Recognition of Human Actions using Edit Distance on Aclet Strings

Luc Brun[1], Pasquale Foggia[2], Alessia Saggese[2] and Mario Vento[2]
[1]*GREYC UMR CNRS 6072, ENSICAEN - Université de Caen Basse-Normandie, 14050 Caen, France*
[2]*Dept. of Computer Eng. Electrical and Eng.Applied Mathematics, University of Salerno*
Via Giovanni Paolo II, 132, Fisciano (SA), Italy

Keywords: Action Recognition, Human Behavior Analysis, String Based Representations, Edit Distance.

Abstract: In this paper we propose a novel method for human action recognition based on string edit distance. A two layer representation is introduced in order to exploit the temporal sequence of the events: a first representation layer is obtained by using a feature vector obtained from depth images. Then, each action is represented as a sequence of symbols, where each symbol corresponding to an elementary action (aclet) is obtained according to a dictionary previously defined during the learning phase. The similarity between two actions is finally computed in terms of string edit distance, which allows the system to deal with actions showing different length as well as different temporal scales. The experimentation has been carried out on two widely adopted datasets, namely the MIVIA and the MHAD datasets, and the obtained results, compared with state of the art approaches, confirm the effectiveness of the proposed method.

1 INTRODUCTION

The recognition of human actions has attracted in the last years several scientists working in the pattern recognition and computer vision fields (Turaga et al., 2008; Poppe, 2010; Weinland et al., 2011; Vishwakarma and Agrawal, 2013). This is mainly due to the following facts: first, the applicative outcome is considerable, ranging from video surveillance to ambient assisted living and business intelligence application fields. Second, this problem can be seen as a rather traditional pattern recognition task: a set of features is computed on the sequence of images and is used to feed a classifier, which is trained on a set of labeled training data.

Starting from this last assumption, the main focus of the scientific community up to now has been on the definition of novel feature sets, tailored to discriminate the different actions of interest. For instance, in (Mokhber et al., 2005) actions are represented through a spatio-temporal cuboid, which is described by a geometrical transform based on Hu moments; in (Dollar et al., 2005) the cuboid is built by exploiting a 2D Gaussian filter for managing the spatial dimensions and a 1D Gabor filter for the temporal one. Scale and translation invariance is the main focus of the methods presented by (Chen et al., 2011), (Wang et al., 2007) and (Yuan et al., 2013); in particular, the former adopts the Radon transform while the last two approaches exploit an extended version, namely the $\mathfrak{R}$ transform.

Independently of the particular way in which the features are designed, one of the main limitations of the above mentioned systems lies in the fact that their decision is strongly influenced by the noise introduced during the feature extraction step. For this reason, in the last years a common trend of the scientific community has been to introduce a high level representation, aimed at increasing the overall recognition performance. In (Zhang and Gong, 2010) and (Sung et al., 2012), for instance, the use of an Hidden Markov Model (HMM) is proposed. In general, the main drawback of HMM based approaches lies in the large amount of labeled data required during the training step, which is often not simple to obtain in real environments. In (Foggia et al., 2014) a deep learning architecture is introduced, aiming at extracting a high level representation of the actions directly from the data. Although they show very promising results, both the methods based on HMM and deep learning architectures are very difficult to configure and in general the achievement of good results requires a significant expertise in the field.

The introduction of a high level representation for actions has been also exploited in (Dollar et al., 2005), (Li et al., 2013) and (Foggia et al., 2013), where a bag of words approach has been adopted. The main idea is to use the first-level feature vectors to recog-

