An Efficient Algorithm for Estimating the Inverse Optical Flow

Javier Sánchez, Agustín Salgado, and Nelson Monzón

Departamento de Informática y Sistemas
University of Las Palmas de Gran Canaria
35017 Las Palmas de Gran Canaria, Spain
{jsanchez,asalgado}dis.ulpgc.es,
nmonzon@ctim.es

Abstract. The aim of this work is to propose a method for computing the inverse optical flow between two frames in a sequence. We assume that the image registration has already been obtained in one direction and we need to estimate the mapping in the opposite direction. The direct and inverse mappings can easily be related through a simple warping formula, which allows us to propose a fast and efficient algorithm. Nevertheless, it is not possible to estimate the inverse function in the whole domain due to the presence of occlusions and disocclusions. Occlusions occur because some objects move to the same position in the next frame. On the other hand, disocclusions are the opposite situation: no correspondences can be established because there is no object moving to those positions. In this case, there is a lack of information and the best we can do is to guess their values from the surrounding values. In the experimental results, we use standard synthetic sequences to study the performance of the proposed method, and show that it yields very accurate solutions.

Keywords: Inverse Optical Flow, Backward Flow, Inverse Mapping, Back Registration, Occlusions, Disocclusions.

1 Introduction

Computing the optical flow is a basic problem in computer vision and image processing. It has many applications in different areas, such as motion analysis, medical imaging, robot guiding, image editing, stereoscopic vision, etc. In this article we address the problem of estimating the inverse optical flow, which is important in problems where it is necessary to find the correspondences back in time.

If we know the optical flow, then it is possible to estimate the backward correspondences with some limitations. In fact, computing the inverse optical flow is a simple problem in the case of a pure bijective transformation. Nevertheless, this is not usual in many applications due to the presence of occlusions and disocclusions. The transformation in these regions is not bijective and it is not

J.M. Sanches, L. Micó, and J.S. Cardoso (Eds.): IbPRIA 2013, LNCS 7887, pp. 390–397, 2013.

possible to estimate the inverse mapping directly. The problem of occlusions may be solved with a relative simple approach and, for a general solution, we need the information of the original images. Disocclusions are more difficult to deal with, since no information is available at these positions.

The use of the backward flow is necessary in several problems. This is the case, for instance, in symmetric optical flow methods, e.g. [2], [5] or [3]. These methods introduce the inverse mapping to improve the coherence between the forward and backward flows, reducing the number of false matchings. The method presented in [5] explicitly computes the inverse mapping using an iterative algorithm. However, this iterative process may slow down the method and it does not work properly in occluded or disoccluded regions. In the symmetric method presented in [4], the inverse optical flow is computed using a Newton scheme. This solution is similar to the previous iterative process, so it presents the same drawbacks, providing poor results in occluded and disoccluded regions. Another interesting symmetric model is proposed in [1]. In this case, the optical flow is computed as a function in the middle position between two frames, so it does not compute the inverse optical flow explicitly.

More recently, the backward flow has been used in temporal optical flow methods, e.g. [8] or [9]. The objective is to preserve the temporal coherence of the optical flows with the previous estimated flows: the inverse mapping is used to find the correspondences back in time and impose some kind of temporal continuity. Another example of the use of the backward flow is given in [6] and [7]. In this case, the authors propose a method that relies on the reverse optical flow to automatically follow the road in autonomous cars. It tracks features from the current position to a past position, so the robot may identify similar shapes at different scales.

In this work we propose a new method for estimating the backward flow directly from the optical flow and the original frames. We suppose that the optical flow has already been computed and we are interested in computing the backward flow with high precision. The proposed algorithm is simple, fast and accurate. It automatically handles occlusions, whereas disocclusions are filled using a min-fill strategy, computing the minimum flow value within a given distance. In the experimental results, we study the precision of our method using synthetic sequences from the Middlebury database and show that the accuracy of this method is high.

In Sect. 2 we explain the basis for estimating the backward flow. The algorithm is presented in Sect. 3. In the experimental results, Sect. 4, we evaluate the algorithm using several sequences from the Middlebury benchmark datasets. Finally the conclusions in Sect. 5.

2 General Framework

Let $\mathbf{h}(\mathbf{x}) = (u(\mathbf{x}), v(\mathbf{x}))$ be a vector field that establishes the correspondences between the pixels of two images. In general, $\mathbf{h}(\mathbf{x})$ is said to be *dense* as it is a mapping of every pixel in the first image to the pixels in the second image.

