Published in Image Processing On Line on 2016–11–07.
Submitted on 2016–02–19, accepted on 2016–10–04.
ISSN 2105–1232 © 2016 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2016.172

Robust Discontinuity Preserving Optical Flow Methods

Nelson Monzón, Agustín Salgado and Javier Sánchez

CTIM, University of Las Palmas de Gran Canaria, Spain
(nmonzon@ctim.es, asalgado@dis.ulpgc.es, jsanchez@dis.ulpgc.es)

Abstract

In this work, we present an implementation of discontinuity-preserving strategies in TV-L^1 optical flow methods. These are based on exponential functions that mitigate the regularization at image edges, which usually provide precise flow boundaries. Nevertheless, if the smoothing is not well controlled, it may produce instabilities in the computed motion fields. We present an algorithm that allows three regularization strategies: the first one uses an exponential function together with a TV process; the second one combines this strategy with a small constant that ensures a minimum isotropic smoothing; the third one is a fully automatic approach that adapts the diffusion depending on the histogram of the image gradients. The last two alternatives are aimed at reducing the effect of instabilities. In the experiments, we observe that the pure exponential function is highly unstable while the other strategies preserve accurate motion contours for a large range of parameters.

Source Code

The source code, its documentation and the online demo are accessible at the IPOL web page of this article[1]. In this page an implementation is available for download.

Keywords: optical flow; motion estimation; regularization strategies; discontinuity-preserving

1 Introduction

Optical flow is a key problem in computer vision that provides consistent motion information from a video sequence. Traditionally, variational methods have been the most accurate and widely used techniques. Their solutions are posed as a minimization problem of an energy functional expressed as a weighted sum of a data and a smoothness terms. The first puts into correspondence the information between the pixels of consecutive images while the second involves a diffusion process for creating more continuous vector fields.

The smoothness term ensures that our solution is unique under the premise that neighboring pixels present a similar motion but, on the other hand, is critical to preserve correct motion boundaries. This topic is important in optical flow studies due to the fact that it is not easy to introduce a simple

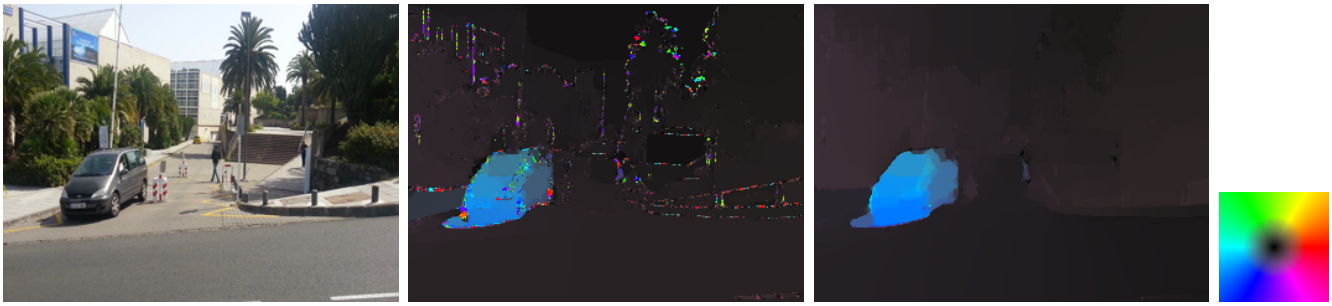

Figure 1: From left to right: The original image; instabilities over the computed flow field due to a wrong parameter; well-defined motion contours using our automatic strategy; the color scheme used for motion representation.

diffusion mechanism to overcome this problem. Many strategies have been proposed to cope with motion discontinuities since the method of Horn and Schunck [8]. This technique hardly preserves flow boundaries as can be seen in the IPOL article by Meinhardt-Llopis et al. [9].

In 1986, Nagel and Enkelman [13] introduced a diffusion tensor that uses the information of the image gradient for regularization. It enables anisotropic diffusion along object contours and isotropic smoothing in homogeneous regions. This tensor was used in Alvarez et al. [2] with a linear scale-space formulation for dealing with large displacement. Black and Anandan [4, 5] introduced robust functionals in the regularization term, which produce piecewise continuous motion fields and were more insensitive to image noise than previous approaches. The generalization in the use of L^1 functionals was proposed in several works, such as in Brox et al. [6] and Zach et al. [18]. These two methods have been analyzed in the IPOL articles by Sánchez et al., [15] and [16], respectively.

These strategies produce piecewise smooth motion fields but, since they do not use any image information in the regularization process, their flow edges do not usually coincide with the object boundaries. This problem can be solved with a decreasing function that uses the gradient information to stop the regularization at image contours. This idea originally comes from Alvarez et al. [1] and is often used in the recent literature [17, 14]. It is a simple solution and, probably, one of the best current alternatives. Nevertheless, it also presents a strong dependency with respect to the parameter of the decreasing function. As we have seen in Monzón et al. [11], if the parameter is underestimated, the result does not improve the basic TV approaches. On the other hand, when it is overestimated, the decreasing function may cancel the regularization and the problem becomes ill-posed.

We see an example of this situation in Figure 1. The second image shows a result where many instabilities appear in the form of blobs at the object limits. In this solution, a wrong setting has cancelled the smoothing in regions where the image gradients are high, providing poor results. The third image depicts a good result because of a correct selection of the discontinuity parameter.

Monzón et al. [10] explored two mechanisms to overcome these drawbacks: on the one hand, they proposed a simple approach that ensures a minimum isotropic smoothing when using decreasing functions; on the other hand, they use a strategy that looks for the best parameter configuration preserving motion contours while avoiding instabilities.

Here, we present an implementation of this method, including these two alternatives. In the experiments, we observe the severe instabilities that the exponential function produces if the discontinuity parameter is not correctly chosen and the benefits of using these two strategies. We analyze their behavior with respect to the smoothness parameter.

[1] https://doi.org/10.5201/ipol.2016.172

2 Robust Discontinuity-Preserving Optical Flow Methods

Let $I_1^c, I_2^c : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^c$ be an image sequence, with $\mathbf{x} = (x, y)^T \in \Omega$, $\{I^c\}_{c=1,\dots,C}$ and C the number of channels. The optical flow is defined as a dense mapping, $\mathbf{w}(\mathbf{x}) = (u(\mathbf{x}), v(\mathbf{x}))^T$, between two consecutive images, where $u(\mathbf{x})$ and $v(\mathbf{x})$ are the horizontal and vertical displacements, respectively.

We assume that the brightness and gradient constancy assumptions [6, 16] are also fulfilled in a multi-channel scheme [12]. Then, according to this notation, our energy functional reads as

$$\begin{aligned} E(\mathbf{w}) = &\int_\Omega \psi\left(\sum_{c=1}^{C}\left(I_2^c(\mathbf{x}+\mathbf{w}) - I_1^c(\mathbf{x})\right)^2\right) + \gamma \int_\Omega \psi\left(\sum_{c=1}^{C}\left|\nabla I_2^c(\mathbf{x}+\mathbf{w}) - \nabla I_1^c(\mathbf{x})\right|^2\right) \\ &+ \alpha \int_\Omega \psi\left(\Phi(\nabla I_1)\cdot\left(|\nabla u|^2 + |\nabla v|^2\right)\right) \mathbf{dx}, \end{aligned} \tag{1}$$

with $\psi\left(s^2\right) = \sqrt{s^2+\epsilon^2}$ and $\epsilon := 0.001$ as a prefixed small constant to ensure that ψ is strictly convex. Parameters γ and α weigh the gradient and smoothness terms, respectively. $\Phi(\nabla I_1)$ is a smoothness function that provides three regularization strategies. We study them in Section 4.

