2013 IEEE International Conference on Computer Vision

The *Moving Pose:* An Efficient 3D Kinematics Descriptor for Low-Latency Action Recognition and Detection

Mihai Zanfir[1], Marius Leordeanu[1], Cristian Sminchisescu[2,1]
[1]Institute of Mathematics of the Romanian Academy
[2]Department of Mathematics, Faculty of Engineering, Lund University

Abstract

Human action recognition under low observational latency is receiving a growing interest in computer vision due to rapidly developing technologies in human-robot interaction, computer gaming and surveillance. In this paper we propose a fast, simple, yet powerful non-parametric Moving Pose (MP) *framework for low-latency human action and activity recognition. Central to our methodology is a moving pose descriptor that considers both pose information as well as differential quantities (speed and acceleration) of the human body joints within a short time window around the current frame. The proposed descriptor is used in conjunction with a modified kNN classifier that considers both the temporal location of a particular frame within the action sequence as well as the discrimination power of its moving pose descriptor compared to other frames in the training set. The resulting method is non-parametric and enables low-latency recognition, one-shot learning, and action detection in difficult unsegmented sequences. Moreover, the framework is real-time, scalable, and outperforms more sophisticated approaches on challenging benchmarks like MSR-Action3D or MSR-DailyActivities3D.*

1. Introduction

Automatic action and activity recognition are important computer vision research problems of broad practical applicability. The new technologies being developed in computer games using RGB-D cameras, or for environmental awareness, require flexible methodologies that are both accurate and have low observational latency, such that actions can be reliably recognized long before the entire observation stream is presented. The task of early action detection was addressed only recently but gains momentum [7, 9] due to the fast development of new technologies in human-robot interaction or surveillance. The accurate real-time tracking of 3D skeletons [21], made possible with the introduction of low-cost RGB-D cameras, led to the development of efficient methods for classification of dance moves [17] and other arbitrary actions [7, 24]. Most published methods, however, require an entire action sequence in order to perform classification. Only very few techniques offer low-latency responses that would allow the rapid identification of an action long before it ends [7, 9], and not all methods are are able to cope with unsegmented test sequences.

In this paper we make several contributions which we collectively refer to as the *Moving Pose* (MP) framework. First, we propose the Moving Pose descriptor–a novel frame-based dynamic representation that captures not only the 3D body pose but also differential properties like the speed and acceleration of the human body joints within a short time window around the current frame. We argue that due to physical constraints like inertia, or latency in muscle actuation, the body movements associated with an action can often be well approximated by a quadratic function, expressed in terms of the first and second derivatives of the body pose with respect to time. Our second contribution is a modified non-parametric kNN classifier that considers not only the global temporal location of a particular frame within the action sequence, but also the different discriminative power associated with moving pose descriptors computed in each frame. This makes our method capable to perform not only state of the art low-latency action recognition, but also accurate action detection in natural, unsegmented sequences captured under weakly controlled conditions. We also show that our method is well suited for the one-shot learning of actions, unlike any of the recent, early action recognition approaches.

Related Work: Traditional research on general action recognition focuses mainly on recognition accuracy using hidden Markov models, and more recently conditional random fields [14, 22], and less on reducing observational latency [1, 10, 20]. In applications where a quick response is needed, such as environment aware systems or robot-computer interaction, the system has to respond quickly. Thus, the action has to be recognized as soon as enough evidence becomes available, ideally long before the whole sequence is observed.

1550-5499/13 $31.00 © 2013 IEEE
DOI 10.1109/ICCV.2013.342

Many authors have observed that primitive actions can be well represented by a few key poses [5, 7, 19] for action classification. Variations on using key poses include matching shape information to prototypes [3] and using discriminative frames to weight features in a bag-of-words classifier [27]. Distinctive key poses from contours are learned by [4], while [11] matches silhouettes between the test frames and learned key frames. Other methods rely on manually selecting key frames [3] or require accurate silhouettes [4,11]. Recent models [7,23] use multiple discriminative key frames, obtained separately or during learning.

An important aspect in the automatic processing of actions is segmentation, since most real-world videos are not partitioned into separate actions. There are a few approaches that perform automatic temporal segmentation [2, 26] but they require processing an entire sequence for classification. In this paper we first assume that actions are already segmented, then show that our approach can be extended to automatically process unsegmented sequences. Recent work on early event detection [9] modifies a structured output max-margin framework, within a bag-of-words representation, by augmenting the training set with partial events and enforcing a temporal monotonicity constraint on outputs–thus partial sequences do not have a higher detection rate than encompassing sequence. Other techniques [16] focus on reducing the computational latency of decoding hidden state sequences, rather than on the observational latency linked with classifying early partial events.

