J. Sens. Actuator Netw. **2013**, *2*, 486-508; doi:10.3390/jsan2030486

OPEN ACCESS

Journal of Sensor and Actuator Networks

ISSN 2224-2708

www.mdpi.com/journal/jsan

Article

Real-Time Recognition of Action Sequences Using a Distributed Video Sensor Network

Rahul Kavi and Vinod Kulathumani *

Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV 26506, USA; E-Mail: rkavi@mix.wvu.edu

* Author to whom correspondence should be addressed; E-Mail: vinod.kulathumani@mail.wvu.edu; Tel.: +1-304-293-9698.

Received: 22 April 2013; in revised form: 20 June 2013 / Accepted: 25 June 2013 / Published: 18 July 2013

Abstract: In this paper, we describe how information obtained from multiple views using a network of cameras can be effectively combined to yield a reliable and fast human activity recognition system. First, we present a score-based fusion technique for combining information from multiple cameras that can handle the arbitrary orientation of the subject with respect to the cameras and that does not rely on a symmetric deployment of the cameras. Second, we describe how longer, variable duration, inter-leaved action sequences can be recognized in real-time based on multi-camera data that is continuously streaming in. Our framework does not depend on any particular feature extraction technique, and as a result, the proposed system can easily be integrated on top of existing implementations for view-specific classifiers and feature descriptors. For implementation and testing of the proposed system, we have used computationally simple locality-specific motion information extracted from the spatio-temporal shape of a human silhouette as our feature descriptor. This lends itself to an efficient distributed implementation, while maintaining a high frame capture rate. We demonstrate the robustness of our algorithms by implementing them on a portable multi-camera, video sensor network testbed and evaluating system performance under different camera network configurations.

Keywords: multimedia sensor network; information fusion; camera network; multi-view; score-fusion; activity recognition

---

1. Introduction

Real-time identification of human activities in urban environments is increasingly becoming important in the context of public safety, assisted living and national security applications [1,2]. Distributed video sensor networks [3,4] that provide multiple views of a scene are ideally suited for real-time activity recognition. However, deployments of multi-camera-based real-time action recognition systems have thus far been inhibited because of several practical issues and restrictive assumptions that are typically made, such as the knowledge of a subjects orientation with respect to the cameras, computational overhead of distributed processing and the conformation of a network deployment during the testing phase to that of a training deployment. The focus of this paper is on designing a framework that allows for relaxing some of these restrictive assumptions and enables recognition of human actions based on data from multiple cameras (See Figure 1). Specifically, some of the challenges with respect to multi-camera activity recognition and our contributions in this paper towards addressing them are described below.

Figure 1. A frame of a subject performing a *kicking* action from six different views. Given visual data from a subset of such views, the objective of the proposed system is to recognize a sequence of actions being performed.

1.1. Contributions

- **Distributed processing for real-time efficiency:** In order to avoid overloading the network with too much data, it is important to ensure that individual frames are locally processed and only relevant data is sent to a fusion center for final classification [5]. At the same time, in the context of real-time recognition, it is equally important to keep the computational overhead low, so that data can be locally processed at a high enough frame rate. Lower frame rates of processing will lead to lower data sampling rates, and key motion information will be lost, resulting in lower classification accuracy. Therefore, there is a need to avoid computationally expensive approaches for local feature extraction.

  In this paper, we show how aggregated locality-specific motion information obtained from the spatio-temporal shape of a human silhouette, when used concurrently with information from multiple views using our fusion strategy, can yield good classification rates. Relying

on such computationally simple operations for local processing lends itself to an efficient distributed implementation.

- **Combining multi-view information:** When using information from multiple views for action recognition, the angle made by the subject with respect to a camera while performing an action is not known. Pose estimation of a human subject based on body posture itself is a hard problem, and it is, therefore, not practical to assume that information. View-invariant techniques, on the other hand, do not fully utilize the variations in multi-view information that is available for classification [6]. The question then arises as to how view-specific classifiers can be used without knowledge of subject orientation.

  To address this challenge, instead of feature-level fusion of multi-camera data, we use an output-level score-based fusion strategy to combine information from a multi-view camera network for recognizing human actions [7]. By doing so, we are able to use the knowledge of camera deployment at run-time and seamlessly fuse data without having to re-train classifiers and without compromising on accuracy. Moreover, we note that the cameras acquiring data may not be deployed in any symmetry. It is not feasible to assume that the entire 360 degree view for the action being performed is available; only data from some viewing directions may be available. It is also infeasible to assume that camera configuration stays the same between the training phase and the actual testing phase. We show how our fusion technique can seamlessly handle these cases. Also, as opposed to using only the *best-view* in classification [8], the proposed design utilizes information from all available views, yielding higher accuracy.

  The proposed score fusion technique is independent of the underlying view-specific classifier applied to generate scores from individual views. In fact, we evaluate the performance of our system using two different supervised-learning classifiers: Support Vector Machines and Linear Discriminant Analysis.

- **Variable length inter-leaved action sequences:** It is not sufficient to evaluate the performance of an action recognition system assuming that each action is of a fixed length and that each action occurs in isolation. In reality, human activity action recognition involves classification of continuously streaming data from multiple views, which consists of an interleaved sequence of various human actions. Single or multi-layer sequential approaches, such as hidden Markov models (HMMs) [9–11], dynamic Bayesian networks (DBNs) [12,13] or context-free grammars [14–17], have been designed to address this challenge and to recognize longer activities and activities involving multiple subjects. However, sequential approaches for activity recognition have mainly been studied only in the context of single views and not for the case of multi-view camera networks without knowledge of subject orientation. Doing the same in a multi-camera video sensor network setting is much more challenging, because data is continuously streaming in from multiple sources, which have to be parsed in real-time.

  In this paper, we describe how our multi-camera score fusion technique can be augmented to achieve real-time recognition of interleaved action sequences. We consider a human activity to be composed of individual *unit actions* that may each be of variable length and interleaved in

a non-deterministic order. Our fusion technique is then applied to streaming multi-view data to classify all unit actions in a given sequence.

- **Performance evaluation using a portable camera network:** We implement our activity recognition framework on an embedded, wireless video sensor network testbed assembled using Logitech 9000 cameras and an Intel Atom 230 processor-based computing platform. We use this system to first evaluate the performance of the score fusion strategy on individual unit actions and, subsequently, on interleaved action sequences. We systematically evaluate the system in the presence of arbitrary subject orientation with respect to the cameras and under failures of different subsets of cameras. We consider action sequences to be composed of an interleaved set of nine pre-trained actions along with some arbitrary actions for which the system is not pre-trained. We demonstrate that the system is also able to accurately recognize actions that belong to the class of trained actions, as well as reject actions that do not belong to the trained set. (This paper is a substantially extended version of our paper titled *Real-time activity recognition using a wireless camera network* that appeared at the IEEE Conference on Distributed Smart Cameras (ICDSC 2011). Specifically, we have developed an algorithm that builds on the fusion framework described in [7] to identify interleaved sequences of human actions. We have also evaluated the performance of this system using a portable wireless video sensor network assembled using off-the-shelf components.)

1.2. Related Work

Human activity recognition has been a widely researched topic with several practical civilian and defense applications. A nice summary of the research to date has been presented in [18–22]. The survey presented in [22] classifies existing techniques for activity recognition into single-layered ones that are suitable for classifying gestures and short duration actions and multi-layered ones [12,13,23,24] that are suitable for activities composed of smaller sub-events. Single layered techniques are further classified as space-time-based approaches and sequential approaches. Space-time classifiers [25–32] model actions based on spatio-temporal motion features extracted from a sequence of images and use the concatenated set of features directly to measure similarities and classify human actions. Sequential classifiers [9–11,33,34], on the other hand, use the transitional characteristics existing in a sequence of observed features to classify human actions and are more suitable when actions are performed with speed variations.

However, most of the above techniques are single node systems where observation, computation and classification are performed at the same node and utilize information only from a single view. Single node systems either require the subject to be in a specific pose with respect to the camera or use view-invariant techniques for classification [6], and do not utilize the three-dimensional information that can be used for improving system accuracy. In operational scenarios with a single camera, it is infeasible to make assumptions about the knowledge of a subject's orientation with respect to the camera. By utilizing data from multiple cameras, we have focused on eliminating this assumption. Moreover, issues related to distributed computing have not been the focus of existing studies. Questions, such as how to balance node-level computation with central computation to achieve fast and accurate recognition

and what features are more suited for classification in multi-camera systems, have not been addressed in the existing literature. This paper seeks to address this gap by jointly considering distributed computing issues in the design of human activity classifiers.

