GRUNTS: Graph Representation for UNsupervised Temporal Segmentation

Francesco Battistone[1], Alfredo Petrosino[1(✉)], and Gabriella Sanniti di Baja[2]

[1] Department of Science and Technology, University of Naples Parthenope, Naples, Italy
alfredo.petrosino@uniparthenope.it

[2] Institute for High Performance Computing and Networking, National Research Council, Naples, Italy

Abstract. We propose GRUNTS, a feature independent method for temporal segmentation via unsupervised learning. GRUNTS employs graphs, through skeletonization and polygonal approximation, to represent objects in each frame, and graph matching to efficiently compute a Frame Kernel Matrix able to encode the similarities between frames. We report the results of temporal segmentation in the case of human action recognition, obtained by adopting the Aligned Cluster Analysis (ACA), as unsupervised learning strategy. GRUNTS has been tested on three challenging datasets: the Weizmann dataset, the KTH dataset and the MSR Action3D dataset. Experimental results on these datasets demonstrate the effectiveness of GRUNTS for segmenting actions, mainly compared with supervised learning, typically more computationally expensive and not prone to be real time.

1 Introduction

The aim of temporal segmentation is to cut an input sequence into segments with different semantic meanings; in particular in human action recognition, some methods focus on simple primitive actions such as walking, running and jumping, without taking into account the fact that normal activity involves complex temporal patterns. In this work the problem of temporal segmentation of human behavior is formulated as a temporal clustering problem.

We propose GRUNTS, a novel graph representation that combines the result of a skeletonization algorithm with a polygonal approximation technique to obtain a structure that schematically represents the silhouette of the object of interest for each frame. The name GRUNTS (unskilled workers) describes and highlights the simple representation that the method uses for each frame. This method, taking an input sequence, allows to obtain for each frame, the graph which approximates at best the skeleton of the relative silhouette. After building all graphs, a graph matching is adopted to estimate the similarities between the graphs in order to achieve the Frame Kernel Matrix (FKM). Finally, we adopt the Aligned Cluster Analysis (ACA)[1] on the FKM to segment the sequence.

© Springer International Publishing Switzerland 2015
V. Murino and E. Puppo (Eds.): ICIAP 2015, Part I, LNCS 9279, pp. 225–235, 2015.
DOI: 10.1007/978-3-319-23231-7_21

The method possesses some advantages: (i) it captures structure information of each action by a graph representation of the silhouette; (ii) it finds an hidden structure in unlabeled data; (iii) it efficiently works on different types of datasets (2D-dataset, 3D-dataset and RGB-D dataset). GRUNTS has been tested in the framework of human action recognition on three challenging datasets: the Weizmann dataset [2], the KTH dataset [3] and the MSR Action3D dataset [4], and the obtained results prove the effectiveness of GRUNTS for segmenting actions.

Sections 2 and 3 discuss how the graph is constructed for each silhouette at each frame, while Section 4 reports the building of the Frame Kernel Matrix by graph matching and in the Section 5 how ACA works on FKM to segment a sequence is shown. Finally, in the Section 6 we report the results of GRUNTS on the three different datasets.

2 Graph Representation

The representation of the meaningful features captured at each frame has an important role in our approach. Once the skeleton of the silhouette is available, polygonal approximation is performed to identify the graph approximating the skeleton branches.

2.1 Skeleton

The skeletonization method adopted by GRUNTS [5] does not require the iterated application of topology preserving removal operations, and does not need checking a condition specifically tailored to end point detection. In fact, skeletonization is accomplished on the distance transform DT of the object, computed according to the (3,4) distance [6]. Thus, end points are automatically identified when the so called centers of maximal discs are found in DT. The skeletal pixels are all found in one raster scan inspection of DT. The set of the skeletal pixels detected in DT has all the properties expected to characterize the skeleton of the object except for unit thickness. Indeed, the set of the skeletal pixels is 2-pixel thick in correspondence of regions of the object with thickness given by an even number of pixels. Thus, a reduction to unit width is obtained by using templates able to erase the marker from suitable skeletal pixels. Finally, a pruning step is also taken into account to simplify the structure of the resulting skeleton by removing some peripheral branches corresponding to scarcely elongated regions. The elongatedness of each object region can be measured by analyzing the skeleton branch mapped into it and a threshold on elongatedness can be set depending on the specific application.

