ORIGINAL PAPER

Background subtraction by combining Temporal and Spatio-Temporal histograms in the presence of camera movement

**Andrea Romanoni · Matteo Matteucci ·
Domenico G. Sorrenti**

Abstract Background subtraction is the classical approach to differentiate moving objects in a scene from the static background when the camera is fixed. If the fixed camera assumption does not hold, a frame registration step is followed by the background subtraction. However, this registration step cannot perfectly compensate camera motion, thus errors like translations of pixels from their true registered position occur. In this paper, we overcome these errors with a simple, but effective background subtraction algorithm that combines Temporal and Spatio-Temporal approaches. The former models the temporal intensity distribution of each individual pixel. The latter classifies foreground and background pixels, taking into account the intensity distribution of each pixels' neighborhood. The experimental results show that our algorithm outperforms the state-of-the-art systems in the presence of jitter, in spite of its simplicity.

1 Introduction

Together with frame difference, background subtraction is one of the most common approaches to detect moving objects in a scene captured by a fixed camera [16,25]. Background subtraction is the first processing step in many computer vision systems; for instance, in a tracking algorithm it locates the moving objects to track, while in industrial settings it is used for parts detection in the workspace. For this reason, background subtraction accuracy and robustness are of uttermost importance, even in challenging situations, e.g., when abrupt brightness changes occur, when the brightness of the scene is low or when shadows introduce undesired moving regions.

Background subtraction has been conceived for fixed cameras, but in everyday life such hypothesis is not always true. For instance, hand-held cameras (e.g., from a mobile device) capture shaking movies, surveillance PTZ (Pan, Tilt and Zoom) cameras deliberately move themselves to monitor a wide area, and also fixed cameras can move because of wind or vibrations. Generalized background subtraction has been proposed to deal with not fixed camera [14]. In these cases, background subtraction usually follows a registration step, which aims at reducing misalignments between the current processed frame and the reference background model. However, this is not enough to remove jitter, and a background subtraction algorithm robust to this is required.

In this paper, we propose a simple background subtraction algorithm robust to camera movements. It uses a combination of Temporal and Spatio-Temporal histograms of pixel intensities; both histograms represent temporal distributions, but the former deals with only a single pixel, while the latter deals with its neighborhood.

In Sect. 2, we illustrate current approaches to generalized background subtraction, while in Sect. 3 we describe the proposed algorithm for robust background subtraction in the presence of jitter. In Sect. 4, we discuss the experimental results on two public datasets.

A. Romanoni (✉) · M. Matteucci
Politecnico di Milano, DEIB, Via Ponzio 34/5, 20133 Milan, Italy
e-mail: andrea.romanoni@polimi.it

M. Matteucci
e-mail: matteucci@polimi.it

D. G. Sorrenti
Universitá degli Studi Milano-Bicocca, DISCo, Build U14,
Viale Sarca, 336, 20126 Milan, Italy
e-mail: domenico.sorrenti@unimib.it

2 Related works

Background subtraction is a widespread technique to detect moving objects in a video stream. A moving object (or foreground) is defined as a region of the image that differs from the background static scene. Classical background subtraction algorithms create a model of the background, and then, subtract it from the current frame: pixels whose intensity differs significantly from the background model are classified as foreground. In [2,5,6,18], the authors review several approaches: nowadays the most common are based on Gaussian Mixture Model (GMM) [23] and non-parametric kernel density estimation [7].

The greatest limitation of classical background subtraction is the fixed camera assumption. In many situations (e.g., hand-held camera, camera on an aerial vehicle, or in robotic vision), the camera is moving and the static background assumption is no longer valid. To overcome this issue, novel algorithms have been proposed to face the more general problem of background subtraction with a moving camera, named *Generalized Background Subtraction* as in [14].

Most of the Generalized Background Subtraction algorithms rely on image registration, which is the process of aligning an image with a reference image of the same scene. A straightforward Generalized Background Subtraction algorithm creates a background model of the whole scene by registering the images into a unique mosaic. Then, the algorithm detects moving objects by registering each new frame to the mosaic and performing classical background subtraction between the mosaic and the aligned frame [17]. Registration misalignments, however, result in false positive detections, and this impacts severely on the final outcome.

Usually, the registration step involves an affine or a projective transformation. The latter is more expressive, but it can only map precisely a plane in the 3D scene between two different point of views [10]. Therefore, these approaches handle accurately only the scenes where the background lies on a single plane, for instance the road surface, in a street, but in a more complex scene, with buildings and 3D structures, this simplification does not give accurate results. In these situations, background subtraction needs to be robust to compensate the unavoidable misalignments.

Different approaches have been developed to overcome the single plane assumption. A first approach relies on multiview geometric constraints among consecutive frames.

In [12,20,26], the authors propose different constraints by estimating the camera parallax displacement by means of the plane+parallax decomposition. Unfortunately, this decomposition holds when the camera translates in a direction orthogonal to the viewing ray, but it fails in other cases. In [24] and [21], the authors propose two alternative constraints over the trajectory of the features extracted, for the registration of consecutive images, which are independent from the plane+parallax decomposition. On the other hand, this approach does need a certain delay to determine an adequate trajectory for each feature.

Other approaches, that aim at overcoming the single plane assumption, do not involve the definition of constraints [1,11,13,14,19]. In [13], the authors align each frame with the reference mosaic repeatedly: at each iteration, the region that best fits the current registration is used to determine a plane that is excluded from the successive registrations steps. This method does not deal with moving objects since it only focuses on the background modelling. In [11], the authors embed a statistical description of the homography estimation errors in the GMM framework. Ren et al. [19] improve the GMM method adding the spatial dimension in the background modelling. Finally, in [14], a Bayesian filter enforces the precision of the estimation process of the motion vector during the registration step.

In [15], the authors use a similar approach to Spatio-Temporal background subtraction with histograms, but they did not consider the generalized background subtraction issue (they consider a fixed camera). Instead, in our contribution, we point out the effectiveness of a Spatio-Temporal approach in the generalized background subtraction setting, and we show that combining this approach with the Temporal one we can improve its results.

3 Background subtraction with Temporal and Spatio-Temporal histograms

In the following, we illustrate our Generalized Background Subtraction algorithm starting with a brief explanation of the image registration steps, which, in principle, makes it possible to directly apply a classical background subtraction algorithm. However, the registration always produces misalignments, which result in false positives foreground.

