Histograms of Motion Gradients for Real-time Video Classification

Ionut C. Duta
University of Trento, Italy
ionutcosmin.duta@unitn.it

Jasper R.R. Uijlings
University of Edinburgh, UK
jrr.uijlings@ed.ac.uk

Tuan A. Nguyen
University of Tokyo, Japan
t_nguyen@hal.t.u-tokyo.ac.jp

Kiyoharu Aizawa
University of Tokyo, Japan
aizawa@hal.t.u-tokyo.ac.jp

Alexander G. Hauptmann
Carnegie Mellon University, USA
alex@cs.cmu.edu

Bogdan Ionescu
University Politehnica of Bucharest, Romania
bionescu@imag.pub.ro

Nicu Sebe
University of Trento, Italy
niculae.sebe@unitn.it

***Abstract*—Besides appearance information, the video contains temporal evolution, which represents an important and useful source of information about its content. Many video representation approaches are based on the motion information within the video. The common approach to extract the motion information is to compute the optical flow from the vertical and the horizontal temporal evolution of two consecutive frames. However, the computation of optical flow is very demanding in terms of computational cost, in many cases being the most significant processing step within the overall pipeline of the target video analysis application. In this work we propose a very efficient approach to capture the motion information within the video. Our method is based on a simple temporal and spatial derivation, which captures the changes between two consecutive frames. The proposed descriptor, Histograms of Motion Gradients (HMG), is validated on the UCF50 human action recognition dataset. Our HMG pipeline with several additional speed-ups is able to achieve real-time video processing and outperforms several well-known descriptors including descriptors based on the costly optical flow.**

***Index Terms*—Real-time Video Classification, Action Recognition, Histograms of Motion Gradients - HMG.**

I. Introduction

Over the recent years an explosive growth in video content has occurred and continues growing. As an example of this fulminant increase, Cisco forecast[1] mentioned that the IP video would account for 80% of all IP traffic by 2019. With this huge amount of multimedia content, computational efficiency has become as important as the accuracy of the techniques.

Even though in the past several years there has been an important progress in video analysis techniques, in particular on improving the accuracy of human action recognition in videos [27, 29, 20, 25, 16, 28], the current methods in terms of computational time are able to run with 1-3 frames per second. For instance in the work [25] is reported that the popular approach of [13] runs with 1.4 frames per second. Fast video analysis is important in many applications and this issue of efficiency became very important for large-scale video indexing systems or automatic clustering of large video collections.

[1]http://newsroom.cisco.com/press-release-content?articleId=1644203

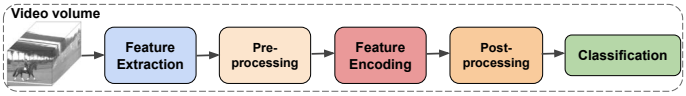

Fig. 1: The general pipeline for video classification.

In this work we propose a very efficient video representation which captures the video temporal motion information and is able to perform in real-time when dealing with video classification tasks. The Bag of Visual Words (BoVW) framework with its variations [14, 27, 29] has been widely used and showed its effectiveness in video analysis challenges. The BoVW pipeline represented in Fig.1, contains in general three main steps: feature extraction, feature encoding and classification. In addition to these main steps, the framework contains some pre/post processing techniques, such as PCA, feature decorrelation and normalization, which can influence considerably the performance of the pipeline. The commonly used approach for classification is employing a fast SVM classifier over the resulted video representations. The encoding step creates a final representation of the video and a very widely used approach is counting the frequency of the visual words. However, recently super-vector based encoding methods, such as Vector of Locally Aggregated Descriptors (VLAD) [9] and Fisher Vector (FV) [18], obtained state-of-the-art results for many tasks.

