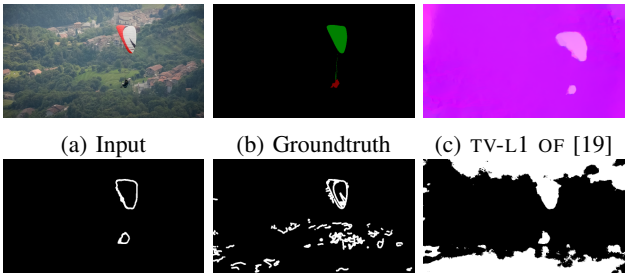
Contour Detection of Multiple Moving Objects in Unconstrained Scenes using Optical Strain

Maria Oliver-Parera, Julien Muzeau, Patricia Ladret, Pascal Bertolino
{maria.oliver-parera, julien.muzeau, patricia.ladret, pascal.bertolino}@gipsa-lab.grenoble-inp.fr
Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab
Grenoble, France

***Abstract*—Moving Object Detection (MOD) is still an active area of research due to the amount of scenarios it can tackle and the different characteristics that may appear in them. Therefore, getting a unique method that performs well in all the situations becomes a challenging task. In this paper we address the MOD problem from a physical point of view: given the optical flow between two images, we propose to find its motion-boundaries by means of the optical strain, which gives information about the deformation of any vector field. As optical strain detects all the motions from a sequence, we propose to work on temporal windows and apply thresholding on them in order to separate noise from real motion. The proposed approach shows competitive results when compared to other methods on known datasets.**

***Index Terms*—Moving Object Contour Detection, Optical Strain, Optical Flow, Temporal Windows, Adaptive Threshold.**



(a) Input (b) Groundtruth (c) TV-L1 OF [19]

(d) Ours. $P = 0.82$, $R = 0.58$, $F_1 = 0.68$ (e) Result from [14]. $P = 0.78$, $R = 0.07$, $F_1 = 0.14$ (f) Result from [13]. $P = 0.17$, $R = 0.59$, $F_1 = 0.26$

Fig. 1: Frame 23 from *paragliding* sequence from [22].

I. Introduction

Moving Object Detection (MOD) aims at recognising the physical movement of the objects in a video scene. It remains an open problem due to the wide range of real scenarios where it can be used, such as video surveillance, trajectory similarity search, traffic monitoring, or tracking, among others [1]–[3]. Moreover, each of these scenarios may present different challenging situations such as illumination variation, occlusions, complex background or object dimension variation.

Moving Object Detection algorithms can be classified into the following approaches: temporal differences, background-foreground subtraction or optical flow methods. Temporal difference models apply pixel-wise difference among consecutive frames [4]–[6]. However these methods fail in the cases of slow motion or similar texture between foreground and background. On the other hand, background-foreground subtraction methods aim at constructing a robust background model that will be subtracted from all frames to get the foreground [7]–[11]. However, they are computationally expensive and require a big amount of frames in order to model the background. Moreover, they fail in the cases of camera instability. In order to overcome these difficulties, optical flow methods use the internal motion information between consecutive frames, which is given by the Optical Flow (OF) [12]–[16]. These methods got recently an upraise thanks to the improvement of the optical flow models [17]–[19].

All authors acknowledge the Auvergne-Rhône-Alpes region for the funding of this work.

In their turn, OF models can be distinguished between region-based techniques, which aim at determining the shape of the moving objects, and contour-based algorithms, which detect their boundary [20], [21]. Let us note that, under the assumption that the boundaries are well detected and complete, both methods are complementary as it is straightforward to obtain the object shape from it boundaries and vice-versa. The main drawback of these methods is their sensibility to noise, which is dissipated when the models are contour-based [14].

We propose in this article a contour-based optical flow model for Moving Object Detection[1]. Contours are computed by means of the Optical Strain (OS), which corresponds to the deformation of the Optical Flow and is able to detect all the motions in a video. In order to distinguish among real motion and noise (such as the movement of tree leaves or water) we propose to work on temporal windows. Each of these windows is considered as the minimum instance of our model, therefore we compute a unique optical flow (by means of Vector Addition, detailed in Eq. (5)) which is able to capture long-distance variations, from the first to the last frame of each window. The underlying goal behind this idea is that the object's motion will have a steady behaviour along the window, while the noise will be erratic. Therefore, the real motion gets accumulated, while the noise is dissipated, allowing a thresholding step to successfully suppress it.

The main contributions presented in this paper are the following:

[1]Code available at: https://github.com/mariaoliverparera/mod-opticalStrain

- Detection of the boundaries of the moving objects by means of the optical strain.
- Inclusion of temporal information by means of overlapping temporal windows.
- Usage of Vector Addition (Eq. (5)) in order to get a unique optical flow through each window.
- Adaptive threshold to each window.

The remainder of this paper is organised as follows: in Section II the concept of optical strain is presented, in Section III we detail our model and emphasize the usage of temporal windows. Section IV is devoted to present the experimental results. Finally, we summarise the paper and present future work in Section V.

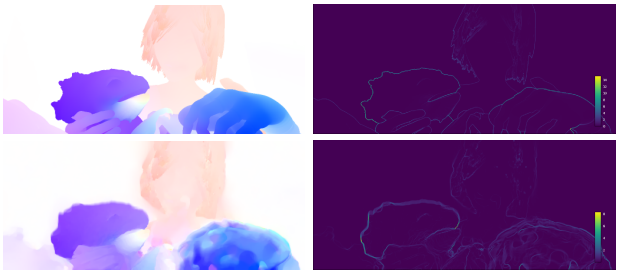


Fig. 2: Behaviour of the optical strain (second column) over the optical flow groundtruth (first row) or the TV-L1 optical flow [19], [23] (second row). We can see that using the OF groundtruth produces sharper results as it is better defined. We can also observe that the values of the optical strain are bigger in the areas where the optical flow has a higher magnitude. The image corresponds to the frame 12 of the sequence *bandage 2* from SINTEL dataset [24].

