Dynamic view selection for multi-camera action recognition

Article *in* Machine Vision and Applications · September 2015

DOI: 10.1007/s00138-015-0715-9

Richard Souvenir
Temple University

Machine Vision and Applications (2016) 27:53–63
DOI 10.1007/s00138-015-0715-9

ORIGINAL PAPER

Dynamic view selection for multi-camera action recognition

Scott Spurlock[1] · Richard Souvenir[2]

Received: 14 October 2014 / Revised: 21 July 2015 / Accepted: 26 August 2015 / Published online: 28 September 2015

Abstract For multi-camera human action recognition methods, there is often a trade-off between classification accuracy and computational efficiency. Methods that generate 3D models or query all of the cameras in the network for each target are often computationally expensive. In this paper, we present an action recognition method that operates in a multi-camera environment, but dynamically selects a single camera at a time. We learn the relative utility of a particular viewpoint compared with switching to a different available camera in the network for future classification. We cast this learning problem as a Markov Decision Process, and incorporate reinforcement learning to estimate the value of the possible *view-shifts*. On two benchmark multi-camera action recognition datasets, our method outperforms approaches that incorporate all available cameras in both speed and classification accuracy.

Keywords Camera networks · Human motion analysis · Reinforcement learning · Markov Decision Process

1 Introduction

Distributed camera networks are increasingly being used for applications in surveillance, sports analysis, and smart homes. In these types of networks, algorithms can leverage the multiple views to overcome occlusions and pose ambiguities. In general, recognition accuracy increases with the number of cameras (e.g., [38,43]), but usually at the cost of computational efficiency; existing approaches to multi-camera action recognition tend to be too computationally expensive for real-time application. Two major factors affecting computational efficiency by current multi-view methods are the cost of computing features for every camera and the overhead incurred by the data aggregation scheme.

We present an approach to multi-camera action recognition that dynamically selects a single camera for use at each time step. Our method is based on the observation that certain poses, *keyposes*, are highly discriminative and can provide much of the evidence needed for a prediction by the system. This observation is not new; Schindler and Van Gool [22] framed the problem of action recognition as detecting poses that are highly predictive for particular action classes. We call such poses *class-discriminative*. Our work extends this observation to *shift-discriminative* poses. Shift-discriminative poses are not predictive of a particular class, but indicate potential ambiguity is imminent from the current viewpoint and a *view-shift* is necessary. Figure 1 shows an example containing both types of discriminative poses where a view-shift leads to a view containing a class-discriminative pose. The addition of shift-discriminative poses extends the utility of class-discriminative poses to the multi-camera setting.

In this paper, which is an extended version of [25], we describe our approach to view-shift learning for action recognition. Our method casts view-shift learning as a Markov decision process and employs reinforcement learning to estimate the utility of the available viewpoints in a distributed camera network for human action recognition. Our approach does not require the camera configuration to be the same in training and testing and is applicable to both batch and

✉ Scott Spurlock
sspurlock@elon.edu

Richard Souvenir
souvenir@uncc.edu

[1] Department of Computing Sciences, Elon University, Elon, USA

[2] Department of Computer Science, University of North Carolina at Charlotte, Charlotte, USA

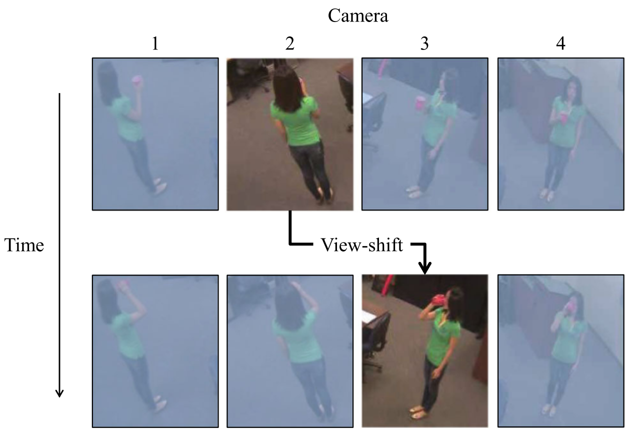

Fig. 1 Our method learns if a pose represents potential future ambiguity (*top*) and which available viewpoints will facilitate disambiguation (*bottom*). For this "drinking" example, the active camera is changed via a learned *view-shift* to a new view with a more discriminative pose

online recognition. We compare our view-shift method with recent multi-camera approaches and show that it is comparable to the state of the art on two standard benchmark datasets in terms of recognition accuracy, but with greatly reduced computational effort.

2 Related work

There has been extensive work in human action recognition from video; see [20,36] for recent surveys. Our focus in on multi-camera action recognition in indoor environments. Methods designed for these scenarios often must balance the gains in accuracy from computationally expensive inference schemes with the costs in efficiency. In this section, we organize the related approaches based upon the explicit use of 3D models and/or information from all of the cameras in the network.

2.1 4D Recognition

One approach in multi-camera networks is to make use of multi-view geometry to explicitly construct 3D models and solve 4D (3D plus time) recognition problems; several recent methods follow this general approach. Motion history volumes (MHV) encode a person's spatial and temporal motion using the circular Fast Fourier Transform [35]. Turaga et al. incorporate MHV in a manifold-learning-based scheme [30]. These methods are computationally expensive, requiring, in addition to models constructed during training, dynamic construction of models from multi-camera input during testing. Some methods construct 3D models during training, but not testing. In [33], 3D silhouette-based exemplars are learned and matched to 2D observations with model projections. Yan et al. [39] construct 4D models and perform recognition by back projecting 2D features. While these methods require less computational expense at test time than approaches that construct 3D models for both training and recognition, they still incur significant overhead due to the size of the search space to fit model parameters [34].

2.2 Multi-view recognition without 3D models

Other approaches, rather than explicitly constructing 3D models, use a set of 2D image views with some aggregation scheme. Many methods simply extend single-view algorithms by, for example, concatenating features from multiple cameras (e.g., [5,26,38]) or using voting schemes among the cameras in the network [16,19,43]. Rather than extending single-view methods to camera networks, some approaches present new features designed for the multi-view setting. Souvenir and Babbs introduce a manifold-based model to estimate how observed 2D features change as a function of viewpoint [24]. Farhadi et al. define discriminative aspect to encode how actions look different from different perspectives [6]. In general, these approaches tend to be computationally more efficient than the 4D methods, but still require computation to occur for each camera in the network in addition to the cost of aggregation. Our approach neither fits 3D models nor requires computation at each camera per target at test time. Our approach learns to select the single best view for recognition dynamically.

2.3 Cross-view recognition

A variant of action recognition that shares similarities to our problem is cross-view recognition, where the goal is to recognize actions from cameras unseen during training. Recent methods (e.g., [15,41,42]) that address this problem most closely follow the style of multi-view methods without 3D models. Our method is not intended to address the problem of cross-view recognition, but instead seeks to predict how to switch views during the recognition process to improve classification.