Brun L., Foggia P., Saggese A. and Vento M..
Recognition of Human Actions using Edit Distance on Aclet Strings.
DOI: 10.5220/0005304700970103
In *Proceedings of the 10th International Conference on Computer Vision Theory and Applications* (VISAPP-2015), pages 97-103
ISBN: 978-989-758-090-1

nize small elements of an action called visual words; then the histogram of the occurrences of such visual words is used as a high-level feature vector to perform the classification of the action. The set of the visual words is defined by constructing a codebook using an unsupervised learning approach. The main drawback of the bag of words methods is that they base their decision only on the occurrence or on the absence of the relevant visual words (the elementary actions) within the analyzed time window; the order in which these words appear is not taken into account. However, for human beings, this order is an important piece of information for discriminating between similar actions. On the other hand, a element-wise comparison of the observed sequence of visual words with the one obtained for a reference action would not yield good results, because of two kinds of problems: first, the speed of execution of the same action by different persons (or even by the same person at different times) may change; the change may even be not uniform within the same action. Second, both because of noise in the first-level representation and of individual differences in the way an action is performed, an observed sequence of visual word will likely contain spurious elements with respect to the corresponding reference action in the training set, and conversely may lack some elements of the latter.

In order to overcome the above mentioned problems, in this paper we propose a system for Human Action Recognition based on a string Edit Distance (HARED); each action is represented as a sequence of symbols (a string) according to a dictionary acquired during the learning step. The similarity between two strings is computed by a string edit distance, measuring the cost of the minimal sequence of edit operations needed to transform one string into the other; the string edit distance is robust with respect to local modifications (such as the insertion or deletion of symbols) even when they change the length of the string, thus dealing in a natural way with speed changes and with spurious elementary actions.

The experimentation, conducted over two standard datasets, confirms the robustness of the proposed approach, both in absolute terms and in comparison with other state of the art methodologies.

2 THE PROPOSED METHOD

In Figure 2c an overview of the proposed approach is shown; more details about each module, namely first layer representation, second layer representation and classification will be detailed in the following.

2.1 Feature Extraction

The feature vector is extracted by analyzing depth images acquired by a Kinect sensor. This choice is mainly justified by the following reasons: first, the device has a very affordable cost, so making such method especially suited for budget-constrained applications. Furthermore, in (Carletti et al., 2013) the authors proved the effectiveness of a set of features obtained by the combination of three different descriptors, respectively based on Hu Moments, $\Re$ transform and Min-Max variations, computed on depth images. Starting from the above considerations, in this paper we decided to adopt the same feature vector. It is worth pointing out that the focus of this paper is on the string-based high level representation, as well as on the measure introduced for evaluating the distance between two actions; it means that any kind of feature vector could be profitably used.

In order to compute the feature vector, we first extract the set of derived images, proposed in (Megavannan et al., 2012) and shown in Figure 2, able to model the spatio-temporal variations of each pixel: in particular, at each frame the last N frames are processed through the employing of a sliding window so as to obtain the *Average Depth Image* (*ADI*), the *Motion History Image* (*MHI*) and the *Depth Difference Image* (*DDI*). In our experiments N has been set to one second, as suggested in (Megavannan et al., 2012).

In particular, the *ADI* is the average depth at position (x,y) over the images at times $t-N+1,\ldots,t$; it uses N temporally adjacent depth images in order to capture the motion information in the third dimension. The *MHI* is able to capture into a single and static image the sequence of motions; In particular, $MHI(x,y,t) = 255$ if the point (x,y) passes from background to foreground at time t, otherwise it is equal to $max\{M(x,y,t-1)-\tau,0\}$. τ is a constant set in our experiments to $(256/N)-1$, as suggested in (Megavannan et al., 2012). Finally, the *DDI* evaluates the motion changes in the depth dimension: $DDI(x,y,t) = D_{max}(x,y,t) - D_{min}(x,y,t)$, where $D_{max}(x,y,t)$ and $D_{min}(x,y,t)$ are the maximum and minimum depth for position (x,y) over the images at times $t-N+1,\ldots,t$, respectively.

