Journal of Mobile Multimedia, Vol. 11, No. 1&2 (2015) 090–102
© Rinton Press

A VIDEO TENSOR SELF-DESCRIPTOR BASED ON VARIABLE SIZE BLOCK MATCHING

HELENA ALMEIDA MAIA
Department of Computer Science, Universidade Federal de Juiz de Fora
Juiz de Fora, Minas Gerais, Brazil
helena.maia@ice.ufjf.br

ANA MARA DE OLIVEIRA FIGUEIREDO
Department of Computer Science, Universidade Federal de Juiz de Fora
Juiz de Fora, Minas Gerais, Brazil
anamara@ice.ufjf.br

FÁBIO LUIZ MARINHO DE OLIVEIRA
Department of Computer Science, Universidade Federal de Juiz de Fora
Juiz de Fora, Minas Gerais, Brazil
fabio@ice.ufjf.br

VIRGÍNIA FERNANDES MOTA
Department of Computer Science, Universidade Federal de Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
virginiaferm@dcc.ufmg.br

MARCELO BERNARDES VIEIRA
Department of Computer Science, Universidade Federal de Juiz de Fora
Juiz de Fora, Minas Gerais, Brazil
marcelo.bernardes@ice.ufjf.br

This paper presents a different and simple approach for video description using only block matching vectors, considering that most works on the field are based on the gradient of image intensities. We first divide the image into blocks of different sizes. The block matching method returns a displacement vector for each block, which we use as motion information to obtain orientation tensors and to generate the final self-descriptor, since it depends only on the input video. The resulting 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, variable size 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. In our previous work [1], we exploited this technique as it has not been extensively applied to human action recognition. Several works on literature use block displacement vectors for other applications, for example [2, 3, 4, 5, 6]. We achieved competitive recognition rates using this approach with a high frame rate and generating compact descriptors. Hence, in the current work, we propose an enhancement of the previous method by exploring a block matching variant: Variable Size Block Matching Algorithm (VSBMA).

This work is motivated by the possibility of generating a compact and easy to compute descriptor and obtain better results than our preceding work. Our main contribution is a new motion descriptor, based on orientation tensors [7], which uses only block matching information. This is a different approach, considering that most works on the field are based on the gradient of image intensities [8, 9]. The global tensor descriptor created is evaluated by classifying videos from KTH, UCF11, and Hollywood2 datasets with a non-linear Support Vector Machine (SVM) classifier.

Previously, we presented three variants of our method. The first, called Single Scale Single Vector (SSSV) 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), considers more than one block size. It requires multiple computations for each pair of frames. The third version, called Multiple Scales Multiple Vectors (MSMV), considers multiple block sizes and more than two consecutive frames. Thus far, MSMV produced the best results, at the cost of having the slowest computation.

In this work we present two new variants. The first, called Variable Scale Single Vector (VSSV), yields better results than all of aforementioned variants. It uses a initial block size and the blocks may be split into four new blocks during the block matching routine, hence the *variable size block* denomination. Only one vector field for each pair of frames is used, similar to MSSV. The second version, called Variable Scale Multiple Vectors (VSMV), yields the best results but is also even slower than all the previous variants. It also considers variable size blocks and goes through more than two consecutive frames, similar to MSMV.

2 Related Works

Several works in literature use the motion intensity obtained from block matching in many applications. Hafiane et al. [2] presents a method for video registration based on Prominent Feature (PF) blocks. In their work, block matching was used to identify moving objects in videos. 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 was shown to be less sensitive to absolute intensity changes between the reference and target images due to normalization, but 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 good results.

Similar to [2], a block matching method was used for extracting motion information in [3].

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 indicated a greater probability of being a shot.

An activity descriptor, consisting of a temporal descriptor and a spatial descriptor, is presented in [4]. 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 [3], is obtained through a matrix containing all the motion vectors norms from each frame.

