Abstract—Multi-view human action recognition has gained a lot of attention in recent years for its superior performance as compared to the single view recognition. In this paper, we propose algorithms for the real-time realization of human action recognition in distributed camera networks (DCNs). We present a new method for fast calculation of motion information by Motion Local Ternary Pattern (Mltp) that is tolerant to illumination change, robust in homogeneous region and computationally efficient. Next, we combine the local interest point detector with Mltp to generate 3D patches containing motion information and introduce two feature descriptors for the extracted 3D patches. Taking advantage of the proposed Mltp, 3D patches generated from background can be further removed automatically and thus the foreground patches can be highlighted. Finally, the histogram representations based on Bag-of-Words modeling, are transmitted from local cameras to the base station for classification. At the base station, a probability model is produced to fuse the information from various views and a class label is assigned accordingly. Compared to the existing algorithms, the proposed methods have three advantages: 1) no preprocessing is required; 2) communication among cameras is unnecessary; and 3) positions and orientations of cameras do not need to be fixed. We further evaluate both descriptors on the most popular multi-view action dataset IXMAS. Experimental results indicate that our approaches repeatedly achieve state-of-the-art results when various numbers of views are tested. In addition, our approaches are tolerant to the various combination of views and benefit from introducing more views at the testing stage.

I. INTRODUCTION

The vision-based human action recognition has become an active research area of computer vision, due to its promise in many application domains, including visual surveillance, video indexing, gesture recognition, human-computer interaction, etc. Different from the single view action recognition, multi-view learning provides robustness to partial occlusions, viewpoint changes, and thus performs better than single view recognition. However, position and orientation variances of cameras also, introduces more diversity, making the problem quite challenging.

In the multi-view action recognition literature, three directions have been heavily investigated, including 1) geometry-based approaches [16], [17] that directly estimate 3D shapes and poses from multi-view inputs; 2) view-invariant representation, such as the Maximum Margin Clustering in [6] to explore the split-based features that are transferable across views, the Bag-of-Bilingual-Words [11] which can bridge the semantic gap across view-dependent vocabularies, and the framework presented in [14] that can give a viewpoint-invariant representation of primitive actions; 3) frame-based feature extraction, such as the Self-Similarity Matrix used in [8] to compute the pairwise similarity between any pair of figure-centric frames.

Although with good performance, most of the existing algorithms in multi-view action recognition focus on the recognition accuracy only and lack practical consideration in terms of the real-time implementation. For example, some algorithms [16], [17] require complicated preprocessing such as silhouette extraction and matching, and others [8] require heavy communication between cameras and base station, which causes large memory and bandwidth consumption. With the emergence of distributed camera networks (DCNs) [4], the deployment of multi-view action recognition in DCN becomes a logical next step. In the DCN environment, an object could be surrounded by multiple cameras and observed by these cameras from different views. These cameras have imaging, on-board processing, and wireless communication capabilities. Collaboratively they could solve computer vision problems through distributed sensing and processing. However, resource constraint is the major limitation of DCNs since each camera has only limited memory and power supply, and the communication between local cameras and base station is also expensive and limited by the bandwidth. These constraints have largely hindered the successful deployment of existing multi-view action recognition algorithms in DCNs.

To resolve the conflict between the constraint resource in DCNs and the need for real-time recognition, two issues need to be investigated. First, what information should be extracted by each local camera in order to satisfy the bandwidth requirement without jeopardizing recognition accuracy? Second, how to fuse the information collected from distributed cameras for recognition purpose? In the DCN environment, the transmission of the raw image data to the base station is not feasible due to the expensive transmission cost, and thus the distributed feature-based recognition is necessary.