3.1 Limits of image registration

Let F_t be a new frame from the video stream captured from a moving camera, and B_t the reference image, i.e., the background model. To align the two images, we first extract the Good Features to Track [22] from both images, then we find matches among features and we estimate the homography between F_t and B_t by means of RANSAC [8]. In Fig. 1, we show B_t in the left and F_t in the right side: the circles represent the extracted features. The RANSAC algorithm chooses the light blue circles, while it rejects the black ones.

As it can be noticed, the chosen features lie on the floor: this is the dominant plane and it induces the homography that RANSAC estimates. The features that do not lay on the floor, e.g., those on the closet, induce a different homography, and they are rejected by RANSAC. In Fig. 2, we show
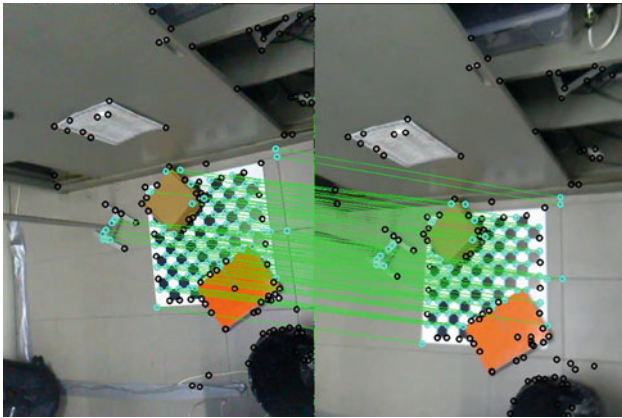
Fig. 1 Features selection for image registration. In this image, we show an example of the features selected by RANSAC. The *left image* is the reference frame, i.e., the background model, while the *right one* is the frame to be registered. The *light blue circles* represent features that are selected by RANSAC to compute the registration homography; the features rejected are *circled in black*. The *lines* between the two frames connect matching features (the *light blue circles*) (color figure online)

Fig. 2 Example of the registration errors introduced by the homography estimated from features, as illustrated in Fig. 1. We show the background overlaid with the registered frame: the *two black lines* represent the border of the registered frame. Notice how the closet is badly aligned, since its features belong to a different plane with respect to the floor

the image F_t transformed according to the estimated homography overlaid with the background B_t: this transformation aligns perfectly the floor, while it produces a significantly misalignment of the closet. In Fig. 3, we show a detail of it: the piece of paper in the closet in the registered image is significantly shifted to the right.
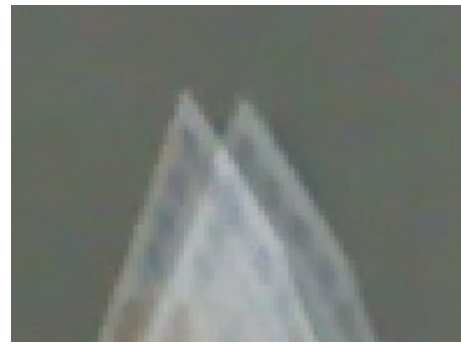

Fig. 3 Detail of Fig. 2: we overlaid the background with the registered frame. This picture clearly shows the misalignment between the background and the registered frame. A classical background subtraction algorithm (Temporal approach) would misclassify the not overlapping parts of the sheet of paper as foreground

3.2 The Temporal + Spatio-Temporal approach

After the registration step, a generalized background subtraction algorithm has to manage the misalignments, which corresponds to a local translation of a set of pixels (e.g., in Fig. 2, the transformation results in the translation of the closet door's pixels).

Classical background subtraction algorithms model the history of each pixel's intensities independently from others—they do that using the median value, a Gaussian distribution, a Mixture of Gaussians or a histogram. Such kind of algorithms classifies each pixel by comparing the pixel intensity in the new frame with the one in the background model. In other words, a classical background subtraction algorithm classifies each pixel individually by taking into account only the temporal changes (Temporal approach). The translations of pixels, induced by the image registration misalignments, would result in false positive detections. For instance, in Fig. 3, the not overlapping region of the piece of sheet would be misclassified as foreground (pixels' intensity differs significantly).

Here, we propose to adopt a *Spatio-Temporal* approach: for each pixel, we model the history of the pixel and its neighborhood—as in the previous case with the median value, a Gaussian distribution, a Mixture of Gaussians or a histogram. Then, we classify each pixel as foreground if both the pixel and its neighborhood intensities differ significantly from the modeled distributions. Therefore, the Spatio-Temporal algorithm classifies each pixel through spatial (pixel neighborhood) as well as temporal information.

Such approach manages the unavoidable misalignments better with respect to the Temporal one. For instance, in Fig. 4 the black dot locates the position of the paper corner in the current frame; the blue dot locates the "real" paper corner position in the background image. A classical, "Temporal" background subtraction algorithm would classify the black dot as foreground, since its intensity differs significantly from

Fig. 4 Detail of Fig. 2 (same detail of Fig. 3): we overlaid the reference background with the registered frame. The *black pixel* of the corner of the paper belongs to the registered frame, while the *blue one* is the real position of this corner of the paper in the reference image. The Spatio-Temporal approach would be able to correctly classify the *black pixel* as background, since it take into account the neighborhood of *black pixel*, i.e., the *circle* in the image, which includes the real position of the pixel (color figure online)

the background. A Spatio-Temporal approach evaluates the neighborhood depicted with the black circle around the black dot, then it would also take into account the *real* background (the blue dot) in the classification process, and, likely, it will not misclassify the pixel.

The Spatio-Temporal algorithm results in a foreground that is less noisy with respect to the one generated by the Temporal algorithm; on the other hand, the misclassified pixels cluster into connected regions. Indeed, if most of the neighborhood of a pixel p is misclassified, then it is likely that p is misclassified too. It has to be noted that the Temporal approach fails more frequently, but most of the misclassified pixels are sparse, and can be rejected by means of morphological operators.

To keep the positive aspects of the two approaches, we apply a binary AND operator to the two images that result from the Spatio-Temporal and the Temporal background subtraction. In Fig. 5, we show the results of the background subtraction for the Temporal, and Spatio-Temporal approaches, as well as for their combination.

3.3 The histogram representation

As mentioned above, there exist different ways to describe both the spatial and temporal distributions of a pixel; in the proposed system, we adopt the histogram representation.

Let f be a distribution and $x_1, x_2, \ldots, x_n$ a set of n i.i.d. samples from this distribution. Let fix a set of m disjoint intervals that cover all the domain of f and build the so-called *bins*; an integer number c_i counts the samples that fall into the ith bin. The bins define a histogram approximating the distribution f, up to a scale factor proportional to n.

