A TV-L1 Optical Flow Method with Occlusion Detection

Coloma Ballester[1], Lluis Garrido[2], Vanel Lazcano[1], and Vicent Caselles[1]

[1] Dept Information and Communication Technologies, University Pompeu Fabra
[2] Dept Applied Mathematics and Analysis, University Barcelona
{coloma.ballester,vanel.lazcano,vicent.caselles}@upf.edu,
lluis.garrido@ub.edu

Abstract. In this paper we propose a variational model for joint optical flow and occlusion estimation. Our work stems from the optical flow method based on a TV-L^1 approach and incorporates information that allows to detect occlusions. This information is based on the divergence of the flow and the proposed energy favors the location of occlusions on regions where this divergence is negative. Assuming that occluded pixels are visible in the previous frame, the optical flow on non-occluded pixels is forward estimated whereas is backwards estimated on the occluded ones. We display some experiments showing that the proposed model is able to properly estimate both the optical flow and the occluded regions.

1 Introduction

The problem of estimating the motion from a sequence of images is a topic that has received a lot of attention in the computer vision community. The objective of motion estimation methods is to compute a flow field that represents the motion of points in two consecutive frames. Currently the most accurate techniques that address this problem are based on the formulation of the optical flow estimation in a variational setting. In this setting the objective is to minimize an energy function which is a weighted sum of two terms: a data term and a regularization term. The data term is usually based on the conservation of some property during motion. A common data term is based on the brightness constancy assumption, which assumes that the object illumination does not change along its motion trajectory. The regularization term allows to define the structure of the motion field and ensures that the optical flow computation is well posed.

After the initial work in [7], a lot of approaches that focus on accuracy have been developed. These works focus on the use of robust estimators [2], either in the data or smoothness terms, to be able to deal with motion discontinuities generated by occlusions. For the data term, e.g. L^2 or L^1 dissimilarity measures have been used. For the smoothness term, isotropic diffusion, image-adaptive, isotropic diffusion with non-quadratic regularizers [3], or anisotropic diffusion (image or flow-adaptive) [9] have been proposed. We refer to [4] for a detailed account of these possibilities. However, these methods may fail in occlusion areas due to forced, but unreliable, intensity matching. The problem can be further accentuated if the optical flow is smoothed across object boundaries adjacent to occlusion areas.

A. Pinz et al. (Eds.): DAGM/OAGM 2012, LNCS 7476, pp. 31–40, 2012.

A step towards taking into account occlusions was done by jointly estimating forward and backwards optical flow in [1]. The authors argue that at non-occluded pixels forward and backward flows are symmetric. Thus, the occlusion is determined by introducing into the formulation an error measure that assesses, for each pixel, the consistency between the forward and backward flows. Intensity matching is still forced at occluded pixels. Optical flow estimation and occlusion detection are decoupled. In [8], the authors propose a formulation that computes optical flow and implicitly detects occlusions, extrapolating optical flow in occluded areas. Occlusions are determined again by assessing the consistency of the forward and backward flows for each pixel. This cue is used to penalize the intensity matching accordingly. Thus, the method does not force matching at occluded pixels. The extrapolation mechanism in the occlusion areas is based on anisotropic diffusion and uses the underlying gradient to preserve optical flow discontinuities. Another joint approach for optical flow and occlusion detection was developed in [18]. This work proposes a two step updating scheme. The first step updates the flow field based only on the data and occlusion cues, given by the mismatch in the intensity value between the two images. The second step performs a flow diffusion process using a bilateral filter that locally adjusts filter strength by means of the occlusion cue.

Layered approaches [16] allow to realistically model occlusion boundaries. In order to do this one has to correctly compute the relative order of the surfaces. Performing inference over the combinatorial range of possible occlusion relationships is challenging. A recent work that explicitly models occlusions and depth ordering can be found in [12]. The authors present a method in which a visibility function is estimated for each layer. Spatial and temporal constraints are imposed to these functions in order to ensure layer consistency. These functions are used to penalize the intensity matching functions correspondingly. The results obtained are very good but the computational load to minimize the associated energy is high.

Other authors try to obtain occlusion boundary information by means of an operator directly applied to the computed motion. In [15], the authors argue that occlusion boundaries can be detected by assessing the points at which the flow changes rapidly. Discontinuities in the optical flow correspond in fact to zeros in the Laplacian fields of each motion component along the direction perpendicular to the occlusion boundary. In [11] video motion is represented as a set of particles. As the authors point out, the divergence of the motion field can be used to distinguish between different types of motion areas. Schematically, the divergence of a flow field is negative for occluded areas, positive for disoccluded, and near zero for the matched areas. Taking this into account, the authors define an intensity matching term that is weighted by a function depending on the divergence of the motion. At each iteration of the optimization procedure, the motion field is filtered, similarly to [18], with a bilateral filter that depends on the divergence of the motion. The latter idea is used in [13] in order to perform a robust median filtering of the motion. In another context, [14] analyzes the problem of estimating the motion of fluids. They use a divergence-curl regularizer to be able to deal with large concentrations of vorticity and divergence.

