A Line Based Pose Representation for Human Action Recognition

Sermetcan Baysal, Pınar Duygulu

Bilkent University, Department of Computer Engineering, 06800, Ankara, Turkey

Abstract

In this paper, we utilize a line based pose representation to recognize human actions in videos. We represent the pose in each frame by employing a collection of line-pairs, so that limb and joint movements are better described and the geometrical relationships among the lines forming the human figure is captured. We contribute to the literature by proposing a new method that matches line-pairs of two poses to compute the similarity between them. Moreover, to encapsulate the global motion information of a pose sequence, we introduce line-flow histograms, which are extracted by matching line segments in consecutive frames. Experimental results on Weizmann and KTH datasets emphasize the power of our pose representation, and show the effectiveness of using pose ordering and line-flow histograms together in grasping the nature of an action and distinguishing one from the others.

Keywords: human motion, action recognition, pose similarity, pose matching, line-flow

Email addresses: `sermetcan@cs.bilkent.edu.tr` (Sermetcan Baysal), `duygulu@cs.bilkent.edu.tr` (Pınar Duygulu)

1. Introduction

Recognizing and analyzing human actions in videos has been receiving increasing attention of computer vision researchers both from academia and industry. A reliable and an effective solution to this problem is essential for a large variety of applications such as athletic performance analysis, medical diagnostics, visual surveillance [1].

However, automatically recognizing human actions in videos is challenging since people can perform the same action in different ways with various execution speeds. Furthermore, recording conditions such as the illuminations or viewpoints may differ as well.

The human brain can more or less recognize what a person is doing in a video even by looking at a single frame without examining the whole sequence. From this observation it can be inferred that the human pose encapsulates useful information about the action being performed. In this study we focus on the representation of actions and use human pose as our primitive representative unit.

Some of the previous studies [4, 5, 27] attempt to represent the shape of a pose by using human silhouettes. Although these approaches are robust to variations in the appearance of actors, they require static cameras and a good background model, which may not be possible under realistic conditions [15]. A more severe limitation of such methods is that they ignore limb movements remaining inside the silhouette boundaries; for example, 'standing still' is likely to be confused with 'hand clapping' when the action is performed facing the camera and hands are in front of the torso.

An alternative shape representation can be established using contour features. Motivated by the work of Ferrari et al. [10], where encouraging results were obtained using line segments as descriptors for object recognition, we represent the shape of a pose as a collection of line segments fitted to the contours of a human figure.

Utilizing only shape information may fail to capture differences between actions with similar pose appearances, such as ‘running’ and ‘jogging’. In such cases the speed and direction of the movement is important in making a distinction. In addition to our pose-based action representation, we also extract global line-flow histograms for a pose sequence by matching lines in consecutive frames in order to identify differences between actions with similar appearances.

The overview of our approach (depicted in Figure 1) is as follows. For each frame, a Contour Segment Network (CSN) consisting of roughly straight lines is constructed. Next, noise elimination is applied and the human figure is detected by utilizing the densest area of line segments. Then an $N \times N$ grid structure is placed over the human figure for localization of the line segments. To obtain the global line-flow of a pose sequence, line displacement vectors are extracted for each frame by matching its set of lines with the ones in the previous frame. Then these vectors are represented by a single compact line-flow histogram. Given a sequence of poses, recognition is performed by combining decisions of separate weighted k-nearest neighbor (k-NN) classifiers for both pose and line-flow features.

In this work, we concentrate on the representation of actions and make two main contributions to the literature. Firstly, we propose a new match-

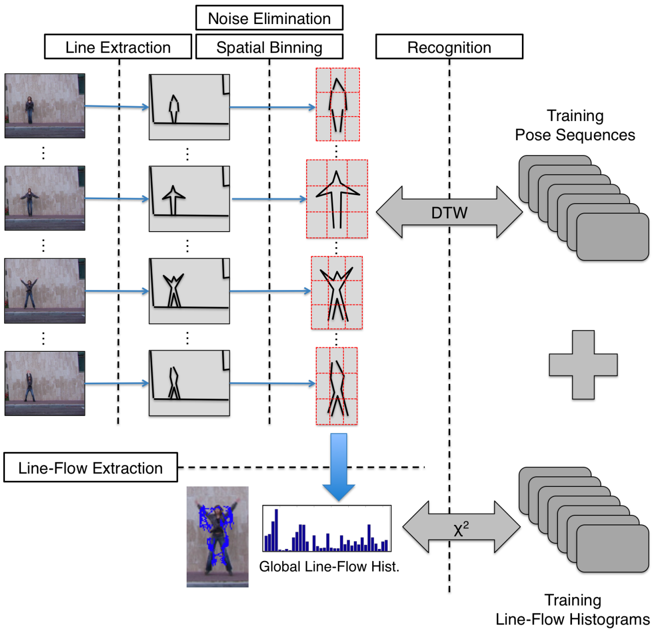


Figure 1: The overview of our approach.

ing method[1] between two poses to compute their similarity. Secondly, we introduce global line-flow to encapsulate motion information for a collection of poses formed by line segments.

2. Related Work

Human action recognition has been a widely studied topic of computer vision. In this section, we will first give a brief review of recent studies focusing on the representation then we will have a discussion.

[1] A preliminary version of this matching method was presented in [3] at International Conference on Pattern Recognition, Istanbul, Turkey, August, 2010.

2.1. Review of Previous Studies

Space-time volumes are utilized for action recognition in the following studies. Blank et al. [4] regard human actions as 3D shapes induced by the silhouettes in the space-time volume. Similarly, Ke et al. [15] segment videos into space-time volumes, however their spatio-temporal shape based correlation algorithm does not require background subtraction.

There are a large number studies which employ space-time interest points (STIP) for action representation. Dollar et al. [7] propose a spatio-temporal interest point detector based on 1D Gabor filters to find local regions of interest in space and time (cuboids) and use histograms of these cuboids to perform action recognition. These linear filters were also applied in [21, 25, 26] to extract STIP. There are also other studies which use different spatio-temporal interest point detectors. Laptev et al. [17] detect interest points using a space-time extension of the Harris operator. However, instead of performing a scale selection, multiple levels of spatio-temporal scales are extracted. The same STIP detection technique is also adopted by Thi et al. in [34]. They extend Implicit Shape Model to 3D, enabling them to robustly integrate the set of local features into a global configuration, while still being able to capture local saliency.

Among the STIP based approaches, [7, 17, 20, 25] quantize local space-time features to form a visual vocabulary and construct a bag-of-words model to represent a video. However, Kovashka et al. and Ta et al. believe that the orderless bag-of-words lacks cues about motion trajectories, before-after relationships and spatio-temporal layout of the local features which may be almost as important as the features themselves. So, Kovashka et al.

[16] propose to learn shapes of space-time feature neighbors that are most representative for an action category. Similarly, Ta et al. [33] present pairwise features, which encode both the appearance and the spatio-temporal relations of the local features for action recognition. In contrast to using hand-designed local features for action recognition, Le et. al. [18] present an extension of the Independent Subspace Analysis algorithm to learn invariant spatio-temporal features from unlabeled video data.

A group of studies use flow-based techniques which estimate the optical field between adjacent frames to represent of actions. In [8], Efros et al. introduce a motion descriptor based on blurred optical flow measurements in a spatio-temporal volume for each stabilized human figure, which describes motion over a local period of time. Wang et al. [38] also use the same motion descriptor for frame representation and represent video sequences by a bag of words representation. Fathi et al. [9] extend the work of Efros to a 3D spatio-temporal volume. Different from the flow-based studies above, Ahmad et al. [2] represent action as a set of multi-dimensional combined local-global (CLG) optic flow and shape flow feature vectors in the spatio-temporal action boundary.