We choose the histogram representation instead of the widespread Mixture of Gaussians, for the following reasons:

1. Histograms provide a more natural descriptor of the pixel intensities. While Gaussians are continuous distributions, both pixel intensities and histograms are discrete. Moreover, the histogram representation naturally manages multi-modal distributions with a number of modes not fixed (as the Mixture of Gaussians) although bounded by m.

2. Histograms are easy to parametrize and to tune. The histogram has only one parameter to set, the bin dimension d. If $d = 1$, each bin represents a single pixel intensities; but we want to be more robust to noise, so we populate a bin with similar intensities pixels (for instance $d > 5$). We also want bins uniformly distributed: d must be a power of 2. Moreover, we reject too big bins, since they collect too different intensities. In conclusion, we choose $d = 16$.

3. With histograms, we easily represent and compare spatial distributions (for the Spatio-Temporal background subtraction). In the Spatio-Temporal background subtraction approach, we need to describe a spatial distribution for each pixel. This distribution is again a histogram which summarizes the distribution of the neighborhood, and the ith bin is the sum of ith bins' population of each neighborhood histogram. Moreover, we easily compare this histogram with the current histogram of neighborhood pixel intensities through the Bhattacharyya distance [3].

The only real drawback of this approach is the large amount of memory needed to store the entire background model by means of histograms. If the bin dimension is $d = 16$, the number of bins in a histogram is 16 too. For instance, if we consider a $1,024 \times 768$ image and for each bin, we store a 32 bit integer, the memory requirement would be $1,024 \times 768 \times 16 \times 32 = 245,760,000$ bit, i.e., $\sim$48 MB, which could be considered a large amount of memory in some applications, e.g., in embedded systems. On the other hand, for state-of-the-art Mixture of Gaussians memory requirement would amount to $1,024 \times 768 \times 5 \times (16 + 16 + 8)$ bit, i.e., ~ 19 MB, to represent the background, for 5 Gaussians per-pixel and 16 bit to store the mean and the standard deviation, and 8 bit for the weight associated with each Gaussian. So, even if histograms require a larger amount of memory, this is not so larger that the one needed by a classical Mixture of Gaussians approach.

3.4 The Temporal + Spatio Temporal histograms algorithm

The diagram in Fig. 6 summarizes our Temporal + Spatio-Temporal Histograms algorithm. In the Temporal algorithm, we model each pixel through a histogram, by populating each histogram with the history of pixel intensities. The intensities corresponding to the most populated bins are those that more likely belong to the background. Then, for each new frame, we classify each pixel checking the bin population corre-

(a)

(b)

(c)

(d)

Fig. 5 Example of background subtraction with the Temporal, the Spatio-Temporal and our approach, which combines the two. This figure gives an idea of the complementarity of the foreground mask resulting from the Temporal and the Spatio-Temporal approaches on the frame (**a**). The Temporal approach in **b** estimates a noisy and sparse foreground. Instead, the Spatio-Temporal background subtraction output in **c** is cleaner and the region classified as foreground is thick. If we combine the two results with a binary AND, we obtain that the noisy misclassified pixels from the Temporal approach are discarded by the clean region from the Spatio-Temporal approach. Then, the thick misclassified area resulting from the Spatio-Temporal background subtraction is "sparsified" with the pixel from the Temporal approach. **a** Frame number 861 from the boulevard video of ChangeDetection dataset. **b** Temporal background subtraction. **c** Spatio-Temporal background subtraction. **d** Binary AND of Temporal and Spatio-Temporal results
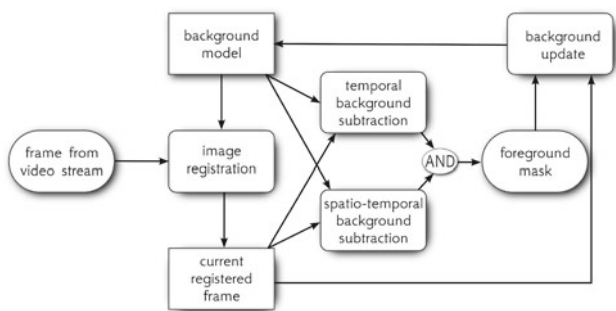
Fig. 6 Overview of the proposed system. We combine the Temporal and the Spatio-Temporal approaches to estimate the foreground. The background model is created after an initialization step and it is then updated according to description in Sect. 3.4.2

sponding to the current intensity: if the population of the bin is high (above a fixed threshold), the pixel is classified as background, otherwise it is classified as foreground.

The Spatio-Temporal background subtraction with histogram starts from our Temporal background subtraction implementation, but, instead of representing only the Temporal distribution of the pixels, it represents also the spatial one. With spatial distribution, we mean the distribution of the intensities in the neighborhood of each pixel.

Let us assume that we know the Temporal background model, i.e., the set of histograms associated with each pixel representing their Temporal distribution. We create the Spatio-Temporal background model from the Temporal one. Let consider the pixel (i, j) and his neighborhood $(i \pm w, j \pm w)$ with w being a small integer, representing the semi-dimension of the neighborhood. For each pixel (i, j), we create a histogram by combining, i.e., by counting the occurrence of all the temporal histograms corresponding to the pixels in the interval $(i \pm w, j \pm w)$. Then, for each new frame, we create another histogram populated with the pixel intensities in the current neighborhood $(i \pm w, j \pm w)$ for the
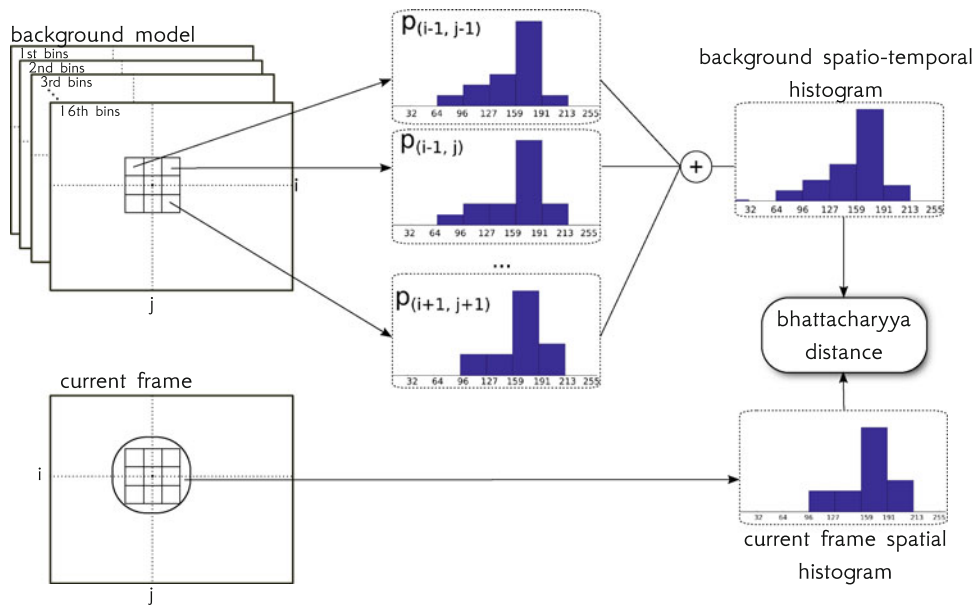
Fig. 7 Simple example of the Spatio-Temporal background subtraction with histograms. In this simple case, the neighborhood is only of one pixel, i.e., $w = 1$. On one side, the background model stores a Temporal histogram for each pixel (each bin is represented by a image), describing its distribution up to the current frame. For the pixel (i, j), we consider its neighborhood. For each bin, we sum the population among the Temporal histogram of each pixel inside the neighborhood. The result is the Spatio-Temporal histogram of the background. On the other side, we populate a spatial histogram with the intensities of the neighborhood of the pixel (i, j) of the current frame. Finally, we compare the two (normalized) histograms (Spatial for the current frame and Spatio-Temporal for the background) with the Bhattacharyya distance