Our contribution in this paper is a novel joint optical flow and occlusion estimation approach. We build up from the ideas behind the layered approaches and the fact that the divergence of the field can be used to detect occlusion areas. Rather than trying to construct a set of depth ordered layers, we estimate a binary occlusion layer. Our joint optical flow and occlusion estimator is based on the work [17]. The authors present a variational discontinuity preserving formulation based on the total variation regularization of the flow and the L^1 norm of the data fidelity term. Both nonlinearities are decoupled by introducing an auxiliary variable that also represents the flow; then the energy is minimized by means of a numerical scheme based on a dual formulation of the TV energy and an efficient point-wise thresholding step.

The paper is organized as follows: Section 2 presents the model for joint optical flow and occlusion estimation, Section 3 describes its numerical implementation, Section 4 shows some qualitative, and quantitative results and Section 5 presents our conclusions.

2 A Model for Joint Computation of Optical Flow and Occlusions

Assume that two consecutive image frames $I_0, I_1 : \Omega \to \mathbb{R}$ are given. As usual, we assume that the image domain Ω is a rectangle in $\mathbb{R}^2$. Our starting point will be the model introduced in [19]. In order to compute the optical flow $\mathbf{u} = (u_1, u_2) : \Omega \to \mathbb{R}^2$ between I_0 and I_1, the authors propose to minimize the energy

$$E(\mathbf{u}, \chi) = \int_\Omega (\lambda |I_0(\mathbf{x}) - I_1(\mathbf{x} + \mathbf{u}(\mathbf{x}))| + |\nabla u_1| + |\nabla u_2|)\, d\mathbf{x}, \tag{1}$$

including robust data attachment and regularization terms (namely, the Total Variation of $\mathbf{u}$) with a relative weight given by the parameter $\lambda > 0$.

Following these ideas, we extend the previous model to jointly compute the optical flow and occlusions. Let $\chi : \Omega \to [0, 1]$ be the function modeling the occlusion mask, so that $\chi = 1$ identifies the occluded pixels, i.e. pixels that are visible in I_0 but not in I_1. Our model is based on the assumption that pixels that are not visible in frame I_1 are visible in the previous frame of I_0. Let $I_{-1} : \Omega \to \mathbb{R}$ be that frame. Thus, if $\chi(\mathbf{x}) = 0$, then we compare $I_0(\mathbf{x})$ and $I_1(\mathbf{x} + \mathbf{u}(\mathbf{x}))$. If $\chi(\mathbf{x}) = 1$, we compare $I_0(\mathbf{x})$ and $I_{-1}(\mathbf{x} - \mathbf{u}(\mathbf{x}))$. On the other hand, the occluded region given by $\chi = 1$ should be correlated with the region where $\mathrm{div}(\mathbf{u})$ is negative. Thus we propose to compute the optical flow by minimizing the energy

$$E(\mathbf{u}, \chi) = E_d(\mathbf{u}, \chi) + E_r(\mathbf{u}, \chi) + \frac{\alpha}{2} \int_\Omega \chi |\mathbf{u}|^2\, d\mathbf{x} + \beta \int_\Omega \chi \mathrm{div}(\mathbf{u})\, d\mathbf{x}, \tag{2}$$

where

$$E_d(\mathbf{u}, \chi) = \lambda \int_\Omega ((1-\chi)|I_0(\mathbf{x}) - I_1(\mathbf{x}+\mathbf{u}(\mathbf{x}))| + \chi |I_0(\mathbf{x}) - I_{-1}(\mathbf{x}-\mathbf{u}(\mathbf{x}))|)\, d\mathbf{x}, \tag{3}$$

$$E_r(\mathbf{u},\chi) = \int_\Omega g(\mathbf{x})(|\nabla u_1| + |\nabla u_2| + |\nabla \chi|)\, d\mathbf{x}, \tag{4}$$

and $\alpha \geq 0, \beta > 0$. We can choose $g(\mathbf{x}) = 1$ or $g = (1 + \gamma|\nabla \tilde{I}_0(\mathbf{x})|)^{-1}$, $\mathbf{x} \in \Omega$, where $\tilde{I}_0$ is a smoothed version of I_0 and $\gamma > 0$. We have included a term to penalize large displacements where $\chi = 1$ (with $\alpha > 0$ but small relative to λ). This is motivated by two observations. On one hand, we are assuming that the occluded background area is moving slower than the occluding foreground. On the other, since images have usually self-similarities, a pixel may have several possibilities to match. Based on this, we take $\alpha > 0$ and small (in practice we took $\alpha = 0.01$) to encourage choosing the smallest displacement.