2.4 Best view selection

Other work has also considered the idea of selecting a single best view. Rudoy et al. [21] present a method to identify the best viewpoint for an action from a human observer's standpoint based on motion features. Other approaches select the best view based on the number of spatiotemporal features detected [37], estimated camera distances and person orientation [23], or silhouette properties [17]. Unlike our approach, which learns to predict the next best view based solely on the current view, these methods still require features to be computed at test time for every camera at each time step. In addition, rather than using proxy measurements of motion, orientation, or silhouette properties, our approach directly incorporates predicted discriminability as the view selection criterion.

3 Method

Our action recognition method is designed for systems of calibrated cameras. In particular, we consider architectures with peer camera nodes, where for a given target, there is an *active* camera for detection and tracking, and the remaining cameras are passive (with respect to a particular target). The initial designation can be fixed, where a particular camera is active for targets in a specified region, or dynamic, e.g., the active camera is selected to track a target [31]. In either case, passive views are represented by their relative offset from the active camera.

Under this model, each camera performs single-view recognition independently. There is a large amount of work in the area of image and video feature representations for human activity (e.g., temporal templates [2], STIP [14], HOG3D [13], Motion Context [28]); our approach is not specific to a particular feature representation. We follow the common approach (e.g., [4,16,29,40]) of feature quantization to learn a dictionary of keyposes. As illustrated in Fig. 2, for most modern feature representations and reasonable quantization schemes, certain clusters will correspond to *class-discriminative* keyposes, namely when the component elements from the cluster are mainly from a single class. Certain neutral or ambiguous poses, however, may be shared among multiple actions. Our model seeks to learn if these keyposes are *shift-discriminative* for a *view-shift* to a new active camera in the network, with the goal of correctly predicting the action. In this section, we describe our keypose learning (Sect. 3.1), training (Sect. 3.2) and recognition (Sect. 3.3) steps.

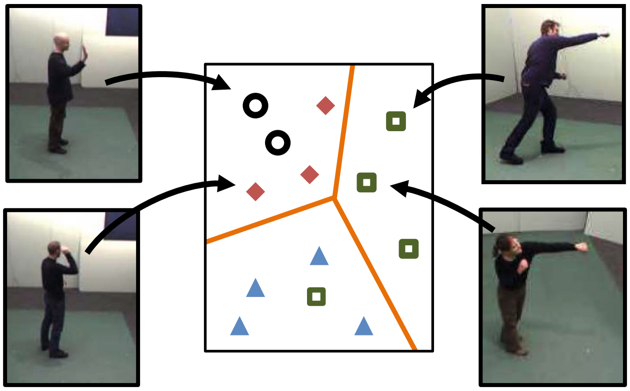

Fig. 2 Keyposes are learned by clustering the feature descriptors for each frame in the training set. Some clusters (*left*) are heterogeneous and include poses from multiple classes. Homogeneous clusters (*right*) represent class-discriminative keyposes and contain primarily examples from the same class

3.1 Keypose learning

The training input is a set of synchronized image streams, $\mathcal{I} = \{\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_C\}$ from C different cameras. Each stream, $\mathbf{I}_j = \{I_{j,1}, I_{j,2}, \ldots, I_{j,N}\}$, consists of N frames and could include example actions from a variety of actors and relative positions. Let $\{y_i\}$ represent the associated class labels for each time step, i.

The first step is to extract features for each frame and learn a keypose dictionary. Let $\mathcal{X} = \{x_1, \ldots, x_K\}$ represent the learned dictionary of keyposes where K is the user-specified dictionary size.

The class-discriminative power of each keypose is estimated by the distribution $P(x_k \mid y)$, which is computed from the training data using Laplace smoothing:

$$P(x_k \mid y) = \frac{N_D + 1}{N_A + K} \tag{1}$$

where N_A is the total number of training frames with label y and N_D is the size of the subset of those frames assigned to keypose x_k. A keypose, x_k, is considered class-discriminative if the correct label, y^*, is most likely, i.e., $\operatorname{argmax}_{y'} P(x_k \mid y') = y^*$. Figure 3 shows a diagram representing typical input. Each frame of each camera stream is matched to a keypose, represented by the intensity in the figure. The class-discriminative keyposes, i.e., the ones most likely to lead to correct classification, are boxed in green. In this example, cameras 1, 3, and 4 each contain class-discriminative keyposes and would likely lead to a correct classification in a single-view setting. Conversely, cameras 2 and 5 see no such poses. Our goal is to learn when to view-shift from viewpoints with potential ambiguity (e.g., 2 and 5, in this example) to viewpoints with a higher likelihood of observing class-discriminative poses.

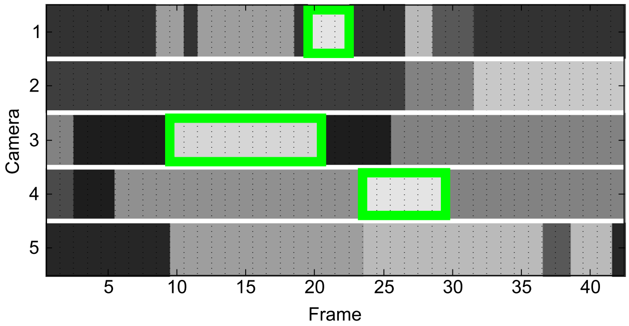

Fig. 3 An action sequence captured by five cameras is shown with each frame's quantized keypose indicated by intensity. Class-discriminative keyposes are indicated with a *green box*

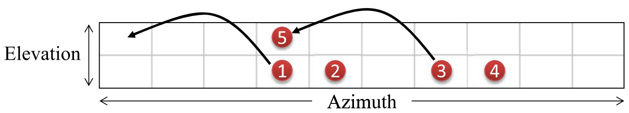

Fig. 4 Each bin represents a discretized portion of the view-sphere for a particular target; in this case, there are 10 azimuth and 2 elevation bins. The *red circles* indicate cameras, and the *arrows* represent view-shifts. Both *arrows* indicate the same view-shift: $\langle +3, -1\rangle$. Relative view-shifts allow our method to learn the utility of view-shifts to viewpoints unavailable during training

3.2 Learning view-shifts

For a given target, the positions of the cameras in a network can be described by a half-sphere centered on the target. A view-shift is a change of viewpoint based on a relative offset around the view-sphere. The possible view-shifts are determined by the physical location of the cameras relative to the target. To facilitate learning, the half-sphere and, hence, view-shifts, are discretized into a fixed number of azimuth and elevation offsets. Let c_m and c_n be the discretized location of two cameras (possibly the same camera) in the network, represented in (cyclic) azimuth and elevation. The view-shift between cameras is $\mathbf{v} = c_n - c_m$. The view-shift $\langle 0, 0\rangle$ represents maintaining the current view, while $\langle +1, -1\rangle$ represents shifting to a camera one offset around (azimuth) and one offset up (elevation) the view-sphere. Figure 4 shows an example configuration where the view-sphere has been divided into two elevation and ten azimuth bins with 5 cameras indicated as circles within the corresponding bin. For $C = 5$ cameras, there are at most 20 $\big(C \times (C-1)\big)$ possible view-shifts. The actual number of potential view-shifts is often lower as there are shared view-shifts. For example, in Fig. 4, the view-shift $\langle +1, 0\rangle$ describes both the shift from camera 2 to camera 1 and camera 4 to camera 3. Because the view-shifts are relative, and not tied to absolute camera locations, the approach is not specific to any particular camera network configuration.