current frame. We classify the pixel (i, j) comparing these two histograms, with the Bhattacharyya distance: if their distance is below a fixed threshold τ, which is unique for the entire image, the pixel is considered foreground, otherwise it is classified as background. In Fig. 7, we give an idea of what happens if $w = 1$.

3.4.1 Abrupt illumination changes

The algorithm illustrated in the previous section does not manage abrupt illumination changes as this may change the whole intensities of a set of images generating a diffused misclassification. Indeed, at time t, our algorithm builds the background histograms only considering the intensities of the pixels up to the time $t - 1$. If in frame t, the illumination changes, for instance, if a cloud shields the sun, current values of the background pixels are slightly different from the values expected from the background model, then they could be misclassified as foreground (Fig. 8a): if the brightness increases, all of the intensities increase and vice versa. This illumination change results in a translation of the histograms' bins along the intensity axis.

To avoid the abrupt illumination issue, we filter out the current difference in median brightness between the background model and the current frame t—we prefer to use median with respect to mean because it is more robust to outliers. We compare the median of the current frame intensities with the median value of the median intensities computed for the past frames. If the two medians differ significantly, we increase or decrease the intensities of each pixel in the current image according to this difference. To cope with drifts in illumination, we compare the current median with the median computed over the last N frame.

More precisely, let us assume that M_t is the median of the pixels intensities p_t in the frame t:

$$M_t = \text{median}\left(p_t^{(i,j)}\right), \quad \forall (i, j) \in \text{frame}_t \tag{1}$$

When a new frame T comes in, we compute the median of the last N median, i.e.:

$$M_N = \text{median}(M_t), \quad \forall t, T - N < t < T. \tag{2}$$

Then, we calculate the difference of the median of the current intensities from this value:

$$O_k = M_N - M_T, \tag{3}$$

and we use this value to change the current image intensities:

$$p_{\text{new}}^{(i,j)} = p_{\text{old}}^{(i,j)} - O_k, \quad \forall (i, j) \in \text{frame}_T, \tag{4}$$

where $p_{\text{old}}^{(i,j)}$ and $p_{\text{new}}^{(i,j)}$ are the pixel of the current image, respectively, before and after the brightness correction.
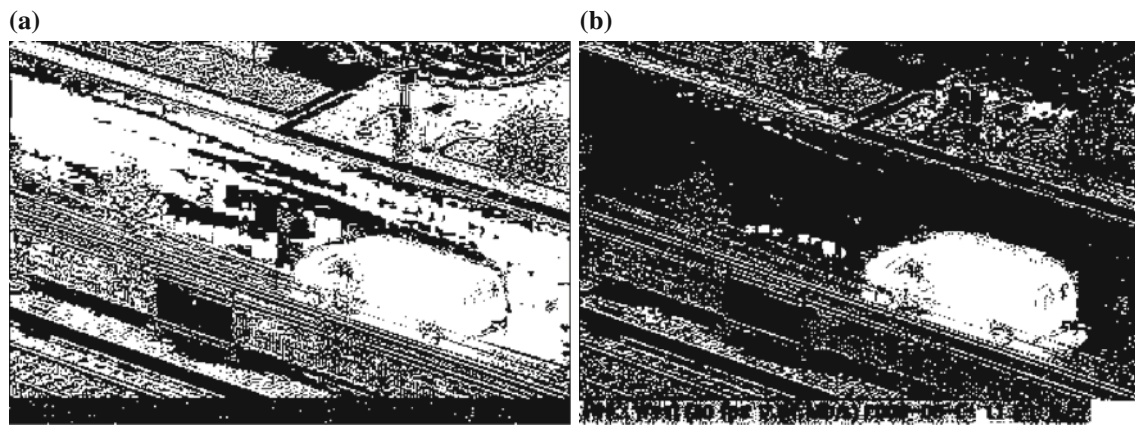
(a) **(b)**

Fig. 8 Effectiveness of our illumination changes management system: In the *left image* (**a**), we show the output of our algorithm without illumination management. Let notice the high amount of misclassified pixels (false positives), i.e., the *white areas* not belonging to the vehicle. Instead the *right image* (**b**) is the foreground mask obtained with the illumination management module. The number of the false positives decreases significantly. **a** Without illumination change management. **b** With illumination change management

In Fig. 8, we show an example of how this strategy diminishes the number of misclassified pixels under abrupt illumination changes. The limit of this approach occurs when a big object appears in the image: the current median value M_T may be polarized by its color, and this may result in misclassified pixels. To overcome this issue, we apply the illumination changes algorithm only to the Spatio-Temporal background subtraction. Therefore, the Temporal algorithm is still robust to the big object issue, since it does not implement the illumination changes module, while the Spatio-Temporal algorithm compensates abrupt illumination changes. The AND combination of the two outcomes removes different kind of misclassified pixels, and then it increases the robustness of the Temporal + Spatio-Temporal algorithm.

3.4.2 Background update

The previous technique deals successfully with abrupt illumination changes, which have a short duration and affect the whole image. In a background subtraction algorithm, we have also to take into account that the global illumination of the scene slowly changes too; we have to update the background model according to these changes.