As in [19], in order to cope with the nonlinearities of both $E_d(\mathbf{u},\chi)$ and $E_r(\mathbf{u},\chi)$ we introduce an auxiliary variable $\mathbf{v}$ representing the optical flow and we penalize its deviation form $\mathbf{u}$. Thus, we minimize the energy

$$E_\theta = E_d(\mathbf{v},\chi) + E_r(\mathbf{u},\chi) + \frac{\alpha}{2}\int_\Omega \chi|\mathbf{v}|^2\, d\mathbf{x} + \beta\int_\Omega \chi \mathrm{div}(\mathbf{u})\, d\mathbf{x} + \frac{1}{2\theta}\int_\Omega |\mathbf{u}-\mathbf{v}|^2, \tag{5}$$

depending on the three variables $(\mathbf{u},\mathbf{v},\chi)$, where $\theta > 0$. This energy can be minimized by alternatively fixing two variables and minimizing with respect to the third one.

To minimize (5) with respect to $\mathbf{v}$ we linearize each expression $|I_0(\mathbf{x}) - I_i(\mathbf{x} + \epsilon_i \mathbf{u}(\mathbf{x}))|$, $i = -1, 1$ ($\epsilon_{-1} = -1, \epsilon_1 = 1$), around a given vector field $\mathbf{u}_0$ and define the residual

$$\rho_i(\mathbf{v}) := I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0(\mathbf{x})) + \epsilon_i \nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0(\mathbf{x})) \cdot (\mathbf{v}(\mathbf{x}) - \mathbf{u}_0(\mathbf{x})) - I_0(\mathbf{x}). \tag{6}$$

This procedure is applied by iteratively minimizing the energy

$$\widetilde{E}_\theta = \widetilde{E}_d(\mathbf{v},\chi) + E_r(\mathbf{u},\chi) + \frac{\alpha}{2}\int_\Omega \chi|\mathbf{v}|^2\, d\mathbf{x} + \beta\int_\Omega \chi \mathrm{div}(\mathbf{u})\, d\mathbf{x} + \frac{1}{2\theta}\int_\Omega |\mathbf{u}-\mathbf{v}|^2, \tag{7}$$

where

$$\widetilde{E}_d(\mathbf{v},\chi) = \lambda \int_\Omega ((1-\chi)|\rho_1(\mathbf{v})| + \chi|\rho_{-1}(\mathbf{v})|)\, d\mathbf{x}.$$

To minimize $\widetilde{E}_\theta$ we alternate between the minimization with respect to each variable keeping the other two fixed. After iteration of these steps, we proceed to redefine $\rho_i(\mathbf{v})$ (see Algorithm 1). The minimization of $\widetilde{E}_\theta$ with respect to $\mathbf{u}$ is done using Chambolle's algorithm [5].

Proposition 1. *The minimum of $\widetilde{E}_\theta$ with respect to $\mathbf{u} = (u_1, u_2)$ is given by*

$$u_i = v_i + \theta \mathrm{div}(g\xi_i) + \theta\beta\frac{\partial \chi}{\partial x_i}, \quad i = 1, 2, \tag{8}$$

ξ_1 and ξ_2 are computed using the following iterative scheme

$$\xi_i^{k+1} = \frac{\xi_i^k + \frac{\tau_u}{\theta} g\nabla(v_i + \theta \mathrm{div}(g\xi_i^k) + \theta\beta\frac{\partial \chi}{\partial x_i})}{1 + \frac{\tau_u}{\theta}|g\nabla(v_i + \theta \mathrm{div}(g\xi_i^k) + \theta\beta\frac{\partial \chi}{\partial x_i})|}, \quad k = 0, 1, 2, ... \tag{9}$$

where $\xi_i^0 = 0$ and $\tau_u \leq 1/8$.