The video contains two important sources of information: the static information in the frames and the motion between frames. The feature extraction step focuses mainly on these two directions. The first direction has the goal to capture the appearance information in frames, such as Histogram of Oriented Gradients (HOG) [5, 14]. The other direction is based on optical flow fields like Histogram of Optical Flow (HOF) [14] and Motion Boundary Histograms (MBH) [6]. These descriptors are extracted and combined using Space Time Interests Points (STIP) [13], dense sampling [30, 25] or extracting the descriptors along some trajectories [22, 27, 29].

Temporal variation within the videos provides an important source of information about its content. Usually, the temporal information is computed with an optical flow method. There is a large number of approaches for extracting the optical flow fields,

978-1-4673-8695-1/16/$31.00

©2016 IEEE

from relatively old methods, such as [15, 8] to relatively recent approaches like [7, 3, 31, 4], which use complex algorithms to compute the motion information. The main drawback of those methods is the high computational cost to extract the motion information from the videos. This drawback becomes the bottleneck in many applications. For instance, the authors in [27] report that optical flow takes more than 50% of the total time for feature extraction. We present in this paper a new efficient descriptor, called Histograms of Motion Gradients (HMG), which is based on the motion information. The proposed HMG descriptor captures the motion information using a very fast temporal derivation, which enables us to have similar computational cost as HOG but with a significant improvement in accuracy.

The main contributions of this work can be summarized with the following:

- We introduce a new descriptor (HMG), which captures the motion information by using a simple temporal derivation, without the need of using the costly optical flow. We make available the code for descriptor extraction[2];
- We adopt several speed-ups, such as fast aggregation of gradients responses, reuse subregions of aggregated magnitude responses, and frame subsampling, which make the pipeline more efficient;
- We propose an integration of our descriptor in a specifically designed video classification framework which allows for real-time performance while maintaining the high accuracy of the results.

The rest of the paper is organized as follows. Section II introduces our new proposed descriptor with the adopted approaches for improving the efficiency. In Section III the experimental evaluation and the comparison with state-of-the-art are presented. Finally, Section IV concludes this work.

II. Proposed method

In this section we introduce the proposed method to capture motion information within the video. We present several speed-ups that make the framework very efficient, being able to achieve real-time processing.

A. Histograms of Motion Gradients

Our descriptor, Histograms of Motion Gradients (HMG), is based on a temporal derivation to compute the motion information. The illustration of the process of capturing the temporal information is presented in Fig. 2. For each two consecutive frames we first compute the temporal derivation:

$$T_{(i,i+1)} = \frac{\partial(F_i, F_{i+1})}{\partial t} \tag{1}$$

The temporal derivation is computed very effectively by applying a simple and fast filter of [1 -1] for each two consecutive frames (F_i, F_{i+1}). The result of this operation is illustrated in the middle image of Fig. 2, where we can observe that the information about the motion between two frames is kept. Obviously, after applying the temporal derivation some values

[2]http://disi.unitn.it/~duta/software.html

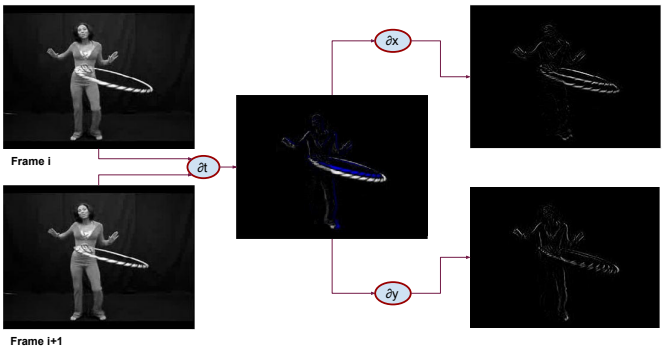

Fig. 2: Visualization of the process for capturing the motion information for HMG descriptor. The blue color represents the negative values after temporal derivation.

are negative, depending on the result of derivation between the pixels in frame i and frame $i+1$. For a better visualization, we represent the negative values with blue color for the middle image of Fig. 2.

