Symmetric Optical Flow

L. Alvarez, C.A. Castaño, M. García, K. Krissian, L. Mazorra,
A. Salgado, and J. Sánchez

Departamento de Informática y Sistemas, Universidad de Las Palmas de Gran Canaria, Campus de Tafira s/n, 35017 Las Palmas de Gran Canaria, Spain
Tel.: +34 928 458708, Fax: +34 928 458711
{lalvarez,ccastano,mgarcia,krissian,lmazorra,
asalgado,jsanchez}@dis.ulpgc.es

Abstract. One of the main technique used to recover motion analysis from two images or to register them is variational optical flow, where the pixels of one image are matched to the pixels of the second image by minimizing an energy functional. In the standard formulation of variational optical flow, the estimated motion vector field depends on the reference image and is asymmetric. However, in most application the solution should be independent of the reference image. Only few symmetrical formulations of the optical flow has been proposed in the literature, where the solution is constraint to be symmetric using a combination of the flow in both directions. We propose a new symmetric variational formulation of the optical flow problem, where the flow is naturally symmetric. Results on the Yosemite sequence show an improved accuracy of our symmetric flow with respect to standard optical flow algorithm.

1 Introduction

The problem of motion analysis or registration between two images is an important problem that has been widely addressed in the literature. One of the main technique used to solve this problem is optical flow, where the pixels of one image are matched to the pixels of the second image. Hence, the estimated motion vector field depends on the reference image and is asymmetric. However, in most application the solution should be independent of the reference image. Symmetrical formulations of the optical flow has been proposed in [1,2,3], where the solution is constraint to be symmetric using a combination of the flow in both directions.

In [1], the consistency (or symmetry) of the mapping between two images I_1 and I_2 is enforced by jointly estimating the mapping from I_1 to I_2 and the mapping from I_2 to I_1 and by constraining those two mapping to be inverse to each other. The authors in [2] propose a different approach where the transformation between the two images, denoted T, is obtained by minimizing an energy functional such that $E(I_1, I_2, T) = E(I_2, I_1, T^{-1})$. To achieve this symmetric property, the energy is defined as the average of a non-symmetric energy applied to T and its inverse T^{-1}. Making the hypothesis that $E(T)$ can be described from the derivatives of T, the energy $E(T^{-1})$ is deduced from $E(T)$ without explicitly computing the inverse transformation T^{-1}. This approach is interesting but

R. Moreno-Díaz et al. (Eds.): EUROCAST 2007, LNCS 4739, pp. 676–683, 2007.

leads to numerical difficulties when the derivatives of the transformation T are small. The approach in [3] is similar to the one of [1], but the authors explicitly take into account the discontinuities of the flow and the possible occlusions.

We propose a new symmetric variational formulation of the optical flow problem, where the flow is naturally symmetric. Results on the Yosemite sequence show an improved accuracy of our symmetric flow with respect to standard optical flow algorithm.

2 Formulation of the Problem

In the standard variational optical flow approach, given two images I_1 and I_2, the problem consists in finding a displacement image $\mathbf{u}$, where $I_1(\mathbf{x}) = I_2(\mathbf{x}+\mathbf{u})$. In order to find a displacement between 2 images I_1 and I_2 in a symmetric way, we consider an intermediate image I_m at half way between I_1 and I_2, so that there exists a displacement field $\mathbf{u}$ which fulfills $\forall \mathbf{x}, I_m(\mathbf{x}) = I_1(\mathbf{x} - \frac{\mathbf{u}}{2}) = I_2(\mathbf{x} + \frac{\mathbf{u}}{2})$. We illustrate this approach in Fig. 1.