As in [19], we have:

Proposition 2. *Assume that* $\chi : \Omega \to \{0, 1\}$. *The minimum of* $\widetilde{E}_\theta$ *with respect to* $\mathbf{v} = (v_1, v_2)$ *is*

$$\mathbf{v} = \begin{cases} \alpha_i \mathbf{u} - \mu_i \epsilon_i \nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0) & \text{if} \quad \Lambda_i(\mathbf{u}) > \mu_i |\nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0)|^2 \\ \alpha_i \mathbf{u} + \mu_i \epsilon_i \nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0) & \text{if} \quad \Lambda_i(\mathbf{u}) < -\mu_i |\nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0)|^2 \\ \mathbf{u} - \epsilon_i \rho_i(\mathbf{u}) \frac{\nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0)}{|\nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0)|^2} & \text{if} \quad |\Lambda_i(\mathbf{u})| \leq \mu_i |\nabla I_i(\mathbf{x} + \epsilon_i \mathbf{u}_0)|^2, \end{cases} \tag{10}$$

where $i = 1$, $\epsilon_1 = 1$, $\alpha_1 = 1$, $\mu_1 = \lambda\theta$, $\Lambda_1(\mathbf{u}) = \rho_1(\mathbf{u})$ *when* $\chi = 0$, *and* $i = -1$, $\epsilon_{-1} = -1$, $\alpha_{-1} = \frac{1}{1+\alpha\theta}$, $\mu_{-1} = \frac{\lambda\theta}{1+\alpha\theta}$, $\Lambda_{-1}(\mathbf{u}) = \rho_{-1}(\mathbf{u}) + \frac{\alpha\theta}{1+\alpha\theta}\mathbf{u} \cdot \nabla I_{-1}(\mathbf{x} + \epsilon_i \mathbf{u}_0)$ *when* $\chi = 1$. *Notice that we omitted the arguments* $\mathbf{x}$ *in* $\mathbf{u}, \mathbf{u}_0$.

Having computed $\mathbf{v}$, let $F = \lambda(|\rho_{-1}(\mathbf{v})| - |\rho_1(\mathbf{v})|)$ and $G = \frac{\alpha}{2}|\mathbf{v}|^2$. As a consequence of [6], we have:

Proposition 3. *Let* $0 < \tau_\eta \tau_\chi < 1/8$. *Given* $\mathbf{u}, \mathbf{v}$, *the minimum* $\bar{\chi}$ *of* $\widetilde{E}_\theta$ *with respect to* χ *can be obtained by the following primal-dual algorithm*

$$\begin{aligned} \eta^{n+1} &= P_B(\eta^n + \tau_\eta \; g \; \nabla \chi^n) \\ \chi^{n+1} &= P_{[0,1]}\left(\chi^n + \tau_\chi \left(\mathrm{div}(g\eta^{n+1}) - \beta \; \mathrm{div}\mathbf{u} - F - G\right)\right), \end{aligned} \tag{11}$$

where $P_B(\eta)$ *denotes the projection of* η *on the unit ball of* $\mathbb{R}^2$ *and* $P_{[0,1]}(r) = \max(\min(r, 1), 0)$, $r \in \mathbb{R}$.

Notice that, by the co-area formula, the level sets of χ are also minimizers of $\widetilde{E}_\theta$ ($\mathbf{u}, \mathbf{v}$ being fixed). Thus, before going to next minimization of $\widetilde{E}_\theta$ with respect to $\mathbf{u}$, we redefine $\chi(\mathbf{x}) = T_\delta(\bar{\chi}(\mathbf{x}))$, where $T_\delta(r) = 1$ (resp. 0) if $r \geq \delta$ (resp. $< \delta$). A different relaxation that also produces good results is obtained by replacing $1 - \chi$ and χ by $(1 - \chi)^2$ and χ^2, respectively, in all terms of (7) but E_r.

3 Numerical Scheme

The numerical algorithm is summarized in Algorithm 1. The inputs are three consecutive frames I_{-1}, I_0, I_1 of a video sequence and the outputs are the flow field $\mathbf{u}$ and the occlusion layer χ.

The minimization of (2) is embedded into a coarse-to-fine multi-level approach in order to be able to deal with large flow fields. Our code is based on the implementation of [10]. We have also added some of the numerical details of [17]. Image gradient is computed using the filter stencils as proposed in [17]. Warping is performed using bicubic interpolation. Optionally, one can use a structure-texture decomposition of the images, performing the optical flow estimation on the texture part. The use of the texture images allows to improve results for real images, whereas it may degrade performance (with respect to using the original images) on synthetic ones.