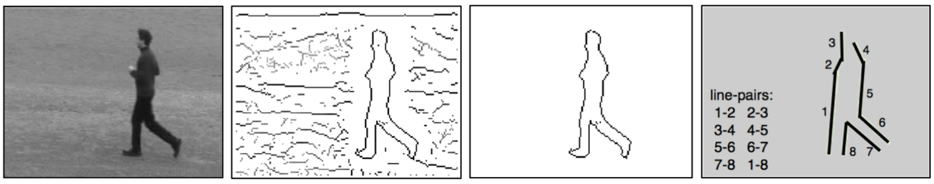
Actions are represented by poses in the following studies. Carlsson et al. [6] demonstrate that specific actions can be recognized by matching shape information extracted from individual frames to stored prototypes representing key frames of an action. Following this study and using the same shape matching scheme, which compares edge maps of poses, Loy et al. [23] present a method for automatically extracting key frames from an image sequence. Ikizler et al. [14] propose a bag-of-rectangles method that represents human body as a collection of rectangular patches and calculate their histograms based on their orientation. Hatun et al. [12] describe pose in each frame using the histogram of gradients (HOG) features obtained from radial partitioning of the frame. Similarly, Thurau et al. [35] extend HOG based descriptor to represent pose primitives. In order to include local temporal context, they compute histograms of n-gram instances. Tran et. al. [36] propose a generative representation of the motion of human body-parts to learn and classify human actions. They transfer motion of different human body-parts into polar histograms.

In another group of studies both shape (pose) and motion (flow) features are combined to represent actions. Ikizler et al.[13] introduce a new shape descriptor based on the distribution of lines fitted to the boundaries of human figures. Poses are represented by employing histogram of lines based on their orientations and spatial locations. Moreover, a dense representation of optical flow and global temporal information is utilized for action recognition. Schindler et al. [30] propose a method that separately extracts local shape, using the responses of Gabor filters at multiple orientations, and dense optic flow from each frame. Then the shape and flow feature vectors are merged by simple concatenation before applying SVM classification for action recognition. Lin et al. [19] capture correlations between shape and motion cues by learning action prototype trees in a joint features space. The shape descriptor is formed by simply counting the number of foreground pixels either in silhouettes or appearance-based likelihoods. Their motion descriptor is an extension of the one introduced by Efros et al. [8], in which background motion components are removed. Shao et. al. [32] propose a color based method and a motion based method for human action temporal segmentation under a stationary background condition. They apply a shape-based feature descriptor: Pyramid Correlogram of Oriented Gradients (PCOG) aiming to detect different action classes within the same video sequence.

2.2. Discussion of Related Studies

Studies of Hatun et al. [12], Ikizler et al.[13, 14] and Thurau et al. [35], share a common property of employing histograms to represent the pose information in each frame. However, using histograms for pose representation results in the loss of geometrical information among the components (e.g. lines, rectangles, gradients) forming the pose. For action recognition such a loss is intolerable since configuration of the components is very crucial in describing the nature of a human action involving limb and joint movements. Representing the pose in a frame as a collection of line-pairs, our work differs from these studies by preserving the geometrical configuration of lines as the components encapsulated in poses.

In this study, we propose to capture the global motion information in a video by tracking line displacements across adjacent frames, which could be compared to optical flow representations in [2, 8, 9, 38]. Although, optical flow often serves as a good approximation of the true physical motion projected onto the image plane; in practice, its computation is susceptible to noise and illumination changes as stated in [37]. Lines are less effected by variations in the appearance of actors and they are easier to track than lower-level features such as color/intensity changes. Thus, we believe that line-flow could be a good alternative to optical flow.



(a) org_img (b) gpb_img (c) edge_img (d) csn_img

Figure 2: This figure illustrates the steps of pose extraction. Given any frame (a), GPB are computed to extract the contours (b). Then hysteresis thresholding is applied to obtain a binary image consisting of edge-pixels (edgels) (c). Next, edgel-chains are partitioned into roughly straight contour segments forming the CSN (d). Finally, CSN is represented by kAS descriptor.

3. Pose Extraction

Before presenting our proposed pose matching method and line-flow histograms, first, we give the details of our line-based pose extraction in this section. Given an action sequence, pose in each frame is extracted as follows (depicted in Figure 2):

1. The global probability of boundaries (GPB), which is presented by Maire et al. as a high-performance detector for contours in natural images (see [24] for details), are computed to extract the edges of the human figure in a frame.
2. To eliminate the effect of noise caused by short and/or weak edges, hysteresis thresholding is applied to obtain a binary image consisting of edge pixels (edgels).
3. Edgels are chained by using closeness and orientation information. The edgel-chains are partitioned into roughly straight contour segments. This chained structure is used to construct a contour segment network

(CSN).

4. The CSN is represented by scale invariant k-Adjacent Segment (kAS) descriptor encoding the geometric configuration of the segments, which was introduced by Ferrari et al. in [10].

As defined in [10], the segments in a kAS form a path of length k through the CSN. Two segments are considered as connected in the CSN, when they are adjacent along some object contour even if there is a small gap separating them physically. More complex structures can be captured as k increases in a kAS. 1AS are just individual lines, 2AS include L-shapes and 3AS can form C, F and Z shapes.

Human pose, especially limb and joint movements, can be better described by using L-shapes. Therefore, in our work we select k=2, and refer to 2AS features as **line-pairs**. Example line-pairs can be seen in Figure 2 (d). As in [10], each line-pair consisting of line segments s_1 and s_2 is represented with the following descriptor:

$$V_{line-pair} = \left(\frac{r_2^x}{N_d}, \frac{r_2^y}{N_d}, \theta_1, \theta_2, \frac{l_1}{N_d}, \frac{l_2}{N_d} \right) \tag{1}$$

where $r_2 = (r_2^x, r_2^y)$ is the vector going from midpoint of s_1 to midpoint of s_2, θ_i is the orientation and $l_i = \|s_i\|$ is the length of s_i $(i = 1, 2)$. N_d is the distance between the two midpoints, which is used as the normalization factor. The center of the two midpoints (center of the vector r_2) is used as the coordinates of the line-pair on the image.

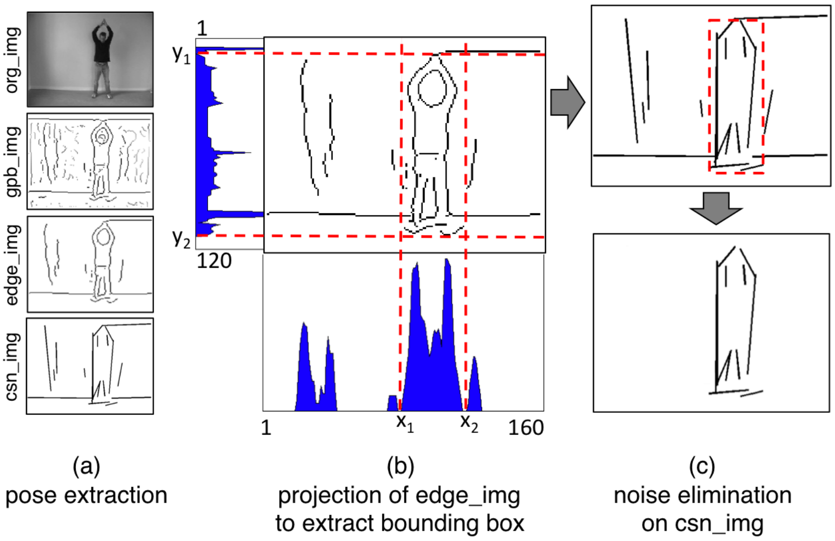


Figure 3: This figure illustrates the steps of noise elimination. Notice that after the pose extraction steps, the CSN contains erroneous line segments that do not belong to the human figure (a). So in (b), *edge_img* is projected onto x and y axes to form a bounding box around the densest area of line segments in the *csn_img*. Line segments that remain outside the bounding box are eliminated form the CSN (c).

3.1. Noise Elimination

Under realistic conditions (varying illumination, cluttered backgrounds, reflection of shadows, etc.) the edge detection results may contain erroneous line segments that do not belong to the human figure. Assuming that the densest area of line segments in the CSN contains the human figure, the following noise elimination steps are applied after pose extraction (depicted in Figure 3):

1. Project *edge_img* onto the x-axis. Then calculate the area under each separate curve peak. Set x_1 and x_2 to be the boundaries of the isolated curve peak with the largest area.