2. Representing Human Actions and Activities

Human actions are often considered to be primitive activities consisting of one or a few atomic body movements or poses such as: walking, sitting, standing, kicking or jumping. More complex activities often involve interactions with objects and other people, and can occur over longer time scales. In this paper we focus on recognizing actions or activities that are well described by the 3D body pose and movement only. To simplify formulation we will generally refer to both types as *actions*, and qualify only when necessary to differentiate between the two.

We represent an action as a sequence of frame descriptors ordered in time. We assume that the 3D joint positions of the human body are available in each frame. Since our goal is to design a fast, low-latency classification method, we will rely on only a few frames for efficient recognition. Consequently, we need local frame descriptors that capture as much information as possible about the action in the neighborhood of a given frame, in terms of both pose and the kinematics of body joints. We observe that:

1. Actions usually involve body movement, not just the body pose. At a given moment in time, a certain pose together with specific movements of the 3D body joints could be highly predictive of the intentions of the human subject and the action performed.

2. Despite the natural skeleton size and proportions variation in the human population, people tend to perform the same action in the same qualitative way, by moving similarly. Thus, for a given action, differential quantities like the relative speed and acceleration of these joints could be more stable across subjects than their actual 3D locations.

It is clear that a good descriptor should capture both the static pose as well as the joint kinematics at a given moment in time. The velocity and acceleration of the different 3D joints capture relevant information about different actions. Velocity describes the direction and speed of the joints. It is important to differentiate between actions spanning similar poses but different directions of movement, such as jumping vs. falling, or standing up vs. sitting down. The acceleration of the 3D joints captures the change in velocity over time. Changes in directions as well as in speed produce nonzero acceleration, which is useful to differentiate between actions involving circular motions such as drawing a circle or waving, versus drawing a line or standing up.

If we view the pose as a continuous and differentiable function of the body joint positions over time, then its second-order Taylor approximation in a window around the current time step t_0 would be defined by expanding around the current pose $\mathbf{P}(t_0)$ based on the first and second order derivatives, $\delta\mathbf{P}(t_0)$ and $\delta^2\mathbf{P}(t_0)$:

$$\mathbf{P}(t) \approx \mathbf{P}(t_0)+\delta\mathbf{P}(t_0)(t-t_0)+1/2\delta^2\mathbf{P}(t_0)(t-t_0)^2. \quad (1)$$

Effectively, the instantaneous pose and its first and second order derivatives at a given time t_0 contain information about the pose function over a time segment around the current t_0. The time window over which the approximation is valid could be much longer than the local window used to approximate derivatives. Therefore, this type of local information could be seen as an approximation of an action snippet [18] and can be used to reliably classify actions. In experiments we actually show that introducing kinematic information, encoded as proposed, induces a significant improvement over the use of poses-only, even when all poses needed to compute derivatives are used within local frame descriptors and fed to classifiers (Table 4).

3. The Moving Pose Descriptor

The skeleton representation of a pose in each frame of a video captures the 3D positions of the skeleton joints $\mathbf{p_i} = (p_x, p_y, p_z)$, where $i \in \{1, \ldots, N\}$, with N the total number of human body joints. For each frame we compute the Moving Pose Descriptor (MP), as a concatenation of the normalized 3D pose $\mathbf{P} = [\mathbf{p_1}, \mathbf{p_2}, \ldots, \mathbf{p_n}]$ and its first and

second order derivatives $\delta\mathbf{P}(t_0)$ and $\delta^2\mathbf{P}(t_0)$. The derivatives are estimated numerically by using a temporal window of 5 frames centered at the current one processed: $\delta\mathbf{P}(t_0) \approx \mathbf{P}(t_1) - \mathbf{P}(t_{-1})$ and $\delta^2\mathbf{P}(t_0) \approx \mathbf{P}(t_2) + \mathbf{P}(t_{-2}) - 2\mathbf{P}(t_0)$. For better numerical approximation we first smooth each coordinate of the normalized pose vector, along the time dimension, with a 5 by 1 Gaussian filter ($\sigma = 1$). Note that Gaussian smoothing produces a lag of two frames which does not significantly impact the overall latency in practice.

In order to suppress noise in the estimated input pose and to compensate for skeleton variations across different subjects, we normalize the poses as described in the following paragraph. The derivative vectors are also rescaled so that they have unit norm–this normalization also removes irrelevant variation in absolute speed and acceleration across different input sequences, while preserving the relative distributions between the different joints. The final frame descriptor $\mathbf{X}_t$ for frame at time t is obtained by concatenating the pose and its derivatives over time: $\mathbf{X}_t = [\mathbf{P}_t, \alpha\delta\mathbf{P}_t, \beta\delta^2\mathbf{P}_t]$. The parameters α and β weight the relative importance of the two derivatives and are optimized over the training set.