Input : Three consecutive frames I_{-1}, I_0, I_1 of a video sequence
Output: Flow field $\mathbf{u}$ and occlusion layer χ for I_0

Compute down-scaled images I^s_{-1}, I^s_0, I^s_1 for $s = 1, \ldots, N_{\text{scales}}$;
Initialize $\mathbf{u}^{N_{\text{scales}}} = \mathbf{v}^{N_{\text{scales}}} = 0$, and $\chi^{N_{\text{scales}}} = 0$;
for $s \leftarrow N_{\text{scales}}$ **to** 1 **do**
  for $w \leftarrow 1$ **to** N_{warps} **do**
    Compute $I^s_i(\mathbf{x} + \epsilon_i \mathbf{u}_0(\mathbf{x}))$, $\nabla I^s_i(\mathbf{x} + \epsilon_i \mathbf{u}_0(\mathbf{x}))$, and ρ_i using (6), $i = -1, 1$;
    $n \leftarrow 0$;
    while $n <$ outer_iterations **do**
      Compute $\mathbf{v}^s$ using (10) (Proposition 2);
      for $k \leftarrow 1$ **to** inner_iterations_u **do**
        Solve for $\xi_i^{k+1,s}$, $i \in \{1, 2\}$, using the fixed point iteration (9);
      end
      Compute $\mathbf{u}^s$ using (8) (Proposition 1);
      for $m \leftarrow 1$ **to** inner_iterations_χ **do**
        Solve for χ^{m+1} using the primal-dual algorithm (11);
      end
    end
  end
  If $s > 1$ **then** scale-up $\mathbf{u}^s, \mathbf{v}^s, \chi^s$ to $\mathbf{u}^{s-1}, \mathbf{v}^{s-1}, \chi^{s-1}$;
end
$u = u^1$ and $\chi = T_\mu(\chi^1)$

Algorithm 1. Algorithm for joint optical flow and occlusion computation

4 Results

In this section we display some experiments done with our occlusion based optical flow algorithm.

Qualitative Results. The first row of Fig. 1 shows three consecutive frames of the Urban2 sequence, publicly available at Middlebury database. They correspond to I_{-1} (frame 9), I_0 (frame 10) and I_1 (frame 11). In this sequence the camera moves to the right. The apparent movement of the buildings produces occlusions between them.

Second row of Fig. 1 shows the optical flow obtained with our algorithm: the optical flow $\mathbf{u}$ is shown on the left image using the color coding scheme of the gimp color palette, as in [10]. In the middle, the motion compensated image is shown. This image is generated using $\mathbf{u}$ and backwards compensating from I_1 (resp. I_{-1}) if the corresponding pixel is not occluded (resp. is occluded). Finally, the normalized (to the range $[0, 255]$) absolute value of the difference between I_0 and the motion compensated image is shown on the right.

In Fig. 2 the occlusion layers χ are shown in red superimposed on the frame I_0 of the sequence. On the left (resp. right), the occlusion layer χ with $\alpha > 0$ (resp. $\alpha = 0$) is shown. As it can be seen, our method is able to properly detect the regions of I_0 that get occluded at frame I_1 due to the apparent movement of the buildings.

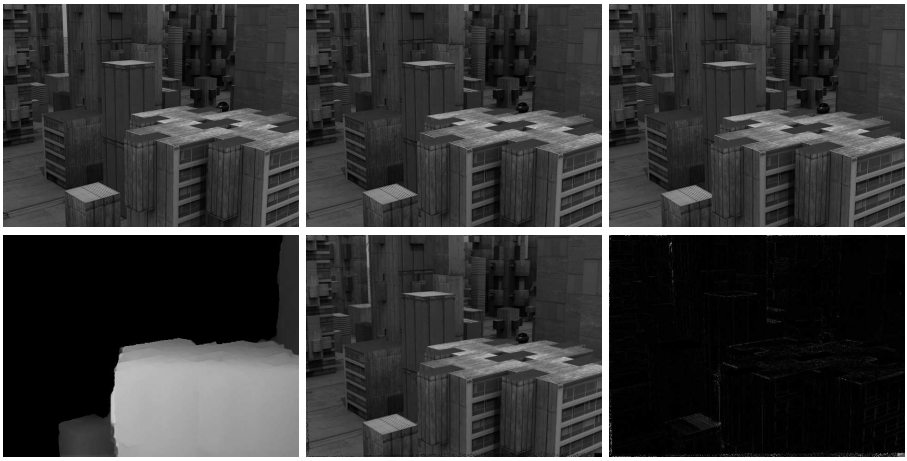

Fig. 1. *First row*: three consecutive frames I_{-1}, I_0, I_1 of the Urban2 Middlebury video sequence. *Second row*: the optical flow $\mathbf{u}$, the backwards motion compensated image using I_1 and I_{-1}, and the normalized absolute value of the difference between the motion compensated image and the original image in the *first row*.