Both the *MHI* and the *ADI* are represented through the seven Hu moments, which are invariant to translation, scale and orientation. *DDI* is represented through a combination of the $\Re$ transform and the Min-Max Depth Variations. The former is an extended 3D discrete Radon transform, able to capture the geometrical information of the interest points. Although its very low computational complexity, $\Re$ transform is robust with respect to errors occurring

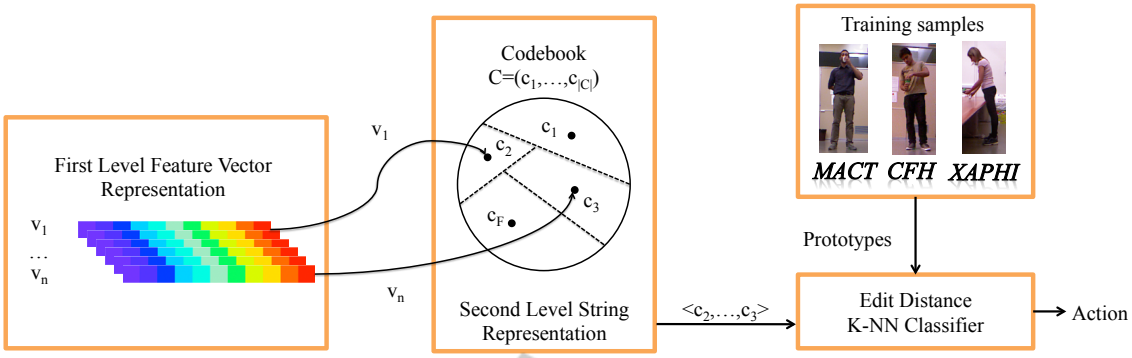


Figure 1: An overview of the proposed method. Each feature vector is associated to a visual symbol (here for simplicity represented as a letter), according to the codebook previously extracted during the learning step. Thus, the sequence of symbols is used to build a string that will be fed to the K-NN classifier based on the edit distance; the classifier finds the K most similar reference images among those provided in the learning step, and uses their classes (using a majority scheme) to assign a class to the observed action.

(a) (b) (c)

Figure 2: An example of the three derived images computed for extracting the first level feature vector, namely ADI (a), MHI (b) and DDI (c).

during the detection phase, such as disjoint silhouettes and holes in the shape. Min-Max Depth Variations is obtained by hierarchically partitioning the bounding box enclosing the silhouette into cells of equal size (1x1, 2x1, 1x2, 2x2, 3x3 and 6x6). For each cell, the minimum and the maximum value is computed, so obtaining a 108-sized vector.

Note that the features we used are complementary in their nature, since they are able to analyze the global distribution of the pixels (Hu Moments and the Min-Max Depth Variations) and at the same time to capture those properties related to the alignment of subregions of the image ($\Re$ transform).

2.2 String based Representation

A second layer representation based on *visual strings* is used to represent the actions. This choice allows our system to take into account the order, and not only the occurrence, of sub-actions for characterizing each action. In particular, (1) each first-level feature vector is encoded by an *aclet*, a visual symbol representing the small and atomic unit of action, namely a sub-action. (2) Then, the consecutive visual symbols are concatenated in order to obtain the visual string.

As for the first step, it is worth pointing out that the space of the possible feature vectors is by its nature ideally infinite. For this reason, during the learning step a preliminary quantization of the space is performed in an unsupervised way by using the well-known K-Means clustering algorithm. In this way, we are able to generate a *codebook* C, that is a finite vocabulary of visual symbols, obtained by assimilating the i-th cluster to its centroid c_i:

$$C = \left(c_1, \ldots, c_{|C|}\right), \quad (1)$$

being $|C|$ the number of clusters and thus the size of the dictionary. It is important to highlight that, as we will show in Section 3, HARED is very robust with respect to different $|C|$ values. This is a very important and not negligible feature since it makes the proposed approach easily configurable also by unexperienced operators.

