A ROBUST VIDEO STABILIZATION SYSTEM BY ADAPTIVE MOTION VECTORS FILTERING

S. Battiato, G. Puglisi

University of Catania
Dipartimento di Informatica
Viale A. Doria 6, Catania, Italy

A. R. Bruna

STMicroelectronics Catania
AST Imaging Group

ABSTRACT

Digital video stabilization allows to acquire video sequences without disturbing jerkiness, removing unwanted camera movements. In this paper we propose a novel fast video stabilization algorithm based on block matching of local motion vectors. Some of these vectors are properly filtered out by making use of ad-hoc rules taking into account local similarity, local "activity" and matching effectiveness. Also a temporal analysis of the relative error computed at each frame has been achieved. Reliable information are then used to retrieve inter-frame transformation parameters. Experiments on real cases confirm the effectiveness of the proposed approach even in critical conditions.

Index Terms— Video Stabilization, Motion Estimation, Block Matching

1. INTRODUCTION

In the last years video stabilization techniques have gained consensus, for they permit to obtain high quality and stable video footages even in non-optimal conditions. The best techniques, by using some mechanical tools, measure camera shake and then control the jitter acting on lens or on the CCD/CMOS sensor [1]. On the other hand, digital video stabilization techniques make use only of information drawn from footage images and do not need any additional knowledge about camera physical motion. Furthermore, these approaches may be implemented easily both in real-time and post-processing systems. Video stabilization systems have been widely investigated and several techniques have been proposed, with different issues and weak points. Optical flow based techniques [2] have low computational complexity (optical flow estimation is fast) but, without multi-scale approaches, they are not able to deal with large displacements. Feature-based algorithms extract features from video images and estimate inter-frame motion using their location. Some authors present techniques [3, 4] combining features computation with other robust filters; these methods have gained larger consensus for their good performances but features computation step can be very time consuming. Global intensity alignment approaches [5] compute inter-frame transformation considering image intensity directly. These techniques are generally less sensitive to local effects but usually have high computational cost. Finally block matching based techniques use different adaptive filters to refine motion estimation from block local vectors [6]. These algorithms generally provide good results and their computation complexity depends on block matching step.

In this paper we introduce a novel block based video stabilization technique. Starting from local vectors, through simple and fast rejection rules, our algorithm computes inter-frame transformation parameters. The rejection criteria are based on local blocks similarity, local blocks "activity" and matching effectiveness. Also a temporal analysis of the involved motion vectors allows to improve the overall robustness of the method. The proposed approach, devoted to work in real-time environment, obtains effective experimental results even in critical conditions (illumination changes, moving objects, image blur). The main novelty of the method is related to the choice of pre-filtering criteria, the temporal analysis of block motion vectors coupled with a fairly robust estimator. The rest of the paper is organized as follows. In Section 2 the overall details of the proposed system are presented. Experimental results are shown in Section 3 while conclusions are summarized in Section 4.

2. PROPOSED SYSTEM

The stabilization algorithm we developed consts of the following modules (Figure 1):

1. BMA (Block Matching Algorithm): Starting from a pair of consecutive frames it computes the local motion vectors through a BM (block matching) algorithm. We call these vectors V_{BMA}.

2. Pre-filtering: It filters V_{BMA} vectors through some simple rules based on the goodness of matching, block homogeneity and near vectors similarity. We call these filtered vectors V_{pf}.

3. Memory filter: By making use of some information collected at previous iteration, V_{pf} are properly filtered. A fixed percentage of the overall new vectors (not belonging to V_{pf} at time $t-1$) together with vectors too far respect to previous estimation are filtered out. We call these filtered vectors V_f.

4. Robust estimation: It estimates the inter-frame transformation parameters through some Least Squares iterations.

5. Error vector M_e computing: For each V_{pf} element it computes an error based on the inter-frame transformation parameters.

In Figure 2 an example of vector filtering is reported. Only reliable motion vectors (V_f) are retained.