Algorithm 1 Skeleton normalization for frame at time t

Let $\mathbf{p}^{(1)}_{start}$ be the position of the root joint.
$\mathbf{p'}^{(1)}_{start} \leftarrow \mathbf{p}^{(1)}_{start}$
for all $(\mathbf{p}^{(i)}_{start}, \mathbf{p}^{(i)}_{end})$ (in breadth first search order) **do**
  $\mathbf{d}_i \leftarrow (\mathbf{p}^{(i)}_{start}, \mathbf{p}^{(i)}_{end})$
  $\mathbf{d}'_i = r_i \frac{\mathbf{d}_i}{||\mathbf{d}_i||}$
  $\mathbf{p'}^{(i)}_{end} \leftarrow \mathbf{p'}^{(i)}_{start} + \mathbf{d}'_i$
end for
return $\mathbf{P'} = [\mathbf{p'_1}, \mathbf{p'_2}, \ldots, \mathbf{p'_n}]$

Pose Normalization (Algorithm 1): Human subjects have variations in body and limb sizes which are not relevant for the action performed. In order to compensate for anthropometric differences, we impose the same limbs (skeleton segments) lengths for poses obtained from all individuals in the dataset. We learn average skeleton segment lengths from training data (a segment is defined by any two linked joints). Let $\mathbf{R} = [r_1, r_2, \ldots, r_m]$ be the expected lengths of skeleton limbs (segments), learned from training data. We then adjust $\mathbf{R}$ to unit norm. Then for a given training or testing pose, we start from a root node (the hip joint), move forward the branches of the kinematic tree associated with the body joints, and successively modify the joint locations accordingly, such that a length of i-th limb segment, say, becomes equal to r_i, while its direction vector is preserved. By using this procedure, the joint angles are not modified, but the same limbs (*e.g.* the arms, or the legs) will have the same length across subjects. In order to make the 3D joint locations invariant to camera parameters, we subtract from each joint the position of the hip center $\mathbf{p}_{hip}$: $\mathbf{P} = [\mathbf{p}_1 - \mathbf{p}_{hip}, \ldots, \mathbf{p}_N - \mathbf{p}_{hip}]$. Carefully normalizing, as proposed, is important, and improves the performance of our methods by at least 5%.

4. Action Classification

The proposed MP descriptor encodes pose and kinematic information to describe action segments. In order to emphasize its discriminative power and for training flexibility (including one-shot learning) we use a non-parametric action classification scheme based on k-nearest-neighbors (kNN). Our basic low-latency classification method for a test sequence is as follows: at time t, after observing t test frames, let each of their kNN descriptors from the training pool vote for its class, for a total of kt votes. For decision, we apply a simple rejection scheme. If the accumulated vote of the most supported class c_j is high enough compared to the other classes, and enough frames have been observed, we report the class with the largest number of votes, *i.e.* output c_j if $\max_j s(c_j, t) \geq \theta$, with s an additive voting model and θ a confidence threshold.

4.1. Learning Discriminative Frames

One of the main difficulties in analyzing action sequences is that not all frames are representative for an action. When performing certain tasks a person often has to go through poses that are common to many other actions or activities. This means that training data is, effectively, only weakly supervised: not all frames in a given sequence, assumed in bulk to belong to a specific action, are relevant for that action. Such irrelevant or neutral configurations, such as standing poses, could negatively impact the standard kNN classification. Our simple solution to this problem is to classify, at training time, each frame in the training set using a kNN classifier (consider each training descriptor as an unknown one and use the others to classify it). The probability that a timestep descriptor 'strongly belongs' to its class can be estimated by the confidence obtained using the votes from its k nearest neighbors. If c is the correct class of the sample $\mathbf{X}$, the confidence function $v(\mathbf{X})$ is defined as the estimated posterior of the class c, given $\mathbf{X}$, and the non-parametric k-nearest neighbors class density estimator.

$$v(\mathbf{X}) = P(c|\mathbf{X}) = \frac{P(\mathbf{X}, c)}{P(\mathbf{X})} \approx k_c/k. \qquad (2)$$

Here k is the number of nearest neighbors and k_c the number of those that belong to class c. This confidence is not a perfect measure as among nearest neighbors, some may not be relevant, but if k is large enough the procedure works well in practice. The confidence value is then used for classification as described in Algorithm 2. In fig.1 we show