2.2 Polygonal Approximation

Several approaches exist in the literature to compute the polygonal approximation of a digital curve. We use a split type approach [7] because it is convenient when working with open curves, like the individual skeleton branches. This type of algorithm can be described as follows. The two extremes of the input open curve are taken by all means as vertices of the polygonal approximation. The Euclidean distance of all points of the curve from the straight line joining the two vertices is computed. In particular, each point of the skeleton at position (x, y) and with distance value k can be interpreted as a point in the $3D$ space with coordinates (x, y, k). Then, the Euclidean distance d of a point C in the $3D$ space from the straight line joining two points A and B, is calculated by using the following expression:

$$d^2 = \|AC\|^2 - P_{ABC} * \frac{P_{ABC}}{\|AB\|^2} \tag{1}$$

where $\|AC\|$ is the norm of the vector AC, and P_{ABC} is the scalar product between vectors AB and AC.

The point with the largest distance is taken as a new vertex, provided that such a distance overcomes an a priori fixed threshold θ(for this work, $\theta = 1.5$, as we aim at a faithful approximation [8]). If a new vertex is detected, such a vertex divides the curve into two sub-curves, to each of which the above split type algorithm is applied. The splitting process is repeated as far as points are detected having distance larger than θ from the straight lines joining the extremes of the sub-curves to which the points belong. When the recursion is completed the points that have been detected as vertexes represent the nodes while the segments that approximate the curve represent the edges of the graph that best approximates the skeleton (see Fig. 1). Note that Fig. 1b has been cropped to the smallest area safely including the foreground once the binarized version of Fig. 1a has been obtained, so as to reduce the amount of data to process.

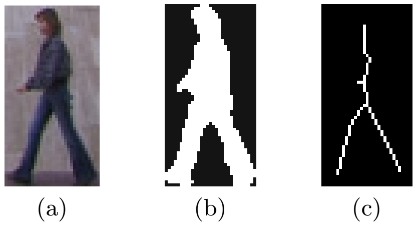


Fig. 1. A frame of the sequence (a); binary image where the foreground is the silhouette (b); the graph obtained by skeletonization and polygonal approximation (c).

2.3 Building the Graph

A graph with n vertexes and m undirected edges is represented as a 4-tuple $Gr = \{P, Q, G, H\}$, where $\mathbf{P} \in \mathbb{R}^{3\times n}$ represent the set of vertexes that have been detected by polygonal approximation. Each vertex has three coordinates, $(x, y, DT_{(3,4)}(x, y))$, where x and y are the spatial coordinates of the vertex while

$DT_{(3,4)}(x, y)$ is the value of DT in position (x, y). The third coordinate is important because it describes the thickness of the object region in correspondence to the skeleton point; in fact, graphs with similar structure may correspond to objects with different shapes. An example is shown in Fig. 2.

$\mathbf{Q} \in \mathbb{R}^{2\times m}$ is the set of edges represented as 2-D vector encoding the Euclidean distance between vertexes and the orientation of the c_{th} edge. The topology of the graph is represented by two binary matrices $\mathbf{G}^{n\times m}$,$\mathbf{H}^{n\times m}$, where $g_{ic} = h_{jc} = 1$ if the c_{th} edge starts from the i_{th} node and ends at the j_{th} node.

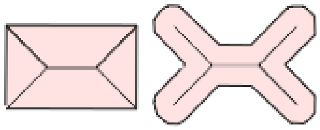


Fig. 2. Objects with different thickness and skeletons with the same geometrical structure.

3 The Frame Kernel Matrix

A Frame Kernel Matrix, $\mathbf{K} \in \mathbb{R}^{n_f \times n_f}$, where n_f is the number of frames of the input sequence, is constructed over the affinity between consecutive frames, measured as graph matching between consecutive graphs. Specifically, we adopt the algorithm of graph matching reported in [9]. A measure of similarity between graphs is achieved through the combination of the distance measures between vertexes and edges, encoded in the pairwise affinity matrix $\mathbf{A}$. Formally, given a pair of graphs $Gr_1 = \{P_1, Q_1, G_1, H_1\}$ and $Gr_2 = \{P_2, Q_2, G_2, H_2\}$, two matrices are computed:

- matrix $K_p \in \mathbb{R}^{n_1\times n_2}$, representing the similarity between the n_1 vertexes of Gr_1 and the n_2 vertices of Gr_2. The value of $k^p_{i,j}$ is calculated as Euclidean distance between the i^{th} vertex of Gr_1 and the j^{th} vertex of Gr_2.
- matrix $K_q \in \mathbb{R}^{m_1\times m_2}$, representing the similarity between the m_1 edges of Gr_1 and the m_2 edges of Gr_2. K_q is calculated as an average between the matrix A^w (affinity of the weights) and the matrix A^θ (affinity of the orientations): $A^w_{ij} = |w^1_i - w^2_j|$, where w^1_i is the weight of the i^{th} edge of Gr_1 and w^2_j is the weight of the j^{th} edge of Gr_2, and $A^\theta_{ij} = |\theta^1_i - \theta^2_j|$, where θ^1_i is orientation of the i^{th} edge of Gr_1 with respect to the z-axis and θ^2_j is orientation of the j^{th} edge of Gr_2 with respect to the z-axis.

Vertex and edge affinities are encoded in a symmetrical matrix $A \in \mathbb{R}^{(n_1\cdot n_2)\times(n_1\cdot n_2)}$, whose elements are computed as follows:

$$A_{i_1,i_2,j_1,j_2} = \begin{cases} k^p_{i1,i2}, & \text{if } i_1 = j_1 \text{and } i_2 = j_2; \\ k^q_{c_1,c_2}, & \text{if } i_1 \neq j_1 \text{ and } i_2 \neq j_2 \text{ and} \\ & g^1_{i_1c_1}h^1_{j_1c_1}g^2_{i_2c_2}h^2_{j_2c_2} = 1; \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where the diagonal and off-diagonal elements encode the similarity between nodes and edges respectively. To construct the Frame Kernel Matrix, $\mathbf{K} \in \mathbb{R}^{n_f \times n_f}$, we need a single value of affinity for each graph matching, encoded in the matrix $\mathbf{A}$. This affinity is calculated as the mean of the maximum elements in each of $nb = n_{min} \cdot n_{min}$ sub-blocks $B^{n_{max} \times n_{max}}$ contained in the $\mathbf{A}$ matrix (where $n_{min} = min(n_1, n_2)$, and $n_{max} = max(n_1, n_2)$). Each entry, k_{ij}, defines the similarity between two frames, $\mathbf{x}_i$ and $\mathbf{x}_j$.

4 Aligned Cluster Analysis (ACA)

GRUNTS uses an extension of kernel k-means, ACA [1], for the temporal segmentation of human action. In contrast to k-means, dynamic time alignment kernel (DTAK) is adopted as measure of distance to establish which action is represented into a segment, defined as a set of consecutive frames. Furthermore, given a sequence $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ composed by n frames, ACA partitions $\mathbf{X}$ into m disjointed segments, each one related to a different class representing a particular action. The i^{th} segment, $\mathbf{Y}_i \doteq \mathbf{X}_{[s_i, s_{i+1})} = [\mathbf{x}_{s_i}, ..., \mathbf{x}_{s_i + 1} - 1] \in \mathbb{R}^{d \times n}$, is composed by frames from position s_i to $s_{i+1} - 1$. The length of the segment is constrained to be $n_i = s_{i+1} - s_i \leq n_{max}$, where n_{max} is the maximum length of the segment and controls the temporal granularity of the factorization.

A matrix $\mathbf{G} \in \{0, 1\}^{k \times m}$ includes information about the assignment of each segment to a class: $g_{ci} = 1$ if $\mathbf{Y}_i$ belongs to class c, else $g_{ci} = 0$. ACA extends previous work [10] by minimizing:

$$J_{aca}(\mathbf{G}, \mathbf{s}) = \sum_{c=1}^{k} \sum_{i=1}^{m} g_{ci} \|\psi(\mathbf{X}_{[s_i, s_{i+1})}) - \mathbf{z}_c\|^2 \tag{3}$$

where k is the number of classes (which is dependent on the dataset), $\mathbf{z}_c \in \mathbb{R}^d$ is the geometric centroid of the data points for the class c and the distance is computed as follows:

$$\|\psi(\mathbf{X}_{[s_i, s_{i+1})}) - \mathbf{z}_c\|^2 = \tau_{ii} - \frac{2}{m_c} \sum_{j=1}^{m} g_{cj_1} \tau_{ij} + \frac{1}{m_c^2} \sum_{j_1, j_2 = 1}^{m} g_{cj_1} g_{cj_2} \tau_{j_1 j_2}, \tag{4}$$

where $m_c = \sum_{j=1}^{m} g_{cj}$ is the number of segments that belong to class c. The dynamic kernel function τ is defined as $\tau_{ij} = \psi(\mathbf{Y}_i)^T \psi(\mathbf{Y}_j)$ based on [11] and $\psi(\cdot)$ denotes a mapping of the sequence into a feature space. Zhou et al. in [12] presented the hierarchical version of ACA (termed HACA) that reduces the computational complexity of ACA and provides a hierarchical decomposition at different temporal scales. HACA replaces the kernel DTAK with the GDTAK (generalized DTAK) to propagate the solution at different levels, even if it does not substantially change the main idea.

