High Speed Action Recognition and Localization in Compressed Domain Videos

Chuohao Yeo, *Student Member, IEEE,* Parvez Ahammad, *Student Member, IEEE,*
Kannan Ramchandran, *Fellow, IEEE,* and S Shankar Sastry, *Fellow, IEEE*

***Abstract*—We present a compressed domain scheme that is able to recognize and localize actions in real-time[1]. The recognition problem is posed as performing an action video query on a test video sequence. Our method is based on computing motion similarity using compressed domain features which can be extracted with low complexity. We introduce a novel motion correlation measure that takes into account differences in motion directions and magnitudes. Our method is appearance invariant, requires no prior segmentation, alignment or stabilization, and is able to localize actions in both space and time. We evaluated our method on a large action video database consisting of 6 actions performed by 25 people under 3 different scenarios. Our classification results compare favorably with existing methods at only a fraction of their computational cost. We also perform a systematic investigation of the effects of various encoding options on our proposed approach. In particular, we present results on the compression-classification trade-off, which would provide valuable insight into jointly designing a system that performs video encoding at the camera front-end and action classification at the processing back-end.**

I. Introduction

THE use of video cameras has become increasingly common as their costs decrease. In personal applications, it is common for people to record and store personal videos that comprise various actions, in part due to the widespread availability of phone cameras and cheap cameras with video recording capabilities. In security applications, multiple video cameras record video data across a designated surveillance area. A good example of this is the large network of surveillance cameras installed in London. This proliferation of video data naturally leads to information overload. It would not only be incredibly helpful but also necessary to be able to perform rudimentary action recognition in order to assist the users in focusing their attention on actions of interest as well as allowing them to catalog their recorded videos easily.

In this paper, we formulate the problem of action recognition and localization as follows: given a query video sequence of a particular action, we would like to detect all occurrences of it in a test video, and thereby recognizing an action as taking place at some specific time and location in the video. The approach should be person independent, hence we want our method to be appearance invariant. In a surveillance setting, it is critical to be able to respond to events as they happen. Even in a consumer application, it is desirable to minimize processing time. Therefore, we want a solution that is fast and can operate in real-time.

Any practical system that records and stores digital video is likely to employ video compression such as H.263+ [2] or H.264 [3]. It has long been recognized that some of the video processing for compression can be reused in video analysis or transcoding; this has been an area of active research (see for example [4], [5]) in the last decade or so. Our approach exploits this insight to attain a speed advantage.

It is reasonable to assume that a surveillance application would consist of a front-end system that records, compresses, stores and transmits videos, as well as a back-end system that processes the transmitted video to accomplish various tasks. One focus in this paper is on the action recognition task that would presumably be performed at the back-end. However, we recognize that various engineering choices, such as the choice of video coding method, made at the front-end can have an impact on the action recognition performance in the back-end. In particular, we would also like to understand how various video coding choices impact the action recognition performance of our approach.

A. Related work

There has been prior work in action recognition using raw video without the use of body landmark points. Efros *et al.* [6] require the extraction of a stabilized image sequence before using a rectified optical flow based normalized correlation measure for measuring similarity. Shechtman and Irani [7] exhaustively test motion-consistency between small space-time (ST) patches to compute a correlation measure between a query video and a test video. With their method, they are able to detect multiple actions (similar or different) in the test video and also perform localization in both space and time. Schüldt *et al.* [8] use an approach based on local features in which Support Vector Machines (SVM) are used to classify actions in a large database of action videos that they collected.

There has also been prior work in performing action recognition in the compressed domain. Ozer *et al.* [9] applied Principal Component Analysis (PCA) on motion vectors from *segmented* body parts for dimensionality reduction before classification. They require that the sequences must have a fixed number of frames and be temporally aligned. Babu *et al.* [10] trained a Hidden Markov Model (HMM) to classify each action, where the emission is a codeword based on the histogram of motion vector components of the *whole*

The authors are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA (e-mail: zuohao@eecs.berkeley.edu; parvez@eecs.berkeley.edu; kannanr@eecs.berkeley.edu; sastry@eecs.berkeley.edu)

This work has been presented in part in [1].

[1]On a Pentium-4 2.6 GHz workstation with 1 GB of RAM

frame. In later work [11], they extracted Motion History Image (MHI) and Motion Flow History (MFH) [12] from compressed domain features, before computing global measures for classification.

B. Contributions

Our proposed method makes use of motion vector information to capture the salient features of actions which are appearance invariant. It then computes frame-to-frame motion similarity with a measure that takes into account differences in both orientation and magnitude of motion vectors. The scores for each space-time candidate are then aggregated over time using a method similar to [6]. Our approach is able to localize actions in space and time by checking all possible ST candidates, much like in [7], except that it is more computationally tractable since the search space is greatly reduced from the use of compressed domain features. Our novelty lies in our ability to perform *real-time* localization of actions in space and time by a novel combination of signal processing and computer vision techniques. The proposed method requires no prior segmentation, no temporal or spatial alignment and minimal training.

We also study how various encoding options affect the performance of our proposed approach. This aspect is often overlooked in most other compressed domain processing work, in which results are typically presented only on a single choice of encoding parameters. However, we recognize that different encoding options not only affect compression performance but also influence the performance of compressed domain processing. Hence, in this work, we undertake a systematic investigation to determine the trade-offs between compression performance and classification performance. This would be useful in understanding how best to choose encoding options to strike a good balance between compression and classification, and between speed and accuracy.

The rest of the paper is organized as follows. Section II outlines our proposed method and describes each step in detail. The experimental setup and results are discussed in section III, and we discuss the effects of different video encoding options in section IV. We then present our concluding remarks in section V.