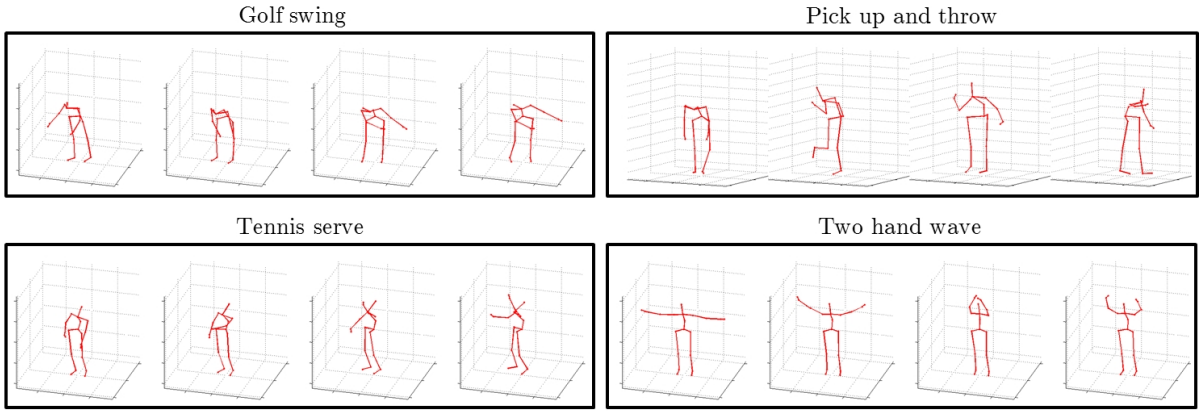

Figure 1. Examples of high confidence frames automatically identified from training sequences using Algorithm 2. The poses shown were among the top ten rated the most confident within their sequence. Note how they indeed look iconic for those specific actions. The poses with low confidence score were usually the common, shared ones, such as standing poses from the start and the end of each sequence.

some of the most discriminative frames, automatically discovered by the algorithm, for a few action classes. On visual inspection, the automatically identified poses appear to be representative, iconic, for their corresponding actions.

4.2. Global Temporal Information for Classification

A limitation of classical kNN classification applied to action recognition is its lack of account of global temporal ordering. The descriptor $\mathbf{X}$ captures only local temporal variations but does not include information that would allow one to identify when during the action sequence a particular short action segment takes place. The global position of a pose within the entire sequence is often relevant. For example, when a tennis player hits the ball, he usually swings the racket first backwards, then forward. The order of these two movements matters. Our approach to incorporate global temporal information within a kNN framework is to gate the search for nearest neighbors only to samples that are located at a similar position in the training sequence, with respect to the first frame.

By combining discriminative local moving pose descriptors like MP with a temporal aware classification scheme, we can now account for two important aspects in action classification: the discriminative power of key poses as well as their local dynamics, and the global temporal course of an action. This turns out to be important for more complex activities that consist of many atomic action units.

5. Experiments

We test our method on both simple and complex actions, present in both segmented as well as unsegmented sequences. We use the MSR Action3D dataset, the MSR-DailyActivity3D dataset, the data of Ellis et al. [7], as well as our own unsegmented action dataset[1]. We compare to current methods and obtain state-of-the art results on all datasets. We also demonstrate the effectiveness of the proposed moving pose framework and the usefulness of our confidence and global temporal constraints, over different baselines. Our results clearly show that differential quantities like local motion and acceleration are powerful cues for action recognition and in conjunction with static 3D poses they can capture the specific dynamics inherently linked to different types of actions, far beyond the static poses themselves. Our method is able to accurately recognize actions with low observational latency at real time rates of $200-500$ FPS (depending on the dataset) on a desktop computer.

Algorithm 2 Low-latency Action Recognition

For all candidate actions c set the score $s(c) = 0$.
for all time steps $t \leq T$ **do**
  Compute the descriptor $\mathbf{X}_t$.
  Find the kNN of $\mathbf{X}_t$ from a similar temporal location in the sequence w.r.t first frame.
  for all $\mathbf{X}_{train} \in kNN(\mathbf{X}_t)$ **do**
    $s(c) \leftarrow s(c) + v(\mathbf{X}_{train})$, where c is the class of $\mathbf{X}_{train}$ and $v(\mathbf{X}_{train})$ is the confidence of $\mathbf{X}_{train}$.
  end for
  If $t > N_{min}$ and $\frac{s(c)}{\sum_b s(b)} > \theta$
  then return $c^* = \text{argmax}(s(c))$.
end for
return Final class $c^* = \text{argmax}(s(c))$

5.1. Action Recognition

The MSR Action3D dataset consists of temporally segmented action sequences captured by a RGB-D camera. There are 20 actions in the dataset, performed $2-3$ times

[1]Publicly available at `www.imar.ro/clvp/actionMP`.

by 10 subjects. In total there are 567 sequences. The 3D skeleton, represented as a set of 3D body joint positions, is available for each frame, being tracked with the method of [21]. About 10 skeleton sequences were not used in [24] because of missing data or highly erroneous joint positions. We follow the same procedure for fair comparisons. Our parameters α and β (§3) were learned over the training set and kept constant for all testing experiments (*i.e.* $\alpha = 0.75$, $\beta = 0.6$). We use the cross-subject test setting as in [10], where the sequences for half of the subjects are used for training (*i.e.* subjects $(1, 2, 3, 4, 5)$), and the remaining sequences of the other half of the subjects for testing. Our

Table 1. Recognition comparison on the MSR Action3D dataset.