View-shift learning can be framed as a Markov Decision Process (MDP) where the goal is to maximize the expected reward (correct classification) by taking actions (view-shifting) given an observed keypose. Decisions are drawn from the available view-shifts (including maintaining the current view).

Reinforcement Learning (RL) provides an efficient approach to solving MDP problems. We apply the Q-learning algorithm [32] to learn the state-action value function, $Q(x, \mathbf{v})$, which encodes the value of applying view-shift, $\mathbf{v}$, after observing keypose x. The Q-learning algorithm iteratively approximates $Q(x, \mathbf{v})$ over a series of episodes. For an observed keypose in the sequence, Q can be updated:

$$\begin{aligned} Q(x,\mathbf{v}) \leftarrow{}& (1-\eta)Q(x,\mathbf{v}) \\ &+\eta\left[r(x',\mathbf{v},y^*) + \gamma \max_{\mathbf{v}'} Q(x',\mathbf{v}')\right] \end{aligned} \tag{2}$$

where η is the learning rate, r is the immediate reward function, x' is the keypose observed after view-shift $\mathbf{v}$, and y^* is the correct label for the training sample. Reward allocation is based on whether the next keypose in the sequence is class-discriminative, with a small penalty for view-shifting. We allocate rewards as follows:

$$r(x,\mathbf{v},y^*) = \Psi(\mathbf{v}) + \begin{cases} +1 & \text{if } \operatorname*{argmax}_{y'} P(x \mid y') = y^* \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

where the view-shift penalty, $\Psi(\mathbf{v}) = -0.1$ if $\mathbf{v} \neq \langle 0, 0\rangle$. So, we provide a positive reward when the most likely class for the keypose, given by the distribution, $P(x \mid y')$, is the correct class, y^*, and a negative reward otherwise (i.e., the frame is misclassified). Figure 5 shows examples of two frames captured from different views and the class probability distributions, $P(x \mid y)$, for each frame with the correct class highlighted in green. Based on the scoring scheme, in both cases, the pose on the right (boxed in green) represents a positive reward and that on the left (boxed in red) represents a negative reward.

The training process continues until the values for $Q(x, \mathbf{v})$ converge. Pseudocode for the training phase is provided in Algorithm 1.

3.3 Recognition

For recognition, we assume that each target is currently associated with a primary camera, so the initial viewpoint might be any camera in the system. For each observed frame in a sequence, features are calculated and quantized to the nearest keypose. Similarly to [29], for a sequence of keyposes, $S = \{s_t\}$, we compute the posterior probabilities, $P(y \mid S)$, using a frame-wise approach with the Naive Bayes and uniform priors assumptions:

Algorithm 1: View-shift value learning

```
Input: Synchronized image sequences, $\mathcal{I}$; frame labels, $\{y_i\}$;
       number of keyposes, $K$; discretized camera locations,
       $\{c_m\}$
Output: State-value function, $Q(x, \mathbf{v})$; keypose dictionary, $\mathcal{X}$
Learn keypose dictionary, $\mathcal{X} \leftarrow \{x_1, \ldots, x_K\}$;
For each $x_k$, calculate $P(x_k \mid y)$ (Eq. 1);
Initialize $Q(x, \mathbf{v})$ with random values drawn from (0, 1);
while not converged do
    Randomly select time, $i$, and camera, $j$;
    Get keypose, $x_k$, for frame, $I_{j,i}$;
    foreach camera location, $c_m$ do
        Compute view-shift, $\mathbf{v} = c_j - c_m$;
        Compute reward, $r(x_k, \mathbf{v}, y_i)$ (Eq. 3);
        Update $Q(x_k, \mathbf{v})$ (Eq. 2)
    end
end
```

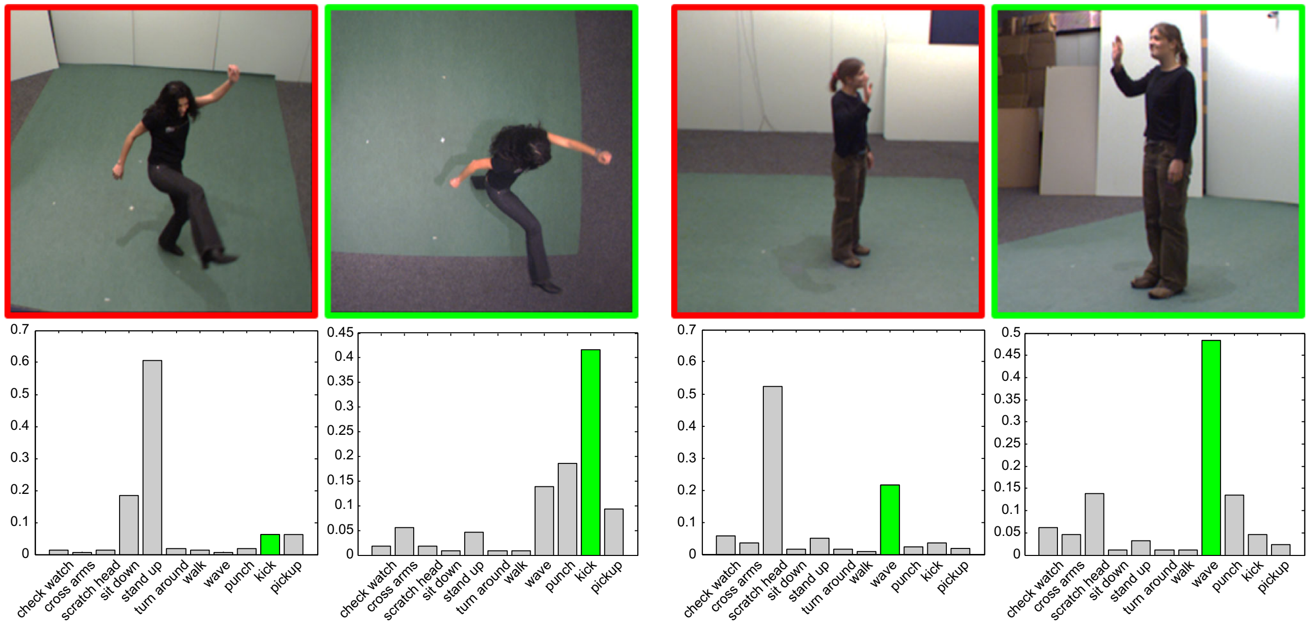

Fig. 5 Examples of kicking (*left*) and waving (*right*) frames captured from two different views and the class probability distributions for each frame with the correct class highlighted in *green*. In our reinforcement learning method, the frames boxed in *green* would trigger a positive reward and the *red-boxed* frames would trigger a negative reward

$$P(y \mid S) \propto \sum_t \log P(s_t \mid y). \tag{4}$$

For recognition, the final prediction for sequence, S, is the class label, $\hat{y}$, that maximizes the posterior probability:

$$\hat{y} = \underset{y'}{\operatorname{argmax}} \sum_t \log P(s_t \mid y') \tag{5}$$