One finds the minimum of (1) by solving the associated Euler-Lagrange equations that yield a system of reaction-diffusion PDEs,

$$\begin{aligned} 0 =& \psi'_D \cdot \left(\sum_{c=1}^{C}\left(I_2^c(\mathbf{x}+\mathbf{w}) - I_1^c(\mathbf{x})\right)\cdot I_{2,x}^c(\mathbf{x}+\mathbf{w})\right) \\ &+ \gamma\, \psi'_G \cdot \left(\sum_{c=1}^{C}\left(\left(I_{2,x}^c(\mathbf{x}+\mathbf{w}) - I_{1,x}^c(\mathbf{x})\right)\cdot I_{2,xx}^c(\mathbf{x}+\mathbf{w}) + \left(I_{2,y}^c(\mathbf{x}+\mathbf{w}) - I_{1,y}^c(\mathbf{x})\right)\cdot I_{2,xy}^c(\mathbf{x}+\mathbf{w})\right)\right) \\ &- \alpha\, \mathrm{div}\left(\psi'_S \cdot \nabla u\right), \end{aligned}$$

$$\begin{aligned} 0 =& \psi'_D \cdot \left(\sum_{c=1}^{C}\left(I_2^c(\mathbf{x}+\mathbf{w}) - I_1^c(\mathbf{x})\right)\cdot I_{2,y}^c(\mathbf{x}+\mathbf{w})\right) \\ &+ \gamma\, \psi'_G \cdot \left(\sum_{c=1}^{C}\left(\left(I_{2,x}^c(\mathbf{x}+\mathbf{w}) - I_{1,x}^c(\mathbf{x})\right)\cdot I_{2,xy}^c(\mathbf{x}+\mathbf{w}) + \left(I_{2,y}^c(\mathbf{x}+\mathbf{w}) - I_{1,y}^c(\mathbf{x})\right)\cdot I_{2,yy}^c(\mathbf{x}+\mathbf{w})\right)\right) \\ &- \alpha\, \mathrm{div}(\psi'_S \cdot \nabla v), \end{aligned} \tag{2}$$

with $\psi'(s^2) = \frac{1}{2\sqrt{s^2+\varepsilon^2}}$. Notice that the influence of the data term is proportional to the number of channels. We compensate this situation by adapting the smoothness parameter as $\alpha = \alpha' \cdot C$, where α' is an input parameter. In order to simplify these equations, we use the notation

$$\begin{aligned} \psi'_D :=& \psi'\left(\sum_{c=1}^{C}\left(I_2^c(\mathbf{x}+\mathbf{w}) - I_1^c(\mathbf{x})\right)^2\right), \\ \psi'_G :=& \psi'\left(\sum_{c=1}^{C}\left|\nabla I_2^c(\mathbf{x}+\mathbf{w}) - \nabla I_1^c(\mathbf{x})\right|^2\right), \\ \psi'_S :=& \Phi(\nabla I_1)\cdot\psi'\left(\Phi(\nabla I_1)\cdot\left(|\nabla u|^2 + |\nabla v|^2\right)\right). \end{aligned} \tag{3}$$

The above equations are nonlinear because of the argument $\mathbf{w}$ and function ψ'; so, in order to linearize the equations, we follow the same strategy of [16], enclosing our numerical scheme in two fixed point iterations. We introduce a first index, n, to remove the nonlinearity in $\mathbf{w}$, using first order Taylor expansions

$$
\begin{aligned}
I(\mathbf{x}+\mathbf{w}^{n+1}) \approx & I(\mathbf{x}+\mathbf{w}^{n}) + I_x(\mathbf{x}+\mathbf{w}^{n})du^n + I_y(\mathbf{x}+\mathbf{w}^{n})dv^n \\
I_x(\mathbf{x}+\mathbf{w}^{n+1}) \approx & I_x(\mathbf{x}+\mathbf{w}^{n}) + I_{xx}(\mathbf{x}+\mathbf{w}^{n})du^n + I_{xy}(\mathbf{x}+\mathbf{w}^{n})dv^n \\
I_y(\mathbf{x}+\mathbf{w}^{n+1}) \approx & I_y(\mathbf{x}+\mathbf{w}^{n}) + I_{xy}(\mathbf{x}+\mathbf{w}^{n})du^n + I_{yy}(\mathbf{x}+\mathbf{w}^{n})dv^n.
\end{aligned}
\tag{4}
$$

As proposed in [6], we work with *motion increments* (du^n, dv^n), so the optical flow can be iteratively estimated as $u^{n+1} = u^n + du^n$ and $v^{n+1} = v^n + dv^n$.

We introduce a second index, m, that accounts for the nonlinearities of the ψ' function. Then, we obtain the system of equations (5) combining both fixed point schemes, which is solved using the successive over-relaxation (SOR) method. This is an iterative procedure that solves linear systems of equations, resulting in faster convergence.

$$
\begin{aligned}
0 =& (\psi'_B)^{n,m} \cdot \left(\sum_{c=1}^{C} \left(I_2^c(\mathbf{y}) + I_{2,x}^c(\mathbf{y})du^{n,m+1} + I_{2,y}^c(\mathbf{y})dv^{n,m+1} - I_1^c(\mathbf{x}) \right) \cdot I_{2,x}^c(\mathbf{y}) \right) \\
&+ \gamma\, (\psi'_G)^{n,m} \cdot \left(\sum_{c=1}^{C} \left(I_{2,x}^c(\mathbf{y}) + I_{2,xx}^c(\mathbf{y})du^{n,m+1} + I_{2,xy}^c(\mathbf{y})dv^{n,m+1} - I_{1,x}^c(\mathbf{x}) \right) \cdot I_{2,xx}^c(\mathbf{y}) \right. \\
&\quad \left. + \sum_{c=1}^{C} \left(I_{2,y}^c(\mathbf{y}) + I_{2,xy}^c(\mathbf{y})du^{n,m+1} + I_{2,yy}^c(\mathbf{y})dv^{n,m+1} - I_{1,y}^c(\mathbf{x}) \right) \cdot I_{2,xy}^c(\mathbf{y}) \right) \\
&- \alpha\, \mathrm{div}\left((\psi'_S)^{n,m} \cdot \nabla(u^{n,m} + du^{n,m+1}) \right) \\
\\
0 =& (\psi'_B)^{n,m} \cdot \left(\sum_{c=1}^{C} \left(I_2^c(\mathbf{y}) + I_{2,x}^c(\mathbf{y})du^{n,m+1} + I_{2,y}^c(\mathbf{y})dv^{n,m+1} - I_1^c(\mathbf{x}) \right) \cdot I_{2,y}^c(\mathbf{y}) \right) \\
&+ \gamma\, (\psi'_G)^{n,m} \cdot \left(\sum_{c=1}^{C} \left(I_{2,x}^c(\mathbf{y}) + I_{2,xx}^c(\mathbf{y})du^{n,m+1} + I_{2,xy}^c(\mathbf{y})dv^{n,m+1} - I_{2,x}^c(\mathbf{x}) \right) \cdot I_{2,xy}^c(\mathbf{y}) \right. \\
&\quad \left. + \sum_{c=1}^{C} \left(I_{2,y}^c(\mathbf{y}) + I_{2,xy}^c(\mathbf{y})du^{n,m+1} + I_{2,yy}^c(\mathbf{y})dv^{n,m+1} - I_{2,y}^c(\mathbf{x}) \right) \cdot I_{2,yy}^c(\mathbf{y}) \right) \\
&- \alpha\, \mathrm{div}\left((\psi'_S)^{n,m} \cdot \nabla(v^{n,m} + dv^{n,m+1}) \right),
\end{aligned}
\tag{5}
$$

with $\mathbf{y} = \mathbf{x} + \mathbf{w}^{n,m}$.

The unknowns $du^{n,m+1}$ and $dv^{n,m+1}$, in pixel (i, j), are expressed as a function of the remaining terms and their values are iteratively updated until the method converges to a steady state solution. In this sense, we introduce an additional fixed point iteration scheme, s, for the SOR method.