Noting the advantages of a multi-camera system and fueled by advances in camera and networking technology, several multi-camera based approaches have been proposed in recent times for action recognition [5,35–38]. However, multi-camera systems have typically made restrictive assumptions in order to be able to fuse data from different cameras. For example, the view combination method presented in [8] and [37] combines feature vectors from individual cameras before performing the classification, and this imposes a requirement on the configuration of cameras to be identical between the test and training phases. In contrast, we combine inputs from different views at a score-level by exploiting the relative orientation between cameras, and as a result, we do not require the camera configuration to be preserved between training and test phases. Furthermore, as opposed to the best view classifier presented in [8], our technique uses data from all available views for classification, and we highlight the robustness of our approach by considering cases when the best view(s) are not available.

In many of the existing multi-camera techniques, the distributed feature extraction is computationally intensive. In [39,40], a sequence of 3D visual hulls generated from multiple views has been used for action recognition. In [36], a human body model is used to extract feature descriptors that describe the motion of individual body parts. However, in a real-time setting, computationally-intensive feature extraction is likely to reduce the frame processing rate, thereby losing key motion information and adversely affecting the accuracy of the system. By way of contrast, one of the contributions of our work is to show how computationally simple feature descriptors can be used in conjunction with information from multiple views to achieve high accuracy in a real-time implementation.

1.3. Outline of the Paper

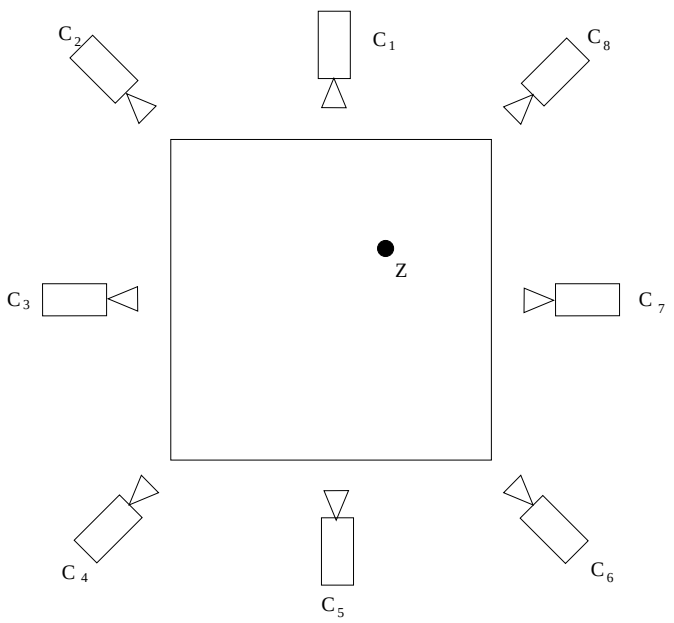
In Section 2, we describe our system model and problem statement. In Section 3, we describe our action recognition system. In Section 4, we evaluate the performance of our system. In Section 5, we present conclusions and state future work.

2. Model and Problem Statement

Our system consists of N_C cameras that provide completely overlapping coverage of a region, R, from different viewing directions. The relative orientations between the cameras are assumed to be known, but the cameras may not conform to the symmetric arrangement shown in Figure 2; and there may be fewer cameras (We systematically quantify performance with different subsets of cameras). An *activity* or an *action sequence* performed by a subject consists of an interleaved sequence of unit actions performed by the subject that are chosen from the following set of nine actions and performed in any (unknown) orientation with respect to the cameras: *clapping hands, waving one arm (left or right), waving two arms, punching, jogging in place, jumping in place, kicking, bending and underarm bowling.* We use $\{A\}$ to denote the set of these unit actions and use $A_a, (1 \leq a \leq N_a)$ to denote individual actions in the set, where N_a is the total number of actions in the set. We have chosen these actions based on some of the recent activity recognition datasets [37,41]. We have assumed that there is only one subject within the

region, R, at any given time. Actions may be of different durations. In between two consecutive actions from this set, there may or may not be a pause, where the subject does nothing (*i.e.*, simply stands still or performs random movements that do not belong in the set, $\{A\}$). The pauses may also be of arbitrary duration. The subject can be at an any location within region R, but this location is fixed for the duration of the action sequence. The objective of the system is to correctly recognize the individual unit actions being performed that belong to the training set and, also, to avoid misclassification of the pause (if any) as belonging to one of the actions in $\{A\}$.

Figure 2. Deployment of cameras in the system.



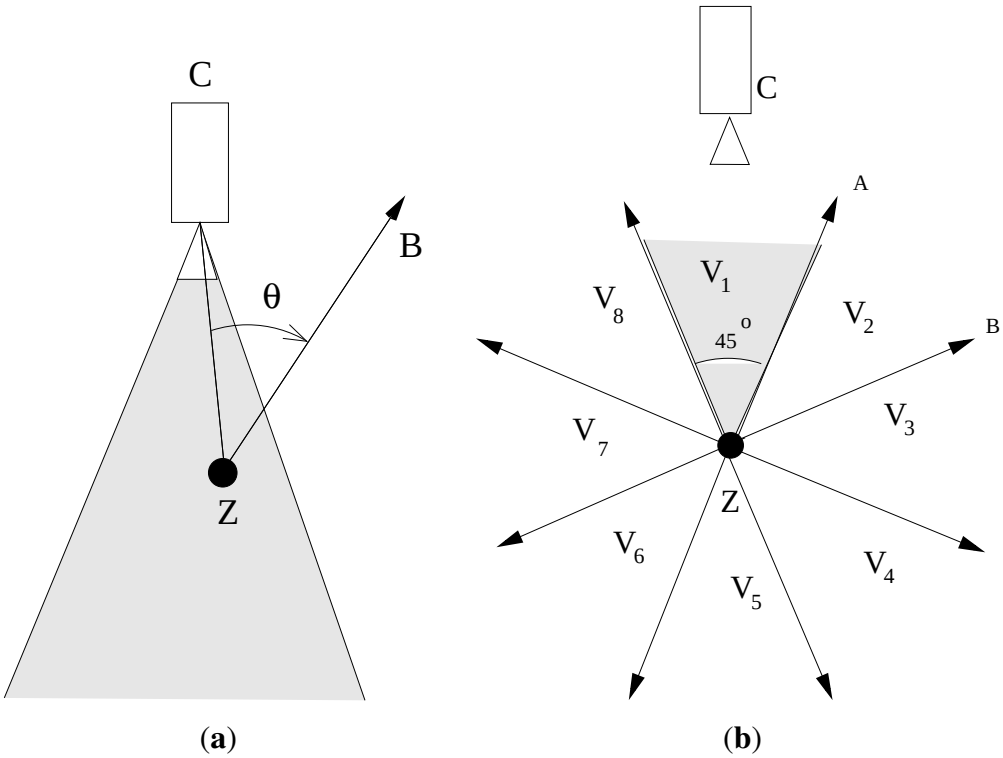
In our specific experimental setting, we use a network of up to eight cameras deployed over a square region of 50 feet by 50 feet. The cameras are denoted as C_i $(1 \leq i \leq 8)$ and are deployed along the square region at a height of eight feet from the ground. The subject can be at any location within this region, such that each camera is able to capture the complete image of the subject (Figure 2). We use the Logitech 9000 USB cameras for our experiments, with data sampled at approximately 15 fps and each frame captured at 960×720 resolution.

We define the *view-angle* of a camera with respect to an action being performed as the angle made by the optical axis of the camera with the direction along which the subject performs the action sequence. View-angle is measured in the clockwise direction from the ray originating at the subject location that is parallel to the optical axis of the camera (illustrated in Figure 3a). We divide the view-angle range of 0–360° into N_v different sets by considering that different instances of the same action captured with small view-angle separations are likely to appear similar. For our specific experimental setting, we consider $N_v = 8$, but we note that N_v can be chosen independent of the number of the cameras in the system. The eight view-angle sets are denoted as $V_j, (1 \leq j \leq 8)$ and are illustrated in Figure 3b for camera C. For example, in Figure 3b, when the subject is facing the region between rays, ZA and ZB, the camera, C, provides view, V_2.