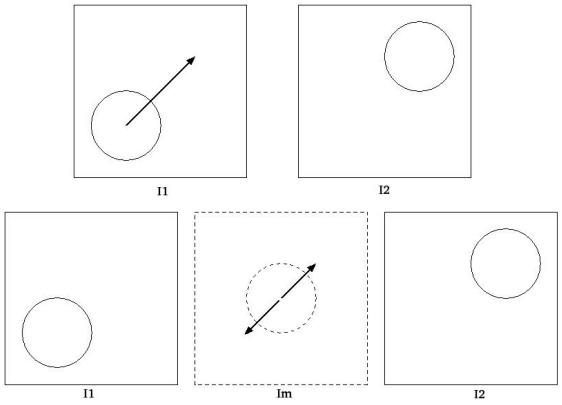

Fig. 1. Illustration of the hypothesis of the symmetric flow

To estimate this displacement, we minimize the energy:

$$E(\mathbf{u}) = \underbrace{\int_{\Omega} \left(I_1(\mathbf{x}^-) - I_2\left(\mathbf{x}^+\right)\right)^2 d\mathbf{x}}_{\text{data term}} + \alpha \underbrace{\int_{\Omega} \|\nabla \mathbf{u}(\mathbf{x})\|^2 d\mathbf{x}}_{\text{regularization term}} , \tag{1}$$

where we denote $\mathbf{x}^+ = \mathbf{u} + \frac{\mathbf{u}}{2}$, $\mathbf{x}^- = \mathbf{x} - \frac{\mathbf{u}}{2}$, and α is a scalar coefficient that weights the regularization (or smoothing) term. Under the assumption of intensity conservation for each voxel, the first term (*data term*) becomes zero when the first image matches the second one : $I_1(\mathbf{x}^-) = I_2(\mathbf{x}^+)$. This term tries to find the vector field that best fits the solution. The second term is a *regularization term* which smooths the vector field. A classification of different regularizers

can be found in [4]. In particular, the authors distinguish between image-driven and flow-driven regularizers and between isotropic and anisotropic ones. In this paper, since we are mainly interested in the symmetrical property of the flow, we simply use the L_2 norm presented above. The coefficient α is normalized to allow invariance under global intensity change of the form $(I_1, I_2) \rightarrow (k\ I_1, k\ I_2)$. To this purpose, α is multiplied by

$$\alpha = \alpha_0 \left(\epsilon + \sqrt{\frac{1}{|\Omega|} \int_\Omega \left\| \frac{\nabla \mathbf{I_1}(\mathbf{x}^-) + \nabla \mathbf{I_2}(\mathbf{x}^+)}{2} \right\|^2 d\mathbf{x}} \right)^2 \tag{2}$$

with $\epsilon = 10^{-3}$.

3 Temporal Regularization

We can use the image sequence information to impose temporal homogeneity and smoothness to the estimated flow. To this end, we re-formulate the equation, considering the whole sequence as a $2D+t$ image $I(\mathbf{y})$ where $\mathbf{y} = (\mathbf{x}, t) = (x, y, t)$. We define $\mathbf{y}^- = \mathbf{y} - \frac{\mathbf{u}(\mathbf{y})}{2}$, and $\mathbf{y}^+ = \mathbf{y} + \frac{\mathbf{u}(\mathbf{y})}{2}$, where the flow $\mathbf{u} = (u, v, dt)^t$ and dt is the constant time between two successive frames of the sequence.

The energy to minimize for the symmetric case is then written as:

$$E(\mathbf{u}) = \underbrace{\int_\Omega \left(I^-(\mathbf{y}) - I^+(\mathbf{y})\right)^2 d\mathbf{y}}_{\text{data term}} + \alpha \underbrace{\int_\Omega (\mathbf{u}_x^2 + \mathbf{u}_y^2) d\mathbf{y}}_{\text{spatial reg.}} + \alpha_t \underbrace{\int_\Omega \mathbf{u}_t^2 d\mathbf{y}}_{\text{temporal reg.}}, \tag{3}$$

where $\mathbf{u}_x^2 = \frac{\partial u}{\partial x}^2 + \frac{\partial v}{\partial x}^2$, $\mathbf{u}_y^2 = \frac{\partial u}{\partial y}^2 + \frac{\partial v}{\partial y}^2$, $\mathbf{u}_t^2 = \frac{\partial u}{\partial t}^2 + \frac{\partial v}{\partial t}^2$ and α_t is a coefficient that weights the temporal regularization. Both coefficients α and α_t are normalized according to (2).

