Recognizing Actions in Wearable-Camera Videos by Training Classifiers on Fixed-Camera Videos

Yang Mi
University of South Carolina
miy@email.sc.edu

Kang Zheng
University of South Carolina
zheng37@email.sc.edu

Song Wang*
Tianjin University
University of South Carolina
songwang@cec.sc.edu

ABSTRACT

Recognizing human actions in wearable camera videos, such as videos taken by GoPro or Google Glass, can benefit many multimedia applications. By mixing the complex and non-stop motion of the camera, motion features extracted from videos of the same action may show very large variation and inconsistency. It is very difficult to collect sufficient videos to cover all such variations and use them to train action classifiers with good generalization ability. In this paper, we develop a new approach to train action classifiers on a relatively smaller set of fixed-camera videos with different views, and then apply them to recognize actions in wearable-camera videos. In this approach, we temporally divide the input video into many shorter video segments and transform the motion features to **stable** ones in each video segment, in terms of a fixed view defined by an anchor frame in the segment. Finally, we use sparse coding to estimate the action likelihood in each segment, followed by combining the likelihoods from all the video segments for action recognition. We conduct experiments by training on a set of fixed-camera videos and testing on a set of wearable-camera videos, with very promising results.

KEYWORDS

Action recognition; Wearable camera; Stable motion features; Sparse coding

ACM Reference Format:
Yang Mi, Kang Zheng, and Song Wang. 2018. Recognizing Actions in Wearable-Camera Videos by Training Classifiers on Fixed-Camera Videos. In *ICMR '18: 2018 International Conference on Multimedia Retrieval, June 11–14, 2018, Yokohama, Japan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3206025.3206041

1 INTRODUCTION

In this paper, we study the problem of recognizing the human actions present in a video that is taken by wearable camera, such as GoPro and Google Glass. Compared to static cameras, wearable camera can cover wider areas and better capture the actions of interest, with the subjective motion of the camera wearer. This research can benefit many important applications. In law enforcement, from the videos taken by police-officer-worn cameras, we can identify abnormal people and behavior to enhance security in public areas. In kindergarten, from the videos taken by teacher-worn cameras, we can identify kids with insufficient or atypical activities, which may imply possible social difficulties.

So far, the most effective approach for video-based action recognition is to first extract spatial-temporal motion features, and then use these motion features to train classifiers for recognizing different human actions. The performance of this approach highly relies on the consistency of the extracted motion features, which may not be well preserved when the input video is taken by a wearable camera. On one hand, the extracted motion features may not accurately reflect the desired human action by mixing undesired camera motion. On the other hand, continuous camera motion leads to varying camera view in the duration of the action, as illustrated in the bottom row of Fig. 1, and it is well known that many motion features are very sensitive to the view change. Clearly, it is very difficult to collect sufficient videos to cover all such variations, i.e., cover all possible camera motions and view changes in the duration of an action, which, however, is necessary for training classifiers with good generalization ability.

Figure 1: Sample videos from a fixed camera (top) and a wearable camera (bottom).

Identifying and compensating camera motions to construct stable motion features can effectively address this problem. If we can transform all the frames in a video and the motion features extracted from these frames to a fixed camera view, then for each action we only need to collect a relatively small set of fixed-camera videos from different view angles for training, since the space of possible fixed-camera view is of much lower dimension than the space of possible wearable-camera motion in the duration of an action. However, given the accumulated errors in estimating and compensating the camera motion between two temporally distant frames, it is difficult to transform all the frames and their motion features in a video of an action to one single fixed camera view.

*corresponding author

ICMR '18, June 11–14, 2018, Yokohama, Japan

ACM ISBN 978-1-4503-5046-4/18/06…$15.00
https://doi.org/10.1145/3206025.3206041

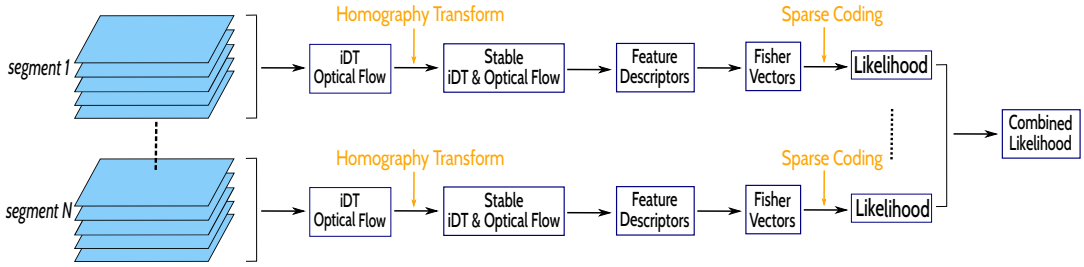

Figure 2: Workflow of the proposed method for action recognition in wearable camera videos.

In this paper, we propose a new approach to address this problem by dividing the video of an action into a sequence of shorter video segments. In each video segment, we extract *stable* motion features corresponding to a fixed camera view, defined by an anchor frame in the segment. This is achieved by computing and applying homography transforms between the anchor frame and other frames assigned to this video segment. By controlling the length of video segment, the desired homography transforms can be estimated by composing the adjacent-frame homography transforms without high accumulation errors. However, stable features extracted for different video segments correspond to different views because each video segment has its own anchor frame. Combining the stable features from all the video segments for action classification still leads to feature inconsistency.

To address this issue, we further propose to use a sparse-coding model to estimate the action likelihood in each video segment independently, followed by combining the likelihoods from all the video segments for final action recognition. In estimating the action likelihood of each video segment, its stable motion features are consistent to those extracted from the corresponding segment of the training videos, since the training videos are taken by fixed cameras in many different views. The combination of the video subdivision and motion-feature stabilization enables the action classifiers trained on fixed-camera videos to be applicable to recognize actions in wearable-camera videos. In the experiments, we apply the proposed method to recognize eight human actions: *jump*, *handwave*, *pickup*, *punch*, *standup*, *sitdown*, *kick*, *throw* from wearable-camera videos, with very promising results.