From here on, we say that a camera, C_i, provides view, V_j, of an action being performed if the view-angle of C_i with respect to the action being performed belongs to set, V_j. At any given instant, it is

not necessary that data from all views, $V_j (1 \leq j \leq 8)$, are available. For instance, some cameras may not be active. It is also possible in certain deployments that the angle between the principal rays of adjacent cameras are small and, therefore, the cameras provide the same views of the action being performed.

Figure 3. View-angle of a camera and view-angle sets. (**a**) View angle of a camera, C; (**b**) view angle sets defined in the system.



3. System Description

In this section, we describe our action recognition system. We divide our presentation into four parts: (1) extraction of feature descriptors; (2) collection of training data; (3) score-fusion and unit action classification; and (4) real-time classification of action sequences.

3.1. Extraction of Local Feature Descriptors

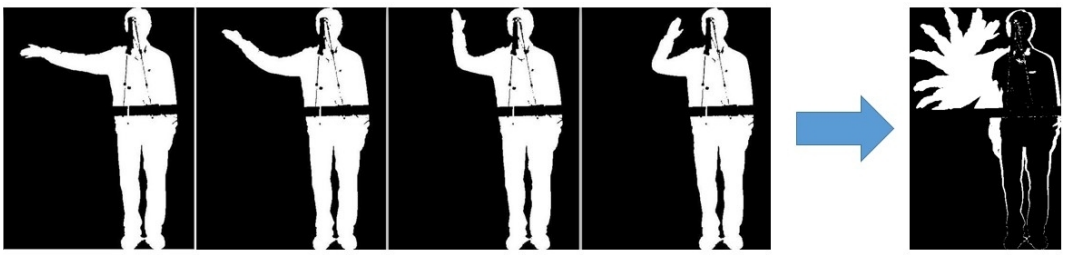
In order to be computationally simple, we use locally weighted motion energy images [42] as the feature vector for classifying actions in a given set of frames. We describe this procedure below. Consider a window of F consecutive frames acquired by a given camera. By subtracting the background from each frame, the silhouettes of the subject performing the action are extracted. A bounding box that envelopes all the background subtracted silhouettes is determined and drawn around each of the F extracted silhouettes. Let $p_i(t)$ denote the value of pixel i in frame t, where $1 \leq t \leq F$ and $p_i \in \{0, 1\}$.

Let E_F denote the motion energy image over the set of F frames. The value of pixel i in E_F is denoted as $E_F(i)$ and computed as follows:

$$E_F(i) = \sum_{x=2}^{x=F} (p_i(x) - p_i(x-1)) \tag{1}$$

Thus, the magnitude of each pixel in the motion energy image denotes the level of activity at the pixel. The motion energy image is then divided into a grid of 7×7. The sum of the pixel magnitudes in each cell of the grid is used as the feature vector for the set of F frames. By dividing the motion energy image into cells, the energy in each local region is captured. Thus, each feature vector obtained over a window of F consecutive frames is of length, 49. A series of binary image blobs extracted at a camera for the single arm waving action, along with the corresponding motion energy image is illustrated in Figure 4. The feature vectors are similar to the motion energy images described in [42], except that we also capture the spatial distribution of the motion energy to aid in activity recognition. In this paper, we refer to this as Localized Motion Energy Image (LMEI) feature descriptor.

Figure 4. A subset of the background subtracted blobs extracted by a camera are shown for the single arm waving action. The background subtracted binary blobs are used to generate the motion energy image, which shows motion concentrated near the top-left region. The motion energy is then represented as a 7-by-7 array feature vector that represents the spatial distribution of motion energy over the set of frames.



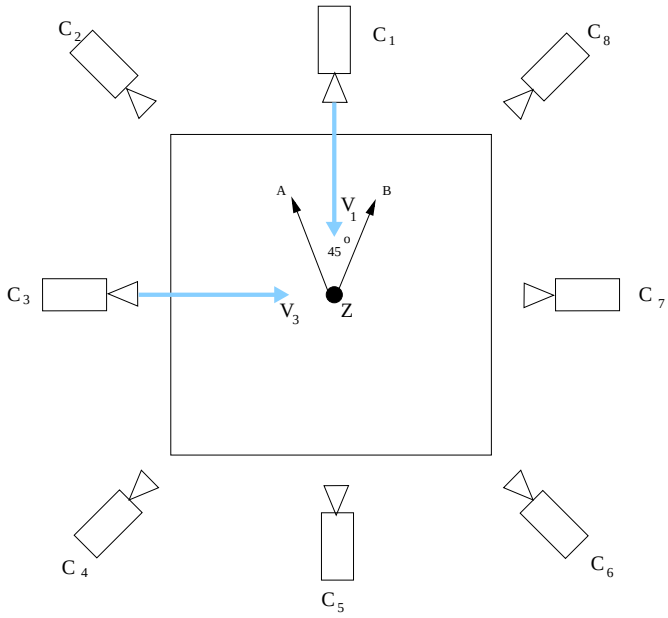
3.2. Collection of Training Data

In order to collect training data, videos of subject performing each unit action are collected with subject standing at the center of the square region and facing a reference camera (camera C_1 in Figure 5). The actions are performed at different viewangles, all of which belong to set V_1 with respect to the reference camera, C_1. Because of symmetry, data collected in camera C_i corresponds to view V_i $(1 \leq i \leq 8)$. Approximately 20 training samples are collected for each unit action in the set, A, performed by three different subjects. Note that the symmetry is introduced only for the ease of collecting training data from every view. During the testing and operational phases, the symmetry does not have to be maintained, and the subject does not have to be at the center.

The duration of each individual action may be different. For each unit action sample from each of the eight views, the motion energy-based feature vectors are computed as described in the previous sub-section. These feature vectors are given as input to supervised learning algorithms to design view-specific classifiers. Note that our score-based fusion technique is independent of the specific classifier that is used for generating view-specific scores. For instance, classifiers, such as Linear

Discriminant Analysis (LDA), Support Vector Machines (SVM), Logistic regression or Multinomial Naive Bayes, could be used for generating view-specific classification scores. For the localized motion energy feature descriptor, we have utilized two-class LDA as the supporting view-specific classifier. For LDA classification, we first obtain a discriminating projection vector, $\eta_{a,j}$, for action A_a and view V_j based on all training data collected for that particular action from the camera that provides view V_j during the training phase. During the fusion stage, the discriminating function is applied to the feature-vector at hand to obtain a score corresponding to that feature vector. The scores from multiple views are used for identifying actions without knowledge of subject orientation and without assuming network symmetry.

Figure 5. Collection of training data. The subject is at the center of the network and performs each training action while facing the region between rays, ZA and ZB. All cameras record the data. Because of symmetry, data collected in camera C_i corresponds to view V_i ($\forall i : 1 \leq i \leq 8$).



3.3. Score Fusion and Unit Action Classification

In this subsection, we describe how the score fusion is applied to identify a unit action that is performed in isolation, *i.e.*, the start and end times for the action is known. In the following subsection, we describe how this technique can be extended to recognize interleaved action sequences.

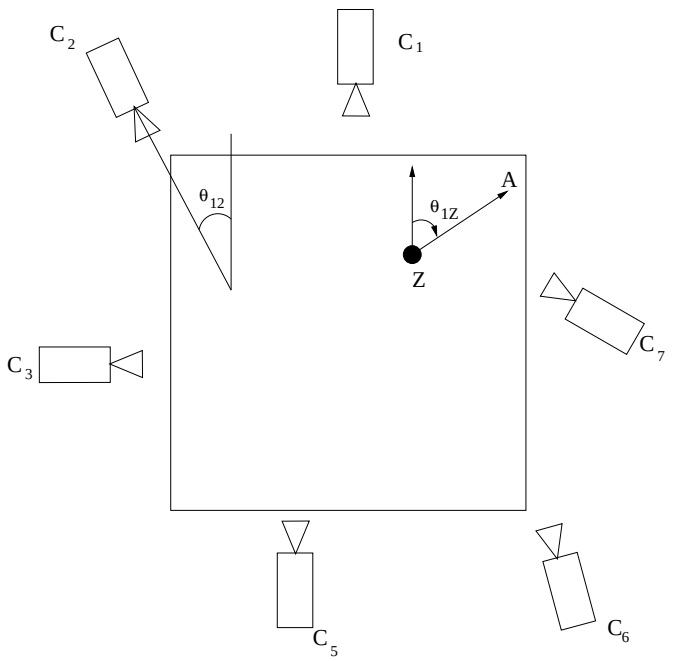
Consider that the subject performing a unit test action is at a point, Z, as shown in Figure 6. Let the view provided by camera C_{ref} with respect to the action being performed be V_j (In Figure 6, $ref = 1$). Note that V_j cannot be determined by C_{ref} (the subject orientation is unknown). However, the angles, $\theta_{r,s}$, between the principal axes of each pair of cameras (r, s) is known. Furthermore, using $\theta_{ref,s}$: $(1 \leq s \leq N_c)$, relative to each of the N_v possible views $V_j(1 \leq j \leq N_v)$ that camera C_{ref} can provide for the action being performed, the views provided by all other cameras can be computed. This

gives rise to a set, ϕ, of N_v possible configurations, which we denote as $\{\phi_k\}, 1 \leq k \leq N_v$. We let ϕ_k^i denote the view provided by camera C_i in configuration k.