4 Implementation

To minimize the energies previously defined (without and with temporal regularization), we first calculate their gradients and we solve the corresponding Euler-Lagrange equations. In this section, we describe the non-temporal case from which the corresponding implementation of (3) can be easily deduced.

Euler-Lagrange equations from (1) yield:

$$(I_1(\mathbf{x}^-) - I_2(\mathbf{x}^+)).\frac{\nabla I_1(\mathbf{x}^-) + \nabla I_2(\mathbf{x}^+)}{2} + \alpha div(\nabla \mathbf{u}) = 0. \tag{4}$$

In order to linearize this equation, we use an iterative scheme where the displacement field $\mathbf{u}$ is successively estimated from the previous result:

$$\begin{cases} \mathbf{u}^0 & = \mathbf{u_0} \\ \mathbf{u}^{k+1} & = \mathbf{u}^k + \mathbf{h}^{k+1}, \end{cases} \tag{5}$$

where we update the vector field $\mathbf{u}$ at each iteration by adding another vector field $\mathbf{h}$. Supposing that $\mathbf{h}$ is small (which is true in practice since we use a

pyramidal approach) and neglecting the terms in spatial second order derivatives, we can write:

$$I_1(\mathbf{x}^-) \approx I_1(\mathbf{x} - \frac{\mathbf{u}^k}{2}) - \nabla I_1^t(\mathbf{x} - \frac{\mathbf{u}^k}{2})\frac{\mathbf{h}}{2} \tag{6}$$

$$I_2(\mathbf{x}^+) \approx I_2(\mathbf{x} + \frac{\mathbf{u}^k}{2}) + \nabla I_2^t(\mathbf{x} + \frac{\mathbf{u}^k}{2})\frac{\mathbf{h}}{2}. \tag{7}$$

Let us denote

$$\mathbf{g}(\mathbf{x}) = \frac{\nabla I_1(\mathbf{x} - \frac{\mathbf{u}^k}{2}) + \nabla I_2(\mathbf{x} + \frac{\mathbf{u}^k}{2})}{2} \tag{8}$$

$$d(\mathbf{x}) = I_1(\mathbf{x} - \frac{\mathbf{u}^k}{2}) - I_2(\mathbf{x} + \frac{\mathbf{u}^k}{2}). \tag{9}$$

The Euler-Lagrange equation (4) is then written as (at the current location $\mathbf{x}$):

$$(d\mathbf{g} + \alpha\Delta\mathbf{u}^k) - \mathbf{g}\mathbf{g}^t\mathbf{h} + \alpha\Delta\mathbf{h} = 0. \tag{10}$$

After discretization using finite differences, the Laplacian operator $\Delta\mathbf{h}$ can be divided in two terms $-2n\mathbf{h}$ and $S(\mathbf{h})$, where the n is the image dimension. The first term only depends on values of $\mathbf{h}$ at the current position $\mathbf{x}$ and the second term only depends on values of $\mathbf{h}$ at neighbor positions of $\mathbf{x}$: the vector $S(\mathbf{h})$ is written:

$$S(\mathbf{h}) = \begin{pmatrix} \sum_{\mathbf{p}\in N^*(\mathbf{x})} h^x(\mathbf{p}) \\ \sum_{\mathbf{p}\in N^*(\mathbf{x})} h^y(\mathbf{p}) \end{pmatrix}, \tag{11}$$

where $N^*(\mathbf{x})$ denotes the direct neighbors of $\mathbf{x}$ (4 in 2D and 6 in 3D), and $\mathbf{h} = (h^x, h^y)^t$.