II. Approach

Given a query video template and a test video sequence, we carry out the steps shown in Figure 1 to compute a score for how confident we are that the action presented in the query video template is happening an each space-time location (to the nearest macroblock and frame) in the test video. We will elaborate on each of these steps in the following subsections.

In this paper, X^p denotes a video, with $p \in \{\text{test}, \text{query}\}$ referring to either the test video or query video. Each video X^p has T^p frames, with each frame containing $N^p \times M^p$ macroblocks. We assume that an *action* induces a motion field that can be observed as a spatio-temporal pattern; let $\vec{V}^p$ be the spatio-temporal pattern (motion field) associated with video X^p. Furthermore, $\vec{V}^p_{n,m}(i) = [V^{p,u}_{n,m}(i) \quad V^{p,v}_{n,m}(i)]$ denotes the motion vector at location (n, m) in frame i of X^p.

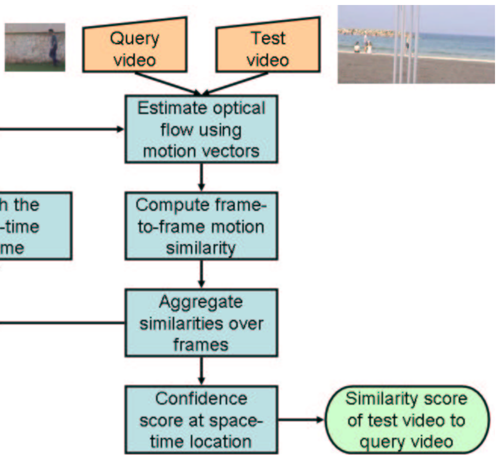

Query video
Test video
Estimate optical flow using motion vectors
Search the space-time volume
Compute frame-to-frame motion similarity
Aggregate similarities over frames
Confidence score at space-time location
Similarity score of test video to query video

Fig. 1. Flow chart of action recognition and localization method. Optical flow in the query and test videos are first estimated from motion vector information. Next, frame-to-frame motion similarity is computed between all frames of the query and test videos. The motion similarities are then aggregated over a series of frames to enforce temporal consistency. To localize, these steps are repeated over all possible space-time locations. If an overall similarity score between the query and test videos is desired, a final step is performed with the confidence scores.

Our working assumption is that similar actions will induce similar motion fields. We will use $(\boldsymbol{u})_+$ as a shorthand for $\max(0, \boldsymbol{u})$.

A. Estimation of coarse optical flow

Motion compensation is an integral component of modern video compression technology and motion vectors are by-products of the motion compensation process. Motion vectors are obtained from block matching and can be interpreted as crude approximations of the underlying motion field or optical flow. In addition, the Discrete Cosine Transform (DCT) coefficients can also be used to provide a confidence measure on the estimate. We follow the approach outlined by Coimbra and Davies [13] for computing a coarse estimate and a confidence map of the optical flow. To generate the optical flow estimate, we use the following rules [13]:

1) Motion vectors are normalized by the temporal distance of the predicted frame to the reference frame, and their directions are reversed if the motion vectors are forward-referencing.
2) Macroblocks with no motion vector information (e.g. macroblocks in I-frames and intra-coded macroblocks) retain the same optical flow estimate as in the previous temporal frame.
3) Macroblocks with more than one motion vector (e.g. bi-directionally predicted macroblocks in B-frames) take as the estimate a weighted average of the motion vectors, where the weights are determined by their temporal distance to the respective reference frames.

It has been recognized that optical flow estimation performance at each image location depends on the amount of texture in its local neighborhood [14]. In particular, if the local neighborhood suffers from the aperture problem, then it is likely to have an unreliable optical flow estimate. By thresholding a confidence measure derived from the DCT coefficients that measures the amount of texture in the block [13], we can filter out optical flow estimates that are likely to be noisy. To compute the confidence measure for intra-coded macroblocks, we use [13]:

$$\lambda = \frac{1}{N}\sum_{i=1}^{N}\left(F_i(0,1)^2 + F_i(1,0)^2\right) \tag{1}$$

where λ is the confidence measure, and $F_i(u,v)$ is the 2D DCT of the ith block, $f_i(x,y)$, out of N blocks within the macroblock. Coimbra and Davies have shown that $F_i(1,0)$ and $F_i(0,1)$ can be interpreted as a weighted average of spatial gradient in the x and y direction respectively [13]. For predicted macroblocks, we update the confidence map by taking a weighted average of the confidence map in the reference frame(s) as indicated by motion vector information.

By thresholding λ, we then decide whether to keep the optical flow estimate for the block or to set it to zero, hence obtaining $\vec{V}^p$. As we will show later in section III-B, this step removes unreliable estimates and greatly improves the classification performance of our proposed algorithm.

B. Computation of frame-to-frame motion similarity

For the purpose of discussion in this section, both the test frame and query frame are assumed to have a spatial dimension of $N \times M$ macroblocks (the equal size restriction will be lifted later). We would like to measure the motion similarity between the motion field of the ith test frame, $\vec{V}^{\text{test}}_{n,m}(i)$, and that of jth query frame, $\vec{V}^{\text{query}}_{n,m}(j)$.

