Compressed-Domain Based Camera Motion Estimation for Realtime Action Recognition

Huafeng Chen[1], Jun Chen[1,2(✉)], Hongyang Li[1], Zengmin Xu[1], and Ruimin Hu[1,2]

[1] National Engineering Research Center for Multimedia Software, School of Computer, Wuhan University, Wuhan, China
{chenhuafeng,chenj,lihy,xzm1981,hrm}@whu.edu.cn
[2] Collaborative Innovation Center of Geospatial Technology, Wuhan, China

Abstract. Camera motions seriously affect the accuracy of action recognition. Traditional methods address this issue through estimating and compensating camera motions based on optical flow in pixel-domain. But the high computational complexity of optical flow hinders these methods from applying to realtime scenarios. In this paper, we advance an efficient camera motion estimation and compensation method for realtime action recognition by exploiting motion vectors in video compressed-domain (a.k.a. compressed-domain global motion estimation, CGME). Taking advantage of geometric symmetry and differential theory of motion vectors, we estimate the parameters of camera affine transformation. These parameters are then used to compensate the initial motion vectors to retain crucial object motions. Finally, we extract video features for action recognition based on compensated motion vectors. Experimental results show that our method improves the speed of camera motion estimation by over 100 times with a minor reduction of about 4 % in recognition accuracy compared with iDT.

Keywords: Action recognition · Camera motion estimation · Compressed-domain

1 Introduction

Automatic human action recognition in video is an important and popular research area with potential applications in video analysis, video retrieval, video surveillance and human-computer interaction [6]. Recent research focuses on realistic datasets collected from surveillance videos, web videos, movies, TV shows, etc. These datasets impose significant challenges on action recognition due to camera motions and other fundamental difficulties. Camera motions abound in real-world video and seriously affect the accuracy of action recognition as they fire anywhere in the whole image and easily drown out the object motions.

Local space-time features [1–4,7–14] are shown to be successful on these datasets due to their aggregation of both spatial appearance feature and temporal motion feature. And some approaches [1–4] further consider to separate camera motions from the temporal motions to preserve defining object motions for

Y.-S. Ho et al. (Eds.): PCM 2015, Part I, LNCS 9314, pp. 85–94, 2015.
DOI: 10.1007/978-3-319-24075-6_9

action recognition. Wu et al. [2] apply a low-rank assumption to decompose feature trajectories into camera-induced and object-induced components. Recently, Park et al. [3] perform weak stabilization to remove both camera and object-centric motions using coarse-scale optical flow for pedestrian detection and pose estimation in video. Jain et al. [4] decompose visual motions into dominant and residual motions for extracting trajectories and computing descriptors. Wang et al. [7] introduce motion boundary histograms in Dense Trajectories (DT) to suppress camera motions, and further propose a camera motion estimation (a.k.a. global motion estimation, GME) method in improved Dense Trajectories (iDT) [1] to explicitly rectify the image to remove the camera motions. Benefited from double camera motion inhibition, iDT performs the best in action recognition accuracy among local space-time features.

While these methods [1–4] have improved the recognition accuracy through camera motion estimation and compensation on the basis of optical flow in pixel-domain, they are extremely time-consuming. For example, the speed of iDT ranges in the order of 3-4 frames per second (fps), which absolutely dissatisfies the requirements of realtime application. The main factor of their inefficiency is the pixel-domain based GME algorithm which must performs inefficient operation: OF calculation between adjacent frames. More seriously, some methods [1,3] calculate OF twice: once for GME, once for feature extraction.

To counteract the high computational complexity problem of local space-time feature, Kantorov et al. [5] make an effective attempt to accelerate the method of DT through replacing OF with motion vectors (a.k.a. MPEG flow, MF) which are obtained from video compressed-domain. The replacement of OF with MF for video feature extraction eliminates the calculation process of OF, thus the method of [5] improves the speed of feature extraction by two orders of magnitude at the cost of minor reduction in recognition accuracy compared with DT. Unfortunately, Kantorov et al. have not considered the interference of camera motions in MF.