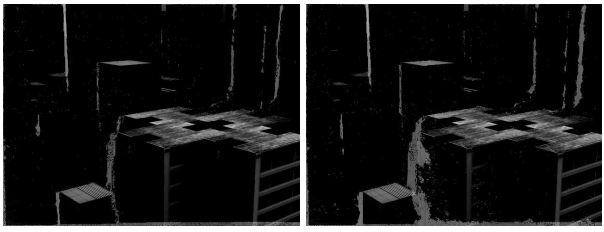

Fig. 2. *Left*: occlusion layer χ (*red*) associated to $\alpha > 0$ superimposed on I_0. *Right*: occlusion layer χ associated to $\alpha = 0$ superimposed on I_0

Fig. 3 shows the interpolation error obtained for the Urban image of the Middlebury evaluation database using our optical flow (left) and the flow of [17] (right). The images shown here have been directly obtained from the Middlebury web evaluation results, with hidden ground-truth flow. As it can be seen, our method behaves better in the occluded regions.

Our next experiment deals with the Backyard sequence, a natural sequence taken with a high-speed camera. The first row of Fig. 4 shows three consecutive frames of this sequence. They correspond to I_{-1} (frame 9), I_0 (frame 10) and I_1 (frame 11). In this sequence the camera pans to the right, in addition the kids move and the ball on the left is falling down. The apparent movement produces different kinds of occlusions.

The second row of Fig. 4 shows the results obtained with our occlusion based optical flow algorithm: the computed optical flow $\mathbf{u}$, the occlusion layer (in red) superimposed on I_0, and the normalized absolute value of the difference between

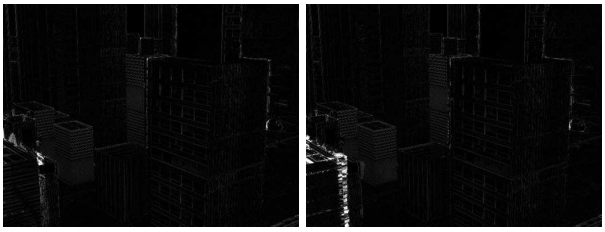

Fig. 3. Interpolation error results obtained from the Middlebury evaluation database for the Urban sequence. *Left*, result of our method. *Right*, result of the method in [17].

Fig. 4. *First row*: three consecutive frames of the Backyard sequence. *Second row*: *from left to right*: the optical flow $\mathbf{u}$, the occlusion layer (*red*) superimposed on I_0, and the normalized absolute difference error between the motion compensated image and I_0.

the motion compensated image and I_0. As can be seen again, our method is able to properly detect occluded pixels around the kids and the ball.

Quantitative Results on the Middlebury test-set. We have used our method on those sequences with at least three frames, using that of [17] if only two frames are available. Table 1 shows the end point error (EPE) and the average angular error (AAE) of the estimated flow fields. These two values are computed using the occlusion layer χ, that is, they are computed only at non-occluded pixels. For this table, we have run the algorithm with the same parameter set: $\lambda = 0.25, \theta = 0.30, \beta = 1, \alpha = 0$, $g = (1 + \gamma|\nabla \tilde{I}_0(\mathbf{x})|)^{-1}$, $\mathbf{x} \in \Omega$, $\gamma = 0.05$, 5 pyramid scales and 10 warpings at each scale. As can be seen in the table, our metod is competitive with the results reported by [17].

The same parameter set has been used to test our method against the Middlebury evaluation test set, see http://vision.middlebury.edu/flow, as well as Fig. 5. At the time of writing this paper (June 2012), our method, called Occlusion-TV-L1 in the Middlebury database, is located at position 24 with respect to the end-point-error, the original TV-L1-improved method being located at position 34.

Table 1. Evaluation results on images of the Middlebury dataset with public ground truth flow. The table shows the end point error (EPE) and the average angular error (AAE) of the estimated flow fields using the same set of parameters for our method and the method presented in [17].

	Grove2	Grove3	Hydra	Rubber	Urban2	Urban3
EPE-TV-L1 improved [17]	0.154	0.665	0.147	0.092	0.319	0.630
Our EPE-Occlusion based	0.121	0.547	0.162	0.093	0.311	0.382
Our AAE-Occlusion based	1.802^o	5.400^o	1.977^o	2.895^o	2.497^o	3.382^o