It is worth nothing that, in our framework, a "sequence" consists of a single keypose at a time, however, different views may be used as a result of view-shifting. So, during the recognition phase, the keypose both serves to provide evidence for the prediction and also drives dynamic selection of a potential view-shift using the learned Q-table values for the viewpoints available during recognition. The selected view-shift determines which camera will be used to view the next frame in the sequence. The details of the algorithm are given in Algorithm 2. The final prediction can be based on either time- or confidence-based thresholds.

Algorithm 2: Dynamic view-shift recognition

```
Input: Synchronized image sequences, $\mathcal{I}$; keyposes, $X$;
       state-value function, $Q(x, \mathbf{v})$; discretized camera
       locations, $\{c_m\}$
Output: Predicted label, $\hat{y}$
Let $a \equiv$ index of active camera;
Let current time, $t = 1$;
while not yet classified do
    Get keypose, $x_k$, for frame, $I_{a,t}$;
    Update classification posterior (Eq. 5);
    Perform (potential) view-shift: $a \leftarrow \operatorname{argmax}_j Q(x_k, c_j - c_a)$;
    $t \leftarrow t + 1$;
end
```

4 Results

In this section, we describe several different action recognition experiments performed to evaluate our framework. All of the methods were implemented in Matlab on a standard PC. We evaluated our approach on two widely used multi-view human action recognition datasets: i3DPost [7] and INRIA Xmas Motion Acquisition Sequences (IXMAS) [35]. i3DPost consists of 8 people each performing 10 actions: walk, run, jump forward, bend, hand wave, jump in place, sit–stand, run–fall, walk–sit, and run–jump–walk. IXMAS contains 10 actors performing 11 actions (check watch, cross arms, scratch head, sit down, stand up, turn around, walk, wave, punch, kick, and pick-up) 3 times each. Example frames from the datasets are shown in Fig. 6.

4.1 Implementation details

Our approach can be applied to any frame-based feature descriptor; these experiments use the Motion Context descriptor [29], which represents the distribution of occu-

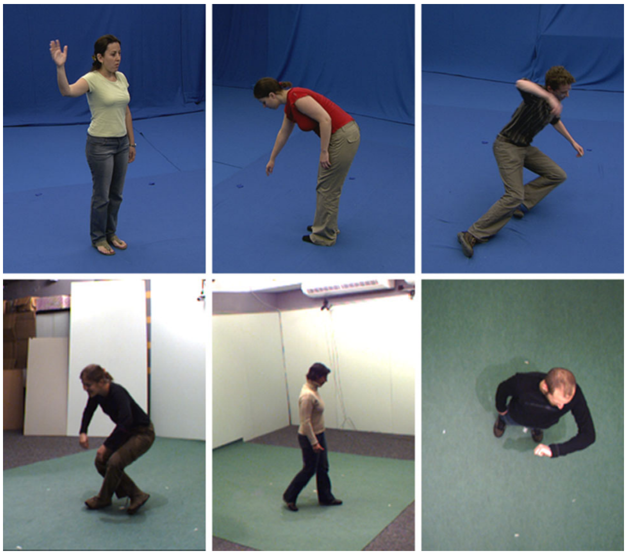

Fig. 6 Example frames from the i3DPost (*top*) and IXMAS (*bottom*) data sets

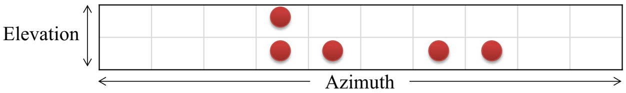

Fig. 7 For the 5-camera IXMAS dataset, this diagram shows the location of each of the cameras (*red circles*) on the discretized view half-sphere

pancy and $x-$ and $y-$components of optic flow in a bounding box surrounding the object of interest combined with a low-dimensional projection of the feature vectors for neighboring frames.

4.1.1 Camera configuration

For view-shift learning, the view half-sphere is divided into discrete bins covering vertical (elevation) and horizontal (azimuth) shifts. For i3DPost, the 8 cameras are situated at the same elevation at 45° azimuth intervals. For this configuration, we discretize the view-sphere into 8 bins for azimuth. IXMAS consists of five cameras. Four of the cameras are at similar elevations, and one is nearly overhead. For this configuration, we discretize the view-sphere into two bins for elevation and 10 for azimuth, as shown in Fig. 7. This results in 30 possible view-shifts: 10 for clockwise shifts in azimuth, and 3 for elevation $\{+1, 0, -1\}$.

4.1.2 Keypose Learning

In Sect. 3, we described how a keypose dictionary is used in both training and recognition. There are a number of approaches for dictionary learning in this context; we evaluated three: k-means (KM), Submodular Dictionary Learning (SDL) [12], and per-class k-means (PCKM). KM is the most basic approach to unsupervised dictionary learning (e.g., [6]). SDL is a recently developed supervised dictionary learning method that optimizes cluster compactness, element similarity, and class discriminativeness. PCKM applies standard (unsupervised) k-means to each class separately and merges all the discovered cluster centers for the final learned dictionary. Finding key poses separately per-class has been similarly considered in other recent work [3,4]. Our method is not specific to any particular method. Other clustering approaches, such as support vector clustering [1] or the information bottleneck method [27], could also be incorporated.

For each method, K, the number of clusters (keyposes), is a free parameter. For PCKM, where clustering is performed separately for each class, K refers to the number of total clusters in the final, combined set for fair comparison with the other methods. To determine the effect of varying K on the clustering results, we used normalized mutual information (NMI), which is a measure of how homogeneous the clusters are with respect to the class label of the examples, balanced against the number of clusters [18]. Figure 8 shows the NMI for increasing values of K for the three dictionary learning methods for both datasets. For all values of K, PCKM results in more homogeneous clusters than KM or SDL. Based on the assumption that more homogeneous clusters lead to more class-discriminative keyposes, we select PCKM for dictionary learning for the remaining experiments. Using the elbow method, we choose $K = 1000$ for i3DPost and $K = 1650$ for IXMAS.