After the computation of temporal derivative, we compute the spatial gradients of the resulted motion image, which allows us to compute the magnitude and the angle of the gradients responses. In the left part of Fig. 2 there are represented the horizontal and vertical gradients, computed with:

$$X_{(i,i+1)} = \frac{\partial T_{(i,i+1)}}{\partial x}, \quad Y_{(i,i+1)} = \frac{\partial T_{(i,i+1)}}{\partial y} \tag{2}$$

For the computation of spatial gradients we use also the simple and fast filter of [1 0 -1], similar as for HAAR-features. The gradients with this mask are computed much faster than, for instance, Gaussian derivatives. Basically, the gradients with this filter are obtained by making the difference between a frame and its shifted values with one position, once horizontally and once vertically. This makes the computation of gradients very efficient.

After we obtain the spatial derivatives, similar as for HOG, we compute the magnitude and the angle:

$$mag = \sqrt{X^2 + Y^2}, \quad \theta = \arctan\left(\frac{Y}{X}\right) \tag{3}$$

where each operation from the above formulas is element-wise.

The result of these operations is a 2-dimensional vector field per each new motion frame. We quantized the magnitude in 8 orientations. The next step is to perform the aggregation of those quantized responses over blocks in both spatial and temporal direction. Then we concatenate the responses over several adjacent blocks. We provide in the next subsection the details about the procedure of dividing the video in blocks and volumes. Afterwords, the pipeline in Fig. 1 continues with the next step by applying some pre-processing operations before feature encoding, such as normalization and PCA with decorrelation of features.

B. Speed-up HMG extraction

For our proposed descriptor we use a dense sampling strategy to extract the features. In addition to the presented approach for capturing the motion information very efficiently and using fast filters for derivatives, we describe several speed-ups that

improve the efficiency of the descriptor extraction process of HMG. The efficiency improvement is performed by taking the advantage of the densely sampled approach and by adopting to our new descriptor several speed-ups presented in [25].

1) Reuse of blocks: Our choice to establish the region of the descriptor extraction is the use of dense sampling since this method has a big potential for efficiency. It can be also easily extended to an even faster version using parallelization. Furthermore, in several works, it has been found to be more accurate than keypoint-based sampling in images [10] and videos [30, 17]. We take advantage of the densely sampled descriptor nature in order to speed up the feature extraction time. Fig. 3 illustrates an example for dividing the video into blocks, and how a volume is created of several adjacent blocks. Our HMG descriptor is extracted on a single scale over each block, which consists of 8 by 8 pixels by 6 frames. The size of the blocks is also our dense sampling rate. The green part from the Fig. 3 represents a video volume, where the responses over several adjacent blocks are concatenated for creating the final descriptor. Each video volume consists of 3 by 3 by 2 blocks, corresponding to x, y and t axis. By choosing the sampling rate equally with the block size, then we can reuse the blocks for making the descriptor extraction efficient. For instance, each block can be reused for 18 times (excepting the blocks on the borders) for the current size of the video volume: 3 by 3 by 2 blocks.

2) Fast aggregation of responses: After we compute the magnitude and the angle, the resulted responses are aggregated for each block. We adopted the approach of [23]. Basically we compute the aggregation of all the frame pixels by doing just a multiplication of three matrices. After the spatial aggregation of 8 by 8 pixels and the temporal aggregation of 6 frames, each block is characterized by 8 values as we consider 8 orientations for quantization of responses. Having 8 bins and a size of 3 by 3 by 2 for video volume, the original dimensionality of our descriptor is therefore 144.

3) Frame subsampling: For efficiency reasons we evaluate HMG by subsampling video frames. Subsequent frames contain redundant information, and the computational cost can be substantially improved by frame subsampling. We evaluate the impact on the accuracy and efficiency of our descriptor by skipping frames. A detailed analysis of the trade-off between accuracy and computational time is presented with the experimental results.