We define the backward flow, $\mathbf{h}^*(\mathbf{x}) = (u^*(\mathbf{x}), v^*(\mathbf{x}))$, as the inverse mapping of $\mathbf{h}(\mathbf{x})$. $\mathbf{h}^*$ puts in correspondence the pixels in the second image with the pixels in the first image. The forward and backward flows can be related as:

$$\mathbf{h}(\mathbf{x}) = -\mathbf{h}^*(\mathbf{x} + \mathbf{h}(\mathbf{x})). \tag{1}$$

The corresponding positions for both functions are given by the warping of the forward function, $\mathbf{x} \rightarrow \mathbf{x} + \mathbf{h}(\mathbf{x})$. This relation can be intuitively derived from the graphic depicted in Fig. 1.

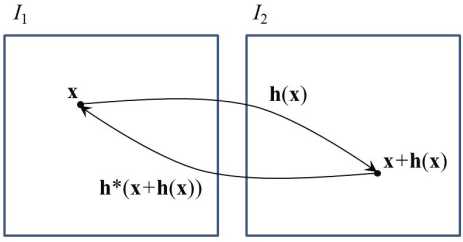

Fig. 1. Backward flow

Typically, $\mathbf{h}(\mathbf{x})$ is not bijective and it is not possible to estimate $\mathbf{h}^*(\mathbf{x})$. This is true, for instance, in occluded and disoccluded regions: occlusions occur when several correspondences map to the same position, yielding a function which is not injective; and disocclusions accounts for locations with no correspondence, representing a non-surjective function.

Since occlusions and disocclusions are typical in most sequences, $\mathbf{h}$ is in general not invertible. Thus, we have to reason about these two problems in order to find dense inverse maps. They are easy to detect but their solutions have to be overcome from different perspectives. Looking at Fig. 2, we observe that occlusions appear in the direction of the moving objects and disocclusions appear in the opposite side.

Occlusions may happen when moving objects occlude other slower moving objects, like depicted in Fig. 2. On the other hand, occlusions may also occur when moving objects displace behind other objects, like in Fig. 3. The former situation is typical in fronto-parallel stereoscopic sequences and it can be efficiently solved, like proposed in [9], so we will refer this situation as the *stereoscopic occlusion*. The latter is typical in general optical flow problems.

We call this last situation the *street-lamp occlusion*, as it typically appears in traffic sequences where a car moves behind a street lamp. Looking at Fig. 3, we observe that the occlusion is not only present in the front of the square but also inside the square, since these pixels fall exactly behind the bar in the second image. The effect of this occlusion is like projecting the bar into the square at a distant equivalent to its motion.

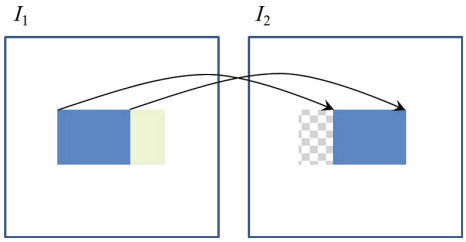

Fig. 2. Occlusions and disocclusions. When the blue square moves horizontally, it creates a disocclusion and occlusion before and after the square, respectively.

The problem of occlusions can be solved by means of a selection process, where several values are associated with the same position, and we need to select the appropriate one. The selection problem may only depend on the vector field itself, as it happends in the stereoscopic occlusion. In this case, we select the value corresponding to the biggest motion. On the other hand, the more general case of the street-lamp occlusion requires not only the values of the optical flow but also the image intensities: we have to select the motion associated with the pixel which is finally visible in the second image. The disocclusion problem is more difficult, since there is no information available for a given position. It can be addressed as a filling procedure, selecting the information which is in the neighborhood of the pixel.

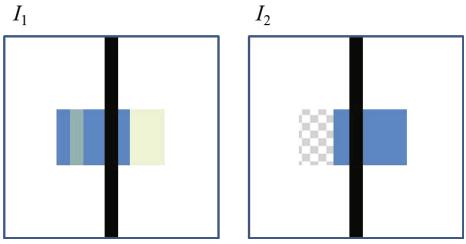

Fig. 3. *Street-lamp Occlusion*: the blue square moves behind the static central bar. Occlusions appear in the front of the square and also inside the square, induced by the static bar.

3 Computing the Inverse Optical Flow

The purpose of our method is to compute a dense backward flow from the information of the forward flow and the image intensities. In this section we will deal with the more general street lamp occlusion case.