In order to compensate the influence of camera motion and accelerate the feature extraction process, in this paper we propose a camera motion estimation and compensation method in the compressed-domain (a.k.a. compressed-domain global motion estimation, CGME), avoiding the OF calculation in pixel-domain. Figure 1 presents the comparison of proposed CGME with traditional GME for action recognition. Based on MF, we estimate camera affine transformation parameters by making use of geometric symmetry and differential theory of motion vector [15]. According to the estimated parameters, we compensate initial MF to retain crucial object motions for action recognition. We extract video feature descriptors based on compensated MF by following the method of [5]. Then, we evaluate the speed and accuracy of our approach on UCF50 [16] and HMDB51 [17] benchmarks. Experimental results show that our method improves the speed of GME by over 100 times with a minor reduction of about 4 % in recognition accuracy compared with iDT. It is proved that the proposed approach completely meets the requirements of realtime action recognition.

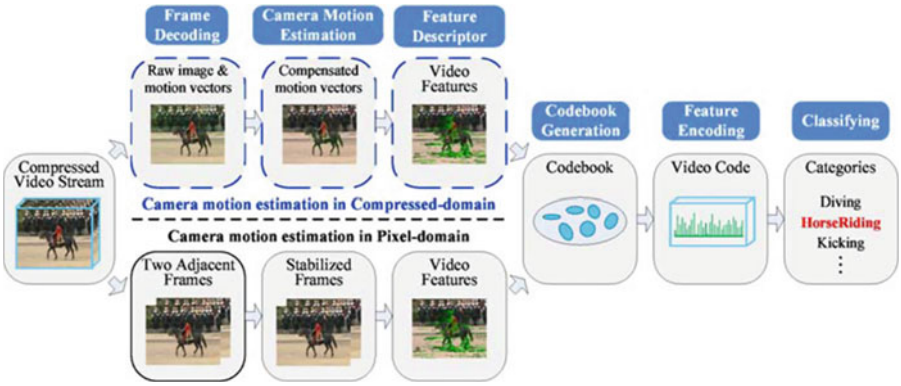

Fig. 1. Comparison of proposed approach to traditional approach for action recognition. **Top**: Pipeline of proposed CGME and feature extraction in compressed-domain. **Bottom**: Pipeline of traditional GME and feature extraction in pixel-domain.

2 Proposed Method

In this section, we first describe the 4-parameter camera affine transformation model for CGME. And then, we estimate the parameters respectively by using initial MF in video compressed-domain. Based on the estimated parameters, we discuss how to rectify the initial MF to eliminate the interference of camera motions. Finally, we extract the video features based on the revised MF by following the method of [5].

2.1 Camera Model

We define the 2D coordinate system of the image for CGME in the first place. The center of the 2D image corresponds to the coordinate origin, the positive direction of **x**-axis to the right, the positive direction of **y**-axis downward, and the image is divided into four quadrants respectively: I quadrant (bottom-right), II quadrant (bottom-left), III quadrant (top-left) and IV quadrant (top-right) (Shown in Fig. 2). Taking any pixel from I quadrant, the spatial coordinates is defined as $z_I = (x, y)^T (x > 0, y > 0)$, it surely determines the symmetry points in other three quadrants: $z_{II} = (-x, y)^T$, $z_{III} = (-x, -y)^T$, and $z_{IV} = (x, -y)^T$.

Based on the 2D coordinate system, we adopt 4-parameter camera affine transformation model for modeling the camera motion [15]. This 4-parameter model can faultlessly model camera translation, scaling, rotation, and their combinations. It is defined by

$$f(z|A,T) = Az + T = \begin{pmatrix} a_1 & -a_2 \\ a_2 & a_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}, \tag{1}$$

where a_1 and a_2 are parameters reflecting scaling and rotation changes in motion, t_x and t_y control translation parameters, $(x, y)^T$ is the point in the image.