$$\phi = \{\{\phi_1\}, .., \{\phi_{Nv}\}\} \quad (2)$$

$$= \{\{\phi_1^1, .., \phi_1^{Nc}\}, .., \{\phi_{Nv}^1, .., \phi_{Nv}^{Nc}\}\} \quad (3)$$

Figure 6. Determining configuration set for subject location, Z. Consider C_1 as the reference camera. The real orientation of the subject with respect to the reference camera is not known. However, the angles, $\theta_{r,s}$, between the principal axes of each pair of cameras (r, s) is known. Then, for each possible view, $V_j(1 \leq j \leq 8)$, that camera C_1 can provide for the action being performed, the views provided by other available cameras can be determined, resulting in N_v possible configurations.



Note that if the cameras retained the symmetric deployment during testing, the eight configurations would have followed a cyclical shift, resulting in:

$$\phi = \{\{V_1, V_2, .., V_8\}, \{V_2, V_3, ..., V_1\}, .., \{V_8, V_1, .., V_7\}\} \quad (4)$$

However, the relative orientations between cameras need not be symmetric, and two cameras, r and s, can provide the same view with respect to an action being performed if $\theta_{r,s}$ becomes very small. For illustration, suppose cameras C1 and C2 provide the same views with respect to a subject performing the action. In this case, the eight configurations would be:

$$\phi = \{\{V_1, V_1, V_3, .., V_8\}, \{V_2, V_2, V_4, ..., V_1\}, ..\} \quad (5)$$

Note that in the scenario where certain cameras are absent or if their views are completely occluded, N_c now reflects the number of cameras from which data is available and each set, ϕ_k, in ϕ contains a fewer number of elements.

Once the configuration set, ϕ, is determined, we use the feature descriptor generated from the test data at every camera to obtain matching scores under every configuration. This is done as follows. Consider score generation, $S_{a,k,i}$, with respect to action A_a for data acquired by camera C_i under configuration ϕ_k. Let $\phi_k^i = j$, *i.e.*, camera C_i is associated with view V_j under configuration ϕ_k. Let FV_i denote the feature vector computed for test data generated by camera C_i. In order to generate score $S_{a,k,i}$, we determine $\eta_{a,j}(FV_i)$, which is normalized to a range of $[0, 1]$, with zero indicating no match and one indicating a perfect match. The function, $\eta_{i,j}$, depends on the particular classifier being used as described in the previous subsection. For the case of the LDA classifier, the score is obtained by computing $FV_i \times \lambda_{a,j}$, calculating the distance of this product from the mean LDA score for action A_a under view V_j and then normalizing the result to a range of $[0, 1]$. If SVM is used as the underlying classifier, we can apply the weight vector, $\omega_{a,j}$, on FV_i and, then, use Platt Scaling [43] to obtain a probability level for belonging to that particular class.

Thus, for each action A_a, $S_{a,k,i}$ represents the likelihood that test data from camera i corresponds to action A_a in configuration ϕ_k. Similarly, a matching score is generated for each camera, C_i, in all configurations, $\{\phi_k\}, 1 \leq k \leq 8$. If certain cameras have failed or are occluded, as shown in Figure 6, then the matching scores corresponding to only the available cameras are computed under each configuration. For each action, A_a, the net matching score, $S_{a,k}$, which represents the likelihood that the test data from all cameras in the system corresponds to action A_a in configuration ϕ_k, is computed as follows:

$$S_{a,k} = \sum_{i=1}^{Nc} S_{a,k,i} \tag{6}$$

After the configuration specific scores are generated, we compute the likelihood that the test data corresponds to action A_a by determining the maximum of $S_{a,k}$ over all configurations in the set, ϕ. We denote this as S_a.

$$S_a = max(S_{a,k})_{k=1,\ldots,8} \tag{7}$$

The action, $A_F (1 \leq F \leq N_a)$, with the highest score is classified as the action corresponding to the test data, where F is determined as follows:

$$F = argmax(S_a)_{a=1,\ldots,N_a} \tag{8}$$

3.4. Real-Time Classification of Interleaved Sequences

Note that classification of a sequence of unit actions in a real-time scenario poses additional challenges. Firstly, the individual background subtracted frames from all cameras are continuously streaming in from multiple cameras, and it is not known exactly where an action starts and ends in these sets of frames. Secondly, the string of individual unit actions have to be correctly divided into multiple unit actions, without generating false positives (*i.e.*, classifying as an action when in reality there is a pause). Thirdly, the frame capture rates at different cameras may not be perfectly matched. As a result, the number of frames captured by different cameras over the same time duration may be different. To address these challenges, we apply a sliding window algorithm, which is described below. We

assume that the cameras are time synchronized and incoming frames are tagged with the corresponding timestamp. We use NTP (Network Time Protocol) for time synchronization across cameras.

To apply the sliding window technique, we consider window sizes ranging from w_{min} to w_{max} based on the training data generated for each unit action. For a given window size, we consider the frames obtained from each camera corresponding to that time window using the timestamps associated with each frame. Note that the number of frames captured by each camera in the same window may be different, due to mismatch in the frame capture and processing rates across cameras. However, the choice of our feature vector, which only considers the aggregate motion energy image over a given window, allows us to overcome this challenge. We apply the score fusion technique described in the previous section on this window to obtain the score, S_a, corresponding to each action in the set, A, and the action, A_F, with the highest score. A match is registered if the maximum score is greater than a pre-determined threshold (τ_F) for the corresponding action, A_F. The thresholds are determined based on a certain fraction of the average score obtained for each action in the training phase.

Algorithm 1: Sliding window algorithm for parsing interleaved action sequences.

```
D := Length of stream IS;
start := 0;
while start ≤ D do
    len := w_min;
    /* Explore window sizes                                    */
    while len ≤ w_max do
        FD := IS[start : start + w];
        FV := LMEI(FD);
        F, S_F := Classify(FV);
        if S_F ≥ τ_F then
            Accept(IS[start : start + len − 1]);
            start = start + len;
        else
            start = start + δ_w;
    end
    Reject(IS[start : start + γ − 1]);
    start := start + γ;
end
```

In Section 4, we analyze the false classification and correct classifications as a function of this threshold. If a match is not registered, we progressively increment the window size in steps of δ_w until a match is found or the maximum window size is reached. If the maximum window size is crossed without finding a match, the sliding window is moved by a unit, γ, and the set of frames that were moved over are classified as a non-action (*i.e.*, an action that does not belong to the training set). Note that there is a tradeoff involved in choosing the step for the window in terms of the required amount of computation and the potential accuracy. A step size of one would ensure that every window size is explored, but this may not be necessary. A large step size might decrease the required computation, but it is possible that excess frames may be

included that distort the motion energy signature of a given action. In our specific design, the parameter, δ_w, is set to three and the parameter, γ, is set to five.

An outline of the sliding window algorithm is presented below. Let IS denote the input stream of total length, D. The input stream, IS, contains multiple unit actions that are interleaved along with actions that do not belong to the training set, A. Let $IS[x : y]$ denote the set of frames between timestamp x and y. Let $LMEI$ be the localized motion energy image for a given set of frames in IS.

4. Performance Evaluation

We present the performance evaluation results in two parts: (1) evaluation with unit test actions and (2) evaluation with interleaved action sequences. Note that in these evaluations, the orientation and position of the subjects is chosen randomly (similar to Figure 6). By doing so, we also change the relative orientations between cameras from those in the training phase. Furthermore, we consider data from different subsets of cameras in the performance analysis. This serves to break the symmetry of the training phase.