Method	Accuracy(%)
Recurrent Neural Network [13]	42.5
Dynamic Temporal Warping [15]	54
Hidden Markov Model [12]	63
Latent-Dynamic CRF [14]	64.8
Canonical Poses [7]	65.7
Action Graph on Bag of 3D Points [10]	74.7
Latent-Dynamic CRF [14] + MP	74.9
EigenJoints [25]	81.4
Actionlet Ensemble [24]	88.2
MP (Ours)	**91.7**

system (see Table 1) improves over the current state-of-the-art by 3.5%. Figure 2 shows that we obtain perfect accuracy on 17 out of 20 classes, compared to just $11/20$ in [24]. The only classes causing confusion are, in our case, the ones involving interactions with other objects (pick up and throw, hand-catch and hammer), which (as also observed by [24]) may not be easily captured by the skeleton information alone. However, even using such limited information and a simple classifier, we clearly outperform the current state of the art based on 3D skeletons, LOP appearance and depth features, global Fourier features, learned actionlets, and a powerful Multiclass-MKL learning framework. Naturally some of these additional features can be incorporated in our framework as well.

In addition to results readily available in the literature, we also perform comparisons with a conditional random field, LDCRF [14]. We used the code available online and validated the model parameters under the same experimental setting. When just 3D pose was used as observation, te recognition rate was of 64.8%. By changing the unary term with our MP descriptor, the accuracy increased significantly to 74.9%. This supports the merits of MP as a descriptor and shows that our non-parametric classifier complements it well. In the context of action detection (discussed later in 5.6), our improvement over LDCRF in terms of per-frame classification rate, is 57.3% over 54%. Note that LDCRF is

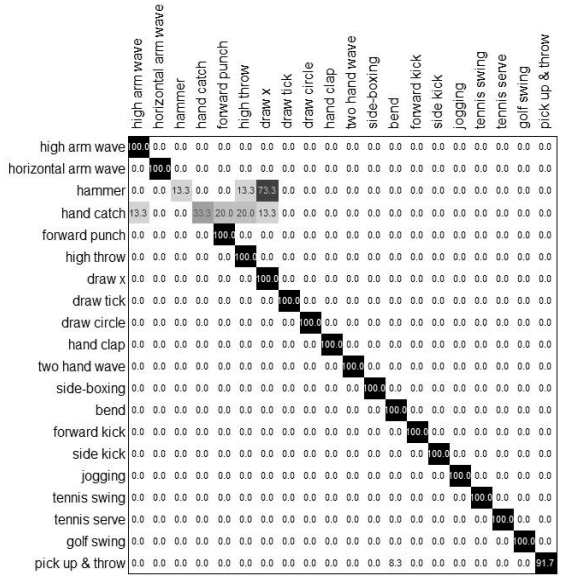

Figure 2. Confusion matrix of our method (MP) on the MSR-Actions3D dataset: $17/20$ actions are perfectly classified.

not immediately applicable for action detection in realistic unsegmented scenarios, where it would require no-action states. One-shot learning would not be entirely straightforward, either. In contrast, it is easy to adapt our approach to such cases as shown later. We also conducted experiments on the dataset introduced in Ellis et al. [7]. This dataset contains $1,280$ temporally segmented action sequences gathered from 16 different human subjects. Each subject performs each of the 16 actions for 5 times. In this dataset, the human poses are encoded in terms of only 15 3D joint positions. As the location of the hip center, necessary for our pose normalization, is not given, we approximate it by taking the mean position of the left and right hip joints. For this dataset we used a 4-fold cross-validation approach as in [7], and averaged the results. It is important to mention that on this dataset we did not retrain our model parameters, but use the ones learned on the training set of MSR Action3D dataset. Nonetheless, our method improves over [7] by 2.5% (see Table 2).

Table 2. Recognition performance on the action dataset of [7].

Method	Accuracy(%)
Canonical Poses [7]	95.94
MP (Ours)	**98.50**

5.2. Activity Recognition

The MSRDailyActivity3D dataset consists of temporally segmented sequences of humans performing daily activi-

ties. The dataset was captured with the Kinect RGB-D camera and covers 16 activities, performed by 10 subjects. Each of the subjects performs an activity twice: once seated on a sofa and once standing. There are 320 temporally segmented activity sequences in total. This dataset is challenging, as most activities involve human-object interactions. Moreover, the joint positions are very noisy when the person comes too close to the sofa. The cross-subject test setting is also used on this dataset. The subjects used for training are $(1, 3, 5, 7, 9)$, whereas the rest are used for testing, as in [24]. In Table 3 we show that our method obtains state-of-the-art results, even when only 3D pose information is used. The learned values of our descriptor parameters (§3) were $\alpha = 0.6$ and $\beta = 0.4$.