II. Preliminaries: Optical Strain

It is well-known that a fluid $\mathbf{u}$ can be decomposed, in a small neighborhood of each point, into the sum of a (rigid) translation, a deformation and a (rigid) rotation vector [25]. For instance, given a point $\mathbf{x} \in \mathbb{R}^n$, and a nearby point $\mathbf{y} = \mathbf{x} + \mathbf{h}$, with $\mathbf{y} \in \mathbb{R}^n$ and $\mathbf{h} \in \mathbb{R}^n$, we can linearize it using Taylor's expansion:

$$\mathbf{u}(\mathbf{y}) = \mathbf{u}(\mathbf{x}) + \mathbf{J_x u}(\mathbf{x})\mathbf{h} + o(|\mathbf{h}|), \quad \mathbf{h} \to 0, \tag{1}$$

where $\mathbf{J_x}$ denotes the Jacobian matrix at point $\mathbf{x}$. The first term represents the translation, while the second term contains information about the deformation (scaling and shearing) and rotation of the fluid. As any matrix, $\mathbf{J_x u}(\mathbf{x})$ can be decomposed into the sum of a **symmetric matrix** and an **anti-symmetric matrix**, therefore the Jacobian can be split up into:

$$\begin{aligned}\mathbf{J_x u}(\mathbf{x}) = &\underbrace{\frac{1}{2}\left(\mathbf{J_x u}(\mathbf{x}) + (\mathbf{J_x u}(\mathbf{x}))^T\right)}_{\boldsymbol{\varepsilon}} + \\ &\underbrace{\frac{1}{2}\left(\mathbf{J_x u}(\mathbf{x}) - (\mathbf{J_x u}(\mathbf{x}))^T\right)}_{\boldsymbol{\eta}},\end{aligned} \tag{2}$$

Fig. 3: Example showing the effect of computing the optical strain over a blurred optical flow, obtained using a Gaussian filter of standard deviation σ_g. In the first row we present the optical flow, while the second row is the corresponding optical strain. In the first column we show the results obtained with the original optical flow, while the second and third columns correspond to a blur of $\sigma_g = 0.8$ and $\sigma_g = 1.8$, respectively. We can observe that blurring more the optical flow results in more highlighted boundaries.

where $\boldsymbol{\varepsilon}$ is the symmetric part and $\boldsymbol{\eta}$ the anti-symmetric one. As proven in [25], the symmetric part contains information about the deformation while the anti-symmetric part describes the rotational movement of the fluid.

As we are interested in the deformation of the motion, we only focus on the symmetric part of the matrix, which is denoted as strain tensor, or **optical strain** in the Computer Vision community.

A. Optical Strain Magnitude

Given a fluid $\mathbf{u} : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^2$ defined as $\mathbf{u}(x, y) = (u_1(x, y), u_2(x, y))$, its expanded optical tensor is given by

$$\boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_{xx} = \frac{\partial u_1}{\partial x} & \varepsilon_{xy} = \frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) \\ \varepsilon_{xy} = \frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) & \varepsilon_{yy} = \frac{\partial u_2}{\partial y} \end{pmatrix}. \tag{3}$$

Then, as shown in [26], the optical strain magnitude for each pixel can be computed as:

$$s = \sqrt{\varepsilon_{xx}^2 + 2\varepsilon_{xy}^2 + \varepsilon_{yy}^2}. \tag{4}$$

Fig. 2 presents two examples of optical strain magnitude: the areas with a higher magnitude for the optical flow produce a stronger value also for the optical strain.

III. The Model

Let us consider a video sequence $I(\mathbf{x}, t) : \Omega \times \{1, \dots, T\} \to \mathbb{R}^{\{1,3\}}$, where $\Omega \subset \mathbb{R}^2$ is the frame domain and $\{1, \dots, T\}$ denotes the set of discrete times. Let $\mathbf{u}_t(\mathbf{x}) : \Omega \to \mathbb{R}^2$ be the optical flow between the frames $I(\mathbf{x}, t)$ and $I(\mathbf{x}, t + 1)$, i. e. the apparent motion between a pixel $\mathbf{x} \in \Omega$ at time t and the corresponding one at time $t+1$. To facilitate the detection of the boundaries of the moving objects we propose to filter the optical flow. As it can be seen in Fig. 2 the optical strain may detect disjoint boundaries. These boundaries can be widened and joined by applying a Gaussian filter, with $\sigma_g = 1.8$, to each channel of the optical flow. Moreover, this process helps to reduce small noise motion produced by the optical flow. An

example showing the effect of the Gaussian filter is presented in Fig. 3, where it is explicit that the boundaries obtained with $\sigma_g = 1.8$ are highlighted.

Then, to distinguish objects from noise, we propose to work on the temporal level by means of frame windows (or subsequences) $\mathcal{W}_i$ with $i \in \mathbb{N}^+$, as illustrated in Fig. 4. Each window $\mathcal{W}_i$ is made up of consecutive optical flows and is determined by two parameters: the number of frames n in the window and the shift σ ($\sigma < n$), which is the step-size between consecutive windows. Therefore, each window $\mathcal{W}_i$ contains the following n consecutive optical flow frames: $\{\mathbf{u}_{(i-1)\sigma+1}(\mathbf{x}), \ldots, \mathbf{u}_{(i-1)\sigma+n}(\mathbf{x})\}$.