The remainder of this paper is organized as follows. Section 2 briefly overviews the related work. Section 3 introduces the proposed method in detail. Section 4 reports the experimental results, followed by a brief conclusion in Section 5.

2 RELATED WORK

Most video-based human action recognition approaches extract **spatial-temporal features**, such as hand-crafted features based on trajectories extracted from videos [9, 30–33, 38]. For example, the improved dense trajectory (iDT) [33] based descriptors of HOG (histograms of oriented gradients) [5], HOF (histograms of optical flow) [3], and MBH (motion boundary histogram) [31] have been shown to be effective in action recognition on many challenging datasets, e.g. HMDB51 [15] and UCF101 [28]. Recently, convolutional neural networks (CNNs) have been used to extract deeply-learned features from videos [6, 10, 13, 16, 27, 29, 35]. Many action-recognition approaches [6, 16, 29, 35] combine deeply-learned features and hand-crafted features to achieve better performance. However, most of these methods do not explicitly consider the feature inconsistency caused by camera motion.

Most recent works [1, 18, 19, 22] on **action recognition based on wearable camera videos** are focused on ego-centric action recognition, i.e., recognizing the actions of the camera wearers, which are completely different from the work proposed in this paper – our goal is to recognize the action of a performer present in the video. [19] developed an approach to recognize actions and interactions present in a wearable camera video by using iDT in foreground. This work, however, does not consider the feature stability – the foreground iDT contains both the motion of the action performer and the camera. [41] detect actions of interest from long streaming wearable-camera videos by directly extracting iDT, without computing stable features. Different from these works, in this paper we not only extract iDT, but also make them stable for action recognition.

Many recent action recognition works [9, 11, 32, 33, 38] include a step of **camera motion compensation** in extracting motion features, e.g., trajectory-based features, from videos taken by moving cameras. In [9], 2D affine motion models are used to approximate the camera motion between adjacent frames. In [38], a low rank optimization algorithm is developed to estimate camera-induced and object-induced motions: only the latter is used for action recognition. In [11], the motion patterns of dense trajectories [30] are clustered to characterize foreground-foreground or foreground-background relationships. In iDT [32, 33], dense trajectory locations [30] between two adjacent frames are transformed to a fixed view for camera motion compensation. In this paper, we extend iDT by making it stable over a video segment instead of only two adjacent frames.

Another group of related works are **cross-view action recognition** [4, 12, 14, 21, 34, 36, 39, 40], which extract view-invariant features from videos recorded in different views. In particular, [36] also used sparse coding for cross-view action recognition as in the proposed work. However, most of these works assume that videos are taken by fixed cameras. This is different from the proposed work where videos are taken by wearable cameras that may move and change view frame by frame. In the experiments, we will also show the performance of the proposed method on fixed-camera videos by comparing with several existing cross-view action recognition methods.

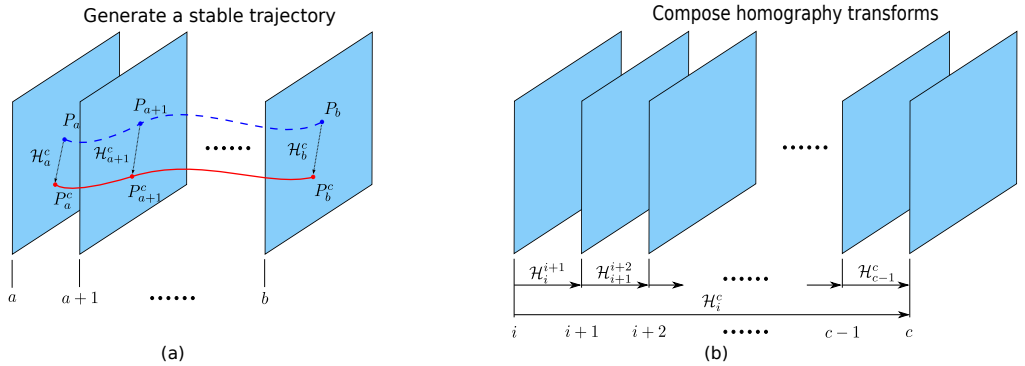

Figure 3: An illustration of (a) transforming a trajectory to a fixed camera view underlying an anchor frame c by (b) composing a sequence of adjacent-frame homography transforms.

3 PROPOSED METHOD

This paper is focused on video-based action recognition [17, 25, 29, 33], where each input video just contains one action of interest and tightly covers its duration. Our goal is to develop classifiers to recognize the action in this video. We will not tackle the more general problem of action detection from a long streaming video [20, 41]. As illustrated in Fig. 2, we uniformly divide an input video into a sequence of video segments and extract stable features for each video segment in terms of the view underlying an anchor frame. Finally we use a sparse-coding based algorithm to estimate the action likelihood for each video segment, followed by combining the likelihoods of all the video segments for action classification.

Many effective spatial-temporal motion features [9, 30–33] are defined on dense trajectories detected over multiple frames and/or optical flow calculated between adjacent frames. In the following, we first calculate stable dense trajectories and optical flow in terms of an anchor frame, and then introduce the anchor frame selection, stable feature definition and the proposed action classifiers.

