Pattern Recognition 44 (2011) 2887–2901

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Fast and accurate global motion compensation

O. Deniz [a,*], G. Bueno [a], E. Bermejo [a], R. Sukthankar [b]

[a] E.T.S. Ingenieros Industriales, Universidad de Castilla-La Mancha, Spain
[b] Intel Labs Pittsburgh and Robotics Institute, Carnegie Mellon, USA

ARTICLE INFO

Article history:
Received 6 April 2010
Received in revised form
17 September 2010
Accepted 24 October 2010
Available online 3 November 2010

Keywords:
Global motion estimation
Action recognition

ABSTRACT

Video understanding has attracted significant research attention in recent years, motivated by interest in video surveillance, rich media retrieval and vision-based gesture interfaces. Typical methods focus on analyzing both the appearance and motion of objects in video. However, the apparent motion induced by a moving camera can dominate the observed motion, requiring sophisticated methods for compensating for camera motion without a priori knowledge of scene characteristics. This paper introduces two new methods for global motion compensation that are both significantly faster and more accurate than state of the art approaches. The first employs RANSAC to robustly estimate global scene motion even when the scene contains significant object motion. Unlike typical RANSAC-based motion estimation work, we apply RANSAC not to the motion of tracked features but rather to a number of segments of image projections. The key insight of the second method involves reliably classifying salient points into foreground and background, based upon the entropy of a motion inconsistency measure. Extensive experiments on established datasets demonstrate that the second approach is able to remove camera-based observed motion almost completely while still preserving foreground motion.

1. Introduction

Advances in computer vision have made it possible to tackle previously difficult problems such as action recognition in video. The implicit objective is to classify human actions by analyzing the motion of body parts. Applications include automatic video annotation (particularly for sports and news), human–computer interaction, games, etc. The recent widespread interest in video surveillance has also brought increased attention to the action recognition problem. Most work in action recognition deals with sequences acquired by stationary cameras with fixed viewpoints (see [1]). Due to camera motion, however, the trajectories of the body parts contain not only the motion of the performing actor but also the motion of the camera. This paper deals with camera motion compensation in the context of action recognition.

Global (camera) motion is typically modeled by either a full projective homography or an affine homography, assuming distant views and planar scenes. Motion compensation methods may be divided into feature-based or direct (featureless) [2]. Feature-based methods are based on a set of salient features whose motion is tracked from frame to frame, whereas direct methods attempt to estimate global motion directly from image intensities.

Direct or image-based methods rely on a direct transformation of the image grid and minimize some image difference criterion [3–5]. However, since the homography parameters are related to the image intensities in a highly nonlinear way, some direct methods use complex and computationally intensive optimization algorithms. Feature-based methods use tracked salient points to estimate the homography or affine transformation describing frame-to-frame motion [6–8]. Given a set of point correspondences, a least-squares (LS) estimate of the homography parameters can be obtained. RANSAC [9] is often used instead of LS to obtain more robust estimates when the point correspondences are noisy.

In the context of an action recognition application, Zhu et al. [10] remove camera motion by continuously tracking the object of interest (a tennis player). A similar approach is used in Panagiotakis et al. [11] for recognizing actions of athletes, exploiting camera motion as an additional feature for recognition of the action. Since the motion of salient points is often used as the main descriptor for modeling actions and efficient trackers are readily available, feature-based methods have been more widely used in the context of the action recognition problem. In Mikolajczyk and Uemura [12], for example, salient point correspondences are used along with RANSAC for removing global motion prior to action recognition. In Hanheide et al. [13] a wearable camera system that recognizes actions is described. In that work tracked patches are used to estimate an affine homography representing camera motion, using the least median of squares (LMEDS) procedure. In Kong et al. [14] the Lucas–Kanade tracker [15] is used to track features in soccer videos. Camera motion is modeled with an affine homography, estimated with a RANSAC-like algorithm and then removed from the images, after which group actions are recognized using latent-dynamic conditional random fields. Of particular interest

* Corresponding author. Tel.: +34 926295300.
E-mail address: Oscar.Deniz@uclm.es (O. Deniz).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.10.019

is the global motion compensation described in Uemura et al. [16], in which SIFT features and the Lucas–Kanade tracker are used. Each input frame is segmented (using color MeanShift) and the homography estimated (using RANSAC) from the points falling inside each of the resulting segments. Starting from the dominant one (i.e., the segment with the largest number of inliers) segments are iteratively merged based on the number of inliers shared by the homographies. The best three segments are selected, the other remaining segments are simply merged with these three. This procedure partitions the image into (up to) three segments (i.e., dominant planes). Within each segment the motion of the tracked points is corrected using the homography associated to the segment. The method was shown to achieve a significant reduction of motion magnitude on background zones, which prompted us to select it as a strong baseline in this paper against which to compare our results.

Note that in our context of human action recognition we are not requiring 'video stabilization', a different but related problem. The objective of video stabilization is to remove undesirable camera motion effects so that only intentional motion effects are retained [17]. In our case, however, we are interested in removing intentional camera motion too. The primary benefit of video stabilization is to improve video quality by recording a stable sequence, whereas in our case we explicitly aim at removing camera motion from human motion. Although a global motion estimation step is common and some results can be used in both problems, note for example that the most popular motion estimation technique in video stabilization, i.e., the Block Matching Algorithm [18], does not take advantage of the salient point tracking commonly used in action recognition applications, while having itself a relatively high computational cost.[1]

The main contribution of this paper is a novel method for motion compensation within the context of the action recognition problem. Camera motion is modeled as a 2D translation, which allows us to use a fast image projection-based estimation procedure performed within a RANSAC framework. Then, an entropy-based method is proposed to determine which tracked features are moving in a manner consistent with the dominant motion. These ones are assumed to be background and their relative motion is zeroed out; the remaining ones are assumed to be foreground. Section 2 describes the underlying motivation and the method itself. Section 3 shows experiments and finally, in Section 4 the main conclusions are drawn.

2. Motivation and proposed method

The planar homography accounts for the perspective effects of camera translation and rotation, assuming a planar scene. There are cases, however, in which there is no camera rotation, such as in television broadcasts, or with shoulder-mounted cameras. As an example, consider the YouTube Action Dataset [19], a challenging set of YouTube videos used in the action recognition community. These videos contain large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background and illumination conditions. In this dataset, the variance of the homography matrix is (Fig. 1) which shows that the largest variations are in 2D translation.

When camera motion is modeled as 2D pixel translations, simpler and faster techniques can be used to tackle the motion compensation problem. The work of [20,21] on video stabilization, for example, uses

$$var\,(H) = \begin{bmatrix} 0.3358 & 0.0040 & 39.9279 \\ 0.0016 & 0.3352 & 20.4549 \\ 0.0000 & 0.0000 & 0.3381 \end{bmatrix}$$

Fig. 1. Variance of the homography matrix calculated between each pair of consecutive frames of the YouTube Action Dataset (1600 videos, 214 389 frames used). The homography matrix is estimated from Harris corners registered using correlation.

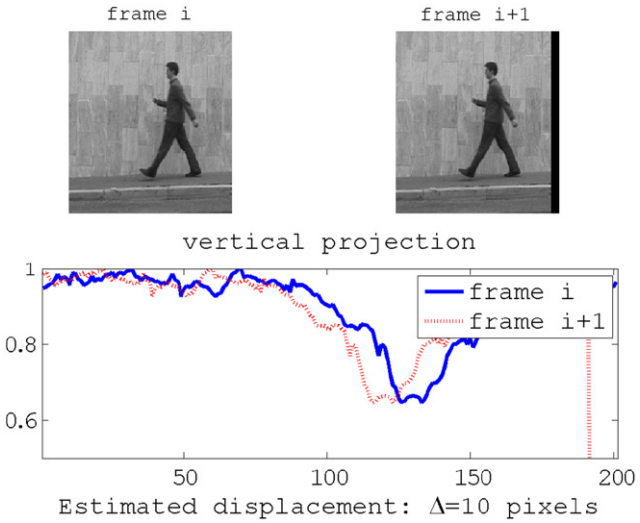

Fig. 2. Vertical projections can be used to get a global horizontal displacement estimate.

integral projections of the intensity image to infer a 2D translation. The vertical and horizontal projections between frames i and $i-1$ are registered (through cross-correlation or another registration technique), thus producing vertical and horizontal displacement estimations. This method proved very fast, in [20] it was used for real-time obstacle detection on high-speed trains. Image projections were also used for global motion estimation in [22,23].

In the method proposed here a number of FAST features [24] are extracted from the first frame and tracked with the Lucas–Kanade pyramidal algorithm. Then the vertical and horizontal image projections are computed, from which a global motion estimate can be obtained through cross-correlation, see illustrative Fig. 2. Global motion estimation from image projections, however can suffer from the effect of objects or individuals moving in the image. Besides, image projections depend on textures. Some zones in the image may be less textured than others, thus biasing the estimated global motion toward zero.