Among other kinds of schemes, the partial derivatives are approximated using central differences and we discretize the divergence using three variables, as follows

$$
\begin{aligned}
\mathrm{div}\left((\psi'_S)^{n,m} \cdot \nabla(u^{n,m} + du^{n,m+1}) \right) &= \mathrm{div}\left((\psi'_S)^{n,m} \cdot \nabla u^{n,m} \right) + \mathrm{div}\left((\psi'_S)^{n,m} \cdot \nabla du^{n,m+1} \right) \\
&\approx div\_u + (div\_du - div\_d \cdot du_{i,j,n}^{n,m+1}),
\end{aligned}
\tag{6}
$$

where div_u discretizes the first divergence term, div_d and div_du correspond to the second term. In the second term, div_du stands for the values corresponding to the neighbors of du, and div_d stands

0	$\frac{(\psi'_S)_{i,j+1} + (\psi'_S)^{n,m}_{i,j}}{2}$	0
$\frac{(\psi'_S)_{i-1,j} + (\psi'_S)^{n,m}_{i,j}}{2}$	$-\frac{(\psi'_S)_{i+1,j} + (\psi'_S)_{i-1,j} + (\psi'_S)_{i,j+1} + (\psi'_S)(\psi'_S)_{i,j-1} + 4(\psi'_S)^{n,m}_{i,j}}{2}$	$\frac{(\psi'_S)_{i+1,j} + (\psi'_S)^{n,m}_{i,j}}{2}$
0	$\frac{(\psi'_S)_{i,j-1} + (\psi'_S)^{n,m}_{i,j}}{2}$	0

Table 1: Coefficients of the diffusion tensor stencil.

for the coefficients accompanying du at the current pixel, $du^{n,m+1}_{i,j}$. These variables are given by

$$\begin{aligned} div\_u :=& \frac{(\psi'_S)_{i+1,j} + (\psi'_S)^{n,m}_{i,j}}{2}\left(u^{n,m}_{i+1,j} - u^{n,m}_{i,j}\right) + \frac{(\psi'_S)_{i-1,j} + (\psi'_S)^{n,m}_{i,j}}{2}\left(u^{n,m}_{i-1,j} - u^{n,m}_{i,j}\right) + \\ & \frac{(\psi'_S)_{i,j+1} + (\psi'_S)^{n,m}_{i,j}}{2}\left(u^{n,m}_{i,j+1} - u^{n,m}_{i,j}\right) + \frac{(\psi'_S)_{i,j-1} + (\psi'_S)^{n,m}_{i,j}}{2}\left(u^{n,m}_{i,j-1} - u^{n,m}_{i,j}\right), \end{aligned} \tag{7}$$

$$\begin{aligned} div\_du :=& \frac{(\psi'_S)_{i+1,j} + (\psi'_S)^{n,m}_{i,j}}{2} du^{n,m+1}_{i+1,j} + \frac{(\psi'_S)_{i-1,j} + (\psi'_S)^{n,m}_{i,j}}{2} du^{n,m+1}_{i-1,j} + \\ & \frac{(\psi'_S)_{i,j+1} + (\psi'_S)^{n,m}_{i,j}}{2} du^{n,m+1}_{i,j+1} + \frac{(\psi'_S)_{i,j-1} + (\psi'_S)^{n,m}_{i,j}}{2} du^{n,m+1}_{i,j-1}, \end{aligned} \tag{8}$$

$$\begin{aligned} div\_d :=& \frac{(\psi'_S)_{i+1,j} + (\psi'_S)^{n,m}_{i,j}}{2} + \frac{(\psi'_S)_{i-1,j} + (\psi'_S)^{n,m}_{i,j}}{2} + \\ & \frac{(\psi'_S)_{i,j+1} + (\psi'_S)^{n,m}_{i,j}}{2} + \frac{(\psi'_S)_{i,j-1} + (\psi'_S)^{n,m}_{i,j}}{2}. \end{aligned} \tag{9}$$

These expressions are the same for the other component of the optical flow, changing u by v. Their mask representation is shown in Table 1. If we define $\mathbf{y} = \mathbf{x} + \mathbf{w}^{n,m}$ and separate the parts of the equation that remain constant during the SOR iterations, we may define the variables

$$\begin{aligned} Au :=& -(\psi'_B)^{n,m} \cdot \left(\sum_{c=1}^{C} (I^c_2(\mathbf{y}) - I^c_1(\mathbf{x})) \cdot I^c_{2,x}(\mathbf{y})\right) + \alpha\, div\_u \\ & - \gamma\, (\psi'_G)^{n,m} \cdot \left(\sum_{c=1}^{C} \left(I^c_{2,x}(\mathbf{y}) - I^c_{1,x}(\mathbf{x})\right) \cdot I^c_{2,xx}(\mathbf{y}) + \left(I^c_{2,y}(\mathbf{y}) - I^c_{1,y}(\mathbf{x})\right) \cdot I^c_{2,xy}(\mathbf{y})\right), \\ Av :=& -(\psi'_B)^{n,m} \cdot \left(\sum_{c=1}^{C} (I^c_2(\mathbf{y}) - I^c_1(\mathbf{x})) \cdot I^c_{2,y}(\mathbf{y})\right) + \alpha\, div\_v \\ & - \gamma\, (\psi'_G)^{n,m} \cdot \left(\sum_{c=1}^{C} \left(I^c_{2,x}(\mathbf{y}) - I^c_{1,x}(\mathbf{x})\right) \cdot I^c_{2,xy}(\mathbf{y}) + \left(I^c_{2,y}(\mathbf{y}) - I^c_{1,y}(\mathbf{x})\right) \cdot I^c_{2,yy}(\mathbf{y})\right), \\ Du :=& (\psi'_B)^{n,m} \cdot \sum_{c=1}^{C} I^c_{2,x}(\mathbf{y}) + \gamma\, (\psi'_G)^{n,m} \cdot \sum_{c=1}^{C} \left(I^c_{2,xx}(\mathbf{y}) + I^c_{2,xy}(\mathbf{y})\right) + \alpha\, div\_d, \\ Dv :=& (\psi'_B)^{n,m} \cdot \sum_{c=1}^{C} I^c_{2,y}(\mathbf{y}) + \gamma\, (\psi'_G)^{n,m} \cdot \sum_{c=1}^{C} \left(I^c_{2,xy}(\mathbf{y}) + I^c_{2,yy}(\mathbf{y})\right) + \alpha\, div\_d, \\ D :=& (\psi'_B)^{n,m} \cdot \sum_{c=1}^{C} \left(I^c_{2,x}(\mathbf{y}) \cdot I^c_{2,y}(\mathbf{y})\right) + \gamma\, (\psi'_G)^{n,m} \cdot \sum_{c=1}^{C} \left(I^c_{2,xx}(\mathbf{y}) + I^c_{2,yy}(\mathbf{y})\right) \cdot I^c_{2,xy}(\mathbf{y}). \end{aligned} \tag{10}$$

We compute expressions like $I_2^c(\mathbf{x}+\mathbf{w}^{n,m})$ using bicubic interpolation. Putting all together, we arrive at the SOR method (index s), which is given by

$$\begin{aligned} du^{n,m,s+1} :=& \frac{(1-w)\, du^{n,m,s} + w\,(Au - D \cdot dv^{n,m,s+1} + \alpha\, div\_du)}{Du}, \\ dv^{n,m,s+1} :=& \frac{(1-w)\, dv^{n,m,s} + w\,(Av - D \cdot du^{n,m,s+1} + \alpha\, div\_dv)}{Dv}, \end{aligned} \tag{11}$$

with $w \in (0,2)$ the SOR relaxation parameter. In our implementation, we choose $w = 1.9$ by default. This numerical approximation is calculated until the method converges to a steady state solution or exceeds a maximum number of iterations. The stopping criterion is

$$\frac{1}{N}\sum_{i,j}\left(\left(du_{i,j}^{s+1} - du_{i,j}^{s}\right)^2 + \left(dv_{i,j}^{s+1} - dv_{i,j}^{s}\right)^2\right) < \varepsilon^2, \tag{12}$$

with N the number of pixels in all frames and ε the stopping criterion threshold. The iterative process stops when condition (12) is true or a maximum number of iterations is reached. Once it has converged, we go to the next inner iteration, m, and restart the variables in (10). If the parameters are correctly chosen, the method converges in few iterations.