3.1 Stable Dense Trajectories

Dense trajectories are points of interests and their tracking over a number of consecutive frames. In this paper, we use the iDT (improved dense trajectories) algorithm [33] to extract dense trajectories from the input video, where the trajectory length L, i.e., the number of frames traversed by the trajectory, can be priorly set. Let's consider one trajectory $(P_a, P_{a+1}, \ldots, P_b)$, $b = a + L - 1$, where P_i is the location of this trajectory on frame i, $a \le i \le b$. These L trajectory locations (and their corresponding frames) are generated under different camera views, as in iDT. Different from iDT, to make dense trajectories stable over a video segment, we transform these locations into a fixed camera view underlying an anchor frame c in this segment. We denote the resulting stable trajectory as $(P^c_a, P^c_{a+1}, \ldots, P^c_b)$. Obviously, if c is a frame traversed by the trajectory, i.e., $a \le c \le b$, we have $P^c_c = P_c$. We will elaborate on the selection of anchor frame later.

As illustrated in Fig. 3(a), for each trajectory point P_i, $i \ne c$, we estimate the homography transform $\mathcal{H}^c_i$ from frame i to frame c and calculate the stable location $P^c_i = \mathcal{H}^c_i(P_i)$. The estimation of the homography transform requires a set of matched feature points between the two frames and these matched feature points can be very difficult to detect when two frames are not adjacent or close to each other. To address this issue, we propose to estimate the homography transform between adjacent frames and then compose them to estimate a homography transform between non-adjacent frames. Without loss of generality, let's consider the case of $i < c$. We estimate the adjacent-frame homography transforms $\mathcal{H}^{j+1}_j$ from frame j to frame $j+1$, $j = i, i+1, \cdots, c-1$, by applying RANSAC [8] to the matched background feature points in terms of SURF descriptors and dense optical flow, as in [33]. Each estimated homography transform is represented by a homography matrix, e.g., $[h_{ij}]_{3\times3}$ and transforms a 2D location (x, y) to (x', y') as

$$\begin{cases} x' = \frac{h_{11}x+h_{12}y+h_{13}}{h_{31}x+h_{32}y+h_{33}} \\ y' = \frac{h_{21}x+h_{22}y+h_{23}}{h_{31}x+h_{32}y+h_{33}}. \end{cases} \quad (1)$$

By composing the adjacent-frame homography transforms, as illustrated in Fig. 3(b), we can construct the transform from frame i to anchor frame c by

$$\mathcal{H}^c_i = \mathcal{H}^c_{c-1} \circ \mathcal{H}^{c-1}_{c-2} \circ \cdots \circ \mathcal{H}^{i+1}_i. \quad (2)$$

Similarly, if $i > c$, we can construct the homography transform from frame i to anchor frame c by

$$\mathcal{H}^c_i = \mathcal{H}^c_{c+1} \circ \mathcal{H}^{c+1}_{c+2} \circ \cdots \circ \mathcal{H}^{i-1}_i. \quad (3)$$

3.2 Stable Optical Flow

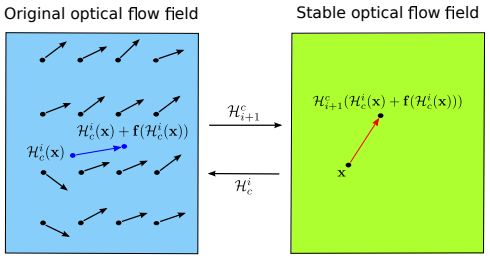

Figure 4: An illustration of calculating stable optical flow with view angle defined by anchor frame c. An original optical flow vector (blue) is transformed to a stable optical flow vector (red).

In this paper, we first use iDT to calculate optical flow on each frame of the video, and then transform this optical flow to a stable one, i.e., corresponding to a fixed view underlying anchor frame c. Note that the optical flow computed in iDT has compensated the camera motion between adjacent frames. In this paper, we

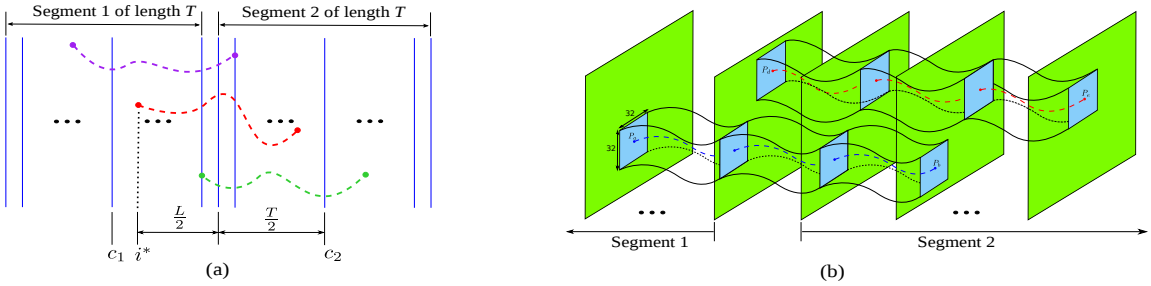

Figure 5: An illustration of (a) the temporal distance $|i^* - c|$ for composing homography transforms and (b) the bundling of the optical flow and trajectories.

further stabilize the optical flow by compensating camera motions between non-adjacent frames, i.e., between the anchor frame and one of its non-adjacent frames. Without loss of generality, let's consider the optical flow calculated on frame i which we denote as $\mathbf{f}(\mathbf{x})$. At each pixel $\mathbf{x} = (x, y)$, the optical flow vector actually points from the location $\mathbf{x}$ in frame i to its corresponding location $\mathbf{x} + \mathbf{f}(\mathbf{x})$ in frame $i + 1$. One straight-forward approach to make it view consistent is to apply the one-way homography transform to transform both endpoints of the flow vector to the view-consistent ones in terms of frame c. However, the transformed location $\mathcal{H}_i^c(\mathbf{x})$ may not show integer $x-$ and $y-$coordinates and the transformed optical-flow vector field may not uniformly cover the image. We need to perform 2D interpolation on the transformed non-uniform 2D vector field to estimate the view-consistent optical flow at each pixel, which can introduce high inaccuracies.