III. Experimental Evaluation

The general pipeline used for evaluation is the one presented in Fig. 1. For evaluation of our proposed HMG descriptor the baseline is to use dense sampling with 8 by 8 pixels by 6 frames as in [25] and the gradient magnitude quantized in 8 orientations. The final descriptor is a concatenation of 3 by 3 by 2 blocks. For the pre-processing step of we perform RootSIFT [1] normalization and then we apply PCA to reduce the dimensionality by a factor of two and decorrelate the features. This yields a final descriptor dimension of 72. We use spatial pyramid in all our experiments, we divide all the frames of the video into three horizontal parts which intuitively roughly correspond to a ground, object, and sky devision.

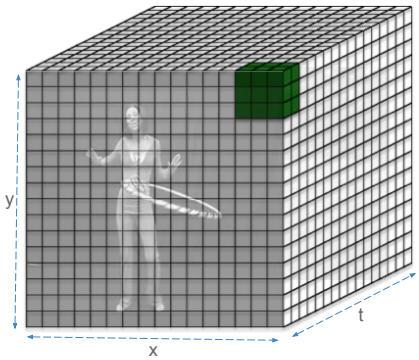

Fig. 3: The process of dividing the video in blocks and volumes. The green part represents an illustration of a volume created from 3 by 3 by 2 blocks.

The codebook for each experiment needed for feature encoding is built from randomly sampled 500K features of the training set. For the resulted vectors after descriptor encoding we apply power normalization followed by L2 for the super-vector based encoding methods and power normalization followed by L1 for all other visual word assignment methods. The parameter α for power normalization is initially fixed to 0.5. We perform the classification with SVMs, with a linear kernel for super-vector based encoding methods and histogram intersection kernel for all other encoding methods, with $C = 100$.

We initially compare our descriptor with dense HOG, HOF, MBHx and MBHy using the available code of [24, 25]. For these descriptors we use the same settings and speed-ups as presented for HMG, see Section II. The optical flow for HOF, MBHx and MBHy is computed with Horn-Schunk method [8] using Matlab Computer Vision System Toolbox as the work [25] recommends this approach as a good trade-off between accuracy and computational cost. The timing measurements are performed on a single core Intel(R) Xeon(R) CPU E5-2690 2.60GHz, using 500 randomly sampled videos from the dataset (10 videos for each class), which is presented in the next section. We report the average of the number of seconds per video and the number of frames per second that the system can process.

A. Dataset

We perform the evaluation on the challenging UCF50 Human Action Recognition dataset [19]. In total, this dataset contains 6,618 realistic videos taken from YouTube with 50 human action categories mutually exclusive, which range from general sports to daily life exercises. For all categories, the videos are split into 25 predefined groups. We perform leave-one-group-out cross validation as recommended by [19], and report average classification accuracy over all 25 folds.

B. Comparison to dense descriptors

In this part we present a first comparison between the proposed HMG descriptor and the most popular descriptors for action recognition: HOG, HOF, MBHx and MBHy [25]. The comparison is conducted in terms of accuracy and computational cost. All the descriptors benefit of the same settings and the same speed-up approaches presented above for HMG descriptor. All dense descriptors are extracted using only the intensity information. All the computational time measurements for descriptor

descriptor	HOG	HOF	MBHx	MBHy	**HMG**
Acc	0.762	0.799	0.784	0.792	**0.814**
sec/video	2.67	4.03	4.37	4.37	2.73
frame/sec	73.50	48.61	44.80	44.84	71.73

TABLE I: Comparison to dense descriptors.

computation include also the loading time of the video and converting the frames to graylevel. For this set of experiments we use Fisher Vector (FV) [18] as encoding method, with the common setting of 256 clusters.