During the operating phase, the codebook will be used in order to compute, for each low-level feature vector v_i, its closest cluster centroid c_j chosen from C and then to associate the i-th visual symbol s_i:

$$s_i = \arg\min_j |v_i - c_j|^2 \text{ for } j \in \{1, \ldots, |C|\}. \quad (2)$$

The concatenation of the last $|S|$ symbols are finally used in order to build the visual string:

$$S_t = < s_{t-|S|}, \ldots, s_t >, \quad (3)$$

where t is the current time instant. The $|S|$ value is adaptively identified during the learning step by computing the average string length over the training set.

2.3 The Decision Step

Once represented each action as a string, a K-NN classifier is used. The similarity between two strings

is evaluated by using an edit distance, and in particular the Levenshtein distance, able to evaluate the differences between two sequences in terms of insertion, deletion and substitution.

The main advantages in the use of a similarity measure based on edit distance are the following: first, edit distance takes into account the ordering of the symbols in the string, differently from methods based on histograms such as the bag of words approach; as we said, the order of sub-actions is a significant discriminative information for the classification of an action. Furthermore, it automatically finds an optimal alignment between the strings being compared, even a complex one involving different combinations of shrinking and expanding on different parts of the strings (Xiao et al., 2008); this is important to deal with possibly different local speeds of execution of actions. Finally, it is robust to small local changes, such as the insertion of spurious symbols, and so it is very well suited to work with noisy input data, such as those obtained by observing a person in a realistic environment.

For these very important properties, the edit distance has been successfully applied in several application domains, ranging from computational biology to signal processing and text retrieval (Navarro, 2001). Among the different edit distances metrics, such as Hamming distance and Longest Common Subsequence, the Levenshtein distance allows the system to deal with different length strings as well as to consider the substitution operators, very useful in order to give a more precise metric.

In a more formal way, let be $a =< a_1, a_2, ..., a_n >$ and $b =< b_1, b_2, ..., b_m >$ the two strings of length respectively n and m. The Levenshtein distance d_{mn} evaluates the minimum-cost sequence of operations required to transform a into b. In particular, when comparing symbols b_j and a_i, three possible edit operations are considered:

- *substitution* of a_j with b_i, with a cost $w_{sub}(a_j, b_i)$ (obviously the cost is 0 if $a_j = b_i$);
- *insertion* of a_j in the string b, with a cost $w_{ins}(a_j)$;
- *deletion* of b_i from the string b, with a cost $w_{del}(b_i)$;

There is a very elegant recursive formulation of the problem of finding the minimum edit distance d_{ij} between the substrings $< a_1, \ldots, a_j >$ and $< b_1, \ldots, b_i >$. The base cases of the recursive formulae are:

$$d_{i0} = \sum_{k=1}^{i} w_{del}(b_k), \text{ for } 1 \leq i \leq m \tag{4}$$

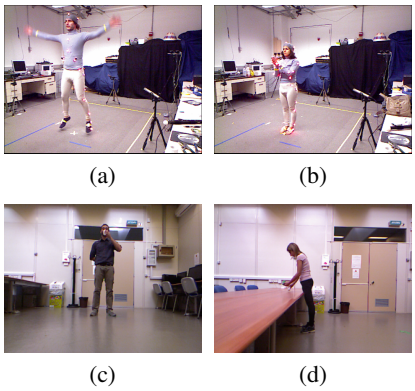


Figure 3: Some examples of the MHAD Dataset (a-b) and of the MIVIA Dataset (c-d).

$$d_{0j} = \sum_{k=1}^{j} w_{ins}(a_k), \text{ for } 1 \leq j \leq n \tag{5}$$

Given the base cases, the generic d_{ij} can be defined by the following recurrence:

$$d_{ij} = \min \begin{cases} d_{i-1,j} + w_{del}(b_i) \\ d_{i,j-1} + w_{ins}(a_j) \\ d_{i-1,j-1} + w_{sub}(a_j, b_i) \end{cases} \tag{6}$$

Finally, the edit distance between the two strings is defined as:

$$D(a,b) = d_{mn} \tag{7}$$

In order to speed up the computation of the edit distance, we computed it by exploiting the dynamic programming version proposed in (Wagner and Fischer, 1974), whose time complexity is $O(mn)$. Furthermore, considering that in our context we are not interested in visualizing the sequence of edit operations, the space complexity can be reduced to $O(min(m,n))$.