We propose a two-way homography transform to address this problem. As illustrated in Fig. 4, starting from each pixel $\mathbf{x}$ on frame i after transforming to the fixed view, we first estimate its corresponding location on the original frame i as $\mathcal{H}_c^i(\mathbf{x})$. On the original frame i, we have uniform optical flow field $\mathbf{f}(\cdot)$ available for each integer-coordinate pixel. Therefore, even if $\mathcal{H}_c^i(\mathbf{x})$ has non-integer coordinates, we can perform 2D interpolation to get an accurate estimation of the optical flow at this location, which we denote as $\mathbf{f}(\mathcal{H}_c^i(\mathbf{x}))$. This way, we estimate the flow location on the original frame $i + 1$ as $\mathcal{H}_c^i(\mathbf{x}) + \mathbf{f}(\mathcal{H}_c^i(\mathbf{x}))$ and its stable location in the fixed view underlying frame c is $\mathcal{H}_{i+1}^c(\mathcal{H}_c^i(\mathbf{x}) + \mathbf{f}(\mathcal{H}_c^i(\mathbf{x})))$. Based on this, after transforming to the fixed view, at each pixel $\mathbf{x}$ on frame i, we can calculate a stable flow vector as

$$\mathcal{H}_{i+1}^c(\mathcal{H}_c^i(\mathbf{x}) + \mathbf{f}(\mathcal{H}_c^i(\mathbf{x}))) - \mathbf{x},$$

which leads to a stable optical flow field on frame i.

3.3 Stable Features

As mentioned above, the homography transform between non-adjacent frames is computed by composing a sequence of adjacent-frame homography transforms. In practice, each adjacent-frame homography transform may contain an error and such errors may accumulate in composition. The larger the value of $|i - c|$, i.e., the temporal distance between frame i and the anchor frame c, the higher the accumulated errors in constructing $\mathcal{H}_c^i$ and $\mathcal{H}_i^c$. In this paper, we relieve such accumulation errors by uniformly dividing the input video into N shorter video segments, as well as selecting an appropriate anchor frame for each video segment.

More specifically, we first apply iDT algorithm to detect length-L dense trajectories on the input video and assign each detected trajectory to the video segment with the largest temporal overlap. Let T be the length of video segment for the considered input video. Typically, we have $L < T$ and by picking the *middle frame as the anchor frame* c for a video segment, we will have smallest possible $|i^* - c| \approx \frac{T+L}{2}$, where i^* indicates the furthest frame (from frame c) that may contain a trajectory location to be transformed to the view underlying frame c, as shown by the red trajectory in Fig. 5(a). For optical flow, it is calculated on each frame using the algorithm in [7]. As in [33], optical flow is bundled with trajectories in defining motion features: optical flow near/around a trajectory, e.g., a 32×32-pixel region around the trajectory on each frame, is included to the video segment to which the trajectory is assigned. As a result, the frames traversed by the trajectory will be assigned to the same video segment and their optical flow will be transformed to the corresponding fixed view. Some frames may be traversed by multiple trajectories that are assigned to different video segments. The optical flow in these frames, e.g., the three middle frames in Fig. 5(b), need to be transformed to different fixed views when bundled to different trajectories. Given the bundling of optical flow and trajectories, we have the same $|i^* - c| \approx \frac{T+L}{2}$ in computing stable optical flows.

To improve the robustness and efficiency of feature extraction, we only extract features associated to the detected person, in the form of a rectangular bounding box on each frame. As in [33], we calculate the motion features of HOF (histograms of optical flow) [3] and MBH (motion boundary histogram) [31] for each video segment, but using the proposed stable trajectories and optical flow. Stable trajectories that are partly located outside any involved frame will be discarded in calculating these motion features. Also following [33], we calculate the appearance feature of HOG (histograms of oriented gradients) [5] without any further transforms. For each video segment, we finally construct a feature vector by concatenating Fisher vectors [23] of stable trajectories, HOG, HOF and MBH, with normalizations. Camera motion over a video segment has been compensated in computing the motion features and therefore, the motion features are consistent to the ones extracted from a fixed camera over the video segment. Together with the video-segment based action classifiers to be discussed in the following, we can achieve the goal of training the action classifiers on fixed-camera videos and then applying them to process wearable-camera videos. The original iDT can only provide feature stabilization between

Figure 6: Exemplar frames of the collected videos for the eight different actions of interest, where each column shows one action separately, The first and second rows show the videos collected in two different scenarios respectively.

two adjacent frames, and this local stabilization could not provide sufficient information for action classification.

3.4 Action Classifiers

Based on the stable features extracted in each video segment, we use a sparse-coding model for supervised action recognition [2]. More specifically, we estimate the likelihood of action k on a video V as $\mathcal{P}(k|V) = \frac{1}{N}\sum_{j=1}^{N}\mathcal{P}(k|V^j)$, where V^j is the j-th video segment and

$$\mathcal{P}(k|V^j) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-\|\mathbf{p}^j - \mathbf{A}_k^j\mathbf{z}^*\|^2}{2\sigma^2}\right) \quad (4)$$

with $\mathbf{p}^j$ being the stable feature extracted from video segment V^j, $\mathbf{A}_k^j$ being the basis matrix for the j-th segment of the action k. The sparse-coding coefficient $\mathbf{z}^*$ can be calculated by

$$\mathbf{z}^* = \min_{\mathbf{z}}\{\|\mathbf{p}^j - \mathbf{A}_k^j\mathbf{z}\|^2 + \lambda\|\mathbf{z}\|_0\}. \quad (5)$$

The basis matrix $\mathbf{A}_k^j$ is constructed from a set of training video samples of action k – Each row of $\mathbf{A}_k^j$ is the feature vector extracted from the j-th segment of a training video. We use the $\mathbf{L}^1$ norm approximation via Orthogonal Matching Pursuit (OMP) implementation [26] to solve this minimization problem.