The comparative results are presented in Table I. Our approach of computing the motion information by applying a simple and efficient temporal filter does not affect significantly the computational cost comparably with the fast HOG descriptor. While the efficiency is kept, in terms of accuracy our HMG descriptor outperforms with a large margin HOG, by 5.2 percentage points. This significant improvement of performance while preserving the efficiency shows that the motion information captured by our descriptor is very discriminative for videos and can be considered as an option for the applications based on video analysis, especially for those where the computational cost is crucially important. Remarkably, HMG outperforms even descriptors based on classical optical flow which are more demanding for computational cost. For instance, HMG outperforms HOF by 1.5 percentage points in terms of accuracy, moreover, the descriptor extraction for HMG runs with approximately 72 frames/second while HOF runs only at around 49 frames/second. This big difference in efficiency is due to the optical flow computation, which can take up to 50% of the cost for HOF extraction.

C. Feature Encoding

After descriptor extraction, the feature encoding is another important step for the system performance, and at the same time, demanding in terms of computational cost. There are different approaches for feature encoding with important differences in accuracy and efficiency. Choosing the proper encoding method is another key factor which can influence substantially the final result. As we are going to present a real-time video classification pipeline, we investigate the proper encoding method for a trade-off between accuracy and computational time.

In this part we compare our dense HSM descriptor with dense HOG, HOF, MBHx and MBHy for Bag-of-Visual-Words (BoVW) using three approaches for visual word assignment: k-means, hierarchical k-means (hk-means) and Random Forests (RF) [2]. In addition we use other two variations of BoVW: Fisher Vectors (FV) [18] and Vector of Locally Aggregated Descriptors (VLAD) [9]. For k-means and hk-means we use the implementation made available of VLFeat [26]. For both we create a codebook of 4096 visual words. For hk-means we learn a hierarchical tree of depth 2 with 64 branches per node. RF are well-known for their speed, they are binary decision trees, learned in a supervised manner by randomly picking several descriptor dimensions at each node with several random thresholds. The split with the highest Entropy Gain is selected. We follow the recommendations of [23, 24, 25], using 4 binary decision trees of depth 10, which create a codebook of 4096 visual words.

	k-means	hk-means	RF	FV	VLAD
HOG	0.731	0.720	0.718	0.762	0.732
HOF	0.789	0.779	0.738	0.799	0.818
MBHx	0.772	0.760	0.731	0.784	0.787
MBHy	0.783	0.774	0.750	0.792	0.798
HMG	0.781	0.759	0.735	0.814	0.810
sec/video	8.42	0.37	0.05	2.09	0.25
frame/sec	24	526	3788	94	794

TABLE II: Trade-off accuracy/efficiency for different visual word assignment methods.

For FV we keep the codebook size of 256 clusters. We tested VLAD representation with 256 and 512 visual words, however, we report in this paper the version with the codebook size of 512 as this version obtains a significant improvement in our experiments. Furthermore, for this codebook size the resulted final vector has equal dimensionality with FV in the presented case of 256 clusters. Different from the traditional approach, instead of sum pooling for VLAD encoding method we perform average pooling. This is a simple technique to cancel the big influence of the most frequent visual words and from our results this has a consistent positive influence on the accuracy. We perform the assignment step for VLAD using dot product to compute the similarity between two vectors instead of euclidian distance, this improves considerably the computational efficiency for the assignment step.

The results for different encoding methods are presented in Table II, which confirm that super-vector encoding methods give a better video representation than the other encoding approaches. The superiority of super-vector encoding methods is due to the fact that it captures information related to the mean and variance of the features and not only the membership information of the features to the clusters. Our HMG descriptor is very competitive for all the encoding methods, especially for super-vector encoding methods, which outperforms all the other descriptors with 0.814 accuracy for FV.

The computational cost for the encoding step is not dependent on the type of features, it depends on the number of visual words and the dimensionality of descriptors. As all our descriptors have the same dimensionality, we reported the computational cost for encoding a descriptor (can be any) with 72 dimensions. The RF approach for encoding step is by far the fastest and takes 0.05 per video, however, the accuracy drops significantly for all descriptors related to the best encoding method for accuracy. The results for hk-means represents a good trade-off between accuracy and computational efficiency, can process the video at a frame rate of 526. When the speed is crucial important then RF is the best choice.