In order to make the displacement estimation more robust, a RANSAC-like approach is adopted. Projections are actually computed from vertical and horizontal bands of the image. Using the projections of each vertical and horizontal band a displacement can be computed (using cross-correlation between frames i and $i-1$), see illustrative Fig. 3. Among the computed displacements we assume that there are inliers (values in accordance with the global image displacement) and outliers. The outliers will be mostly caused by objects or individuals with motion different than the background motion. We therefore use a RANSAC algorithm to obtain a robust estimate of the horizontal and vertical global displacement. The procedure is shown in Fig. 4.

Segments are fixed-size continuous sections of the 1D signal, so the selection of segment is actually the random selection of a

[1] The Block Matching Algorithm for motion estimation divides the image into non-overlapping regions, typically squares. For each square, motion estimation is done by identifying another square (through correlation, for example) that best matches the first one. The displacement is provided as a pair (x,y) of horizontal and vertical displacement values. From the set of local displacements a global displacement is then estimated by some technique.

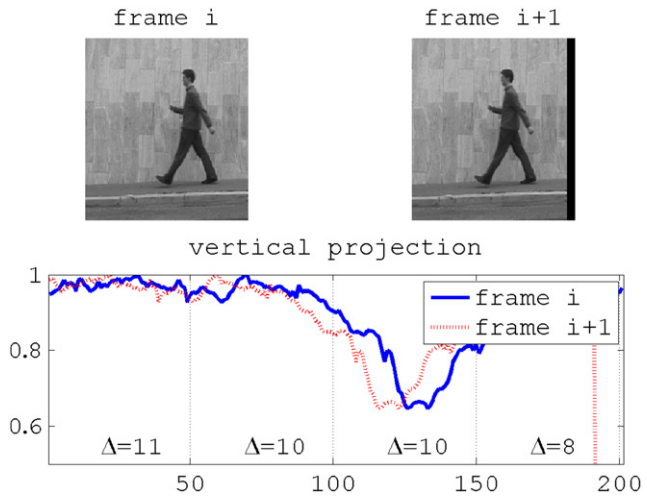

Fig. 3. One horizontal displacement estimate can be obtained from each vertical band of the image.

Input :
A = vertical (or horizontal) projection of frame t
B = vertical (or horizontal) projection of frame $t-1$

$MaxInliers = 0$
$l = 1$
while $l < MaxIterations$ do
  Choose a segment i of input signals, randomly
  $Disp$ = Register_Signals_By_Cross_Correlation (A_i, B_i)
  $NumInliers = 0$
  for $m = 1$ to L do
    Choose a segment j of input signals, randomly
    E_j = Align_Signals_And_Compute_SSD_Error $(A_j, B_j, Disp)$
    if $E_j < Threshold$ then
      $NumInliers = NumInliers + 1$
    end
  end
  if $NumInliers > MaxInliers$ then
    $MaxInliers = Inliers$
    $BestDisp = Disp$
  end
  if $\left(\frac{L-Inliers}{L}\right) <= LowerThreshold$ then
    break;
  end
  $l = l + 1$
end
return $BestDisp$

Fig. 4. Proposed algorithm for RANSAC of projections.

starting point. The threshold used to decide on inliers as well as *LowerThreshold* must be fixed beforehand (we used a value of 0.25 for the first one and *LowerThreshold*=0.38. These thresholds were used with all the datasets in the experiments).

Note that, as opposed to other global motion estimation work that uses RANSAC, here RANSAC is not applied to the motion of tracked features, but to a number of segments of the image projections. This is a potentially faster approach, since the number of tracked points is typically large, whereas with this approach the worst case depends on the chosen L. Moreover, note that each iteration of the loop can be run in parallel. The length m of the segments should be fixed to a submultiple of the image width or height.

Note that the performance of this method will depend on the number of outliers, which in turn depends on the size of the individuals (or objects with independent motion) in the image. If individuals appear too large the estimation will fail. To avoid this, we impose an additional threshold:

$$\frac{L-MaxInliers}{L} \geq UpperThreshold \quad (1)$$

if the threshold is exceeded then a non-RANSAC projection-based estimation is used instead.

Let us call $(disp_x,disp_y)$ the estimated global motion. Motion compensation is carried out by subtracting, in each frame, the vector $(disp_x,disp_y)$ from each tracked point. The method described up to this point is later called 'Proposed Method 1' in the experiments.

A second method is also proposed that uses as input global motion estimates obtained with Method 1. This second method keeps a history of 'motion inconsistency' for each tracked point. Motion inconsistency basically measures the departure of each tracked point's motion from the global motion. Then an entropy-based threshold is used on this motion inconsistency to separate tracked points between background and foreground points. The following paragraphs describe this second method in detail.

A 'motion inconsistency' history (MIH) is maintained for each tracked point and for the last K frames. The motion inconsistency of a tracked point i at time t, P_i^t, is the difference between the estimated current global displacement $D^t=(disp_x,disp_y)$ and the actual displacement of the point between frames $t-1$ and t. The motion inconsistency history of a tracked point is thus defined as in Eq. (2). Points with larger accumulated motion inconsistency are those corresponding to individuals moving with independent motion, whereas those with smaller inconsistency correspond mostly to background points (that are subject to camera motion only). Thus, the MIH value for a given tracked point integrates the degree to which the motion of that point is consistent with the global observed motion of the background:

$$\mathtt{MIH}(P_i^t) = \sum_{k=0}^{K} \|\|P_i^{t-k} - P_i^{t-k-1}\| - D^{t-k}\|, \quad i = 1,\ldots,n. \quad (2)$$

From the accumulated MIH we now attempt to separate background and foreground points. That is, we would like to have a threshold of accumulated MIH values that separate background points and foreground points. Such separating thresholds cannot be obtained from MIH values alone, for typically they form a continuum (i.e., without definite clusters). Thus, we must consider the spatial position of the points. Foreground points usually belong to individuals in the image with independent motion. These points will tend to be spatially clustered in the image. We therefore consider the entropy of the positions of the points. Entropy is a dispersion measure similar to variance. As opposed to variance, which measures dispersion only around the sample mean, entropy measures dispersion between samples. Here it is used to detect 'lumps' of tracked points. Actually, we compute the entropy only in the x dimension.[2]

Entropy estimation is not straightforward. The simplest approach of histogram-based density estimation requires careful tuning of parameters such as bin sizes. We estimate entropy with a method based on sample spacings [25]. Let us assume that $x_1,\ldots,x_n$

[2] Note that in order to calculate entropy the set of x coordinates of the tracked points must be first normalized to zero mean and unit variance (this has the effect of compensating for the variance, which measures sample dispersion with respect to the mean). Had we opted for computing the (x,y) (2D) entropy, a 'whitening' normalization would then be needed, which could be achieved using PCA. Here, the x dimension suffices because we assume that individuals appear on horizontal surfaces.

is a sample of i.i.d. real valued random variables. Let $x_1 \le x_2 \le \cdots \le x_n$ be the corresponding order statistics. Then $x_{i+m} - x_i$ is called a spacing of order m, or m-spacing ($1 = i < i+m = n$). The m-spacing estimate of entropy, for fixed m, is defined as [25]

$$H = \frac{1}{n}\sum_{i=1}^{n-m} \ln\left(\frac{n}{m}(x_{i+m} - x_i)\right) - \psi(m) + \ln m \quad (3)$$

where ψ is the digamma function. We used $m=1$ in our experiments. In practice we computed Eq. (3) by first ordering the x coordinates of the points in ascending order and then computing the mean of (the logarithm of) differences between consecutive elements.

In each frame, we ordered the tracked points by increasing MIH value and, for each point, computed the entropy of the set of tracked points with equal or higher MIH value. That is, for n points we have a set of indices:

$$i = 1, \ldots, n \quad s.t \ \forall i \ge 2 \ \ \mathtt{MIH}(P_i^t) \ge \mathtt{MIH}(P_{i-1}^t) \quad (4)$$

and

$$E(i) = Entropy(\{P_j^t\}), \quad j = i, .., n \quad (5)$$

If we represent these two variables in a figure we obtain a pattern like that shown in Fig. 5.

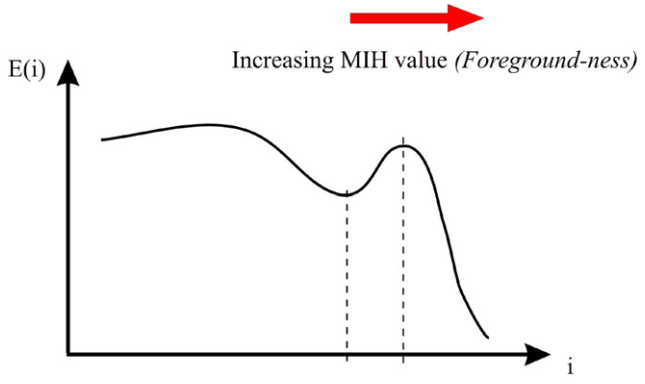

Fig. 5. Entropy of tracked points with equal or higher MIH value. A valley-pattern appears on the left, see the text.