4.1. Evaluation with Unit Test Actions

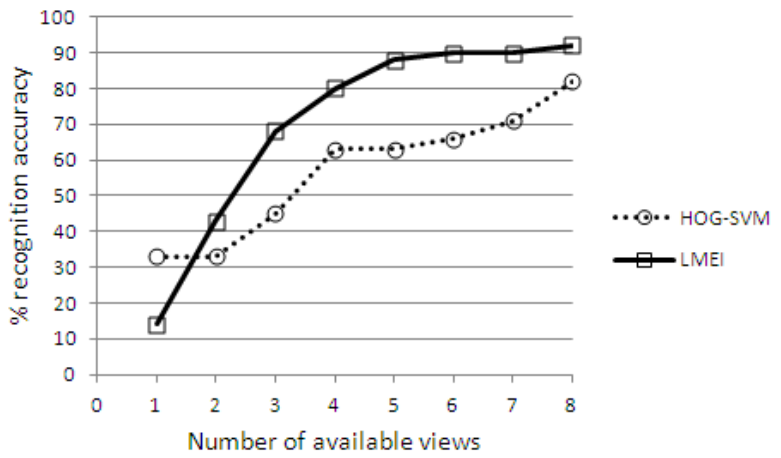
In this section, we evaluate the performance of our system with *unit test actions*. Here, the actions are not interleaved—each unit test action is provided separately to the classifier. This allows us to isolate the performance of the action classifier from that of the effects introduced by interleaving multiple actions in a sequence. We compute the overall recognition accuracy of the system with different number of available camera views using our multi-view fusion framework. The availability of camera views are ordered based on their suitability for recognizing a particular action: we order the favorable views for identifying each action based on the average motion energy observed in the training dataset for those actions from a given view.

We also note that the fusion framework presented in this paper does not depend on any specific feature extraction technique or classifier. We have utilized the localized motion energy based technique, because it is computationally simple and because computing the motion energy separately in each block allows us to capture the spatial distribution of the motion energy in the action. Other feature descriptors and view-specific classifiers can be used as a foundation to compute scores from individual cameras—our framework allows these scores to be combined and yield a classification result. Just as one specific alternate example, we also utilized another technique for feature extraction and classification based on computing the *Histogram of Oriented Gradients (HOG)* of a motion history image. HOGs are popular feature descriptors, widely used on still images and videos, that utilize the direction of edges in the underlying image for classification [44–47]. For action recognition, we compute the HOGs on the motion energy images over a given set of frames. The gradients are divided into eight discrete intervals that are 45 degrees apart, and a histogram is obtained for this distribution. The histograms are used as underlying feature descriptors in the classification. SVM with a linear kernel is used for classification.

In Figure 7, we show the recognition accuracy for the system using the localized motion energy feature descriptor (LMEI) that we have used in this paper and using the alternate HOG feature descriptor with an SVM classifier. As seen in both of the curves, when there is only a single view, the accuracy is

quite low, because the subject could be in an arbitrary orientation that is not favorable for classification. However, by combining data from different views, we are able to improve the accuracy. With all eight views, we can achieve an accuracy of 90% with the LMEI feature descriptors and 82% with the HOG feature descriptors. This also shows that our fusion framework can be used in conjunction with alternate feature descriptors and view-specific classifiers. While the trend of increasing accuracy with a greater number of views remains the same, we notice that with a given number of views, the LMEI is able to offer a better recognition accuracy. This is because the localized motion energy feature descriptors capture the spatial distribution of motion energy within a given set of frames, which aids in improving the recognition accuracy.

Figure 7. Recognition accuracy for the system with a different number of available camera views.



4.2. Evaluation with Interleaved Action Sequences

In this subsection, we describe the performance evaluation of our wireless, multi-camera activity recognition system with interleaved action sequences (start and end times are not known, and successive actions appear one after the other). Here, also, we have considered the impact of a different number of camera views missing. We first describe the performance metrics used in the evaluation of our system.

4.2.1. Performance Metrics

The input stream, IS, is modeled as a string of unit actions, which may or may not belong to the set of trained actions, $\{A\}$. As an example, an input stream may look like $IS = \{A_3, A_1, X_1, X_2, A_2, A_4\}$. Here, the actions, X_i, denote actions that are not in the set of trained actions, while the rest belong in the set, A. Each unit action may be of a different duration. Let $t_s(A_i)$ denote the starting timestamp of A_i in the sequence and $t_e(A_i)$ denote the ending timestamp of A_i in the sequence. Likewise, let $t_s(X_i)$ denote the starting timestamp of X_i in the sequence and $t_e(X_i)$ denote the ending timestamp of X_i in the sequence. Let $n(A)$ denote the number of unit actions that belong to A in the input stream, IS. Let OS denote the out stream after the input is processed by the score fusion and sliding window algorithms. The stream, OS, also consists of a sequence that contains classified actions and intervals when no matches

are found. Let $\{A_{os}(t_x, t_y)\}$ denote the set of recognized actions in OS between time t_x and t_y. Our goal in this evaluation is to compare the string of actions in IS with that of OS.

- For each action $A_j \in \{A\}$ in IS, if $A_j \in \{A_{os}(t_s(A_j), t_e(A_j))\}$, then we increment the number of true matches (TM) for IS.
- For each action, $A_j \in \{A\}$, in IS, if $A_P \in \{A_{os}(t_s(A_j), t_e(A_j))\}$, where $A_P \neq A_j$ and A_p are not neighboring actions of A_j in IS, then we increment the number of misclassifications for IS.
- For each action, $X_j \ni \{A\}$ in IS, if $A_P \in \{A_{os}(t_s(X_j), t_e(X_j))\}$, where $A_P \in \{A\}$ and A_p are not neighboring actions of X_j in IS, then we increment the number of false matches for IS.

In summary, a true match indicates that a unit action in the input stream is correctly matched to a unit action in output stream within the interval in which the input action occurs. A misclassification indicates that an input action is classified as some other action in the output stream. A false match indicates that an action that does not belong to the set, $\{A\}$, is matched to an action in the set, $\{A\}$, in the output stream. The reason we have considered the action detected to be a *true* match if bordering with the actual action is the following. We are considering an interleaved action scenario, where the start and end times are not known. Therefore, a given window could contain *frames* from both the actions, and an action could be classified as a neighboring one. In an operational context, we believe that this is acceptable, because by examining the sequence of actions that are detected by the system, a higher level classifier will still be able to successfully stitch together the correct input sequence—provided the correct action was also identified somewhere within the time frame. Our focus here is not on identifying the exact start and end times for each action. Instead, we are more interested in ensuring that the sequence of actions being performed is correctly identified—the start and end may not perfectly coincide. Note that if the correct action is not detected at all within the time-frame of the input action, *true matches* are not incremented, thus reflecting as a false negative in the performance evaluation.

4.2.2. Classification of Action Sequences

Using the above metrics, we evaluated the performance of our system to recognize and parse sequences of actions. Specifically, we considered 12 sequence of actions performed by two subjects. The combined duration of the test sequences is approximately 17 minutes. Each sequence comprised of actions randomly chosen from the set, $\{A\}$, along with pauses and actions involving random movements that are not part of the set. The performance is characterized in the following figures.

In Figure 8, we show the variations in false matches and true matches as a function of the threshold (τ_F) in the sliding window algorithm that is used to register a match for an action in the input stream. The results are shown separately for the two feature descriptors and classifiers that are used for generating view-specific scores (LMEI-LDA and HOG-SVM). LMEI-LDA refers to the localized motion energy feature descriptors supported by an LDA classifier to generate view-specific scores. HOG-SVM refers to the Histogram of Oriented Gradients feature descriptor supported by an SVM classifier to generate view-specific scores. These results are with data from all eight cameras available for classification. The x axis in Figure 8a,b indicates what the fraction of the average matching score in the training phase is used as a threshold in the online classification phase. We observe that lower values of the

threshold result in higher false matches. However, by appropriately selecting the threshold, we are able to achieve high true match rates, while preserving low false match rates. When fusion is applied with the LMEI-based classifier, a true match rate of approximately 90% can be achieved at a false match rate of 20%. For the HOG-SVM-based system, a true match rate of approximately 70% can be achieved at a false match rate of 20%. In Figures 9–12, we show the variations in false matches and true matches, when two, four, six and seven favorable views are removed from the system (the results have been shown for both the LMEI and HOG-SVM classifier). The results are summarized in Figure 13 by showing the true match rate and misclassification percentage *at a fixed false match rate of* 20% (false rejects are added to the misclassifications). The results in Figure 13 can be used to understand the impact of the number of views and also understand the difference in performance based on the two feature descriptors. These observations are summarized below.