To build the dictionary of keyposes, the feature descriptors for every fourth frame in the training data were clustered via the PCKM algorithm. We initialized PCKM 5 times and selected the cluster assignment with minimum energy, as measured by average intra-class similarity. Figure 9 shows example frames from IXMAS from two clusters. The first row depicts a cluster that is mainly homogeneous; most of the frames are examples of waving. The second row shows the more common case of a pose shared among multiple classes.

4.1.3 Q-learning parameters

For Q-learning, the free parameters were determined empirically. The discount factor, γ, is 0.5, and the learning rate, η, is initially 1.0 with a decay rate of 0.997. In addition to the reward allocation scheme represented in Eq. 3, we evaluated other schemes, including rewards proportional to the keypose classification margin or based on logistic regression; in general, the selected scheme performed at least as well as these alternatives. Training terminates when Q-table values have converged (i.e., successive entries are updated by less than 10^{-5}). With these values, the training phase in our experi-

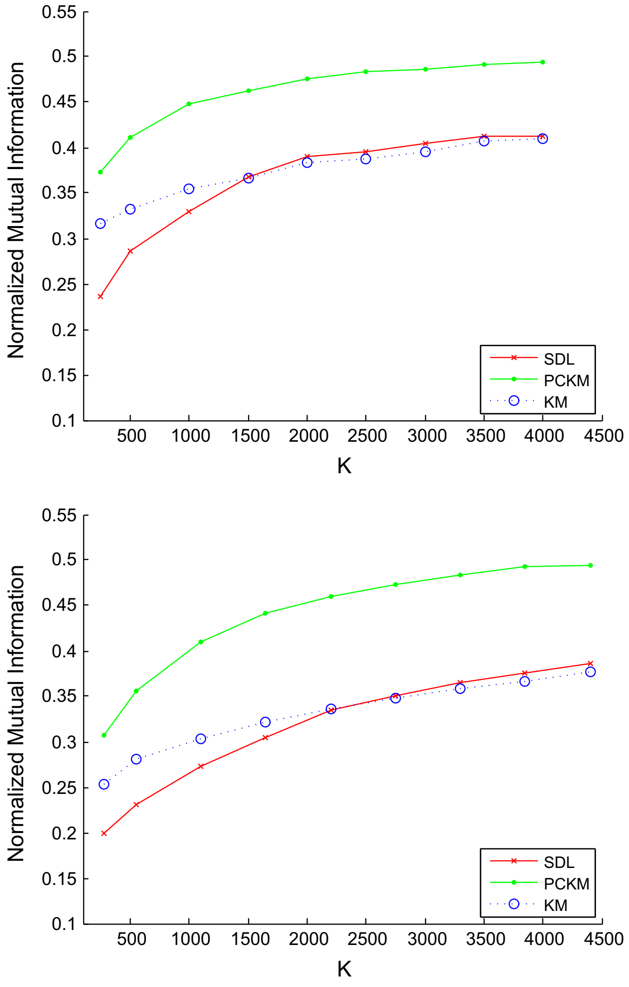

Fig. 8 Normalized mutual information for clustering on i3DPost (*top*) and IXMAS (*bottom*) with three different methods over a range of k values

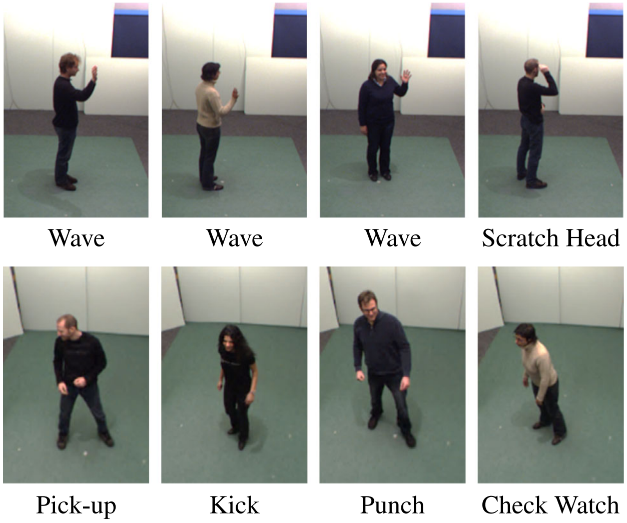

Fig. 9 Each row shows frames co-clustered after unsupervised dictionary learning (*Top*). The cluster is mostly homogeneous; most of the frames are examples of waving (*Bottom*). The cluster is heterogeneous. From *left* to *right*, the actions are pick-up, kick, punch, and check watch

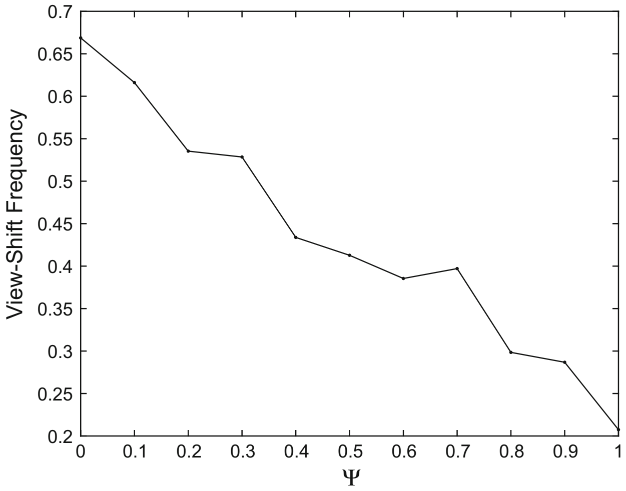

Fig. 10 The frequency of view-shifts as a function of the view-shift penalty, Ψ

ments typically requires between 5000 and 10,000 iterations to converge, which takes about 30 sec on a standard PC.

The view-shift penalty, Ψ, represents the cost for switching to a different camera in the network. Figure 10 shows the number of view-shifts per frame as a function of Ψ for a system trained and tested on the IXMAS data, averaged over 10 trials. (Other data showed a similar trend.) Increasing Ψ decreases the frequency of view-shifts. In a deployed system with a measurable physical or computation cost for view-shifting, this parameter can be tuned to the desired setting. For our experiments, we set the view-shift penalty, $\Psi = 0.1$, reflecting relatively little penalty to switching views.

4.2 Experiments

We apply our method to three different action recognition experiments, including complete sequences, early recognition, and different training and testing environment camera configurations. We refer to our view-shift action recognition method as `vs` and compare to alternative frame-based multi-view classification schemes that use the same features and aggregation methods:

- The single-camera (`sc`) method is our implementation of an algorithm described in [29], which uses Naive Bayes classification on single-view action sequences without view-shifting.
- The multi-camera voting (`vote`) method applies a common multi-view aggregation technique where the majority decision serves as the final classification.

For each experiment, we followed the leave-one-actor-out (LOAO) cross-validation experimental protocol, which is most commonly used in the literature for both i3DPost and IXMAS (e.g., [8,9,16,35,43]). For LOAO, all action sequences for a particular actor are used for testing, while the remaining sequences are used for training. Accuracy is averaged over all permutations.