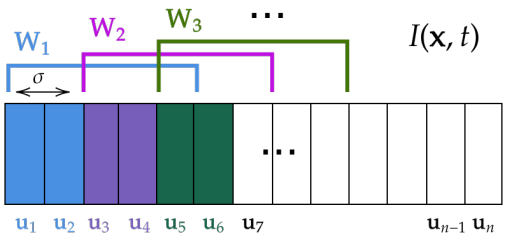


Fig. 4: Scheme showing the temporal overlapping windows of Optical Flow used to compute the Optical Strain. In this example we use $n = 5$ and $\sigma = 2$.

The temporal window organization allows us to work on small sequences, where the real motion is accumulated, producing high values for the optical strain, contrary to the noise, for which the optical strain will have low values. In this way, as it can be seen in Fig. 7, we can separate real motion from noise with a two step threshold over the image: in the first step we compute the thresholds taking into account all the movement in each specific window $\mathcal{W}_i$, while the second step is devoted to the threshold application to the individual frames. Finally, we apply mathematical morphology in order to join thin boundaries which might have been too much narrowed in the thresholding process. In the following of this section we give deeper insights to each of these steps.

1) **Threshold computation:** In the situation of this paper, the optical flow between two consecutive frames is irrelevant, as some values can be caused by noise, from the scene or from the algorithm itself. Actually, obtaining the optical flow from the first to each of the other images of the window is more interesting because an object with a regular movement has greater values, while objects with irregular movement (considered as noise) get lower values. Unfortunately, the optical flow [27], [28], as illustrated in Fig. 6b, fails in large distances. In order to overcome this fact and compute motion vectors between non-consecutive frames, we propose to follow the idea from [29] and compute a unique optical flow, which goes from the first frame of the window until the last one, by means of Vector Addition (VA), as represented in Fig. 5. For each frame, VA can be computed with the following recursive

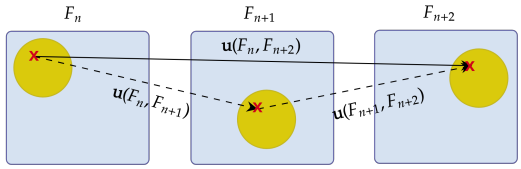


Fig. 5: Scheme showing the computation of the Vector Addition. It is obtained by following the flow through the desired frames.

(a) OF groundtruth (frame 8). (b) OF from frame 1 to 8 [19]. (c) VA from frame 1 to 8 (Eq. (5)).

Fig. 6: Comparison between vector addition and optical flow in long distances. This example shows that vector addition gives a better approximation of the groundtruth.

equation:

$$\begin{aligned} \mathbf{v}_1(\mathbf{x}) &= \mathbf{u}_1(\mathbf{x}) \\ \mathbf{v}_k(\mathbf{x}) &= \mathbf{v}_{k-1}(\mathbf{x}) + \mathbf{u}_k\left(\mathbf{x} + \mathbf{v}_{k-1}(\mathbf{x})\right) \quad k \geq 2, \end{aligned} \tag{5}$$

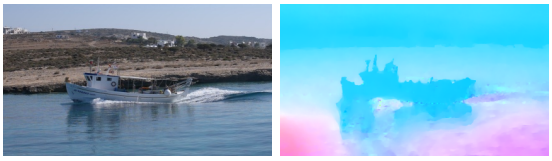
where $\mathbf{v}$ is the vector addition and k corresponds to the frame index.

As it can be observed in Fig. 6c, vector addition shows better performance than the optical flow between two non-consecutive images. Then, the optical strain magnitude is computed over each VA of the window using Eq. (3), i. e. we obtain a succession of optical strain magnitudes for each window $\mathcal{W}_i$: $\{s_{(i-1)\sigma+1}(\mathbf{x}), \ldots, s_{(i-1)\sigma+n}(\mathbf{x})\}$. Once the optical strain magnitudes for the whole window are computed, we need to determine the threshold τ_i for each window $\mathcal{W}_i$ that allows to remove small noise magnitudes and thus to keep only relevant information about the boundaries. Following [30], we propose the next window-adaptive threshold:

$$\begin{aligned} \tau_i = &\min_j \min_{\mathbf{x}\in\Omega} s_j(\mathbf{x}) \\ &+ \rho \left(\max_j \max_{\mathbf{x}\in\Omega} s_j(\mathbf{x}) - \min_j \min_{\mathbf{x}\in\Omega} s_j(\mathbf{x}) \right), \end{aligned} \tag{6}$$

with $\rho \in (0, 1)$. This parameter is a percentile within the range of optical strain magnitudes. For example, applying this threshold with $\rho = 0.05$ means removing 5% of the lowest values, which corresponds to noisy motion. The full algorithm for threshold computation is summarized in Algorithm 1.

2) **Threshold application:** In this step we apply the threshold values obtained for each window to each frame. As $\sigma < n$, each optical strain frame has several associated thresholds, in particular it has as many thresholds as windows it belongs to. We propose to get a unique threshold for each optical strain frame of the video by using the Weighted Moving Average (WMA). For each frame, the weight given to each threshold depends on the degree of belonging of this frame to each of the windows it belongs to. This information is provided by the



(a) Input (b) Optical Flow

(c) Optical Strain (d) Thresholded OS

Fig. 7: Example showing the importance of applying the thresholding step. In this case, it is observable that the Optical Strain result (image 7c) catches also the moving background and the waves. In contrast, after thresholding ($\tau = 0.25$), only the boat is left. Frame 22 from *boat* sequence from [22].