Table 3. Recognition comparisons on MSRDailyActivity3D.

Method	Accuracy(%)
Dynamic Temporal Warping [15]	54.0
Actionlet Ensemble (3D pose only) [24]	68.0
MP (Ours)	**73.8**

5.3. Benefit of Using $\delta\mathbf{P}$ and $\delta^2\mathbf{P}$

The addition of kinematic features $\delta\mathbf{P}$ and $\delta^2\mathbf{P}$ dramatically increases the recognition accuracy over the 3D poses alone, as can be seen in Table 4. In order to estimate the derivatives numerically we use information from other frames around the target one t_0. Thus, when only the first derivative $\delta\mathbf{P}$ is used in combination with the pose $\mathbf{P}$, we need information from three frames at times (t_{-1}, t_0, t_{+1}) around the current time t_0. When the second derivative $\delta^2\mathbf{P}$ is included, we need the five frames at times $(t_{-2}, t_{-1}, t_0, t_{+1}, t_{+2})$. The benefit of using the derivatives becomes evident when we compare with descriptors that consider only the static 3D poses from these frames. In Table 4 we use the following notation: $3\mathbf{P} = [\mathbf{P}_{-1}, \mathbf{P}_0, \mathbf{P}_{+1}]$ and $5\mathbf{P} = [\mathbf{P}_{-2}, \mathbf{P}_{-1}, \mathbf{P}_0, \mathbf{P}_{+1}, \mathbf{P}_{+2}]$. It is interesting to notice that even though poses from a time window implicitly contain movement information, the explicit estimation of speed and acceleration makes a significant difference in the overall performance. This empirical result confirms the intuition that the actual body movement is highly predictive of the action class. For example on MSR-Action 3D dataset, adding the speed of the joints ($\delta\mathbf{P}$) improves the accuracy by a substantial 25% margin over 3D pose ($\mathbf{P}$) alone, whereas the acceleration ($\delta^2\mathbf{P}$) adds another significant 6%. The acceleration component is very important for actions with curved movements, such as drawing a circle. Such movements cannot be captured by pose and speed alone. A similar trend was observed on the actions dataset from [7], where the accuracy dropped from 98.5% to 83.9% using poses alone.

Table 4. Recognition accuracy for different feature types and extensions to the kNN method on Actions (MSR-Actions3D dataset) and Activities (DailyActivities3D dataset).

Feature	Actions	Activities
joint angles	26.5	24.4
$\mathbf{P}$	60.5	63.7
$\mathbf{3P}$	71.1	62.5
$\mathbf{5P}$	73.5	61.8
$\delta\mathbf{P}$	77.0	51.3
$\delta^2\mathbf{P}$	74.6	46.3
$\mathbf{P}\ \delta\mathbf{P}$	84.5	67.5
$\mathbf{P}\ \delta\mathbf{P}\ \delta^2\mathbf{P}$	89.7	68.1
$\mathbf{P}\ \delta\mathbf{P}\ \delta^2\mathbf{P}$ + conf.	91.7	69.3
$\mathbf{P}\ \delta\mathbf{P}\ \delta^2\mathbf{P}$ + conf. + temp.	**91.7**	**73.8**

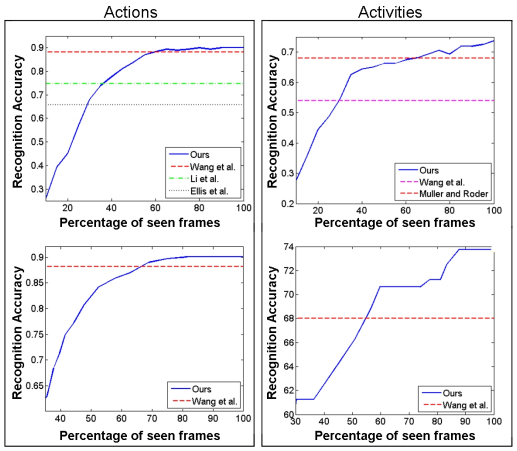

Figure 3. Improvement in accuracy as latency increases for MP. We also plotted the accuracy reported by some of our competitors after processing the entire sequence. *Left column:* recognition accuracy on MSR-Actions3D. *Right column:* accuracy on DailyActivities3D. *Top row:* accuracy reported when stopping after observing a fixed percentage of the frames. *Bottom row:* the realistic scenario, where the length of the input sequence is not known in advance. We vary the threshold θ (Algorithm 2) and plot the accuracy vs. average latency as θ increases from 0 to 1.