2. Project *edge_img* onto the y-axis. Then calculate the projected length of each seperate curve peak on the y-axis. Set y_1 and y_2 to be the boundaries of the longest isolated curve peak.
3. Place a bounding box on the *csn_img* with (x_1, y_1) and (x_2, y_2) being its upper left and lower right corner coordinates respectively.
4. Recall that the *csn_img* contains a set of line segments such that $csn\_img = \{l_1, l_2, \ldots, l_n\}$. Eliminate a line segment $l_i \in csn\_img$ from the CSN, if its center's coordinates is not in the bounding box.

3.2. Spatial Binning

The descriptor presented in [10] (Equation 1), encodes scale, orientation and length of the line-pairs, but it lacks positional information. Therefore, in order to capture spatial locations of the line-pairs; first, the human figure is cropped from the frame using the bounding box which was previously formed in the noise elimination process. Then, to be used in the latter stages, the human figure is divided into equal-sized spatial bins forming an $N \times N$ grid structure. Finally, each line-pair is assigned to a specific bin depending on its coordinates.

4. Finding Similarity Between Poses

Recall that pose in each frame is represented by a set of line-pair descriptors. The similarity between two line-pair descriptors v_a and v_b is computed by the following formula as suggested in [10]:

$$d_{line-pair}(a, b) = w_r \cdot \|r_2^a - r_2^b\| + w_\theta \cdot \sum_{i=1}^{2} D_\theta(\theta_i^a, \theta_i^b) + \sum_{i=1}^{2} \left|\log(l_i^a / l_i^b)\right| \quad (2)$$

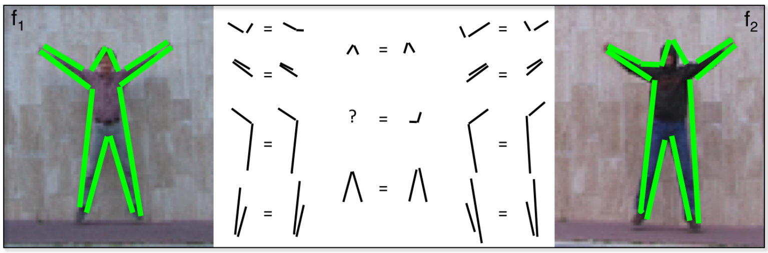


Figure 4: This figure illustrates the matched line-pairs in two frames having similar poses.

where the first term is the difference in the relative location of the line-pairs, the second term measures the orientation difference of the line-pairs and the last term accounts for the difference in lengths. The weights of the terms are $w_r = 4$ and $w_\theta = 2$. Note that Equation 2, proposed in [10], computes the similarity only between two individual line-pairs. However, we need to compare two poses. Therefore, in this paper, we introduce a method to find similarity between two poses consisting of multiple line-pairs.

4.1. Pose Matching

To compute a similarity value between two poses, first of all, we need to find a correspondence between their line-pairs. Any two poses consisting of multiple line-pair descriptors can mathematically be thought of as two sets X and Y with different cardinalities. We seek for a 'one-to-one' match between two subsets $X' \subset X$ and $Y' \subset Y$, so that an element in X' is associated with exactly one element in Y'. For instance, x_i and y_j are matched if and only if $g(x_i) = y_j$ and $h(y_j) = x_i$ where $g : X' \to Y', \quad h : Y' \to X', \quad x_i \in X', \quad y_j \in Y'$.

To describe our pose matching mechanism more formally, let f_1 and f_2 be two poses having a set of line-pair descriptors $V_1 = \{v_1^1, v_2^1, \ldots, v_n^1\}$ and

$V_2 = \{v_1^2, v_2^2, \ldots, v_m^2\}$, where n and m are the number of line-pair descriptors in V_1 and V_2 respectively. We compare each line-pair descriptor $v_i^1 \in V_1$ with each line-pair descriptor $v_j^2 \in V_2$ to find matching line-pairs. v_i^1 and v_j^2 are matched if and only if among descriptors in V_2, v_j^2 has the minimum distance to v_i^1 and among descriptors in V_1, v_i^1 has the minimum distance to v_j^2. To include location information, we apply a constraint in which matching is allowed only between line-pairs within the same spatial bin.

As an output of our pose matching method two matrices, D and M of size $n \times m$, are generated. Distance matrix D stores similarity of each line-pair in f_1 to each line-pair in f_2, where $D(i, j)$ indicates the similarity value between v_i^1 and v_j^2. Match matrix M is a binary matrix, where $M(i, j) = 1$ indicates that i-th line-pair in f_1 and j-th line-pair in f_2 are matched. These matrices are utilized when an overall similarity distance between two poses is calculated.

4.2. Calculating a Similarity Value

Having established a correspondence between poses f_1 and f_2 by matching their line-pairs (as shown in Figure 4), now we need to numerically express this correspondence. The first approach would be to take the average of the matched line-pair distances. This could be calculated by utilizing the matrices D and M as follows:

$$sim_1(f_1, f_2) = \frac{sum(D \wedge M)}{|match(f_1, f_2)|} \tag{3}$$

where $sum(D \wedge M)$ is the sum of distances between matched line-pairs and $|match(f_1, f_2)|$ is the number of matched line-pairs between f_1 and f_2.

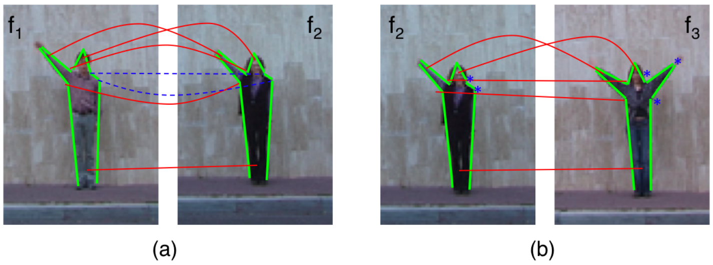


Figure 5: This figure illustrates matched line-pairs in similar (a) and slightly different (b) poses. Red lines (straight) denote the matched line-pairs common in both (a) and (b). Blue lines (dashed) indicate that these line-pairs are only matched in (a). $sim_1(f_1, f_2)$ is calculated by taking the average of red and blue lines (assuming that they represent a distance value between matching line-pairs) and $sim_1(f_2, f_3)$ is calculated by averaging only the red lines. Since red lines are common in both scenarios, similarity distance in (a) may be very close to or even greater than (b) depending on the distances represented by blue lines. Therefore, unmatched line-pairs, shown by blue dots in (b), should be utilized to produce a 'stronger' similarity distance.

The function sim_1, calculates a 'weak' similarity value between f_1 and f_2, since it utilizes distances between only the matched line-pairs. However, poses of distinct actions may be very similar, differing only in configuration of a single limb (see Figure 5). To compute a 'stronger' similarity value, unmatched line-pairs in both f_1 and f_2 should be utilized. Thus, we present another similarity value calculation function sim_2, which assumes that a perfect match between sets X and Y is established when both sets have the equal number of elements and both 'one-to-one' and 'onto' set properties are satisfied, so that each element in X is exactly associated with one element in Y. The function sim_2 calculates the overall similarity distance by penalizing unmatched line-pairs in the frame having more number elements as follows:

$$sim_2(f_1, f_2) = \frac{sum(D \wedge M) + p \cdot (max(m,n) - |match(f_1, f_2)|)}{max(m,n)} \tag{4}$$

where $p = mean(D \wedge \neg M)$ is the penalty value denoting the average dissimilarity between two poses; $max(m, n)$ is the maximum of m and n; m and n is the number of line-pairs in f_1 and f_2 respectively. The penalty value p is computed by excluding the matched line-pair values in the distance matrix D by bitwise-anding it with the complement of the match matrix M and taking average of the positive values. This is simply the average of the pairwise similarity values of all the unmatched line-pairs. Relative performance of sim_1 and sim_2 will be evaluated in Section 7.

5. Line-Flow Extraction

By utilizing only shape information, it is sometimes difficult to distinguish actions having similar poses such as jogging and running. In such cases, the speed of transitions from one pose to the next one is crucial in distinguishing actions. In our work, we characterize this transition by extracting global flow of lines throughout an action sequence. We have experimentally found that flow of lines better describes motion patterns than flow of line-pairs.