Since points with a low MIH correspond to the background, points on the right side of the figure are mainly from the foreground, typically associated with one or more individuals with

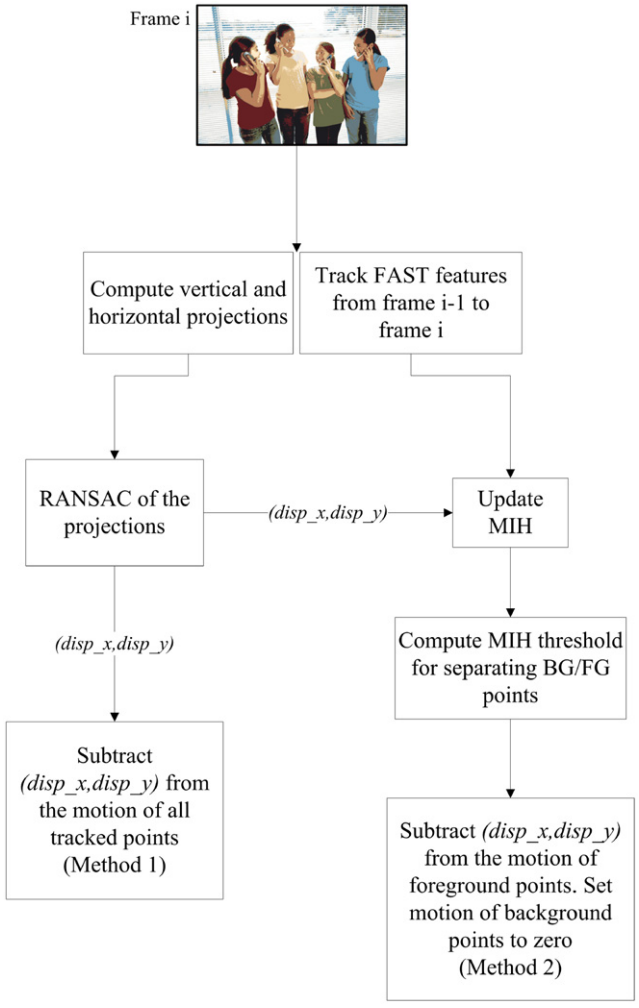

Fig. 7. Steps in the two proposed algorithms.

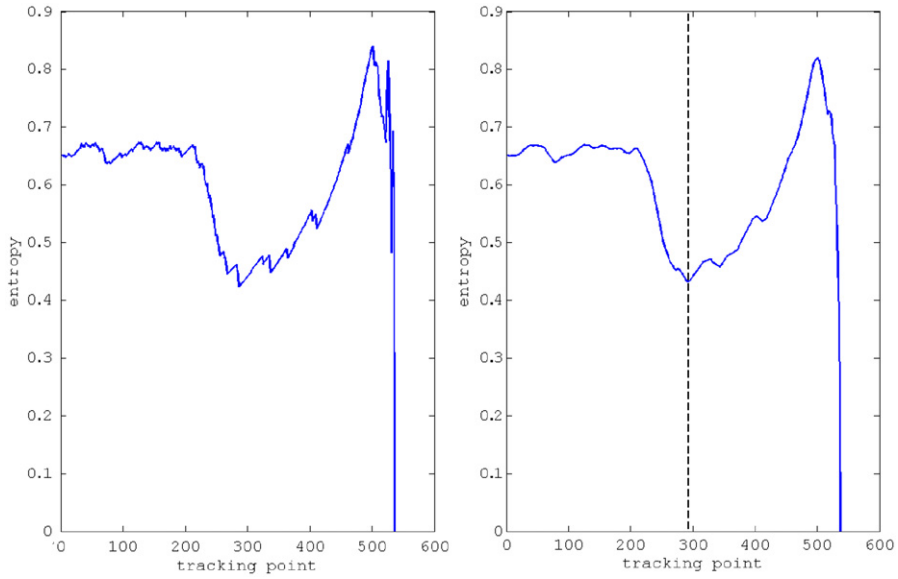

Fig. 6. Actual entropy values for frame 40 of one of the experiment sequences (see below). Left: raw signal, Right: smoothed signal with the obtained threshold.

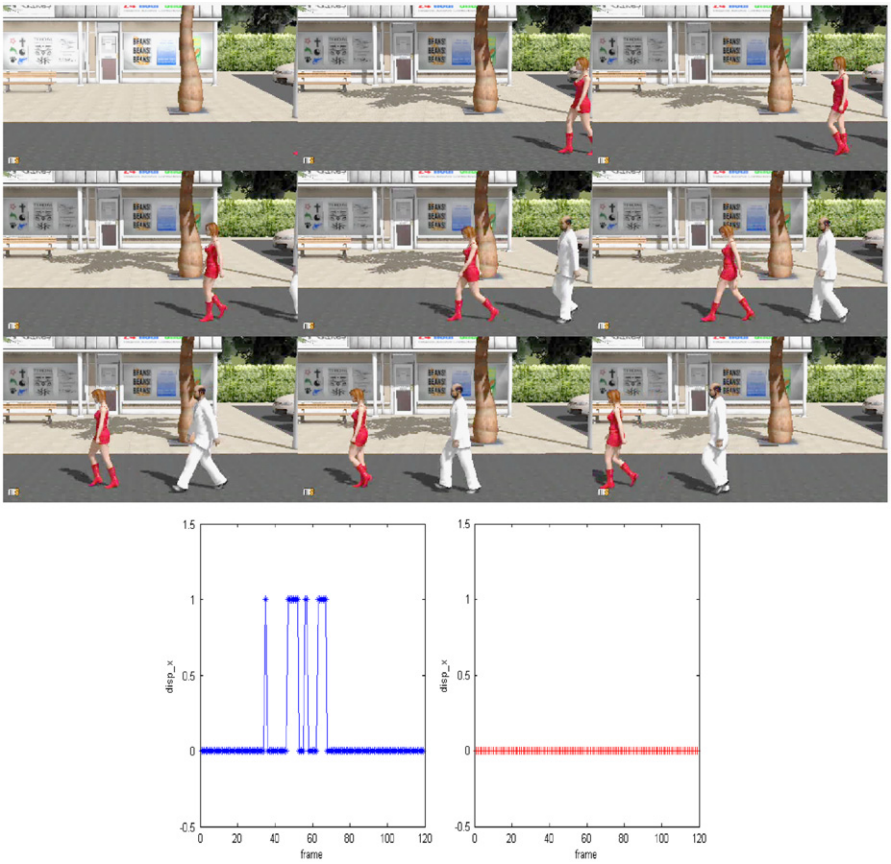

Fig. 8. Effect of RANSAC of projections in the estimation of horizontal global motion. The camera is static throughout the whole sequence, only the individuals are moving from right to left. Bottom figures, Left: estimation using only cross-correlation between the projections of consecutive frames, Right: RANSAC of projections. (Synthetic video created with the Moviestorm ^{TM} real-time 3D movie creation software.)

independent motion. It can be observed in the figure that there is a valley. Going from right to left, on the right half of the valley more and more foreground points are considered (which decreases the average dispersion of the set, thus decreasing the entropy). Then, the situation reverses, as more and more background points are considered the relative separation between points increases, thus increasing the entropy. The position of the rightmost valley can be therefore used to get a MIH threshold, with which we can separate background and foreground points. In practice, a good value for the threshold must be set around the bottom of the valley. If we set the threshold too low background points will be considered as foreground. If we set it too high foreground points will be considered as background.[3] The entropy signal is first smoothed with a running average of width = 15 samples, see Fig. 6.

Once we can separate background from foreground points we correct their motion as follows. For background points we directly set their motion to zero (i.e., since we now know that they are background points they must not suffer displacement). For foreground points we subtract the estimated global displacement ($disp_x$,$disp_y$) obtained with the Method 1 above.

The method described up to this point is later called 'Proposed Method 2' in the experiments. Although in Method 2 we have used the so-called Method 1 for estimating global motion, note that it is possible to use any other global motion estimation methods, including those based on affine and full projective homographies. Fig. 7 summarizes the main steps of the two proposed methods.

3. Experiments

In order to illustrate the steps of the methods proposed we first show an example of the effect of the RANSAC of projections. In Fig. 8 we show a synthetic video in which two individuals enter the scene from the right. The estimated horizontal displacement without using RANSAC (i.e., using only the cross-correlation between the projections of consecutive frames) is negatively affected by the presence of the individuals. The RANSAC-version gives a more robust estimate.