To explore the motion feature, we propose Mltp as a new operator to calculate the motion information among frames. Mltp compares each pixel at every frame with its vertical and horizontal neighboring pixels in the previous and the next frames, respectively. A ternary pattern can then be generated to describe the motion directions based on the comparisons. Specifically, Mltp takes advantage of Local Binary Pattern
(LBP) [12] in terms of computational efficiency, tolerance to illumination change and robustness in homogeneous regions. Based on the proposed Mltp, two distinctive descriptors of 128 and 64 dimensions, respectively, can be constructed with one based on the bin-coding method and the other on random projection. In order to further reduce the transmission cost, the Bag-of-Word (BoW) technique is used to generate a histogram representation for every video. Given a 50 frames video clip with image resolution as 120 x 160 pixels, instead of transmitting 960,000 pixels, the amount of data transfer required by our system is only \( n \) (\( n < 300 \)) bins where each bin is represented by 4 bytes. During the whole process, communication among cameras is unnecessary. To fuse the information transmitted from local cameras, we use Naive Bayesian to integrate the prediction results from the Support Vector Machine (SVM) [3] to give the final decision.

Recently, a distributed and lightweight multi-view action classification scheme [15] has been proposed to perform action recognition in DCN environment. However, this work requires the actor’s orientation to be along one of the cameras. In this paper, we focus on the design of algorithms for distributed and robust multi-view action recognition, which requires low memory and bandwidth consumption. Here, “distributed” refers to the fact that each camera processes its data locally and just sends limited information to the base station, and “robust” refers to the fact that the recognition accuracy is not affected much by the selection of cameras. Different from the work of [15], our system does not need to fix the orientation of actors and no communication among cameras is necessary.

The main contributions of our work are summarized as follows: 1) we propose Mltp to calculate the motion information among frames, which is computationally efficient and distinctive for motion patterns without any additional information such as background subtracted silhouettes or 3D visual hull volumes; 2) we propose two compact but effective feature descriptors based on the motion information calculated from Mltp, a bin coding method to shorten the histogram representation of Mltp and random projection based dimensionality reduction without training process; and 3) we use the Naive Bayesian fusion technique and SVM to integrate the information sent from distributed cameras without the need of communication between local cameras. Compared to the existing algorithms [10], [16], [17] and applications [15] on the similar DCN environment, our proposed methodology achieves higher recognition accuracy, requires less memory and is with lower bandwidth consumption.

II. PROPOSED METHODOLOGY

Our methodology for distributed multi-view action recognition involves four main steps: 1) feature detection that localizes the positions of interest points; 2) feature description that generates the distinctive feature descriptors for any 3D patches around the detected interest points; 3) feature quantization which aims to generate the histogram representation for every action video; and 4) distributed multiple view action recognition that fuses information sent from local cameras to generate the final class label for a specific action.

A. Feature Detection

In this paper, the Cuboid detector proposed by Dollar [5] is used for spatio-temporal interest points detection. The Cuboid detector has been widely used for feature detection in the field of multi-view action recognition [10], [15]. Instead of using a 3D filter on the spatio-temporal domain, it applies two separate linear filters respectively to both spatial and temporal dimensions. The response function at pixel location \( I(x,y,t) \) is of the form \( R = (I(x,y,t) \ast g\sigma(x,y) \ast h\sigma(t))^2 + (I(x,y,t) \ast g\od(t))^2 \), where \( g\sigma(x,y) \) is the 2D Gaussian smoothing function that applied only in the spatial domain. The \( h\sigma \) and \( h\od \) are a quadrature pair of 1D Gabor filters that applied in the temporal direction. The 1D Gabor filters are defined as \( h\sigma(t;\tau,\omega) = -\cos(2\pi t \omega) \exp^{-t^2/\tau^2} \) and \( h\od(t;\tau,\omega) = -\sin(2\pi t \omega) \exp^{-t^2/\tau^2} \), where \( \omega = 4/\tau \). The parameters \( \sigma \) and \( \tau \) roughly correspond to the spatial and temporal scale. The interest points are detected at locations where the response is a local maximum. The spatio-temporal patches around the points are extracted for feature description purpose. Hereinafter, we use the terms “3D patches” and “extracted cuboids” interchangeably.

B. Feature Description

Feature description is a process to describe the detected 3D patches. After feature detection, many 3D patches can be generated. Instead of using the original 3D patches, we derive two feature descriptors of much lower dimension based on the so-called 3D motion patch. In the following, we first describe the proposed Mltp that generates the 3D motion patch. We then detail the two feature descriptors designed based on the 3D motion patch.