Algorithm 1. Inverse Optical Flow

Input: I_1, I_2, u, v
Output: u^*, v^*

Create a $buffer$ the same size as the images
Initialize each position of the $buffer$ to a big number

for $y \leftarrow 1$ **to** $size_y$ **do**
  for $x \leftarrow 1$ **to** $size_x$ **do**
    $x_w \leftarrow x + u(x, y)$
    $y_w \leftarrow y + v(x, y)$
    Find the four neighbors around $(x_w, y_w) : (x^{+/-}, y^{+/-})$
    Compute the weights: w_1, w_2, w_3, w_4 (see Fig. 4)
    $d_1 \leftarrow |I_1(x, y) - I_2(x^-, y^-)|^2$
    $d_2 \leftarrow |I_1(x, y) - I_2(x^+, y^-)|^2$
    $d_3 \leftarrow |I_1(x, y) - I_2(x^-, y^+)|^2$
    $d_4 \leftarrow |I_1(x, y) - I_2(x^+, y^+)|^2$
    if $w_1 \geq 0,25$ ***and*** $buffer(x^-, y^-) \geq d_1$ **then**
      $u^*(x^-, y^-) \leftarrow -u(x, y)$
      $v^*(x^-, y^-) \leftarrow -v(x, y)$
      $buffer(x^-, y^-) \leftarrow d_1$
    end
    if $w_2 \geq 0,25$ ***and*** $buffer(x^+, y^-) \geq d_2$ **then**
      $u^*(x^+, y^-) \leftarrow -u(x, y)$
      $v^*(x^+, y^-) \leftarrow -v(x, y)$
      $buffer(x^+, y^-) \leftarrow d_2$
    end
    if $w_3 \geq 0,25$ ***and*** $buffer(x^-, y^+) \geq d_3$ **then**
      $u^*(x^-, y^+) \leftarrow -u(x, y)$
      $v^*(x^-, y^+) \leftarrow -v(x, y)$
      $buffer(x^-, y^+) \leftarrow d_3$
    end
    if $w_4 \geq 0,25$ ***and*** $buffer(x^+, y^+) \geq d_4$ **then**
      $u^*(x^+, y^+) \leftarrow -u(x, y)$
      $v^*(x^+, y^+) \leftarrow -v(x, y)$
      $buffer(x^+, y^+) \leftarrow d_4$
    end
  end
end
Fill disocclusions

Algorithm 1 shows the steps to compute the inverse optical flow. The input data is the forward flow and both images. At the beginning, we initialize the

backward flow to a big number. Then, the algorithm goes over each position of the vector field, carrying out the following steps: first, it finds the corresponding position in the second image, using the vector field; it obtains the four neighbors around this float position, given by $(x^{+/-}, y^{+/-})$; then, it computes the pixel area weights, w_1, w_2, w_3, w_4, as shown in Fig. 4 (these weights represent the proportional area of the pixel that corresponds to each neighbor); it estimates the similarity between the images at the given positions; finally, it assigns the negative value of the original flow to each of the neighbors if it has the best similarity and lies close to the destiny pixel. In the last step, the algorithm fills disocclusions. Note that similarities are stored in a $buffer$, so that when there are several pixels that arrive to the same position, i.e., in the situation of occlusions, the algorithm retains the value corresponding to the most similar pixels in both images. The thresholds, w_1, w_2, w_3, w_4, must be greater or equal than 0,25 to assure that the correspondence is near the destiny pixels. The use of the $buffer$ allows us to automatically deal with occlusions. This algorithm is simple and very fast: only one pass over the image is necessary to compute the backward flow. In order to fill disocclusions, we assume that the motion at disocclusions corresponds to the minimum motion in the region. We look for the minimum value in a window around the current position. This process is carried out iteratively until every disocclusion is filled.

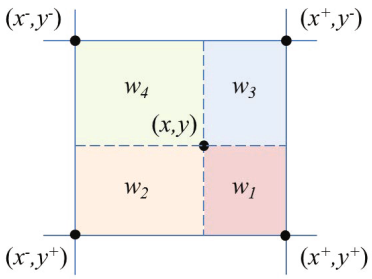

Fig. 4. Notation

4 Experimental Results

In order to evaluate the method, we use a *reprojection error* by computing the backward flow twice, $(\mathbf{h}^*)^*$, and then we compare with the original ground truth using the average End-point Error (EPE). Figure 5 depicts the results using the Middlebury benchmark database. We have used the test sequences for which the ground truths are known. The reprojection errors are shown in Table 1. In the third column of Fig. 5, we show the computed backward flow with disocclusions highlighted in white. We observe that disoccluded regions are very large in some examples. For instance, in the penultimate sequence there is a large disocclusion in the border of the paper. In Table 1, we observe that the errors are very small.

For the last experiment, we modify the Urban2 sequence by introducing two static bars, as shown in the last row of Fig. 5. This allows us to study the behavior of the method with respect to the street-lamp occlusion type. In the last column of Table 1, we show its reprojection error.