Given an action sequence, consecutive frames are compared to find matching lines. The same pose matching method in Section 4.1 is applied, however this time lines are matched instead of line-pairs. To do so, Equation 2 is modified as follows to compute a distance between two line segments :

$$d_{line}(a, b) = w_\theta \cdot D_\theta(\theta^a, \theta^b) + \left|\log(l^a/l^b)\right| \tag{5}$$

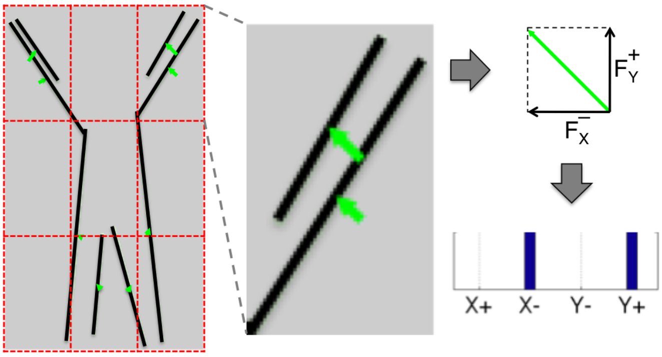


Figure 6: This figure illustrates extraction of line-flow vectors and histograms for a single frame. Given an action sequence, i-th frame is matched with the previous $(i-1)$-th frame. Line-flow vectors (in green) show the displacement of matched lines with respect to the previous frame. Each line-flow vector is then separated into 4 non-negative components. We employ a histogram for each spatial bin to represent these line-flow vectors.

where the first term is the the orientation difference of the lines and the second term accounts for the difference in lengths. The weighting coefficient is $w_\theta = 2$.

As depicted in Figure 6, after finding matches between consecutive frames, the displacement of each matched line with respect to the previous frame is represented by a line-flow vector $\vec{F}$. Then this vector is separated into 4 non-negative components $\vec{F} = \{F_x^+, F_x^-, F_y^+, F_y^-\}$, representing its magnitudes when projected on $x+$, $x-$, $y+$ and $y-$ axes on the xy-plane. For each j-th spatial bin, where $j \in \{1, \ldots, N \times N\}$, we define line-flow histogram $h_j(i)$ as follows:

$$h_j(i) = \sum_{k \in B_j} \vec{F}_k \tag{6}$$

where $\vec{F}_k$ represent a line-flow vector in spatial bin j. B_j is the set of flow vectors in spatial bin j and $i \in \{1, \dots, n\}$, where n is the number of frames in the action sequence. To obtain a single line-flow histogram $h(i)$ for the i-th frame, we concatenate line-flow histogram h_j of each spatial bin j.

6. Recognizing Actions

Given the details of our feature extraction steps in the previous sections, we now describe our action recognition methods in the following subsections.

6.1. Using Pose Ordering

In this classification method, recognition is performed by comparing two action sequences and finding a correspondence between their pose orderings. However, comparing two pose sequences is not straightforward since actions can be performed with various speeds and periods, resulting in sequences with different lengths. Therefore, first we align two sequences by means of Dynamic Time Warping (DTW) [28] and then utilize the distance between aligned poses to derive an overall similarity.

DTW is an algorithm to compare time series and find the optimal alignment between them by means of dynamic programming. As formalized in [29], given two action sequences $A = a_1, a_2, \dots, a_i, \dots, a_{|A|}$ and $B = b_1, b_2, \dots, b_j, \dots, b_{|B|}$ of lengths $|A|$ and $|B|$, DTW constructs a warp path $W = w_1, w_2, \dots, w_K$ (depicted in Figure 7) where K is the length of the warp path and $w_k = (i, j)$ is the k-th element of warp path indicating that the i-th element of A and the j-th element of B are aligned. Using the aligned poses, the distance between two action sequences A and B is calculated as follows:

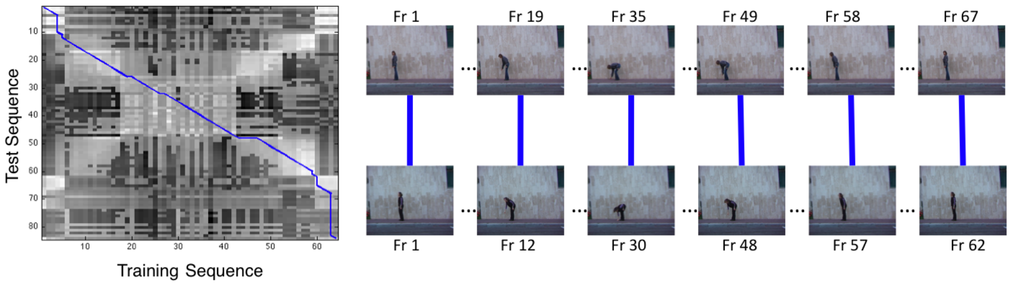


Figure 7: This figure illustrates the alignment of two action sequences. Frame-to-frame similarity matrix of two actions can be seen **on the left**. Brighter pixels indicate smaller similarity distances (more similar frames). The 'blue line' overlaid on the matrix indicates the warp path obtained by DTW. The frame correspondence based on the alignment path is shown **on the right**.

$$Dist_{DTW}(A,B) = \frac{\sum_{k=1}^{K} dist(w_{ki}, w_{kj})}{K} \tag{7}$$

where $dist(w_{ki}, w_{kj})$ is the distance between two frames $a_i \in A$ and $b_j \in B$, which are aligned at the k-th index of the warp path, calculated using our pose matching function. Refer to [29] for the details of finding the minimum-distance warp path using a dynamic programming approach.

We use a weighted k-NN classifier, which assigns a given test pose sequence to the class most common amongst its k nearest training pose sequences using $Dist_{DTW}$ (Equation 7) as its distance metric. In addition we weight the contributions of the neighbors by $1/d$, where d is the distance to the test sequence, so that nearer neighbors contribute to the decision more than the distant ones. We denote this classifier as $\mathbf{c}_{pose}$ to be used in Section 6.3.

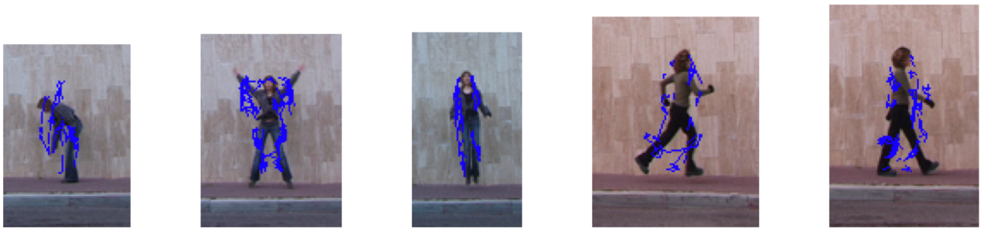


Figure 8: This figure illustrates the global line-flow of different actions (**from left to the right**): bend, jumping jack, jump in place, running and walking. Notice that the line-flow vectors are in different orientations in different spatial locations so that 'bend', 'jumping jack' and 'jump in place' can be easily distinguished. Although, the global line-flow of 'running' and 'walking' seem similar, notice the difference in the density of the lines. Sparser lines represent faster motion, whereas dense lines represent actions with slower motion.

6.2. Using Global Line-Flow Histograms

In Section 5, the extraction of a line-flow histogram $h(i)$ for a single frame was shown. In order to represent a video, we simply sum up line-flow histograms of each frame to from a single compact representation of the entire action sequence consisting of n frames as follows:

$$H = \sum_{i=1}^{n} h(i) \tag{8}$$

We compute the flow similarity between two action sequences A and B by comparing their global line-flow histograms H_a and H_b using chi-square distance χ^2 as follows:

$$\chi^2(A, B) = \frac{1}{2} \sum_{j} \frac{(H_a(j) - H_b(j))^2}{H_a(j) + H_b(j)} \tag{9}$$

where $j \in 1, 2, \ldots, k$ and k is the number of bins in the histogram.

In order to classify a given pose sequence, we employ a weighted k-NN classifier (as in Section 6.1) which uses χ^2 (Equation 9) as its distance metric. This classifier is denoted as $\mathbf{c}_{flow}$ to be used in Section 6.3. The global line-flow of different actions can be seen in Figure 8.

6.3. Using Combination of Pose Ordering and Line-Flow

In the previous sections, two action recognition methods were introduced. The first one utilizes pose ordering of an action sequence and the second one captures the global motion cues by using line-flow histograms. These two methods are combined in this final classification scheme, in order to overcome limitations of either shape or flow-based behaviors and achieve a higher accuracy.