Average endpoint error	avg. rank	Army (Hidden texture) GT	Army im0	Army im1	Mequon (Hidden texture) GT	Mequon im0	Mequon im1	Schefflera (Hidden texture) GT	Schefflera im0	Schefflera im1	Wooden (Hidden texture) GT	Wooden im0	Wooden im1	Grove (Synthetic) GT	Grove im0	Grove im1	Urban (Synthetic) GT	Urban im0	Urban im1	Yosemite (Synthetic) GT	Yosemite im0	Yosemite im1	Teddy (Stereo) GT	Teddy im0	Teddy im1
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
SimpleFlow [52]	22.9	0.09 14	0.24 13	0.08 21	0.24 32	0.78 29	0.20 35	0.43 29	0.96 30	0.21 30	0.16 15	0.77 12	0.09 11	0.71 13	1.04 14	0.55 18	1.47 55	1.59 37	0.76 49	0.13 22	0.12 6	0.22 26	0.50 12	1.04 14	0.72 13
Occlusion-TV-L1 [68]	24.0	0.09 14	0.26 23	0.07 7	0.22 19	0.74 25	0.18 20	0.51 37	1.15 39	0.21 30	0.18 24	0.91 25	0.10 24	0.87 29	1.25 28	0.72 28	0.47 18	1.38 23	0.36 28	0.10 4	0.12 6	0.11 2	0.83 41	1.78 44	0.96 38
Adaptive [20]	25.9	0.09 14	0.26 23	**0.06** 1	0.23 29	0.78 29	0.18 20	0.54 39	1.19 43	0.21 30	0.18 24	0.91 25	0.10 24	0.88 32	1.25 28	0.73 31	0.50 23	1.28 20	0.31 11	0.14 28	0.16 46	0.22 26	0.65 27	1.37 26	0.79 23
DPOF [18]	27.1	0.12 41	0.33 41	0.08 21	0.26 36	0.80 33	0.20 35	0.24 4	0.49 4	0.20 26	0.19 29	0.83 17	0.13 34	0.66 8	0.98 9	0.40 4	1.11 47	1.41 28	0.57 42	0.25 67	0.14 29	0.55 67	0.51 15	1.02 12	0.54 2
Adapt-Window [34]	27.2	0.10 24	0.24 13	0.09 30	0.19 7	0.59 4	0.15 11	0.27 7	0.64 9	0.17 6	0.18 24	0.82 16	0.11 27	0.74 16	1.07 17	0.56 19	1.78 64	1.73 44	0.95 59	0.22 61	0.16 46	0.45 64	0.70 31	1.28 22	0.88 31
ACK-Prior [27]	28.0	0.11 31	0.25 17	0.09 30	0.18 3	0.59 4	0.13 3	0.27 7	0.64 9	0.16 4	0.15 8	0.78 13	0.09 11	0.82 23	1.14 21	0.71 27	1.90 66	1.90 51	0.99 62	0.23 65	0.17 52	0.49 66	0.77 37	1.44 28	0.91 33
Complementary OF [21]	28.5	0.11 31	0.28 30	0.10 39	0.18 3	0.63 10	0.12 2	0.31 19	0.75 19	0.18 11	0.19 29	0.97 32	0.12 31	0.97 43	1.31 39	1.00 48	1.78 64	1.73 44	0.87 57	0.11 9	0.12 6	0.22 26	0.68 28	1.48 29	0.95 36
ComplOF-FED-GPU [36]	29.6	0.11 31	0.29 33	0.10 39	0.21 16	0.78 29	0.14 7	0.32 21	0.79 23	0.17 6	0.19 29	0.99 33	0.11 27	0.89 33	1.29 34	0.73 31	1.25 49	1.74 46	0.64 46	0.14 28	0.13 17	0.30 51	0.64 25	1.50 31	0.83 26
Classic++ [32]	30.0	0.09 14	0.25 17	0.07 7	0.23 29	0.78 29	0.19 28	0.43 29	1.00 32	0.22 33	0.20 33	1.11 38	0.10 24	0.87 29	1.30 36	0.66 25	0.47 18	1.62 38	0.33 17	0.17 48	0.14 29	0.32 56	0.79 40	1.64 38	0.92 34
Aniso. Huber-L1 [22]	30.3	0.10 24	0.28 30	0.08 21	0.31 43	0.88 38	0.28 46	0.56 42	1.13 37	0.29 46	0.20 33	0.92 29	0.13 34	0.84 26	1.20 24	0.70 26	0.39 5	1.23 18	0.28 5	0.17 48	0.15 39	0.27 41	0.64 25	1.36 25	0.79 23
TriangleFlow [30]	32.8	0.11 31	0.29 33	0.09 30	0.26 36	0.95 42	0.17 17	0.47 36	1.07 35	0.18 11	0.16 15	0.87 23	0.09 11	1.07 50	1.47 55	1.10 51	0.87 41	1.39 25	0.57 42	0.15 36	0.19 62	0.23 30	0.63 24	1.33 23	0.84 27
TV-L1-improved [17]	34.6	0.09 14	0.26 23	0.07 7	0.20 13	0.71 18	0.16 14	0.53 38	1.18 42	0.22 33	0.21 37	1.24 43	0.11 27	0.90 34	1.31 39	0.72 28	1.51 57	1.93 53	0.84 53	0.18 52	0.17 52	0.31 54	0.73 33	1.62 37	0.87 30
LocallyOriented [55]	35.5	0.12 41	0.35 45	0.08 21	0.33 46	1.01 46	0.25 44	0.61 49	1.30 51	0.28 43	0.18 24	0.80 14	0.13 34	0.93 39	1.29 34	0.79 37	0.98 44	1.48 33	0.56 41	0.12 15	0.14 29	0.21 24	0.73 33	1.48 29	0.95 36

Fig. 5. Average end point error on Middlebury flow benchmark (June 22nd 2012)

Our method needs about 538 seconds to compute the optical flow for the "Urban" evaluation sequence using a single CPU. It should be noted that the Teddy sequence has only two images. Thus, for this sequence we used a two frame TV-L1 estimator [17] without occlusion detection using the previously specified parameters for λ and θ, number of pyramid scales, and number of warpings.