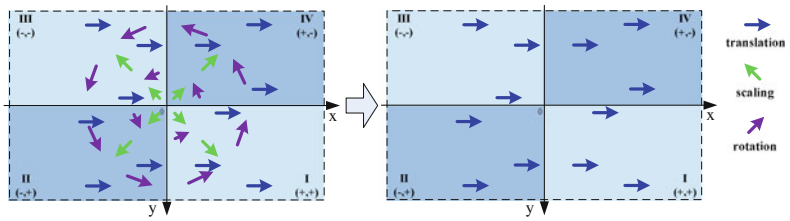

Fig. 2. Example of symmetrical counteraction of scaling/rotation motion vectors.

Thus, a camera motion vector at any point z in image can be expressed as

$$MV(z) = f(z|A,T) - z = (A-I)z + T. \tag{2}$$

2.2 Estimation of T

A camera motion vector can be decomposed into translation, scaling and rotation sub-vectors. The scaling/rotation sub-vectors in image possess the characteristics of symmetrical around the origin. So, any two symmetrical scaling/rotation sub-vectors around the origin can counteract each other (we call it symmetrical counteraction), that is, the sum of them is zero (Shown in Fig. 2).

We make use of the symmetrical counteraction to estimate the parameter T which controls translation sub-vectors. Given two symmetrical around the origin motion vectors $MV(z_I)$ $(z_I = (x,y)^T)$ in I quadrant sector and $MV(z_{III})$ $(z_{III} = (-x,-y)^T)$ in III quadrant sector, we sum them and can get translation parameter T. The sum equation is

$$\begin{aligned} MV(z_I) + MV(Z_{III}) &= f(z_I|A,T) - z_I + f(z_{III}|A,T) - z_{III} \\ &= (A-I)(z_I + z_{III}) + 2T \\ &= 2T. \end{aligned} \tag{3}$$

Similarly, we can calculate translation parameter T by summing two symmetrical around the origin motion vectors $MV(z_{II})$ $(z_{II} = (-x,y)^T)$ in II quadrant sector and $MV(z_{IV})$ $(z_{IV} = (x,-y)^T)$ in IV quadrant sector,

$$MV(z_{II}) + MV(Z_{IV}) = 2T. \tag{4}$$

By applying Eqs. (3) and (4) on initial MF, we can calculate out a set of initial T parameters $T_{init} = \{T_1, T_2, ..., T_N\}$. Ideally, the values in the set T_{init} are equal $(T_1 = T_2 = ... = T_N)$ under the environments without any object motions except camera motions. But generally, camera motions and object motions are mixed together in real videos. So the values in T_{init} are not exactly equal, and we are not sure which truly reflects the real translational motion in T_{init}. To estimate the parameter T from T_{init}, we adopt the mean distance threshold determination algorithm [18]. As can be seen from Algorithm 1, we calculate the mean of all

Algorithm 1. Estimation of parameter T from T_{init}.

Require:
$T_{init} = \{T_1, T_2, ..., T_N\}$;
Ensure:
T_{esti};
1: $T_{mean} \Leftarrow (T_1 + T_2 + \cdots + T_N)/N$
2: **for** $i = 1$ to N **do**
3: $R_i \Leftarrow |T_i - T_{mean}|$
4: **end for**
5: $R_{mean} \Leftarrow (R_1 + R_2 + \cdots + R_N)/N$
6: $count \Leftarrow 0$
7: **for** $i = 1$ to N **do**
8: **if** $R_i > R_{mean}$ **then**
9: $T_i \Leftarrow 0$
10: $count \Leftarrow count + 1$
11: **end if**
12: **end for**
13: $T_{esti} = (T_1 + T_2 + \cdots + T_N)/count$
14: **return** T_{esti}

elements in T_{init} firstly, and compute the absolute residuals of all the data based on the mean value. We put the mean of these residuals as a threshold, and weed out the outliers from T_{init} according to whether the element's residual is greater than the threshold. We calculate the mean of rest elements in T_{init}, and put it as the final estimated parameter T_{esti}.

