A Video Tensor Self-descriptor Based on Block Matching

Ana M. O. Figueiredo[1], Helena A. Maia[1], Fábio L. M. Oliveira[1], Virgínia F. Mota[2], and Marcelo B. Vieira[1]

[1] Universidade Federal de Juiz de Fora, Juiz de Fora, Brazil
{anamara,helena.maia,fabio,marcelo.bernardes}@ice.ufjf.br
[2] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
virginiaferm@dcc.ufmg.br

Abstract. In this paper, we propose a new motion descriptor which uses only block matching vectors. This is a different and simple approach considering that most works on the field are based on the gradient of image intensities. The block matching method returns displacements vectors as a motion information. Our method computes this information to obtain orientation tensors and to generate the final descriptor. It is considered a self-descriptor, since it depends only on the input video. The global tensor descriptor is evaluated by a classification of KTH, UCF11 and Hollywood2 video datasets with a non-linear SVM classifier. Our results indicate that the method runs fast and has fairly competitive results compared to similar approaches. It is suitable when the time response is a major application issue. It also generates compact descriptors which are desirable to describe large datasets.

Keywords: self-descriptor, compact descriptor, block matching, human action recognition.

1 Introduction

Human action recognition has been extensively researched over the past years due to its application in many areas such as: video indexing, surveillance, human-computer interfaces, among others. In order to approach this problem, many authors have proposed video descriptors using motion representation, which is one of the main characteristics that describes the semantic information of videos.

Among the methods to detect motion, block matching is used to find vectors that indicate block displacements between two video frames. We chose this technique as it has not been extensively applied to human action recognition and several works on literature use block displacement vectors for other applications, for example [1–3]. Moreover, this method runs fast and can potentially generate compact descriptors since it is widely used in video compression.

This work is motivated by the possibility of generating a compact and easy to compute descriptor. Our main contribution is a new motion descriptor, based on orientation tensor [4], which uses only block matching information. This is

B. Murgante et al. (Eds.): ICCSA 2014, Part VI, LNCS 8584, pp. 401–414, 2014.

a different approach, considering that most works on the field are based on the gradient of image intensities [5, 6]. The global tensor descriptor created is evaluated by a classification of KTH, UCF11 and Hollywood2 video datasets with a non-linear Support Vector Machine (SVM) classifier.

We present three variants of our method. The first, called Single Scale Single Vector (SSSV), is the simplest and fastest. It has the same elements as the block matching method: one fixed block size and one vector field generated for each pair of frames. The second, called Multiple Scales Single Vector (MSSV), yields better results than the first, at the cost of slower execution speed. Since it considers more than one block size, it requires multiple computations for each pair of frames. The third version, called Multiple Scales Multiple Vectors (MSMV), yields even better recognition rates but is also even slower. It considers multiple block sizes and also goes through more than two consecutive frames.

2 Related Works

Several works in literature use the motion intensity obtained from block matching in many applications. Hafiane et al [1] presents a method for video registration based on Prominent Feature (PF) blocks. Block matching was used to identify moving objects in a video. Structured tensors sensitive to edge and corners were used to extract a point of interest in each block. In order to find region correspondences between images, block matching was used along with Normalized Cross-Correlation (NCC) or Sum of Absolute Differences (SAD) as an error estimate. NCC is less sensitive to absolute intensity changes between the reference and target images due to normalization, but is much more expensive to compute than SAD. In this work, we employ SAD as an error function and the Four Step Search (4SS) as a fast search strategy since it is computationally more efficient than Full Search (FS). As another less costly alternative for handling intensity outliers, we apply a smoothing filter on each frame, so that SAD obtains quality results.

Similar to [1], a block matching method was used for extracting motion information in [2]. However, this information was used to generate a motion activity descriptor for shot boundary detection in video sequences. The chosen method for quickly computing the motion vectors was Adaptive Rood Pattern Search (ARPS). These vectors were used to calculate the intensity of motion and also classify among the categories presented by the authors. Vectors with higher values indicate a greater probability of being a shot.

An activity descriptor, consisting of a temporal descriptor and a spatial descriptor, is presented in [3]. The temporal descriptor is obtained through the ratios between moving blocks and all the blocks on each frame. In order to be labelled as a moving block, the error must be within a margin of tolerance. These ratios are then adjusted into quantized levels. The spatial descriptor, also used in [2], is obtained through a matrix containing all the motion vectors norms from each frame.