5.4. Latency Analysis

Our method is well suited for action recognition with low observational latency, since at any point in time we could output the action that accumulated the largest number of votes, at a given rejection threshold. The question is when can we confidently classify the action. On one hand, we have to observe a minimum number of frames and on the other hand we need a measure of classifier response. The decision function we use (Algorithm 2) is the value of the accumulated votes for the winning class divided by the total number of votes for all actions, counted from the beginning of a sequence. The intuition is that the higher this value relative to the votes from the other classes, the more confident

we are that our winning action is indeed correct. In fig.3 we show that by varying a threshold θ on the function output we can confidently classify actions long before the end of the sequence. For example we outperform the method of [7], as measured over entire sequences in MSR-Actions3D and that of [15] in the DailyActivities3D dataset after observing only $\approx 30\%$ of the frames, on average. After $\approx 60-70\%$ of the frames are observed, the accuracy of MP supersedes the state-of-the-art.

5.5. One-shot Learning

Another advantage of our non-parametric approach over current methods is the ability to recognize actions using a very limited amount of training data. Since we use a non-parametric classification approach we can apply our method in the case of one-shot learning, when only a single training sequence for each action is available. We tested the MP approach in the one-shot learning scenario on both MSR-Actions3D and DailyActivities: in the previous experiments (Tables 1 and 3) there are about 13 training sequences per action and 10 per activity. For the results presented in this section (Table 5) we performed 100 different experiments where a single training sequence was chosen randomly for each action with decision performed based on such 'early' learners. As expected, recognition accuracy dropped, but the system was still able to perform relatively well. For example, on MSR-Actions3D dataset we obtain a similar accuracy (using a single training sequence per action) to that of [7] (using on average 13.3 training sequences per action). Note that none of the methods we compared against

Table 5. Average recognition accuracy under one-shot learning.

Feature	MSR-Actions3D	DailyActivities3D
Poses only	$44.1 \pm 4.6\%$	$41.6 \pm 3.4\%$
Motion Poses	$65.2 \pm 5.2\%$	$42.6 \pm 4.0\%$

can perform one-shot learning. [24] uses a Multiclass-MKL with actionlet mining. [7] uses multiple instance learning of canonical action poses based on many training sequences. Neural networks [13], motion templates [15], HMMs [12] and action graphs [10] need sufficient data to learn models.

5.6. Action Detection in Unsegmented Sequences

We test the efficiency of our method on a significantly more difficult scenario–that of action recognition and detection in unsegmented sequences. Our method can be easily modified to handle low-latency action detection in unsegmented sequences. The action classification responses per frame are obtained as before. Since in the unsegmented case, we do not know the temporal length of the action performed, we take a sliding window approach and learn a window of size W by cross-validation. Given votes accumulated over this window, we estimate the probability of observing action class c at a given moment t as:

$$R(c,t) = \frac{\sum_{f=t-W}^{t} s(c,f)}{\sum_{b}\sum_{f=t-W}^{t} s(b,f)}. \quad (3)$$

Here $s(c,f)$ is the frame-wise classification response of action c computed exactly as in the unsegmented case. After computing responses for each action, the winning action produces the final frame label $c^*(t) = \text{argmax}(R(c,t))$. The frame labeling procedure effectively produces a segmentation of the whole sequence into contiguous (connected component) segments having the same label. After discarding very short segments (5 frames), we also record the maximum response O_S over each segment S. This maximum response is the classification output associated with the entire segment, which will be used in detection evaluation, together with the segment endpoints.

Since most published RGB-D datasets deal with classification where actions are already pre-segmented, we created two different experimental setups: first, we took the MSR-Actions3D and DailyActivities datasets and randomly concatenated all test sequences into a single long test sequence, one per dataset. We performed action detection, following precision-recall experimental protocols widely used in object detection from images, such as Pascal VOC Challenge [8] (our overlapping threshold is $\mathbf{0.2}$ as in [6]). For each dataset we performed 100 random concatenations and present the statistics of our results in Table 6. Note that our method performs well even in the unsegmented case, when all test sequences are concatenated into a single input sequence. The very small standard deviation is also an indication that performance is not impacted by the ordering of actions.