The histogram representation makes this update step quite straightforward: we add a value λ to the bin of each pixel histogram of the current image. λ weights the current pixel intensity in the update step, defining the algorithm responsiveness. If we assume the segmentation to be perfect, we would update only the background pixels so to avoid including the intensity of a moving object in the background model. In this case, we would set $\lambda = 1$ for background pixels, and $\lambda = 0$ for foreground ones. As a perfect segmentation is not realistic, we choose experimentally $\lambda = 0.5$ for foreground pixels to partially integrate their values in the background model.
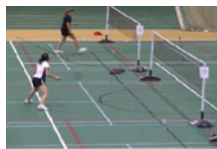
4 Experimental results
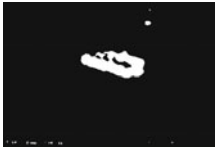
We tested our algorithm, specifically designed for moving camera, on the ChangeDetection dataset [9], which provides real case videos together with the ground-truth foreground masks. We choose the four jittering camera (illustrative results in Table 1), because these videos simulate the problems to be faced after performing the image to background registration: for this reason, we applied directly the background subtraction algorithm (i.e., without the registration step) to the videos.

We compare our results with the state-of-the-art algorithm whose results are published in the ChangeDetection website. For each algorithm, the website reports the following metrics: recall, specificity, false-positive rate (FPR), false-negative rate (FNR), percentage of wrong classifications (PWC), F-Measure and precision[1].

In Table 2, we show the results of our algorithm, which we named TSTH (Temporal and Spatio Temporal Histograms), in comparison to the algorithm listed in the ChangeDetection website (in Fig. 1 we show a sample results comparison among TSTH and the best algorithms in the camera jitter ChangeDetection dataset). Despite the simpleness of the approach, TSTH algorithm outperforms all the others, according to the *average ranking*, which is the overall metric used by the ChangeDetection website to rank an algorithm: average ranking is the mean of each metrics' ranking of the algorithm. The worst metric for TSTH is the recall as the AND operation increases the false-negative count, but the

---

[1] If TP are true positives, FP are false positives, FN are false negatives and TN are true negatives, then: recall = $\frac{TP}{TP+FN}$; specificity = $\frac{TN}{TN+FP}$; FPR = $\frac{FP}{FP+TN}$; FNR = $\frac{FN}{TP+FN}$; PWC = $100 * \frac{(FN+FP)}{TP+FN+FP+TN}$; F-Measure = $\frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$; precision = $\frac{TP}{TP+FP}$.

Table 1 Experimental results with ChangeDetection.net dataset

	badminton	boulevard	sidewalk	traffic
input				
ground truth				
TSTH (proposed)				
CwisarD				
STLBP				
DPGMM				

We list the results of our algorithm and the three best ranking algorithms

very low number of false positives and a high rate of true negatives largely compensate, allowing TSTH to obtain the best result.

In Table 3, we show how the simple binary AND combination and a good illumination management increase the performances of the Temporal and Spatio-Temporal algorithms. All the four combinations that we listed (Temporal + Spatio-Temporal, or just one of the two, with or without illumination management), outperform the individual algorithms results with the exception of the false negative ratio and the recall.

Table 3 also highlights the effectiveness of our illumination management approach (Sect. 3.4.1): when applied to the Temporal and Spatio-Temporal algorithm it increases the overall performances.

We also list some details of the videos (Table 4): the resolution, how many frames we used to initialize the background model and the number of processed frames. We also show the processing frequency: the main implemen-

tation is written in MATLAB, but we also wrote a prototype version in C++/OpenMP: we determined the processing times for both algorithms on a Core i7-2630QM CPU at 2.2 Ghz.

In the previous tests, we fixed the parameters, for all the videos, as follows. The histograms are made up of $d = 16$ bins, as explained in Sect. 3.3. For the Spatio-Temporal algorithm, we choose $w = 6$ (the neighborhood is a 13×13 px square). This value is proportional to the misalignments' translation: a large translation requires a large value of w. In the next paragraph, we show the results when w is fixed and the overall translation changes. After some preliminary test, we set the Bhattacharyya threshold of the Spatio-Temporal algorithm to $\tau = 0.758$ (Sect. 3.4). Finally, we set the window of the illumination compensation module (Sect. 3.4.1) at about the number of the background model initialization frames ($N = 690$), and in the background update module (Sect. 3.4.2) we choose $\lambda = 0.5$ for foreground pixel and $\lambda = 1$ for background ones.

Table 2 Average metrics in the four camera jitter video of the ChangeDetection dataset

Method	Average ranking	Average recall	Average specificity	Average FPR	Average FNR	Average PWC	Average precision	Average FMeasure
TSTH (proposed)	**3.29**	0.7497	0.9912	0.0088	**0.0084**	**1.6440**	0.8255	0.7738
CwisarD	4.14	0.7645	0.9916	0.0084	0.2355	1.7886	0.8091	0.7814
STLBP	5.14	0.8797	0.9841	0.0159	0.1203	1.9191	0.8124	**0.8272**
DPGMM	7.14	0.6988	**0.9930**	**0.0070**	0.3012	1.7707	**0.8426**	0.7477
PSP-MRF	7.87	0.8211	0.9825	0.0175	0.1789	2.2781	0.7009	0.7502
Spectral-360	10.29	0.6709	0.9906	0.0094	0.3291	2.0806	0.8392	0.7156
Multi-layer background subtraction	10.43	0.6903	0.9905	0.0095	0.3097	2.1628	0.7905	0.7311
PBAS	10.57	0.7373	0.9838	0.0162	0.2627	2.4882	0.7586	0.7220
SGMM	11.43	0.7088	0.9869	0.0131	0.2912	2.3761	0.7752	0.7251
KDE—Integrated Spatio-Temporal features	11.86	0.7316	0.9857	0.0143	0.2684	2.4238	0.6993	0.7110
KDE—Spatio-Temporal change detection	11.86	0.7562	0.9816	0.0184	0.2438	2.7450	0.6793	0.7122
SOBS	12.00	0.8007	0.9787	0.0213	0.1993	2.7479	0.6399	0.7086
SGMM-SOD	12.14	0.6351	0.9918	0.0093	0.3649	2.1683	0.8040	0.6724
SC-SOBS	12.71	0.8113	0.9768	0.0232	0.1887	2.8794	0.6286	0.7051
KNN	14.43	0.7351	0.9778	0.0222	0.2649	3.1104	0.7018	0.6894
Bayesian background	16.43	0.5441	0.9886	0.0114	0.4559	2.8807	0.6678	0.5988
GMM—KaewTraKulPong	16.57	0.5074	0.9888	0.0112	0.4926	3.0233	0.6897	0.5761
Chebyshev prob. with Static Object detection	17.43	0.7223	0.9725	0.0275	0.2777	3.6203	0.5960	0.6416
GMM—Stauffer & Grimson	18.00	0.7334	0.9666	0.0334	0.2666	4.2269	0.5126	0.5969
KDE—ElGammal	18.57	0.7375	0.9562	0.0438	0.2625	5.1349	0.4862	0.5720
GMM—RECTGAUSS-Tex	18.71	0.7649	0.9497	0.0503	0.2351	5.6663	0.4179	0.5370
Color histogram backprojection	20.29	0.4688	0.9821	0.0179	0.5312	3.7175	0.5296	0.4822
Local-self similarity	20.43	**0.9764**	0.6158	0.3842	0.0236	36.9570	0.1202	0.2074
GMM—Zivkovic	21.14	0.6900	0.9665	0.0335	0.3100	4.4057	0.4872	0.5670
Mahalanobis distance	21.43	0.7356	0.9431	0.0569	0.2644	6.4390	0.3813	0.4960
Euclidean distance	22.71	0.7115	0.9456	0.0544	0.2885	6.2957	0.3753	0.4874
CDPS	23.57	0.6025	0.9613	0.0387	0.3975	5.3593	0.4397	0.4865
Histogram	24.43	0.7111	0.8412	0.1588	0.2889	16.2797	0.1756	0.2784

