes, respectively. We present hereafter a graph construction method producing graphs with far fewer but more expressive hyper-edges, which significantly reduces complexity and also increases robustness to non-rigid transformations. Given two sets of ST-points, we construct two corresponding graphs for the model video and the scene video, i.e. we construct the two sets E^m and E^s of hyper-edges (triangles). Without loss of generality, we present the construction of the model graph G^m, the scene graph G^s is constructed in a similar way.

Temporal aspects — One of the most important properties of video is the nature (and importance) of the temporal order, which is very often dominated by causal relationships. We exploit this in two ways: (i) we put a constraint on the preservation of the correct temporal order of pairs of triplets (see section 3.2) (ii) we restrict the number of hyper-

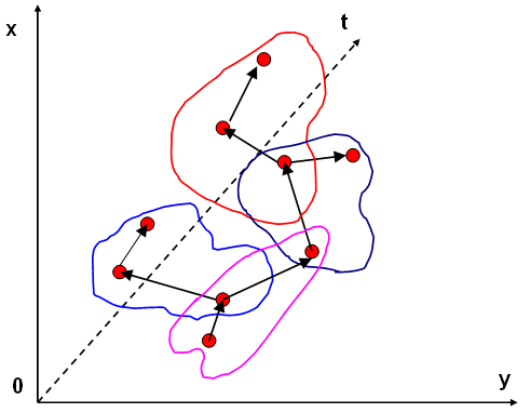


Figure 1. Illustration of a partial view of our graph: circles are ST-points; three close points are grouped to form a triangle; arrows indicate the temporal order of the points in a triangle.

edges, i.e., we keep only one hyper-edge per triplet of tree points. This first filter allows us to sample the number of triplets from n^3 to $|C_n^3|$.

ST-Proximity — Fully connected graphs create triangles between all possible triplets of points, even between very distant ST-points (in time and/or space). While, generally speaking, a higher number of triangles tends to increase the discriminative power of the matching method, it also tends to decrease robustness. This is especially true for triangles between very distant points, as the space-time geometrical transformation between two instances of the same human activity is not necessarily rigid. We propose to filter the set of triangles by a simple thresholding rule keeping only triangles which are close in space and time using two different thresholds, one for the spatial dimensions and one for the temporal dimension (see figure 1 for an illustration). The points in a triangle are then ordered according to the temporal order. If there are more than one point in the same frame, they are ordered according to their spatial coordinates. In practice, we fix the thresholds such that $|E^m| \leq 8 \times N_1$.

The scene graph G^s is constructed in a similar way, but taking more triplets (i.e., using bigger thresholds) to deal with noise and scale changes. This gives the sub graph matching algorithm a larger set of scene triangles to choose from for each model triangle. In practice, we choose the thresholds such that $|E^s| \leq 50 \times N_2$.

3.2. An objective function for video sequences

In order to boost the discriminative power of the matching algorithm, we propose the following compatibility matrix H for the objective function in equation (2) designed for activity recognition in videos:

$$H_{i,i',j,j',k,k'} = H^t_{i,i',j,j',k,k'} \times \phi((i,j,k),(i',j',k')) \quad (3)$$

where $\phi(.,.)$ and H^t govern temporal and space-time geometrical aspects of the transformation:

Temporal aspects — As mentioned in section 3.1, we think that it is crucial to exploit the important nature of the temporal dimension in video sequences. We therefore introduce the functional $\phi(.,.)$ which verifies if two triangles are equally oriented taking into account the (temporal) order of their points:

$$\phi((i,j,k),(i',j',k')) = \begin{cases} 1 & \text{if } D_{ijk} = D_{i'j'k'} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where D_{ijk} is the sign of the determinant of the triplet computed from their spatio-temporal coordinates:

$$D_{ijk} = \text{sign}\left(\begin{vmatrix} i_x & i_y & i_t \\ j_x & j_y & j_t \\ k_x & k_y & k_t \end{vmatrix} \right) \quad (5)$$

Note that the row order in the determinant must be the same as the point order in the triplet.