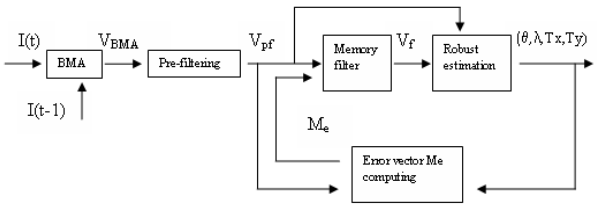

Fig. 1. Motion estimation algorithm architecture. Starting from a pair of consecutive frames, it computes inter-frame transformation parameters.

2.1. Pre-filtering

BMA module tipically computes many wrong motion vectors. To filter out these vectors, not useful for global motion estimation, we make use of the following considerations:

- The SAD (sum of absolute difference) values have to be low (effective match).
- Local motion vectors have to share in their neighbourhood similar values (motion continuity).
- Local motion vectors referred to homogeneous blocks are not reliable.

The above rules have been derived after an exhaustive experimental phase devoted to achieve a suitable trade-off between overall complexity and real-time constraints. According to the previous consideration we have derived four indexes: SAD (goodness of matching), NS (neighbourhood similarity), $Unhom_x$ and $Unhom_y$ (gradient information along x and y axes respectively). Pre-filtering module filters local motion vectors computing first the values SAD, NS, $Unhom_x$, $Unhom_y$. All the vectors with both $Unhom_x$ and $Unhom_y$ less than th_{Hom} (a threshold experimentally fixed) are filtered out. Starting from V_{BMA} vectors V_{pf2} vectors are produced. V_{pf2} vectors are then sorted in ascending order according to SAD (and NS) values, labeling also a percentage p_{pf1} (p_{pf2}) of vectors with high SAD (NS) values as deleted. Moreover V_{pf2} vectors are sorted in descending order according to $UnHom_x$ (and $UnHom_y$) values, labeling a percentage p_{pf3} of vectors with low $Unhom_x$ ($Unhom_y$) values as deleted. Finally the labeled vectors are rejected obtaining as final output a set of vectors V_{pf}.

2.2. Memory Filter

All the filtering above described are devoted to eliminate all the wrong vectors computed by a generic BM algorithm. However, due to moving objects in the scene, there are vectors correctly computed by BM (hence not filtered by pre-filtering-module) that must be deleted in order to have a good inter-frame parameters estimation. If the moving objects are not very big, their vectors probably will be rejected by robust estimation module. On the contrary if the moving objects are very big (even greater than 50% of the scene), single image information is not enough to filter out these vectors. In order to make robust our algorithm to such situation, memory filter module uses previous frames information. All the V_{pf} vectors are split into V_{pfnew} (vectors not belonging to V_{pf} at the previous iteration $t-1$) and V_{pfOld} (vectors belonging to V_{pf} at the previous iteration $t-1$). A percentage p_{mf1} of V_{pfnew} vectors is then rejected. They are considered less reliable than V_{pfOld} vectors. We call the remaining vectors (not rejected) V_{pfnewF}. According to $M_e(t-1)$ vector error values, V_{pfOld} vectors are sorted and filtered out (a percentage p_{mf2} with high error). We call the remaining vectors (not rejected) V_{pfOldF}. Finally V_{pfnewF} and V_{pfOldF} vectors are merged together into V_f.

2.3. Robust Estimation

Global motion between adjacent frames can be estimated with a two-dimensional linear model, usually a good trade-off between effectiveness and complexity. This model describes inter-frame motion using four different parameters, namely two shifts, one rotation angle and a zoom factor, and it associates a point (x_i, y_i) in frame I_n with a point (x_f , y_f) in frame I_{n+1} with the following transformation:

$$\begin{cases} x_f = x_i \lambda \cos\theta - y_i \lambda \sin\theta + T_x \\ y_f = x_i \lambda \sin\theta + y_i \lambda \cos\theta + T_y \end{cases} \quad (1)$$

where λ is the zoom parameter, θ the rotation angle, Tx and Ty respectively X-axis and Y-axis shifts. Vectors computation may be affected by noise so it is useful to apply a linear Least Squares Method on a set of redundant equations. The whole set of local motion vectors probably includes wrong matches or correct matches belonging to self-moving objects in the filmed scene. Obviously there are some correct pairs that do represent real camera shakes but several points simply