Data: Video I
Result: Threshold τ_i for each window $\mathcal{W}_i$
for $\mathcal{W}_i \in I$ **do**
  for $F_j \in \mathcal{W}_i$ **do**
    Compute VA from F_1 to F_j ▷ Eq. (5)
    Compute OS s_j ▷ Eq. (6)
  end
  Compute threshold τ_i ▷ Eq. (7)
end

Algorithm 1: Threshold computation.

position of the frame according to the end of each window it belongs to. Thus, more weight is given to the first frames of the window, less to the last ones and the weight evolution in between is linear. This decision comes from the fact that the first frames of each window suffer less deformation of the vector addition thanks to their proximity to the first frame of the window. In particular, for each frame t of the video, we get a unique threshold as follows:

$$\tilde{\tau}_t = \frac{\sum_{i=a}^{b}(1+(i-1)\sigma+n-t)\tau_i}{\sum_{i=a}^{b}(1+(i-1)\sigma+n-t)}, \tag{7}$$

where n is the number of frames on each window, σ is the shift between consecutive windows, τ_i is the threshold that corresponds to the window i, a is the index of the first window frame t belongs to, and b is the index of the last one. Specifically, these two indexes are computed as:

$$a = \max\left(\left\lceil \frac{t-n}{\sigma} \right\rceil, 0\right) + 1, \tag{8}$$

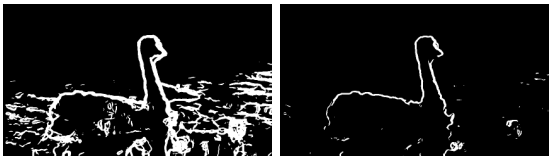
$$b = \left\lfloor \frac{\min(t,T)-1}{\sigma} \right\rfloor + 1. \tag{9}$$

For instance, let us consider the window scheme in Fig. 4 where $n = 5$ and $\sigma = 2$ and in particular, in the optical flow frame $\mathbf{u}_5$. In this case $a = 1$ and $b = 3$. Indeed, the frame $\mathbf{u}_5$ belongs to windows $\mathcal{W}_1, \mathcal{W}_2$ and $\mathcal{W}_3$. Then, the expression of the filtered threshold is given by:

$$\tilde{\tau}_5 = \frac{1\times\tau_1 + 3\times\tau_2 + 5\times\tau_3}{9}, \tag{10}$$

which gives more weight to the frame of the third window, as frame 5 is the first frame of this window.

3) **Mathematical morphology:** The thresholded image may contain disconnected boundaries. In order to close them, we propose to apply a mathematical morphology operator, namely a closing, which consists in a dilation followed by an erosion. The structural element is chosen to be a disk with radius 3 pixels.



(a) $\sigma = 1$ and $n = 1$. (b) $\sigma = 5$ and $n = 10$.

Fig. 8: Comparison of the performance of the proposed method in two different situations. We can observe as in the limit case with $\sigma = n = 1$ the model is not able to remove the noise as well as when we use bigger values. Both results are computed over the same frame of the sequence blackswan from DAVIS dataset [22].

A. *Limit case:* $\sigma = 1$ *and* $n = 1$

In order to highlight the importance of the temporal windows, we show with one example how the model behaves when we use $\sigma = 1$ and $n = 1$, i. e. when we compute the optical strain and its threshold frame by frame. In Fig. 8, we can see as the noise remains (especially the wavelets on the water) when the temporal information is not used. As previously explained, without the temporal notion, all the motions have the same importance, keeping the noise.

IV. Experimental Results

This section provides the results of our method and the comparison against two other models for Moving Object Detection that use the Optical Flow.

A. *Datasets*

We evaluate our model on three datasets that contain real data: the public datasets DAVIS [22] and SBI 2015 [31], and a private dataset built to ensure the security on the skilifts in the ski resorts, therefore it contains videos of skilifts tracks.

1) DAVIS *[22]:* This dataset includes 90 sequences, which contain from 25 to 104 images (its median is 71), acquired with a non-static camera, with one to four relevant objects. The main challenges of this dataset are object deformation, i. e. the objects are non-rigid; fuzzy boundaries produced by fast motion; occlusions, i. e. the objects may be partially or totally occluded; out-of view, i. e. the objects may be clipped by the boundaries of the image; camera vibration; hetereogeneous objects, i. e. the same object may have different colors;

interacting objects, i. e. there are multiple objects on the scene; or, dynamic background.

2) SBI 2015 [31]: It consists in 13 sequences with 90 to 740 frames (its median is 350). The main challenge of this dataset is the low resolution of the images and the presence of big moving objects which occupy most of the spatial domain.

3) Skilifts: The dataset presents 76 sequences of 40 frames each for several skilifts from different ski resorts. The sequences are recorded in both, summer and winter, so that the background is not always homogenous, as the images may have white-snowy background or green. The dataset also contains different environmental conditions, from sunny days to blizzard. The main challenges of this dataset are object-occlusions; out-of-view; high scale-variation, i. e. the area of bounding box enclosing the objects have big variations; shape complexity, i. e. the objects have complex boundaries containing thin parts and holes; appearance change, mainly due to the different illumination along the day; or, camera vibrations. Moreover, the associated groundtruth (see Fig. 11b for instance) was determined manually through pixel by pixel annotation.

B. Models Compared

We compare our model with the methods proposed by Patel and Parmar [13] and Tang et al. [14], which also use the optical flow as input to obtain the moving objects of the scene. Both methods work on the spatial domain, taking into account only one frame each time.

1) Patel and Parmar [13]: This method proposes to compute the edges directly over the optical flow. In particular, they propose an optical flow thresholding method based on the mean and the standard deviation of each channel of the optical flow. Then, morphological operations are applied.

2) Tang et al [14]: A model that works with both, the image and the optical flow magnitude, is presented. It is proposed to compute the edges over the images using an edge detector method. Then, they apply the Otsu threshold method [32] over the optical flow magnitude and the Minimimum Error Threshold [33] for the image containing the edges in order to highlight the main edges. Subsequently, only the edges appearing in both, the image and the optical flow, are considered. Finally, a dilation is applied.