3 Pyramidal Structure

In order to estimate large displacements, we use a coarse-to-fine scheme based on a pyramidal structure. We follow the same strategy presented in the IPOL articles [9] and [16]. Before downsampling, the algorithm smoothes the images with a Gaussian kernel of a standard deviation that depends on $\sigma \in (0,1)$. Then, it creates a pyramid of downsampled images using a scale factor $\eta \in (0,1)$. For a set of scales $s = 0, 1, \ldots, N_{scales} - 1$, the pyramid of images is built as

$$I^s(\eta\mathbf{x}) := G_\sigma * I^{s-1}(\mathbf{x}). \tag{13}$$

Next, we use bicubic interpolation for sampling the images after the convolution. The value of σ depends on η and is calculated as

$$\sigma(\eta) := \sigma_0\sqrt{\eta^{-2} - 1},\ with\ \sigma_0 := 0.6. \tag{14}$$

Finally, we solve the system of equations at each scale, starting at the coarsest scale, to get successive approximations of the optical flow. Every scale is initialized using the previous result as

$$\begin{aligned} u^{s-1}(\mathbf{x}) &:= \frac{1}{\eta}u^s(\eta\mathbf{x}), \\ v^{s-1}(\mathbf{x}) &:= \frac{1}{\eta}v^s(\eta\mathbf{x}). \end{aligned} \tag{15}$$

4 Regularization Strategies

The regularization behavior relies on $\Phi(\nabla I_1)$. Table 2 shows the three alternatives: the first one corresponds to the original approximation while the others are the two proposals for solving the instability problems. In this work, we named the strategies *DF*, *DF-β* and *DF-Auto*, respectively.

The *DF-β* scheme includes a constant parameter (β) to ensure a minimum isotropic diffusion. This maintains a sufficient amount of diffusion even when the gradient is very large. On the other

Strategy	$\Phi(\nabla I_1)$	Index in the algorithm
DF	$e^{-\lambda\|\nabla I_1\|}$	1
DF-β	$e^{-\lambda\|\nabla I_1\|}+\beta$	2
DF-Auto	$e^{-\lambda_{auto}\|\nabla I_1\|}$	3

Table 2: Regularization strategies. The first alternative offers regularization with a decreasing scalar function. The second ensures constant diffusion when using the previous scheme. The third provides an automatic adaptation of the parameter that controls the decreasing scalar function.

hand, the *DF-Auto* approach differs from the others in that λ_{auto} is automatically computed to avoid instabilities. This is calculated by expanding the smoothness term in (2),

$$0 \cong \alpha \operatorname{div}(\psi'_S \cdot \nabla u),$$
$$0 \cong \alpha \operatorname{div}(\psi'_S \cdot \nabla v).$$

The instabilities normally arise when $\alpha \cdot \psi'_S \cong 0$. In this sense, one way to avoid the ill-posedness is to ensure the following condition

$$\alpha \cdot \psi'_S = \frac{\alpha \cdot e^{-\lambda|\nabla I_1|}}{\sqrt{e^{-\lambda|\nabla I_1|} \cdot (|\nabla u|^2 + |\nabla v|^2) + \epsilon^2}} \geq \xi > 0,$$

with ξ a small constant.

We can impose the following constraint: $\alpha e^{-\lambda|\nabla I_1|} \geq \xi > 0$. Then, we obtain λ as

$$\lambda(\mathbf{x}) := \frac{-\ln(\xi) + \ln(\alpha)}{|\nabla I_1(\mathbf{x})|}.$$

Note that the gradient of the image is a function of $\mathbf{x}$. When the gradient is close to zero, λ tends to ∞. This calculation must discriminate whether a pixel belongs to an inhomogeneous region or not. We propose to calculate λ_{auto} at each pixel relying on the histogram of the gradient as

$$\lambda_{auto} := \min\{\lambda_\Omega, \lambda(\mathbf{x})\},$$

where

$$\lambda_\Omega := \frac{-\ln(\xi) + \ln(\alpha)}{|\nabla I_1(\mathbf{x}')|}, \tag{16}$$

where $\mathbf{x}'$ is such that $|\nabla I_1(\mathbf{x}')|$ is the rank $\tau \times |\Omega|$ among the image gradients and $\tau := 0.94$ is a suitable value, as seen in Monzón et al. [10].

5 Parameters of the Method

Our algorithm depends on the parameters given in Table 3 and the constants of Table 4. The first parameter is an integer for selecting the diffusion behavior (see Table 2). The weighting parameters used in the energy functional are α and γ; λ is used in the non-automatic strategies for controlling the diffusion at image borders.

The remaining parameters are the following: N_{scales}, η stand for the number of scales and the downsampling factor in the pyramidal scheme and σ_0 used in the image convolution; the parameters for the numerical scheme, given by the inner and outer iterations, and the stopping criterion threshold (ε); β, ξ and τ are constants that support the regularization strategies as explained in the previous section.

Table 3: Parameters of the method.

Parameter	Explanation
method_type	Integer that selects the regularization strategy (see Table 2).
α	Regularization parameter. It determines the smoothness strength. The bigger this parameter, the smoother the solutions we obtain. In our algorithm, it is adapted to the number of channels.
γ	Parameter associated with the gradient constancy term in Equation (1).
λ	Used in *DF* and *DF-*β methods. It determines the influence of the exponential function in the regularization. Its value is automatic using the *DF-Auto* approach.

Table 4: Constant parameters of the method.

Parameter	Explanation
N_{scales}	Number of scales in the pyramidal structure. If the flow field is very small (about one pixel), it can be set to 1. Otherwise, it should be set so that $(1/\eta)^{N-1}$ is larger than the expected size of the largest displacement. N_{scales} is automatically calculated so that the image size, at the coarsest scale, is around 16×16 pixels.
η	Downsampling factor. It is used to downscale the original images in order to create the pyramidal structure. Its value must be in the interval $(0, 1)$. With $\eta = 0.5$, the images are reduced to half their size in each dimension from one scale to the following. We fix it to $\eta := 0.75$.
σ_0	It is used in the image convolution in Equation (14). We use $\sigma_0 := 0.6$.
ε	Stopping criterion threshold. It is the threshold used to stop the SOR iterations, given in Equation (12). Its value is $\varepsilon := 0.001$.
inner_iter	Number of inner iterations in the numerical scheme. It corresponds to index n in Equation (11). We use *inner_iterations* $:= 1$
outer_iter	Number of outer iterations in the numerical scheme. It corresponds to index m in Equation (11). We use *outer_iterations* $:= 10$
β	Used in *DF-*β method. It is a constant that ensures a minimum diffusion. We fix its value to $\beta := 0.001$.
ξ	Used in *DF-Auto* method. It is a constant that determines if the diffusivity is sufficient to avoid the ill-posedness. Its value is $\xi := 0.05$.
τ	Used in *DF-Auto* method. It is a constant that determines the conservative behavior of the automatic λ. It prevents the occurrence of instabilities. We fix it value to $\tau := 0.94$ (see [10] for a detailed explanation).

6 Algorithm

Next, we describe the algorithm that implements the numerical scheme in Equation (11). The algorithm takes two color images as input data and computes the optical flow. We separate the algorithm in two modules: one procedure that computes the optical flow at each scale and the main algorithm that is in charge of handling the pyramidal structure.

The main Algorithm 1 creates the pyramidal structure, as explained in Section 3, and calls

Algorithm 2 for computing the optical flow, starting from the coarsest scale. It adapts the result at each scale to be used as the initial approximation at the following scale.

Algorithm 1: Pyramidal structure management

Input: $I_1^c, I_2^c, u, v, method\_type, \alpha, \gamma, \lambda, N_{scales}, \eta = 0.5, \varepsilon := 0.001, inner\_iter := 1, outer\_iter := 10, nchannels$

Output: u, v

```
Normalize multi-channel images between 0 and 255
Convolve the images with a Gaussian of σ ← 0.8
Create the pyramid of images I_1^{c,s}, I_2^{c,s} using η (with s ← 0, ..., N_scales − 1)
α_c ← α· nchannels
for s ← (N_scales − 1) to 0 do
    robust_DF_methods(I_1^{c,s}, I_2^{c,s}, u^s, v^s, method_type, α_c, γ, λ, inner_iter, outer_iter)
    if s > 0 then
        u^{s−1}(x) ← (1/η) u^s(η x)
        v^{s−1}(x) ← (1/η) v^s(η x)
    end
end
```

Algorithm 2 calculates Φ using the regularization scheme defined by the parameter *method_type*. The result is used before the inner-iterations to calculate Equation (3) using procedure `exponential_calculation`. When using *DF* or *DF-β* strategies, the program looks for the maximum gradient in every pixel of the multichannel image in order to calculate the exponential function.