After these experiments we can take the conclusion that super-vector based encoding methods give the best performance. For feature encoding, VLAD represents the best trade-off between accuracy and computational efficiency, running at a frame rate of 794. Considering this, for the further experiments we will consider only FV and VLAD as encoding methods.

D. Normalization

After feature encoding, for the resulted vector of the video representation we apply before classification Power Normalization followed by L2-normalization ($||sign(x)|x|^{\alpha}||_2$, where

	$\binom{\text{frames/block}}{\text{sample rate}}$	$\binom{6}{1}$	$\binom{3}{2}$	$\binom{2}{3}$	$\binom{1}{6}$
HOG	VLAD	0.750	0.751	0.751	0.763
	FV	0.820	0.817	0.814	0.820
	sec/video	2.67	1.54	1.15	0.78
	frame/sec†	73.50	127.07	170.99	250.79
HOF	VLAD	0.824	0.817	0.805	0.784
	FV	0.834	0.820	0.817	0.799
	sec/video	4.03	2.28	1.68	1.06
	frame/sec†	48.61	86.04	116.27	184.73
MBHx	VLAD	0.797	0.793	0.791	0.772
	FV	0.816	0.806	0.797	0.779
	sec/video	4.37	2.45	1.80	1.12
	frame/sec†	44.80	80.03	108.96	174.60
MBHy	VLAD	0.808	0.804	0.803	0.785
	FV	0.824	0.819	0.814	0.794
	sec/video	4.37	2.44	1.80	1.12
	frame/sec†	44.84	80.27	108.67	174.37
HMG	VLAD	0.821	0.813	0.820	0.818
	FV	0.850	0.845	0.843	0.829
	sec/video	2.73	1.59	1.19	0.80
	frame/sec†	71.73	123.47	164.17	245.45

TABLE III: Trade-off between frame sampling rate and accuracy. We keep video volumes from which descriptors are extracted the same for all sampling rates. †Frames/second is measured in terms of the total number of frames of the original video, not in terms of how many frames are actually processed during descriptor extraction.

$0 \leq \alpha \leq 1$ is the normalization parameter), we call this PNL2. The effect of this normalization is reducing the peaks within the vector. We perform the α parameter tuning with the step 0.1 and the conclusion is that a very small α improves considerably the accuracy. The best results are obtained with $\alpha = 0.1$ and the difference to the previous results can be noticed in the first column with results of the Table III, where for instance, the accuracy of HMG with FV increases from 0.814 to 0.850.

E. Frame subsampling

Subsequent video frames contain similar information. In this set of experiments we investigate the impact on the accuracy results when frames are skipped, with the goal of speeding up the feature extraction process. We evaluate when skipping 2, 3 and 6 frames. The modality of frame subsampling is similar as in the work [25]. For a fair comparison, the descriptors describe the same video volume for the process of subsampling frames. For instance, if we sample every 2 frames, our baseline for the size of the block of 8 by 8 pixels by 6 frames is changing to 8 by 8 by 3 frames; for skipping 3 frames we have only 8 by 8 pixels by 2 frames; and when sampling every 6 frames the block size became 8 by 8 pixels by 1 frame. The results for frame subsampling with PNL2 are presented in Table III.

By subsampling frames the computational cost is significantly improved, making the pipeline more efficient. HOG descriptor is not negatively affected by skipping frames because this descriptor capture the appearance information and subsequent video frames contain similar information. Therefore, for HOG descriptor we can skip frames with a step of 6 without loosing accuracy for both FV and VLAD, being able to process more than 250 frames per second. For the descriptors based on optical flow a frame sampling rate of 3 gives a good trade-off, improving considerably the computational cost. HMG with VLAD can have a frame sampling rate of 6 without decreasing significantly the accuracy.