4.2.1 Trimmed video sequences

For classification on video sequences of prescribed length, we compared our method, `vs`, to `sc` and `vote`, as well as other recent multi-view recognition methods on both datasets. Table 1 shows the overall accuracy of each of the methods. In general, our method, `vs`, outperformed competing approaches. For each dataset, our results are better than previously reported results from related multi-view methods. To the best of our knowledge, the only methods that outperform our approach on these data sets (e.g., 99.22 % for i3DPost [11], 98.78 % for IXMAS [30]) are 4D approaches that either explicitly build a 3D model from the multiple views or rely on an expensive model parameter fitting step. Compared to both the multi-view and 4D approaches, ours is more efficient since, per target, processing occurs on only a single-view per time step.

Figure 11 shows the confusion matrices for our method, `vs`, for this classification experiment on each data set. Each row represents the actual class and each column represents the predicted class. For many actions, accuracy is 100 %. For i3DPost, the greatest confusion is between run and jump (forward), while for IXMAS the most challenging case involves confusion between waving and scratching head. These results are reasonable as the confusion in each case is between actions which share similar poses. Figure 12 shows representative examples of view-shifts for the IXMAS experiment. For each example, the opaque images correspond to active cameras and reflect the input processed by our method. The figure shows the expected value of a view-shift to each camera in the network (including not shifting) and the graphs in the rightmost column show the class probability distribution associated with the pose observed in the frame. The first two examples are representative of the general case, where the learned Q-value leads to a future class-discriminative pose. The third example shows a case where the view-shift leads to an incorrectly classified pose.

The overall accuracy on the i3DPost dataset is generally higher than on IXMAS, not only for our method, but for those reported in the literature. This may be explained by the difference in the complexity of the actions between the datasets. Compared with i3DPost, IXMAS contains far more self-occlusions and similar-looking actions with subtle variations, such as between scratching head and waving. In addition, the position and orientation (relative to the cameras) of the IXMAS actors are not prescribed, so the relative pose of an actor is not a function of which camera is recording the action sequence. The same complexity difference is reflected

Table 1 Multi-view classification rates on the i3DPost and IXMAS data sets. The method type refers to the number of simultaneous views

Method	type	i3DPost (%)	IXMAS (%)
`vs`	Single	**97.65**	**94.24**
`vote`	Multi	96.25	93.33
`sc`	Single	93.91	86.06
Iosifidis et al. [11]	4D	99.22 (8 actions)	–
Iosifidis et al. [10]	Multi	95.50 (8 actions)	–
Gkalelis et al. [7]	Multi	90.00 (5 actions)	–
Holte et al. [9]	Multi	80.00 (10 actions)	–
Turaga et al. [30]	4D	–	98.8
Liu et al. [16]	Multi	–	93.7
Wu et al. [38]	Multi	–	88.2
Zhu et al. [43]	Multi	–	88.0

(Top) rates for our approach, `vs`, and variants, (Bottom) representative multi-view recognition rates reported in the literature

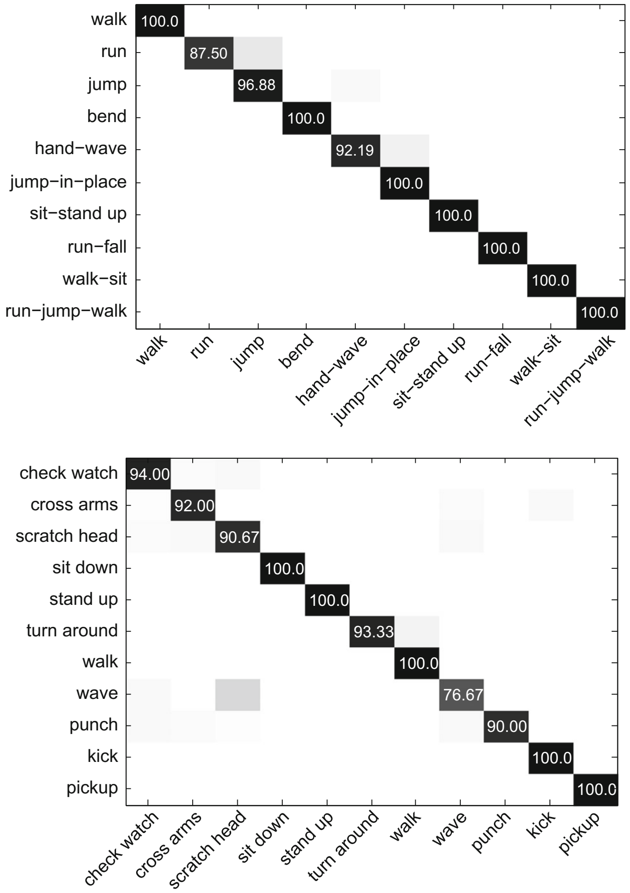

Fig. 11 Confusion matrices for the proposed view-shift method on the i3DPost (*top*) and IXMAS (*bottom*) datasets

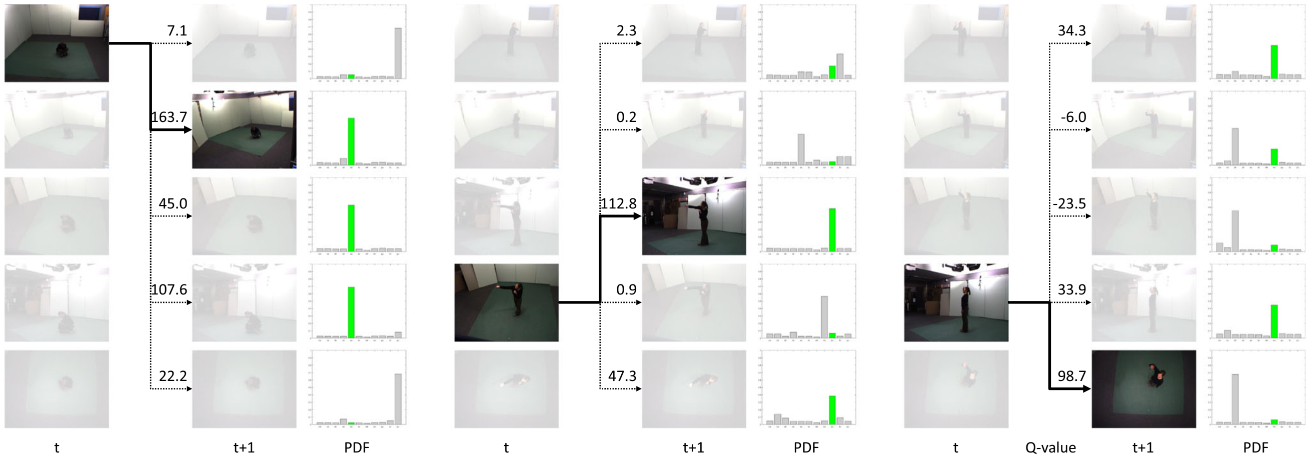

Fig. 12 Example view-shifts for stand up, punch, and wave, respectively. For the active camera (opaque image), the assigned keypose is associated with Q-values for each available view-shift. The highest scoring view-shift determines the active camera for the next frame. The *graphs* in the rightmost column show the class probability distribution associated with the pose observed

in lower NMI scores for IXMAS than i3DPost (Fig. 8). For the remaining experiments, we focus on the more challenging IXMAS dataset.