To classify a given pose sequence, we employ decision vectors $\vec{\mathbf{d}}_{pose}$ and $\vec{\mathbf{d}}_{flow}$, generated by the weighted k-NN classifiers $\mathbf{c}_{pose}$ (see Section 6.1) and $\mathbf{c}_{flow}$ (see Section 6.2) respectively. Each decision vector is normalized such as $\vec{\mathbf{d}}(i) \in [0,1]$ for all $i \in \{1, 2, \ldots, n\}$, where $\vec{\mathbf{d}}(i)$ is the probability of the test sequence belonging to the i-th class and n is the number of classes. We combine the two normalized decision vectors $\vec{\mathbf{d}}_{pose}$ and $\vec{\mathbf{d}}_{flow}$ using a simple linear weighting scheme to obtain the final decision vector $\vec{\mathbf{d}}_{combined}$ as follows:

$$\vec{\mathbf{d}}_{combined} = \alpha \cdot \vec{\mathbf{d}}_{pose} + (1 - \alpha) \cdot \vec{\mathbf{d}}_{flow} \tag{10}$$

where α is the weighting coefficient of the decision vectors. It determines the relative influence of pose (shape) and line-flow (motion) features on the final classification. Finally, the test pose sequence is assigned to the class having the highest probability value in the combined decision vector $\vec{\mathbf{d}}_{combined}$. The effect of choosing α will be evaluated in Section 7.

7. Experiments

In this section we evaluate the performance of our approach. First we introduce the state-of-art action recognition datasets. Then we give details of our experiments and results. Finally, we compare our results to the related studies and provide a discussion.

7.1. Datasets

In our experiments, we evaluate our method on the Weizmann and the KTH datasets, which are currently considered as the benchmark datasets for single-view action recognition. We adopt leave-one-out cross validation as our experimental setup on all the datasets in order to compare our performance fairly and completely with other studies as recommended in [11].

7.1.1. Weizmann Dataset

This single-view dataset was introduced by Blank et al. in [4] containing 10 actions performed by 9 different actors. We use the same set of 9 actions for our experiments as in [4]; which are bend, jumping jack (jack), jump forward (jump), jump in place (pjump), run, gallop sideways (side), walk, one-hand wave (wave1) and two-hands wave (wave2). Example frames are shown in Figure 9. For this dataset we used the available silhouettes, which were obtained using background subtraction, and applied canny edge detection to extract edges. So we start our pose extraction process (see Section 3) from step 3.

7.1.2. KTH Dataset

This dataset was introduced by Schuldt et al. in [31]. It contains 6 actions: boxing, hand clapping, hand waving, jogging, running and walking.

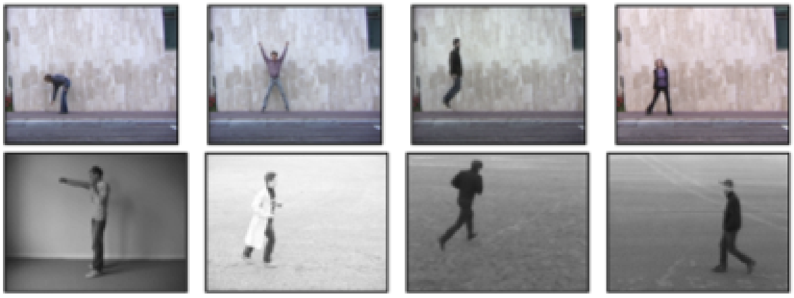


Figure 9: Example frames are shown from different datasets. **Top row**: Weizmann, **bottom row**: KTH.

Each action is performed by 25 subjects in 4 different shooting conditions: outdoor recordings with a stable camera (**sc1**), outdoor recordings with camera zoom effects and different viewpoints (**sc2**), outdoor recordings in which the actors wear different outfits and carry items (**sc3**), indoor recordings with illumination changes and shadow effects (**sc4**). Example frames are shown in Figure 9. KTH is considered as a more challenging dataset compared to Weizmann due to its different realistic shooting conditions. In addition, it contains two similar actions: jogging and running. In this dataset, the length of a video generally exceeds 100 frames and actions are performed multiple times in a video. In order to reduce extensive computational cost, we trim the action sequences to 20-50 frames for our experiments so that an action in a video is performed only once or twice. Note that since global line-flow histogram is calculated from all the frames in an action video, the number of frames influences the results. So trimming action sequences to a specific number of repetitions results in better performance of the global line-flow approach. Although the action sequences were segmented manually in our experiments, this can be easily automatized. DTW already contains the required information since it seeks for an alignment between two pose sequences. We can first apply DTW detect the matching subsequence and extract the line-flow using only those poses.

7.2. Experimental Results

In this section, we present the experimental results evaluating our approach in recognizing human actions. First, the effect of applying spatial binning is examined (Section 7.2.1). Then, pose matching is evaluated and the optimal configuration of our pose similarity calculation function is founded (Section 7.2.2). Next, pose and flow features are evaluated (Section 7.2.3); and the effect of applying noise elimination is discussed (Section 7.2.4). Afterwards, regarding classification, the weighting between pose ordering and line-flow is examined. Finally, computational cost of approach is addressed (Section 7.2.6).

7.2.1. Evaluation of Spatial Binning

Recall that in Section 3.2, we place an $N \times N$ imaginary grid structure over the human figure in order to capture the locations of line segments in a frame. The choice of N is important, because in our pose matching method we only allow matching between line-pairs within the same spatial bin. Similarly, during line-flow extraction between consecutive frames, lines are required to be in the same spatial bin in order to be matched. More importantly, since a line-flow histogram is extracted for each spatial bin, the choice of N directly effects the size of the global line-flow feature vector.

Table 1 compares the use of different-sized grid structures. The worst results are obtained when $N = 1$, which means that no spatial binning is used and matching is allowed between lines or line-pairs located anywhere in the frame. $N = 2$ gives better results compared to no spatial binning and the best results are obtained when a 3×3 grid structure is placed over the human figure. This justifies that the spatial locations of the line-pairs provide useful clues when comparing poses. Regarding line-flow, we can infer from the results that using spatial binning and histogramming line-flow in each spatial bin better describes the local motion of separate body parts.

7.2.2. Evaluation of Pose Similarity Calculation Function

In addition to our classification methods, in order to experimentally evaluate our pose matching mechanism, we employ a single pose (SP) based classification scheme. This experiment discards the order of poses and performs classification based on individual votes of each frame. Therefore, the performance of this method directly depends on the accuracy of our pose matching.

Given a sequence of images $A = \{a_1, a_2, \ldots, a_n\}$ to be classified as one of the available classes $C = \{c_1, c_2, \ldots, c_m\}$, we calculate the similarity distance $d_i(j)$ of each frame $a_i \in A$ to each class $c_j \in C$, by finding the most similar training frame from class c_j (depicted in Figure 10). In order to classify A, we seek for the class having smallest average distance, where the average distance to each class $c_j \in C$ is computed as follows:

$$D(j) = \frac{\sum_{i=1}^{n} d_i(j)}{n} \tag{11}$$

Observing the results of single pose based classification on Table 1, we can say that it achieves acceptable results on both of the datasets consider-

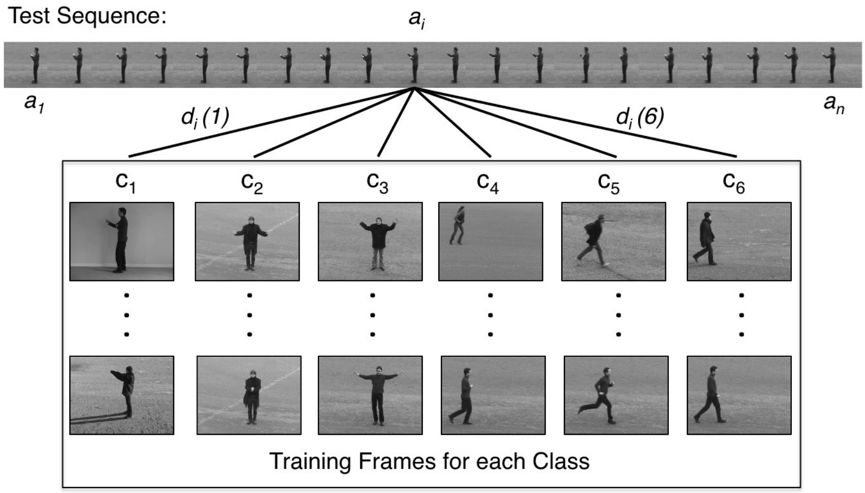


Figure 10: This figure illustrates the classification of an action sequence utilizing only single pose information throughout the video. For each frame in the test sequence, its distance to each class is computed by finding the most similar training frame from that class. In order to classify the sequence, we take the average distance of all frames to each class and assign the class label with the smallest average distance.

ing that the ordering of the poses is totally discarded in this classification method. This demonstrates the power of pose matching mechanism since the performance of this method mainly depends on the accuracy of our pose similarity function. A higher accuracy is obtained by sim_2 because of its strict constraints on pose matching which results in a 'stronger' function. More importantly, when comparing test poses to the stored templates, there is always a frame obeying these strict constraints, since the single pose based classification seeks for a matching pose within the set of all training frames.