In Fig. 9 we show the tracked points and associated entropies, for frame 50 of the same video sequence.

In order to assess the performance of the proposed methods, experiments were also carried out with publicly available video datasets, with both synthetic and real camera motion. Table 1 summarizes the video datasets used. A total of 167 videos were used in the experiments.

The KTH action recognition dataset [28], widely used in the action recognition community, was not used here since the backgrounds are mostly static and uniform.

We compared the two proposed methods with Uemura et al.'s homography-estimation based method [16], discussed in Section 1. In that work, performance is measured as the average magnitude of motion for foreground (i.e., individuals) and background regions before and after motion compensation, assuming that a good performance maximizes the first quantity and minimizes the

[3] Note that the threshold is actually one of the tracked points (once they have been ordered by increasing MIH). In our implementation we use a peak finding algorithm for locating both the valley and the rightmost peak of Fig. 5. Let those two points be i_a and i_b. The index of the cut-off tracked point, i_c, is always between those two points: at configuration time we adjust an $\alpha \in [0,..,1]$ and then we make $i_c = round(\alpha \cdot i_a + (1-\alpha) \cdot i_b)$.

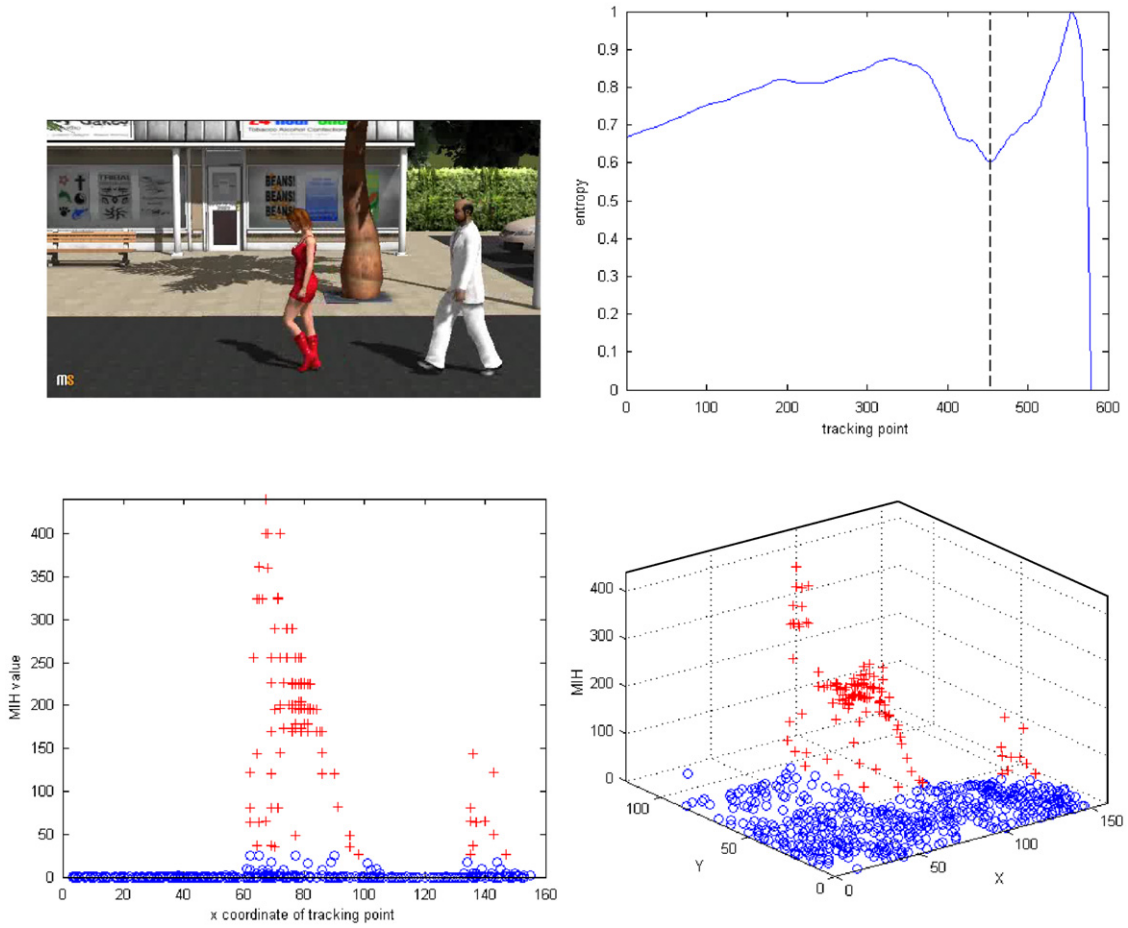

Fig. 9. Top left: frame 50 of the video, Top right: entropy as a function of the tracked point considered, showing the point of the estimated MIH threshold. Bottom left: *x*-coordinate of each tracked point *vs.* its MIH value, Bottom right: 3D representation of point coordinates *vs.* MIH value. Points estimated as background are shown as circles. Points estimated as foreground are shown as crosses.

Table 1
Video datasets used in the experiments.

Dataset	Description	Camera motion	Ref.
Weizmann	'Robust' set: people walking in various difficult scenarios in front of different non-uniform backgrounds. 'Classification' set: people performing 10 natural actions such as run, walk, skip, bend, etc., with a more uniform background	Synthetic	[26]
MultiKTH	Four sequences which show one indoor and three outdoor scenes with moving foreground people, complex background, multiple dominant planes and camera motion (a fifth sequence from the original set was not used because it did not show people)	Yes	[16]
UCF sports action dataset	Actions collected from various sports which are typically featured on broadcast television channels	Yes	[27]
YouTube action dataset	YouTube videos with large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background and illumination conditions	Yes	[19]

second. We argue that the first quantity is inappropriate, since the magnitude of motion in foreground regions can be large after correction even when we apply a bad correction. Therefore, for synthetic camera motion experiments we define a new measure: *error per tracked point* (EPTP). For each frame t, the EPTP is calculated as follows:

$$EPTP = \frac{E_b + E_f}{n_b + n_f} \tag{6}$$

with n_b and n_f being the number of background and foreground tracked points, respectively. E_b is the compensation error for the background points $\mathbf{p_i}$:

$$E_b = \sum_{i=1}^{n_b} \|\Delta\mathbf{p_i} - \Delta_{\mathbf{synthetic}}\| \tag{7}$$

where $\Delta\mathbf{p_i}$ refers to the estimated camera-motion-related 2D displacement for point $\mathbf{p_i}$ between frames t and $t-1$. $\Delta_{\mathbf{synthetic}}$ is the synthetic 2D displacement that was applied between those two frames. Ideally, the motion compensation algorithm would estimate the synthetic displacement correctly, thus making $E_b = 0$. E_f is defined analogously for foreground points.

For real camera motion experiments, the actual camera motion in each frame is not available. Thus, for that case we define $E_f = 0$

and E_b as the residual motion after compensation:

$$E_b = \sum_{i=1}^{n_b} \|(\mathbf{p_i}(\mathbf{t}) - \mathbf{p_i}(\mathbf{t-1})) - \Delta\mathbf{p_i}\| \quad (8)$$

Again, the ideal motion compensation algorithm would estimate the camera motion correctly, which would make $E_b = 0$.

The division into background and foreground points is obtained by considering the ROI's of the individuals present in the image (the ground-truth of the video datasets).

The proposed methods were also compared with a simple RANSAC-based affine motion estimator and with the method described in [29], in which the authors also used the Weizmann dataset. The latter method is based on computing optical flow vectors from consecutive frames and then applying a local median filter. The resulting motion vectors are then used for compensation. In our implementation we used a size of 25 for the median filter neighborhood, the same as used in [29].

3.1. Synthetic camera motion

Camera motion was simulated by the following procedure. First, the frame is zoomed in 20%, then a random 2D displacement is applied, and the resulting image is cropped to the original size. This avoids black borders after the displacement step. The algorithm

Fig. 10. Left: sample frame from a Weizmann 'robust' set video. Right: sample frame from a Weizmann 'classification' set video.

Table 2
Results for the Weizmann 'robust' set.