Other video descriptors were proposed using different methods for extracting information, such as the human action descriptor shown in [4–8]. Klaser et

al [9] propose a local feature based descriptor for video sequences generalizing Histogram of Oriented Gradient (HOG) concepts to 3D. In [7], they extend the Features from Accelerated Segment Test (FAST) method for the 3D space. The information of shape and motion was obtained detecting spatial and temporal features. Following [9], they produced a descriptor based on HOG3D which describes corner features. Both use KTH and Hollywood2 databases to evaluate performance. We also use these databases to evaluate our descriptor, but we generate a global descriptor using information from Motion Estimation (ME).

Mota et al [10] presented a tensor motion descriptor for video sequences using optical flow and HOG3D information. They use an aggregation tensor based technique. This technique combines two descriptors, one includes polynomial coefficients which approximate optical flow, and the other accumulates data from HOGs. This descriptor is evaluated by a SVM classifier using KTH, UCF11 and Hollywood2 datasets. In our work, the approach of using block matching vectors reduces considerably the effort of tracking motion as compared to the use of HOG3D. Moreover, the bidimensional nature of block displacements reduces significantly the size of the histogram coded into a tensor. Compared to [10], our descriptor is more compact and easier to compute, while still yielding competitive results.

3 Block Matching Descriptor: Single Scale Single Vector

There are several block matching methods which can be used to extract motion information. The simplest is the Full Search. This method searches for each 16×16 block from the reference (current) frame in the target (next) frame. The corresponding, or matching, block is the one which minimizes a cost function such as SAD or MAD (Mean Absolute Difference), representing high similarity between blocks. The search window on the target consists of all the possible blocks differing from -7 to 7 pixels in both directions from the reference frame block. Thus, all the 225 neighbouring blocks are evaluated. Although it is a precise method, it is computationally expensive. Therefore, several fast methods were proposed such as 4SS [11], ARPS [12] and DS (Diamond Search) [13], based on steepest descent methods.

The 4SS consists of four steps with three distinct search patterns. In the first step, it checks nine points in a 5×5 window. The point referring to the block with the lowest Block Distortion Measure (BDM) becomes the center of the search window in the following step. Whenever the minimum BDM is at the center window point, the algorithm proceeds to the fourth step. In the second and third steps, it checks five or three blocks depending on whether the previous step results on a corner or a side point, respectively. The last step consists on checking eight points in a 3×3 window. In the worst case scenario, 27 blocks are evaluated.

The DS is fairly similar to 4SS. In both, the first step checks nine points and the following steps check three or five points. However, DS uses a diamond shaped search window and instead of having four steps, it repeats the second

step until the lowest BDM is found at the center of the pattern or it reaches an iteration limit. It then proceeds to the third step for the final four BDM evaluations.

In this work, we use the displacement map generated by the 4SS algorithm. The input is a video, i.e., a set of frames $V = \{F_k\}$, where $k \in [1, n_\mathrm{f}]$, n_f is the number of frames and the output is a tensor descriptor $\mathbf{T} \in \mathbb{R}^{n \times n}$ which, in fact, can be viewed as a vector in $\mathbb{R}^m$, $m = n^2$. Our descriptor is considered a self-descriptor, since it depends only on the input video. It is computed by extracting and accumulating information from each frame of the video. Basically, the frame is divided into blocks and their displacement vectors are computed. As proposed in [10], this vector field is represented by a histogram which is encoded into an orientation tensor. The tensors of each frame are accumulated to form the final tensor descriptor $\mathbf{T}$ of the video. This first method is simpler and faster than the following methods.

3.1 Computing the Motion Estimation Histogram

In the 4SS schemes, each frame k of the video is subdivided into $n_\mathrm{x} \times n_\mathrm{y}$ non-overlapping blocks of exactly $s \times s$ pixels. Thus, our first method (SSSV) is constrained to use square sub-images of the frame. If the frame dimension is not a multiple of s, the remaining right and bottom pixels do not form blocks. For each block, displacement vectors $\boldsymbol{v}_k(i,j) = (x,y) \in \mathbb{R}^2$ are calculated, where $(i,j) \in [1, n_\mathrm{x}] \times [1, n_\mathrm{y}]$ are the block indexes. These vectors are converted to equivalent polar coordinates $\boldsymbol{c}_k(i,j) = (\theta, r)$ with $\theta = \tan^{-1}(\frac{y}{x})$, $\theta \in [0, 2\pi]$ and $r = \| \boldsymbol{v}_k(i,j) \|$.