2.3 Estimation of a

We can use the differential principle of motion vectors to calculate the parameter A. Firstly, we deduce general equations of the differences of motion vectors. Given two pixels located on the same line $z_1 = (i_1, c_y)^T$ and $z_2 = (i_2, c_y)^T$ ($i_2 = i_1 + s_x$). By Eq. (2), we can get

$$\begin{aligned} & MV_x(z_2) - MV_x(z_1) \\ & = (f_x(z_2|A,T) - i_2) - (f_x(z_1|A,T) - i_1) \\ & = ((a_1 - 1) \times i_2 - a_2 \times c_y + t_x) - ((a_1 - 1) \times i_1 - a_2 \times c_y + t_x) \\ & = (a_1 - 1)(i_2 - i_1) \\ & = (a_1 - 1) \times s_x. \end{aligned} \tag{5}$$

Thus, we can obtain a differential equation on a_1 parameters,

$$a_1 = \frac{MV_x(z_2) - MV_x(z_1)}{s_x} + 1. \tag{6}$$

Similarly, we can get a differential equation with respect to a_2 parameters,

$$a_2 = \frac{MV_y(z_2) - MV_y(z_1)}{s_x}. \tag{7}$$

Given two pixel coordinates on the same column $z_3 = (c_x, j_1)^T$ and $z_4 = (c_x, j_2)^T$ ($j_2 = j_1 + s_y$), according to the above mentioned, we can get another set of differential equations on the parameters a_1 and a_2,

$$a_2 = -\frac{MV_x(z_4) - MV_x(z_3)}{s_y}, \tag{8}$$

$$a_1 = \frac{MV_y(z_4) - MV_y(z_3)}{s_y} + 1. \tag{9}$$

By applying Eqs. (6), (7), (8) and (9) on initial motion vectors, we can calculate out two sets of initial A parameters $a_{1init} = \{a_{11}, a_{12}, ..., a_{1M}\}$ and $a_{2init} = \{a_{21}, a_{22}, ..., a_{2K}\}$. We estimate parameter A_{esti} based on a_{1init} and a_{2init} by following the algorithm of parameter T estimation in Sect. 2.2.

2.4 Camera Motion Compensation

According to the estimated parameters T_{esti} and A_{esti}, we compensate the initial video motion vectors by

$$\begin{aligned} MV'(z) &= MV(z) - estiGM(z) \\ &= MV(z) - (Az + T) \\ &= MV(z) - \left(\begin{pmatrix} a_1 & -a_2 \\ a_2 & a_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \right), \end{aligned} \tag{10}$$

where $MV(z)$ is the initial MF in compressed-domain, $estiGM(z)$ is the estimated camera motion vectors, and $MV'(z)$ is the compensated motion vectors that preserved the defining object motions for action recognition.

2.5 Feature Descriptor Extraction

We follow the design of previously proposed local space-time descriptors [5] and define our descriptor by histograms of the compensated motion vectors in a video patch. We compute HOF descriptors as histograms of compensated motion vectors discretized into eight orientation bins and a non-motion bin. For MBHx and MBHy descriptors the spatial gradients of the v_x and v_y components of the compensated motion vectors are similarly descretized into nine orientation bins. The final descriptor is obtained by concatenating histograms from each cell of the 2×2×3 descriptor grid followed by l_2-normalization of every temporal slice. HOG descriptors are computed at the same sparse set of points.

3 Experimental Results

In this section we evaluate the proposed approach on two publicly available datasets, the UCF50 [16] and HMDB51 [17] (see Fig. 3). The UCF50 dataset has 50 action categories, consisting of real-world videos taken from YouTube. The actions range from general sports to daily life exercises. For all 50 categories, the videos are split into 25 groups. For each group, there are at least 4 action clips. In total, there are 6,618 video clips in UCF50. The HMDB51 dataset is collected from a variety of sources ranging from digitized movies to YouTube videos. There are 51 action categories and 6,766 video sequences in HMDB51.

We compare the speed and action recognition accuracy of the proposed approach to recent methods [1,5]. We follow satandard evaluation setups and report

Fig. 3. Sample frames of standard datasets. Top: UCF50 [16], Bottom: HMDB51 [17].

mean accuracy (Acc) for UCF50 and HMDB51 datasets. The processing speed is reported in frames-per-second (Fps), run at a single-core Intel Xeon X3430 (2.4 GHz) with no multithreading.