Method	EPTP	Average FG point error ($=E_f/n_f$)	Average BG point error ($=E_b/n_b$)	Time (ms)
No compensation	3.368	7.17	1.83	1.2
Proj+XCorr	2.612	7.20	0.73	**2.0**
Simple RANSAC	2.416	7.13	0.46	75.3
Uemura et al.'s	2.431	7.21	0.34	5814.9
Oikonomopoulos et al.'s	3.187	7.54	1.46	19.2
Proposed Method 1	2.339	**7.13**	0.35	3.6
Proposed Method 2	**2.218**	7.63	**0.03**	91.1

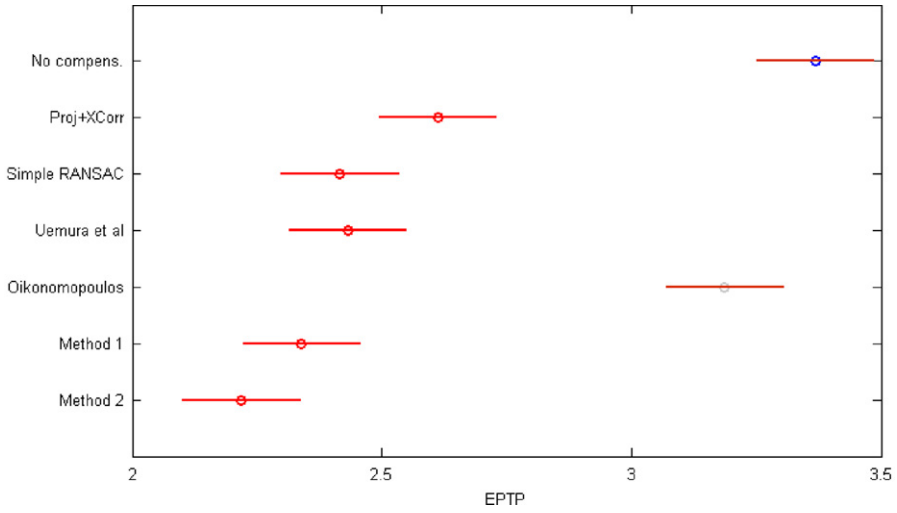

Fig. 11. Results of the one-way ANOVA followed by a multiple comparison (using the Tukey–Kramer method), performed with the seven methods considered, on the Weizmann robust set.

Table 3
Results for the Weizmann 'classification' set.

Method	EPTP	Average FG point error ($=E_f/n_f$)	Average BG point error ($=E_b/n_b$)	Time (ms)
No compensation	2.976	7.18	2.16	0.5
Proj+XCorr	2.267	7.26	1.29	**1.3**
Simple RANSAC	2.002	7.13	1.00	40.8
Uemura et al.'s	2.033	7.23	1.01	3317.8
Oikonomopoulos et al.'s	2.789	7.37	1.91	18.7
Proposed Method 1	1.982	**7.13**	0.98	2.8
Proposed Method 2	**1.334**	7.44	**0.16**	21.0

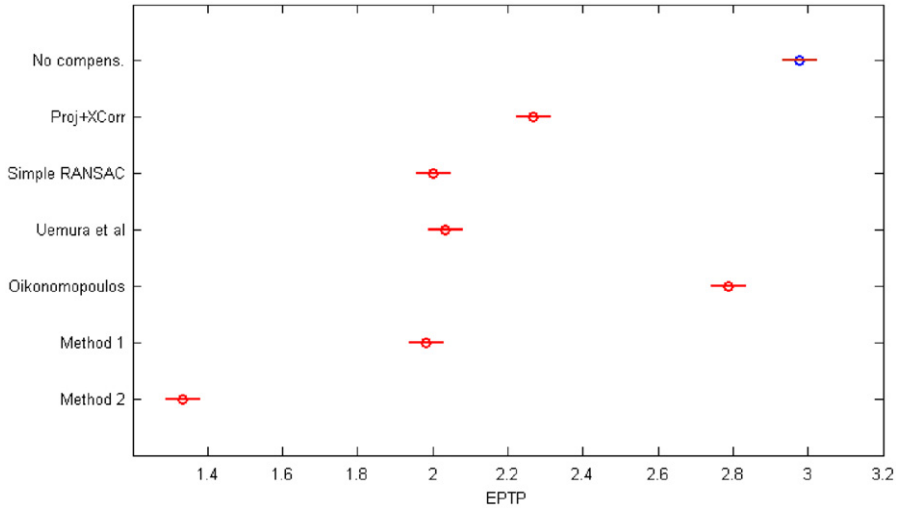

Fig. 12. Results of the one-way ANOVA followed by a multiple comparison, performed with the five methods considered, on the Weizmann classification set. The EPTP results of Method 2 for this set are statistically different from those of the other six methods.

chooses a random displacement point and successively moves the image toward that point. Then it chooses another point and the process is repeated. The algorithm has two parameters: speed s (defined as the number of steps between any two random points) and maximum coordinate value d of chosen points from the (0,0) position. The experiments shown here were carried out using $s=7$ and $d=\pm 12$.

Uemura et al. [16] used Meanshift to segment the images. We used the same segmentation parameter values as those used in that work for the MultiKTH videos (which they also used in their experiments). In order to make the comparison fair, for other datasets we manually adjusted the parameters to get a better segmentation.

The Weizmann dataset is actually divided into two sets of videos. The 'robust' set (10 videos) shows people walking in various difficult scenarios in front of different non-uniform backgrounds, see Fig. 10-left. The 'classification' set shows people performing 10 natural actions such as run, walk, skip, bend, etc., with a more uniform background, see Fig. 10-right. Table 2 shows the obtained errors for the Weizmann robust dataset. 'Proj+XCorr' refers to the use of image projections and cross-correlation to get estimates of global motion, i.e., the method of [20]. Fig. 11 shows the result of an ANOVA test performed with the five methods considered, considering all the videos of the set. It can be shown that Method 1 is not significantly different from Uemura et al.'s, although Method 2 achieves a significant reduction in EPTP, mainly due to a large reduction in background error.

First, note that the results for Oikonomopoulos et al.'s method are relatively poor. We hypothesize that this is due to border effects in the local median filtering (even though we replicated border pixels in our implementation).

Table 4
Results for the MultiKTH dataset.

Method	EPTP	Time (ms)
No compensation	0.018	0.3
Proj+XCorr	0.017	**0.9**
Simple RANSAC	0.023	35.0
Uemura et al.'s	0.026	1157.5
Oikonomopoulos et al.'s	0.016	11.9
Proposed Method 1	0.016	2.9
Proposed Method 2	**0.007**	15.4

The average processing times (per frame) are significantly higher for the method of Uemura et al. mainly because of the MeanShift segmentation (which for this experiment took 98.7% of its processing time). Note that the Method 2 proposed here also performs a segmentation of background and foreground points, although it is much faster (it is a segmentation of points, not of the image itself, and the segmentation itself is accomplished by a relatively simple entropy-based threshold). The code developed for the experiments was not explicitly optimized.[4] All the experiments were completed on a Intel® Core™ 2 CPU at 3 GHz.

[4] Intel's Integrated Performance Primitives™ were used. Input frames were first converted to 160×120.

Table 3 and Fig. 12 show the results for the Weizmann classification set.

3.2. Real camera motion

This subsection describes experiments with real camera motion videos taken of three different public datasets. The MultiKTH dataset, which was also used in [16], contains four sequences which show one indoor and three outdoor scenes with moving foreground people, complex background, multiple dominant planes and camera motion (a fifth sequence from the original dataset was not used because it did not show people). Since in our experimental implementation FAST points are extracted only once at the beginning, we split the videos into a number of equal parts and run the implementation for each part independently. Table 4 and Fig. 13 show the results.

Experiments were also carried out using a subset of the UCF Sports Action dataset [27], which consists of a set of actions collected from various sports which are typically featured on broadcast television channels. Only videos with camera motion were selected (one video from each section of sets: Golf-Swing-Front, Kicking-Front, Kicking-Side, Riding-Horse, Run-Side, SkateBoarding-Front, Swing-Bench, Swing-SideAngle and Walk-Front, see Fig. 14). As explained above, in this case the EPTP is equal to the average magnitude of motion for background points (after global motion subtraction). Due to extreme differences between videos, the results are given here independently for each video, see EPTP's in Table 5. Fig. 15 shows the corresponding ANOVA tests.

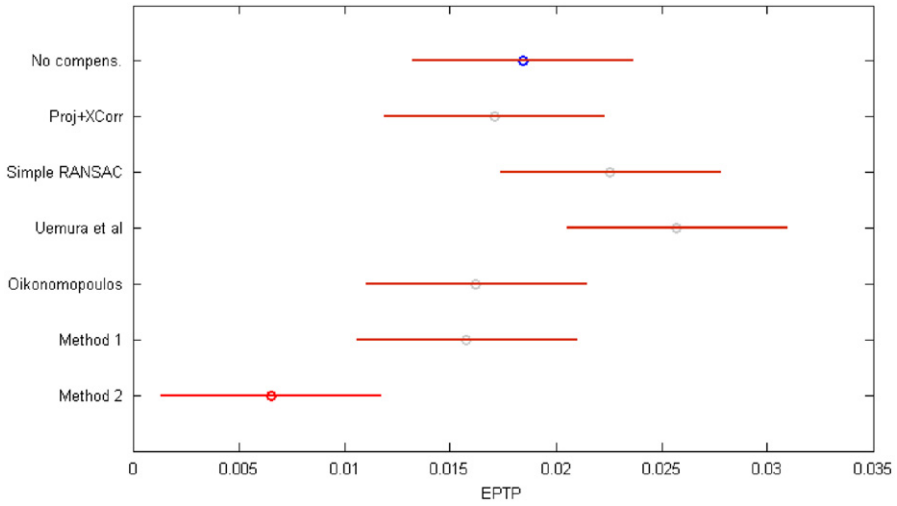

Fig. 13. Results of the one-way ANOVA followed by a multiple comparison, performed with the seven methods considered, on the MultiKTH dataset.