Using $\mathbf{h}^{k+1}$ for the current location $\mathbf{x}$ and $\mathbf{h}^k$ for its neighbors, (10) becomes:

$$A\mathbf{h}^{k+1} = b, \tag{12}$$

with $A = \mathbf{g}\mathbf{g}^t + \alpha 2n\, I$, and $b = d\mathbf{g} + \alpha div(\nabla\mathbf{u}^k) + S(\mathbf{h}^k)$. The matrix A is real, symmetric and positive definite, so it can be inverted and we can compute for each position $\mathbf{x}$, $\mathbf{h}^{k+1} = A^{-1}b$. To improve the convergence rate, we use a Gauss-Seidel method which updates the displacement $\mathbf{h}^{k+1}$ at position $\mathbf{x}$ using the values of $\mathbf{h}^{k+1}$ already calculated. This scheme is recursive and to avoid privileging the direction of scanning the image, we apply two successive iterations of Gauss-Seidel in reverse directions. Furthermore, we use a pyramidal approach to compute the displacement flow at several scales, using the results from a given scale to initialize to the following higher scale.

5 Experiments and Results

In order to quantify the accuracy of our symmetric variational optical flow, we use the standard Yosemite sequence. The performance of the algorithm is measured using the angular error as described in [5]. If we consider velocities as 3D vectors were the third component has a constant value of 1, and if we denote

$\mathbf{u}_c = (u_c, v_c, 1)^t$ the correct velocity and $\mathbf{u}_e = (u_u, v_e, 1)^t$ the estimated one, the angular error is defined as:

$$\psi_E = arccos(\frac{\mathbf{u}_c \cdot \mathbf{u}_e}{\|\mathbf{u}_c\| \, \|\mathbf{u}_e\|}). \tag{13}$$

To compare our results with the ground truth, we have to transform the symmetric flow $\mathbf{u}$ into a standard flow $\mathbf{v}$ defined as $I_1(\mathbf{x}) = I_2(\mathbf{x} + \mathbf{v}(\mathbf{x}))$, according to $\mathbf{v}(\mathbf{x} - \frac{\mathbf{u}}{2}) = \mathbf{u}(\mathbf{x})$.

The flow $\mathbf{v}(\mathbf{x})$ is computed as a weighted average of the values of $\mathbf{v}(\mathbf{x} - \frac{\mathbf{u}}{2})$ in the neighborhood of $\mathbf{x}$. This transformation, described in appendix, is similar to the one proposed in [6]. In some sense, we can consider this transformation as a spatial inversion of the flow. To estimate the error introduced by this inversion, we applied it twice to the ground truth and compared the result to the initial ground truth. The angular error obtained between both flow was about 0.3 degrees which can be interpreted as a error of 0.15 degrees for a single inversion. However, this error can be reduced by upsampling the flow image before inversion and downsampling the result.

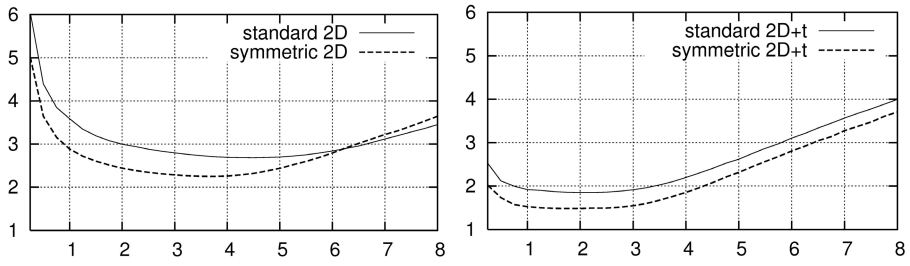

Fig. 2. Angular error obtained on Yosemite sequence, for the standard and the symmetric approaches to variational optical flow. Left: 2D algorithms, right: 2D+t algorithms including a temporal regularization of the flow.

In our experiments, we used the same parameters for the symmetric and the non-symmetric algorithms. The initial images have been smoothed by a Gaussian convolution of standard deviation 0.6 before the estimation, the image intensity at floating-point position was interpolated using a second-order spline interpolation and we used 3 pyramidal scales were the image dimensions were divided by two in X and Y directions from one scale to the next. For the temporal version, the coefficient α_t was set to 2 before its normalization. We applied a series of tests with different values of the regularization parameter α ranging from 0.25 to 8.0 with a step of 0.25. Results for both algorithms are depicted in Fig. 2. Despite the potential error introduced by the inversion of the flow, we observe a better behavior of the symmetrical version of the algorithm. The symmetric version of the optical flow reaches a better result both in the 2D and in the 2D+t cases, with minimal angular errors of 2.25 and 1.48 degrees respectively, while the standard approach reaches minimal angular errors of 2.68 and 1.85 degrees respectively.