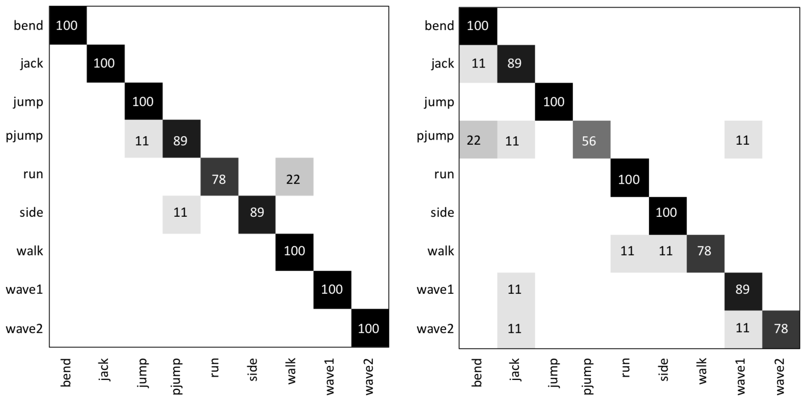
After finding a correspondence between two poses by matching their line-

		Weizmann			**KTH**		
		$N = 1$	$N = 2$	$N = 3$	$N = 1$	$N = 2$	$N = 3$
SP	sim_1	64.2%	71.6%	74.1%	61.7%	63.3%	66.3%
	sim_2	92.6%	92.6%	**93.8%**	71.3%	75.0%	**75.3%**
PO	sim_1	69.1%	81.5%	85.2%	74.3%	77.2%	**81.3%**
	sim_2	92.6%	92.6%	**95.1%**	56.2%	68.5%	73.3%
LF		48.1%	64.2%	**87.7%**	71.3%	74.8%	**80.5%**

Table 1: Action recognition accuracies on Weizmann and KTH datasets using different classification methods (SP: Single Pose, PO: Pose Ordering, LF: Line-Flow) with respect to choice of pose similarity calculation functions (sim_1 and sim_2) and different spatial binnings ($N \times N$). N=1 indicates that no spatial binning is applied.

pairs, in order to calculate an overall similarity between the frames, two pose similarity calculation functions were introduced in Section 4.1. Recall that sim_1 utilizes only the distances between matching line-pairs, whereas sim_2 also penalizes the unmatched line-pairs. Table 1 compares the relative performances of these functions.

When pose ordering classification is used, notice that the accuracy of sim_1 significantly increases for both of the datasets; whereas sim_2 is about the same for Weizmann, but decreases so that its below sim_1 for KTH dataset. First of all, the increase in the accuracy of sim_1, shows the importance of including the ordering of poses in action recognition. Regarding the performance of sim_2, we can say that since the data is 'clean' in the Weizmann dataset, similar pose sequences can still be found under strict matching constraints, which slightly increases the accuracy. However, the accuracy of sim_2 drops below sim_1 for the KTH dataset. This means that requiring



	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	100								
jack		100							
jump			100						
pjump			11	89					
run					78		22		
side				11		89			
walk							100		
wave1								100	
wave2									100

(a) Pose ordering: 95.1%

	bend	jack	jump	pjump	run	side	walk	wave1	wave2
bend	100								
jack	11	89							
jump			100						
pjump	22	11		56				11	
run					100				
side						100			
walk					11	11	78		
wave1		11						89	
wave2		11						11	78

(b) Global line-flow: 87.7%

Figure 11: Confusion matrix of each classification method for the Weizmann dataset. Misclassifications of pose ordering (PO) method belong to actions having similar poses such as in run and walk, pjump and jump, pjump and side (both include standing still human poses). Global line-flow (LF) confuses actions having similar line-flow directions and magnitudes in the same spatial bin, however its set of misclassifications do not overlap with PO. Therefore, when they are combined in PO+LF, we obtain a perfect accuracy of 100%.

strict matching constraints when comparing two poses in a 'noisy' dataset, results in addition of unrealistic penalty due to the high number of unmatched line-pairs that actually do not even belong to the human figure.

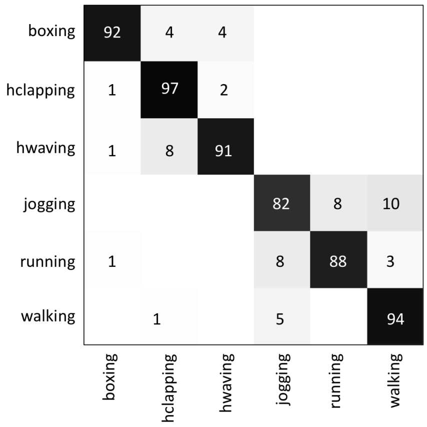
In summary, sim_2 is more accurate when the edges of the human figure are successfully extracted and at classifying individual poses when pose ordering is not available. However, it is wiser to employ sim_1 in more realistic data. Hence, sim_2 function is used in the Weizmann dataset where the edges are extracted from background subtracted silhouettes; sim_1 is used in the KTH dataset where edges are extracted from contour information.

7.2.3. Evaluation of Pose and Flow Features

Having decided on the optimal spatial binning value and chosen a suitable pose similarity calculation function depending on the conditions, in this section, we evaluate the performance of pose and flow features in recognizing human actions on single-camera datasets by comparing the action recognition accuracies of different classification methods, namely, pose ordering (**PO**), global line-flow (**LF**) and combination of pose ordering and global line-flow (**PO+LF**).

Confusion matrices for the Weizmann dataset in Figure 11 contain insightful information to compare pose and flow features by examining the misclassifications made by each recognition method. As expected, the best results are achieved when pose information is combined with global motion cues as in the PO+LF classification method, in which we obtain a perfect accuracy of **100%**. We achieve an overall recognition rate of **90.7%** using PO+LF on KTH dataset (Figure 12 shows the misclassifications). The decrease in the performance with respect to the Weizmann dataset is reasonable, considering the relative complexity of the KTH dataset.

Figure 13 compares recognition performances on individual scenarios of the KTH dataset. As expected, the highest performance is obtained in sc1, which is the simplest scenario of the KTH dataset. This shows that combination of pose ordering and line-flow features can achieve high recognition rates when line segments are accurately extracted. The second and third highest performances are obtained in sc3 and sc4 respectively. Notice that, in these scenarios the accuracy of pose ordering is lower than global line-flow. This can be explained by the decrease in the performance of our pose match-



	boxing	hclapping	hwaving	jogging	running	walking
boxing	92	4	4			
hclapping	1	97	2			
hwaving	1	8	91			
jogging				82	8	10
running	1			8	88	3
walking		1		5		94

Figure 12: Confusion matrix of PO+LF classification method for the KTH dataset. The average of all scenarios accuracy we achieve in this dataset is 90.7%. Most of the confusions occur among jogging, running and walking, which is quite reasonable considering their visual similarity.

ing, due to the different outfits (e.g. long coats) worn by actors resulting in unusual configuration of line segments in sc3; and due to the existence of erroneous line segments belonging to the floor and shadows reflected on the walls in sc4. In contrast, performance of line-flow is lower than pose ordering in sc2, which implies that zooming and viewpoint variance has a negative effect on line-flow extraction. Although the relative performances of pose ordering and line-flow alter from one scenario to another, the overall accuracy is always boosted when these features are combined together in PO+LF classification method.