One way of measuring similarity is to follow the approach taken by Efros *et al.* [6]. Each motion field is first split into non-negative motion channels, e.g. $(V^{p,u}_{n,m}(i))_+$, $(-V^{p,u}_{n,m}(i))_+$, $(V^{p,v}_{n,m}(i))_+$ and $(-V^{p,v}_{n,m}(i))_+$. We can then vectorize these channels and stack them into a single vector $\vec{U}^p(i)$. The similarity between frame i of the test frame and frame j of the query frame, $\tilde{S}(i,j)$, is then computed as a normalized correlation:

$$\tilde{S}(i,j) = \frac{\langle \vec{U}^{\text{test}}(i), \vec{U}^{\text{query}}(j)\rangle}{\|\vec{U}^{\text{test}}(i)\|\|\vec{U}^{\text{query}}(j)\|} \tag{2}$$

We will refer to this similarity measure as *Non-negative Channels Normalized Correlation* (NCNC).

NCNC does not take into account the differences in magnitudes of individual motion vectors. To address this, we propose a novel measure of similarity:

$$\tilde{S}(i,j) = \frac{1}{Z(i,j)}\sum_{n=1}^{N}\sum_{m=1}^{M} d(\vec{V}^{\text{test}}_{n,m}(i), \vec{V}^{\text{query}}_{n,m}(j)) \tag{3}$$

where if $\|\vec{V}_1\| > 0$ and $\|\vec{V}_2\| > 0$,

$$\begin{aligned} d(\vec{V}_1,\vec{V}_2) &= \frac{\left(\langle \vec{V}_1,\vec{V}_2\rangle\right)_+}{\|\vec{V}_1\|\|\vec{V}_2\|}\cdot \min\left(\frac{\|\vec{V}_1\|}{\|\vec{V}_2\|}, \frac{\|\vec{V}_2\|}{\|\vec{V}_1\|}\right) \\ &= \frac{\left(\langle \vec{V}_1,\vec{V}_2\rangle\right)_+}{\max\left(\|\vec{V}_2\|^2, \|\vec{V}_1\|^2\right)} \end{aligned} \tag{4}$$

and $d(\vec{V}_1,\vec{V}_2) = 0$ otherwise. Also, the normalizing factor, $Z(i,j)$, in (3) is:

$$Z(i,j) = \sum_{n=1}^{N}\sum_{m=1}^{M} \mathbb{I}[\|\vec{V}^{\text{test}}_{n,m}(i)\| > 0 \text{ or } \|\vec{V}^{\text{query}}_{n,m}(j)\| > 0] \tag{5}$$

In other words, we want to ignore macroblocks in both the query and test video which agree on having no motion. This has the effect of not penalizing corresponding zero-motion regions in both the query and test video. We term this novel measure Non-Zero Motion block Similarity (NZMS).

C. Aggregation of frame-to-frame similarities

Section II-B describes how to compute $\tilde{S}(i,j)$, which tells us how similar the motion fields of frame i of the test frame and frame j of the query frame are. To take temporal dependencies into account, we need to perform an aggregation step. We do this by convolving $\tilde{S}(i,j)$ with a $T \times T$ filter parameterized by α, $H_\alpha(i,j)$, to get an aggregated similarity matrix $S(i,j) = (\tilde{S} * H_\alpha)(i,j)$ [6]. $S(i,j)$ tells us how similar a T-length sequence centered at frame i of the test video is to a T-length sequence centered at frame j of the query video. $H_\alpha(i,j)$ can be interpreted as a bandpass filter that "passes" actions in the test video that occur at approximately the same rate as in the query video. We use the following filter [6]:

$$H_\alpha(i,j) = \sum_{r\in R} e^{-\alpha(r-1)}\left(\chi(i,rj) + \chi(j,ri)\right)\ , -T/2 \le i,j \le T/2 \tag{6}$$

where

$$\chi(u,v) = \begin{cases} 1 & \text{if } u = \text{sign}(v)\cdot\lfloor|v|\rfloor \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

R is the set of rates (which has to be greater than one) to allow for and α ($\alpha \ge 1$) allows us to control how tolerant we are to slight differences in rates; the higher α is, the less tolerant it is to changes in the rates of actions. Figure 2(a) shows this kernel graphically for $\alpha = 2.0$.

Figure 2(b) shows a pre-aggregation similarity matrix, $\tilde{S}(i,j)$. Note the presence of near-diagonal bands, which is a clear indication that the queried action is taking place in those frames. Figure 2(c) shows the post-aggregation similarity matrix, $S(i,j)$, which has much smoother diagonal bands.

We will show later in section III-C that this aggregation step is crucial in performing action classification. However, the choice of α is not that important; experimental results show that performance is relatively stable over a range of α.

D. Space-time localization

Sections II-B and II-C tell us how to compute an aggregated similarity between each frame of a T^{test}-frames test sequence and each frame of a T^{query}-frames query sequence, both of which are $N \times M$ macroblocks in spatial dimensions. To compute an overall score on how confident we are that frame i of the test frame is from the query sequence, we use:

$$C(i) = \max_{\substack{\max(i-\frac{T}{2},1)\le k\le \min(i+\frac{T}{2},T^{\text{test}}) \\ 1\le j\le T^{\text{query}}}} S(k,j) \tag{8}$$

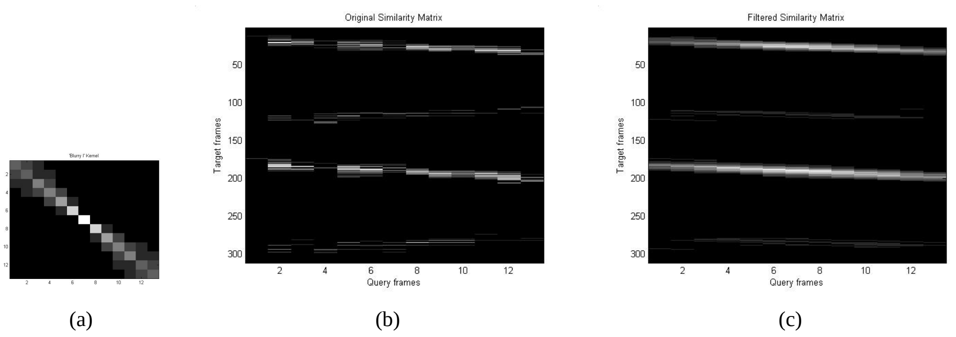


Fig. 2. An example similarity matrix and the effects of applying aggregation. In these graphical representations, bright areas indicate a high value. (a) Aggregation kernel, (b) Similarity matrix before aggregation, (c) Similarity matrix after aggregation. Notice that the aggregated similarity matrix is less noisy than the original similarity matrix.