Puri et al. [10] describes a VSBMA scheme that improves the motion estimation performance by subdividing blocks of images into sub-blocks and estimating their movements. The presented technique was used to encode the motion-compensated interframe differential pels. Similarly, Chan et al. [5] proposed an algorithm to adaptively divide the image into blocks of variable size to meet the assumption on uniform motion for all blocks. The scheme was applied to simple interframe video coding. Recently, Muralidhar et al. [11] proposed an efficient architecture for implementation of VSBMA as specified by the h.264 video compression format. Following the same idea, we use VSBMA to be able to extract coarse and fine movements, since some blocks will have regions with different motion vectors.

A variable block size motion estimation technique based on a hierarchical structure is proposed in [12]. It improved the motion vector encoding efficiency and reduced the number of motion vectors to be transmitted as well. An efficient quadtree-based algorithm for VSBMA is presented in [13]. The scheme allowed the dimensions of blocks to adapt to local activity within the image, and the total number of blocks in any frame could be varied while still accurately representing true motion. We use these works as inspiration for our quadtree-based implementation.

Other video descriptors were proposed using different methods for extracting information, such as the human action descriptor shown in [7, 8, 9, 14, 15]. Klaser et al. [16] propose a local feature based descriptor for video sequences generalizing Histogram of Oriented Gradient (HOG) concepts to 3D. In [14], 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 [16], 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.

Mota et al. [17] presented a tensor motion descriptor for video sequences using optical flow and HOG3D information. They use an aggregation tensor based technique, combining two descriptors, one carries polynomial coefficients which approximate optical flow, and the other carries 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 bi-dimensional nature of block displacements reduces significantly the size of the histogram coded into a tensor. Compared to [17], our descriptor is more compact and easier to compute,

while still yielding competitive results.

3 Variable Size Block Matching Descriptor: Variable 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 [18], ARPS [19] and DS (Diamond Search) [20], 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.

All previous methods use fixed block size. Another approach to solve this problem is the Variable Size Block Matching Algorithm. This method is similar to BMA approaches: it searches for each block with an initial size from the reference frame in the target frame based on any search strategy of BMA variants. If the match error found is greater than a fixed threshold, the block is split into four half-sized blocks until the error is below the threshold or it reaches a minimum size. A quad-tree is used for this purpose with leaf nodes corresponding to blocks of varying sizes. The goal is to make the edge of the blocks coincide with the border of objects in the scene, forming regions with uniform displacement.

The input of our method 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 $\mathbf{T} \in R^{n \times n}$ which, in fact, can be viewed as a vector in R^m, $m = n^2$. To generate the tensor descriptor, we use the displacement map generated by the VSBMA with the 4SS strategy. We have already proposed a method to exploit 4SS method with multiple size blocks (MSSV and MSMV [1]). However, those methods use the displacement maps for all of the block sizes to form the descriptor. In Variable Size Single Vector (VSSV), we select the most representative size for each region of the frame, that is, the frame is divided into blocks with an initial size, which are split only when the lowest BDM is still above the established threshold. This way, one single frame can have blocks of various sizes, and their displacement vectors are all within the same map. The vector field is then represented by a histogram, which in turn is encoded into an orientation tensor. The tensors of each frame are accumulated to form the final tensor descriptor $\mathbf{T}$ of the video.

3.1 Computing the Motion Estimation Histogram

In the VSBMA scheme, each frame k of the video is subdivided into n non-overlapping square blocks with fractions of the initial size s. Note that, each frame may have different number of

blocks since we use variable sizes. For each block, displacement vectors $\vec{v}_k(i,j) = (x,y) \in R^2$ are calculated, where (i,j) are the block coordinates in frame. These vectors are converted to equivalent polar coordinates $\vec{c}_k(i,j) = (\theta, r)$ with $\theta = \tan^{-1}(\frac{y}{x})$, $\theta \in [0, 2\pi]$ and $r = \parallel \vec{v}_k(i,j) \parallel$.

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 $\vec{h}_k = (h_1, h_2, \ldots, h_{n_\theta})^T$, 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 $\vec{h}_k$ with n_θ elements, regardless of the number of vectors from the field.