The average ranking is the average of the each metric' rankings. The boldface highlights the best value for each metric

Table 3 Comparison of results of Temporal, Spatio-Temporal and TSTH approaches with and without the abrupt illumination change management (for each metric, the bold value underlines the best score)

	Recall	Specificity	FPR	FNR	PWC	Precision	FMeasure
Temporal w/o ill.	**0.8568**	0.9629	0.0371	**0.0044**	3.9633	0.6779	0.7130
Temporal with ill.	0.8458	0.9757	0.0243	0.0049	2.7815	0.7146	0.7503
Spatio-Temporal w/o ill.	0.7962	0.9669	0.0331	0.0066	3.7963	0.6627	0.6874
Spatio-Temporal with ill.	0.7846	0.9796	0.0205	0.0072	2.6375	0.7016	0.7242
TSTH							
Temporal w/o ill. + Spatio-Temporal w/o ill.	0.7754	0.9712	0.0289	0.0071	3.4374	0.7049	0.6957
Temporal w/o ill. + Spatio-Temporal with ill.	0.7500	**0.9912**	**0.0088**	0.0084	**1.6453**	**0.8253**	**0.7738**
Temporal with ill. + Spatio-Temporal w/o ill.	0.7518	0.9901	0.0099	0.0082	1.7336	0.8149	0.7678
Temporal with ill. + Spatio-Temporal with ill.	0.7644	0.9834	0.0166	0.0077	2.3106	0.7614	0.7416

The first four rows show results with the two individual algorithm (Temporal and Spatio-Temporal) with or without illumination management. The other four rows show the results obtained by combining the two algorithm (our TSTH algorithm) with or without the illumination management. We choose the second combination, i.e., Temporal w/o ill. + Spatio-Temporal with ill (see Sect. 3.4.1)

Table 4 Technical parameters of the datasets' videos and processing time in the Matlab and C++ implementation

name	resolution	Num. Bootstrap Frames	Num. processed Frames	fps (MATLAB)	fps (C++/OpenMP prototype)
Badminton	720×480	800	450	0.07	1.34
Boulevard	352×240	790	1810	0.33	8.09
Sidewalk	352×240	800	500	0.32	8.48
Traffic	320×240	900	770	0.38	7.88
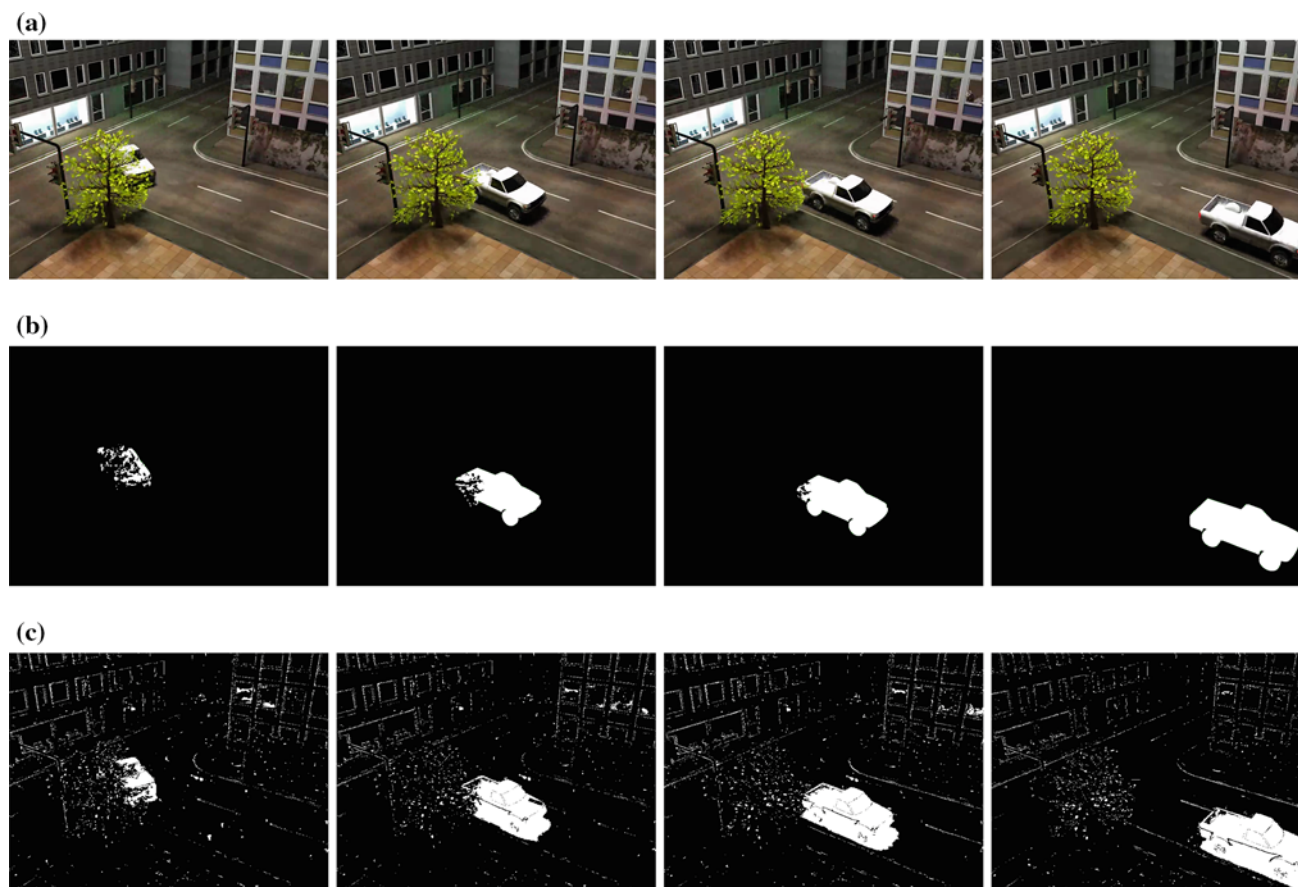
Fig. 9 Illustrative frames of the SABS sequence and our results with the translated version. The salt and pepper noise in our results could be easily removed with morphological operator or with a median filter: **a** Original frames. **b** Ground truth. **c** Our Results