Table 6. Unsegmented action detection performance

Scoring method	MSR-Actions3D	Activities
Detection (AP)	0.890 ± 0.002	0.603 ± 0.011

For a more realistic testing setup, we also captured our own dataset, with the Microsoft Kinect camera and using software for 3D human pose estimation. The data contains 30 videos performed by 10 different actors with both 3D joint and depth information gathered. Each subject was instructed to freely perform 3 sequences of multiple actions, 7 different actions per set, chosen randomly from the pool of actions present in MSR-Actions3D. The subjects were asked to perform them in any order and number of times, with any intermediate actions in between the fixed ones. The dataset was designed to imitate as closely as possible typical user-computer interactions. For training, we used only sequences from MSR-Actions3D. We present results

(see Table 7) for both unsegmented action detection average precision as well as action classification for comparison (when the ground truth segmentations were available). The experiments indicate that our system is able to efficiently detect actions in difficult unsegmented sequences, in the presence of different subjects (than in training) and in uncontrolled environments.

Table 7. Experiments on our own test sequences using for training, the training set of MSR-Actions3D (Train Set 1) or the entire MSR-Actions3D dataset (Train Set 2).

Scoring method	Train Set 1	Train Set 2
Detection AP	0.765	0.776
Classification rate	0.853	0.870

6. Conclusions

We have presented a novel *moving pose descriptor framework* for action recognition which uses pose and kinematic information encoded as differential 3D quantities. We have also proposed a novel, modified non-parametric kNN classifier, based on discriminative key frames with augmented temporal information. We demonstrated the power of our methodology by obtaining state of the art results on recent, challenging benchmarks for action and activity recognition. Moreover, we show that our techniques are well suited for action detection in difficult unsegmented sequences and for low latency recognition, unlike most previous approaches. Our approach further enables one-shot learning, all within a scalable real-time framework. In future work, we plan to explore more powerful classifiers as well as the design of human-object interaction descriptors.

Acknowledgments: Support in part by CNCS-UEFISCDI under CT-ERC-2012-1, PCE-2011-3-0438.

References

[1] S. Ali and M. Shah. Human action recognition in videos using kinematic features and multiple instance learning. *PAMI*, 32, 2010.

[2] L. Cao, Z. Liu, and T. Huang. Cross-dataset action detection. In *CVPR*, 2010.

[3] S. Carlsson and J. Sullivan. Action recognition by shape matching to key frames. In *WMECV*, 2001.

[4] S. Cheema, A. Eweiwi, C. Thurau, and C. Bauckhage. Action recognition by learning discriminative key poses. In *ICCV Workshop*, 2011.

[5] N. Cuntoor and R. Chellappa. Key frame-based activity representation using antieigenvalues. In *ACCV*, 2006.

[6] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce. Automatic annotation of human actions in video. In *ICCV*, 2009.

[7] C. Ellis, S. Masood, M. Tappen, J. L. Jr., and R. Sukthankar. Exploring the trade-off between accuracy and observational latency in action recognition. *IJCV*, August 2012.

[8] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[9] M. Hoai and F. de la Torre. Max-margin early event detectors. In *CVPR*, 2012.

[10] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *WCBA-CVPR*, 2010.

[11] F. Lv and F. Nevaita. Single view human action recognition using key pose matching and viterbi path searching. In *CVPR*, 2007.

[12] F. Lv and R. Nevatia. Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. In *ECCV*, 2006.

[13] J. Martens and I. Sutskever. Learning recurrent neural networks with hessian-free optimization. In *ICML*, 2011.

[14] L.-P. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *CVPR*. IEEE Computer Society, 2007.

[15] M. Müller and T. Röder. Motion templates for automatic classification and retrieval of motion capture data. In *SCA*, 2006.

[16] M. Narasimhan, P. Viola, and M. Shilman. Online decoding of markov models under latency constraints. In *ICML*, 2006.

[17] M. Raptis, D. Kirovski, and H. Hoppes. Real-time classification of dance gestures from skeleton animation. In *Symp. on Comp. Anim.*, 2011.

[18] K. Schindler and L. van Gool. Action snippets: How many frames does human action recognition require ? In *CVPR*, 2012.

[19] L. Shao and L. Ji. Motion histogram analysis based key frame extraction for human action/activity representation. In *CRV*, 2009.

[20] Y. Shen and H. Foroosh. View-invariant action recognition from point triplets. *PAMI*, 31, 2009.

[21] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *CVPR*, 2011.

[22] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Conditional models for contextual human motion recognition. In *CVIU*, 2006.

[23] A. Vahdat, B. Gao, M. Ranjbar, and G. Mori. A discriminative key pose sequence model for recognizing human interactions. In *IWVS*, 2008.

[24] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, 2012.

[25] X. Yang and Y. Tian. Eigenjoints-based action recognition using naïve-bayes-nearest-neighbor. In *CVPR Workshops*, pages 14–19, 2012.

[26] J. Yuan, Z. Liu, and Y. Wu. Discriminative subvolume search for efficient action detection. In *CVPR*, 2009.

[27] Z. Zhao and A. Elgammal. Information theoretic key frame selection for action recognition. In *BMVC*, 2008.