To recognize actions, we follow [1,5] to train a GMM model with $K = 256$ Gaussians. Each video is, then, represented by a $2DK$ dimensional Fisher vector for each descriptor type (HOG, HOF, MBHx and MBHy), where D is the descriptor dimension. Finally, we apply l_2-normalization to the Fisher vector. To combine different descriptor types, we concatenate their normalized Fisher vectors. A linear SVM is used for classification.

3.1 GME Evaluation

Table 1 presents action recognition accuracy and speed of the proposed GME approach compared to the GME method adopted by iDT. The performance of iDT (90.9 % in UCF50 and 55.6 % in HMDB51) is approximately four percent higher compared to our proposed approach.

Table 1. Comparison of action classification accuracy and the speed of proposed CGME to GME of iDT. The speed is reported for video of spatial resolution 320×240 pixels on UCF50 and 360×240 pixels on HMDB51.

	Classification (Acc)		Speed (Fps)	
	CGME(Proposed)	GME(iDT)	CGME(Proposed)	GME(iDT)
UCF50	86.3 %	90.9 %	**853.5**	6.5
HMDB51	51.9 %	55.6 %	**912.3**	6.7

When comparing the speed of GME for both methods in UCF50, our CGME method achieves 853.5 fps which is about 24 times faster than real-time and 131 times faster compared to iDT [1]. And when comapring the speed of GME for both methods in HMDB51, our CGME method achieves 912.3 fps which is about 36 times faster than real-time and 136 times faster compared to iDT. From Table 1 we can see, the runtime of proposed CGME method is < 1 % of GME in iDT by avoiding motion vector calculation, and can fully meets the requirements of real-time applications.

3.2 Feature Descriptor Evaluation

We compare our descriptor to iDT [1] and MF [5] on the UCF50 and HMDB51. iDT performs the best in action recognition accuracy, while MF is the fasted algorithm among all existing local space-time descriptor methods.

Proposed method vs iDT [1] - Table 2 presents action recognition accuracy and speed of the proposed approach compared to iDT. The action recognition accuracy of iDT (90.9 % in UCF50 and 55.6 % in HMDB51) is approximately 4 percent higher compared to proposed descriptor (86.3 % in UCF50 and 51.9 % in HMDB51). When comparing the speed of feature extraction for both methods, our method (514.1 fps in UCF50 and 582.2 fps in HMDB51) is far faster than iDT (3.7 fps in UCF50 and 3.9 fps in HMDB51) because proposed approach works in compressed-domain and keeps away from inefficient OF calculation.

Table 2. Comparison of action classification accuracy and speed of proposed feature descriptor to iDT [1].

	Classification (Acc)		Speed (Fps)	
	Proposed	iDT [1]	Proposed	iDT [1]
UCF50	86.3 %	90.9 %	**514.1**	3.7
HMDB51	51.9 %	55.6 %	**582.2**	3.9

Table 3. Comparison of action classification accuracy and speed of proposed feature descriptor to MF [5].

	Classification (Acc)		Speed (Fps)	
	Proposed	MF [5]	Proposed	MF [5]
UCF50	**86.3**%	82.2 %	514.1	698.4
HMDB51	**51.9**%	46.7 %	582.2	752.2

Proposed method vs MF [5] - From Table 3 we can see, the action recognition accuracy of proposed descriptor method (86.3 % in UCF50 and 51.9 % in HMDB51) is approximately 5 percent higher compared to MF feature (82.2 % in UCF50 and 46.7 % in HMDB51). The reason of accuracy increasement between MF and proposed descriptor is that the method of proposed GME significantly inhibite the camera motions (shown in Fig. 4). While the speed of proposed method (514.1 fps in UCF50 and 582.2 fps in HMDB51) is a little slower than MF (698.4 fps in UCF50 and 752.2 fps in HMDB51), it also can meet the needs of realtime processing because it is about 21 times faster than realtime.