5 Conclusions

In this work we presented a variational model for joint optical flow and occlusion estimation. Our work stems from the optical flow method presented in [17] and incorporates information that allows to detect occlusions. This information is based on the divergence of the flow, and the proposed energy contains a term that favors the location of occlusions on regions where this divergence is negative. Based on the mild assumption that, when computing the flow at time t, occluded pixels are visible in the previous frame, the optical flow on non-occluded pixels is forward estimated whereas is backward estimated on the occluded ones.

Our numerical experiments show that the proposed approach is able to properly estimate the optical flow and the occluded regions. It also improves the original method [17].

References

1. Alvarez, L., Deriche, R., Papadopoulo, T., Sanchez, J.: Symmetrical dense optical flow estimation with occlusions detection. International Journal of Computer Vision 75(3), 371–385 (2007)

2. Black, M., Anandan, P.: The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. Computer Vision and Image Understanding 63(1), 75–104 (1996)
3. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High Accuracy Optical Flow Estimation Based on a Theory for Warping. In: Pajdla, T., Matas, J. (eds.) ECCV 2004, Part IV. LNCS, vol. 3024, pp. 25–36. Springer, Heidelberg (2004)
4. Bruhn, A.: Variational optic flow computation: Accurate modeling and efficient numerics. Ph.D. thesis, Department of Mathematics and Computer Science, Saarland University (2006)
5. Chambolle, A.: An algorithm for total variation minimization and applications. Mathematical Imaging and Vision 20(1), 89–97 (2004)
6. Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. Journal of Mathematical Imaging and Vision 40(1), 120–145 (2011)
7. Horn, B., Schunk, B.: Determining optical flow. Artificial Intelligence 20 (1981)
8. Ince, S., Konrad, J.: Occlusion-aware optical flow estimation. IEEE Trans. Image Processing 17(8), 1443–1451 (2008)
9. Nagel, H.H., Enkelmann, W.: An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. IEEE Transactions on Pattern Analysis and Machine Intelligence 8(5), 565–593 (1986)
10. Sánchez, J., Meinhardt-Llopis, E., Facciolo, G.: TV-L^1 optical flow estimation. Image Processing Online (January 2012), `http://www.ipol.im`
11. Sand, P., Teller, S.: Particle video: Long-range motion estimation using point trajectories. International Journal of Computer Vision 80(1), 72–91 (2008)
12. Sun, D., Sudderth, E.B., Black, M.J.: Layered image motion with explicit occlusions, temporal consistency, and depth ordering. In: Advances in Neural Information Processing Systems, pp. 2226–2234 (2010)
13. Sun, D., Roth, S., Black, M.J.: Secrets of optical flow estimation and their principles. In: CVPR (2010)
14. Corpetti, T., Mémin, E., Pérez, P.: Dense estimation of fluid flows. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(3), 365–380 (2002)
15. Thompson, W.B., Mutch, K.M., Berzins, V.A.: Dynamic occlusion analysis in optical flow fields. IEEE Transactions on Pattern Analysis and Machine Intelligence 7(4), 374–383 (1985)
16. Wang, J.Y.A., Adelson, E.H.: Representing moving images with layers. IEEE Transactions on Image Processing 3(5), 625–638 (1994)
17. Wedel, A., Pock, T., Zach, C., Bischof, H., Cremers, D.: An Improved Algorithm for TV-L1 Optical Flow. In: Cremers, D., Rosenhahn, B., Yuille, A.L., Schmidt, F.R. (eds.) Visual Motion Analysis 2008. LNCS, vol. 5604, pp. 23–45. Springer, Heidelberg (2009)
18. Xiao, J., Cheng, H., Sawhney, H.S., Rao, C., Isnardi, M.: Bilateral Filtering-Based Optical Flow Estimation with Occlusion Detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 211–224. Springer, Heidelberg (2006)
19. Zach, C., Pock, T., Bischof, H.: A Duality Based Approach for Realtime TV-L^1 Optical Flow. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 214–223. Springer, Heidelberg (2007)