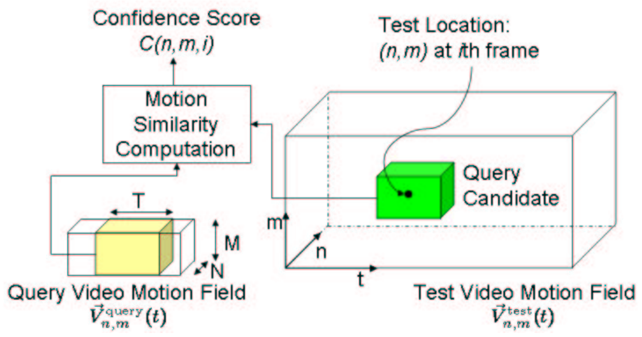

Confidence Score
C(n,m,i)
Motion Similarity Computation
Test Location: (n,m) at ith frame
Query Candidate
T
M
N
m
n
t
Query Video Motion Field
$\vec{V}^{\text{query}}_{n,m}(t)$
Test Video Motion Field
$\vec{V}^{\text{test}}_{n,m}(t)$

Fig. 3. Illustration of space-time localization. The query video space-time patch is shifted over the entire space-time volume of the input video, and the similarity, $C(n,m,i)$ is computed for each space-time location.

Maximizing $S(k,j)$ over j of the query video allows us to pick up the best response that a particular frame of the test video has to the corresponding frame in the query video. We also maximize $S(k,j)$ over k in a T-length temporal window centered at i. The rationale is that if a T-length sequence centered at frame k of the test video matches well with the query video, then all frames in that T-length sequence should also have at least the same score.

The above steps can be easily extended to the case where the test video and query video do not have the same spatial dimensions. In that case, as proposed by Shechtman and Irani [7], we simply slide the query video template over all possible spatial-temporal locations (illustrated in figure 3), and compute a score for each space-time location using (8). This results in a action confidence volume, $C(n,m,i)$, which represents the score for the (n,m) location of the ith frame of the test video. A high value of $C(n,m,i)$ can then be interpreted as the query action being likely to be occurring at spatial location (n,m) in the ith frame.

While this exhaustive search seems to be computationally intensive, operating in the compressed domain allows for a real-time implementation.

E. Action video similarity score

Given $C(n,m,i)$, we can compute a non-symmetric similarity, $\rho(X^{\text{test}}, X^{\text{query}})$, of the test video to the query video by using:

$$\rho(X^{\text{test}}, X^{\text{query}}) = \frac{1}{L}\sum_{i=1}^{T_{\text{test}}} \eta(i)\left(\max_{n,m} C(n,m,i)\right) \tag{9}$$

where the normalization factor L is given by:

$$L = \sum_{i=1}^{T_{\text{test}}} \eta(i) \tag{10}$$

and $\eta(i)$ is an indicator function which returns one if at least T frames in the $(2T+1)$-length temporal neighborhood centered at frame i have significant motion and returns zero otherwise:

$$\eta(i) = \mathbb{I}\left[\sum_{j=i-T}^{i+T} \mathbb{I}\left[Q(j) \geq \delta\right] \geq T\right] \tag{11}$$

and the fraction of significant motion vectors in frame j, $Q(j)$, is given by:

$$Q(j) = \frac{\sum_{n=0}^{N^{\text{test}}-1}\sum_{m=0}^{M^{\text{test}}-1} \mathbb{I}\left[\|\vec{V}^{\text{test}}_{n,m}(j)\| > \epsilon\right]}{N^{\text{test}} \cdot M^{\text{test}}} \tag{12}$$

A frame is asserted to have significant motion if at least δ proportion of the macroblocks have reliable motion vectors (reliable in the sense defined in section II-A) of magnitude greater than ϵ, *i.e.* $Q(j) \geq \delta$.

III. Experimental results

We evaluate our proposed algorithm on a comprehensive database compiled by Schüldt *et al.* [8][2]. As illustrated in figure 4, their database captures 6 different actions (boxing, handclapping, handwaving, running, jogging and walking), performed by 25 people, over 4 different environments (outdoors, outdoors with scale variations, outdoors with different clothes and indoors). Since our system was not designed to handle scale-varying actions, we considered only the three environments that do not have significant scale variations.