1) Motion Local Ternary Pattern (Mltp): We first describe a new operation used for capturing motion information by combining the effective description properties of Local Binary Pattern (LBP) [12] with the intensity similarity characteristic of the neighboring pixels among adjacent frames. For any pixel \((x,y)\) at frame \(t\), motion will cause the intensity change of its neighboring pixels at the previous frame \(t - \Delta t\) and the next frame \(t + \Delta t\). Mltp is designed to capture the effect of motion at local structure.

![Fig. 1. Encoding process of Mltp.](image_url)
Figure 1 shows the details of Mltp using a 4-neighbor definition. For an image patch centered at \( n_c \), pixels \( n_i \sim n_3 \) and \( n_i \sim n_7 \) are the four inner and outer neighbors of the center pixel, respectively. The radius of the neighborhood \( D \) is defined as the distance between the center pixel and any outer pixel, i.e., 2 in Figure 1. At frame \( t \), the Mltp feature of the center pixel \( n_c \) is calculated by three steps. First, the mean values of the four inner and outer neighbors are calculated by \( N_i = 0.55 \times n_i + 0.45 \times n_{i+1} \), \( i = 0, 1, 2, 3 \), in order to reduce the effect of noise. The parameters 0.55 and 0.45 are empirically selected based on extensive experimental study. Second, differences between the \( N_i \)s and the center pixel intensity, \( n_c \), are calculated for the previous frame \( D_i = |N_i - n_c| \) as well as for the next frame \( D'_i = |N'_i - n_c| \). Third, a ternary pattern is assigned to each comparison:

\[
\text{Mltp} = \sum_{i=0}^{3} f(D_i - D'_i)^3 \quad (1)
\]

\[
f(x) = \begin{cases} 
1 & x < - \max\{T_m \times n_c, t_m\} \\
2 & x > \max\{T_m \times n_c, t_m\} \\
0 & \text{else}
\end{cases} \quad (2)
\]

where \( T_m \) is a parameter that need to be selected. The value \( T_m \times n_c \) is considered as a dynamic threshold depending on the center pixel’s intensity to ensure its sensitivity at dark regions. The parameter \( t_m \) is used for noise compression, and its value is assigned to be 5 (for any 256-level gray scale images). This value is obtained under extensive experimental study.

Figure 2 shows the result of Mltp on nearby frames from the IXMAS multi-view action dataset [16]. Figure 2(a) are three subsequent frames representing the motion of hands up, while Figure 2(c) showing the opposite motion of hands down. The directions of motion are encoded in three gray scale values (0 or black, 125 or gray and 255 or white) as shown in Figure 2(b) and Figure 2(d). Comparing the locations of pixels with 125 intensity value (motion caused by the difference between the current frame and the next frame) and pixels with 255 intensity value (motion caused by the difference between the current frame and the previous frame), the two opposite motions are distinctively described. Moreover, all the background pixels, as well as body parts with no contribution to the motion, are captured and assigned with zero values.

Although Mltp is inspired by the work of Yeffet and Wolf [18], it differs from the previous work in the field of action recognition in three aspects. 1) The pixel-wise comparison, as compared to the patch-wise comparison in [18], generates clear motion image with thin and smooth edges, making the detected/sampled motion patches distinctive even at smaller scales. 2) Mltp uses a dynamic threshold that ensures the capture of motion information even in dark regions. 3) Mltp generates less number of codes, which facilitates the realization of compact representation.

For pixels within an extracted 3D patch from feature detection, the Mltp values can be calculated according to Eq. 1. The resulting 3D patch is referred to as the 3D motion patch.

We further present two descriptors to explore the effectiveness of the calculated 3D motion patch.