Fig. 14. Sample frames for UCF videos. Top to bottom, left to right: Golf-Swing-Front, Kicking-Front, Kicking-Side, Riding-Horse, Run-Side, SkateBoarding-Front, Swing-Bench, Swing-SideAngle, Walk-Front.

The results on this dataset are in accordance with the results obtained with synthetic camera motion both in accuracy and speed. Method 2 is the best in terms of EPTP. On the other hand, as opposed to the results with synthetic motion, here Uemura et al.'s method could not improve over 'No compensation'. We hypothesize that this is due to the fact that most of the videos have a relatively uniform green background (the sports field). This affects negatively to the method since it is affected by the larger tracking errors. Video Swing-SideAngle, for example, does not have such uniform background and there Uemura et al.'s performed better.

Uemura et al.'s method bad performance can be also attributed to the fact that it is strongly dependent on the segmentation parameters. In the experiments, we used the same parameters for all the videos in the dataset. Note that in this dataset the videos are very different from each other, which was not the case with the Weizmann dataset.

Note also that, as opposed to the case in previous databases, here Method 1 does not improve much over 'Proj+XCorr'. This is due to the fact that individuals in the images have a much larger size than in previous databases, which translates into more outliers.

Finally, experiments were also carried out using videos from the already mentioned YouTube Action Dataset [19], a challenging set of YouTube videos used in the action recognition community. These videos contain large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background and illumination conditions. Thirty-one randomly selected videos were used, including actions like walking, tennis swing, volleyball spiking, basketball shooting, horse riding, trampoline jumping, soccer juggling, diving and biking. Table 6 and Fig. 16 show the results on this dataset.

Figs. 17–22 show how the Proposed Method 2 can remove real camera motion.

Since the evolution of modern hardware places more and more importance on exploiting parallelism, here we make explicit the possibilities of the algorithms introduced. The steps than can be parallelized are: (a) the Lucas–Kanade tracker, for which parallel implementations already exist, (b) the computation of image projections (slices of the image can be allocated to different processors), (c) the RANSAC of image projections and (d) the computation of MIH values of tracked points, which can be shared among different processors.

4. Conclusions

The human action recognition problem has attracted much interest in the last years. The implicit objective is to classify human actions by analyzing the motion of body parts. Due to camera motion, however, the trajectories of these parts contain not only the motion of the individual but also the motion of the camera. Thus, camera motion must be first compensated for. In this context, this paper makes two contributions. First, it introduces RANSAC in the estimation of global motion using image projections. The method proposed is shown to improve the estimation when the area of the image occupied by the individuals is not too large. Second, a method is proposed for segmenting salient points into background and foreground points. The method is based on thresholding the entropy of a motion inconsistency measure. Experiments show that this second method is able to remove background (camera) motion almost completely, while still retaining the useful foreground motion. Besides,

Table 5
Results for the UCF set.

Method	EPTP	Time (ms)
No compensation	0.130	0.6
Proj+XCorr	0.123	**1.2**
Simple RANSAC	0.167	28.5
Uemura et al.'s	0.436	1954.9
Oikonomopoulos et al.'s	0.127	17.9
Proposed Method 1	0.121	6.6
Proposed Method 2	**0.057**	21.3

Table 6
Results for the YouTube set.

Method	EPTP	Time (ms)
No compensation	0.055	0.4
Proj+XCorr	0.046	**0.8**
Simple RANSAC	0.049	22.9
Uemura et al.'s	0.059	1187.7
Oikonomopoulos et al.'s	0.054	13.8
Proposed Method 1	0.045	3.7
Proposed Method 2	**0.027**	15.9

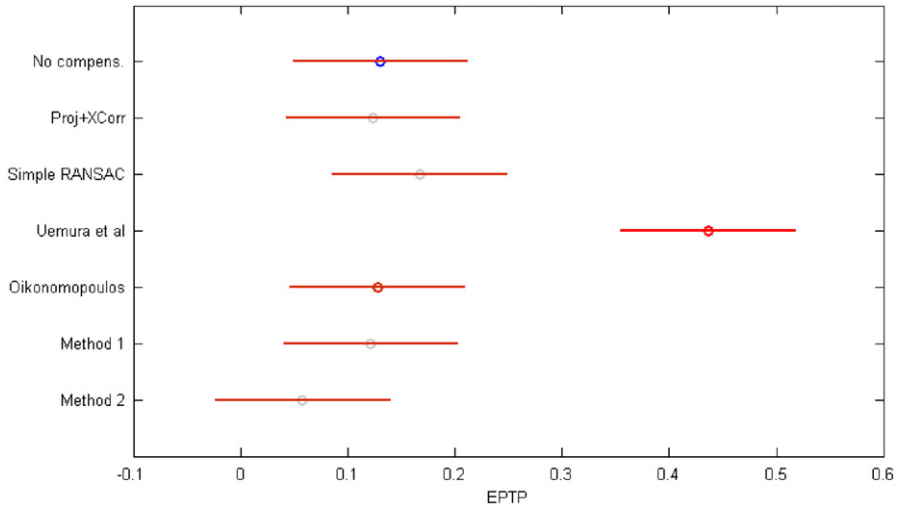

Fig. 15. Results of the one-way ANOVA followed by a multiple comparison, performed with the seven methods considered, on the UCF set.

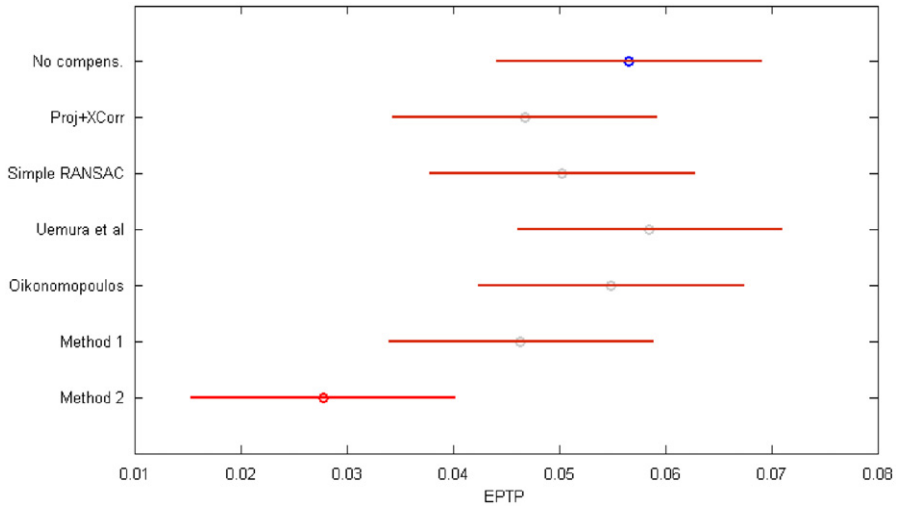

Fig. 16. Results of the one-way ANOVA followed by a multiple comparison, performed with the seven methods considered, on the YouTube set.