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 [21]. Given a vector $\vec{v} \in R^n$, it can be represented by the tensor $\mathbf{T} = \vec{v}\vec{v}^{\mathrm{T}}$. We use the orientation tensor to represent the histogram $\vec{h}_k \in R^{n_\theta}$. The frame tensor for the initial size s, $\mathbf{T}_k(s) \in R^{n_\theta \times n_\theta}$, is given by:

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

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

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

$$\mathbf{T}(s) = \sum_{k=1}^{n_\mathrm{f}} \frac{\mathbf{T}_k(s)}{\parallel \mathbf{T}_k(s) \parallel_2} \ ,$$

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, if accumulation results in an anisotropic tensor, it carries meaningful average motion information of the frame sequence [17].

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.

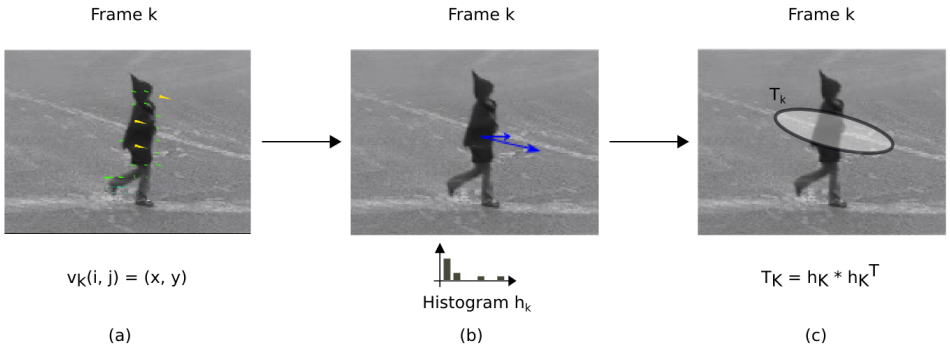

Fig. 1. Example of a tensor descriptor computed for one frame. The ellipse is merely an illustration since generally $n_\theta > 2$. (a) Extracted block displacement vectors. (b) Vectors represented by a histogram h_k. (c) Coding histogram into an orientation tensor.

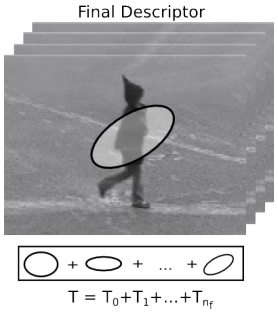

Fig. 2. Frame tensors accumulated in order to model the temporal evolution of motion.

4 Variable Size Block Matching Descriptor: Variable Scale Multiple Vector

In this variant we want to exploit the information of object trajectory through the video. This way, the VSMV is going to use t pairs of adjacent frames from a sequence. The correspondent block found for the previous pair is used as a reference block for the following match.

Since the aim of this variant is to follow the same object, only the first pair in a sequence can have its blocks divided. The following pairs use the same block size configuration as the first pair in the sequence. Figure 3 shows trajectories starting in frame k *(top)* and frame $k+1$ *(bottom)*. Note that, in frame k a block is split and the size stays the same throughout the whole trajectory. However, when the trajectory starts in frame $k+1$, the block can be split between frame $k+1$ and frame $k+2$.

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 $\vec{v}_{k,a}(i,j) = (x,y) \in R^2$, 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 Equation 1), where $r_{k,a}(i,j)$ is the magnitude of the displacement vector $\vec{v}_{k,a}(i,j)$. Then, the frame tensor using t frames and