As mentioned in Section 1, given that the features extracted in each video segment is stable, i.e., corresponding to a fixed view, we only need collect training video samples such that 1) the camera view stay unchanged in each video segment, 2) training videos well cover all possible camera views in each video segment. This can be achieved by using fixed-camera videos – *for each action, we simply use fixed cameras to collect multiple videos of this action from different views as the training set.* In practice, due to the use of the sparse-coding model, we only need to collect videos from a sparsely sampled set of views for training. The constructed basis matrix $\mathbf{A}_k^j$ can help represent features from other views not covered in the training videos. This not only reduces the load and difficulty in collecting the training videos, but also improves the algorithm efficiency.

4 EXPERIMENTS

4.1 Dataset and Experimental Setup

The proposed method takes fixed-camera videos from different views for training (as the basis of the sparse-coding matrix) and then test on videos taken by wearable cameras with complex motions. We did not find existing video datasets that are appropriate for our experiments. Therefore, we collect our own video dataset, which will be released to public. We choose eight human actions that are very common in daily life: *jump*, *handwave*, *pickup*, *punch*, *standup*, *sitdown*, *kick*, *throw*. Each of the collected video reflects one of these actions, as illustrated in Fig. 6. In each video, only one person (the performer of action) is present and the video temporally well cover the duration of the underlying action. The average length of the collected videos is approximately 3.5 seconds. All the videos are recorded by GoPro cameras with 30fps frame rate. Each frame is downsampled to the spatial resolution of 720×405 pixels.

Training data are collected in an outdoor setting by fixed cameras that faces the performer from eight uniformly distributed horizontal views. For each action, we take ten videos from each view, resulting in 640 training videos in total. For the testing data, each video also contains one action performed by one person. But it is taken by a GoPro camera worn by a camera wearer over his head. The wearer freely moves in recording the action of the performer – the wearer's motion includes both body motion, such as walking, and head motion, such as head rotation. Four different performers and five different camera wearers are used to collect the testing videos. The testing videos are recorded in two different outdoor scenarios. One of them is in an area surrounded by buildings which is the same scenario where we collect the training videos. The other scenario is an open playground. We collected 237 and 171 testing videos in the first and second scenarios respectively, which leads to 408 videos in total, with 51 testing videos for each action. We annotate these videos for the performer (in the form of a bounding box on each frame) and each instance of the actions, by using the open-source video annotation tool of VATIC [24].

For feature descriptors, we choose trajectory length $L = 9$. Other than using the proposed stable iDT and stable optical flow to replace the original iDT and optical flow, all the other settings and parameters in defining HOG and MBH follow [33]. Different from [33], we use 8 bins without an additional zero bin, for HOF. In particular, optical flow is bundled to nearby trajectories for defining these feature descriptors and their bundling relation is defined by a distance no more than 32 pixels in both x- and y-directions on each frame. The dimensions of these descriptors are: 18 for stable trajectories, 96 for HOG, 96 for HOF and 192 for MBH.

We use the fisher vector [23] for feature encoding., For each type of feature descriptor, we construct a 256-component Gaussian Mixture Model (GMM) using 256,000 randomly sampled features from our training data. We follow the setting in [23] to encode features into fisher vectors and normalize them, within each segment of each video. Finally, we concatenate the normalized fisher vectors of different feature descriptors as the feature vector of this video segment.

One sparse-coding (SC) classifier is trained for each action. We divide each video (either training and testing one) into $N = 10$ equal-length video segments. We set parameters $\sigma = 6$ in Eq. (4) and $\lambda = 10^{-2}$ in Eq. (5). The action $k^* = \arg\max_k \mathcal{P}(k|V)$ with the highest likelihood will be taken as the recognized action for video V.

4.2 Results

Figure 7 shows the action recognition accuracies of the proposed method on the testing data, when dividing each video into different number of video segments. We can see that, when dividing a video into $N = 10$ segments, we achieve the highest accuracy of 70.1%. An overly large N may lead to very short video segments with low-discriminative motion features. An overly small N, e.g., $N = 1$ (no video subdivision), may lead to very long video segments with high accumulation errors in composing homography transforms for computing stable features. We use $N = 10$ for the remaining experiments.

Figure 8 shows the recognition accuracy for each action class in each recording scenario. Figure 9 shows the confusion matrix of the proposed method on the testing data. We can see that the performance on the testing data collected in the second scenario is not good as that in the first scenario. This is partly due to the use of the first scenario for collecting the training data. Actions *jump* and *kick* show high accuracy, because their features are more discriminative than other action classes. Recognition accuracy of *handwave* are low since they only involve relatively small-scale hand movements.

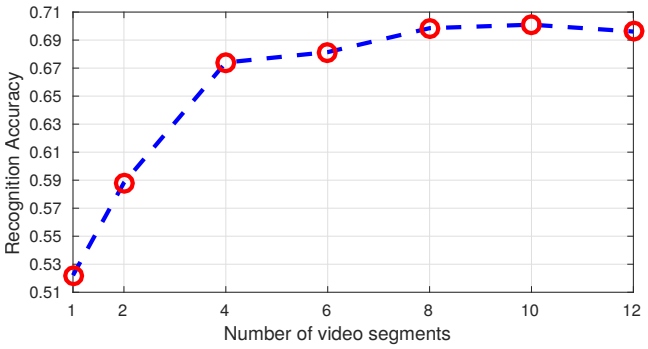

Figure 7: The testing performance of the proposed method, when dividing each video into different number of video segments.

Figure 10 shows the accumulation error in composing homography transforms over a temporal distance $|i - c|$. We can see that, with the increase of $|i - c|$, the composed homography transform gets more inaccurate – the (red) edge points in the background of anchor frame c are dislocated from the edge after applying the composed homography transform over a sequence of 11 frames, i.e., $i = c - 11$. In this paper, we divide each video into shorter video segments to avoid such accumulation errors.