C. Comparison Methodology

As stated in Section III, our method computes the boundaries of the moving objects of the scene. In contrast, all the aforementioned datasets provide the complete shape (pixel by pixel) of the objects as groundtruth. In order to give a fair comparison, we propose to compute the Bounding Box (BB) of the different outputs, as well as for the objects of the groundtruth. The BB is computed as follows. Let us suppose the output image only has one connected component, i. e. only one object, which contains $\tilde{n}$ pixels. Then this connected component can be defined as $\mathcal{O} = \{(x_1, y_1), \ldots, (x_{\tilde{n}}, y_{\tilde{n}})\}$.

Fig. 9: Examples showing the bounding box (in purple) around the objects (in white). The first column shows groundtruth examples, while the second column shows our results. Finally, the first row provides an example with one object, while in the second row there are several objects. Images from [22].

Consequently, the BB can be retrieved as the rectangle whose corners are:

$$x_{\min} = \min_i x_i, \qquad x_{\max} = \max_i x_i, \tag{11}$$

$$y_{\min} = \min_i y_i, \qquad y_{\max} = \max_i y_i. \tag{12}$$

In contrast, if the image have more than one connected component, the same process is applied to each one of the connected components of the scene, i. e. we compute the BB for each connected component. In particular, for all the models the number of objects is given as input, and we assume that the image contains the same number of connected components than objects. Two examples are shown in Fig. 9.

We compare these bounding boxes with the groundtruth BB using the F_1-score and the Jaccard (J) index (also known as Intersection over Union (IOU)). The F_1-score is the harmonic mean of the precision (P) and recall (R):

$$F_1 = 2\frac{P \cdot R}{P + R}. \tag{13}$$

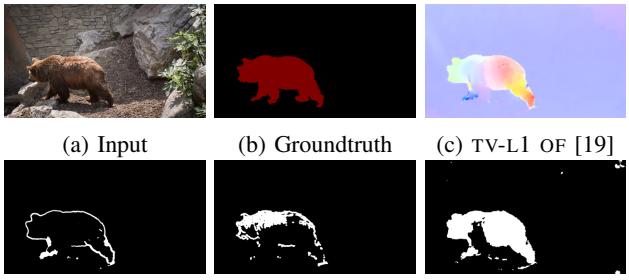
Precision is the proportion of correctly positive labeled pixels, while recall is the fraction of pixels correctly labeled, with respect to the groundtruth. They are computed as:

$$P = \frac{TP}{TP + FP} \tag{14}$$

$$R = \frac{TP}{TP + FN} \tag{15}$$

where TP denotes the true positives (pixels that are denoted as foreground in both the groundtruth and the predicted results), FP are the false positives (pixels that are labelled as foreground but are actually part of the background) and FN are the false negatives (pixels labelled as background when they really are foreground). The Jaccard index represents how well the predicted foreground aligns to the groundtruth. It is defined as:

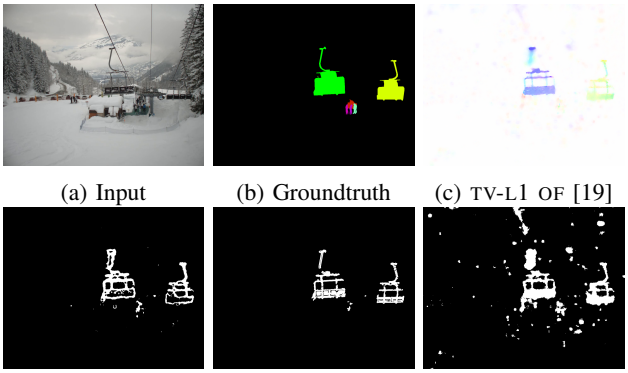
$$J = \frac{TP}{TP + FN + FP}. \tag{16}$$



(a) Input (b) Groundtruth (c) TV-L1 OF [19]

(d) Ours. $P = 0.88$, $R = 1$, $F_1 = 0.93$ (e) Result from [14]. $P = 0.88$, $R = 1$, $F_1 = 0.93$ (f) Result from [13]. $P = 0.22$, $R = 1$, $F_1 = 0.36$

Fig. 10: First frame from the *bear* sequence from [22].



(a) Input (b) Groundtruth (c) TV-L1 OF [19]

(d) Ours. $P = 0.83$, $R = 0.94$, $F_1 = 0.88$ (e) Result from [14]. $P = 0.96$, $R = 0.63$, $F_1 = 0.76$ (f) Result from [13]. $P = 0.50$, $R = 0.96$, $F_1 = 0.0.66$

Fig. 11: Frame from a sequence of our dataset.

D. Analysis of the Results

In order to compare the three methods, we used as input the same optical flow. In particular, we use the TV-L1 optical flow model provided in [19] because of its fast computation and good performance. For Patel and Parmar [13] and Tang et al. [14] we use the parameter values proposed in the corresponding paper. For our method, we tried several values for the window size, from $n = 2$ to $n = 20$, and kept the one yielding to the best f_1-score. The shift is fixed to half the window size: $\sigma = \lfloor n/2 \rfloor$.

Tables I, II and III give an excerpt of the results (Precision, Recall, F_1-score and Jaccard index) obtained by our method as well as the two other algorithms on 14 sequences of the DAVIS dataset [22], 10 sequences from out dataset and the 13 sequences of the SBI 2015 dataset [31]. The last row, of all tables, shows the averaged metrics over all the sequences of the dataset. The best values for each dataset and for each metric are depicted in bold.

We can see as in all datasets our method outperforms in terms of F-measure and Jaccard index and shows comptetitive results in terms of Precision (we obtained the best average results in the DAVIS dataset and ours, and a competitive one in the SBI dataset). This is related to the fact that our model focuses on noise removal and due to the threshold in some situations, which removes parts of the object. Nevertheless, we obtain good results in terms of precision as our model is able to detect correctly the boundaries of the objects. Regarding to the recall, it represents the true positive rate, in other words it measures how good a method is able to detect foreground pixels. One can easily reach a 100% recall value by setting all pixels to foreground, no matter the problem: of course, this solution is to be proscribed as it is not representative of the scene. This phenomenon happens with algorithm [13]: one can see from Fig. 1e that the method considers most of the pixels in the image as foreground, leading to a high Recall value (a perfect one in fact due to the BB computation).