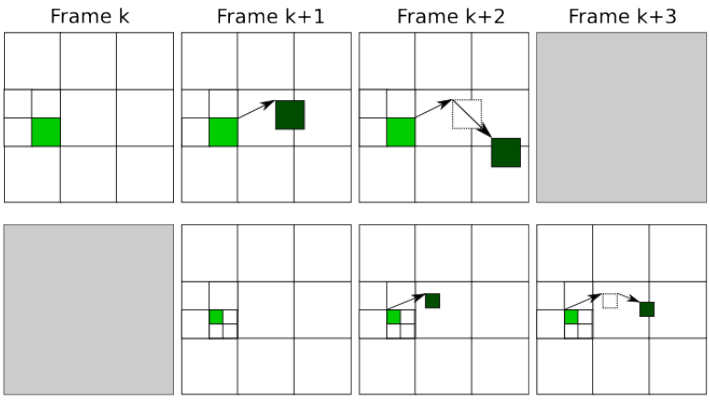

Fig. 3. Block trajectory scheme with two vectors. *(top)* Trajectory starting in frame k. In the first match, a split occurs. The displacement vector found in frame $k+1$ indicates the position of the new reference block (dark green). This block is used to make the second match between frame $k+1$ and frame $k+2$ generating another displacement vector. From the second match onwards, it cannot be split any further. *(bottom)* Trajectory starting in frame $k+1$.

the initial size s is defined as:

$$\mathbf{T}_k^t(s) = \sum_{a=k}^{k+t} \frac{\mathbf{T}_{k,a}(s)}{\| \mathbf{T}_{k,a}(s) \|_2} \ , \tag{3}$$

and the final video tensor is given by:

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

5 Experimental Results

The experiments were made using the two methods shown in Section 3 and 4. Following the same protocol of our previous work [1], we use a SVM classifier to evaluate our descriptor on KTH [22], UCF11 [23] and Hollywood2 [24] datasets, which contains six, eleven, and twelve human action classes, respectively. Just as the number of classes, the image resolution and complexity of the scenes portrayed increases from KTH to UCF11 and even further in Hollywood2, making the latter the most challenging dataset used.

With BMA-based methods, we achieved 84.8% of recognition rate in SSSV, 86.8% in MSSV and 87.7% in MSMV on KTH dataset. For the VSSV method, we evaluate the descriptor varying its main parameters: block size, number of histogram cells and error threshold value. The results for KTH dataset with the threshold fixed in 10 are shown in Table 1, where the best result was achieved with 32×32 blocks and 26 cells. This recognition rate is even better than MSMV results. Note that, other block sizes and number of cells also produce satisfactory rates. In Table 2 we show test results with different threshold values with the block size and number of cells from the best result in Table 1. In this case, the recognition rate increases up to threshold values equal to 10, and decreases as the threshold goes beyond that. This

happens because as the threshold values get higher, meaning higher acceptable error values, less blocks are split, leading to coarser displacement maps. In an extreme scenario with no blocks split at all, the method degenerates to fixed size block matching. As seen on our previous work [1], fixed size blocks tend to yield worse results, since it is difficult to strike a balance between capturing fine movement, movement of homogeneous regions, and noise.

Table 1. Experiments on KTH dataset with different initial block sizes and number of cells using VSSV with threshold value equal to 10.

Block	**Cells**	**Recognition Rate (%)**	**Block**	**Cells**	**Recognition (%) Rate (%)**	**Block**	**Cells**	**Recognition Rate(%)**
16	16	81.9	24	16	83.3	32	16	86.0
16	18	84.9	24	18	85.2	32	18	86.9
16	24	85.8	24	24	85.5	32	24	87.8
16	26	86.0	**24**	**26**	**86.3**	**32**	**26**	**89.0**
16	**28**	**87.0**	24	28	85.6	32	28	88.5
16	30	86.7	24	30	86.2	32	30	88.4

Table 2. Experiments on KTH dataset with different threshold values, 32×32 blocks and 26 cells using VSSV.

Threshold	**Recognition Rate (%)**
3	86.1
5	88.1
7	88.4
10	**89.0**
12	88.9
14	88,2
20	88.6
24	87.0
100	77.3

We achieve 89.0% recognition rate on KTH dataset with the previous parameters and the confusion matrix of this experiment is shown in Table 3. The method obtains good recognition rates for the *walking* class because this motion class has many blocks moving in the same direction. This is the same reason why it has difficulty to differ *clapping* from *boxing* where the key motion occurs in small regions of the frame. One may see the classical problem to differ *running* from *jogging* because of their similarity. The VSSV method has consistently better results than SSSV and MSSV methods. This is due to VSSV's property of grouping information from different blocks sizes without all the redundant information of multiple computations over the same sequence of frames, as it is in MSSV.