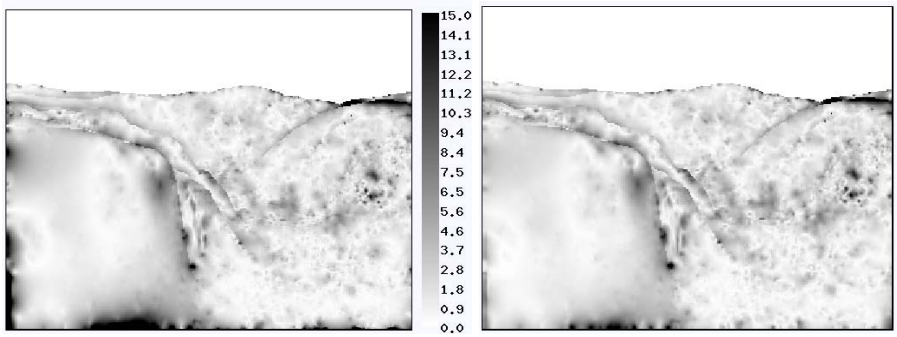

Fig. 3. Angular error obtained on the Yosemite sequence. Left, best result obtained for the 2D standard approach, right, best result obtained for the 2D symmetric approach.

As we can see in Fig. 3, most of the difference between the symmetric and the standard approach lies close to the image boundaries, mainly because the symmetric flow just needs half of the total displacement in each direction instead of the full vector. However, few differences of the angular error appear in the center of the image, where it is difficult of appreciate if an approach is better than the other.

6 Conclusion

In this paper, we proposed a new approach to variational symmetric optical flow. This approach has the advantage of simplicity over previously proposed approaches and it is similar to symmetric approaches used in cross-correlation techniques. We detailed our numerical scheme and we evaluated our approach on the standard Yosemite sequence using both a 2D and a 2D+t regularizations. Results show that the symmetrical version have a better behavior at the image borders leading to a improved mean angular error compared to a standard approach. In our future work, we plan to apply this new symmetric approach to more standard sequences used in optical flow, to experiment different regularizers like the Nagel-Enkelmann regularizer, and to apply this approach in the context of 2D and 3D Particle Image Velocity images.

Acknowledgments

Founded by the European Project FLUID (contract no. FP6-513663).

References

1. Christensen, G., Johnson, H.: Consistent image registration. IEEE Transactions on Medical Imaging 20(7), 568–582 (2001)
2. Cachier, P., Rey, D.: Symmetrization of the non-rigid registration problem using inversion-invariant energies: Application to multiple sclerosis. In: Delp, S.L., DiGoia, A.M., Jaramaz, B. (eds.) MICCAI 2000. LNCS, vol. 1935, pp. 472–481. Springer, Heidelberg (2000)

3. Alvarez, L., Deriche, D., Papadopoulo, T., Sánchez, J.: Symmetrical dense optical flow estimation with occlusions detection. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 721–735. Springer, Heidelberg (2002)
4. Weickert, J., Schnörr, C.: A theoretical framework for convex regularizers in pde-based computation of image motion. International Journal of Computer Vision 45(3), 245–264 (2001)
5. Barron, J., Fleet, D., Beauchemin, S.: Performance of optical flow techniques. IJCV 12(1), 43–77 (1994)
6. Salgado, A., Sánchez, J.: Optical flow estimation with large displacements: A temporal regularizer. Technical report, Instituto Universitario de Ciencias Tecnology

Computing the Backward Flow, v

In this appendix we examine how to compute the flow from I_1 to I_2 from the symmetric flow. The correspondence between both flows is

$$\mathbf{u}(\mathbf{x}) = \mathbf{v}\left(\mathbf{x} - \frac{\mathbf{u}}{2}\right). \tag{14}$$

The main difficulty here is to deal with discrete images. The displacement flow $\mathbf{v}$ that we are looking for takes values at pixel location, but we only have its values at the locations $\mathbf{x} - \frac{\mathbf{u}}{2}$, which are not centered on pixels in general. We consider each pixel as a square. As we can see in Fig. 4 we have to adjust the value of $\mathbf{v}(\mathbf{x})$ depending on the portion of pixels that arrive into the pixel $\mathbf{x}$. In general, there will be several correspondences that distribute their values on a single pixel, so we propose to compute an average of all the portions of the flow that fall into each pixel.