2) Feature Descriptors Mltp-hist and Mltp-rp: For the LBP-like features, the traditional method is to construct a histogram representation. In this paper, we construct a histogram of Mltp in an unconventional manner, which reduces the number of possible bins from \( 3^4 = 81 \) to 16. The string of Mltp contains four digits with 3 possible values for each. We first divide the 4-digit string into 2 shorter strings with each string containing 2 digits. In this manner, each short string is presented by a histogram of \( 2 \times 2^2 = 8 \) bins, where each short string is counted twice. Finally, concatenating the two histograms generated by short strings, a full length histogram of \( 2 \times 8 = 16 \) bins is constructed. For any 3D motion patch, we divide it into \( n_{\sigma} \times n_{\sigma} \times n_c \) cells and concatenate the normalized histograms into a feature vector at the length of \( 16n_{\sigma}^2n_c \). This feature descriptor is referred to as Mltp-hist.

For the second feature descriptor, we apply dimensionality reduction technique on the extracted 3D motion patches. A long vector representation can be formulated by concatenating the Mltp values within any 3D motion match. In order to avoid the training phase as principal component analysis, we apply the Random Projection (RP) [2] on the vector representation for dimensionality reduction. In practice, any normally distributed random matrix \( R \) with zero mean and unit variance serves the purpose. For a vector representation of 3D motion patch \( X \in \mathbb{R}^n \), where \( n \) is the number of pixels within each 3D motion patch, the new descriptor \( Y \in \mathbb{R}^d \) can be obtained by \( Y = RX \) where \( R \in \mathbb{R}^{d \times n} (d \ll n) \). We refer to this feature descriptor as d-dimension Mltp-rp.

Mltp-hist and Mltp-rp explore different characteristics of the 3D motion patches. For Mltp-hist, it records the occurrences of various codes and keeps the spatial-temporal information by dividing every 3D patch into small cells. For Mltp-rp, it generates a low-dimensional feature vector by linear projection which will not introduce distortion in data.

C. Feature Quantization

In order to further reduce the transmission cost, we use feature quantization to obtain the histogram representation of the collected feature descriptors from a video and transmit the histogram to the base station. Feature quantization is a process that maps every feature vector to the codewords of a trained codebook. In this paper, the well known Bag-of-Words (BoW) model is explored for codebook training. Every video is further presented as a histogram of codeword occurrences.

The BoW model contains two main steps. The first step is to train the codebook. After collecting a specific number of feature vectors (Mltp-hist or Mltp-rp) from different subjects and different actions in the training dataset, k-means clustering can be used for learning the codewords. Every cluster center corresponds to a codeword. The second step is the histogram representation. Each feature vector from a video clip can be assigned to a specific codeword that is closest to it in terms of Euclidean distance.
D. Distributed Multiple View Action Recognition

In our system, one object is surrounded by several cameras capturing various views. Cameras from distinctive views will record the same action simultaneously. For the video recorded by different cameras, the histogram representation calculated by the BoW model is used as a new representation of the original video. Each camera can process the data independently and only transmits the calculated histogram to the base station for further analysis. It should be pointed out that our system does not need data transmission between cameras and only the histogram representation of every video is transmitted to the base station. The DCN systems like WiCa [9], CITRIC [4] and Imote2 [1] are some examples of smart camera platforms where our algorithms can be implemented. At the base station, we use a Naive Bayesian technique to obtain the final decision based on the histogram collections from multiple views.

At the base station, we first apply the Support Vector Machine (SVM) [3] with the $\chi^2$-kernel [7] for prediction based on each cameras input:

$$K(H_i, H_j) = \exp \left( -\frac{1}{2A} \sum_{n=1}^{S} \frac{(h_{in} - h_{jn})^2}{h_{in} + h_{jn}} \right)$$  \hspace{1cm} (3)

where $H_i = \{h_{in}\}$ and $H_j = \{h_{jn}\}$ are the histograms calculated by BoW, and $S$ is the number of codewords. $A$ is the mean value of distances between all training samples [19].