We choose three existing methods for comparison. The first one is Zheng et al [41], where videos from two different fixed views are taken for training. It uses traditional iDT, HOG, HOF and MBH features, encoded by bag-of-features (BOF). Sparse coding is used as the classifiers. The second one is Wang et al [32], which detects iDT

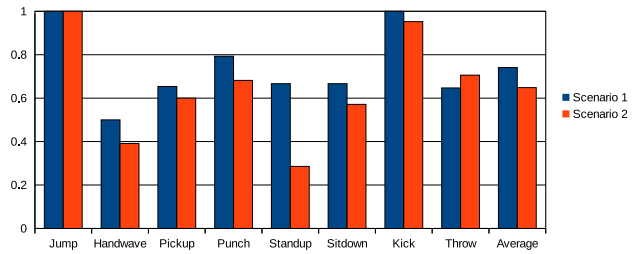

Figure 8: The recognition accuracy of the proposed method for each action class and each recording scenario.

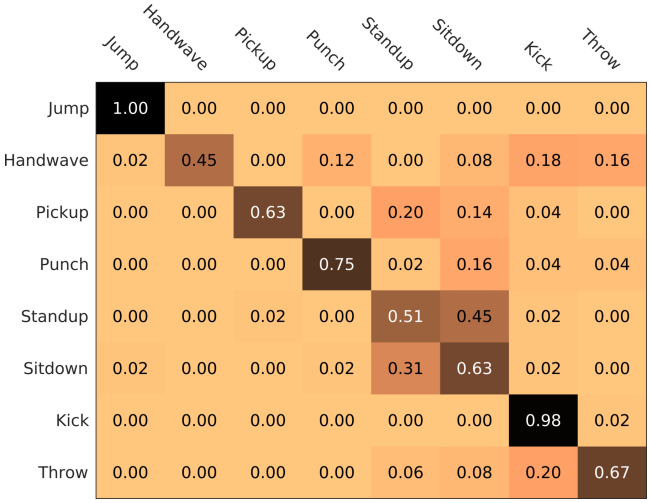

Figure 9: Confusion matrix of the proposed method on the testing data.

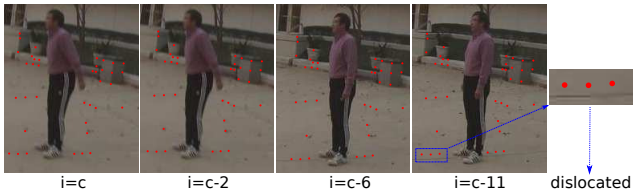

Figure 10: An illustration of accumulation error in composing homography transforms.

for action recognition, with camera motion compensated between adjacent frames. It uses iDT, HOG, HOF and MBH features encoded by the fisher vector [23]. A linear SVM (support vector machine) is used as the classifiers. The third one is Tran et al [29], where deep 3D convolutional networks are used to learn deep features. All the deep features are extracted from fixed-length video clips without any further feature encoding, followed by using linear SVM as the classifiers. Wang et al [32] and Tran et al [29] have reported state-of-the-art performances on several challenging action recognition datasets, including UCF101 [28], which contains videos recorded by moving cameras. For all the three comparison methods, we use their default settings and parameters.

Method	Training Set	Testing Set	Classifier	Accuracy
Zheng et al [41]	Fixed-camera videos from eight views	All wearable-camera videos	Sparse coding	25%
Tran et al [29]	Fixed-camera videos from eight views	All wearable-camera videos	linear SVM	20.1%
	One subset of wearable-camera videos	The other subset of wearable-camera videos	linear SVM	34.1%
Wang et al [32]	Fixed-camera videos from eight views	All wearable-camera videos	linear SVM	54.4%
	One subset of wearable-camera videos	The other subset of wearable-camera videos	linear SVM	45.8%
Baseline	Fixed-camera videos from eight views	All wearable-camera videos	Sparse coding	55.6%
Proposed + linear SVM	Fixed-camera videos from eight views	All wearable-camera videos	linear SVM	62.3%
Proposed	Fixed-camera videos from eight views	All wearable-camera videos	Sparse coding	**70.1**%

Table 1: Comparison results against several existing action recognition methods on the testing dataset.

From Table 1, we can see that, when trained on fix-camera videos and testing on wearable-camera videos, all three comparison methods obtain much lower action recognition accuracy, than the proposed method, due to serious feature inconsistency between training data and testing data. Another approach to address this feature inconsistency problem is to use wearable camera videos for training. We conduct additional experiments for Wang et al [32] and Tran et al [29] by randomly separating all the collected wearable camera videos into two subsets of similar size: one subset used for training and the other subset used for testing. This way, we can perform two-fold cross-validation for performance evaluation. From Table 1, we can see that, by using wearable camera videos for training, Wang et al [32] even does not show any performance improvement over the use of fixed camera videos for training. The main reason is the complexity of the wearable camera motion in practice: it is difficult to use a limited number of training videos to cover all the possible motions of wearable cameras in the duration of an action.

The effectiveness of sparse-coding (SC) classifier is also verified: if we replace them by linear SVMs in the proposed method, the action recognition accuracy substantially decreases, as shown in the row of "Proposed + Linear SVM" in Table 1. The main reason is that sparse coding can linearly combine the features from seen views to approximate the features from unseen views. In Table 1, we also include the performance of a "Baseline" method, which follows the general setting of the proposed method, but does not divide each video into segments, i.e., $N = 1$, and does not transform the trajectories and optical flow to stable ones. We can see that "Baseline" performs much worse than the proposed method with $N = 10$. This further verifies the effectiveness of the proposed video subdivision and feature stabilization methods.