(a) (b)

Fig. 2. Overall vectors obtained by BMA (16x16 blocks) (a) and residual vectors after the filtering process (b).

do not relate to such information. Least Squares Method does not perform well when there is a large portion of outliers in the total number of features, as in this case. However, outliers can be identified, filtered out of the estimation process, resulting in better accuracy [7]. In order to obtain real-time performances we have implemented a fast rejection technique:

- Starting from V_f values it computes a first Least Squares estimation of the inter-frame transformation parameters ($\lambda_1, \theta_1, Tx_1, Ty_1$).
- For each V_{pf} element it computes an error E given by the following formula:

$$E = \frac{\sqrt{e_x^2 + e_y^2}}{Param + mNorm} \quad (2)$$

$$e_x = x_s - x_f \quad (3)$$

$$e_y = y_s - y_f \quad (4)$$

$$normV1 = \sqrt{(x_f - x_i)^2 + (y_f - y_i)^2} \quad (5)$$

$$normV2 = \sqrt{(x_s - x_i)^2 + (y_s - y_i)^2} \quad (6)$$

$$mNorm = \frac{normV1 + normV2}{2} \quad (7)$$

  where (x_i, y_i) is the center of a block (related to a vector) in the frame $t-1$, (x_f, y_f) is the position of the block at frame t computed by BMA, (x_s, y_y) is the position estimated according to inter-frame parameters ($\lambda_1, \theta_1, Tx_1, Ty_1$) and $Param$ is a constant.

- According to the error E, it sorts all V_{pf} elements in increasing order and filters out a percentage p_{re} with high error value. We call the remaining vector (not filtered out) V_s.
- Starting from V_s values it computes the Least Squares estimation of the inter-frame transformation parameters (λ, θ, Tx, Ty).

When there are rotational movement, error E gives better results than Euclidean distance. When a rotation occurs, vectors close to the border have norm greater than vector close to the center and, due to the quantization (an image is a grid of pixels), border vectors give better accuracy than the others in the rotational estimation. $Param$ parameter is introduced to avoid problems with low (or even null) $mNorm$ values.

2.4. Error Vector M_e Computing

For each V_{pf} element it computes an error (Euclidean Distance) used to fill the M_e error. This metric is simple to compute and able to efficiently distinguish between vectors belonging to objects entering in the scene and vectors describing the scene movements. Usually $V_{pf}(t)$ contains elements corresponding to blocks that were not present in $V_{pf}(t-1)$. Hence some vectors have not an M_e entry when they are considered in the Memory filter. In order to partially solve this problem, we propagate error vectors value to its neighbourhood, simply coping each error value into its neighbours with no value associated. The algorithm is computed from left to right, from up to down.

3. EXPERIMENTAL RESULTS

In order to confirm the effectiveness of our approach a series of tests have been conducted on real videos captured by hand held digital camera in critical video stabilization conditions (illumination changes, moving objects, variable zooming factor and forward walking of the user). All sequences have been acquired with an high-end camera with high quality settings (without blocking artifacts). Frame resolution size of each sequence has been of 762x512. In our experiments we have considered only the luminance channel. The same set of involved parameters (th_{Hom}, p_{pf1}, p_{pf2}, p_{pf3}, p_{mf1}, p_{mf2}, p_{re}, $Param$), derived after an exhaustive experimental phase, has been used in all cases. All the motion vectors have been computed through SLIMPEG (Substantially Light Motion Picture Estimator for mpeG) [8]. For evaluation we have used the ITF (Interframe Transformation Fidelity) measure [9]:

$$ITF = \frac{1}{N_{frame} - 1} \sum_{k=1}^{N_{frame}-1} PSNR(k) \quad (8)$$

$$PNSR(k) = 10 \log_{10} \frac{I_{max}^2}{MSE(k)} \quad (9)$$

where $MSE(k)$ is the Mean Square Error between consecutive frames, I_{max} is the maximum intensity value of a pixel and N_{frame} is the video frames number. Typically, a stabilized sequence has higher ITF values than original sequence. Table 1 shows ITF values for original sequences, stabilized sequences with a simple Least Squares estimator, with RANSAC [7] and with our method. Both simple Least Squares estimator and RANSAC compute interframe transformation parameters considering all BMA vectors (V_{BMA}) without any filtering. Although realized in critical conditions (even motion blur is present in the videos), all the sequences have been effectively stabilized obtaining an average gain, with respect to the original sequences, of 6.3 dB. Moreover, our approach outperforms both simple Least Squares estimator, that is not able to deal with moving objects, and RANSAC that shows its limits in presence of a large number of dissimilar outliers (typically due to wrong vectors produced by BMA). To visually assess the performances of the proposed system both original and stabilized sequences can be downloaded from

`http://www.dmi.unict.it/iplab/download/video-stab`

Sequence	Original ITF (dB)	Least Squares ITF (dB)	RANSAC ITF (dB)	Our alg. ITF (dB)
illumination changes + zoom	28.2	30.8	34.4	**36.8**
moving objects	27.4	29.1	**32.7**	**32.7**
forward walking of the user	24.9	29.5	29.7	**30.0**

Table 1. ITF on original sequences, stabilized sequences with a simple Least Squares estimator, with RANSAC and with our method.

For sake of comparison we have measured the ITF of our system with a recent features based technique [3] on 3 videos kindly provided us by the authors (Table 2). In simple conditions like a book over a table, algorithm [3] performs better than ours. On the contrary in critical conditions like a car moving in the scene or a man walking in the snow (matching algorithms have many problems), our algorithm performances are better.

Sequence	Original ITF (dB)	Alg. [3] ITF (dB)	Our alg. ITF (dB)
book	24.1	**33.7**	32.6
car	20.9	24.1	**26.7**
walk	17.6	24.9	**28.0**

Table 2. ITF comparison between our technique and [3].

4. CONCLUSIONS

In this paper we have proposed a novel block based video stabilization technique. Our algorithm starting from local vectors, through simple and fast rejection rules, computes interframe transformation parameters. The effectiveness of our approach have been demonstrated through a series of experiment in critical conditions (illumination changes, moving objects, image blur) and comparisons with some standard and recent techniques. Future work will be devoted to find some strategies able to make our algorithm robust in presence of periodic patterns.

5. REFERENCES

[1] Canon Inc., "Canon faq: What is vari-angle prism?," http://www.canon.com/bctv/faq/vari.html.

[2] J. Chang, W. Hu, M. Cheng, and B. Chang, "Digital image translational and rotational motion stabilization using optical flow technique," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 1, February 2002.

[3] J. Yang, D. Schonfeld, C. Chen, and M. Mohamed, "Online video stabilization based on particle filters," in *Proceeding of IEEE ICIP*. Atlanta (USA), October 2006.

[4] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato, "SIFT features tracking for video stabilization," in *Proceeding of ICIAP*. Modena (Italy), September 2007, pp. 825–830.

[5] M. Tico, S. Alenius, and M. Vehvilainen, "Method of motion estimation for image stabilization," in *Proceeding of IEEE ICASSP*. Toulouse (France), May 2006.

[6] S. Auberger and C. Miro, "Digital video stabilization architecture for low cost devices," in *Proceedings of the 4th International Symp. on Image and Signal Processing and Analysis*. Zagreb (Croatia), September 2005.

[7] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24(6), pp. 381–395, 1981.

[8] F.S. Rovati, D. Pau, E. Piccinelli, L. Pezzoni, and J.-M. Bard, "An innovative, high quality and search window independent motion estimation algorithm and architecture for mpeg-2 encoding," *IEEE Transactions on Consumer Electronics*, vol. 46, no. 3, pp. 697–705, August 2000.

[9] L. Mercenaro, G. Vernazza, and C. Regazzoni, "Image stabilization algorithms for video-surveillance application," in *Proceedings of IEEE ICIP*. Thessaloniki (Greece), October 2001.