ST-geometrical distance — The factor H^t in (3) describes the geometric similarity between two triangles and is given as follows:

$$H^t_{ijki'j'k'} = \exp\left\{ -\frac{||\boldsymbol{\alpha}(i,j,k) - \boldsymbol{\alpha}(i',j',k')||}{\sigma_g} \right\} \quad (6)$$

where $||\cdot||$ is the L_2 norm and $\boldsymbol{\alpha}(\cdot,\cdot,\cdot)$ is a vector of the cosines of the first and second angles in the triangle (recall that the points of a triangle are ordered), as the third one is linearly dependent. The parameter σ_g governs the intra-class variations of the angles, we set it to half the mean over the distances between all triangles.

ST-geometrical proximity — To further decrease complexity, and to increase discriminative power, triangles in the model graph are only allowed to match triangles in the scene graph if their geometrical shapes (given above) are close enough. More precisely, only the first ranked distances as given by (6) are kept non-zero in H^t, where the ranking can be computed efficiently using a k-d tree. In practise we keep the first 350 scene triangles for each model triangle.

3.3. Adding space-time features

The proposed matching compatibility function H does not use any features associated with the ST-points — we argue that the matching algorithm itself is more robustly controlled through ST-geometry, as distances in space-time are difficult to compare with distances in most features spaces in a single energy function. However, we propose to benefit from the power of feature descriptors to better initialize the matching algorithm. In our experiments we chose the cuboid descriptor introduced by Dollar [4], which calculates a histogram of gradients at the position of each ST-point. Denoting by f_i^m and f_j^s the features calculated on

point i of the model graph and point j of the scene graph, respectively, we initialize the relaxed assignment matrix X as follows:

$$X_{ij} = \exp\left\{-\frac{||f_i^m - f_j^s||}{\sigma_f}\right\} \quad (7)$$

where the parameter σ_f captures the variation of distances in feature space. Only the values for $Nb(i)$ nearest neighbors in feature space (efficiently identified through a k-d tree) are kept non-zero in the matrix X. In practice, we set $Nb(i)$ to $0.35|N_2|$. Additionally, the required norm constraint described in section 2 is then enforced through normalization.

3.4. Action detection in videos

In order to decide whether an instance of the graph (a video in our case) has been detected or not, most energy based graph matching methods consider the final matching score, which is denoted by $score(X^*)$ and given as the maximum of equation (2), calculated as a sum of all matching scores between triangles after matching. This, however, is not an optimal choice. For instance due to noise, several triangles in G^s are matched to the same triangle in G^m, and we cannot remove such irrelevant matches by filtering based on only the matching score. Moreover, these matching scores depend on the geometric features of triangles used to verify compatibility between triangles, and therefore make the choice of a threshold difficult.

We propose a different score, called $score_d(X)$, which removes the uncertainties from the matrix X. More precisely, we de-relax the values of the optimal matrix X^* (obtained as the maximum of (2)) by taking the maximum for each node in the model graph and setting the others to zero:

$$Z_{ij} = \begin{cases} X_{ij} & \text{if } X_{ij} = \max_k X_{ik} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Then, $score_d$ is defined by the mean projection:

$$score_d(X) = \frac{1}{N_1}\sum_i Z(i,j) \quad (9)$$

Clearly, $score_d \in [0..1]$ and it is equal to 1 for the ideal case where every node in G^m is well matched. While the matching $score(X)$ indicates how similar two graphs are, $score_d(X)$ measures us the percentage of correctly matched points. We combine them through a global score as follows:

$$score_{Global}(X) = \begin{cases} score_d(X) & \text{if } score(X) \geq \tau_0 \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

where τ_0 is a preliminary threshold obtained through experiments.

4. Computational complexity and running time

Our system has been implemented entirely in Matlab and for the moment processes videos as entire sequences. However, the algorithm itself can be implemented to deal with video streams by processing small overlapping consecutive blocks. The size of these blocks can be kept small since the model activities are short — around 5 to 30 frames in our case, with a few longer examples of 60 frames in our model dataset.

Our matlab implementation does not run in realtime, as 1 second of video for the moment requires 46.7 seconds of processing on a single core processor with 2GHz and 2GB of RAM, including feature extraction, scene graph construction, and matching against 98 model graphs. However, given the notorios lack of efficiency of matlab, a reimplementation in C++ should provide real-time or near-realtime performance on a state of the art machine. We would also like to point out that the algorithm is inherently parallel since matching with different model graphs can be done in parallel. Furthermore, the matrix computations of the power-iteration method should run very efficiently on a GPU.