On the other hand, for the *DF-Auto* approach, we use procedure `automatic_lambda` to calculate λ and the maximum gradient per pixel for the exponential, as in the previous case. This permits to accelerate the method by doing a unique search over the image. This allows us to solve both problems at once. Finally, we have used the OpenMP library to parallelize the algorithm making it faster and achieving fast running times (see the online demo).

7 Experiments

In this section, we compare the strategies described in this work. First, we show their parametric stability and the effects on the flow contours. Second, we analyze the influence of the smoothness weight with respect to different values of γ. Parameters α, γ and λ are modified in each experiment. The optical flows are represented with the color scheme shown in Figure 1. The color represents the orientation and the intensity its magnitude.

Figures 2 and 3 show the behavior of the methods with respect to the discontinuity parameter using the *Urban2* and *Shaman 2* sequences of the Middlebury [3] and MPI-Sintel [7] datasets, respectively. The first row presents the original image and the average End-Point Error (EPE) evolution with respect to λ. The second row depicts the true flow, the result of using Brox and the flow obtained with the *DF-Auto* approach. The third and fourth rows show the motion fields for the *DF* and *DF-β* strategies when increasing λ. Figures 4 and 5 show similar examples using a pair of frames from the natural sequences of *Rheinhafen* and *Ettlinger-Tor*.

In general, the Brox solutions have problems with motion contours. In fact, the flow discontinuities are typically not aligned with the object borders except in *Urban2*. Besides, in the *Rheinhafen* and *Ettlinger-Tor* results, some cars do not appear in the motion field because of a strong regularization.

Algorithm 2: robust_DF_methods($I_1^c, I_2^c, u, v, method\_type, \alpha, \gamma, \lambda, inner\_iter, outer\_iter$)

Input: $I_1^c, I_2^c, method\_type, \alpha, \gamma, \lambda, \varepsilon := 0.001, inner\_iter := 1, outer\_iter := 10, \omega := 0.9$
Output: u, v

Compute, for all channels, $I_{1,x}^c, I_{1,y}^c, I_{2,x}^c, I_{2,y}^c$
Compute, for all channels, $I_{2,xx}^c, I_{2,yy}^c, I_{2,xy}^c$
`exponential_calculation` $(I_{1x}, I_{1y}, \alpha, \lambda, method\_type, \Phi)$
for $no \leftarrow 0$ **to** $outer\_iter - 1$ **do**
    Compute $I_2^c(\mathbf{x}+\mathbf{w}), I_{2,x}^c(\mathbf{x}+\mathbf{w}), I_{2,y}(\mathbf{x}+\mathbf{w})$ using bicubic interpolation
    Compute $I_{2,xx}^c(\mathbf{x}+\mathbf{w}), I_{2,xy}^c(\mathbf{x}+\mathbf{w}), I_{2,yy}^c(\mathbf{x}+\mathbf{w})$ using bicubic interpolation
    Compute the flow gradients u_x, u_y, v_x, v_y
    Compute ψ'_S using Equation (3) and Φ
    Compute div_u, div_v, div_d using equations (7) and (9)
    $(du, dv) \leftarrow (0, 0)$
    for $ni \leftarrow 0$ **to** $inner\_iter - 1$ **do**
        Compute ψ'_D, ψ'_G using Equation (3)
        Compute Au, Av, Du, Dv, D using Equation (10)
        $nsor \leftarrow 0$
        while $error > \varepsilon$ **and** $nsor < MAXITER$ **do**
            $du \leftarrow (1-\omega)\, du + \omega \cdot \frac{Au - D\, dv + \alpha\, div\_du}{Du}$
            $dv \leftarrow (1-\omega)\, dv + \omega \cdot \frac{Av - D\, du + \alpha\, div\_dv}{Dv}$
            Compute $error$ with Equation (12)
            $nsor \leftarrow nsor + 1$
        end
    end
    $(u, v) \leftarrow (u + du, v + dv)$
end

The exponential strategies offer an interesting improvement at flow edges. The corners of the buildings in *Urban2* are better preserved compared with the Brox solution and the borders of the vehicles in *Rheinhafen* and *Ettlinger-Tor* sequences are better defined. However, we also observe a staircasing problem in the floor of the street in *Urban2*. This negative effect occurs because the decreasing function depends on the image gradient treating some image borders as flow edges.

These figures are a good example of the good stability of the new proposals in comparison with the pure exponential function. The errors of the *DF* method rapidly turn unstable once the best flow is achieved. However, the instabilities are strongly diminished using the *DF-β* approach. In the case of the *Urban2* sequence, the method does not completely eliminate the instabilities but they are strongly reduced.

Interestingly, the *DF-Auto* method usually calculates good optical flows without relevant blobs. The automatic parameter adapts its value according to the image gradient. This simplifies the parametric configuration required by the other alternatives. Besides, according to the EPE graphics, we see that this strategy usually finds a solution close to the best result.

The graphics of the first and second columns of Figure 6 show the EPE evolution with respect to α for $\gamma = 0$ and $\gamma = 1$. The analysis of the Brox method carried out in the IPOL article [16]

Procedure `exponential_calculation`$(I^c_{1,x}, I^c_{1,y}, \alpha, \lambda, \beta, method\_type, \Phi)$

Input: $I^c_{1,x}, I^c_{1,y}, \alpha, \lambda, \beta := 0.001, method\_type$
Output: Φ

switch *method_type* **do**
  case *1:2*
    $\beta_v \leftarrow 0$
    if $method\_type = 2$ **then** $\beta_v \leftarrow \beta$
    Compute maximum gradiend (g_{max})
    foreach *pixel p* **do**
      $\Phi(p) \leftarrow e^{(-\lambda \cdot g_{max})} + \beta_v$
    end
  case *3:*
    $\lambda_\Omega \leftarrow$ `automatic_lambda` $(I_x, I_y, \alpha, \lambda_p, g_{max})$
    foreach *pixel p* **do**
      $\lambda_{auto} \leftarrow \lambda_\Omega$
      if $\lambda_\Omega > \lambda_p(p)$ **then**
        $\lambda_{auto} \leftarrow \lambda_p(p)$
      end
      $\Phi(p) \leftarrow e^{(-\lambda_{auto} \cdot g_{max})}$
    end
  end
end

Procedure `automatic_lambda`

Input: $I_x, I_y, \alpha, \tau := 0.94, \xi := 0.05$
Output: λ_Ω, λ_p, g_{max}

foreach *pixel p* **do**
  $g_{max}(p) \leftarrow$ Compute the maximum gradient for each image channel at every pixel
  $\lambda(p) \leftarrow \frac{-\log(\xi)+\log(\alpha)}{g_{max}(p)}$
  if $max\_gradient(p) < g_{max}(p)$ **then** $max\_gradient(p) \leftarrow g_{max}(p)$
end
$gradient\_sorted \leftarrow sort(max\_gradient)$
$pos\_ref \leftarrow \tau \cdot image\_width \cdot image\_height$
$\lambda_\Omega \leftarrow \frac{-\log(\xi)+\log(\alpha)}{gradient\_sorted(pos\_ref)}$
return λ_Ω

concluded that $\gamma = 7$ provides the best results for the entire *Middlebury* dataset. We use this value for the graphics of the third column of Figure 6.