Figure 11 shows the recognition accuracy in terms of each action class, using the proposed method and the three comparison methods, respectively. Note that, for Tran et al [29] and Wang et al [32], their performances are dependent on the adopted training set. As shown in Table 1, Tran et al [29] performs better when using wearable camera videos for training, while Wang et al [32] performs better when using 640 fixed camera videos for training. In Fig. 11, we report the best performance for each of these comparison methods. We can see that, the proposed method outperforms the comparison methods on six out of eight action classes, and achieves the second highest performance in the other two action classes. We notice that the comparison methods also show low accuracy on the

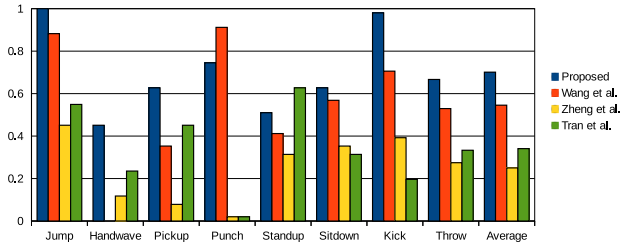

Figure 11: Per-class action recognition accuracy of the proposed method and comparison methods on the testing dataset.

action of *handwave*, where Wang et al [32] obtains an accuracy of indeed 0%, due to only small-scale motion of hand.

4.3 Results on Fixed-Camera Videos

We also conduct experiments on a multi-view action recognition dataset IXMAS [37], to further test the performance of the proposed method on fixed-camera videos, and show the effectiveness of the proposed video-subdivision approach and the sparse-coding classifier. The IXMAS dataset contains twelve human actions, each of which is performed three times by eleven actors, and recorded by four side view cameras and one top view camera. The proposed method is trained on the data from four views and tested on the data from the remaining view. We still divide each video into 10 video segments as in Section 4.2.

Method	Cam1	Cam2	Cam3	Cam4	Cam5	Average
Junejo et al [12]	66.6%	65.5%	65.0%	62.4%	49.6%	61.9%
Yan et al [39]-1	75.5%	74.6%	77.1%	69.7%	63.3%	72.1%
Yan et al [39]-2	77.3%	74.1%	79.3%	71.3%	65.1%	73.4%
Wang et al [36]	81.9%	82.1%	80.5%	78.5%	73.0%	79.2%
Ciptadi et al [4]	83.9%	81.8%	87.6%	83.0%	73.6%	82.0%
Proposed	**97.5%**	**98.2%**	**95.5%**	**91.2%**	**79.0%**	**92.3%**

Table 2: Action recognition performance of the proposed method and several comparison methods on the IXMAS dataset.

Figure 12 shows the confusion matrix of the proposed method on the IXMAS dataset. Table 2 shows the recognition accuracies of the proposed method and several comparison cross-view action recognition methods, on this dataset. Cam *i* indicates the case of

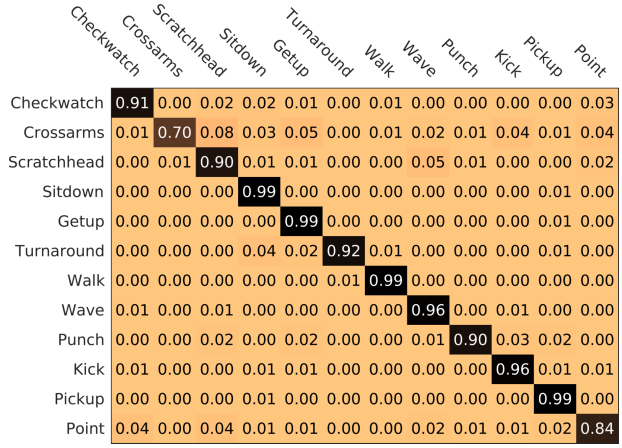

Figure 12: Confusion matrix of the proposed method on the IXMAS dataset.

testing on data from view-*i*, and training on data from other four views. We can see that, in general, the action recognition accuracies on fixed-camera videos are much higher than those on wearable-camera videos. On this dataset, the proposed method outperforms all the comparison methods. We believe this owes to the capability of sparse coding in using motion features taken from very few sparse views to represent the ones taken from many other views. The result proves the effectiveness of the proposed video subdivision approach and the sparse-coding classifier. It also shows that the proposed method can deal with the fixed-camera videos. The recognition accuracies of *crossarms* and *point* are low due to their small-scale motions involving only a small part of the body. Performance on "Cam5" is worse, because the classifiers are trained on the data from four side views and tested on the data from the top view.

5 CONCLUSION

This paper introduced an approach to recognizing human actions in wearable-camera videos by training classifiers on fixed-camera videos. In this method, we first divided a video of an action into a sequence of shorter video segments, and then extracted stable motion features in each video segment, by transforming them to a fixed view defined by an anchor frame in this segment. We then used a sparse-coding model to compute the action likelihood for each video segment independently, and finally averaged the likelihoods over all the video segments for action recognition. The combination of video subdivision and feature stabilization enables the motion feature consistency between training (fixed-camera) videos and testing (wearable-camera) videos. We tested the proposed method on both wearable/fixed-camera videos, with competitive results. We believe the proposed method can also handle action recognition in videos taken by other moving cameras, besides wearable cameras. We plan to study this in the future.

Acknowledgment This work was supported in part by NSFC-61672376 and NSF-1658987.

REFERENCES

[1] Alejandro Betancourt, Pietro Morerio, Carlo S Regazzoni, and Matthias Rauterberg. 2015. The evolution of first person vision methods: A survey. *IEEE Transactions on Circuits and Systems for Video Technology* 25, 5 (2015), 744–760.

[2] Yu Cao, Daniel Barrett, Andrei Barbu, Siddharth Narayanaswamy, Haonan Yu, Aaron Michaux, Yuewei Lin, Sven Dickinson, Jeffrey Mark Siskind, and Song Wang. 2013. Recognize human activities from partially observed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2658–2665.

[3] Rizwan Chaudhry, Avinash Ravichandran, Gregory Hager, and René Vidal. 2009. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1932–1939.