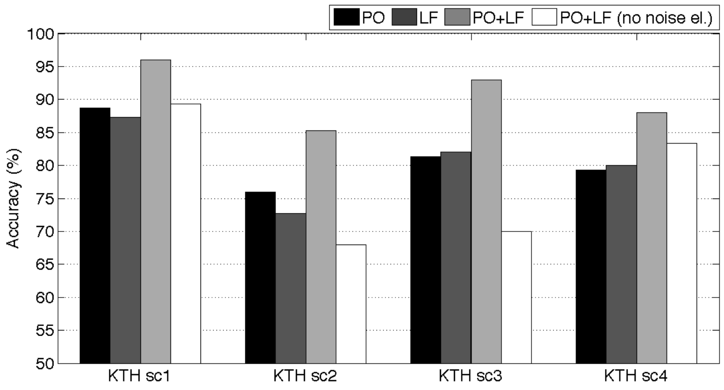


Figure 13: Recognition accuracies on each scenario in the KTH dataset using different classification methods. 'White bars' show the overall accuracy when noise elimination is not applied. In addition, spatial binning is also omitted since a bounding box around the human figure can not be formed. It is apparent that applying noise elimination and then spatial binning significantly improves the performance in all of the scenarios.

7.2.4. Effect of Noise Elimination

To evaluate the effect of our noise elimination algorithm (see Section 3.1), we test our approach without applying any noise elimination. Note that, when noise elimination is not applied we can not form a bounding box around the human figure so that spatial binning is also omitted in this case. Figure 13 reports the overall accuracy of our approach in each scenario of the KTH dataset when noise elimination is not applied. It is obvious that, applying noise elimination and spatial binning significantly improves the recognition rate of each scenario. More specifically, our approach is less effected by noise in the standard outdoor (sc1) and indoor (sc4) settings. However, the recognition rates on sc2 and sc3 are significantly effected by noise due to existence of cluttered backgrounds in these conditions, resulting in inaccurate line segments.

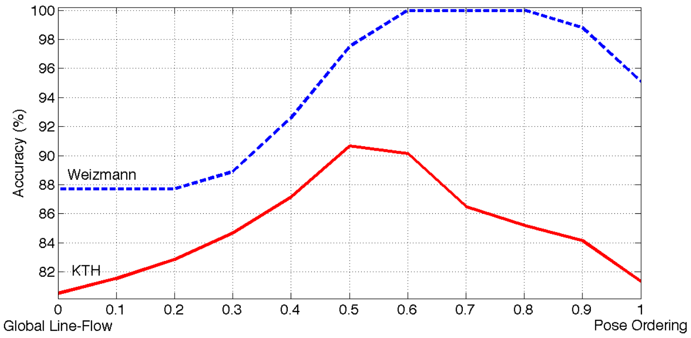


Figure 14: This graph shows the change in the recognition accuracy on the Weizmann and KTH datasets with respect to choice of α (weighting coefficient). $\alpha = 0$ means that only line-flow features are used, whereas $\alpha = 1$ corresponds to using only pose ordering information.

7.2.5. Weighting Between Pose Ordering and Line-Flow

Recall that in the PO+LF classification method (see Section 6.3), pose ordering is combined with global line-flow features in a linear weighting scheme where α is the weighting coefficient in this combination, which determines the influence of individual components on the final classification decision. Figure 14 shows the change in recognition rates with respect to choice of α.

In the KTH dataset, the individual performances of pose ordering and global line-flow are about the same. So the best accuracy is achieved when they are combined with equal weights at $\alpha = 0.5$. We obtain similar results to those of Ikizler et al. [13] finding the best combination of line and optic-flow features at $\alpha = 0.5$. This is also in agreement with the observations of Ke et al. [15], stating that the shape and motion features are complimentary to each other.

The perfect accuracy rate of 100% is reached on Weizmann dataset, when

Operation	Execution Time (s)	per
Edge detection	0.01100	frame
Line-pair extraction	1.28000	frame
Noise elimination	0.46000	frame
Compare line-pairs in two frames	0.00090	frame-pair
Line-flow histogram extraction	0.02980	video
Compare two line-flow histograms	0.00030	hist-pair
DTW classification	0.00320	video
Line-Flow classification	0.00004	video
DTW + Line-Flow classification	0.00329	video

Table 2: Computational cost of our approach on the KTH dataset.

pose ordering has more influence on the final classification decision. This is because, the individual performance of pose ordering is better than line-flow, since actions are mostly differentiable based on their appearances in the Weizmann dataset.

7.2.6. Computational Cost

We have implemented our method in MATLAB and have not applied any significant optimizations. Table 2 shows the computational cost of our approach on the KTH dataset for pose extraction, similarity matrix calculation and classification. All the results are obtained using a 2.2 Ghz Intel Core I7 laptop and are averaged over all frames/videos.

Our pose extraction process consisting of edge detection, line-pair extraction and noise elimination takes about 1.75 seconds per frame. These steps are the most time consuming ones of our approach. However, their running times can be dramatically reduced when coded in OpenCV or when GPU is utilized. Having extracted the line-pairs in each frame of a video, comparing the line-pairs in two frames takes about 0,0009 seconds. The number of line-pairs in a frame for different actions vary between 10 and 20, so that in the worst case of comparing all the possible line-pairs in two frames takes about 0.36 seconds, whereas in the best case it takes about 0.09 seconds. The remaining operations are significantly simpler in terms of computational cost than the previous steps. Extracting line-flow histograms from a video using all the frames takes about 0.03 seconds on the average and comparing line-flows of two videos takes 0.0003 seconds. You can see that the computational costs of the classification steps are negligible after computing the similarity matrices in the previous steps.

8. Comparison to Related Studies

In this section, we compare our method's performance to other studies in the literature that reported results on the KTH dataset. A comparison of results over the Weizmann dataset is not given since most of the recent approaches, including ours, obtain perfect recognition rates on this simple dataset. A comparison over the KTH dataset is given, although making a fair and an accurate one is difficult since different researches employ different experimental setups. As stated by Gao et al. in [11], the performances on the KTH dataset can differ by 10.67%, when different n-fold cross-validation methods are used. Moreover, the performance is dramatically effected by the choice of scenarios used in training and testing. To evaluate our approach, as recommended in [11], we use a simple leave-one-out as the most easily

Method	**Evaluation**	**Accuracy (%)**
Lin [19]	leave-one-out	95.77%
Ta [33]	leave-one-out	93.00%
Liu [20]	leave-one-out	91.80%
Wang [38]	leave-one-out	91.20%
Our Approach	**leave-one-out**	**90.70%**
Fathi [9]	split	90.50%
Ahmad [2]	split	88.33%
Nowozin [26]	split	87.04%
Niebles [25]	leave-one-out	83.30%
Dollar [7]	leave-one-out	81.17%
Ke [15]	leave-one-out	80.90%
Liu [22]	leave-one-out	73.50%
Schuldt [31]	split	71.72%

Table 3: Comparison of our approach to other studies over the KTH dataset.

replicable clear-cut partitioning.