Table 3. Confusion matrix of the best result on KTH dataset with VSSV method. The average recognition rate is 89.0%.

	Box	HClap	HWav	Jog	Run	Walk
Box	96.5	3.5	0.0	0.0	0.0	0.0
HClap	19.4	79.9	0.7	0.0	0.0	0.0
HWav	0.0	4.2	95.8	0.0	0.0	0.0
Jog	0.7	0.0	0.0	86.1	5.6	7.6
Run	0.0	0.0	0.0	20.8	76.4	2.8
Walk	0.0	0.0	0.0	0.7	0.0	99.3

Table 4 shows threshold variation tests using 24×24 blocks and 30 cells on UCF11 dataset. The best result occurs with threshold value equal to 12. The confusion matrix of this experi-

ment with VSSV is shown in Table 5. We achieve 61.6% recognition rate in this experiment with previously mentioned parameters. Note that other objects moving in the scene cause some considerable difficulty to describe human movement. The confusion between *biking* and *riding* is a clear example. The motion direction in both actions is the same but it is hard to infer the vehicle. As in KTH dataset, the most distinctive classes were the ones with many vectors pointing to similar directions. Using the VSSV we achieve 61.6% which is a considerable improvement of recognition rates over SSSV, 57.2%, and MSSV, 58.8%.

Table 4. Experiments on UCF11 dataset with different threshold, 24×24 blocks and 30 cells using VSSV.

Threshold	**Recognition Rate (%)**
3	60.0
5	60.4
7	60.4
10	59.8
12	**61.6**

Table 5. Confusion matrix of the best result on UCF11 dataset with VSSV. The average recognition rate is 61.6%.

	Bike	Dive	Golf	Juggle	Jump	Ride	Shoot	Spike	Swing	Tennis	WDog
Bike	57.1	2.3	0.0	1.0	1.9	19.4	1.0	1.0	5.4	2.7	8.1
Dive	0.6	80.7	0.6	0.6	0.7	1.9	6.2	3.0	0.7	2.9	2.2
Golf	0.6	5.2	80.9	5.0	0.0	0.0	3.0	1.7	0.0	2.7	1.0
Juggle	3.2	2.0	7.3	57.8	5.7	1.1	3.0	3.3	5.6	10.3	0.7
Jump	2.6	0.0	0.0	7.4	75.7	0.0	2.0	0.0	12.3	0.0	0.0
Ride	7.2	0.9	0.0	0.0	0.0	79.6	0.0	2.9	0.7	1.2	7.6
Shoot	1.2	9.0	5.9	10.9	2.1	4.7	35.3	13.7	1.0	13.4	2.8
Spike	1.7	9.9	2.0	1.8	1.0	3.7	5.5	58.0	0.0	11.5	5.0
Swing	5.2	0.0	2.5	7.5	13.2	0.0	0.0	0.8	66.9	1.2	2.6
Tennis	3.0	8.4	9.3	9.0	1.8	1.3	8.0	9.1	4.0	44.3	1.7
WDog	14.2	5.5	3.5	0.8	0.0	20.9	2.7	2.8	3.5	5.2	40.9

The best result achieved on Hollywood2 dataset is 36.5% with 24×24 blocks, 30 cells and threshold value equal to 7. The average precision (AP) for each class of Hollywood2 dataset is presented in Table 6. Again, we achieve better recognition rates in classes with more expressive movement in one direction. As expected, this is a challenging dataset where the human 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 [17], but with faster processing (refer to Table 13 for performance details).

Table 6. Best result on Hollywood2 dataset with VSSV method. The average recognition rate is 36.5%.