In terms of the number of frames of each window, we can observe as the values remain very similar in our dataset and SBI 2015, but they vary much more on DAVIS dataset. This is due to the fact that the first two contain scenes which move to a similar speed, while DAVIS sequences have a higher variability on this aspect. Moreover, we are also able to observe that high values of n comes with scenes with a higher relative speed.

We present four visual examples in Figures 1, 10, 11, and 12. In the first case (Fig. 1), our method is able to capture only the moving object and completely remove noise, unlike other algorithms. Noise removal is also visible in Figure **??** and 11. One can observe a similar behaviour in Figure 10: the proposed technique performs well compared to the others in the cases of camera motion or motion produced by background objects. Finally, Figure 12 shows how the first step of the method is important. Actually, obtaining a satisfying optical flow is a clue element: the boundaries are not recovered properly when the optical flow does not clearly distinguish among the objects and background.

V. Conclusions

In this paper we proposed a model to detect the contours of moving objects in unconstrained scenes. It works by computing the optical strain, which corresponds to the deformation of the optical flow. The motion boundaries are subsequently extracted from the information brought by the optical strain. Moreover, we proposed to take the temporal evolution of the boundaries into account by adding sliding windows, thus allowing a temporally local thresholding. Our method is compared to two other models and, as it is shown in Section IV, promising results are obtained. As a matter of fact, the proposed approach outperforms in terms of precision and F-score.

Nonetheless, there are still some improvements to be considered. A first step is to extend the model by using spatial information as well. For instance, a possibility is to use some statistics to be able to decide an adaptive threshold for each area of the image. Another possible improvement is to modify the two parameters of the model, namely the shift between two sliding windows and the number of frames per window, and make them dependent on the values of the optical flow. In fact, we have observed that these parameters depend on the speed of the objects. On the other hand, as it can be

Sequence	Ours				[14]				[13]			
	P	R	F	J	P	R	F	J	P	R	F	J
bear (n=5)	**0.801**	0.96	**0.868**	**0.776**	0.563	0.993	0.667	0.558	0.232	**1.0**	0.364	0.232
blackswan (n=20)	0.477	0.968	0.633	**0.468**	**0.45**	**1.0**	0.614	0.45	0.305	**1.0**	0.467	0.305
bmx-bumps (n=18)	**0.32**	0.78	**0.397**	**0.297**	0.277	0.732	0.335	0.252	0.209	**0.838**	0.294	0.208
breakdance (n=20)	**0.403**	0.995	**0.563**	**0.402**	0.229	**1.0**	0.369	0.229	0.345	0.998	0.503	0.344
bus (n=20)	**0.876**	0.972	**0.914**	**0.848**	0.81	0.992	0.883	0.805	0.371	**1.0**	0.537	0.371
camel (n=2)	**0.935**	0.951	**0.938**	**0.89**	0.746	0.971	0.779	0.719	0.307	**0.999**	0.454	0.307
disc-jockey (n=2)	**0.886**	0.44	0.539	0.419	0.788	0.278	0.402	0.261	0.855	**0.716**	**0.751**	**0.615**
drift-chicane (n=20)	**0.135**	0.967	**0.218**	**0.134**	0.025	**1.0**	0.048	0.025	0.028	**1.0**	0.053	0.028
boat (n=15)	**0.594**	0.743	**0.654**	**0.489**	0.345	0.966	0.505	0.34	0.195	**1**	0.326	0.195
drift-turn (n=19)	0.392	0.976	0.511	0.377	**0.458**	0.932	**0.57**	**0.426**	0.259	**0.998**	0.357	0.258
elephant (n=15)	**0.942**	0.957	**0.948**	**0.902**	0.596	0.987	0.682	0.585	0.234	**0.998**	0.372	0.234
koala (n=4)	**0.826**	0.892	**0.83**	**0.748**	0.646	0.933	0.723	0.582	0.722	**0.967**	0.799	0.693
lab-coat (n=2)	**0.557**	0.662	**0.557**	0.413	0.419	0.738	0.443	0.297	**0.448**	**0.954**	0.552	0.444
lucia (n=8)	**0.824**	0.917	**0.857**	**0.757**	0.799	0.826	0.751	0.645	0.498	**0.966**	0.624	0.483
Average	**0.638**	0.804	**0.645**	**0.521**	0.443	0.828	0.48	0.36	0.401	**0.9**	0.477	0.355

TABLE I: Precision, Recall, F_1-score and Jaccard index of the results from sequences of DAVIS dataset [22]. Our model is compared with the ones proposed in [13] and [14]. The last row shows the average over all the 90 sequences from the dataset.