Figure 8. True/false matches *vs.* threshold with all views intact for Localized Motion Energy Image (LMEI)-Linear Discriminant Analysis (LDA)-based classifier and Histogram of Oriented Gradients (HOG)-Support Vector Machines (SVM)-based classifier. (**a**) LMEI-LDA classifier. (**b**) HOG-SVM classifier.

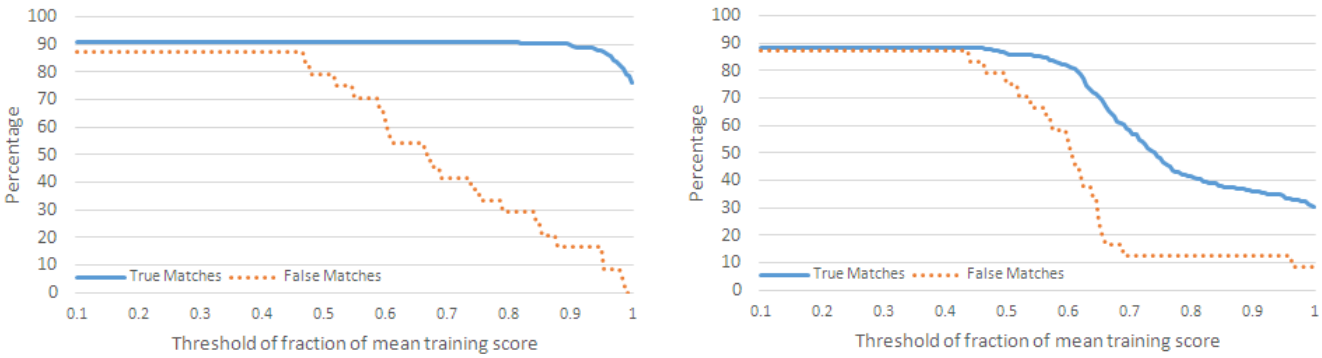


Figure 9. True/false matches *vs.* threshold with two views removed for LMEI-LDA-based classifier and HOG-SVM-based classifier. (**a**) LMEI-LDA classifier. (**b**) HOG-SVM classifier.

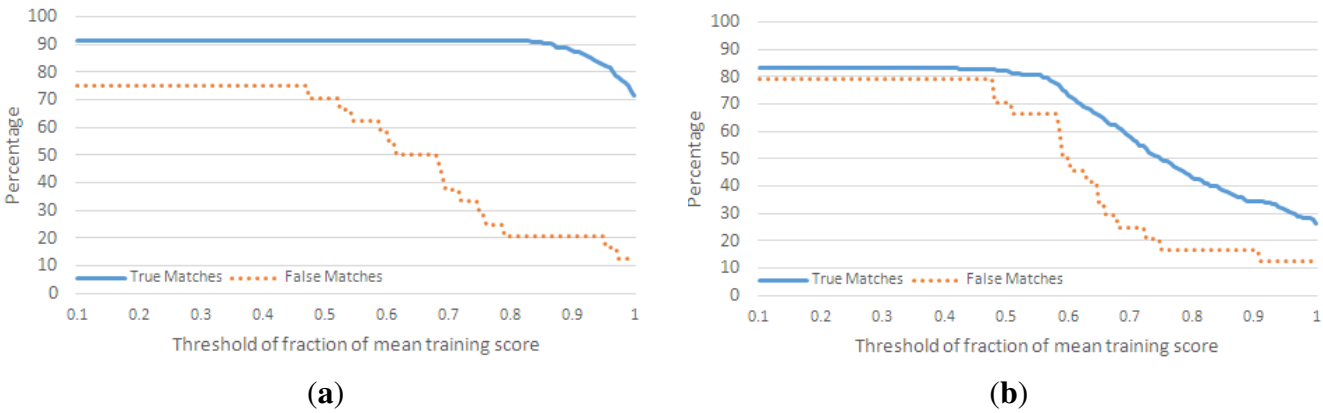
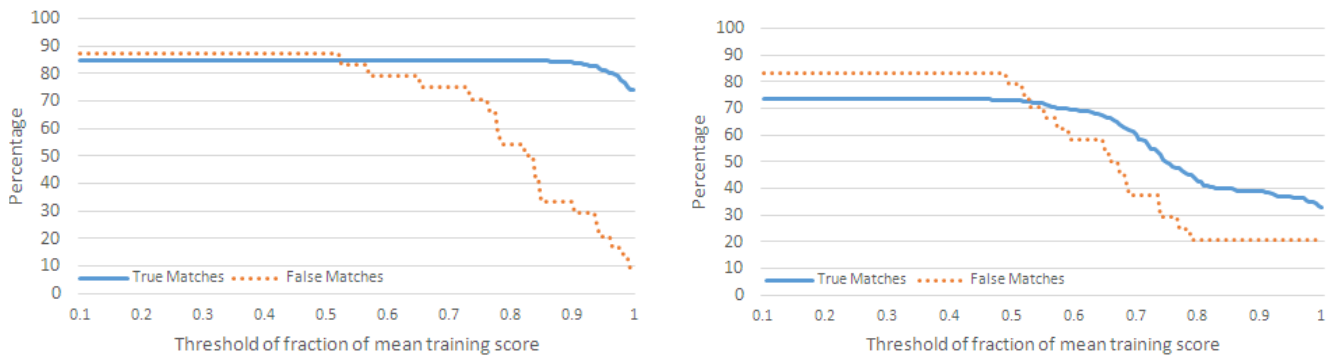
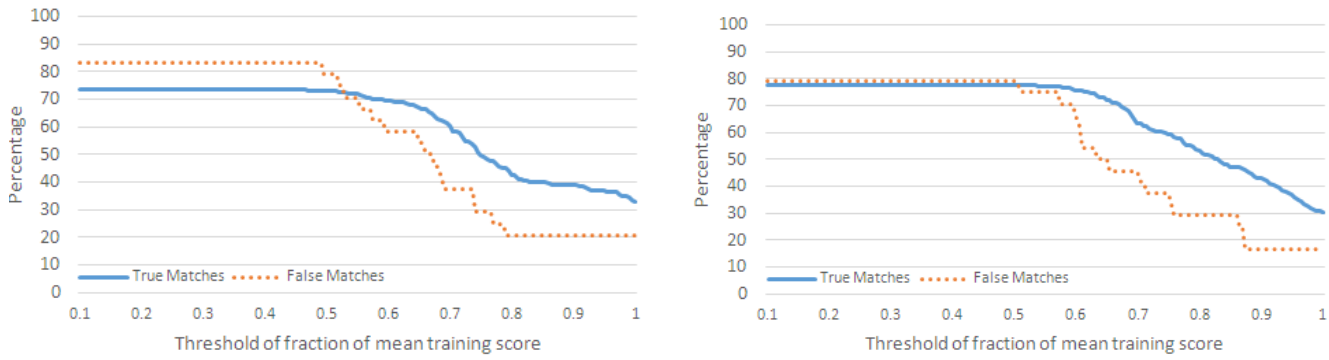


Figure 10. True/false matches *vs*. threshold with four views removed for LMEI-LDA-based classifier and HOG-SVM-based classifier. (**a**) LMEI-LDA classifier. (**b**) HOG-SVM classifier.



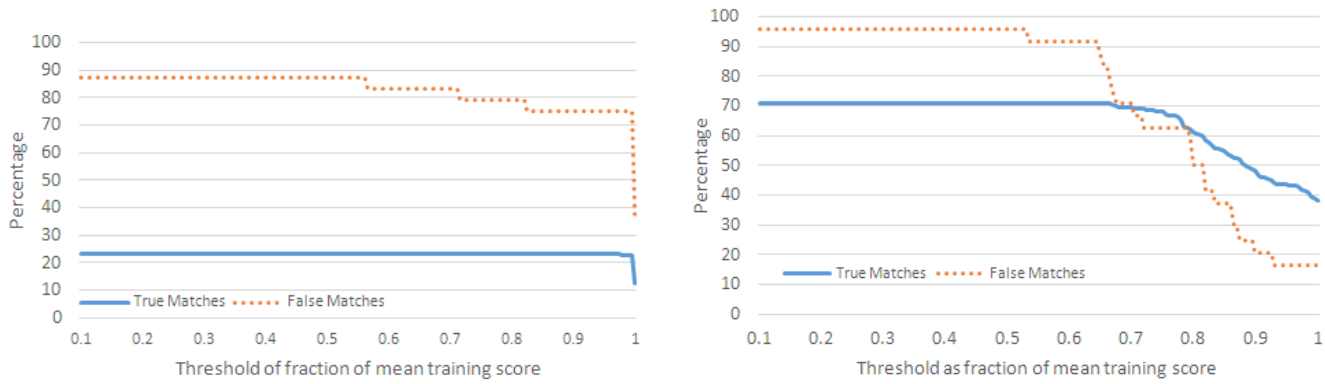
(**a**) (**b**)

Figure 11. True/false matches *vs*. threshold with six views removed for LMEI-LDA-based classifier and HOG-SVM-based classifier. (**a**) LMEI-LDA classifier. (**b**) HOG-SVM classifier.