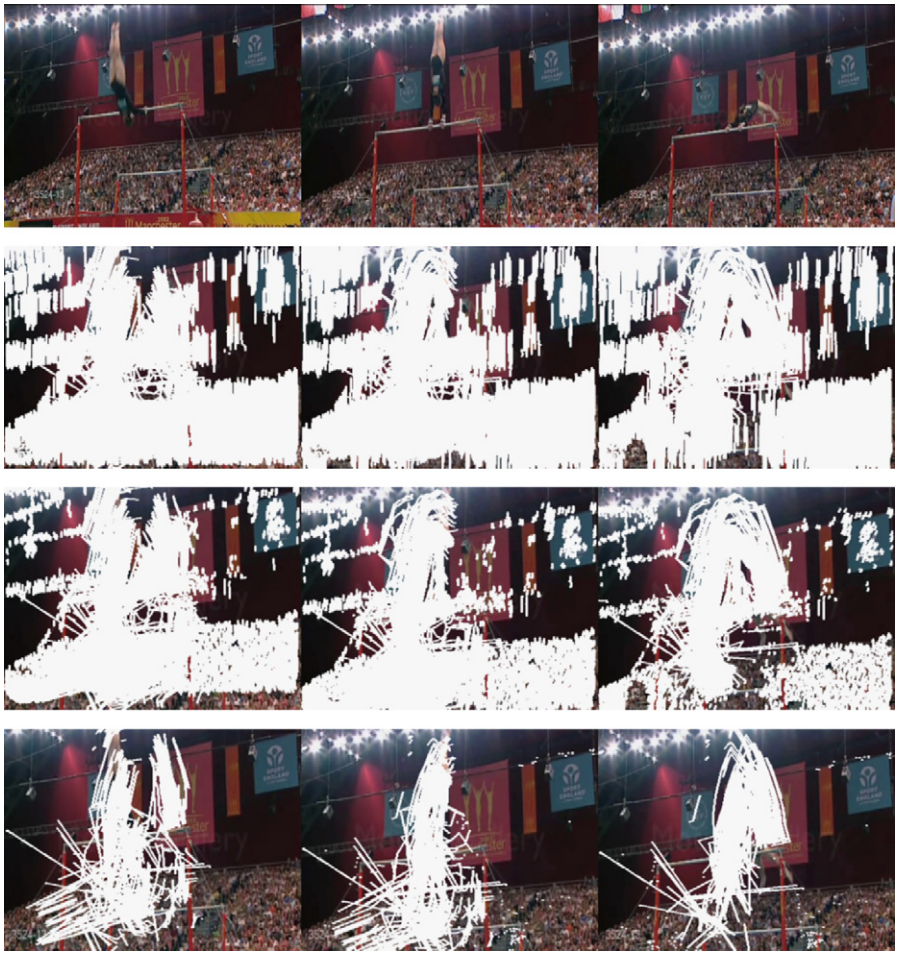

Fig. 17. Salient point trajectories for three (non-consecutive) frames of video Swing-SideAngle from UCF dataset (tracks are shown as white lines connecting the last $K=30$ point positions after global motion correction). Top-row: original frames, second row: no camera motion compensation, note the significant background motion when the camera follows the swings of the individual on the high bar, third row: result using Method 1, bottom row: result using Method 2. Note that there are some strokes that represent tracking errors.

the proposed method's processing time compares well with other approaches and is low enough to be useful for real-time applications.

Future work shall explore the option of using the FG/BG point segmentation as a feedback for improving the global motion estimation step. Zones of the image previously estimated as

Fig. 18. Salient point trajectories for three (non-consecutive) frames of video Skateboarding-Front from UCF dataset.

Fig. 19. Salient point trajectories for three (non-consecutive) frames of video Soccer-Juggle from YouTube dataset.

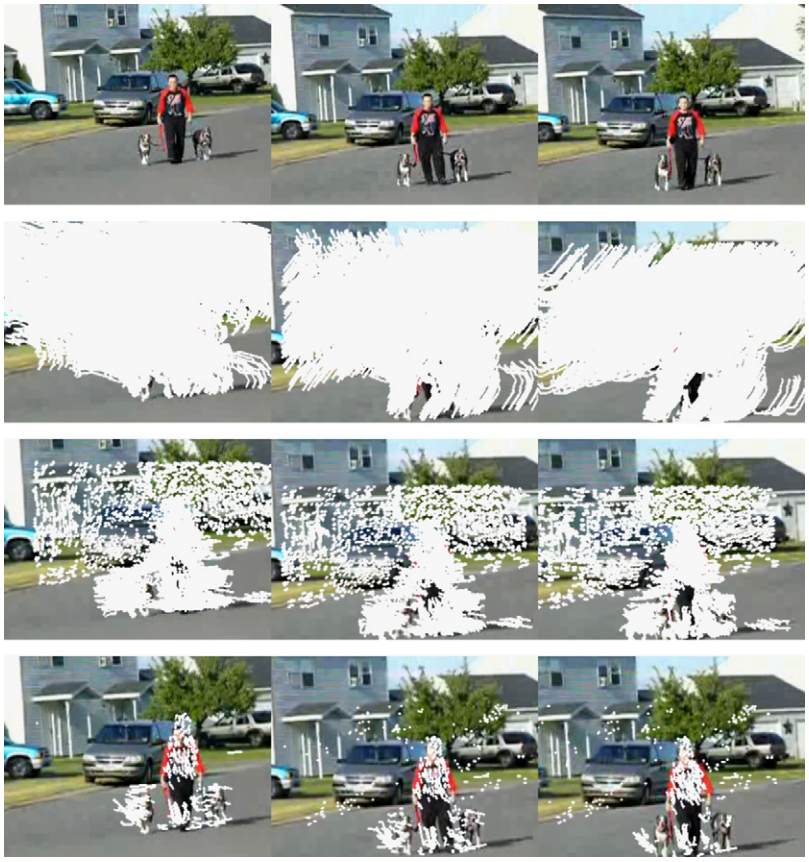

Fig. 20. Salient point trajectories for three (non-consecutive) frames of video Dog-walk from YouTube dataset. In this sequence the individual is walking towards the cameraman. This sequence had a shaky motion because of a handheld camera.

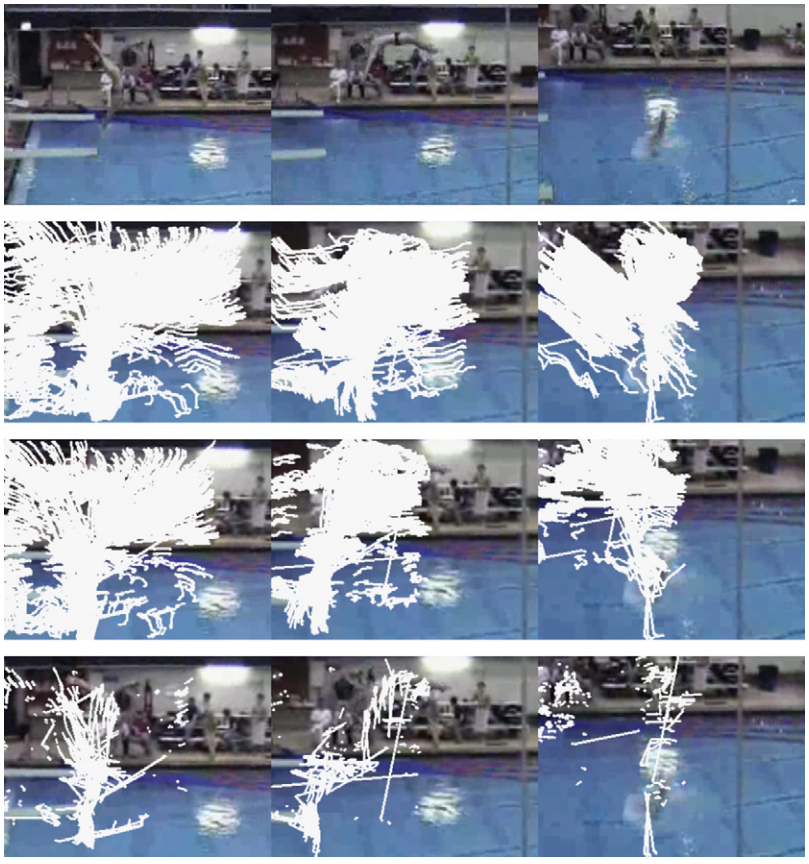

Fig. 21. Salient point trajectories for three (non-consecutive) frames of video Diving from YouTube dataset.