A motion estimation histogram is used as a compact representation of the motion vector field obtained from each frame. It is defined as the column vector $\boldsymbol{h}_k = (h_1, h_2, \ldots, h_{n_\theta})^T$, where n_θ is the number of cells for the θ coordinate. We use an uniform subdivision of the angle intervals. Each interval is populated as the following equation:

$$h_l = \sum_{i,j} r(i,j) \cdot \omega(i,j) \ , \tag{1}$$

where $l = 1, 2, \ldots, n_\theta$ and $\omega(i,j)$ is a vector weighting factor, which is a Gaussian function with $\sigma = 0.01$ in our experiments. The whole frame vector field is thus represented by a vector $\boldsymbol{h}_k$ with n_θ elements.

3.2 Tensor Descriptor: Coding the Motion Estimation Histogram

An orientation tensor is a representation of local orientation which takes the form of a $n \times n$ real symmetric matrix for n-dimensional signals [14]. Given a vector $\boldsymbol{v} \in \mathbb{R}^n$, it can be represented by the tensor $\mathbf{T} = \boldsymbol{v}\boldsymbol{v}^\mathrm{T}$. Then, we use the orientation tensor to represent the histogram $\boldsymbol{h}_k \in \mathbb{R}^{n_\theta}$. The frame tensor for the size s, $\mathbf{T}_k(s) \in \mathbb{R}^{n_\theta \times n_\theta}$, is given by:

$$\mathbf{T}_k(s) = \boldsymbol{h}_k \cdot \boldsymbol{h}_k^\mathrm{T} \ . \tag{2}$$

Individually, these frame tensors have the same information as $\boldsymbol{h}_k$, but several tensors can be combined to find component covariances.

3.3 Orientation Tensor: Accumulating the Motion Estimation Tensors

The motion average of consecutive frames can be expressed using a series of tensors. The average motion is given by

$$\mathbf{T}(s) = \sum_{k=1}^{n_\mathrm{f}} \frac{\mathbf{T}_k(s)}{\| \mathbf{T}_k(s) \|_2} \ ,$$

using all video frames. By normalizing $\mathbf{T}$ with a L_2 norm, we are able to compare different video clips or snapshots regardless their length or image resolution. Since $\mathbf{T}$ is a symmetric matrix, it can be stored with $d = \frac{n_\theta(n_\theta+1)}{2}$ elements.

If the motion captured in the histograms are too different from each other, we obtain an isotropic tensor which does not hold useful motion information. But,

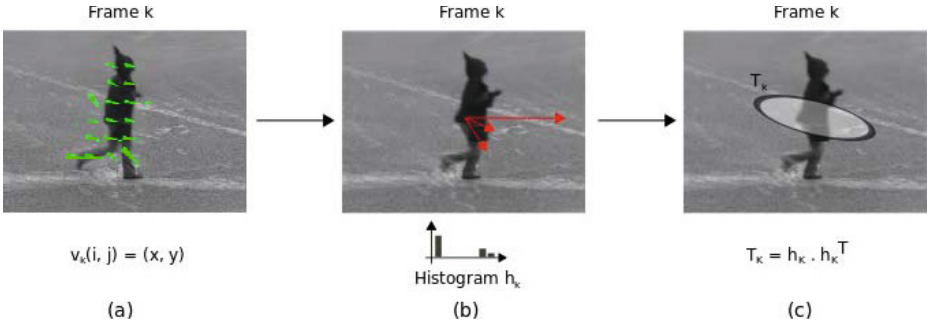

Fig. 1. Example of a tensor descriptor computed for one frame. The ellipse is merely an illustration since generally $n_\theta > 2$. (a) Extracted block displacement vectors. (b) Vectors represented by a histogram h_k. (c) Coding histogram into an orientation tensor.

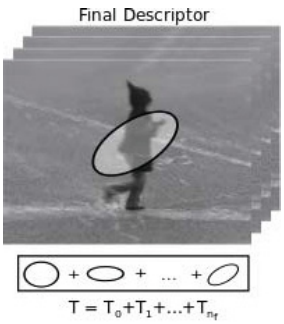

Fig. 2. Frame tensors accumulated in order to model the temporal evolution of motion

if accumulation results in an anisotropic tensor, it carries meaningful average motion information of the frame sequence [10].

Figure 1 shows an example of a video tensor descriptor. To a better understanding of the method, the tensors are represented as ellipses. However, this is only for illustrative purposes since, at this point, the tensor is totally anisotropic and generally $n_\theta > 2$. Figure 2 shows the video tensor, which is the sum of all frame tensors, and its ellipse representation.