The purpose of this experiment is to observe the dependency of the exponential strategies with respect to the gradient parameter. We fix $\lambda := 0.2$ for *DF* and *DF-*β. We use the synthetic sequences of *Grove2* and *Urban3* from Middlebury and *Alley 1* and *Shaman 2* from the Sintel dataset. The

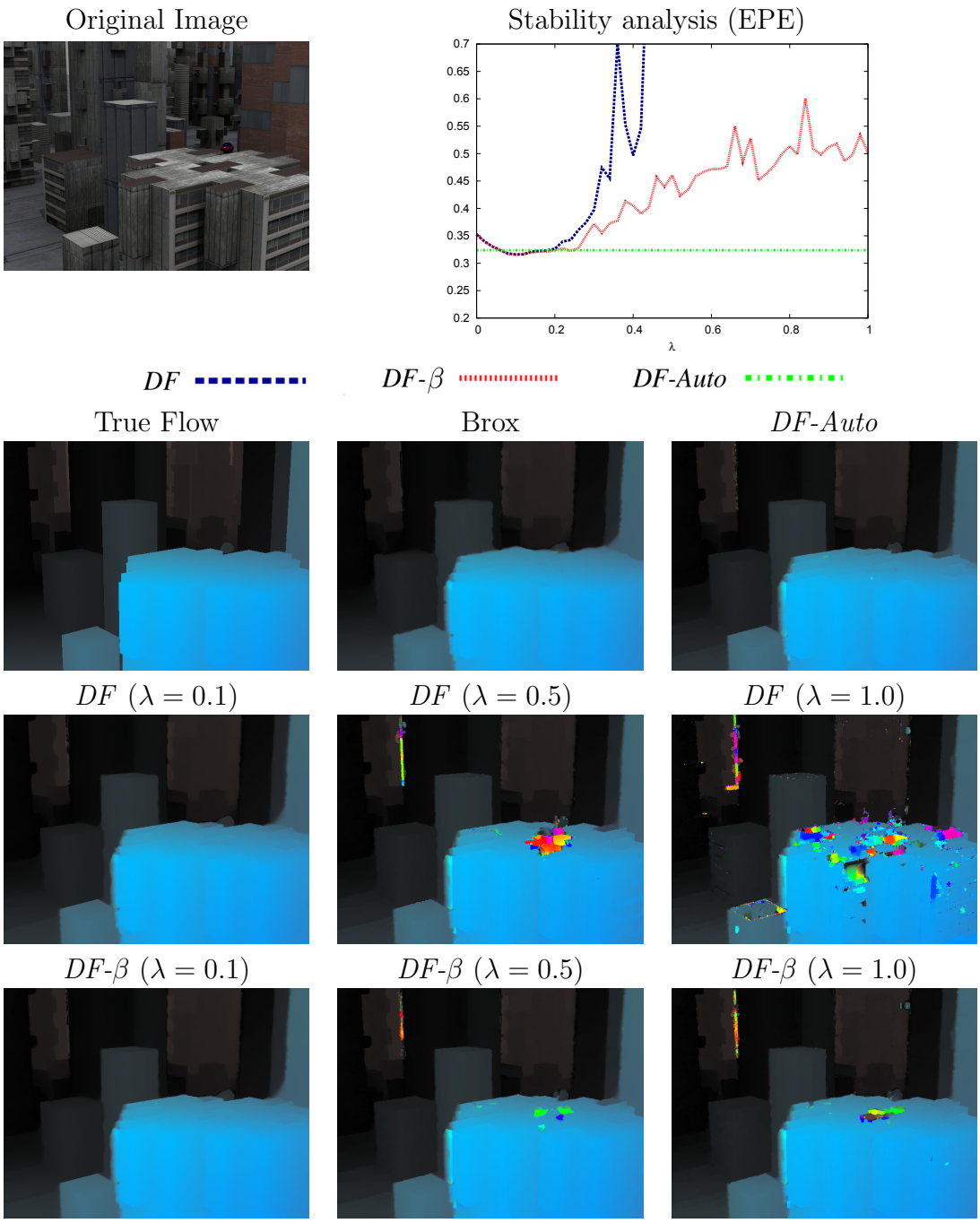

Figure 2: *Urban2* sequence. First row: original image and its EPE graphic with respect to λ. Second row: the true flow and the solutions obtained with Brox and *DF-Auto* methods, respectively. Third and fourth rows: Flow fields for increasing values of the λ parameter (*DF* and *DF-*β approaches, respectively). We observe fewer artifacts over the flow with the new proposals. The automatic approach finds a solution with strongly reduced instabilities.

DF, *DF-*β and *DF-Auto* methods are represented with a blue, red and green line, respectively.

According to the graphics, we see that the behavior of *DF* and *DF-*β are similar in many cases. The results of *DF-Auto* are more stable in general, especially in the *Urban3* and *Shaman 2* sequences. This means that the search space of the α and γ parameters is reduced for the *DF-Auto* method, so these are easier to configure.

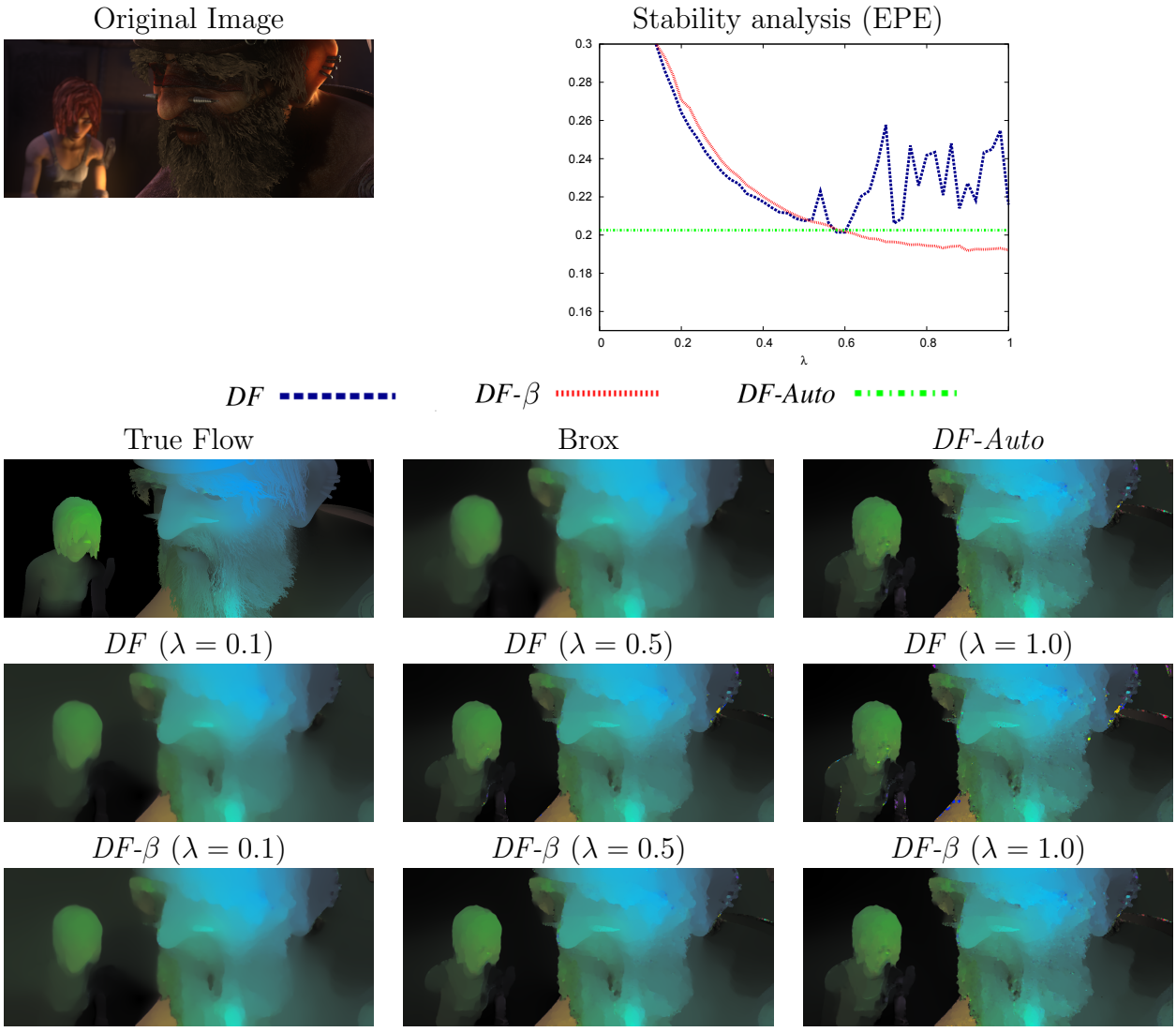

Figure 3: *Shaman 2* sequence. First row: original image and its EPE graphic with respect to λ. Second row: the true flow and the solutions obtained with Brox and *DF-Auto* methods, respectively. Third and fourth rows: Flow fields for increasing values of the λ parameter (*DF* and *DF-*β approaches, respectively). We can observe that *DF-Auto* adapts the discontinuity parameter to achieve a good preservation of the motion contours.