The theoretical complexity of the algorithm is given as follows for the different steps:

Graph construction (see section 3.1) Constructing the triangles of the fully connected graphs would need $O(N_1^3 + N_2^3)$ operations. Since we highly subsample the number of triangles, and since this is done in small blocks of space-time cubes, the real complexity is far below.

Computation of the compatibility matrix H (see section 3.2) Since the number of model and scene triangles is a linear function of the number of the respective vertices, and we also restrict the number of scene triangles per model triangle, the complexity can be given as $O(C_1 \cdot N_1 \cdot C_2 \cdot N_2 \log(N_2))$ where C_1 is the amount of triangles per node and C_2 is the number of nearest scene triangles per model triangle. This complxity involves a factor $O(N_2 \cdot \log(N_2))$ for the access to the k-d tree.

Initialization (see section 3.3) In a similar manner, the complexity of the initialization step is given as $O(N_1 \cdot C_3 \cdot N_2 \cdot \log(N_2))$ where $C_3 \leq N_2$ is the number of nearest neighbors searched for each node of the model graph.

Optimization As mentioned before, the power-iteration method introduced by [5] and described in section 2 is of complexity $O(N_M^3 + N_M^d \log(n))$ where $N_M = \max(|N_1|, |N_2|)$. However, this complexity is for the worst case of fully connected graphs. In our case the real complexity is far from this.

5. Experimental results

We evaluated the performance of our proposed algorithm with regards to two different tasks:

- The classification of entire video sequences according to the activity of a single activity performed by a single person in the video. Since standard databases are available for this kind of task, we are able to give quantitative performance figures (classification accuracy).

- Detection and localization of multiple activities performed by multiple people at different locations and time instants in the same video. Up to our knowledge, no standard database is available for this more difficult problem. We illustrate the performance of our method qualitatively.

5.1. Classification of entire sequences

Datasets — Our experiments are carried out on the standard KTH and Weizmann human action datasets. The KTH dataset was provided by Schuldt et al. [18] in 2004 and is the largest public human activity video dataset. It contains a total of 2391 sequences, comprising 6 types of actions (boxing, hand clapping, hand waving, jogging, running and walking) performed by 25 subjects in 4 different scenarios including indoor, outdoor, changes in clothing and variations in scale. Each video clip contains one subject performing a single action. The Weizmann dataset was first used in by Blank et al. [1] in 2005, which consists of 90 video clips of 10 actions (walking, running, jumping, gallop sideways, bending, one-hand-waving, two-handswaving, jumping in place, jumping jack, and skipping) performed by 9 different subjects. Again, each video clip contains one subject performing a single action.

Testing protocol — We took advantage of the ST-interest point detector developed by Dollar et al. [4] and their cuboid descriptor as the unary measurement associated to each STIPs. Using leave-one-out cross-validation, we apply our method for the classification of activities and report the average accuracies, even though the focus of our method is to detect/localize the activities. To this end, we employ a group of videos from a single subject in the dataset as the testing videos (i.e., scene graphs), and the remaining videos as the model videos (i.e., model graphs). This was repeated so that each group of videos in the dataset is used once as the testing videos. For each loop, we match each test video against all model videos, and take the label (i.e., the activity) of the model video which gives maximum score from equation (10).

Table 1 presents a comparison of our results with state-of-the-art results. The results indicate that our method is at least comparable with previous methods and outperforms most of them, while we do not require any prior information about the actions to be recognized.

Table 1. Comparison of our method with different methods, tested on KTH and Weizmann datasets.

