Mode based hierarchical optical flow estimation.

J. Otero (jotero@lsi.uniovi.es), A. Otero and L. Sánchez
Oviedo University, Computer Science Department ,33204 Gijón, Asturias, Spain

Abstract. In this paper a robust estimation of the optical flow field that preserves the boundaries of the movement is shown. Arising from the techniques based on the Optical Flow Constraint (OFC), an estimation that takes several measures around a given pixel, discarding the erroneous ones, has been developed. This is done performing a bidimensional clustering of the velocities obtained from the intersection of couples of OFCs. In this way the clustering is done in the velocity space and not in the (slope, intercept) parameter space of the OFCs. Finally, a hierarchical implementation that has less error, when large displacements are present in the image, is shown.

Key words: Computer vision, motion analysis, optical flow, robust estimation

1. INTRODUCTION

There are basically three ways to perform the calculation of the optical flow field: correlation based techniques, frequency based techniques and gradient based techniques.

Correlation based techniques [1] try to maximize a measure of similarity between patches (taken from two consecutive frames) centered in a given pixel. The displacement that maximizes the selected measure divided by the time interval between the acquisition of the frames is the velocity of the pixel. These approaches are computationally expensive, except those that use custom VLSI chips.

Frequency based techniques use a set of tuned spatiotemporal filters to search for the velocity of a pixel [7]. Other researchers [4] are in the opinion that this is the most precise approach, but it is very expensive in terms of computational cost too.

Gradient based techniques use the well known "optical flow constraint" (OFC) shown in equation 1 in order to compute the optical flow [8].

$$-\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} = \frac{\partial f}{\partial x}u + \frac{\partial f}{\partial y}v = \nabla(f) \cdot \overrightarrow{c} \qquad (1)$$

This equation makes the assumption that intensity changes in a sequence of images are due only to the movement of the objects in the scene: a single pixel will have constant brightness in the different positions that it takes during the sequence. Unfortunately, the "aperture problem" (see [9]) states that there is no way to recover the complete optical flow vector using only local (one pixel) information. Some authors try to solve the aperture problem with the incorporation of some kind of global information, involving a process of regularization [8]. Some researchers perform a clustering of the OFCs themselves in order to find the most reliable one. Once obtained, the corresponding

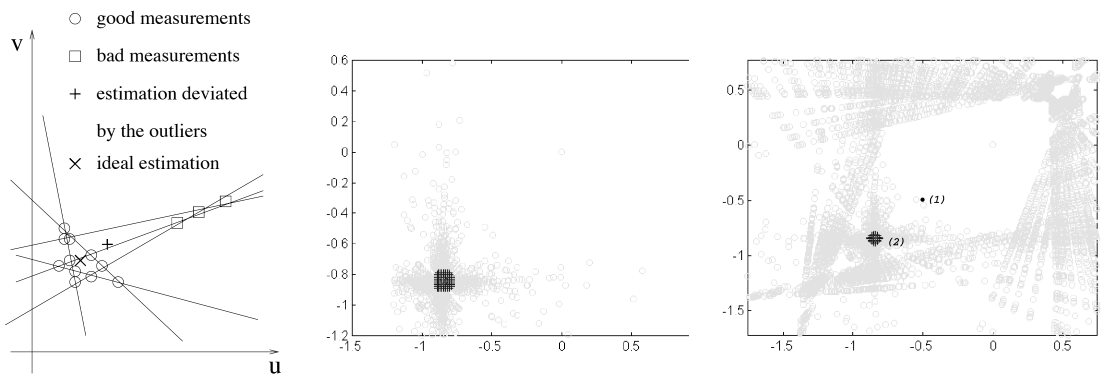

Fig. 1. Left: Clusters of intersections of OFCs pairs. Most of them belong to a dominant cluster, but a few ones can deviate the estimation if they are not rejected. Center and Right: Intersections of OFCs taken from real images: systems obtained from pixels with similar velocity (left) are grouped together in a well defined cluster. When two velocities are present (right) the least squares deviated estimation (1) is far from the dominant cluster (2).

normal flow to that OFC is obtained [12] [11]. Another alternative is to analyze the measurements in the space of the velocities, that is, performing an estimation of the velocity with the results of many systems of OFC equations. Each system of equations is obtained from one pair of pixels (see figure 1) or fitting the data to a model in order to estimate the velocity. In this way, the analysis is performed directly in the domain of the data that we want to recover, that is, the u, v space [14]. In this paper the last approach is followed.