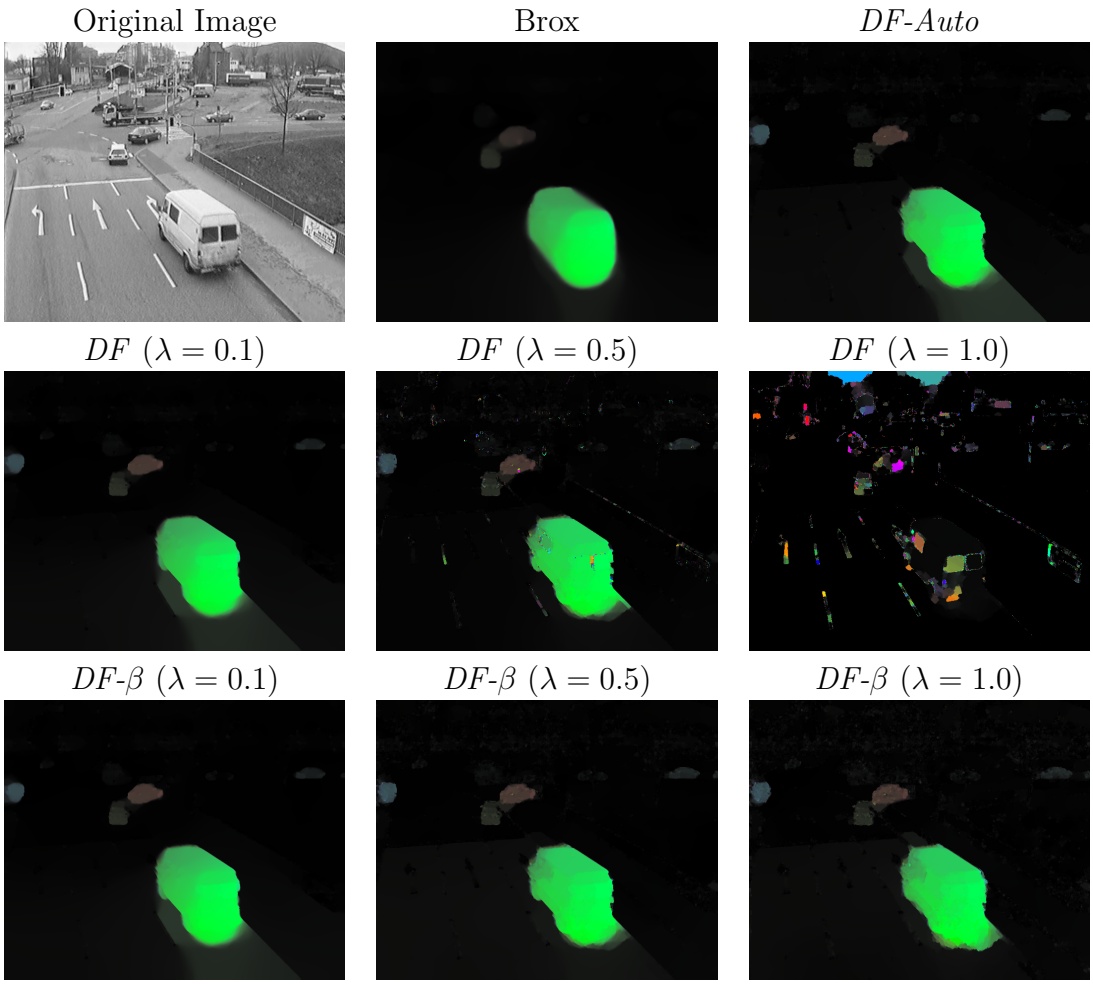

Figure 4: *Rheinhafen* sequence. First row: Original image, Brox and *DF-Auto* solutions. Second and third row: Flow fields for increasing values of the λ parameter (*DF* and *DF-*β approaches, respectively). We observe an interesting stability of the λ parameter when using the *DF-*β approach. The new proposals present a better definition of the motion contours in comparison to Brox method.

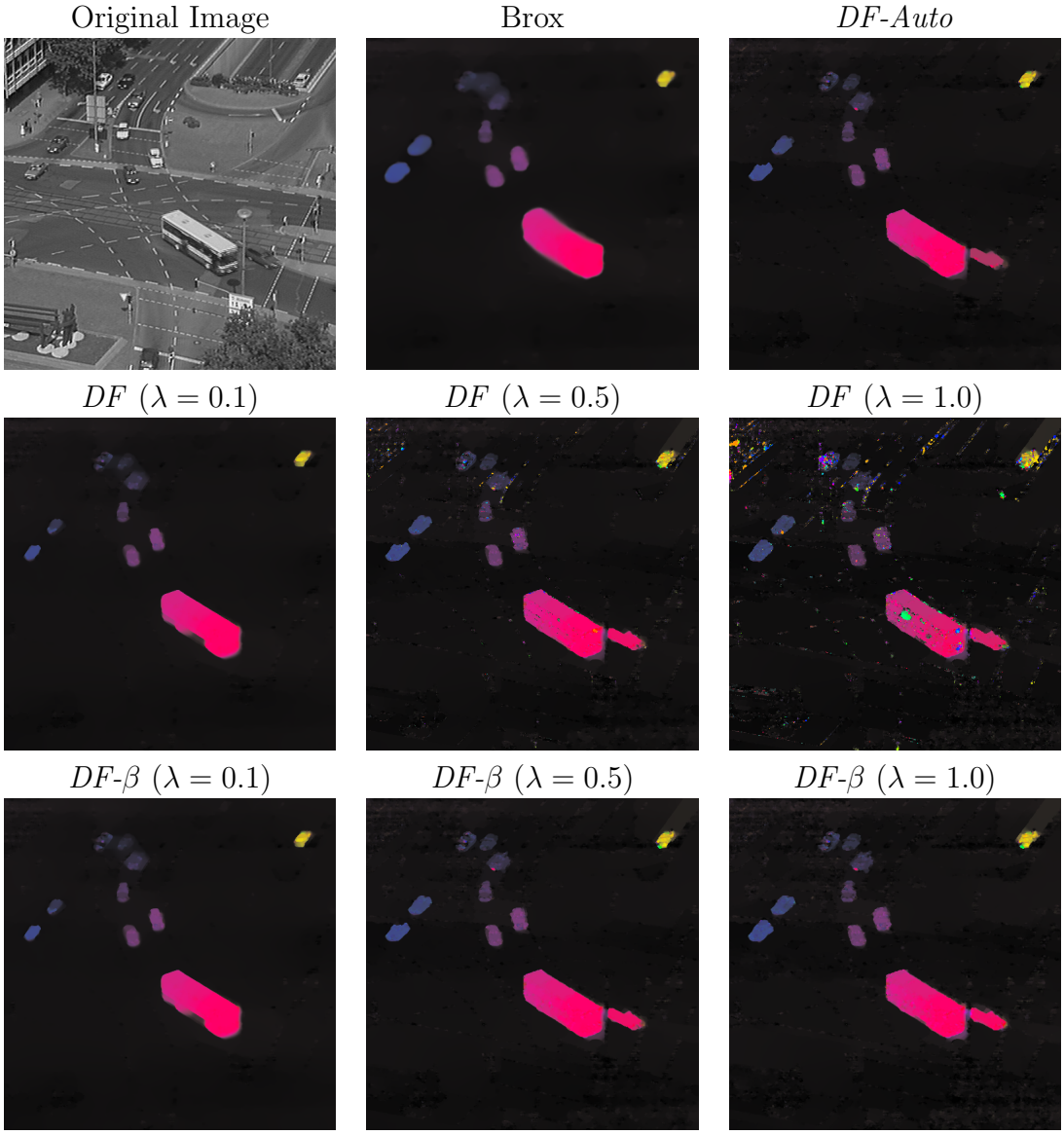

Figure 5: *Ettlinger-Tor* sequence. First row: Original image, Brox and *DF-Auto* solutions. Second and third row: Flow fields for increasing values of the λ parameter (*DF* and *DF-*β approaches, respectively). Some cars disappear in the flow because of a strong regularization when using Brox. However, the exponential methods preserve these vehicles and permit an improvement in the object borders.