Method	KTH	Weizmann
Our method	**91.2**	**100.0**
Schindler and Gool [17]	92.7	100.0
Fathi and Mori [6]	90.5	100.0
Gorelick et al. [8]	-	99.6
Sun et al. [20]	94.0	97.8
Niebles et al. [14]	83.3	90.0
Kim et al. [10]	95.3	-
Liu and Shah [13]	94.2	-
Ryoo and Aggarwal [15]	93.8	-
Zhang et al. [26]	91.3	-
Gilbert et al. [7]	89.9	-
Savarese et al. [16]	86.8	-
Dollar et al. [4]	81.2	-

5.2. Detection and localization of multiple and individual actions

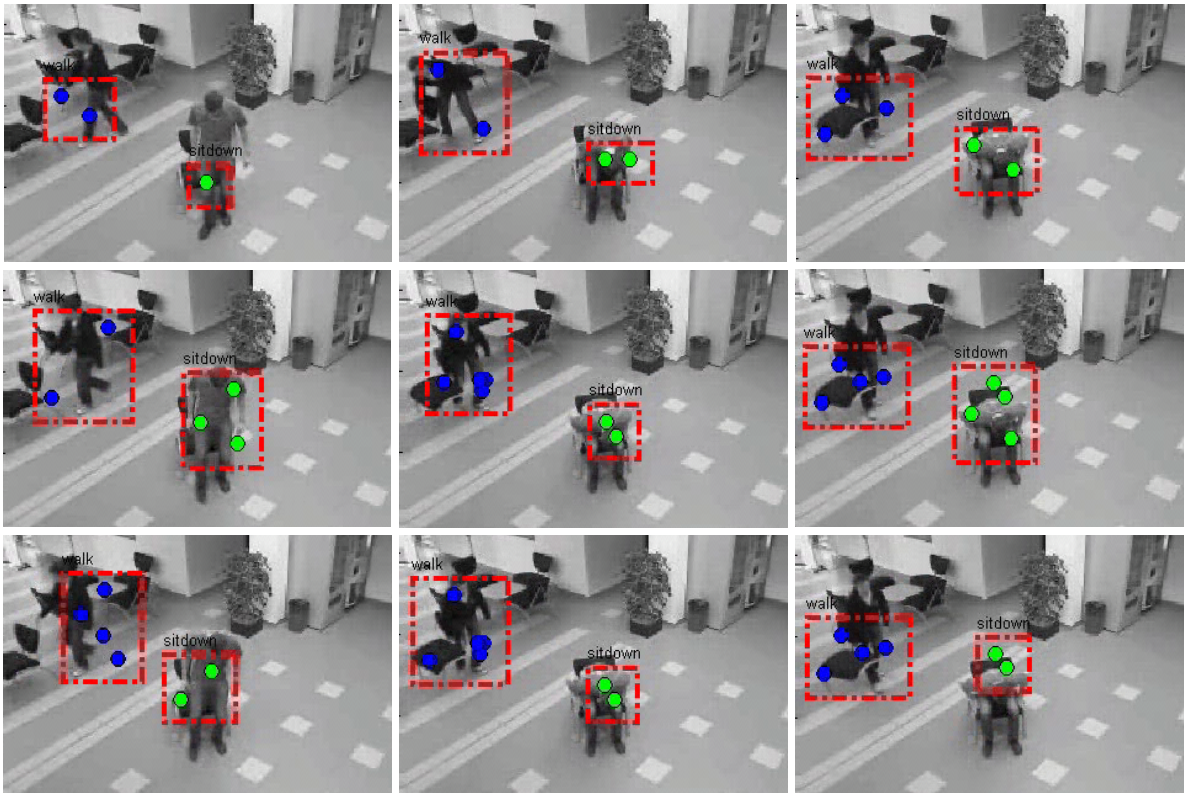
The main contribution of our method is the possibility of detecting and localizing multiple activities from unsegmented video sequences. Unfortunately, to the best of our knowledge, there is no standard dataset for evaluating such applications (i.e., the ones like the KTH and Weizmann datasets for activity recognition evaluation). This is due to the fact that this problem is still incompletely understood.[3] To evaluate our system, we have performed a third experiment on our own dataset, which contains 120 videos in 4 classes (run, walk, vertical jump, sitdown).[4] Our dataset is different from those of KTH and Weizmann in two points: i) different activities are performed in different directions with respect to the camera; and ii) we included videos as short as between 3 and 10 frames.

For each pair of videos, we first perform matching as described in section 5.1 and we localize the action by projecting the points in the model graph onto the scene graph based on the obtained solution. A threshold is used to exclude the unwanted matches. Finally, the detected action is localized around the detected points.