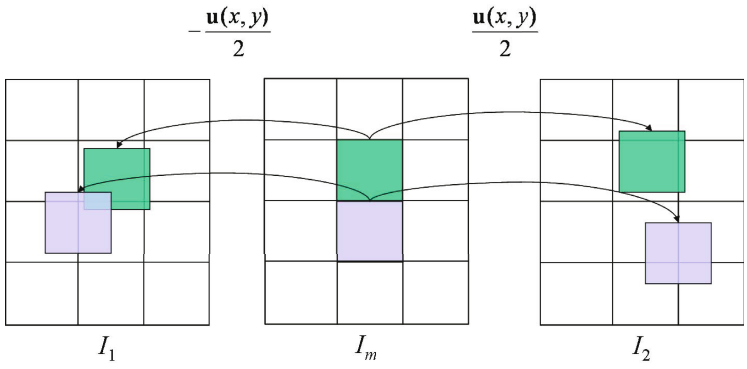

Fig. 4. Estimation the flow from I_1 to I_2 given the symmetric flow. As we can see in this figure, we have to divide each correspondence in four different estimates –four different pixels– in order to compute the values of the discrete function $\mathbf{v}(\mathbf{x})$.

Considering the discrete nature of $\mathbf{v}(\mathbf{x})$ we may compute its value in each position according to this functional

$$\mathbf{v}(\mathbf{x}_i) = -\frac{\sum_{j=1}^{N} \mathbf{u}(\mathbf{x}_j) p_{i,j}\left(\mathbf{x}_i, \mathbf{x}_j - \frac{\mathbf{u}(\mathbf{x}_j)}{2})\right)}{\sum_{j=1}^{N} p_{i,j}\left(\mathbf{x}_i, \mathbf{x}_j - \frac{\mathbf{u}(\mathbf{x}_j)}{2}\right)},$$

where N is the size of the image and $p_{i,j}$ stands for the area of the pixel j that fall into pixel i as can be seen in Fig. 4. Each $\mathbf{x}_j - \frac{\mathbf{u}(\mathbf{x}_j)}{2})$, we generate four different $p_{i,j}$ that will lie on neighbouring pixels. Lets see how we compute these weights. To simplify we call $\mathbf{a} = (a_x, a_y) = \mathbf{x}_i$ and $\mathbf{b} = (b_x, b_y) = \mathbf{x}_j - \frac{\mathbf{u}(\mathbf{x}_j)}{2}$

$$p_{i,j}(\mathbf{a}, \mathbf{b}) = \begin{cases} 0 & if\ max(|a_x - b_x|, |a_y - b_y|) \geq 1 \\ (1 - (b_x - a_x)) \cdot (1 - (b_y - a_y)) & if\ a_x < b_x\ and\ a_y < b_y \\ (1 - (b_x - a_x)) \cdot (1 + b_y - a_y) & if\ a_x < b_x\ and\ a_y > b_y \\ (1 + b_x - a_x) \cdot (1 - (b_y - a_y)) & if\ a_x > b_x\ and\ a_y < b_y \\ (1 + b_x - a_x) \cdot (1 + b_y - a_y) & if\ a_x > b_x\ and\ a_y > b_y \end{cases}.$$

A further issue to consider are the empty pixels,$\mathbf{x}_i$, that have no correspondence in the other image –usually due to occlusions–. Normally these pixels are situated close to the object boundaries in the direction of their displacements. For these reasons we apply a post–processing step in order to fill up the holes. This is a simple step in where after several iterations we complete the information of the holes by averaging with the information from the neighbours.