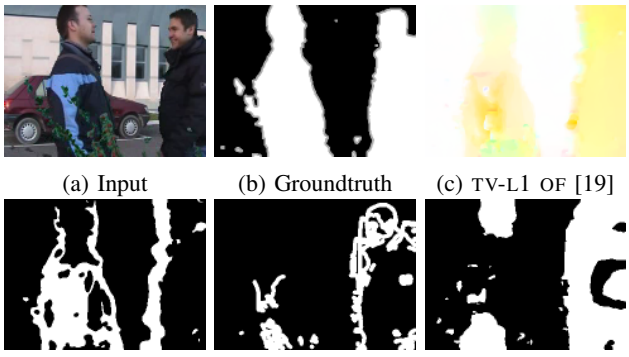
Sequence	Ours				[14]				[13]			
	P	R	F	J	P	R	F	J	P	R	F	J
Sequence 1 (n=2)	0.932	**0.744**	**0.819**	0.703	**0.987**	0.584	0.719	0.579	0.911	0.743	0.803	0.685
Sequence 2 (n=2)	0.849	0.724	**0.78**	0.641	**0.941**	0.396	0.556	0.386	0.723	**0.812**	0.762	0.62
Sequence 3 (n=2)	0.912	0.555	0.688	0.526	**0.965**	0.387	0.55	0.381	0.62	**0.81**	**0.695**	0.534
Sequence 4 (n=3)	0.742	0.593	**0.649**	0.484	**0.78**	0.508	0.594	0.426	0.576	**0.689**	0.614	0.446
Sequence 5 (n=4)	0.584	0.566	0.562	0.396	**0.809**	0.569	**0.61**	**0.456**	0.079	**0.962**	0.145	0.079
Sequence 6 (n=2)	**0.678**	0.799	**0.729**	**0.577**	0.204	**0.983**	0.333	0.203	0.153	0.932	0.26	0.151
Sequence 7 (n=3)	0.846	0.951	0.883	0.805	**0.961**	0.884	**0.916**	**0.85**	0.509	**0.96**	0.619	0.493
Sequence 8 (n=2)	0.698	0.536	**0.603**	0.437	**0.705**	0.516	0.589	0.425	0.365	**0.745**	0.457	0.306
Sequence 9 (n=2)	**0.731**	0.461	**0.521**	0.396	0.623	0.495	0.465	0.351	0.455	**0.775**	0.502	0.378
Sequence 10 (n=2)	0.869	0.303	0.445	0.289	**0.925**	0.149	0.247	0.147	0.567	**0.801**	**0.637**	0.474
Average	**0.675**	0.578	**0.586**	**0.448**	0.669	0.62	0.538	0.413	0.408	**0.851**	0.47	0.348

TABLE II: Precision, Recall, F_1-score and Jaccard index of the obtained results from some sequences our dataset. We compare our model with the ones proposed in [13] and [14]. The last row shows the average over all the 76 sequences from the dataset.

Sequence	Ours				[14]				[13]			
	P	R	F	J	P	R	F	J	P	R	F	J
Board (n=2)	0.869	**0.859**	**0.839**	**0.743**	**0.915**	0.648	0.721	0.607	0.9	0.775	0.814	0.706
CAVIAR 1 (n=2)	**0.748**	0.259	0.352	0.23	0.742	0.246	0.326	0.205	0.535	**0.587**	**0.49**	**0.344**
CAVIAR 2 (n=3)	0.61	0.479	**0.425**	**0.308**	**0.64**	0.464	0.406	0.297	0.091	**0.698**	0.141	0.079
Ca Vignal (n=2)	0.431	0.707	0.492	0.378	**0.53**	0.822	**0.582**	**0.483**	0.255	**0.925**	0.38	0.249
Candela m1.10 (n=2)	0.52	0.355	0.356	0.276	**0.638**	0.445	0.447	0.33	0.442	**0.754**	**0.457**	**0.334**
Foliage (n=2)	0.956	**0.975**	**0.958**	**0.933**	**0.967**	0.838	0.873	0.812	0.96	0.966	0.955	0.929
Hall & Monitor (n=2)	0.806	0.562	**0.605**	**0.455**	**0.817**	0.536	0.562	0.428	0.259	**0.849**	0.367	0.233
Highway I (n=2)	0.811	**0.794**	**0.773**	**0.653**	**0.82**	0.579	0.636	0.492	0.799	0.773	0.744	0.624
Highway II (n=2)	0.877	0.316	0.429	0.292	**0.904**	0.236	0.323	0.213	0.821	**0.435**	**0.478**	**0.337**
Human Body 2 (n=2)	0.895	0.636	**0.708**	**0.584**	**0.944**	0.525	0.636	0.504	**0.773**	0.662	0.668	0.536
IBM test 2 (n=6)	0.845	0.714	**0.721**	**0.6**	**0.9**	0.664	0.706	0.595	**0.735**	0.733	0.651	0.514
People & Foliage (n=2)	0.987	**0.767**	**0.841**	**0.756**	0.992	0.527	0.647	0.523	**0.99**	0.712	0.807	0.705
Snellen (n=2)	0.94	**0.945**	**0.933**	**0.889**	**0.983**	0.61	0.723	0.6	0.95	0.801	0.855	0.762
Average	0.792	0.644	**0.649**	**0.546**	**0.830**	0.549	0.584	0.468	0.654	**0.744**	0.601	0.489

TABLE III: Precision, Recall, F_1-score and Jaccard index of the obtained results on SBI 2015 dataset [31]. We compare our model with the ones proposed in [13] and [14]. The last row shows the average over all the 13 sequences from the dataset.

seen in the results, the model is threshold-dependent: it is then possible to study other threshold combinations. Finally, this model can integrate a contour-tracking model in order to establish a correspondence for the contours on different frames.



(a) Input (b) Groundtruth (c) TV-L1 OF [19]

(d) Ours. $P = 0.1$, $R = 0.65$, $F_1 = 0.79$ (e) Result from [14]. $P = 0.99$, $R = 0.43$, $F_1 = 0.60$ (f) Result from [13]. $P = 0.99$, $R = 0.53$, $F_1 = 0.69$

Fig. 12: Frame 40 from *People And Foliage* sequence of SBI 2015 dataset [31].

REFERENCES

[1] S. Coşar, G. Donatiello, V. Bogorny, C. Garate, L. O. Alvares, and F. Brémond, "Toward abnormal trajectory and event detection in video surveillance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 3, pp. 683–695, 2016.

[2] H. Zhang, R. Hu, Y. Wang, Q. Leng, and Q. Chen, "A novel method of similarity search for moving object trajectories," 2012.