Action	APhone	DCar	Eat	FPerson	GetOutCar	HShake
AP(%)	10.4	70.3	34.2	65.9	28.1	9.5
Action	HPerson	Kiss	Run	SDown	SUp	StandUp
AP(%)	19.5	42.1	52.3	50.1	11.8	44.0

In VSMV experiments we use multiple vectors and the results for KTH are shown in Table 7. An 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 91.5% using 7 vectors. The confusion matrix corresponding to the best result is shown in Table 8. A major benefit of this variant is the recognition gain in *clapping* class, reducing the confusion between *clapping* and *boxing*.

Table 7. Experiments on KTH dataset with different number of vector using VSMV.

Number of vectors	**Recognition rate (%)**
2	88.5
3	89.3
4	89.7
5	90.0
6	90.5
7	**91.5**
8	91.1
9	91.0

Table 8. Confusion matrix of the best result on KTH dataset with VSMV method. The average recognition rate is 91.5%.

	Box	HClap	HWav	Jog	Run	Walk
Box	95.8	3.5	0.0	0.7	0.0	0.0
HClap	6.2	93.1	0.7	0.0	0.0	0.0
HWav	0.0	0.7	99.3	0.0	0.0	0.0
Jog	0.7	0.0	0.0	86.8	6.9	5.6
Run	0.7	0.0	0.0	23.6	75.0	0.7
Walk	0.0	0.0	0.0	0.0	0.7	99.3

We obtain 64.0% recognition rate on UCF11 dataset using VSMV with 32×32 initial block size, 30 cells, threshold value equal to 12 and 4 vectors. The confusion matrix is shown on Table 9. Most classes were improved, which contributed to the recognition rate rise. The Hollywood2 dataset rates are shown in Table 10, we also obtain a better result than previous variants of our method: 36.9%. The parameters are 32×32 initial block size, 30 cells, threshold value equal to 3 and 2 vectors.

Table 9. Confusion matrix of the best result on UCF11 dataset with VSMV. The average recognition rate is 64.0%.

	Bike	Dive	Golf	Juggle	Jump	Ride	Shoot	Spike	Swing	Tennis	WDog
Bike	60.9	2.3	0.0	0.0	0.7	21.1	1.0	2.0	3.2	2.7	6.2
Dive	2.5	82.6	0.0	0.6	0.7	1.2	2.8	4.0	0.0	2.4	3.2
Golf	0.6	1.6	80.9	7.0	0.0	0.0	3.3	0.5	0.0	3.8	2.3
Juggle	0.6	0.7	8.3	59.9	5.0	1.2	6.0	3.7	6.5	6.3	1.7
Jump	2.6	0.0	0.0	5.4	80.0	0.0	1.0	0.0	11.0	0.0	0.0
Ride	7.1	0.4	0.0	0.6	0.0	78.6	0.0	3.6	0.7	1.2	7.9
Shoot	2.8	4.7	5.9	11.2	0.0	3.9	39.2	8.3	2.2	18.7	3.0
Spike	0.7	11.1	2.7	2.8	0.8	2.9	3.0	63.7	0.0	8.3	4.0
Swing	3.1	0.8	4.2	8.0	9.4	0.0	1.0	0.0	71.3	0.7	1.6
Tennis	2.9	7.7	7.0	8.8	1.1	1.2	9.4	6.2	4.1	48.6	2.9
WDog	16.0	3.3	6.7	0.8	0.0	22.6	2.8	3.1	3.0	3.3	38.4

Table 11 shows a summary of our current and previous methods. Note that, variable size blocks provide a consistent improvement in recognition rates for all datasets over the fixed block size methods. 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 [26, 27]. However this

Table 10. Average precision for each class of Hollywood2 dataset with MSMV. The average recognition rate is 36.9%.

Action	APhone	DCar	Eat	FPerson	GetOutCar	HShake
AP(%)	20.9	73.4	27.6	59.6	24.4	18.8
Action	HPerson	Kiss	Run	SDown	SUp	StandUp
AP(%)	22.1	42.3	52.6	47.6	10.2	43.5

method is much more complex than ours. It combines trajectories, HOG, Histogram of Optical Flow (HOF) and Motion Boundary Histogram (MBH), along 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 Table 13. All experiments were generated with an Intel Xeon E-4610, 2.4 GHz, 32 GB of memory using 10 parallel threads. VSSV achieves around 1.5 ms per frame on KTH dataset. As expected, in Hollywood2 the frame rate is lower due to its higher resolution.