4.2.2 Early recognition

Previous work (e.g., [22]) has included the observation that many action sequences can be identified with snippets of only a few frames. For this experiment, using the IXMAS dataset, we classified only the first portion of an action sequence for various lengths from 10 to 100 %. Figure 13 shows the results for our method and the representative single- and multi-view methods. As expected, the classification accuracy of all methods increases as a greater portion of each sequence is observed. As with the experiment with the trimmed sequences, our method achieves comparable performance to the method that incorporates all views simultaneously. For sequences comprising just the first 40 % of the available frames, the view-shift method accuracy is 85 %, which is competitive with recent methods that observe the entire sequence and utilize all the cameras simultaneously (as reported in Table 1).

4.2.3 Camera network configuration

To evaluate the performance of our method when different camera configurations are used in training and testing, we perform experiments using various combinations of the network cameras. Table 2 shows results for an experiment where four of five cameras were used in training, and all five cameras were used for testing. That is, the test environment contains a camera view unseen during training. Single-camera recognition (sc) accuracy drops by an average of

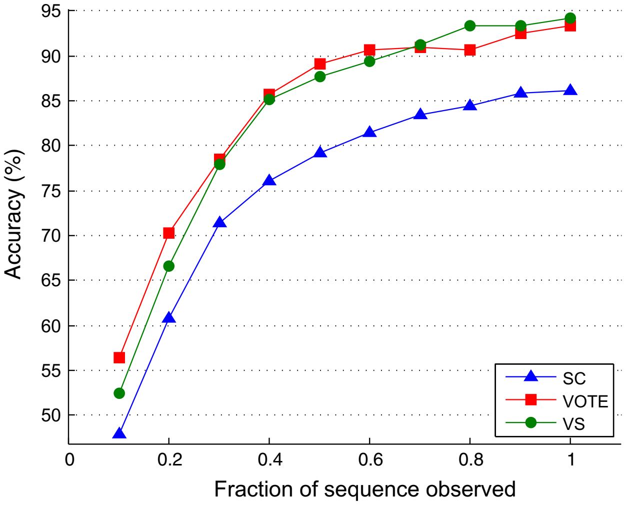

Fig. 13 Accuracy of action recognition as a function of the observed fraction of each sequence using the IXMAS dataset. As the fraction increases, the accuracy of all methods improves. Accuracy of our view-shift method, vs, is similar to vote, despite using only a single camera per frame

12 %, while, with the view-shift method, vs, recognition accuracy does not significantly drop. As before, our method achieves similar performance to the more computationally expensive approach, vote, which uses multiple views simultaneously. For IXMAS, the overhead viewpoint, camera 5, captures much different representations of the actions compared to the other four cameras. So, in the case when this view is unavailable for training, but present in testing, it is effectively ignored. Compared to the other combinations of training viewpoints, in this case where the overhead camera is excluded, we observe the lowest overall classification accuracy. However, the effect is limited since the overhead camera

Table 2 Classification accuracy on the IXMAS dataset with various camera combinations

Training views	`sc` (%)	`vote` (%)	`vs` (%)
2 3 4 5	77.70	91.52	92.00
1 3 4 5	74.79	91.82	92.12
1 2 4 5	76.06	90.61	91.70
1 2 3 5	75.52	92.12	91.94
1 2 3 4	74.36	90.61	91.09
Average	75.69	91.34	91.77
Testing views	`sc`	`vote`	`vs`
4 cameras	86.12	91.52	91.93
3 cameras	86.06	90.91	90.66
2 cameras	85.99	84.30	90.33

(Top) The left column shows which cameras were used for training, while all views were used for testing. (Bottom) The left column shows how many cameras were used for testing while all views were used for training; these results are averaged over all the 4-, 3-, and 2-camera combinations, respectively

is, in general, the least useful for distinguishing among different actions.

In addition, we perform an experiment using fewer cameras in testing compared to training, so that fewer view-shift options are available during testing. We train the method using all five cameras, and perform recognition using a subset of these views. Averaging the results over all the permutations of cameras, our method, `vs`, achieves 91.93, 90.66, 90.33 % for four, three, and two cameras, respectively. As the number of cameras in the test environment decreases, the accuracy of our method decreases only slightly, while the accuracy of `vote` decreases sharply when only two cameras are available for testing. While these experiments may not replicate disparate camera networks, they provide evidence that relative view-shifts are not tied to specific camera configurations.

5 Conclusions and future Work

In this paper, we presented a new approach to multi-camera action recognition, which makes use of a single active camera at a time, resulting in computational efficiency, while achieving results equal to or slightly better than methods that incorporate multiple views simultaneously. Our method is applicable to a wide variety of image features and distributed camera network architectures.

For the future, we plan to investigate discriminative keypose learning approaches designed for view-shifting. Our current approach is based on the keypose dictionaries initially learned. In our experiments with different dictionary learning methods, we observed significant variation in cluster homogeneity based on the learning method. Homogeneity directly relates to how class-discriminative a particular keypose will be. We plan to investigate supervised keypose learning, and supervised dictionary learning more broadly, to find more discriminative poses. Decreasing the total number of keyposes should lead to further improvements in both accuracy and speed. In addition, we plan to investigate whether the types of poses that are discriminative for recognition are different from poses that are discriminative for triggering view-shifts.