To explain the role of the neighborhood dimension, i.e., the parameter w, in the Spatio-Temporal algorithm, we performed another test on the SABS dataset [4] (in this case we choose $w = 6$, $\tau = 0.8$, $N = 400$, $\lambda = 0.5$ for foreground pixel and $\lambda = 1$ for background ones). This dataset is made up of synthetic videos, so the annotations are very accurate (Fig. 9). For each experiment, we fix a value T, and we translate each frame i by t_i, which is a random value extracted from an uniform distribution in $[-T, T]$. The aim of the random translation is to simulate, for each pixel, the misalignments which can occur in the registration step itself: the misalignments are induced on the whole image, and not only on a limited area.

In Table 5, we list the results for different values of T: notice that for a certain value of T, pixels translate inside a square window of size $2 * T + 1$ px, while the neighborhood considered in the Spatio-Temporal module of our algorithm is a fixed 13 window. As the translation increases (value T), the overall performance degrades as expected. The degradation becomes relevant when $T > 6$, which corresponds to the value of the parameter w (w is the neighborhood window semi-size of the Spatio-Temporal algorithm). As we state in Sect. 3, the well-managed misaligned pixels are those inside the neighborhood; so, the parameter w has to be proportional to the expected length of misalignments' translations.

Table 5 Results with the SABS dataset: we translate each frame i by a random quantity t_i, such that $-T \leq t_i \leq T$

T	Window dimension	Recall	Specificity	FPR	FNR	PWC	Precision	FMeasure
0	no translation	0.7785	0.9929	0.0071	0.0047	1.1523	0.7378	0.7011
1	3×3	0.7855	0.9929	0.0071	0.0046	1.1466	0.7405	0.7003
2	5×5	0.7915	0.9928	0.0072	0.0044	1.1381	0.7431	0.7002
3	7×7	0.7930	0.9928	0.0072	0.0044	1.1361	0.7003	0.7486
4	9×9	0.7954	0.9927	0.0073	0.0043	1.1430	0.7433	0.6975
5	11×11	0.7936	0.9924	0.0076	0.0044	1.1725	0.7378	0.6894
6	13×13	0.7936	0.9915	0.0085	0.0044	1.2637	0.7232	0.6642
7	15×15	0.7957	0.9894	0.0106	0.0043	1.4670	0.6930	0.6138
8	17×17	0.7971	0.9871	0.0129	0.0043	1.6856	0.6619	0.5660
9	19×19	0.7952	0.9813	0.0187	0.0044	2.2572	0.5944	0.4746
10	21×21	0.7999	0.9781	0.0219	0.0042	2.5600	0.5644	0.4360

We list the results with different values of T, and the corresponding window dimensions, to show how the algorithm performs if the misalignment increases. In these experiments, the neighborhood dimension is $w = 6$, i.e., the neighborhood is a 13×13 square. The performance degrades when the misalignment exceeds this neighborhood window, as a consequence of the reasoning in Sect. 3 and in Fig. 4

5 Conclusion

In this paper, we proposed a simple, but effective, approach to background subtraction with moving camera. The state-of-the-art algorithms usually face this problem performing three steps: the image registration, some refinement of the alignment results and the background subtraction. Indeed, to apply the classical background subtraction algorithms, which rely on the fixed camera assumption, a camera alignment step is needed. We designed our background subtraction algorithm to take into account the unavoidable misalignments resulting from the registration.

Our main contribution is to combine a Temporal and a Spatio-Temporal approaches. The former is the most common in classical background subtraction literature: it classifies each pixel relying on its Temporal distribution. The latter, instead, considers the neighborhood of each pixel, so it classifies pixels relying on their spatial distribution. To the best of our knowledge, this paper is the first that clarifies how a Spatio-Temporal approach to background subtraction can be effective in background subtraction with a moving camera, to manage some of the misclassified pixels generated from misalignment of the registration step.

Moreover, we noticed that the foreground mask estimated by the two approaches are in some sense complementary: Temporal approach is less robust against misalignments, but produces sparse misclassified pixels which are easy to remove with morphological operators. The Spatio-Temporal approach produces a cleaner result in most of the images, but in some areas, it generates dense groups of misclassified pixels. The proposed combination of the two approaches is simple, just a binary AND, and effective, as the experimental results show.

In our implementation, we adopt a histogram-based representation of the pixel intensities distribution, since we believe it is the most adequate tool for this framework: histograms are easy to implement and they also represent more naturally the discrete intensity distribution of the pixel with respect to other tools as the Gaussian distribution, or the Mixture of Gaussians. However, the proposed approach could be extended to other non-parametric models.

We tested our algorithm with the camera jitter data-set of ChangeDetection.net and, despite its simplicity, it outperformed the state-of-the-art algorithms.