The K-NN classifier uses the edit distance for finding the K reference actions in the training sets; then, the class of the relative majority of the K actions is output as the result of the classifier.

3 EXPERIMENTAL RESULTS

The experimentation has been carried out over two widely adopted datasets, namely the Berkeley Multimodal Human Action Detection (hereinafter MHAD) Dataset (Ofli et al., 2013) and the MIVIA Dataset (Foggia et al., 2013), both providing RGB-D images and background for each sequence.

Some examples are reported in Figure 3, while the actions included in each dataset are listed in Table 1.

Table 1: Description of the MHAD (a) and of the MIVIA (b) datasets.

(a)

Action	Length per Recording
Jumping in place	5 secs
Jumping jacks	7 secs
Bending - hands up all the way down	12 secs
Punching (boxing)	10 secs
Waving - two hands	7 secs
Waving - one hand (right)	7 secs
Clapping hands	5 secs
Throwing a ball	3 secs
Sit down then stand up	15 secs
Sit down	2 secs
Stand up	2 secs

(b)

Action	Length per Recording
Opening a jar	2 sec
Drinking	3 secs
Sleeping	3 secs
Random Movements	11 secs
Stopping	7 secs
Interacting with a table	3 secs
Sitting	3 secs

In particular, the MHAD dataset is based on actions with movement in both upper and lower extremities, actions with high dynamics in upper extremities or actions with high dynamics in lower extremities. It contains 11 actions performed by 7 male and 5 female subjects. All the subjects performed 5 repetitions of each action, yielding about 660 action sequences which correspond to approximatively 82 minutes of total recording time. The MIVIA dataset is composed by 7 actions performed by 14 subjects, 7 male and 7 female. All the subjects performed 2 repetitions of each action. This dataset is more challenging, since it is mainly devoted to actions with movement involving only upper extremities.

As for the parameters of the edit distance and of the classifier, we have set the costs of the edit operations to be uniform:

$$w_{ins}(a_j) = w_{del}(b_i) = 1 \tag{8}$$

$$w_{sub}(a_j, b_i) = \begin{cases} 0 \text{ if } a_j = b_i \\ 1 \text{ if } a_j \neq b_i \end{cases} \tag{9}$$

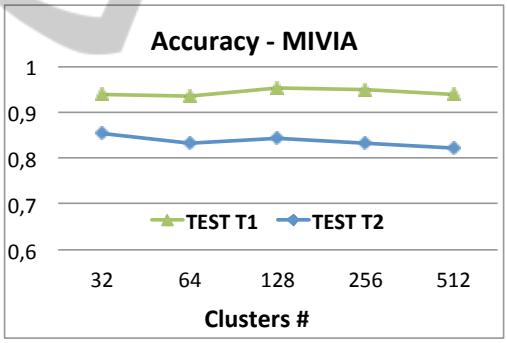
For parameter K of the classifier, we have set $K = 5$ after experimenting with several values over a subset of the training set.

The experimentation has been carried out according to two different protocols simulating different real scenarios:

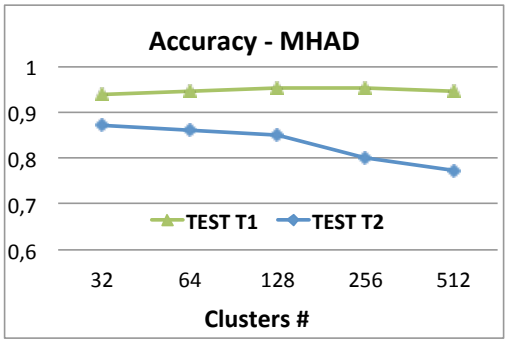
- **Protocol T1:** the person whose actions have to be recognized is known to the system. It means that the test person is included in the training set; in particular, given r repetitions of each action of the test person, one repetition is included in the training set while the other $r-1$ in the test set. This protocol simulates, for instance, the home monitoring of an elderly person, where it is possible to adapt the system during the configuration step to the person under test.
- **Protocol T2:** the test person is not known to the system. It implies that the test person is not included in the training set, which is formed by all the repetitions of all the other persons in the dataset. This protocol simulates those applications for surveilling public places, where the system has to recognize actions of unknown persons.

For each protocol, the tests have been carried out by a leave-one-out cross-validation strategy; in particular, each test has been repeated for all the repetitions of all the persons of each dataset and finally the average performance have been reported.

The results obtained by varying the $|C|$ parameter are reported in Figures 4a and 4b for the MIVIA and MHAD datasets, respectively. We can note that in



(a)



(b)

Figure 4: Accuracy computed respectively over the MIVIA (a) and the MHAD (b) datasets by varying the $|C|$ parameter.

Table 2: Comparison with state of the art approaches conducted over both the MHAD (a) and MIVIA (b) datasets.

(a)

Method	Source	Accuracy
HARED	Proposed	**87.1**
Bow	(Foggia et al., 2013)	72.9
BM1	(Cheema et al., 2013)	77.7
Deep	(Foggia et al., 2014)	85.8
SMIJ	(Ofli et al., 2012)	**94.2**
HMIJ	(Ofli et al., 2012)	82.9
HMW	(Ofli et al., 2012)	81.1
LDSP	(Ofli et al., 2012)	82.2
BM2	(Cheema et al., 2013)	**87.8**

(b)

Method	Source	Accuracy
HARED	Proposed	**85.2**
Cuboids	(Dollar et al., 2005)	74.4
Reject	(Carletti et al., 2013)	79.8
Bow	(Foggia et al., 2013)	84.1
HAcK	(Brun et al., 2014)	80.1
Deep	(Foggia et al., 2014)	84.7

both the datasets the performance achieved are very stable with respect to the number of the clusters. It makes the proposed approach very suited for real applications, since the set up of the system can be performed in a very simple way also for an unexperienced operators.

Finally, in order to further confirm the effectiveness of the proposed approach, a proper comparison has been carried out with several state of the art approaches, as shown in Table 2. It is worth pointing out that among the methods listed in Table 2a only *Bow*, *BM1* and *Deep* are based on depth information; in fact, the other methods are based on the skeleton acquired by a Mocap system, able to capture 3D position of active LED markers. Even if improving the performance of the action recognition system, this kind of marker can not be easily applied in real environments since it requires ad hoc hardware mounted on the persons under test. Thus, the performance obtained by the proposed approach can be considered even better in the light of the above considerations.

4 CONCLUSIONS

In this paper we proposed HARED, a novel method able to recognize human actions by string edit distance. The main advantages of the proposed approach are the following: first, the string based representation allows the system to explicitly take into account the order of sub-actions within an action; furthermore, the introduction of the edit distance for evaluating the similarity between two strings enables the system to deal with actions of different lengths as well as with changes in the speed of execution; finally, the edit distance, being robust to the presence of spurious symbols, makes the approach suitable to work in noisy environments.

The results obtained by testing the proposed methods over two standard datasets confirm its effectiveness: in fact, the accuracy achieved on the MIVIA dataset (85.2%) is higher with respect to any other considered state of the art method; the one obtained on the MHAD dataset (87.1%) is still higher with respect to all the methods exploiting depth images instead of skeleton information. Furthermore, the stability observed when varying the number of clusters confirms its usability in real applications, where unexperienced human operator may easily set the system up.

Future work will investigate the possibility of further improving the algorithm performance by suitably tuning the edit operation costs during the training phase.