2. ROBUST ESTIMATION OF OPTICAL FLOW

Usually, in real sequences of images, there are many independent moving objects. In this situation, the problem of occluding surfaces arises: places where the velocity changes suddenly from one pixel to another. In this case it is not possible to use approaches like [8] because the global smoothness criterion does not hold in those boundaries. Other authors [10] propose to relax the smoothness criterion in those places where the gradient is high, smoothing the flow along the contour but not through the contour. But the problem remains: the OFC has two unknowns, it is necessary to take measurements from at least two pixels. If the pixels are chosen from objects with different velocities, the solution to the system of equations obtained will give an erroneous velocity. The analysis of the velocity distributions presented in a neighborhood of each pixel (in the direction addressed in [14]) is necessary to determine which velocities are supposed to be correct and which one is the dominant velocity in the neighborhood.

The idea behind our approach could be summarized on three steps:

1. For each pixel we consider a neighborhood of a given size. For each possible couple of pixels, we solve the system of equations given by the OFC applied to the couple of pixels, discarding solutions whose module exceeds a given constant as errors. We discard also the solutions of systems whose condition number is too high.
2. After this, we compute the bidimensional histogram of the velocity distribution. We filter the data with a Gaussian kernel in order to erode the isolated peaks.
3. The velocity that corresponds to the maximum is an estimation by itself but, to improve the precision, we perform another estimation: for every data that is closer to the maximum than a given percent of it, we search for the gravity center of them. This is our final estimation.

Note that in [14] all couples of used OFCs share one component with the central pixel of the neighborhood. In that way, the clustering is performed along the OFC corresponding to the central pixel, reducing the complexity of the estimation but making the quality of the estimation strongly dependent on how reliable the data on a single pixel is. This situation is shown in figure 2 . If the clustering is performed along the central pixel OFC, the estimation is deviated from the cluster obtained from the intersection of all possible pairs of OFCs. Since there is no "a priori" information on how reliable a given OFC is, to depend on a single OFC does not seem to be convenient. Another approach [11] consists on finding the most reliable OFC and obtaining an optical flow estimation from that single line, without performing any intersection between pairs of constraints. This is done by performing the clustering in the $(slope, intercept)$ parameter space of the OFCs, using Combinatorial Hough Transform (see figure 3). This is achieved by estimating the distribution of two parameters that defines the characteristic line along most of the pairs (slope, intercept) lie. One of the parameters is obtained from the (slope,intercept) pair corresponding to the central pixel with the same data from the other pixels in the neighborhood. The other one is obtained from the first parameter with the data of each pixel of the neighborhood. With these parameters, an estimation of optical flow is obtained directly.

In the previous approaches, the amount of data is $c^2 - 1$, being $c \times c$ the size of the neighborhood. In our approach the amount of data is the number of possible pairs of OFCs in the neighborhood, that is $\frac{c^2}{2}\left(c^2 - 1\right)$. This means that if we perform an estimation with the same amount of data, our approach uses sets of pixels that are more dense than the ones used in the other approaches. For instance, in a neighborhood of 15×15, 224 measurements are made if the clustering is done along the central pixel OFC. With a neighborhood of 5×5 our approach performs 300 measurements. Because of this, the basic assumption of these methods, the closer the pixels are, more similar the velocities are, holds better in our approach.

In this work we use all the possible couples of OFCs because we believe that if the assumption in [14] holds (pixels that are near in space have similar velocities) then it

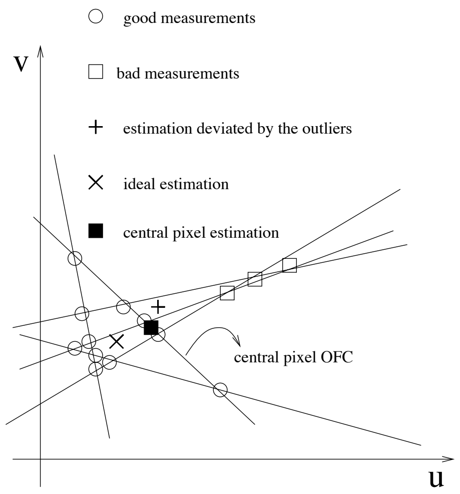

Fig. 2. Clustering of intersections of OFCs along central pixel OFC. If this is deviated, the estimation will fail.