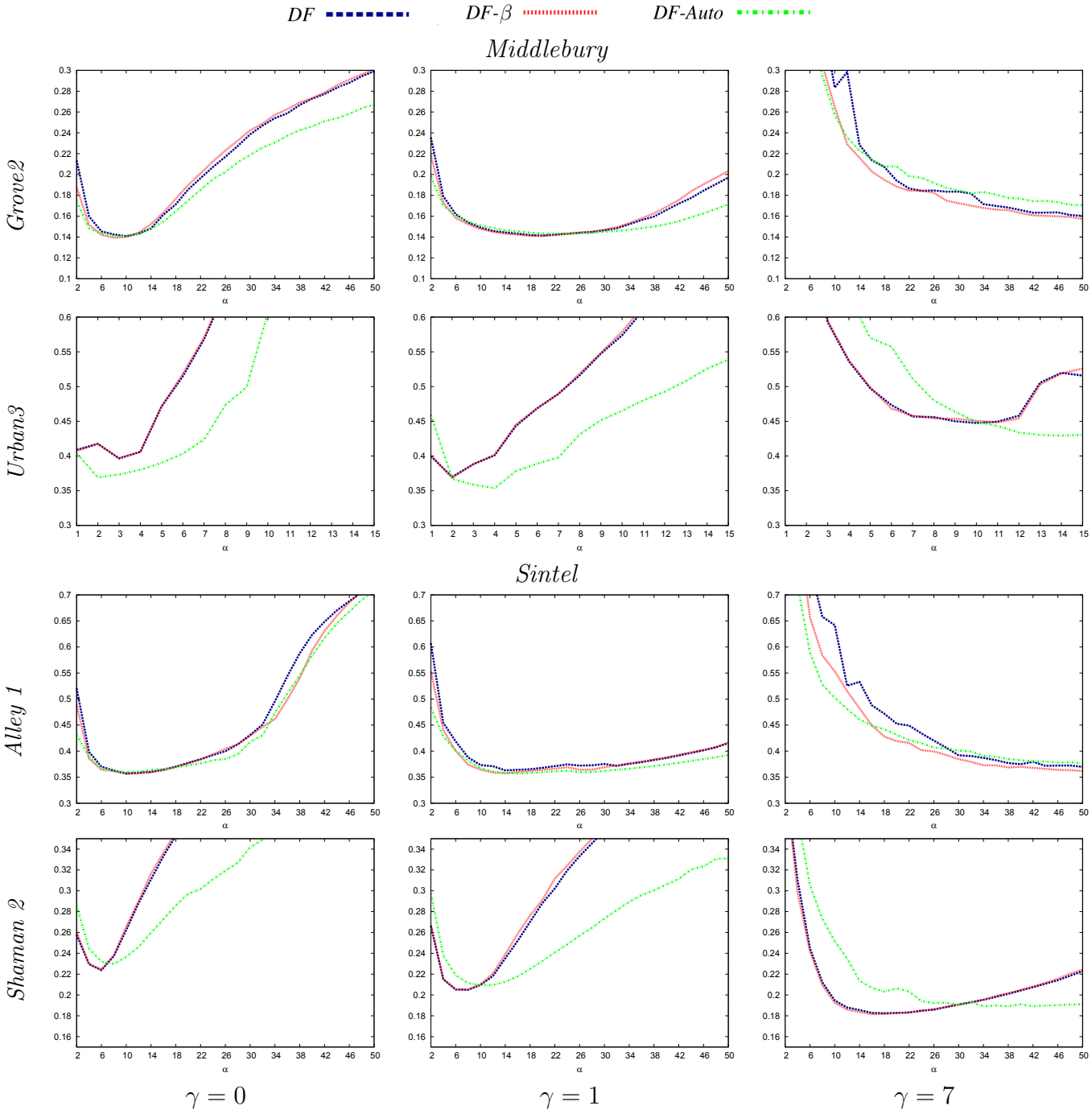

Figure 6: EPE evolution with respect to α for some *Middlebury* and *Sintel* sequences fixing γ with different values. $\lambda = 0.2$ for *DF* and *DF-β* methods while it is automatically calculated in *DF-Auto*.

Acknowledgements

This work has been partly founded by the BPIFrance and Région Ile de France, in the framework of the FUI 18 Plein Phare project, the European Research Council (advanced grant Twelve Labours) and the Office of Naval research (ONR grant N00014-14-1-0023).

Image Credits

All images by the authors except:

 Daniel Scharstein, Middlebury benchmark database[2].

 H.H. Nagel, Rheinhafen[3].

 Henner Kollnig, Ettlinger-Tor[4].

 Michael J. Black, MPI-Sintel dataset[5].

References

[1] L. Álvarez, J. Esclarín, M. Lefébure, and J. Sánchez. A PDE model for computing the optical flow. In *XVI Congreso de Ecuaciones Diferenciales y Aplicaciones, C.E.D.Y.A. XVI*, pages 1349–1356, Las Palmas de Gran Canaria, Spain, 1999.

[2] L. Álvarez, J. Weickert, and J. Sánchez. Reliable estimation of dense optical flow fields with large displacements. *International Journal of Computer Vision*, 39(1):41–56, 2000. http://dx.doi.org/10.1023/A:1008170101536.

[3] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *International Conference on Computer Vision*, pages 1–8, 2007. http://dx.doi.org/10.1109/ICCV.2007.4408903.

[4] M.J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *Proceedings of Fourth International Conference on Computer Vision*, pages 231–236, 1993. http://dx.doi.org/10.1109/ICCV.1993.378214.

[5] M.J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75 – 104, 1996. http://dx.doi.org/10.1006/cviu.1996.0006.

[6] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision (ECCV)*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36, Prague, Czech Republic, May 2004. Springer. http://dx.doi.org/10.1007/978-3-540-24673-2_3.

[2] http://vision.middlebury.edu/flow/data/
[3] http://i21www.ira.uka.de/image_sequences/
[4] http://i21www.ira.uka.de/image_sequences/
[5] http://sintel.is.tue.mpg.de/

[7] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

[8] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981. http://dx.doi.org/10.1016/0004-3702(81)90024-2.

[9] E. Meinhardt-Llopis, J. Sánchez, and D. Kondermann. Horn-Schunck optical flow with a multi-scale strategy. *Image Processing On Line*, 2013:151–172, 2013. http://dx.doi.org/10.5201/ipol.2013.20.

[10] N. Monzón, A. Salgado, and J. Sánchez. Regularization strategies for discontinuity-preserving optical flow methods. *IEEE Transactions on Image Processing*, 25(4):1580–1591, April 2016. http://dx.doi.org/10.1109/TIP.2016.2526903.

[11] N. Monzón, J. Sánchez, and A. Salgado. Optic flow: Improving discontinuity preserving. In *EUROCAST'13: Proceedings of the 14th international conference on Computer aided systems theory*, pages 114–116, Berlin, Heidelberg, 2013. Springer-Verlag. ISBN 978-84-695-6971-9.

[12] N. Monzón, J. Sánchez, and A. J. Salgado. Implementation of a robust optical flow method for color images. Technical Report 4, Universidad de Las Palmas de Gran Canaria, September 2014.

[13] H H Nagel and W Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transanctions on Pattern Analysis and Machine Intelligence*, 8:565–593, September 1986. http://dx.doi.org/10.1109/TPAMI.1986.4767833.

[14] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. *arXiv preprint arXiv:1501.02565*, 2015.

[15] J. Sánchez Pérez, E. Meinhardt-Llopis, and G. Facciolo. TV-L1 optical flow estimation. *Image Processing On Line*, 2013:137–150, 2013. http://dx.doi.org/10.5201/ipol.2013.26.

[16] J. Sánchez Pérez, N. Monzón López, and A. Salgado de la Nuez. Robust optical flow estimation. *Image Processing On Line*, 2013:252–270, 2013. http://dx.doi.org/10.5201/ipol.2013.21.

[17] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1385–1392. IEEE, 2013.

[18] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Pattern Recognition*, volume 4713 of *Lecture Notes in Computer Science*, chapter 22, pages 214–223. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. http://dx.doi.org/10.1007/978-3-540-74936-3_22.