ACKNOWLEDGEMENTS

This research has been partially supported by A.I.Tech s.r.l. (http://www.aitech-solutions.eu).

REFERENCES

Brun, L., Percannella, G., Saggese, A., and Vento, M. (2014). Hack: Recognition of human actions by kernels of visual strings. In *Advanced Video and Signal-Based Surveillance (AVSS), 2014 IEEE International Conference on*.

Carletti, V., Foggia, P., Percannella, G., Saggese, A., and Vento, M. (2013). Recognition of human actions from rgb-d videos using a reject option. In *ICIAP 2013*, volume 8158, pages 436–445.

Cheema, M. S., Eweiwi, A., and Bauckhage, C. (2013). Human activity recognition by separating style and content. *Pattern Recognition Letters*, (0):–.

Chen, Y., Wu, Q., and He, X. (2011). Human action recognition based on radon transform. In *Multimedia Analysis, Processing and Communications*, volume 346, pages 369–389. Springer Berlin Heidelberg.

Dollar, P., Rabaud, V., Cottrell, G., and Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72.

Foggia, P., Percannella, G., Saggese, A., and Vento, M. (2013). Recognizing human actions by a bag of visual words. In *IEEE SMC 2013*.

Foggia, P., Saggese, A., Strisciuglio, N., and Vento, M. (2014). Exploiting the deep learning paradigm for recognizing human actions. In *Advanced Video and Signal-Based Surveillance (AVSS), 2014 IEEE International Conference on*.

Li, W., Yu, Q., Sawhney, H., and Vasconcelos, N. (2013). Recognizing activities via bag of words for attribute dynamics. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2587–2594.

Megavannan, V., Agarwal, B., and Babu, R. (2012). Human action recognition using depth maps. In *SPCOM 2012*, pages 1 –5.

Mokhber, A., Achard, C., Qu, X., and Milgram, M. (2005). Action recognition with global features. In Sebe, N., Lew, M., and Huang, T., editors, *Computer Vision in Human-Computer Interaction*, volume 3766 of *Lecture Notes in Computer Science*, pages 110–119. Springer Berlin Heidelberg.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88.

Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., and Bajcsy, R. (2012). Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. In *CVPRW 2012*, pages 8–13.

Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., and Bajcsy, R. (2013). Berkeley mhad: A comprehensive multimodal human action database. In *WACV*.

Poppe, R. (2010). A survey on vision-based human action recognition. *Image Vision Comput.*, 28(6):976–990.

Sung, J., Ponce, C., Selman, B., and Saxena, A. (2012). Unstructured human activity detection from rgbd images. In *ICRA*, pages 842–849. IEEE.

Turaga, P., Chellappa, R., Subrahmanian, V. S., and Udrea, O. (2008). Machine recognition of human activities: A survey. *IEEE T Circuits Syst*, 18(11):1473–1488.

Vishwakarma, S. and Agrawal, A. (2013). A survey on activity recognition and behavior understanding in video surveillance. *Visual Comput.*, 29(10):983–1009.

Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *J. ACM*, 21(1):168–173.

Wang, Y., Huang, K., and Tan, T. (2007). Human activity recognition based on r transform. In *CVPR 2007*, pages 1 –8.

Weinland, D., Ronfard, R., and Boyer, E. (2011). A survey of vision-based methods for action representation, segmentation and recognition. *Comput. Vis. Image Und.*, 115(2):224 – 241.

Xiao, C., Wang, W., and Lin, X. (2008). Ed-join: An efficient algorithm for similarity joins with edit distance constraints. *Proc. VLDB Endow.*, 1(1):933–944.

Yuan, C., Li, X., Hu, W., Ling, H., and Maybank, S. (2013). 3d r transform on spatio-temporal interest points for action recognition. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 724–730.

Zhang, J. and Gong, S. (2010). Action categorization with modified hidden conditional random field. *Pattern Recogn.*, 43(1):197–203.