4 Block Matching Descriptor: Multiple Scales Single Vector

In this variation, we use multiple block sizes to obtain vectors between only two consecutive frames. This allow us to extract coarse and fine movements, since some blocks will have regions with different motion vectors. We first define a block size set $S = \{s_1, s_2, ..., s_{n_s}\}$. Each block size results in a tensor as defined in 2. The final tensor for a given frame k is the combination of the multiple scales by

$$\mathbf{T}_k(S) = \sum_{s \in S} \frac{\mathbf{T}_k(s)}{\parallel \mathbf{T}_k(s) \parallel_2} \ . \tag{3}$$

Note that the histogram will have the same size for any block size, but this parameter is important to define how many vectors will be represented.

Then, the final video tensor is given by:

$$\mathbf{T}(S) = \sum_{k=1}^{n_\mathrm{f}} \mathbf{T}_k(S) \ .$$

5 Block Matching Descriptor: Multiple Scales Multiple Vectors

Heretofore, our methods used only one successor frame to extract displacement vectors. However, a sequence of successor frames could be used in order to track the block displacement. Thus, this new method (MSMV) uses pairs of adjacent frames of this sequence. The correspondent block found for the previous pair is used as a reference block for the next matching (Fig. 3). Note that with this method it is possible to have block overlaps, which might lead to redundant information in contrast with the previous variations.

Thus, we use the frame k and its t successor frames, generating t vectors for each trajectory starting in the original grid. The parameter t is fixed for all frames. The vector that describes the displacement between a block in frame a and $a + 1$ is defined by $\boldsymbol{v}_{k,a}(i, j) = (x, y) \in \mathbb{R}^2$, where $a \in [k, k + t]$. All the displacement vectors are included in the histogram of the base frame k, i.e. $h_l = \sum_{i,j} r_{k,a}(i, j) \cdot \omega(i, j)$ (analogous to 1), where $r_{k,a}(i, j)$ is the magnitude of

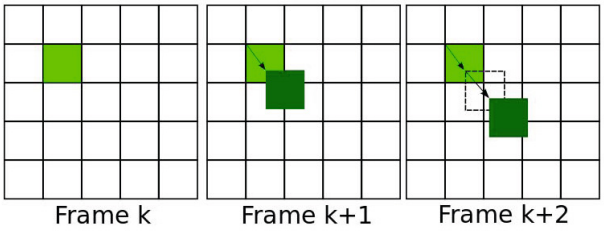

Fig. 3. Block trajectory scheme with two vectors. The first match for frame k is similar to SSSV. The vector found (frame $k+1$) indicates the position of the new reference block (dark green block in frame $k+1$) and this block is used to make a second match between frame $k+1$ and frame $k+2$, generating another displacement vector.

the displacement vector $\boldsymbol{v}_{k,a}(i,j)$. Then, the frame tensor using t frames and S scales is defined as:

$$\mathbf{T}_k^t(S) = \sum_{a=k}^{k+t} \sum_{s \in S} \frac{\mathbf{T}_{k,a}(s)}{\parallel \mathbf{T}_{k,a}(s) \parallel_2} \quad . \tag{4}$$

and the final video tensor is given by:

$$\mathbf{T}^t(S) = \sum_{k=1}^{n_\mathrm{f}} \mathbf{T}_k^t(S) \quad .$$

6 Experimental Results

We chose the 4SS method to generate our descriptor because it is a fast block matching method. We apply a Gaussian filter on each frame to reduce noise. The experiments were made using the three methods shown in Sect. 3, 4 and 5. We use a SVM classifier to evaluate our descriptor on KTH [15], UCF11 [16] and Hollywood2 [17] datasets, which contains six, eleven and twelve human action classes, respectively.

For the SSSV method, we evaluate the descriptor varying its main parameters: block size and the number of histogram cells. The results for KTH dataset are shown in Tab. 1. The best result was achieved with 12×12 blocks and 28 cells. Note that other block sizes and number of cells also produce satisfactory rates. In some applications involving large datasets, for example, the size of the final descriptor play a major role. In that case, the number of cells might be reduced to obtain smaller descriptors.

We achieve 84.8% of recognition rate on KTH dataset with the previous parameters and the confusion matrix of this experiment is shown in Tab. 2. Note that the method obtains good recognition rates for walking because this motion class has many blocks moving to the same direction. This is the same reason that it has difficulty to differ clapping to boxing where the key motion occurs in

Table 1. Experiments on KTH dataset with different block sizes and number of cells

Block Size	Cells	Recognition Rate (%)	Block Size	Cells	Recognition Rate (%)
8	26	81.6	12	24	84.1
10	26	82.9	12	26	84.5
12	**26**	**84.5**	**12**	**28**	**84.8**
14	26	83.2	12	30	83.7
16	26	81.6	18	24	81.9
18	26	83.1	18	26	83.1
20	26	81.7	18	28	84.4
22	26	81.1	18	30	84.1
24	26	81.6			

Table 2. Confusion matrix of the best result on KTH dataset with SSSV method. The average recognition rate is 84.8%.