Fig. 4. Comparison of initial motion vectors and compensated motion vectors. Left: (a)-UCF50 initial vectors (b)-UCF50 compensated vectors, Right: (a)-HMDB51 initial vectors (b)-HMDB51 compensated vectors. Green Point: motion start point from previous frame, Green Line: motion from the start point to end point in current frame (Color figure online).

4 Conclusion

In this work, we present a method called CGME for realtime action recognition, different from recent mainstream methods. The core idea is: we make full use of motion vectors in compressed domain for GME and feature extraction to avoid inefficient OF calculation in pixel domain. Taking advantage of geometric symmetry and differential theory of motion vectors, we estimate the parameters of camera affine transformation and compensate the initial motion vectors based on the estimated parameters. The proposed method is proved to be more efficient than iDT and completely suitable for realtime action recognition.

Acknowledgement. The research was supported by National Nature Science Foundation of China (No. 61231015), National High Technology Research and Development Program of China (863 Program, No. 2015AA016306), National Nature Science Foundation of China (61170023), Internet of Things Development Funding Project of Ministry of industry in 2013(No. 25), Technology Research Program of Ministry of Public Security (2014JSYJA016), Major Science and Technology Innovation Plan of Hubei Province (2013AAA020), and Nature Science Foundation of Hubei Province (2014CFB712).

References

1. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: IEEE International Conference on Computer Vision (ICCV) (2013)

2. Wu, S., Oreifej, O., Shah, M.: Action recognition in videos acquired by a moving camera using motion decomposition of lagrangian particle trajectories. In: IEEE International Conference on Computer Vision (ICCV) (2011)
3. Park, D., Zitnick, C.L., Ramanan, D., Dollr, P.: Exploring weak stabilization for motion feature extraction. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
4. Jain, M., Jgou, H., Bouthemy, P.: Better exploiting motion for better action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
5. Kantorov, V., Laptev, I.: Efficient feature extraction, encoding, and classification for action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
6. Aggarwal, J.K., Ryoo, M.S.: Human activity analysis: A review. ACM Computing Surveys (CSUR) (2011)
7. Wang, H., Klaser, A., Schmid, C., Liu, C.-L.: Action recognition by dense trajectories. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011)
8. Wang, H., Klaser, A., Schmid, C., Liu, C.-L.: Dense trajectories and motion boundary descriptors for action recognition. International Journal of Computer Vision (IJCV) (2013)
9. Laptev, I.: On space-time interest points. International Journal of Computer Vision (IJCV) (2005)
10. Scovanner, P., Ali, S., Shah, M.: A 3-dimensional sift descriptor and its application to action recognition. In: ACM International Conference on Multimedia (ACM MM) (2007)
11. Willems, G., Tuytelaars, T., Van Gool, L.: An efficient dense and scale-invariant spatio-temporal interest point detector. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 650–663. Springer, Heidelberg (2008)
12. Klaser, A., Marszalek, M.: A spatio-temporal descriptor based on 3d-gradients. In: British Machine Vision Conference (BMVC) (2008)
13. Yeffet, L., Wolf, L.: Local trinary patterns for human action recognition. In: IEEE International Conference on Computer Vision (ICCV) (2009)
14. Chen, M., Hauptmann, A.: Mosift: Recognizing human actions in surveillance videos (2009)
15. Zheng, Y., Tian, X., Chen, Y.: Fast global motion estimation based on symmetry elimination and difference of motion vectors. Journal of Electronics & Information Technology (2009)
16. Reddy, K., Shah, M.: Recognizing 50 human action categories of web videos. In: Machine Vision and Applications (MVA) (2012)
17. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: Hmdb: A large video database for human motion recognition. In: IEEE International Conference on Computer Vision (ICCV) (2011)
18. Chen, H., Liang, C., Peng, Y., Chang, H.: Integration of digital stabilizer with video codec for digital video cameras. IEEE Transactions on Circuits and Systems for Video Technology (TCSVT) (2007)