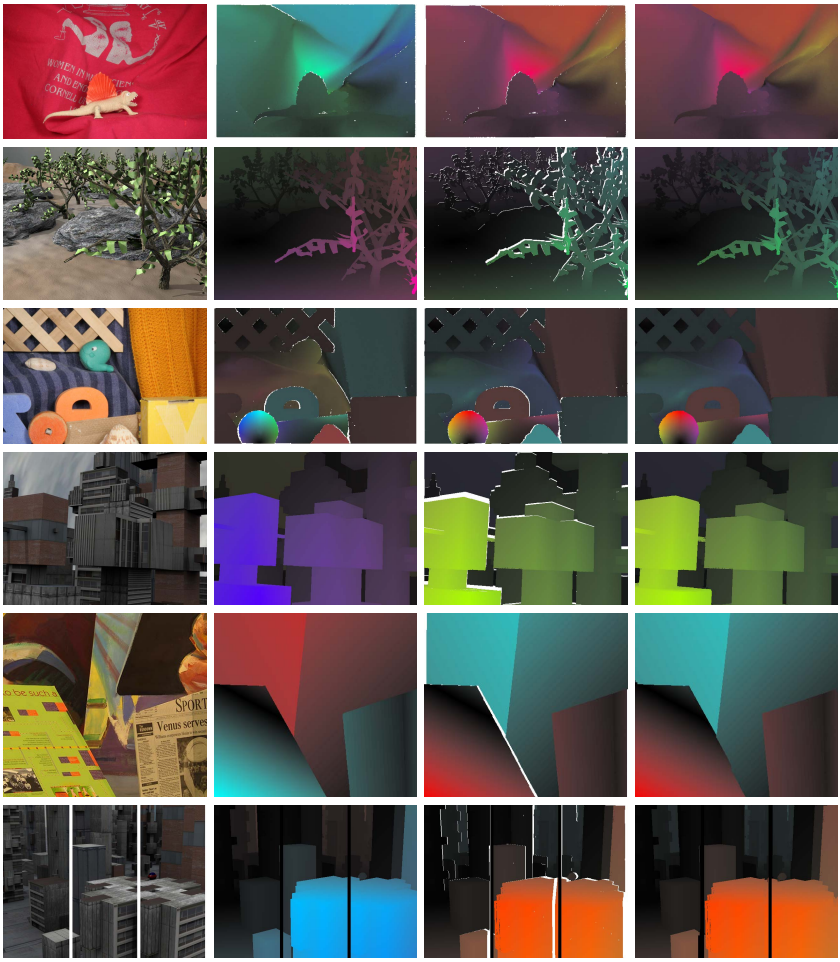

Fig. 5. Middlebury test sequences. First column, the source image; second, the ground truth; third, the backward flow (disocclusions in white color); fourth, the inverse flow.

Table 1. Middlebury backward flow errors

Sequence	Dimetrodon	Grove3	RubberWhale	Urban3	Venus	Urban2 with bars
Reproj. error	0.007	0.140	0.006	0.091	0.032	0.120

5 Conclusions

In this work, we have proposed a very accurate method for estimating the inverse optical flow. This method relies on the optical flow and two frames in a sequence. It automatically deals with general image sequences, coping with any type of occlusions. In order to fill dissoclusions, we have proposed a filling strategy based on the minimum flow. The algorithm is very efficient, since it only carries out one pass over the image. The storage requirements are also very low and only an intermediate buffer is necessary to account for the occlusions.

Acknowledgment. This work has been partly founded by the Spanish Ministry of Science and Innovation through the research project TIN2011-25488.

References

1. Alvarez, L., Castaño, C.A., García, M., Krissian, K., Mazorra, L., Salgado, A., Sánchez, J.: Symmetric Optical Flow. In: Moreno Díaz, R., Pichler, F., Quesada Arencibia, A. (eds.) EUROCAST 2007. LNCS, vol. 4739, pp. 676–683. Springer, Heidelberg (2007)
2. Álvarez, L., Deriche, R., Papadopoulo, T., Sánchez, J.: Symmetrical dense optical flow estimation with occlusions detection. International Journal of Computer Vision 75(3), 371–385 (2007)
3. Ashburner, J.: A fast diffeomorphic image registration algorithm. NeuroImage 38(1), 95–113 (2007)
4. 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)
5. Christensen, G.E., Johnson, H.J.: Consistent image registration. IEEE Transactions on Medical Imaging 20(7), 568–582 (2001)
6. Lieb, D., Lookingbill, A., Thrun, S.: Adaptive road following using self-supervised learning and reverse optical flow. In: Proceedings of Robotics: Science and Systems, Cambridge, USA (June 2005)
7. Lookingbill, A., Rogers, J., Lieb, D., Curry, J., Thrun, S.: Reverse optical flow for self-supervised adaptive autonomous robot navigation. International Journal of Computer Vision 74, 287–302 (2007)
8. Salgado, A., Sánchez, J.: A temporal regularizer for large optical flow estimation. In: IEEE International Conference on Image Processing, ICIP, pp. 1233–1236 (2006)
9. Sánchez, J., Salgado, A., Monzón, N.: Direct estimation of the backward flow. In: 8th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Application (VISAPP), vol. 2, pp. 268–274. INSTICC (2013) ISBN:978-989-8565-47-1