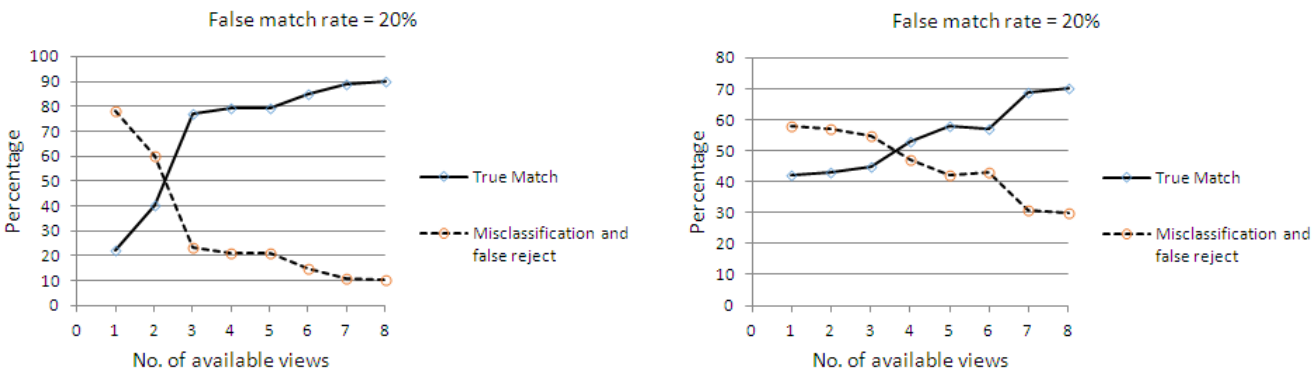
(**a**) (**b**)

Figure 12. True/false matches *vs*. threshold with seven views removed for LMEI-LDA-based classifier and HOG-SVM-based classifier. (**a**) LMEI-LDA classifier. (**b**) HOG-SVM classifier.



(**a**) (**b**)

Figure 13. True match rate and misclassification rate at false match rate of 20% for LMEI-LDA-based classifier and HOG-SVM-based classifier. (**a**) LMEI-LDA classifier. (**b**) HOG-SVM classifier.



(**a**) (**b**)

Firstly, the trend of improving accuracy with more available views remains the same, when the score fusion framework is applied with either feature descriptors. This highlights the generality of the score fusion framework proposed in this paper for multi-view action recognition. Secondly, we note that in comparison with the localized motion energy image descriptors that we have used, the HOGs yield poorer accuracy. Our explanation for this is that the localized motion energy image descriptors that we have used capture the spatial distribution of the motion energy over the duration of the action, thus aiding in system accuracy. Thirdly, we note that with the LMEI feature descriptor, even with five views removed, the system still has an accuracy of approximately 80% at a corresponding false match rate of 20%. However, when more than five favorable views are removed, the performance of the system starts decreasing rapidly. This fact also highlights the impact of multi-view fusion by showing that multiple views indeed add significant value in accurate classification.

5. Conclusions and Future Work

We have described a score-based fusion strategy to combine information from a multi-view camera network for recognizing human actions and, then, applied it towards recognizing actions in an interleaved sequence with varying durations by applying a sliding window algorithm. By systematically collecting training data from different views for an action and combining data from cameras at a score-level, we are able to accommodate arbitrary camera orientations during the testing phase. The reason for exploring the use of a multi-view camera setup is that in operational scenarios with a single camera, it is infeasible to make assumptions about the knowledge of a subject's orientation with respect to the camera. By utilizing multiple cameras, we have focused on eliminating this assumption. The subject could be in any orientation with respect to the cameras. Each camera computes *scores* corresponding to all trained views, and the inputs from all cameras are used to determine the most likely action. Furthermore, camera configuration can be different in the testing/operational phase compared to the training phase.

We tested the performance of our system using data collected from a camera network with up to eight cameras and showed that our system is able to achieve a true-match rate of 90%, while retaining

a low false-match and misclassification rate. This aspect was also tested with missing camera views on multiple view-specific classifiers. We highlighted that our system can tolerate the non-availability of data from cameras that provide the *best* views for classifying a given action.

An important application that we envision for this system is the surveillance of critical regions, such as alleys, entrances/exits to secure places or a focused area in a stadium/gathering. The cameras are required to have a common field of view, but they need not be physically close to the area being monitored—they could be zoomed in to the area being monitored. Another potential application is situational awareness or scene understanding in mobile environments; for example, law enforcement officers and military personnel requiring to set up monitoring of a remote area using a portable set of cameras.

We have applied our fusion technique on view-specific classifiers designed over computationally simple feature descriptors, which allows for deployments using embedded cameras that have low processing power. Individual cameras only need to compute the aggregate motion energy image over different window sizes and the resulting feature vectors. Since these are computationally non-intensive, each camera can process them at high frame acquisition rates. Furthermore, by sending only the aggregate data over different window sizes, the amount of data being transmitted reduces significantly. That being said, the fusion framework that we have presented does not depend on any specific feature descriptor or classifier. Other feature descriptors and view-specific classifiers can be used as a foundation to compute scores from individual cameras—our framework allows these scores to be combined and yield a classification result. As a case in point, we provided results that incorporate Localized Motion Energy Images (LMEI) and Histogram of Oriented Gradients (HOG) as the underlying feature descriptors in our score fusion framework.

In order to handle arbitrary orientation of a subject with respect to the cameras and to handle asymmetric deployment of cameras, our fusion approach relies on, first, systematically collecting training data from all view-angle sets and, then, using the knowledge of relative camera orientation during the fusion stage. Hence, we collected our own dataset for experiments in this paper (the training dataset for unit actions and the test dataset comprising action sequences is available at [48]).

In this paper, we have ignored network effects, such as transmission delays, data corruption and data dropouts, and their impact on real-time classification of events. Incorporating these into our fusion framework is one of the topics of our current research. In future work, we would also like to evaluate achievable performance with data collected in different settings and a greater number of unit action classes.

Conflict of Interest

The authors declare no conflict of interest.

References

1. Micheloni, C.; Remagnino, P.; Eng, H.; Geng, J. Introduction to intelligent monitoring of complex environments. *IEEE Intell. Syst.* **2010**, *25*, 12–14.