There are several works on literature proposing compact image descriptors. For video descriptors, to the best of our knowledge, this property has not been exploited yet. One compact image descriptor is presented in [25] and each image is represented using 256 bits. According to them, this property is an advantage for large datasets and also enables fast search in retrieval systems. With 26 histogram cells, our resulting video descriptor has only 351 single precision floating-point elements. For the KTH, this video descriptor is only 0.078% the average video size in bits.

Table 11. Block matching descriptor summary.

Variant	**KTH**	**UCF11**	**Hollywood2**
SSSV [1]	84.8	57.2	33.9
MSSV [1]	86.8	58.8	33.9
MSMV [1]	87.8	59.5	34.9
VSSV	89.0	61.6	36.5
VSMV	91.5	64.0	36.9

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**
Figueiredo et al. (2014)	87.7	59.5	34.9
Our method	**91.5**	**64.0**	**36.5**

Table 13. Frame rate for each method.

Method	**KTH (fps)**	**UCF11 (fps)**	**Hollywood2 (fps)**
VSSV	675.6	111.1	36.8
VSMV	184.9	47.1	18.3

6 Conclusion

In this paper, we present a extension of our previous work that presented 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 state-of-the-art approaches in the literature but fairly competitive in KTH, UCF11 and Hollywood2 datasets, if compared to other self-descriptors. We obtain 89.0% recognition rate with KTH in the VSSV version with the processing rate of 675.6 frames per second.

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

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 re-computation 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. Figueiredo, Ana MO and Maia, Helena A and Oliveira, Fábio LM and Mota, Virgínia F and Vieira, Marcelo Bernardes (2014) *A Video Tensor Self-descriptor Based on Block Matching*, Computational Science and Its Applications–ICCSA 2014, pp 401–414.
2. Hafiane, Adel and Palaniappan, Kannappan and Seetharaman, Guna (2008), *UAV-Video Registration Using Block-based Features*, IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Vol. 2, pp. 1104-1107.
3. Amel, Abdelati Malek and Abdessalem, Ben Abdelali and Abdellatif, Mtibaa(2010), *Video Shot Boundary Detection Using Motion Activity Descriptor*, Journal of Telecommunications, Vol.2, pp. 54-59.
4. Sun, Xinding and Divakaran, Ajay and Manjunath, BS (2001), *A Motion Activity Descriptor and its Extraction in Compressed Domain*, Proceedings of the 2nd IEEE Pacific Rim Conference on Multimedia, Springer, pp 450-457.
5. Chan, MH and Yu, YB and Constantinides, AG (1990), *Variable size block matching motion compensation with applications to video coding*, IEE Proceedings I (Communications, Speech and Vision), IET, Vol.137, pp 205-2012.
6. Horowitz, Steven L and Pavlidis, Theodosios (1976),*Picture segmentation by a tree traversal algorithm*, Journal of the ACM (JACM), ACM, Vol.23, pp 368-388.
7. Mota, Virginia F and Souza, Jéssica IC and Araújo, Arnaldo de A and Vieira, Marcelo Bernardes (2013), *Combining Orientation Tensors for Human Action Recognition*, Conference on Graphics, Patterns and Images (SIBGRAPI), pp 328-333.
8. Sad, Dhiego and Mota, Virginia Fernandes and Maciel, Luiz Maurlio and Vieira, Marcelo Bernardes and Araújo, Arnaldo de A (2013), *A Tensor Motion Descriptor Based on Multiple Gradient Estimators*, Conference on Graphics, Patterns and Images (SIBGRAPI),pp 70-74.
9. Perez, Eder de Almeida and Mota, Virgnia Fernandes and Maciel, Luiz Maurlio and Sad, Dhiego and Vieira, Marcelo Bernardes (2012) , *Combining Gradient Histograms Using Orientation Tensors for Human Action Recognition*, 21st International Conference on Pattern Recognition (ICPR), pp