References

1. Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik, V.: Support vector clustering. J. Mach. Learn. Res. **2**, 125–137 (2002)
2. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. IEEE Trans. Pattern Anal. Mach. Intell. **23**(3), 257–267 (2001)
3. Chaaraoui, A.A., Climent-Pérez, P., Flórez-Revuelta, F.: Silhouette-based human action recognition using sequences of key poses. Pattern Recogn. Lett. **34**(15), 1799–1807 (2013)
4. Cheema, S., Eweiwi, A., Thurau, C., Bauckhage, C.: Action recognition by learning discriminative key poses. In: IEEE International Conference on Computer Vision Workshops, pp. 1302–1309 (2011)
5. Cilla, R., Patricio, M.A., Berlanga, A., Molina, J.M.: Fusion of single view soft k-nn classifiers for multicamera human action recognition. In: Hybrid Artificial Intelligence Systems, pp. 436–443. Springer (2010)
6. Farhadi, A., Tabrizi, M., Endres, I., Forsyth, D.: A latent model of discriminative aspect. In: IEEE International Conference on Computer Vision, pp. 948–955 (2009)
7. Gkalelis, N., Kim, H., Hilton, A., Nikolaidis, N., Pitas, I.: The i3dpost multi-view and 3d human action/interaction database. In: Visual Media Production, 2009. CVMP'09. Conference for, pp. 159–168. IEEE (2009)
8. Holte, M.B., Chakraborty, B., Gonzalez, J., Moeslund, T.B.: A local 3-d motion descriptor for multi-view human action recognition from 4-d spatio-temporal interest points. IEEE J. Sel. Top. Signal Process. **6**(5), 553–565 (2012)
9. Holte, M.B., Moeslund, T.B., Nikolaidis, N., Pitas, I.: 3d human action recognition for multi-view camera systems. In: 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on, pp. 342–349. IEEE (2011)
10. Iosifidis, A., Tefas, A., Pitas, I.: Multi-view human action recognition under occlusion based on fuzzy distances and neural networks. In: Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European, pp. 1129–1133. IEEE (2012)
11. Iosifidis, A., Tefas, A., Pitas, I.: View-independent human action recognition based on multi-view action images and discriminant learning. In: IVMSP Workshop, 2013 IEEE 11th, pp. 1–4 (2013)
12. Jiang, Z., Zhang, G., Davis, L.S.: Submodular dictionary learning for sparse coding. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pp. 3418–3425. IEEE (2012)
13. Kläser, A., Marszałek, M., Schmid, C.: A spatio-temporal descriptor based on 3d-gradients. In: British Machine Vision Conference, pp. 995–1004 (2008)
14. Laptev, I.: On space-time interest points. Int. J. Comput. Vision **64**(2–3), 107–123 (2005)
15. Liu, J., Shah, M., Kuipers, B., Savarese, S.: Cross-view action recognition via view knowledge transfer. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 3209–3216 (2011)

16. Liu, L., Shao, L., Rockett, P.: Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition. Pattern Recogn. **46**(7), 1810–1818 (2013)
17. Määttä, T., Härmä, A., Aghajan, H.: On efficient use of multiview data for activity recognition. In: Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras. ICDSC ’10, pp. 158–165. ACM, New York, NY, USA (2010)
18. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval, vol. 1. Cambridge university press, Cambridge (2008)
19. Parrigan, K., Souvenir, R.: Aggregating low-level features for human action recognition. In: Advances in Visual Computing, Lecture Notes in Computer Science, pp. 143–152 (2010)
20. Poppe, R.: A survey on vision-based human action recognition. Image Vision Comput. **28**(6), 976–990 (2010)
21. Rudoy, D., Zelnik-Manor, L.: Viewpoint selection for human actions. Int. J. Comput. Vision **97**(3), 243–254 (2012)
22. Schindler, K., Van Gool, L.: Action snippets: how many frames does human action recognition require? In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
23. Shen, C., Zhang, C., Fels, S.: A multi-camera surveillance system that estimates quality-of-view measurement. In: Image Processing, 2007. ICIP 2007. IEEE International Conference on, vol. 3, pp. III–193. IEEE (2007)
24. Souvenir, R., Babbs, J.: Learning the viewpoint manifold for action recognition. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–7 (2008)
25. Spurlock, S., Souvenir, R.: Multi-view action recognition one camera at a time. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2014)
26. Srivastava, G., Iwaki, H., Park, J., Kak, A.C.: Distributed and lightweight multi-camera human activity classification. In: Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on, pp. 1–8. IEEE (2009)
27. Tishby, N., Slonim, N.: Data clustering by markovian relaxation and the information bottleneck method. In: Advances in Neural Information Processing Systems, pp. 640–646 (2000)
28. Tran, D., Sorokin, A.: Human activity recognition with metric learning. In: Proceedings of the 10th European Conference on Computer Vision: Part I, pp. 548–561. Springer-Verlag (2008)
29. Tran, D., Sorokin, A.: Human activity recognition with metric learning. In: European Conference on Computer Vision, pp. 548–561 (2008)
30. Turaga, P., Veeraraghavan, A., Chellappa, R.: Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
31. Wang, X.: Intelligent multi-camera video surveillance: a review. Pattern Recogn. Lett. **26**, 1–25 (2015)
32. Watkins, C.J., Dayan, P.: Q-learning. Mach. Learn. **8**(3–4), 279–292 (1992)
33. Weinland, D., Boyer, E., Ronfard, R.: Action recognition from arbitrary views using 3d exemplars. In: Proceedings of International Conference on Computer Vision, pp. 1–7 (2007)
34. Weinland, D., Özuysal, M., Fua, P.: Making action recognition robust to occlusions and viewpoint changes. In: Computer Vision–ECCV 2010, pp. 635–648. Springer (2010)
35. Weinland, D., Ronfard, R., Boyer, E.: Free viewpoint action recognition using motion history volumes. Comput. Vision Image Underst. **104**(2), 249–257 (2006)
36. Weinland, D., Ronfard, R., Boyer, E.: A survey of vision-based methods for action representation, segmentation and recognition. Comput. Vision Image Underst. **115**(2), 224–241 (2011)
37. Wu, C., Khalili, A.H., Aghajan, H.: Multiview activity recognition in smart homes with spatio-temporal features. In: Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras, pp. 142–149. ACM (2010)
38. Wu, X., Xu, D., Duan, L., Luo, J.: Action recognition using context and appearance distribution features. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 489–496 (2011)
39. Yan, P., Khan, S.M., Shah, M.: Learning 4d action feature models for arbitrary view action recognition. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pp. 1–7. IEEE (2008)
40. Zhao, Z., Elgammal, A.M.: Information theoretic key frame selection for action recognition. In: Proceedings of the British Machine Vision Conference, pp. 1–10 (2008)
41. Zheng, J., Jiang, Z.: Learning view-invariant sparse representations for cross-view action recognition. In: Proceedings of International Conference on Computer Vision, pp. 3176–3183. IEEE (2013)
42. Zheng, J., Jiang, Z., Phillips, P.J., Chellappa, R.: Cross-view action recognition via a transferable dictionary pair. In: Proceedings of the British Machine Vision Conference, p. 7 (2012)
43. Zhu, F., Shao, L., Lin, M.: Multi-view action recognition using local similarity random forests and sensor fusion. Pattern Recogn. Lett. **33**, 438–445 (2012)

Scott Spurlock received his Ph.D. from the University of North Carolina at Charlotte and his B.S. from the University of Florida. He is an assistant professor in the Computing Sciences Department at Elon University. His research focuses on human motion analysis in camera networks.

Richard Souvenir received his B.S., M.S., and D.Sc. degrees from Washington University in St. Louis in 2001, 2003, and 2006, respectively. He is an Associate Professor in the Department of Computer Science at The University of North Carolina at Charlotte where he is affiliated with the Video and Image Analysis Lab, Charlotte Visualization Center, and Center for Biomedical Engineering and Science. His main research interests are computer vision and machine learning with application to human activity understanding and biomedical video analysis.