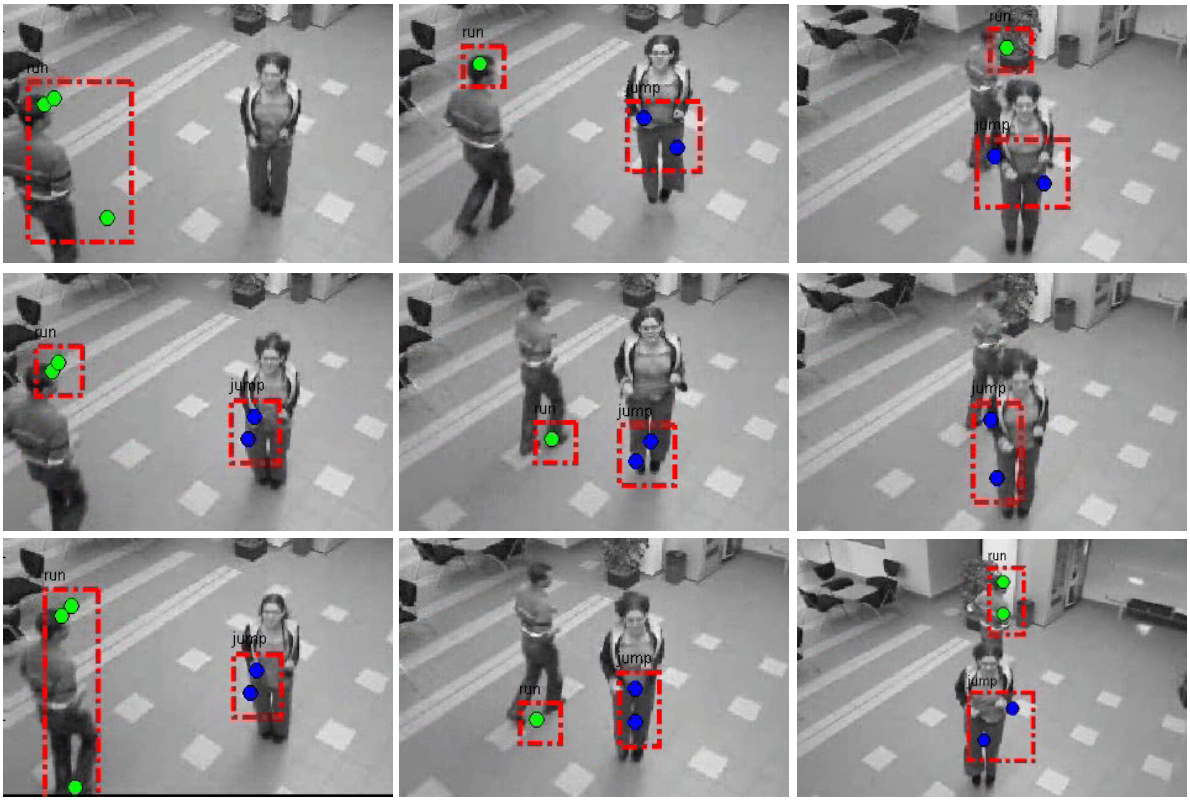
Figure 2 shows some visual results for our dataset. The detected actions are delineated with red bounding boxes at the frame level. Circles are the detected ST-points, from which we localize activities. From this figure, we can see that our method can detect any kind of activities, i.e. not limited to the periodic activities (e.g. walking) and the interaction activities. Note that in our dataset, several actions are of very short duration (e.g. sitdown), the methods that

[3] Several teams have proposed algorithms and tested them on their own datasets which are not publicly available, e.g. [19, 20].

[4] Our dataset will be available soon.



a) recognition results for videos containing simultaneous walking and sitdown



b) recognition results for videos containing simultaneous running and jumping

Figure 2. Recognition results on several consecutive frames of two videos of our dataset (top to bottom and left to right). This figure should be viewed in color.

are based on a bag of word model often fail for such actions.