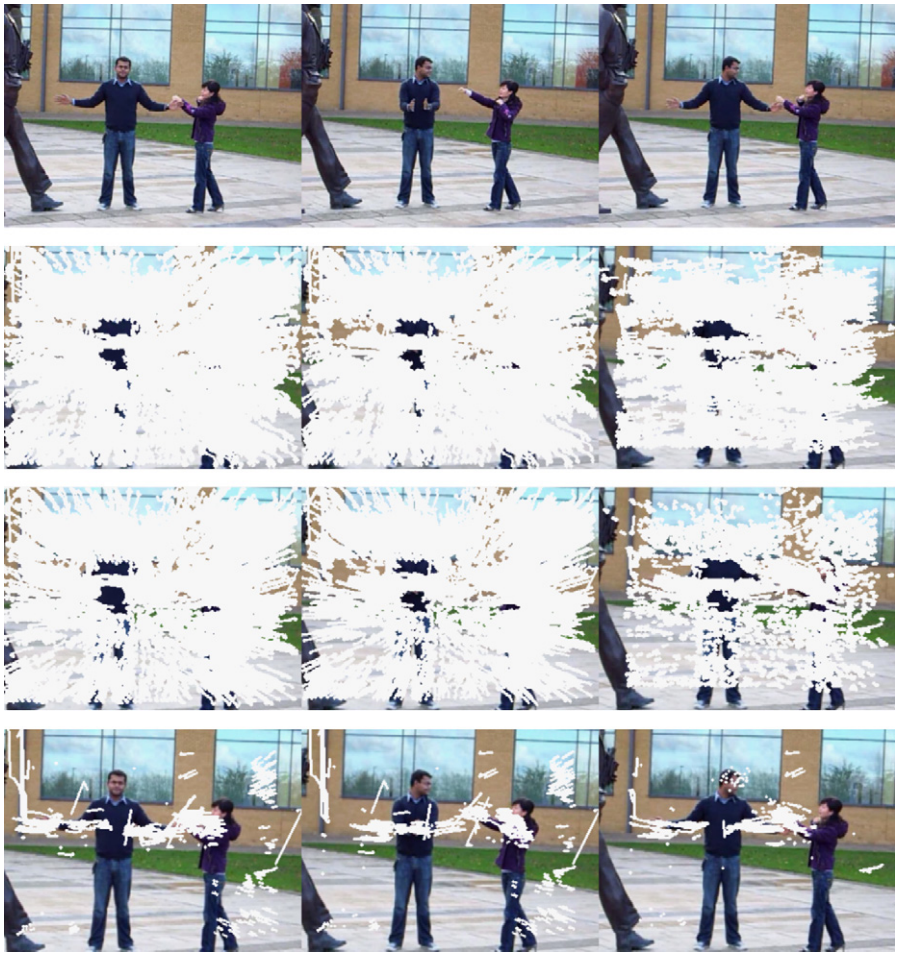

Fig. 22. Salient point trajectories for three (non-consecutive) frames of video MultiKTH from MultiKTH dataset. In the sequence the camera is zooming in and out. The man is applauding and turning his head, while the woman is moving her arms.

foreground would be given less importance in the global motion estimation step. The subsequent more robust global motion estimations would produce a better MIH, and this in turn a better FG/BG point segmentation.

Acknowledgments

The authors want to thank Dr. K. Mikolajczyk for providing the MultiKTH dataset used in the experiments. This work was partially funded by project PII2I09-0043-3364 of Castilla-La Mancha Regional Government and from the Spanish Research Ministry through project RETIC COMBIOMED.

References

[1] A. Yilmaz, M. Shah, Matching actions in presence of camera motion, Comput. Vision Image Understanding 103 (2–3) (2006) 221–231.
[2] R.F.C. Guerreiro, P.M.Q. Aguiar, Global motion estimation: feature-based, featureless, or both? in: ICIAR06, Povoa de Varzim, Portugal, 2006, pp. 721–730.
[3] A.J. Crawford, H. Denman, F. Kelly, E. Pitie, A.C. Kokaram, Gradient based dominant motion estimation with integral projections for real time video stabilisation, in: ICIP04, Singapore, 2004, pp. V: 3371–3374.
[4] Y. Altunbasak, R.M. Mersereau, A.J. Patti, A fast parametric motion estimation algorithm with illumination and lens distortion correction, Image Process. 12 (4) (2003) 395–408.
[5] B.E. Pires, P.M.Q. Aguiar, Featureless global alignment of multiple images, in: ICIP05, Genoa, Italy, 2005, pp. I: 57–60.
[6] P. Perez, N. Garcia, Robust and accurate registration of images with unknown relative orientation and exposure, in: ICIP05, Genoa, Italy, 2005, pp. III: 1104–1107.
[7] R.I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2000 ISBN:0521623049.
[8] S. Battiato, G. Gallo, G. Puglisi, S. Scellato, Fuzzy-based motion estimation for video stabilization using SIFT interest points, in: Proceedings of SPIE Electronic Imaging 2009 System Analysis for Digital Photography V, 2009.
[9] M. Fischler, R. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.
[10] G.Y. Zhu, C.S. Xu, W. Gao, Q.M. Huang, Action recognition in broadcast tennis video using optical flow and support vector machine, in: CVHCI06, Graz, Austria, 2006, pp. 89–98.
[11] C. Panagiotakis, E. Ramasso, G. Tziritas, M. Rombaut, D. Pellerin, Shape-motion based athlete tracking for multilevel action recognition, in: AMDO06, Mallorca, Spain, 2006, pp. 385–394.
[12] K. Mikolajczyk, H. Uemura, Action recognition with motion-appearance vocabulary forest, in: CVPR08, Anchorage, USA, 2008, pp. 1–8.
[13] M. Hanheide, N. Hofemann, G. Sagerer, Action recognition in eWearable assistance system, in: ICPR (2), 2006, pp. 1254–1258.
[14] Y. Kong, X.Q. Zhang, Q.D. Wei, W.M. Hu, Y.D. Jia, Group action recognition in soccer videos, in: ICPR08, 2008, pp. 1–4.
[15] J. Shi, C. Tomasi, Good features to track, in: CVPR94, 1994.
[16] H. Uemura, S. Ishikawa, K. Mikolajczyk, Feature tracking and motion compensation for action recognition, in: BMVC08, Leeds, UK, 2008, pp. 293–302.
[17] S. Battiato, R. Lukac, Video stabilization techniques, in: Encyclopedia of Multimedia, 2008, pp. 941–945.
[18] A.R. Bruna, A. Capra, S. Battiato, G. Puglisi, Digital video stabilisation in modern and next generation imaging systems, in: Proceedings of NEM Summit, Saint-Malo, France, 2008.

[19] J.G. Liu, J.B. Luo, M. Shah, Recognizing realistic actions from videos 'in the wild', in: CVPR09, Miami, USA, 2009, pp. 1996–2003.
[20] S. Piva, M. Zara, G. Gera, C.S. Regazzoni, Color-based video stabilization for real-time on-board object detection on high-speed trains, in: AVSS '03: Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, Washington, DC, USA, 2003.
[21] A. Bosco, A. Bruna, S. Battiato, G. Bella, G. Puglisi, Digital video stabilization through curve warping techniques, IEEE Transactions on Consumer Electronics 54 (2) (2008) 220–224.
[22] J.H. Lee, J.B. Ra, Block motion estimation based on selective integral projections, in: ICIP02, New York, USA, 2002, pp. I: 689–692.
[23] K. Sauer, B. Schwartz, Efficient block motion estimation using integral projections, CirSysVideo 6 (5) (1996) 513–518.
[24] E. Rosten, T. Drummond, Machine learning for high-speed corner detection, in: European Conference on Computer Vision, vol. 1, Graz, Austria, May 2006, pp. 430–443.
[25] J. Beirlant, E.J. Dudewicz, L. Gyrfi, E.C. Meulen, Nonparametric entropy estimation: an overview, International Journal of the Mathematical Statistics Sciences 6 (1997) 17–39.
[26] L. Gorelick, M. Blank, E. Shechtman, M. Irani, R. Basri, Actions as space–time shapes, Transactions on Pattern Analysis and Machine Intelligence 29 (12) (2007) 2247–2253.
[27] M.D. Rodriguez, J. Ahmed, M. Shah, Action MACH a spatio-temporal maximum average correlation height filter for action recognition, in: CVPR08, Anchorage, USA, 2008, pp. 1–8.
[28] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local SVM approach, in: Proceedings of the ICPR, Cambridge, UK, 2004, pp. 32–36.
[29] A. Oikonomopoulos, M. Pantic, I. Patras, Sparse b-spline polynomial descriptors for human activity recognition, Image Vision Comput. 27 (12) (2009) 1814–1825.

Oscar Deniz received his MSc and PhD from Universidad de Las Palmas de Gran Canaria, Spain, in 1999 and 2006, respectively. He has been associate professor at Universidad de Las Palmas de Gran Canaria from 2003 to 2007 and currently at Universidad de Castilla-La Mancha, Spain. His main research interests are human–robot interaction and computer vision. He is a research fellow of the Institute of Intelligent Systems and Numerical Applications in Engineering and member of IEEE, AEPIA and AERFAI.

Enrique Bermejo received his MSc in Computer Engineering in 2009. Currently he is a PhD Student and associated researcher at E.T.S. Ingenieros Industriales, Universidad de Castilla-La Mancha, Spain. His research interests are computer vision and machine learning.

Gloria Bueno received her MsC from Universidad Complutense de Madrid in 1993, and her PhD from Coventry University in 1998. From 1998 to 2000 Gloria worked as a postdoctoral researcher at Université Louis Pasteur, Strasbourg. In 2000–2001 she worked at CNRS-Institut de Physique Biologique-Hôpital Civil and from 2001 to 2003 she was a senior researcher at CEIT, San Sebastián, Spain. She is currently an Associate Professor at Universidad de Castilla-La Mancha, Spain. Her main research interests include image processing, computer vision and artificial intelligence.

Rahul Sukthankar is senior principal research scientist at Intel Labs Pittsburgh and adjunct research professor in the Robotics Institute at Carnegie Mellon. He received his PhD in Robotics from Carnegie Mellon and his BSE in Computer Science from Princeton. His current research focuses on object recognition and video event detection.