5 The GRUNTS Algorithm

The main advantage of our technique surely consists in its feature independence on the input datasets. This is stressed in the detailed description of Algorithm 1 (see below). Given an input sequence, GRUNTS is performed as shown, and after the Frame Kernel Matrix is built, the aligned cluster analysis ACA is executed to obtain the temporal segmentation.

Algorithm 1 GRUNTS

```
 1: Take a sequence in input
 2: for all frames do
 3:     if frame is not a skeleton of the $RGB-D$ dataset then
 4:         Read each frame of a sequence;
 5:         if frame is in the $2D-space$ then
 6:             if frame is not binary then
 7:                 Background Subtraction by SOBS [13]
 8:             end if
 9:             Calculate $(3-4)DT$
10:             Skeletonization
11:         end if
12:         if frame is in the $3D-space$ then
13:             Skeletonization
14:         end if
15:         Polygonal Approximation
16:     end if
17: end for
18: Calculate FKM
19: Execute ACA
```

6 Experiments

GRUNTS has been evaluated on three different types of datasets: Weizmann Dataset, KTH Dataset and MSR Action3D Dataset. We set the pruning threshold to $dist^2 = 20$ and the threshold for polygonal approximation to $\theta = 1.5$. The Frame Kernel Matrix is calculated by the Gaussian kernel, $k_{ij} = exp(-\frac{dist(\mathbf{x}_i,\mathbf{x}_j)^2}{2\sigma^2})$, where $dist(\mathbf{x}_i, \mathbf{x}_j)$ is obtained by Graph Matching as following:

$$dist(\mathbf{x}_i, \mathbf{x}_j) = 1 - (\text{similarity}_{GM}(Gr_i, Gr_j)), \tag{5}$$

where Gr_i and Gr_j are the graphs respectively related to the frames $\mathbf{x}_i$ and $\mathbf{x}_j$.

To execute ACA on a sequence, we need to set the parameters n_{min} and n_{max} that represent the minimal ad the maximal length of a segment and are adopted to control the granularity of the factorization. To evaluate the clustering accuracy, we compute the confusion matrix C and its accuracy as in [12].

6.1 Weizmann Dataset

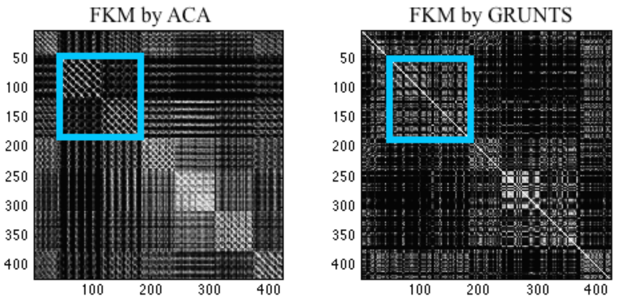
The Weizmann dataset [2] contains 90 videos of 10 individuals performing nine different actions. In this Section we compare GRUNTS with ACA that calculates the frame kernel matrix as described in [9]. The goal of these tests is to verify that GRUNTS is characterized by the following two features: i) independence from actors and action direction, and ii) independence from length and number of actions:

Independence from Actors and Action Direction. GRUNTS, unlike ACA, is not sensitive to sequences containing the same actions performed by different actors, and especially the changes of direction in different actions do not lead to a decrease in performance. In our experiments, GRUNTS and ACA are executed on fifty sequences, of seven actions, containing some equal actions executed by different actors. We show a sample sequence to prove this:

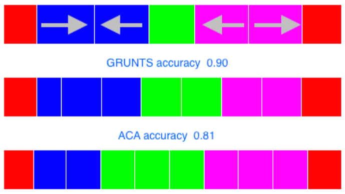
Daria_run, Eli_walk, Denis_walk, Daria_skip, Denis_jump, Daria_jump, Eli_run.