Within each environment, we divide the action videos into a training set of videos performed by 16 subjects, and a test set of videos performed by 9 subjects. To classify each of the test video, we simply use nearest neighbor classification (NNC) by

[2]http://www.nada.kth.se/cvap/actions/

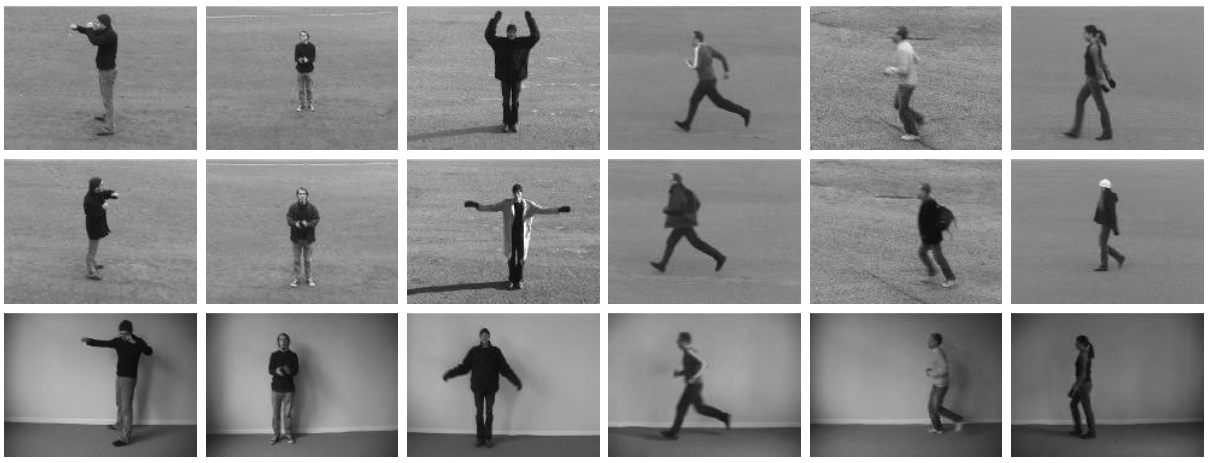


Fig. 4. Snap-shot of frames from action videos in database [8]. From left to right: boxing, handclapping, handwaving, running, jogging, walking. From top to bottom: outdoors environment, outdoors with different clothing environment, indoors environment. The subjects performing each action is the same across the different environments.

evaluating the action video similarity score (see section II-E with each of the videos in the training set.

In our experiments, we used $K = 9$, $\delta = \frac{1}{30}$, $\epsilon = 0.5$ pels/frame, $\alpha = 2.0$ and $T = 17$. For comparison, we also tested both NCNC (2) and NZMS (3) when computing frame-to-frame motion similarity.

A. Classification performance

The action classification confusion matrix for our algorithm when using NZMS is shown in table I, while that using NCNC [6] is shown in table II. Each entry of the matrix gives the fraction of videos of the action corresponding to its row that were classified as an action corresponding to the column. Our overall percentage of correct classification is 86%, which compares favorably to Schüldt *et al.* 's [8] best reported result (just under 80%) on the same data set.

Looking at the confusion matrices, we see that our proposed NZMS measure vastly outperforms NCNC. This is due to the fact that our measure looks at each corresponding pair of macroblocks separately instead of looking across all of them. NZMS also considers both differences in motion vector orientations and norms, and ignores matching zero-motion macroblocks.

Using NZMS, most of the confusion is between "Running" and "Jogging", with a significant proportion of "Jogging" videos being erroneously classified as "Running". Looking at the actual videos visually, we found it hard to distinguish between some "Running" and "Jogging" actions. There are cases where the speed of one subject's "Jogging" is faster than the speed of another subject's "Running"!!

B. Performance gain from thresholding optical flow confidence map

Table III shows the effects of thresholding on action classification performance using our proposed approach. By removing noisy estimates of the optical flow, we are able to achieve a 10% gain in classification performance.

TABLE I
CONFUSION MATRIX USING NZMS

	Box	Hc	Hw	Run	Jog	Walk
Boxing	0.82	0.11	0.00	0.00	0.00	0.07
Hand**c**lapping	0.01	0.95	0.04	0.00	0.00	0.00
Hand**w**aving	0.07	0.04	0.89	0.00	0.00	0.00
Running	0.00	0.00	0.00	0.91	0.00	0.09
Jogging	0.00	0.00	0.00	0.41	0.58	0.01
Walking	0.00	0.00	0.00	0.00	0.00	1.00

TABLE II
CONFUSION MATRIX USING NORMALIZED CORRELATION [6]

	Box	Hc	Hw	Run	Jog	Walk
Boxing	0.80	0.00	0.03	0.00	0.00	0.17
Hand**c**lapping	0.88	0.11	0.01	0.00	0.00	0.00
Hand**w**aving	0.09	0.00	0.87	0.00	0.00	0.00
Running	0.00	0.00	0.00	0.75	0.25	0.00
Jogging	0.01	0.00	0.00	0.01	0.98	0.00
Walking	0.00	0.00	0.00	0.55	0.00	0.45

C. Effect of α variation on classification performance

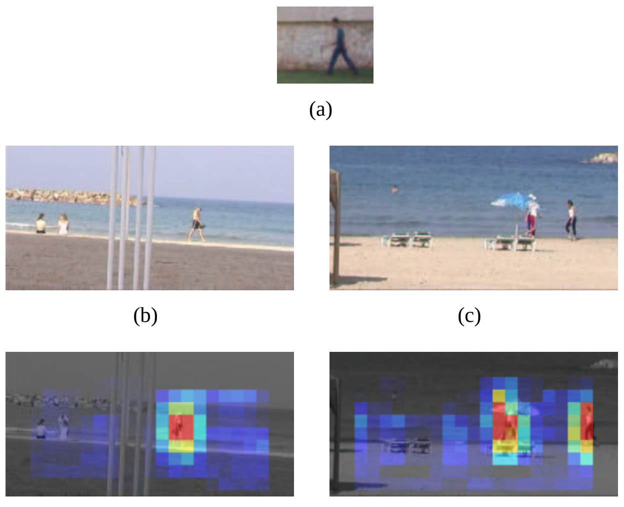
To understand the effect of α on classification, we ran an experiment using NZMS with varying values of α. Table IV shows the results of this experiment. We see that the classification performance is relatively stable over a range of α. More importantly, it is also clear that the aggregation step described in Section II-C is critical for action classification.