In Table 3, we compare our method's performance to the results of other studies on the KTH dataset. Our main concern in this study is to present a new pose representation, but still our action recognition results are higher than a considerable number of studies. Taking into account its straightforward approach in combining pose and line-flow features, our results are also comparable to the best ones [19, 20, 33, 38]. Although our recognition results are slightly lower than these top studies, we claim that our approach is advantageous in terms of its pose (shape) representation. In order to demon-

	Dataset					
	Weizmann			KTH		
	s	m	$s+m$	s	m	$s+m$
Our Study	**95.1%**	87.7%	100%	**81.3**%	80.5%	90.7%
Lin et. al. [19]	81.1%	88.9%	100%	60.9%	86.0%	95.8%

Table 4: Comparison of our results to [19], with respect to different features: shape only (s), motion only (m), combined shape and motion ($s+m$). For our study, s and m refer to pose ordering and line-flow respectively. Although, s+m results reported in [19] are higher than our PO+LF, we outperform their shape-only results by more than 20% on KTH and by 14% on Weizmann dataset using our pose-ordering based classification.

strate this, we provide a detailed comparison in Table 4 with the work of Lin et. al. [19], which lies at the top position of our rankings table for the KTH dataset. From Table 4 we can observe that the combined shape and motion result reported in [19] (95.8%) is better than our PO+LF classification (90.7%) on the KTH dataset. Although, motion-only action recognition of [19] performs slightly better than our global line-flow based classification; we outperform their shape-only results by more than 20% on KTH and by 14% on Weizmann dataset using our pose ordering based classification. This reflects the effectiveness of the pose features and pose similarity function presented in this study. It also reveals the disadvantage of our linear classification results scheme when compared to the action prototype-tree learning approach used in [19] to combine shape and motion features. So in these experiments, we have demonstrated the potential of our line-pair features in human pose representation. We will further research and expand our studies on classification techniques for better utilization of these features which will

Flow-Based Feature	**Classification**	**Accuracy (%)**
Codebook from optic flow in [38]	S-LDA	91.20 %
Mid-level motion features in [9]	AdaBoost	90.5%
Codebook from optic flow in [38]	S-CTM	90.33 %
Optic flow histogram in [13]	SVM	$\approx$ 84%
Global line-flow	k-NN	80.5%
Flow based correlation in [15]	SVM	$\approx$ 70%

Table 5: Comparison of our global line-flow to other flow-based studies over the KTH dataset.

results in higher recognition performance.

In Table 5, we compare the performance of global line-flow with the results of other flow-based studies on the KTH dataset. Examining the results, we can say that global line-flow histograms perform better than flow-based correlation method used in [15]. In the study of Ikizler et. al. [13] a similar approach is used for flow-based action recognition. They utilize optic flow histograms and perform action recognition using an SVM classifier. Our 80.5% recognition rate is close to the result reported in [13]; we believe that the minor difference is due to our simple k-NN classification scheme when compared to the SVM classifier. The top studies in Table 5 [9, 38] use optic flow as their low-level features and build mid-level features and codebook on top of them. In addition they use more sophisticated classification methods compared to k-NN to get the maximum out of their flow features. Recall that for motion-only classification, we simply aggregate our line-flow vectors extracted from each frame into a histogram. So, in our feature studies we plan to exploit line-flow features by seeking for alternative approaches to global histogramming and by using more complicated classification methods.

9. Conclusions

In this paper, we introduce a line based pose representation and explore its ability in recognizing human actions. We encapsulate a human pose into a collection of line-pairs, preserving the geometrical configurations of the components forming the human figure. The correspondences between the set of line-pairs in two frames are captured by means of the proposed matching mechanism, in order to compute a pose similarity. To include the ordering of poses, we compare two sequences and find the optimal alignment between them using Dynamic Time Warping. In addition to our pose-based representation, the speed and direction of movement in an action sequence is embodied into global line-flow histograms. Experimental results show that combination of pose ordering and line-flow features overcome the limitations of either shape or motion behaviors, thus increase the overall recognition accuracy. When our approach is compared to the other studies combining shape and motion features, we observe that they obtain higher accuracies using features with relatively lower individual performances. This reflects the effectiveness of our pose and motion features; also reveals the disadvantage of our simple combination scheme. It is apparent that the overall recognition rates could be increased by employing a more complex method to combine pose ordering and line-flow features.

The limitation of our approach is that it relies on good edge detection so that Contour Segment Networks consisting of accurate lines can be constructed for each frame. The experiments on the Weizmann and KTH datasets, show that our approach can successfully distinguish actions with high recognition rates when the lines are accurately extracted. However, our pose matching performance is negatively affected when the number of erroneous line segments in each frame increases. Although line-flow is less tolerant to zoom effects than pose features, it performs better under noisy conditions.

In this study we mainly concentrated on the representation of actions. As future work many improvements can be made regarding classification. First, a more sophisticated method can be developed for combining pose ordering and line-flow features. Second, to always extract accurate lines, edge detection scheme can be specialized just for human actions. Finally, our powerful pose matching mechanism can be applied to recognize actions in still images.

References

[1] J. K. Aggarwal and Q. Cai. Human motion analysis: a review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

[2] M. Ahmad and S. Lee. Human action recognition using shape and clg-motion flow from multi-view image sequences. *Pattern Recognition*, 41(7):2237–2252, 2008.

[3] S. Baysal, M. C. Kurt, and P. Duygulu. Recognizing human actions using key poses. In *ICPR*, 2010.

[4] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *ICCV*, 2005.

[5] A. F. Bobick and J. W. Davis. The recognition of human motion using temporal templates. *Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[6] S. Carlsson and J. Sullivan. Action recognition by shape matching to key frames. In *Workshop on Models versus Exemplars in Computer Vision*, 2001.

[7] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, 2005.

[8] A. A. Efros, A. C.Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, 2003.

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

[10] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Trans. Pattern Anal. Mach. Intel l.*, 30(1):36–51, 2008.

[11] Z. Gao, M. Chen, A. G. Hauptmann, and A. Cai. Comparing evaluation protocols on the kth dataset. *Human Behavior Understanding LNCS*, 6219:88–100, 2008.

[12] K. Hatun and P. Duygulu. Pose sentences: a new representation for action recognition using sequence of pose words. In *ICPR*, 2008.

[13] N. Ikizler, R. G. Cinbis, and P. Duygulu. Human action recognition with line and flow histograms. In *ICPR*, 2008.

[14] N. Ikizler and P. Duygulu. Human action recognition using distribution of oriented rectangular patches. *Image and Vision Computing*, 27(10), 2009.

[15] Y. Ke, R. Sukthankar, and M. Hebert. Spatio-temporal shape and flow correlation for action recognition. In *Visual Surveillance Workshop*, 2007.

[16] A. Kovashka and K. Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *CVPR*, 2010.

[17] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.

[18] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

[19] Z. Lin, Z. Jiang, and L. S. Davis. Recognizing actions by shape-motion prototype trees. In *ICCV*, 2009.

[20] J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos in the wild. In *CVPR*, 2009.

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

[22] J. Liu, J. Yang, Y. Zhang, and X. He. Action recognition by multiple features and hyper-sphere multi-class svm. In *ICPR*, 2010.

[23] G. Loy, J. Sullivan, and S. Carlsson. Pose based clustering in action sequences. In *Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, 2003.

[24] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik. Using contours to detect and localize junctions in natural images. In *CVPR*, 2008.

[25] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3):299–318, 2008.

[26] S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classication. In *ICCV*, 2007.

[27] H. Qu, L. Wang, and C. Leckie. Action recognition using space-time shape difference images. In *ICPR*, 2010.

[28] L. Rabiner and B. Juang. *Fundamentals of speech recognition.* Prentice-Hall, 1993.

[29] S. Salvador and P. Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. In *KDD Workshop on Mining Temporal and Sequential Data*, 2004.

[30] K. Schindler and L. V. Gool. Action snippets: how many frames does human action recognition require? In *CVPR*, 2008.

[31] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *ICPR*, 2004.

[32] L. Shao, L. Ji, Y. Liu, and J. Zhang. Human action segmentation and recognition via motion and shape analysis. *Pattern Recognition Letters*, 33:438–445, 2012.

[33] A. Ta, C. Wolf, G. Lavoue, A. Baskurt, and J. Jolion. Pairwise features for human action recognition. In *ICPR*, 2010.

[34] T. Thi, L. Cheng, J. Zhang, L. Wang, and S. Satoh. Weakly supervised action recognition using implicit shape models. In *ICPR*, 2010.

[35] C. Thurau and V. Hlavac. Pose primitive based human action recognition in videos or still images. In *CVPR*, 2008.

[36] K. N. Tran, I. A. Kakadiaris, and S. K. Shah. Part-based motion descriptor image for human action recognition. *Pattern Recognition*, 45(7):2562–2572, 2012.

[37] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: a survey. *Circuits and Systems for Video Technology*, 18(11):1473–1488, 2008.

[38] Y. Wang, P. Sabzmeydani, and G. Mori. Semi-latent dirichlet allocation: a hierarchical model for human action recognition. In *ICCV Workshop on Human Motion*, 2007.