	Box	HClap	HWav	Jog	Run	Walk
Box	90.2	7.7	2.1	0.0	0.0	0.0
HClap	20.1	79.2	0.7	0.0	0.0	0.0
HWav	3.5	8.3	88.2	0.0	0.0	0.0
Jog	2.1	0.7	0.0	82.6	8.3	6.2
Run	0.7	0.7	0.0	21.5	71.5	5.6
Walk	1.4	0.0	0.0	0.0	1.4	97.2

small regions of the frame. One may see the classical problem to differ running and jogging because of their similarity.

The confusion matrix of the best UCF11 experiment with SSSV is shown in Tab. 3. We achieve 57.2% of recognition rate in this experiment with the same parameters of KTH test. Note that other objects moving in the scene causes difficulty to describe the human movement. It confuses the biking action as riding, for example. The motion direction in both actions is the same but it is hard to infer the vehicle. As in KTH dataset, the best recognition rate was in classes with many vectors having similar directions.

The best result achieved on Hollywood2 dataset was 33.9% and the average precision (AP) for each class of Hollywood2 dataset is given in the Tab. 4. Again, we achieve better recognition rates in classes with more expressive movement in one direction. As expected, this is a challenging dataset where the actions in the video are highly mixed with uncontrolled scenes and are subjected to several sudden cuts. Our result in this dataset is competitive if compared to other global descriptors [10], but with faster processing (Tab. 13).

In MSSV experiments, we combine the block sizes to obtain better results. Using KTH as a reference, the best combination was $S = \{18, 12\}$ (Tab. 5). This combination improves the previous results because bigger blocks tracks bigger objects reducing motion confusion with smaller blocks that capture more

Table 3. Confusion matrix of the best result on UCF11 dataset with SSSV. The average recognition rate is 57.2%.

	Bike	Dive	Golf	Juggle	Jump	Ride	Shoot	Spike	Swing	Tennis	WDog
Bike	58.5	1.7	1.0	0.0	0.7	18.5	1.0	1.7	6.2	4.3	6.4
Dive	0.6	72.2	2.3	2.7	0.0	3.4	8.0	4.8	1.2	1.2	3.5
Golf	0.0	2.6	74.3	5.4	0.0	2.0	1.7	4.0	0.0	9.4	0.7
Juggle	1.4	5.6	5.5	32.1	13.3	1.2	9.5	7.8	11.5	8.7	3.2
Jump	2.5	0.7	1.0	6.9	71.3	0.0	2.7	0.0	14.3	0.7	0.0
Ride	6.9	0.9	0.0	1.2	0.0	81.0	1.3	2.2	0.7	0.6	5.2
Shoot	2.4	16.8	6.3	8.9	3.0	3.0	26.8	12.3	2.9	14.8	2.9
Spike	0.7	11.9	5.0	4.0	1.8	1.1	2.7	60.4	0.0	7.3	5.0
Swing	6.4	0.8	0.5	8.9	17.2	0.0	0.0	0.0	59.4	2.6	4.1
Tennis	2.49	8.6	10.6	6.7	1.7	2.3	6.9	5.9	1.1	50.5	3.4
WDog	14.2	4.6	5.9	3.4	0.0	18.9	3.5	1.8	2.6	2.6	42.5

Table 4. Best result on Hollywood2 dataset with SSSV method. The average recognition rate is 33.9%.

Action	APhone	DCar	Eat	FPerson	GetOutCar	HShake
AP(%)	14.4	70.4	19.8	56.5	21.3	10.2
Action	HPerson	Kiss	Run	SDown	SUp	StandUp
AP(%)	21.3	41.5	48.4	49.3	8.0	45.5

Table 5. Experiments on KTH dataset using multiple scale

Block Sizes Set	**Recognition Rate (%)**
{24, 18, 12}	86.2
{24, 18}	85.1
{24, 12}	86.1
{14,12}	85.8
{14,18}	85.4
{18,14, 12}	85.5
{18, 12}	**86.8**

details. MSSV has increased the recognition rate for all classes (Tab. 6) resulting in 86.8% recognition rate.