References

1. Azzari, P., Stefano, L.D., Bevilacqua, A.: An effective real-time mosaicing algorithm apt to detect motion through background subtraction using a PTZ camera. In: IEEE Conference on Advanced Video and Signal Based Surveillance, 2005. AVSS 2005, pp. 511–516 (2005). doi:10.1109/AVSS.2005.1577321
2. Benezeth, Y., Jodoin, P., Emile, B., Laurent, H., Rosenberger, C.: Review and evaluation of commonly-implemented background subtraction algorithms. In: 19th International Conference on Pattern Recognition, 2008. ICPR 2008, pp. 1–4 (2008). doi:10.1109/ICPR.2008.4760998
3. Bhattacharyya, A.: On a measure of divergence between two multinomial populations. Sankhyā Indian J. Stat. (1933–1960) 7(4), 401–406 (1946)
4. Brutzer, S., Hoferlin, B., Heidemann, G.: Evaluation of background subtraction techniques for video surveillance. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1937–1944 (2011). doi:10.1109/CVPR.2011.5995508
5. Cheung, S.S., Kamath, C.: Robust techniques for background subtraction in urban traffic video. In: Proc. SPIE, Visual communications and image processing, 2004, vol. 5308, pp. 881–898 (2004). doi:10.1117/12.526886
6. Cristani, M., Farenzena, M., Bloisi, D., Murino, V.: Background subtraction for automated multisensor surveillance: a comprehen-

sive review. EURASIP J. Adv. Signal Process **2010**, 43:1–43:24 (2010). doi:10.1155/2010/343057

7. Elgammal, A., Harwood, D., Davis: Non-parametric model for background subtraction. In: IEEE ICCV Frame-Rate Workshop (1999)

8. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)

9. Goyette, N., Jodoin, P., Porikli, F., Konrad, J., Ishwar, P.: Changedetection.net: A new change detection benchmark dataset. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1–8 (2012). doi:10.1109/CVPRW.2012.6238919

10. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, vol. 2. Cambridge University Press, Cambridge (2000)

11. Hayman, E., Eklundh, J.: Statistical background subtraction for a mobile observer. In: Proceedings of the Ninth IEEE International Conference on Computer Vision, 2003, pp. 67–74. IEEE, New York (2003)

12. Irani, M., Anandan, P.: A unified approach to moving object detection in 2D and 3D scenes. IEEE Trans. Pattern Anal. Mach. Intell. **20**(6), 577–589 (1998). doi:10.1109/34.683770

13. Jin, Y., Tao, L., Di, H., Rao, N., Xu, G.: Background modeling from a free-moving camera by multi-layer homography algorithm. In: 15th IEEE International Conference on Image Processing, 2008. ICIP 2008, pp. 1572–1575 (2008). doi:10.1109/ICIP.2008.4712069

14. Kwak, S., Lim, T., Nam, W., Han, B., Han, J.H.: Generalized background subtraction based on hybrid inference by belief propagation and Bayesian filtering. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 2174–2181 (2011). doi:10.1109/ICCV.2011.6126494

15. Li, B., Yuan, B., Miao, Z.: Moving object detection in dynamic scenes using nonparametric local kernel histogram estimation. In: 2008 IEEE International Conference on Multimedia and Expo, pp. 1461–1464 (2008). doi:10.1109/ICME.2008.4607721

16. Migliore, D.A., Matteucci, M., Naccari, M.: A revaluation of frame difference in fast and robust motion detection. In: Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks, VSSN '06, pp. 215–218. ACM, New York (2006). doi:10.1145/1178782.1178815

17. Mittal, A., Huttenlocher, D.: Scene modeling for wide area surveillance and image synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2000, vol. 2, pp. 160–167 (2000). doi:10.1109/CVPR.2000.854767

18. Piccardi, M.: Background subtraction techniques: a review. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3099–3104 (2004). doi:10.1109/ICSMC.2004.1400815

19. Ren, Y., Chua, C.S., Ho, Y.K.: Statistical background modeling for non-stationary camera. Pattern Recogn. Lett. **24**(1–3), 183–196 (2003). doi:10.1016/S0167-8655(02)00210-6. http://www.sciencedirect.com/science/article/pii/S0167865502002106

20. Sawhney, H., Guo, Y., Asmuth, J., Kumar, R.: Independent motion detection in 3D scenes. In: Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999, vol. 1, pp. 612–619 (1999). doi:10.1109/ICCV.1999.791281

21. Sheikh, Y., Javed, O., Kanade, T.: Background Subtraction for freely moving cameras. In: 2009 IEEE 12th International Conference on Computer Vision, vol. 2, pp. 1219–1225 (2009). doi:10.1109/ICCV.2009.5459334

22. Shi, J., Tomasi, C.: Good features to track. In: 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94, pp. 593–600. IEEE, New York (1994)

23. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999, vol. 2, pp. 2 vol. (xxiii+637+663) (1999). doi:10.1109/CVPR.1999.784637

24. Sugaya, Y., Kanatani, K.: Extracting Moving Objects from a Moving Camera Video Sequence. In: 10th Symposium on Sensing via Image Information, pp. 279–284 (2004)

25. Trucco, E., Verri, A.: Introductory Techniques for 3-D Computer Vision, vol. 93. Prentice Hall, New York (1998)

26. Yuan, C., Medioni, G., Kang, J., Cohen, I.: Detecting motion regions in the presence of a strong parallax from a moving camera by multiview geometric constraints. IEEE Trans. Pattern Anal. Mach. Intell. **29**(9), 1627–1641 (2007). doi:10.1109/TPAMI.2007.1084

Author Biographies

Andrea Romanoni received the B.S. and M.S. degrees in Computer Engineering at Politecnico di Milano in 2009 and 2012, respectively. He then was a Research Assistant at Università degli Studi Milano-Bicocca, for the SMELLER project. Currently, he is a PhD student at Politecnico di Milano. His research interests are in Computer Vision and Robotics: in particular, in Vehicle Tracking, 3D Dense Reconstruction and Visual SLAM.

Matteo Matteucci received the Laurea degree in Computer Engineering from Politecnico di Milano, in 1999, the M.S. in Knowledge Discovery and Data Mining from Carnegie Mellon University in 2002, and the Ph.D. in Computer Engineering and Automation from Politecnico di Milano, in 2003. Since 2005 he is a Faculty member at Politecnico di Milano. Currently, he is an assistant professor in the Department of Electronics Information and Bioengineering. He has published more than 100 refereed journal and conference papers. His research activity is in Robotics mainly focused on perception, tracking, sensor fusion, simultaneous localization and mapping. He has been involved in the Euron Special Interest Group on Good Experimental Methodologies and Benchmarking and he is in the IEEE RAS Standard Group for the definition of IEEE P1873/D1 Draft Standard for Robot Map Data Representation for Navigation.

Domenico G. Sorrenti obtained a maturità classica (high school, with literature and philosophy orientation, diploma) at Liceo G. Carducci, Milano, in July 1981. He studied Electronic Engineering at Politecnico di Milano, and obtained the Laurea (Master) in February 1989. He then enrolled in the PhD programme in Computer Engineering and Automation at Politecnico di Milano, and obtained the PhD degree in December 1992. Currently, he is an associate professor in computer engineering with the Department Informatica, Sistemistica e Comunicazione of Università degli Studi di Milano-Bicocca. His research interests are in robotic perception for autonomous vehicles, and video-surveillance. His research has been funded by EEC, National agencies (mainly MIUR), Lombardy region, and private companies.