	HOG	HOF	MBH	HMG	sec/video	frames/sec
IDT [29]	0.826	0.851	0.889	-	50.5	3.9
dense	0.820	0.834	0.832	0.850	10.9	18.0

TABLE IV: Comparison to IDT [29] in terms of accuracy and computational cost.

F. Comparison with Improved Trajectories approach

The Improved Dense Trajectories (IDT) [29] represents a state-of-the-art video representation approach. We compare our approach with IDT in terms accuracy and of computational efficiency without subsampling frames. As in [29] there are reported the results for FV with 256 clusters, we perform the comparison with our dense approach using the same encoding method. As the code of [29] provides four main descriptors (HOG, HOF, MBHx and MBHy), for a fair comparison we compare its extraction time with dense extraction time also for four descriptors: HOF, MBHx, MBHy and HMG. Notice that dense HOG and HMG has similar computational time, so it is not relevant for time measurement which is chosen. The comparison with IDT is presented in Table IV. For the computational efficiency the dense approach outperforms by a large margin IDT, being 4.6 times faster. Dense approach is able to process a video with 18 frames per second while IDT can process only 3.9 frames per second. Even though [29] provides a fast code in C++, the Matlab implementation for dense descriptors is considerably less demanding for the computational cost due to several factors. First, IDT uses a more complicate algorithm to extract the descriptors and furthermore, their approach improves the accuracy by canceling the camera motion. For doing this it is necessary to compute two times the optical flow, which makes the algorithm more demanding for computational efficiency. Another reason is that the dense descriptors are computed more efficiently, being able to reuse the blocks for many times and without the need to compute any trajectories. Very interesting that our HMG descriptor is able to compete even with HOF from IDT, with almost similar performance of 0.85.

G. Real-time video classification

For a real-time video classification system we recommend two frameworks. The first framework is using FV (with 256 clusters) for feature encoding and performing early fusion by concatenating HOG (with sampling rate of 6) and HMG (without subsampling frames). Applying a linear SVM is very fast, for instance to get the predicted class for a video the computational cost is less then 0.001 seconds, this means that the classification time is negligible compared to other steps along the pipeline. This first framework obtains an accuracy of 0.854 and is able to process the video in real-time at 27 frames per second.

The second pipeline for real-time video classification that we recommend is using VLAD (with 512 clusters) for encoding step and is doing early fusion (before applying linear SVM) of HOG, HMG (both with frame sampling rate of 6), HOF, MBHx add MBHy. To compute optical flow only once, the frame sampling rate should be equal for all descriptors based on optical flow, frame sampling rate of 3 is a good trade-off for all

Method	Accuracy
Kliper-Gross et al. [11] (2012)	0.727
Solmaz et al. [21] (2012)	0.737
Reddy et al. [19] (2012)	0.769
Uijlings et al. [25] (2014)	0.818
Wang et al. [27] (2013)	0.856
Wang et al. [29] (2013)	0.912
Wang et al. [28] (2015)	0.917
Peng et al. [17] (2014)	0.923
HMG + IDT	**0.930%**

TABLE V: Comparison with the state-of-the-art.

these descriptors. This second real-time framework obtains an accuracy of 0.842 and can process a video with an appreciable efficiency of 38 frames per second.

H. Comparison to state-of-the-art

When accuracy is crucially important for the application we recommend using FV (with 256 clusters) for feature encoding and combining our HMG descriptor with IDT descriptors. We extract all the descriptors of IDT (HOG, HOF, MBHx and MBHy) with the default settings provided in [29]. We perform early fusion between HMG and IDT by concatenating all features. For all features we apply separately before early fusion PNL2 normalization with $\alpha = 0.1$. This combination improves the accuracy from 0.912 reported in [29] to 0.930. This significant improvement of performance shows that our HMG descriptor brings complementary information for IDT and can be used to boost the performance of the system. Table V presents the comparison with state-of-the-art approaches. The proposed combination for accuracy between HMG and IDT obtains state-of the art results outperforming including the recent work of [28] which considers as encoding method the spatial FV [12] together with spatio-temporal pyramid [14]. Our results are better than [28] which considers a hybrid representation by combining two different representations.