We also used the block sizes $S = \{18, 12\}$ to test the UCF11 dataset. It resulted in 58.8% average recognition as shown in Tab. 7. Note that most recognition rates have increased compared to Tab. 3. The same test was performed with Hollywood2 dataset yielding exactly the same average recognition of 33.9% but with different rates for some classes.

In MSMV experiments, we used multiple vectors in two cases: with and without multiple scales. The results for KTH are shown in Tab. 8. Multiple vectors with one block size produces better results than the SSSV method. However,

Table 6. Confusion matrix of the best result on KTH dataset with MSSV method. The average recognition rate is 86.8%.

	Box	HClap	HWav	Jog	Run	Walk
Box	91.6	7.7	0.7	0.0	0.0	0.0
HClap	17.4	81.9	0.7	0.0	0.0	0.0
HWav	2.8	7.6	89.6	0.0	0.0	0.0
Jog	2.1	0.7	0.0	85.4	7.6	4.2
Run	0.7	0.0	0.0	22.9	74.3	2.1
Walk	1.4	0.0	0.0	0.0	0.7	97.9

Table 7. Confusion matrix of the best result on UCF11 dataset with MSSV. The average recognition rate is 58.8%.

	Bike	Dive	Golf	Juggle	Jump	Ride	Shoot	Spike	Swing	Tennis	WDog
Bike	56.2	1.0	0.0	0.0	1.0	21.1	1.0	1.	7.0	4.3	6.8
Dive	0.6	73.4	1.6	4.0	0.0	2.8	8.5	1.0	1.2	2.4	4.5
Golf	0.0	2.3	80.1	5.4	0.0	2.0	1.0	2.4	0.7	5.4	0.7
Juggle	0.6	4.6	7.9	36.0	11.9	1.2	10.3	4.3	9.1	10.8	3.2
Jump	1.8	0.0	1.0	8.1	74.8	0.0	2.0	0.0	12.3	0.0	0.0
Ride	6.7	1.4	0.0	0.5	0.0	80.0	0.7	2.9	1.3	0.7	5.9
Shoot	1.8	18.4	4.4	13.7	3.0	4.0	25.0	13.8	2.1	10.9	2.8
Spike	0.7	9.1	3.0	1.0	2.8	2.1	6.0	62.9	0.0	7.3	5.0
Swing	5.4	0.8	1.0	7.1	16.4	0.8	0.0	0.0	64.6	2.0	2.0
Tennis	3.0	8.0	11.1	7.8	1.7	1.7	7.5	4.1	1.1	53.4	0.6
WDog	12.7	4.7	5.6	1.6	0.0	21.2	3.5	3.8	3.6	2.6	40.7

we achieve even better results combining multiple vectors and multiple scales. Another interesting observation is that the recognition rate increases along with the number of trajectory vectors up to a certain point, and then decreases. This occurs because using more frames augments the probability of objects disappearing from the scene or to cover other blocks causing redundancies. The best recognition rate was 87.7% with $S = \{24, 18, 12\}$ and 3 vectors. Its confusion matrix is shown in Tab. 9. Note that the major benefit in this variant is the recognition gain in clapping class, reducing the confusion between clapping and boxing.

We obtain 59.5% on UCF11 dataset using MSMV and its confusion matrix is shown on Tab. 10. Note that dive and swing improved considerably, which contributed to improve the recognition rate. The Hollywood2 dataset rates are shown in Tab. 11, we also obtain a better result than previous variants of our method: 34.9%.

Table 12 shows that our results are close to, but lower than the state-of-the-art results. The best results for these databases are presented in [18, 19]. However this method is more complex than ours. It combines trajectories, HOG, Histogram of Optical Flow (HOF) and Motion Boundary Histogram (MBH), along

Table 8. Experiments on KTH dataset using multiple vectors

Block Sizes Set	Vectors	Recognition Rate(%)	Block Sizes Set	Vectors	Recognition Rate(%)
{18}	**2**	**85.5**	{18,12}	2	86.9
{18}	3	85.5	{18,12}	3	86.4
{18}	4	84.7	{18,12}	4	86.4
{18}	5	84.4	{18,12}	5	86.0
{18}	6	85.1	{24,12}	2	86.2
{12}	2	84.7	{24,12}	3	86.5
{12}	3	85.2	{24,12}	4	87.1
{12}	4	84.8	{24,12}	5	86.7
{12}	5	84.8	{24,18}	2	86.9
{12}	6	84.7	{24,18}	3	87.0
			{24,18}	4	86.2
			{24,18}	5	86.0
			{24,18,12}	2	87.6
			{24,18,12}	**3**	**87.7**
			{24,18,12}	4	86.5
			{24,18,12}	5	85.9

Table 9. Confusion matrix of the best result on KTH dataset with MSMV method. The average recognition rate is 87.7%.