6. Conclusion

In this paper, we propose a graph matching method for action recognition in video sequences. Our method features several advantages for activity recognition such as: it can detect/localize multiple activities in the same video sequence without any preprocessing; our method avoids the non-trivial problem of selecting the optimal number of codewords for codebook construction; unlike many other methods it does not need training, background/foreground segmentation, tracking, and does not require any prior knowledge on the action. Through experiments, we have shown that it is feasible to apply graph matching methods to action recognition.

Besides the advantages mentioned above, our method still has limitations, e.g. detecting multiple instances of the same activity in a video. In this case, we need to detect the first instance, eliminate it, and restart a new process for other instances. These limitations will be considered in our ongoing work. In addition, we also aim at extending this method to capture higher-order geometric structures among the local features, e.g. relationships between triangles.

Acknowledgments

This work was partly financed through two French National grants: ANR-CaNaDA *Comportements Anormaux : Analyse, Détection, Alerte*, No. 128, which is part of the call for projects CSOSG 2006 *Concepts Systèmes et Outils pour la Sécurité Globale.*, as well as ANR-Sattic, which is part of the call *Blanc 07-1_184534*.

References

[1] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. volume 2, pages 1395–1402 Vol. 2, 2005. 6

[2] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *IJPRAI*, 18(3):265–298, 2004. 2

[3] T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. In *NIPS*, 2007. 3

[4] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. pages 65–72. VS-PETS, 2005. 1, 4, 6

[5] O. Duchenne, F. R. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. In *CVPR*, pages 1980–1987, 2009. 2, 3, 5

[6] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *CVPR*, 2008. 6

[7] A. Gilbert, J. Illingworth, and R. Bowden. Scale invariant action recognition using compound features mined from dense spatio-temporal corners. In *ECCV*, pages 222–233, Berlin, Heidelberg, 2008. Springer-Verlag. 6

[8] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *PAMI*, 29(12):2247–2253, December 2007. 6

[9] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *IEEE International Conference on Computer Vision*, October 2007. 1

[10] T.-K. Kim and R. Cipolla. Canonical correlation analysis of video volume tensors for action categorization and detection. *PAMI.*, 31(8):1415–1428, 2009. 6

[11] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV '05:*, pages 1482–1489, Washington, DC, USA, 2005. 2, 3

[12] J. Liu, S. Ali, and M. Shah. Recognizing human actions using multiple features. In *CVPR*, 2008. 1

[13] J. Liu and M. Shah. Learning human actions via information maximization. In *CVPR*, 2008. 1, 6

[14] J. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *IJCV, 2008*. 1, 2, 6

[15] M. S. Ryoo and J. K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *ICCV*, 2009. 1, 6

[16] S. Savarese, A. Del Pozo, J. Niebles, and F. Li. Spatial-temporal correlatons for unsupervised action classification. In *In WMVC*, pages 1–8, 2008. 1, 2, 6

[17] K. Schindler and L. van Gool. Action snippets: How many frames does human action recognition require? In *CVPR*. IEEE Press, June 2008. 6

[18] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *ICPR*, volume 3, pages 32–36 Vol.3, September 2004. 1, 6

[19] E. Shechtman and M. Irani. Space-time behavior-based correlation—or—how to tell if two underlying motion fields are similar without computing them? *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(11):2045–2056, 2007. 1, 6

[20] X. Sun, M. Chen, and A. Hauptmann. Action recognition via local descriptors and holistic features. In *CVPR Workshop*, pages 58–65, 2009. 1, 6

[21] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *ECCV*, pages 596–609, Berlin, Heidelberg, 2008. Springer-Verlag. 2

[22] L. Wang, X. Geng, C. Leckie, and K. Ramamohanarao. Moving shape dynamics: A signal processing perspective. In *CVPR*, 2008. 1

[23] D. Weinland, E. Boyer, and R. Ronfard. Action recognition from arbitrary views using 3d exemplars. In *ICCV*, pages 1–7, 2007. 1

[24] S. Zampelli1, Y. Deville, and C. Solnon. Solving subgraph isomorphism problems with constraint programming. *Constraints*, 2009. 2

[25] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *CVPR*, 2008. 2

[26] Z. Zhang, Y. Hu, S. Chan, and L.-T. Chia. Motion context: A new representation for human action recognition. In *ECCV (4)*, pages 817–829, 2008. 1, 6