TABLE III
CLASSIFICATION PERFORMANCE WITH AND WITHOUT THRESHOLDING CONFIDENCE MAP

Method	With thresholding	Without thresholding
NZMS	85.2%	75.3%
NCNC	72.2%	71.6%

TABLE IV
CLASSIFICATION PERFORMANCE WITH VARYING α

α	Classification performance
1.0	83.3%
2.0	85.2%
3.0	84.6%
4.0	84.6%
No aggregation	62.3%



(d) (e)

Fig. 5. Localization Results. The false color in (d) and (e) denotes detection responses, with blue and red indicating a low and high response respectively. (a) A frame from the query video, (b) An input video frame with one person walking, (c) An input video frame with two people walking, (d) Detection of one person walking, (e) Detection of two people walking.

D. Localization performance

Unlike most other methods, with the notable exception of [7], we are able to localize an action in space and time and as well as detect multiple and simultaneous occurring activities in the test video. Figure 5 shows an example (the “beach” test sequence and walking query sequence from Shechtman and Irani [7]) which demonstrates our algorithm’s ability to detect multiple people walking in the test video.

E. Computational costs

On a Pentium-4 2.6 GHz machine with 1 GB of RAM, it took just under 11 seconds to process a test video of 368×184 pixels with 835 frames on a query video that is of 80×64 pixels with 23 frames. We extrapolated the timing reported in [7] to this case; it would have taken about 11 hours. If their multi-grid search was adopted, it would still have taken about 22 minutes. Our method is able to perform the localization, albeit with a coarser spatial resolution, up to *3 orders of magnitude faster*. On the database compiled in [8], each video has a spatial resolution of 160×120 pixels, and has an average of about 480 frames. For each environment, we would need to perform 22500 cross-comparisons. Yet, each run took an average of about 8 hours. In contrast, [7] would have taken an extrapolated run time of 3 years!

IV. EFFECTS OF VIDEO ENCODING OPTIONS

In the experiments described in the previous section, we have used input video compressed with MPEG [15], with a group-of-pictures (GOP) size of 15 frames, and a GOP structure of I-B-B-P-B-B-, where ‘I’ refers to an Intra-frame, ‘P’ refers to a Predicted-frame, and ‘B’ refers to a Bi-directionally predicted-frame. It would be interesting to see if there is any discernible difference when different encoding options, such as GOP size, GOP structure and the use of half-pel or quarter-pel motion estimation, are used. In addition, while MPEG uses 16×16 macroblocks as the basis of motion compensation, newer encoding standards such as H.263+ and H.264 allow the use of smaller block sizes [2], [3].

These experiments would be useful for a systems engineer in choosing a video encoder and its encoding options. While storage space and video quality are important considerations, it would be helpful to know if sacrificing a little compression performance would yield large gains in surveillance tasks such as action detection.

In the experiments below, we have used the publicly available “FFMPEG” video encoder[3]. When applicable, we will describe the encoder options, and specify the actual flags used with FFMPEG in parentheses. Unless otherwise mentioned, the encoding options used are that the MPEG-4 video codec is used (“-vcodec mpeg4”), the output video is of similar quality to the input video (“-sameq”), and the “AVI” container format is used.

A. GOP size and structure

We look at how varying GOP size and structure affects classification performance. We consider two commonly used GOP structure, I-B-B-P-B-B- (“-bf 2”) and I-P-P-P-P-P-. We also look at a variety of GOP sizes $\{9, 12, 15, 18, 30, 60, 120, 240\}$ (“-g [GOP size]”). By looking at how classification performance varies with compression performance, we can get an idea of what trade-offs are possible by varying GOP parameters when performing video encoding. In these experiments, the output video quality is kept relatively similar over all GOP size and structure.

It should be expected, and is in fact the case, that the larger the GOP size, the smaller the compressed videos, since predicted frames such as P and B frames can be more efficiently compressed than I frames. The results in Figure 6 further shows that in general, increasing GOP size also results in decreasing classification performance. This could be due to the fact that the update of the confidence measure computed as in Section II-A suffers from error propagation with each P frame. To test out this hypothesis, we also ran experiments where the confidence measure is computed from the *DCT of the actual decoded frame pixels* instead. Looking at the curve for the I-P-P-P-... GOP structure with no texture propagation error, we see that the classification accuracy is indeed fairly constant over a wide range of GOP size. This confirms that the main source of performance degradation with increasing GOP size is due to the propagation errors in computing the confidence measure.

[3]Available at http://ffmpeg.mplayerhq.hu/

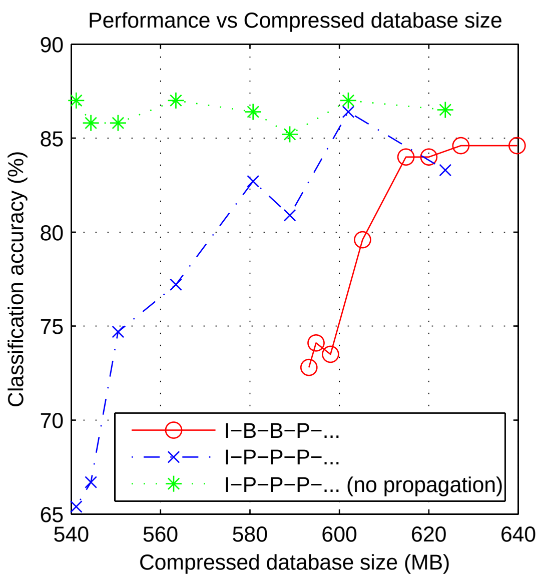


Fig. 6. Effect of varying GOP size on classification performance and compression performance. In general, increasing GOP size results in decreasing classification performance. Also, having no B frames in the GOP structure offers a better compression-classification trade-off. The fairly constant performance of the scheme using I-P-P-P-... with no texture propagation error indicates that the main source of performance degradation with increasing GOP size is due to propagation errors in computing block texture.