	Box	HClap	HWav	Jog	Run	Walk
Box	91.6	7.0	1.4	0.0	0.0	0.0
HClap	13.2	86.1	0.7	0.0	0.0	0.0
HWav	2.1	4.2	93.8	0.0	0.0	0.0
Jog	1.4	0.0	0.0	84.7	6.2	7.6
Run	0.7	0.0	0.0	25.0	70.8	3.5
Walk	0.7	0.0	0.0	0.0	0.0	99.3

with a Bag-of-Features (BoF) technique, which includes a clustering overhead in order to generate the final descriptor. Yet, for KTH dataset using block matching approach, our method can achieve high execution speed as shown in Tab. 13. All experiments where generated in a machine with Intel Xeon E-4610, 2.4 GHz, 32 GB of memory using 10 threads. Using the SSSV on KTH dataset achieves around 2 ms per frame. As expected, in Hollywood2 the frame rate is lower because of its higher resolution.

There are several works on literature proposing compact image descriptors. For video descriptors, this property has not been exploited yet. One compact image descriptor is presented in [20] and each image is stored using 256 bits. With 28 histogram cells, our resulting video descriptor has only 406 elements. In KTH dataset for example, this represents an average of 138.2 bits per frame. According to them, this property is an advantage for large datasets and also enables fast search in retrieval systems.

Table 10. Confusion matrix of the best result on UCF11 dataset with MSMV. The average recognition rate is 59.5%.

	Bike	Dive	Golf	Juggle	Jump	Ride	Shoot	Spike	Swing	Tennis	WDog
Bike	56.0	1.0	0.0	0.0	2.3	20.9	2.0	0.7	5.2	4.3	7.6
Dive	0.6	76.8	1.6	4.0	0.0	1.8	7.4	1.7	1.2	2.2	2.8
Golf	0.0	3.3	78.3	4.9	0.0	1.0	1.0	2.4	0.0	7.8	1.2
Juggle	1.4	4.5	10.8	35.3	9.0	0.6	9.3	4.0	9.4	11.9	3.7
Jump	1.8	0.0	1.0	8.3	73.6	0.0	2.0	0.0	13.3	0.0	0.0
Ride	7.2	1.4	0.0	0.0	0.0	78.9	0.7	2.9	1.3	0.5	7.1
Shoot	1.0	19.2	7.1	10.9	2.3	3.0	27.5	14.7	3.4	9.8	1.2
Spike	1.7	10.6	3.7	1.0	2.8	1.1	6.0	61.5	0.0	7.7	4.0
Swing	5.4	0.8	1.0	5.1	13.0	0.8	0.0	0.0	71.3	1.3	1.3
Tennis	4.2	8.1	8.8	7.8	1.7	0.6	5.0	7.0	2.3	53.9	0.6
WDog	15.0	4.3	3.5	2.4	0.0	21.8	1.0	3.8	4.9	1.8	41.6

Table 11. Average precision for each class of Hollywood2 dataset with MSMV. The average recognition rate is 34.9%.

Action	APhone	DCar	Eat	FPerson	GetOutCar	HShake
AP(%)	16.1	68.7	24.1	60.6	23.9	8.9
Action	HPerson	Kiss	Run	SDown	SUp	StandUp
AP(%)	18.7	42.9	50.3	48.6	11.5	45.2

Table 12. Comparison with state-of-the-art for KTH, UCF11 and Hollywood2 datasets

	KTH	**UCF11**	**Hollywood2**
Klaser et al. (2008)	91.0		24.7
Liu et al. (2009)	93.8		
Mota et al. (2013)	93.2	72.7	40.3
Sad et al. (2013)	93.3	72.6	41.9
Wang et al. (2013)	**95.3**	**89.9**	59.9
Wang and Schmid (2013)			**64.3**
Our method	**87.7**	**59.5**	**34.9**

Table 13. Frame rate for each method

Method	**KTH (fps)**	**UCF11 (fps)**	**Hollywood2 (fps)**
SSSV	502.7	117.6	26.8
MSSV	319.3	69.2	15.5
MSMV	89.1	18.5	2.2

7 Conclusion

In this paper, we present a novel approach for motion description in videos using block matching. The resulting tensor descriptor is a simple but effective approach for video classification. It is simple because of its low complexity in terms of time and space. Our recognition rate is lower than the approaches in the literature but fairly competitive in KTH, UCF11 and Hollywood2 datasets, if compared to other self-descriptors. We obtain 84.8% recognition rate with KTH in the SSSV version with the processing rate of 502.7 frames per second.