3460-3463.
10. Puri, A and Hang, HM and Schilling, DL (1987), *Interframe coding with variable block-size motion compensation*, Globecom, Vol. 87, pp 65-69.
11. Muralidhar, P and Rao, CB Rama and Kumar, I Ranjith (2012) , *Efficient architecture for variable block size motion estimation of h. 264 video encoder*, International Conference on Solid-State and Integrated Circuit (ICSIC), Vol.32.
12. Kim, JongWon and Lee, Sang Uk (1994),*Hierarchical variable block size motion estimation technique for motion sequence coding*, Visual Communications' 93, International Society for Optics and Photonics, pp 372-383.
13. Rhee, Injong and Martin, Graham R and Muthukrishnan, S and Packwood, Roger A (2000), *Quadtree-structured variable-size block-matching motion estimation with minimal error*, Circuits and Systems for Video Technology, IEEE Transactions on, Vol.10, pp 42-50.
14. Ji, Yanli and Shimada, Atsushi and Taniguchi, R-i (2010), *A Compact 3D Descriptor in ROI for Human Action Recognition*, IEEE TENCON, pp 454-459.
15. Mota, Virgínia Fernandes and Perez, Eder de Almeida and Vieira, Marcelo Bernardes and Maciel, LM and Precioso, Frédéric and Gosselin, Philippe Henri (2012), *A Tensor Based on Optical Flow for Global Description of Motion in Videos*, Conference on Graphics, Patterns and Images (SIBGRAPI), pp 298-301.
16. Alexander Kläser and Marcin Marszałek and Cordelia Schmid (2008), *A Spatio-Temporal Descriptor Based on 3D-Gradients*, British Machine Vision Conference (BMVC), September, pp 995-1004.
17. Mota, Virginia Fernandes and Perez, Eder de Almeida and Maciel, Luiz Maurlio and Vieira, Marcelo Bernardes and Gosselin, Philippe-Henri (2013), *A Tensor Motion Descriptor Based on Histograms of Gradients and Optical Flow*, Pattern Recognition Letters, Vol.31, pp 85-91.
18. Po, Lai-Man and Ma, Wing-Chung (1996), *A Novel Four-step Search Algorithm for Fast Block Motion Estimation*, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 3, pp 313-317.
19. Nie, Yao and Ma, Kai-Kuang (2002), *Adaptive Rood Pattern Search for Fast Block-matching Motion Estimation*, IEEE Transactions on Image Processing, Vol.11, pp1442-1449.
20. Zhu, Shan and Ma, Kai-Kuang (2000), *A New Diamond Search Algorithm for Fast Block-matching Motion Estimation*, IEEE Transactions on Image Processing, Vol.9, pp 287-290.
21. Johansson, Björn and Farnebäck, Gunnar (2002), *A theoretical comparison of different orientation tensors*, Proceedings SSAB02 Symposium on Image Analysis, Citeseer, pp 69-73.
22. Schuldt, Christian and Laptev, Ivan and Caputo, Barbara (2004), *Recognizing Human Actions: a Local SVM Approach*, Proceedings of the 17th International Conference on Pattern Recognition (ICPR), Vol.3, pp 32-36
23. Liu, Jingen and Luo, Jiebo and Shah, Mubarak (2009), *Recognizing Realistic Actions from Videos in the wild*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1996-2003
24. Marszalek, Marcin and Laptev, Ivan and Schmid, Cordelia (2009), *Actions in Context*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2929-2926.
25. Torralba, Antonio and Fergus, Robert and Weiss, Yair (2008) *Small Codes and Large Image Databases for Recognition*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
26. Wang, Heng and Kläser, Alexander and Schmid, Cordelia and Liu, Cheng-Lin (2013), *Dense Trajectories and Motion Boundary Descriptors for Action Recognition*, International Journal of Computer Vision, Vol.103, pp 60-79.
27. Wang, Heng and Schmid, Cordelia and others (2013), *Action recognition with improved trajectories*, International Conference on Computer Vision.