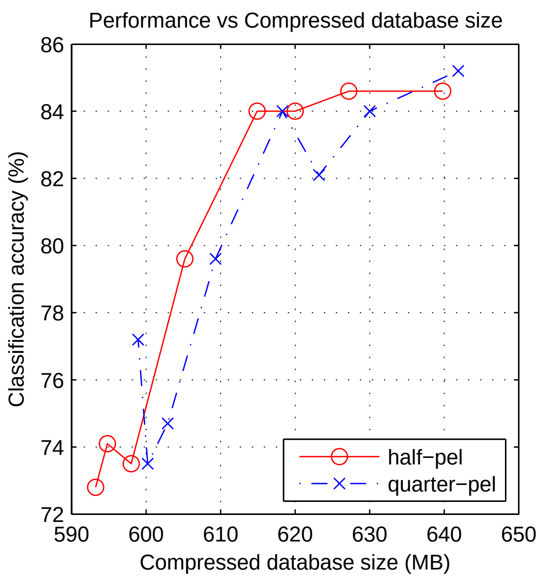


Fig. 7. Effect of quarter-pel accuracy motion estimation on classification performance and compression performance. There seems to be no significant improvement in the compression-classification trade-off by using motion estimation with quarter-pel accuracy instead of half-pel accuracy.

Figure 6 also shows that for the most part, the I-P-P-P-... GOP structure offers a better classification-compression trade-off than the I-B-B-P... GOP structure. There are two possible reasons for this. First, because of the complexity of articulated motion, B-frames are unable to provide any substantial compression gains over P-frames, while suffering from overhead. Hence, the I-B-B-P-... structure, for the same GOP size, actually performs worse in terms of compression performance. Second, the I-B-B-P-... structure introduces inaccuracy into the optical flow estimation process. The P frames are spaced 3 frames apart, and hence its estimated motion is actually over 3 temporal frames and not over 1 frame.

The experiments in this section seem to suggest that if action classification is an important factor in determining encoding options, then no B frames should be used in the encoding. This also has other advantages such as simpler encoders and decoders requiring less frame buffer memory. Further, if we used the confidence measure as computed in Section II-A, the GOP size should not be too large. A GOP size of 12, 15 or 18 seems to offer a good balance between compression and action classification. There might also be other factors in determining GOP size however, such as ease of random access.

B. *Quarter-pel accuracy motion estimation*

In MPEG, motion estimation was carried out to half-pel accuracy. It was found that better motion compensation is possible with a further increase in accuracy to quarter-pel [3], [16]. This motivates us to investigate if an increase in motion estimation accuracy (“-qpel 1”) would also translate into better action classification performance.

Figure 7 shows that using quarter-pel accuracy in motion estimation does not actually improve the classification-compression trade-off. There are two main reasons for this. First, we observe that on this set of action videos, for the same GOP size, using quarter-pel accuracy actually performs worse than half-pel accuracy in terms of compression performance. This could be due to the storage overhead of motion vectors with increased accuracy. Second, quarter-pel accuracy does not translate into better action recognition performance.

C. *Block size in motion compensation*

As mentioned earlier, newer encoding standards have the option of allowing smaller block sizes to be used in motion compensation [2], [3]. We compare the effect of forcing smaller blocks in motion compensation (“-mv4 1”) on both action classification performance and compression performance. In this set of experiments, we used a GOP structure of I-B-B-P-...

Figure 8 shows that using smaller blocks in motion compensation does result in a better performance vs compression trade-off. Smaller blocks allows for a more refined motion compensation and prediction, hence resulting in better compression performance. At the same time, with higher resolution motion vectors, action classification performance also improves. Of course, while using smaller blocks for motion compensation improves the trade-off, it has to be weighted by the increase in computation time. In our experiments, increasing the motion estimation resolution by 2 in each dimension resulted in about 5 times increase in run-time.

V. Conclusion

We have designed, implemented and tested a system for performing action recognition and localization by making use of compressed domain features such as motion vectors and DCT coefficients which can be obtained with minimal decoding. The low computational complexity of feature extraction

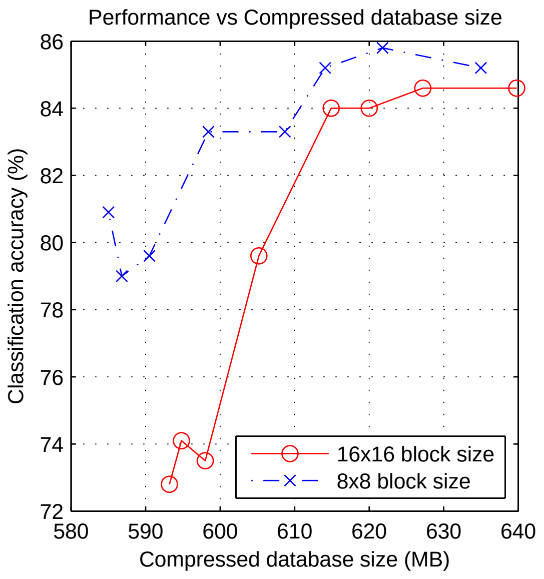


Fig. 8. Effect of using different block sizes in motion compensation on classification performance and compression performance. Using a smaller block size results in a better compression-classification trade-off, but this has to be weighed against the resulting increase in computational time.

and the inherent reduction in search space makes real-time operation feasible. We combined existing tools in a novel way in the compressed domain for this purpose and also proposed NZMS, a novel frame-to-frame motion similarity measure. Our results compare favorably with existing techniques [8] on a publicly available database.