IV. Conclusion

We introduce in this work a new descriptor, HMG (Histograms of Motion Gradients), that captures motion information without the need of computing optical flow, which obtains very competitive results while achieving a low computational complexity. Tested on UCF50 dataset, our descriptor is able to outperform including well-known descriptors based on the expensive computation of optical flow. In the end, after several evaluations of the trade-off between accuracy and computational efficiency we propose two frameworks which can be used for real-time video classification. For the future work we will focus on further improvement of computational efficiency by using parallel computation. Furthermore, for new features we consider the idea of learning a new representation by training a convolutional neural network with the resulted motion frames obtained after temporal derivation.

V. Acknowledgments

This work was partially supported by Erasmus Mundus Program TEAM for Information and Communication Technologies and by the FP7 EU project xLiMe. This material is based in part on work supported by the National Science Foundation (NSF) under grant number IIS-1251187.

References

[1] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.
[2] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
[3] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*. 2004.
[4] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *TPAMI*, 33(3):500–513, 2011.
[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.
[6] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *Computer Vision–ECCV 2006*, pages 428–441. Springer, 2006.
[7] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image analysis*, pages 363–370. Springer, 2003.
[8] B. K. Horn and B. G. Schunck. Determining optical flow. In *1981 Technical symposium east*, pages 319–331. International Society for Optics and Photonics, 1981.
[9] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *TPAMI*, 34(9):1704–1716, 2012.
[10] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *ICCV*, 2005.
[11] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion interchange patterns for action recognition in unconstrained videos. In *ECCV*, 2012.
[12] J. Krapac, J. Verbeek, and F. Jurie. Modeling spatial layout with fisher vectors for image categorization. In *ICCV)*, 2011.
[13] I. Laptev. On space-time interest points. *IJCV*, 64(2-3):107–123, 2005.
[14] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.
[15] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.
[16] I. Mironică, I. C. Duță, B. Ionescu, and N. Sebe. A modified vector of locally aggregated descriptors approach for fast video classification. *Multimedia Tools and Applications*, pages 1–28, 2015.
[17] X. Peng, L. Wang, X. Wang, and Y. Qiao. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *arXiv:1405.4506*, 2014.
[18] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*. 2010.
[19] K. K. Reddy and M. Shah. Recognizing 50 human action categories of web videos. *Machine Vision and Applications*, 24(5):971–981, 2013.
[20] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.
[21] B. Solmaz, S. M. Assari, and M. Shah. Classifying web videos using a global video descriptor. *Machine vision and applications*, 24(7):1473–1485, 2012.
[22] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. In *CVPR*, 2009.
[23] J. R. Uijlings, A. W. Smeulders, and R. J. Scha. Real-time visual concept classification. *TMR*, 12(7):665–681, 2010.
[24] J. R. R. Uijlings, I. C. Duta, N. Rostamzadeh, and N. Sebe. Realtime video classification using dense hof/hog. In *ICMR*, 2014.
[25] J. R. R. Uijlings, I. C. Duta, E. Sangineto, and N. Sebe. Video classification with densely extracted hog/hof/mbh features: an evaluation of the accuracy/computational efficiency trade-off. *International Journal of Multimedia Information Retrieval*, 4(1):33–44, 2014.
[26] A. Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *ACM Multimedia*, 2010.
[27] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, 103(1):60–79, 2013.
[28] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. *IJCV*, 2015.
[29] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.
[30] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009.
[31] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Pattern Recognition*, pages 214–223. Springer, 2007.