[4] Arridhana Ciptadi, Matthew S Goodwin, and James M Rehg. 2014. Movement pattern histogram for action recognition and retrieval. In *Proceedings of the European Conference on Computer Vision*. 695–710.

[5] Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1. 886–893.

[6] César Roberto de Souza, Adrien Gaidon, Eleonora Vig, and Antonio Manuel López. 2016. Sympathy for the details: Dense trajectories and hybrid classification architectures for action recognition. In *Proceedings of the European Conference on Computer Vision*. 697–716.

[7] Gunnar Farnebäck. 2003. Two-frame motion estimation based on polynomial expansion. *Image analysis* (2003), 363–370.

[8] Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (1981), 381–395.

[9] Mihir Jain, Herve Jegou, and Patrick Bouthemy. 2013. Better exploiting motion for better action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2555–2562.

[10] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2013. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 1 (2013), 221–231.

[11] Yu-Gang Jiang, Qi Dai, Xiangyang Xue, Wei Liu, and Chong-Wah Ngo. 2012. Trajectory-based modeling of human actions with motion reference points. In *Proceedings of the European Conference on Computer Vision*. 425–438.

[12] Imran N Junejo, Emilie Dexter, Ivan Laptev, and Patrick Perez. 2011. View-independent action recognition from temporal self-similarities. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 1 (2011), 172–185.

[13] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1725–1732.

[14] Yu Kong, Zhengming Ding, Jun Li, and Yun Fu. 2017. Deeply Learned View-Invariant Features for Cross-View Action Recognition. *IEEE Transactions on Image Processing* 26, 6 (2017), 3028–3037.

[15] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. 2011. HMDB: A large video database for human motion recognition. In *Proceedings of the IEEE International Conference on Pattern Recognition*. 2556–2563.

[16] Yingwei Li, Weixin Li, Vijay Mahadevan, and Nuno Vasconcelos. 2016. Vlad3: Encoding dynamics of deep features for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1951–1960.

[17] Xiaodan Liang, Liang Lin, and Liangliang Cao. 2013. Learning latent spatio-temporal compositional model for human action recognition. In *Proceedings of the ACM Multimedia Conference*. 263–272.

[18] Minghuang Ma, Haoqi Fan, and Kris M Kitani. 2016. Going deeper into first-person activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1894–1903.

[19] Sanath Narayan, Mohan S Kankanhalli, and Kalpathi R Ramakrishnan. 2014. Action and interaction recognition in first-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 512–518.

[20] Xiaojiang Peng and Cordelia Schmid. 2016. Multi-region two-stream R-CNN for action detection. In *Proceedings of the European Conference on Computer Vision*. 744–759.

[21] Hossein Rahmani and Ajmal Mian. 2015. Learning a non-linear knowledge transfer model for cross-view action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2458–2466.

[22] Michael S Ryoo and Larry Matthies. 2013. First-person activity recognition: What are they doing to me?. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2730–2737.

[23] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. 2013. Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision* 105, 3 (2013), 222–245.

[24] Christian Schuldt, Ivan Laptev, and Barbara Caputo. 2004. Recognizing human actions: A local SVM approach. In *Proceedings of the IEEE International Conference on Pattern Recognition*, Vol. 3. 32–36.

[25] Paul Scovanner, Saad Ali, and Mubarak Shah. 2007. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the ACM Multimedia Conference*. 357–360.

[26] Victor Shia, Allen Y Yang, S Shankar Sastry, Andrew Wagner, and Yi Ma. 2011. Fast l1-minimization and parallelization for face recognition. In *ASILOMAR on Signals, Systems, and Computers*. 1199–1203.

[27] Karen Simonyan and Andrew Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the Neural Information Processing Systems Conference*. 568–576.

[28] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).

[29] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 4489–4497.

[30] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. 2011. Action recognition by dense trajectories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3169–3176.

[31] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. 2013. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision* 103, 1 (2013), 60–79.

[32] Heng Wang, Dan Oneata, Jakob Verbeek, and Cordelia Schmid. 2016. A robust and efficient video representation for action recognition. *International Journal of Computer Vision* 119, 3 (2016), 219–238.

[33] Heng Wang and Cordelia Schmid. 2013. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*. 3551–3558.

[34] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. 2014. Cross-view action modeling, learning and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2649–2656.

[35] Limin Wang, Yu Qiao, and Xiaoou Tang. 2015. Action Recognition With Trajectory-Pooled Deep-Convolutional Descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4305–4314.

[36] Wei Wang, Yan Yan, Luming Zhang, Richang Hong, and Nicu Sebe. 2016. Collaborative sparse coding for multiview action recognition. *IEEE Transactions on Multimedia* 23, 4 (2016), 80–87.

[37] Daniel Weinland, Remi Ronfard, and Edmond Boyer. 2006. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding* 104, 2 (2006), 249–257.

[38] Shandong Wu, Omar Oreifej, and Mubarak Shah. 2011. Action recognition in videos acquired by a moving camera using motion decomposition of lagrangian particle trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*. 1419–1426.

[39] Yan Yan, Elisa Ricci, Ramanathan Subramanian, Gaowen Liu, and Nicu Sebe. 2014. Multitask linear discriminant analysis for view invariant action recognition. *IEEE Transactions on Image Processing* 23, 12 (2014), 5599–5611.

[40] Jingjing Zheng and Zhuolin Jiang. 2013. Learning view-invariant sparse representations for cross-view action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*. 3176–3183.

[41] Kang Zheng, Yuewei Lin, Youjie Zhou, Dhaval Salvi, Xiaochuan Fan, Dazhou Guo, Zibo Meng, and Song Wang. 2014. Video-based action detection using multiple wearable cameras. In *Proceedings of the European Conference on Computer Vision Workshops*. 727–741.