The accuracy for this sequence was 0.90 for GRUNTS, and 0.81 for ACA. From the Frame Kernel Matrix and the segmentation of the sequence shown in Fig. 3, it is evident that ACA suffers particularly the presence of actions (especially walk, run, skip and jump) equally performed by different actors and especially with opposite directions (the gray arrow in the ground truth of the sequences shows the direction of action). Indeed, the construction of the FKM (matrix on the left) in the original ACA does not allow to identify the similarity between the two subsequences "Eli_walk" and "Denis_walk" in the example (note the differences between the FKM obtained with ACA and GRUNTS and highlighted with a blue square), which is more clear in the FKM obtained by GRUNTS (matrix on the right). This also justifies the great improvement in accuracy achieved by the proposed technique. Refer to *http://cvprlab.uniparthenope.it/GRUNTS.pdf* to compare the results obtained by GRUNTS and ACA on the fifty sequences. For both techniques the best parameters have been found for each sequence. The average accuracy achieved by GRUNTS and ACA is respectively 0.88 and 0.83, showing that GRUNTS has better performance. It is also interesting to highlight the accuracy achieved by GRUNTS on the sequences: 1, 12, 18, 29, 37, 40 and 49, shown at *http://cvprlab.uniparthenope.it/GRUNTS.pdf*, since these sequences contain actions that are performed by different actors and in different directions.

Independence from Length and Number of Actions. The performance of GRUNTS is not related to the length of the input sequences and even to the number of actions. Fig. 4 shows the graphs obtained by the experiments executed to confirm the thesis on the independence of the segmentation from the number of classes (Fig. 4a) and the length of the sequence (Fig. 4b). Looking at the first graph, it is evident that the results obtained by our method are less related to



(a)



(b)

Fig. 3. (a) Frame Kernel Matrices for ACA and GRUNTS on the sample sequence; (b) Ground Truth (first row) and Temporal Segmentation by GRUNTS (second row) and ACA (third row) on the sample sequence.

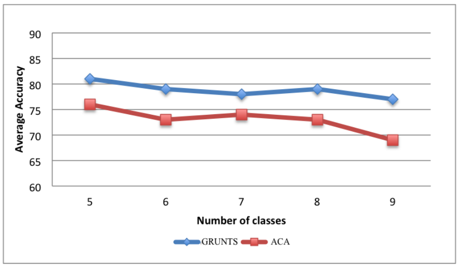
the number of distinct classes, because the range of GRUNTS is smaller than that obtained by ACA. The graph on the number of frames, and therefore the length of the sequence, shows that GRUNTS obtained results always close to the average value (around 76%), while the results achieved by ACA generally differ from the average value more than those obtained by GRUNTS.

6.2 KTH Dataset

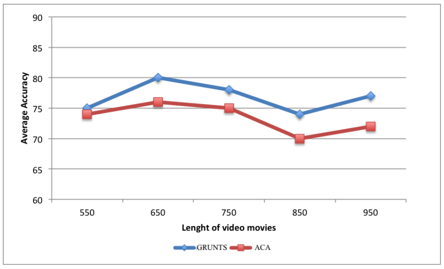
The KTH dataset [3] contains six types of human actions performed by 25 subjects in different scenarios. For this dataset, as claimed by the authors, ACA computes the FKM with a technique based on optical flow [9]. We generated 10 random testing videos for the KTH dataset and each of the videos contains 10-20 clips of different actions. The average accuracies have been: GRUNTS 0.86, ACA 0.80, HACA 0.83,Spectral Clustering (SC) [10] 0.72, setting $n_{min} = 7$ and $n_{max} = 29$ for all experiments of GRUNTS on KTH.

6.3 MSR Action3D Dataset

This dataset [4] contains twenty actions and each action was performed three times by ten subjects. GRUNTS is executed without skeletonization and polygonal approximation, since the dataset includes a skeleton already computed for



(a)



(b)

Fig. 4. Graphs of average accuracy calculated on 25 tests of 10 actions and on 25 tests containing 20 actions: (a) Graph of the average accuracy calculated on the number of classes; (b) Graph of the average accuracy on the length of the sequence.

each frame. Wang et al. [14] divide the dataset into three subsets, on which we perform GRUNTS. The results on this subsets and the average accuracy are shown in Table 1. GRUNTS does not outperform in this case, even if its accuracy is comparable with that of the two techniques with the highest accuracy values, but it is important to highlight that GRUNTS is an unsupervised method, while all other techniques are supervised.