The second step performed at the base station is the integration of probability outputs of SVM [13] using the Naive Baysian technique for the purpose of multiple view action recognition. Same kernel function is used as Eq. 3. Each view $V_j$ will have a probability for action label $l_i$ as $P(l_i|V_j)$. Let the number of cameras be $v$ and the number of action classes be $c$. Our objective is to calculate the probability of $l_i$ when $V_1, \ldots, V_v$ are available, referred to as $P(l_i|V_1, V_2, \ldots, V_v), i = 1, \ldots, c$. To simplify the problem, we assume the selection of each camera is independent, so that $P(V_1, \ldots, V_v) = P(V_1) \times \cdots \times P(V_v)$. In addition, the probability of every action is assumed to be equal.

For any class label $i, i = 1, \ldots, c$ in a $v$-camera environment, its probability can be calculated as

$$P(l_i|V_1, \ldots, V_v) = \frac{P(V_1|l_i) \cdots P(V_v|l_i) P(l_i)}{P(V_1) \cdots P(V_v)}$$  \hspace{1cm} (4)

Since $P(l_1) = P(l_2) = \cdots P(l_c)$, the probability of action class $i$ can be calculated by multiplying all the local probabilities $P(l_i|V_j), j = 1, \ldots, v$. For any test sequence, the class label can be assigned to the one with the highest value of Eq. 4.

III. EXPERIMENTS AND RESULTS

We evaluate our approaches on the IXMAS multi-view action data set [16] which contains 13 daily-life motions performed each 3 times by 11 actors. The actors choose free position and orientation. Figure 3 shows two actions captured from 4 cameras. In order to compare with other algorithms, we choose the same experimental setting as [15], [16] which use 11 action categories and 10 subjects. In addition, we select four views excluding the top view, same as [10], [15].
A. Parameter Setup

From each video 200 cuboids are extracted with spatial scale \( \sigma = 2 \) and temporal scale \( \tau = 2.5 \). The size of the cuboids (or 3D patches) is then \( 13 \times 13 \times 17 \) pixels, which corresponds to 2873 pixels in total. For Mltp, the parameters for \( \Delta t = 3 \), \( T_m = 0.1 \) and \( D = 1 \) are experimentally selected. In addition, as shown in Figure 2(b) and 2(d), the Mltp images can reflect the locations where no motion exists. We further evaluate every 3D motion patch by counting the number of pixels with zero Mltp value and mark it as \( m_0 \). Any 3D motion patch with \( m_0/2873 \geq 0.9 \) is considered non-informative and removed from further analysis since no obvious motion is captured. Figure 4 shows the results of applying this scheme. The colorful squares show the extracted cuboids [5] in the spatial domain. For both figures, the left images reflect the location of original patches, and the right images reflect the results after removing patches without obvious motion. It is clear to see that the noisy patches generated from background are removed, which better highlights the foreground patches.

As described in Sec. II-B, feature descriptors Mltp-hist and Mltp-rp can be constructed. For Mltp-hist, the dimension of the feature descriptor is 128 since we divide every cuboid into \( 2 \times 2 \times 2 \) cells. For Mltp-rp, the dimension of feature descriptor is set to be half of the Mltp-hist, and thus we generate a random matrix \( R \) with size of \( 64 \times 2873 \).

For feature quantization, we select 4 videos of actors for training the codebook. For the size of codebook, we evaluated various numbers of codewords and set it to be 180 for Mltp-hist and 120 for Mltp-rp.

B. Single View Recognition

We first evaluate the proposed algorithms in the case of single view action recognition. The training samples are collected from all the four views, and only one view is available during the testing phase. We use the Leave-One-Out strategy, namely videos from 9 actors for learning and the rest one actor’s videos for testing. We then use the probability output of SVM for classification as described in Sec. II-D. Table I compares our proposed approaches with state-of-the-art algorithms for the single view recognition, which shows that our models outperform other algorithms by \( 2\% \sim 12\% \).

It should be pointed out that our models do not need any preprocessing such as background subtraction or 3D visual hull volumes. In addition, the adopted random projection-based dimensionality reduction does not need any training samples, which are necessary for approaches in [10], [15]. Moreover, we trained the codebook on 4 actors only, as compared to 9 used in [15].