[3] O. Ossama and H. M. Mokhtar, "Similarity search in moving object trajectories," in *Proceedings of the 15th International Conference on Management of Data*. Citeseer, 2009, pp. 1–6.

[4] N. Paul, A. Singh, A. Midya, P. P. Roy, and D. P. Dogra, "Moving object detection using modified temporal differencing and local fuzzy thresholding," *The Journal of Supercomputing*, vol. 73, no. 3, pp. 1120–1139, 2017.

[5] K. Abdulrahim, R. A. Salam *et al.*, "Cumulative frame differencing for urban vehicle detection," in *First International Workshop on Pattern Recognition*, vol. 10011. International Society for Optics and Photonics, 2016, p. 100110G.

[6] S. S. M. Radzi, S. N. Yaakob, Z. Kadim, and H. H. Woon, "Extraction of moving objects using frame differencing, ghost and shadow removal," in *2014 5th International Conference on Intelligent Systems, Modelling and Simulation*. IEEE, 2014, pp. 229–234.

[7] J. Cho, Y. Jung, D.-S. Kim, S. Lee, and Y. Jung, "Moving object detection based on optical flow estimation and a gaussian mixture model for advanced driver assistance systems," *Sensors*, vol. 19, no. 14, p. 3217, 2019.

[8] M. Vijayan and M. Ramasundaram, "A fast dgpso-motion saliency map based moving object detection," *Multimedia Tools and Applications*, vol. 78, no. 6, pp. 7055–7075, 2019.

[9] J. Huang, W. Zou, Z. Zhu, and J. Zhu, "An efficient optical flow based motion detection method for non-stationary scenes," in *2019 Chinese Control And Decision Conference (CCDC)*. IEEE, 2019, pp. 5272–5277.

[10] S. M. Roy and A. Ghosh, "Foreground segmentation using adaptive 3 phase background model," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[11] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, vol. 2. IEEE, 1999, pp. 246–252.

[12] S. Aslani and H. Mahdavi-Nasab, "Optical flow based moving object detection and tracking for traffic surveillance," *International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, vol. 7, no. 9, pp. 1252–1256, 2013.

[13] M. P. Patel and S. K. Parmar, "Moving object detection with moving background using optic flow," in *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*. IEEE, 2014, pp. 1–6.

[14] C. Tang, H. Hu, M. Zhang, W.-J. Wang, X.-F. Wang, F. Cao, and W. Li, "Real-time detection of moving objects in a video sequence by using data fusion algorithm," *Transactions of the Institute of Measurement and Control*, vol. 41, no. 3, pp. 793–804, 2019.

[15] K. Kale, S. Pawar, and P. Dhulekar, "Moving object tracking using optical flow and motion vector estimation," in *2015 4th international conference on reliability, infocom technologies and optimization (ICRITO)(trends and future directions)*. IEEE, 2015, pp. 1–6.

[16] Y.-H. Tsai, M.-H. Yang, and M. J. Black, "Video segmentation via object flow," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3899–3908.

[17] F. P. Gamonal, C. Ballester, G. Haro Ortega, E. Meinhardt-Llopis, and R. P. Palomares, "An analysis and speedup of the faldoi method for optical flow estimation," *Image Processing On Line. 2019; 9: 94-123.*, 2019.

[18] R. P. Palomares, E. Meinhardt-Llopis, C. Ballester, and G. Haro, "Faldoi: A new minimization strategy for large displacement variational optical flow," *Journal of Mathematical Imaging and Vision*, vol. 58, no. 1, pp. 27–46, 2017.

[19] J. Sánchez Pérez, E. Meinhardt-Llopis, and G. Facciolo, "TV-L1 optical flow estimation," *Image Processing On Line*, vol. 2013, pp. 137–150, 2013.

[20] M. Yokoyama and T. Poggio, "A contour-based moving object detection and tracking," in *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. IEEE, 2005, pp. 271–276.

[21] C. I. Patel and R. Patel, "Contour based object tracking," *International Journal of Computer and Electrical Engineering*, vol. 4, no. 4, p. 525, 2012.

[22] S. Caelles, J. Pont-Tuset, F. Perazzi, A. Montes, K.-K. Maninis, and L. Van Gool, "The 2019 davis challenge on vos: Unsupervised multi-object segmentation," *arXiv:1905.00737*, 2019.

[23] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L 1 optical flow," in *Joint pattern recognition symposium*. Springer, 2007, pp. 214–223.

[24] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.

[25] A. J. Chorin, J. E. Marsden, and J. E. Marsden, *A mathematical introduction to fluid mechanics*. Springer, 1990, vol. 3.

[26] M. A. Shreve, "Automatic macro-and micro-facial expression spotting and applications," 2013.

[27] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[28] B. K. Horn and B. G. Schunck, "Determining optical flow," in *Techniques and Applications of Image Understanding*, vol. 281. International Society for Optics and Photonics, 1981, pp. 319–331.

[29] M. Shreve, J. Brizzi, S. Fefilatyev, T. Luguev, D. Goldgof, and S. Sarkar, "Automatic expression spotting in videos," *Image and Vision Computing*, vol. 32, no. 8, pp. 476–486, 2014.

[30] S. T. Liong, J. See, R. C. W. Phan, Y.-H. Oh, A. C. Le Ngo, K. Wong, and S.-W. Tan, "Spontaneous subtle expression detection and recognition based on facial strain," *Signal Processing: Image Communication*, vol. 47, pp. 170–182, 2016.

[31] L. Maddalena and A. Petrosino, "Towards benchmarking scene background initialization," in *International conference on image analysis and processing*. Springer, 2015, pp. 469–476.

[32] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[33] J. Kittler and J. Illingworth, "Minimum error thresholding," *Pattern recognition*, vol. 19, no. 1, pp. 41–47, 1986.