Table 1. Performance on MSR Action3D dataset: (a) Accuracy of GRUNTS on three subsets; (b) average accuracy of GRUNTS and other approaches.

(a)

SubSet	**GRUNTS**
AS1	83.4
AS2	83.7
AS3	88.9
Average	85.3

(b)

Method	**Accuracy**
HON4D + D_{disc} [15]	88.9
Jiang et al [14]	88.2
Dollar + BOW [16]	72.4
Vieira [17]	78.2
Klaser [18]	81.4
GRUNTS	85.3

7 Conclusions

The new graph representation for unsupervised temporal segmentation of human actions reported in this paper gets surely advantage of the adopted structured representation to deal with feature independence on different datasets. It can work on different types of datasets and the achieved results are comparable with the state-of-the-art algorithms of temporal segmentation - with the specificity to be unsupervised. Indeed, we tested GRUNTS on three different datasets (Weizmann, KTH and MSR Action3D Dataset) and as clustering we have chosen the ACA clustering for its ability to be robust to noise and invariant to temporal scaling factor. We also experimented the hierarchical clustering version (HACA), but we did not report the results since: the results of the non-hierarchical techniques are comparable to those of hierarchical techniques; the average execution time of a given sequence of seven actions is 1.90 seconds for non-hierarchical techniques and about 60 seconds for the hierarchical ones.

References

1. Zhou, F., De la Torre Frade, F., Hodgins, J.K.: Aligned cluster analysis for temporal segmentation of human motion. In: IEEE Conference on Automatic Face and Gestures Recognition, September 2008
2. Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. Transactions on Pattern Analysis and Machine Intelligence **29**(12), 2247–2253 (2007)
3. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local svm approach. In: 17th International Conference on Proceedings of the Pattern Recognition, (ICPR 2004), vol. 3, pp.32–36. IEEE Computer Society, Washington, DC (2004)
4. Li, W., Zhang, Z., Liu, Z.: Action recognition based on a bag of 3d points (2010)
5. Sanniti di Baja, G.: Well-shaped, stable, and reversible skeletons from the (3,4)-distance transform. Journal of Visual Communication and Image Representation **5**(1), 107–115 (1994)
6. Borgefors, G.: Distance transformations in digital images. Comput. Vision Graph. Image Process. **34**(3), 344–371 (1986)
7. Ramer, U.: An iterative procedure for the polygonal approximation of plane curves. Computer Graphics Image Process. **1**(3), 244–256 (1972)
8. Rosenfeld, A.: Digital straight line segments. IEEE Transactions on Computers **23**(12), 1264–1269 (1974)
9. Zhou, F., De la Torre, F.: Factorized graph matching. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 127–134, June 2012
10. Dhillon, I.S., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2004, pp. 551–556. ACM, New York (2004)
11. Shimodaira, H., ichi Noma, K., Science, S.O.I., Nakai, M., Sagayama, S.: Dynamic time-alignment kernel in support vector machine (2001)
12. Zhou, F., De la Torre, F., Hodgins, J.: Hierarchical aligned cluster analysis for temporal clustering of human motion. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(3), 582–596 (2013)
13. Maddalena, L., Petrosino, A.: A self-organizing approach to background subtraction for visual surveillance applications. Trans. Img. Proc. **17**(7), 1168–1177 (2008)
14. Wang, J., Liu, Z., Chorowski, J., Chen, Z., Wu, Y.: Robust 3D action recognition with random occupancy patterns. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part II. LNCS, vol. 7573, pp. 872–885. Springer, Heidelberg (2012)

15. Oreifej, O., Liu, Z.: Hon4d: histogram of oriented 4d normals for activity recognition from depth sequences. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 716–723, June 2013
16. Dollar, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: Proceedings of the 14th International Conference on Computer Communications and Networks, ICCCN 2005, pp. 65–72. IEEE Computer Society, Washington, DC (2005)
17. Vieira, A.W., Nascimento, E.R., Oliveira, G.L., Liu, Z., Campos, M.F.M.: STOP: space-time occupancy patterns for 3D action recognition from depth map sequences. In: Alvarez, L., Mejail, M., Gomez, L., Jacobo, J. (eds.) CIARP 2012. LNCS, vol. 7441, pp. 252–259. Springer, Heidelberg (2012)
18. Kläser, A., Marszałek, M., Schmid, C.: A spatio-temporal descriptor based on 3d-gradients. In: British Machine Vision Conference, pp. 995–1004, september 2008