Our experimental results provide justification for the engineering choices made in our approach. In particular, we showed the value of filtering motion vectors with low texture, and of aggregating frame-to-frame similarities. We also systematically investigated the effects of various encoding options on the action classification performance of our proposed approach. The results showed that for action videos, using a GOP structure with only P frames results in a better compression-classification trade-off. We also found that while a larger GOP size might result in a lower classification performance, it is mostly due to the effects of drift in computing block texturedness. Furthermore, quarter-pel accuracy in motion estimation does not appear to provide any benefits. While using smaller blocks in motion compensation does lead to better action classification and compression performance, the increased computational time of both encoding and action classification should be taken into account.

In future work, we plan to extend this to a multi-grid platform which would allow us to approach the spatial resolution of existing methods at a lower computational cost. For example, while we can encode video with smaller blocks in motion compensation, the algorithm can first perform action recognition at a coarser level, and then perform a finer level search in promising regions. We also plan to investigate the use of more sophisticated classifiers such as support vector machines (SVM) to improve our classification results. While our method is robust to small variations in scale, we would like to explore a truly scale-invariant approach in future work.

Another interesting angle to consider is the type of motion estimation used at the encoder. RD optimization is commonly performed in sophisticated video encoders to seek an optimum trade-off between compression and reconstruction quality [17]. It has also been used in the motion compensation process to reduce the rate used for coding motion vectors [18], [19]. This has the effect of smoothing the motion vector field which can be interpreted as a de-noising process. We hypothesize that this has a positive influence on the compression-classification trade-off, but this would have to be verified.

Acknowledgment

This work is supported in part by the National Science Foundation (NSF) under Grant No. CCR-0330514 and Army Research Office (ARO) Award W911NF-06-1-0076. Chuohao Yeo is funded by the Agency for Science, Technology and Research, Singapore (A*STAR). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or ARO or A*STAR.

References

[1] C. Yeo, P. Ahammad, K. Ramchandran, and S. S. Sastry, “Compressed domain real-time action recognition,” in *Proc. IEEE Workshop on Multimedial Signal Processing*, Victoria, BC, Canada, Oct. 2006.
[2] G. Cote, B. Erol, M. Gallant, and F. Kossentini, “H. 263+: video coding at low bit rates,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 8, no. 7, pp. 849–866, 1998.
[3] T. Wiegand, G. Sullivan, G. Bjntegaard, and A. Luthra, “Overview of the H. 264/AVC video coding standard,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, 2003.
[4] S.-F. Chang, “Compressed-domain techniques for image/video indexing and manipulation,” in *Proc. IEEE International Conference on Image Processing*, 1995, pp. 314–317.
[5] S. Wee, B. Shen, and J. Apostolopoulos, “Compressed-domain video processing,” Hewlett-Packard, Tech. Rep. HPL-2002-282, 2002.
[6] A. Efros, A. Berg, G. Mori, and J. Malik, “Recognizing action at a distance,” in *Proc. IEEE International Conference on Computer Vision*, Nice, France, Oct. 2003.
[7] E. Shechtman and M. Irani, “Space-time behavior based correlation,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, USA, Jun. 2005, pp. 405–412.
[8] C. Schüldt, I. Laptev, and B. Caputo, “Recognizing human actions: A local SVM approach.” in *Proc. International Conference on Pattern Recognition*, Cambridge, UK, Aug. 2004, pp. 32–36.
[9] B. Ozer, W. Wolf, and A. N. Akansu, “Human activity detection in MPEG sequences,” in *Proc. IEEE Workshop on Human Motion*, Austin, USA, Dec. 2000, pp. 61–66.
[10] R. V. Babu, B. Anantharaman, K. Ramakrishnan, and S. Srinivasan, “Compressed domain action classification using HMM,” *Pattern Recognition Letters*, vol. 23, no. 10, pp. 1203–1213, Aug. 2002.
[11] R. V. Babu and K. R. Ramakrishnan, “Compressed domain human motion recognition using motion history information,” in *Proc. IEEE International Conference on Image Processing*, Barcelona, Spain, Sep. 2003, pp. 321–324.
[12] J. Davis and A. Bobick, “The representation and recognition of action using temporal templates,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1997, pp. 928–934.
[13] M. T. Coimbra and M. Davies, “Approximating optical flow within the MPEG-2 compressed domain.” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 1, pp. 103–107, 2005.
[14] J. Barron, D. Fleet, and S. Beauchemin, “Performance of optical flow techniques,” *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43–77, 1994.
[15] D. Le Gall, “MPEG: a video compression standard for multimedia applications,” *Communications of the ACM*, vol. 34, no. 4, pp. 46–58, 1991.

[16] T. Wedi and H. Musmann, “Motion-and aliasing-compensated prediction for hybrid video coding,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 577–586, 2003.

[17] G. Sullivan and T. Wiegand, “Rate-distortion optimization for video compression,” *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, 1998.

[18] G. Sullivan and R. Baker, “Rate-Distortion Optimized Motion Compensation for Video Compression Using Fixed or Variable Size Blocks,” *Proceedings of the IEEE Global Telecommunications Conference*, vol. 3, pp. 85–90, 1991.

[19] M. Chen and A. Willson Jr, “Rate-Distortion Optimal Motion Estimation Algorithms for Motion-Compensated Transform Video Coding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 2, p. 147, 1998.