2. Fatima, I.; Fahim, M.; Lee, Y.; Lee, S. A unified framework for activity recognition-based behavior analysis and action prediction in smart homes. *Sensors* **2013**, *13*, 2682–2699.
3. Pham, C. Coverage and activity management of wireless video sensor networks for surveillance applications. *Int. J. Sens. Netw.* **2012**, *11*, 148–165.
4. Akyildiz, I.; Melodia, T.; Chowdhury, K. A survey on wireless multimedia sensor networks. *Comput. Netw.* **2007**, *51*, 921–960.
5. Wu, C.; Aghajan, H.; Kleihorst, R. Real-Time Human Posture Reconstruction in Wireless Smart Camera Networks. In Proceedings of 7th International Conference on Information Processing in Sensor Networks (IPSN), St. Louis, MO, USA, 22–24 April 2008; pp. 321–331.
6. Ji, X.; Liu, H. Advances in view-invariant human motion analysis: A review. *IEEE. Trans. Syst. Man. Cybern. C* **2010**, *40*, 13–24.
7. Ramagiri, S.; Kavi, R.; Kulathumani, V. Real-Time Multi-View Human Action Recognition Using a Wireless Camera Network. In Proceedings of 2011 Fifth ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), Ghent, Belgium, 22–25 August 2011; pp. 1–6.
8. Wu. C.; Khalili, A.; Aghajan, H. Multiview Activity Recognition in Smart Homes with Spatio-Temporal Features. In Proceedings of International Conference on Distributed Smart Cameras (ICDSC), Atlanta, GA, USA, 31 August–4 September 2010; pp. 142–149.
9. Yamato, J.; Ohya, J.; Ishii, K. Recognizing Human Action in Time-Sequential Images Using Hidden Markov Model. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Champaign, IL, USA, 15–18 June 1992; pp. 379–385.
10. Natarajan, P.; Nevatia, R. Coupled Hidden Semi Markov Models for Activity Recognition. In Proceedings of IEEE Workshop on Motion and Video Computing, Austin, TX, USA, 23–24 February 2007; pp. 1–10.
11. Hongeng, S.; Nevatia, R. Large-Scale Event Detection Using Semi-Hidden Markov Models. In Proceedings of International Conference on Computer Vision, Nice, France, 13–16 October 2003; Volume 2, pp. 1455–1462.
12. Buxton, H.; Gong, S. Visual surveillance in a dynamic and uncertain world. *Artif. Intell.* **1995**, *78*, 431–459.
13. Remagnino, P.; Tan, T.; Baker, K.D. Agent Orientated Annotation in Model Based Visual Surveillance. In Proceedings of International Conference on Computer Vision, Bombay, India, 4–7 January 1998; pp. 857–862.
14. Joo, S.; Chellappa, R. Recognition of Multi-Object Events Using Attribute Grammars. In Proceedings of IEEE International Conference on Image Processing, Atlanta, GA, USA, 8–11 October 2006; pp. 2897–2900.
15. Ryoo, M.S.; Aggarwal, J.K. Recognition of Composite Human Activities through Context-Free Grammar Based Representation. In Proceedings of 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; pp. 1709–1718.
16. Moore, D.J.; Essa, I.A. Recognizing Multitasked Activities from Video Using Stochastic Context-Free Grammar. In Proceedings of AAAI/IAAI, Edmonton, AB, Canada, 28 July–1 August 2002; pp. 770–776.

17. Ivanov, Y.A.; Bobick, A.F. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. Patt. Anal. Mach. Int.* **2000**, *22*, 852–872.
18. Turaga, P.; Chellappa, R.; Subrahmanian, V.S.; Udrea, O. Machine recognition of human activities: A survey. *IEEE Trans. Circ. Syst. Video. T.* **2008**, *18*, 1473–1488.
19. Aggarwal, J.K.; Cai, Q. Human motion analysis: A review. *Comput. Vis. Image Understand.* **1999**, *73*, 428–440.
20. Kruger, V.; Kragic, D.; Ude, A.; Geib, C. The meaning of action: A review on action recognition and mapping. *Adv. Robot.* **2007**, *21*, 1473–1501.
21. Xu, X.; Tang, J.; Zhang, X.; Liu, X.; Zhang, H.; Qiu, Y. Exploring techniques for vision based human activity recognition: Methods, systems, and evaluation. *Sensors* **2013**, *13*, 1635–1650.
22. Aggarwal, J.K.; Ryoo, M.S. Human activity analysis: A review. *ACM Comput. Surv.* **2011**, *43*, 16:1–16:43.
23. Park, S.; Aggarwal, J.K. A hierarchical Bayesian network for event recognition of human actions and interactions. *Multimed. Syst.* **2004**, *10*, 164–179.
24. Oliver, N.M.; Rosario, B.; Pentland, A. A Bayesian computer vision system for modeling human interactions. *IEEE Trans. Patt. Anal. Mach. Int.* **2000**, *22*, 831 –843.
25. Laptev, I.; Lindeberg, T. Space-Time Interest Points. In Proceedings of IEEE International Conference on Computer Vision (ICCV), Nice, France, 13–16 October 2003; pp. 432–439.
26. Niebles, J.C.; Wang, H.; Li, F. Unsupervised learning of human action categories using spatial-temporal words. *Int. J. Comput. Vis.* **2008**, *79*, 299–318.
27. Ryoo, M.S.; Aggarwal, J.K. Spatio-Temporal Relationship Match: Video Structure Comparison for Recognition of Complex Human Activities. In Proceedings of 12th IEEE International Conference on Computer Vision (ICCV), Kyoto, Japan, 29 September–2 October 2009; pp. 1593–1600.
28. Shechtman, E.; Irani, M. Space-Time Behavior Based Correlation. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20–25 June 2005; pp. 405–412.
29. Gorelick, L.; Blank, M.; Shechtman, E.; Irani, M.; Basri, R. Actions as Space-Time Shapes. In Proceedings of International Conference on Computer Vision, Beijing, China, 17–21 October 2005; pp. 1395–1402.
30. Chandrashekhar, V.; Venkatesh, K.S. Action Energy Images for Reliable Human Action Recognition. In Proceedings of ASID, New Delhi, India, 8–12 October 2006; pp. 484–487.
31. Oikonomopoulos, A.; Patras, I.; Pantic, M. Spatiotemporal salient points For visual recognition of human actions. *IEEE Trans. Syst. Man. Cybern.* **2005**, *36*, 710–719.
32. Ke, Y.; Sukthankar, R.; Hebert, M. Spatio-Temporal Shape and Flow Correlation for Action Recognition. In Proceedings of Computer Vision and Pattern Recognition (CVPR), Minneapolis, MN, USA, 18–23 June 2007; pp. 1–8.
33. Darrell, T.; Pentland, A. Space-Time Gestures. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA, 15–17 June 1993; pp. 335–340.

34. Gavrila, D.M.; Davis, L.S. Towards 3-D Model-Based Tracking and Recognition of Human Movement: A Multi-view Approach. In Proceedings of International workshop on automatic face-and gesture-recognition, Coral Gables, FL, USA, 21–23 November 1995; pp. 253–258.
35. Holte, M.B.; Moeslund, T.B.; Nikolaidis, N.; Pitas, I. 3D Human Action Recognition for Multi-View Camera Systems. In Proceedings of 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), Hangzhou, China, 16–19 May 2011; pp. 342–349.
36. Aghajan, H.; Wu, C. Layered and Collaborative Gesture Analysis in Multi-Camera Networks. In Proceedings of 2007 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007), Honolulu, HI, USA, 15–20 April 2007; Volume 4, pp. IV-1377–IV-1380.
37. Srivastava, G.; Iwaki, H.; Park, J.; Kak, A.C. Distributed and Lightweight Multi-Camera Human Activity Classification. In Proceedings of Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), Como, Italy, 30 August–2 September, 2009; pp. 1–8.
38. Natarajan, P.; Nevatia, R. View And Scale Invariant Action Recognition Using Multiview Shape-Flow Models. In Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
39. Weinland, D.; Ronfard, R.; Boyer, E. Free viewpoint action recognition using motion history volumes. *Comput. Vis. andImage Underst.* **2006**, *104*, 249–257.
40. Yan, P.; Khan, S.; Shah, M. Learning 4D Action Feature Models for Arbitrary View Action Recognition. In Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK, USA, 24–26 June 2008; pp. 1–7.
41. INRIA. IXMAS action dataset. Available online: http://4drepository.inrialpes.fr/public/datasets (accessed on 1 December 2012).
42. Bobick, A.; Davis, J. Real-Time Recognition of Activity Using Temporal Templates. In Proceedings of 3rd IEEE Workshop on Applications of Computer Vision, Sarasota, FL, USA, 2 December 1996; pp. 39–42.
43. Platt, J.C. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers*; MIT Press: Cambridge, MA, USA, 1999; pp. 61–74.
44. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
45. Yang, X.; Zhang, C.; Tian, Y. Recognizing Actions Using Depth Mmotion Maps-Based Histograms of Oriented Gradients. In Proceedings of 20th ACM International Conference on Multimedia, Nara, Japan, 29 October–2 November 2012; pp. 1057–1060.
46. Huang, C.; Hsieh, C.; Lai, K.; Huang, W. Human Action Recognition Using Histogram of Oriented Gradient of Motion History Image. In Proceedings of First International Conference on Instrumentation, Measurement, Computer, Communication and Control, Beijing, China, 21–23 October 2011; pp. 353–356.

47. Kläser, A.; Marszałek, M.; Schmid, C. A Spatio-Temporal Descriptor Based on 3D-Gradients. In Proceedings of British Machine Vision Conference, Leeds, UK, 1–4 September 2008; pp. 995–1004.
48. Kulathumani, V.; Ramagiri, S.; Kavi, R. WVU multi-view activity recognition dataset. Available online: http://www.csee.wvu.edu/ vkkulathumani/wvu-action.html (accessed on 1 April 2013).