The main advantage of our method is that it reaches good recognition rates depending uniquely on the input video. This is a different and simple approach considering that most works on the field are based on the gradient of image intensities.

Our method is fast and the descriptor is compact, making it suitable for big datasets. The addition of new videos and/or new action categories with our approach does not require any recomputation or changes to the previously computed descriptors. Finally, it might be valuable in a scenario where the application demands fast processing and a compact descriptor.

Acknowledgements. Authors thank to FAPEMIG, CAPES and UFJF for funding.

References

1. Hafiane, A., Palaniappan, K., Seetharaman, G.: Uav-video registration using block-based features. In: IEEE International Geoscience and Remote Sensing Symposium (IGARSS), vol. 2, pp. 1104–1107 (2008)
2. Amel, A.M., Abdessalem, B.A., Abdellatif, M.: Video shot boundary detection using motion activity descriptor. Journal of Telecommunications 2(1), 54–59 (2010)
3. Sun, X., Divakaran, A., Manjunath, B.S.: A motion activity descriptor and its extraction in compressed domain. In: Shum, H.-Y., Liao, M., Chang, S.-F. (eds.) PCM 2001. LNCS, vol. 2195, pp. 450–457. Springer, Heidelberg (2001)
4. Mota, V.F., Souza, J.I., de A. Araújo, A., Vieira, M.B.: Combining orientation tensors for human action recognition. In: Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 328–333. IEEE (2013)
5. Sad, D., Mota, V.F., Maciel, L.M., Vieira, M.B., de A. Araújo, A.: A tensor motion descriptor based on multiple gradient estimators. In: Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 70–74. IEEE (2013)
6. Perez, E.A., Mota, V.F., Maciel, L.M., Sad, D., Vieira, M.B.: Combining gradient histograms using orientation tensors for human action recognition. In: 21st International Conference on Pattern Recognition (ICPR), pp. 3460–3463. IEEE (2012)
7. Ji, Y., Shimada, A., Taniguchi, R.I.: A compact 3d descriptor in roi for human action recognition. In: IEEE TENCON, pp. 454–459 (2010)
8. Mota, V.F., Perez, E.A., Vieira, M.B., Maciel, L., Precioso, F., Gosselin, P.H.: A tensor based on optical flow for global description of motion in videos. In: Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 298–301. IEEE (2012)

9. Kläser, A., Marszałek, M., Schmid, C.: A spatio-temporal descriptor based on 3d-gradients. In: British Machine Vision Conference (BMVC), pp. 995–1004 (September 2008)
10. Mota, V.F., Perez, E.A., Maciel, L.M., Vieira, M.B., Gosselin, P.H.: A tensor motion descriptor based on histograms of gradients and optical flow. Pattern Recognition Letters 31, 85–91 (2013)
11. Po, L.M., Ma, W.C.: A novel four-step search algorithm for fast block motion estimation. IEEE Transactions on Circuits and Systems for Video Technology 6(3), 313–317 (1996)
12. Nie, Y., Ma, K.K.: Adaptive rood pattern search for fast block-matching motion estimation. IEEE Transactions on Image Processing 11(12), 1442–1449 (2002)
13. Zhu, S., Ma, K.K.: A new diamond search algorithm for fast block-matching motion estimation. IEEE Transactions on Image Processing 9(2), 287–290 (2000)
14. Johansson, B., Farnebäck, G.: A theoretical comparison of different orientation tensors. In: Proceedings of the SSAB Symposium on Image Analysis, pp. 69–73 (2002)
15. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local svm approach. In: Proceedings of the 17th International Conference on Pattern Recognition (ICPR), vol. 3, pp. 32–36. IEEE (2004)
16. Liu, J., Luo, J., Shah, M.: Recognizing realistic actions from "videos in the wild". In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1996–2003. IEEE (2009)
17. Marszalek, M., Laptev, I., Schmid, C.: Actions in context. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2929–2936. IEEE (2009)
18. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Dense trajectories and motion boundary descriptors for action recognition. International Journal of Computer Vision 103(1), 60–79 (2013)
19. Wang, H., Schmid, C., et al.: Action recognition with improved trajectories. In: International Conference on Computer Vision (2013)
20. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large image databases for recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2008)