Figure 5 shows the performance of 4 cameras for the single view action recognition, when we use the model of Mltp-hist+BoW. It shows that actions with large motions such as ”sit down”, “get up”, ”turn around” and “walk” have high accuracy (100%) even for the single view recognition.

C. Multiple View Recognition

In the case of multiple view recognition, we use Eq. 4 to obtain the probability of every action and assign the class label to the one with the highest value. Same as the single view recognition, parameters and models are trained from four views, and different numbers of views are tested respectively. The various combination of views are explored and the maximum and averaged recognition accuracies are reported in Table II. From the table, we observe that the performance of Mltp-hist is slightly better than that of Mltp-rp, but it requires larger number of codewords and has slightly larger standard deviation values. Recall that the standard deviation of recognition accuracies using different combinations of views reflect the robustness of the proposed algorithms - the smaller the derivation, the less affected the algorithm to the selection of different views.

Experimental results indicate that our proposed methods
perform better than existing distributed multi-view action recognition algorithms, and even superior to the complicated algorithms that require preprocessing and large data transmission. For example, our methods achieve higher accuracies using 3 views than other algorithms using 4 views, as shown in Table II. In addition, our algorithms are less sensitive to the different combinations of views selected for testing, since the average accuracies are close to the highest accuracies and the standard deviation values are small. Different from the work of [16], [17], the performance of our algorithms is steadily improved when more views are available at testing stage. This further indicates that information from various views can be well represented and fused in our systems. The improvements of recognition accuracy is more obvious from 2 views to 3 views or 1 view to 2 views, than 3 views to 4 views. In other words, 3 views are enough to achieve competitive accuracy.

D. Memory and Bandwidth Requirements

For the 128-dimension Mltp-hist, only the trained codebook with 180 codewords should be stored on every camera. For the 64-dimension Mltp-rp, the codebook with 120 codewords should be stored as well. In addition, every camera also needs to store the random projection matrix of size $64 \times 2573$. For both Mltp-hist and Mltp-rp, every camera only transmits the calculated histogram to the base station for classification. Therefore, the Mltp-hist+BoW transmits 4 histograms (from 4 cameras) with 180 bins to the base station, and Mltp-rp+BoW transmits 4 histograms with 120 bins. Using the same calculation as [15], both the memory and bandwidth requirements for our system can be calculated. Table III compares the memory and bandwidth consumption of our models and the ones in [15], [16]. It shows that our models require even less resource on memory and transmission bandwidth. In addition, the Mltp-hist+BoW is the best choice considering both the accuracy and resource consumption.

IV. CONCLUSION

In this paper, we proposed Mltp to extract the motion information among frames. Combined with the Cuboid detector, Mltp helps to remove non-informative patches generated from background. Two descriptors, Mltp-hist and Mltp-rp, generated by histogram representation of Mltp values and random projection, receptively, were mapped onto codebook trained by BoW. The proposed approaches do not need complicated operation for feature extraction or description. Experimental results indicated that both Mltp-hist and Mltp-rp outperform existing algorithms for distributed multi-view action recognition. Especially, both descriptors require less memory and bandwidth consumption than existing algorithms. In addition, communication among local cameras is unnecessary for information fusion in the proposed system.

ACKNOWLEDGEMENT

This work was supported in part by National Science Foundation under Grant NSF CNS-1017156.

REFERENCES


<table>
<thead>
<tr>
<th>Method</th>
<th># Cameras used for testing</th>
<th>Memory(Mbyte/s)</th>
<th>Bandwidth(Mbyte/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liu &amp; Shah [10]</td>
<td></td>
<td>0.092</td>
<td>0.689</td>
</tr>
<tr>
<td>Yan et al. [17]</td>
<td></td>
<td>0.315</td>
<td>1.72</td>
</tr>
<tr>
<td>Weinland et al. [16]</td>
<td></td>
<td>0.96</td>
<td>2.7</td>
</tr>
<tr>
<td>Srivastava et al. [16]</td>
<td></td>
<td>0.315</td>
<td>47</td>
</tr>
</tbody>
</table>

TABLE III

MEMORY AND BANDWIDTH REQUIREMENTS FOR OUR MODELS.