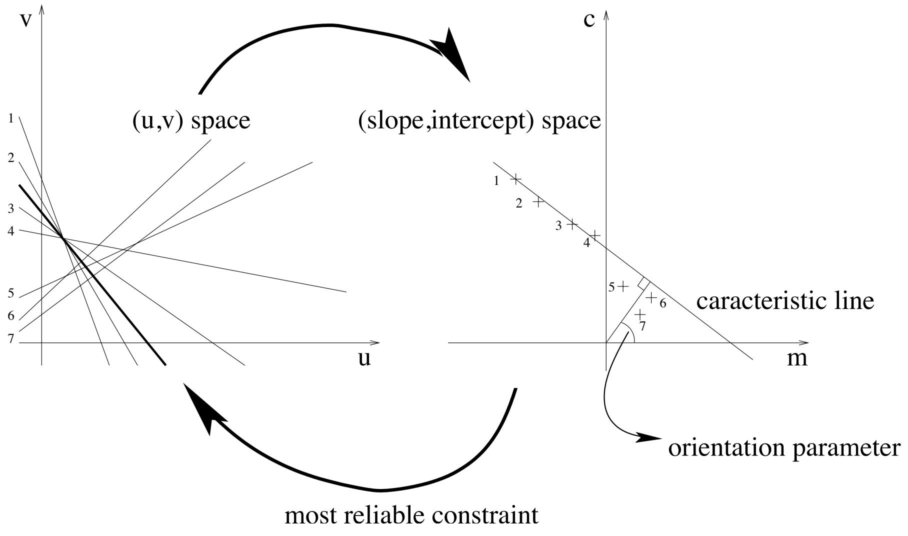

Fig. 3. Clustering of (slope,intercept) space parameters of OFC.

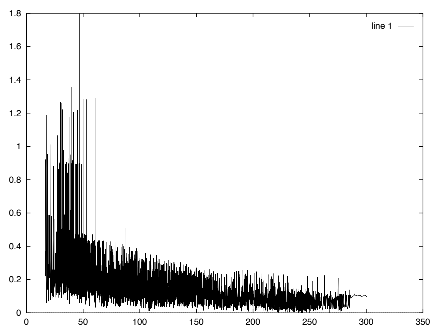

Fig. 4. Plot of the density value (x axis) versus the error of the estimation (y axis) as defined in [13] and later in this paper. As it can be seen, there is a decreasing tendency in the figure.

must be applied to every pixel in the neighborhood (the estimation must be obtained from every possible pair of OFCs in the neighborhood, not only the ones with the central pixel.) We suggest performing a bidimensional clustering in the (u, v) space, using all the possible intersections of OFCs that belong to a neighborhood of a pixel.

It is clear that the straight application of the enumerated steps would be computationally expensive. But, if we realize that the mentioned steps are no more than an estimation of the *mode* of the velocity field, we can optimize our algorithm by replacing the vote scheme with the numerical search of the maximum of a estimation of the density function of the velocity field, thus avoiding the explicit computation of the histogram.

We have analyzed two different methods to perform the mode estimation: (1) k-neighborhood estimators and (2) a compact support kernel. In both cases it is possible to reduce the computations as follows: after the first step, the solutions to the system of OFCs are stored into an array with three columns: the components of the velocity (x and y), and the number of votes. The key point here is to *order* the array: If we use, for instance, a pyramidal kernel with a square basis of size "d", we can compute the value of the density function in a point stored in row "i", (u_i, v_i) by evaluating the kernel function in the points $(u_{i-1}, v_{i-1}), (u_{i-2}, v_{i-2}), \ldots$ and $(u_{i+1}, v_{v+1}), (u_{i+2}, v_{i+2}), \ldots$ for which $|u_j - u_i| < d/2$. For the k-neighborhood method, the reasoning is similar. The method returns the velocity of the point in the array in which the value of the density function is maximum, which is a good approximation to the true point in which the density function is maximum. The value of the density is also useful, because it is a measure of the reliability of the estimation. For example, in Figure 4 the mode density value (x axis) is plotted versus the error of an experiment (y axis) for all points in the sequence "MysineC-16". Clearly it can be seen that, as the value of the density grows, the error decreases.

3. HIERARCHICAL ESTIMATION OF OPTICAL FLOW

In equation 1 it is shown how the velocity is related with the spatiotemporal derivatives. Due to the use of this equation, the spatiotemporal derivatives must be estimated in every one of the pixels in the sequence of images from which we want to compute the optical flow field. In [8] the derivatives are computed from the data in a spatiotemporal neighborhood of size $2 \times 2 \times 2$

In the following, let i and j stand for rows and columns of each frame, and k index for frame number.

If the displacement of a given pixel is such that from frame to frame it falls out of the neighborhood like the one in [8] then we can see an inconsistency: we are capable of computing the derivatives without knowing the values of the involved pixels in the second frame. If we obtain a displacement for the pixel in (i, j) of $(2, 2)$, then it falls out of the neighborhood and so, the derivatives were calculated with values that are not related with the pixels in the first frame.

It is necessary to develop a method to overcome the problem of derivatives estimation when the displacements are bigger than the neighborhood size. This leads us to a hierarchical implementation [5][1][6] of our algorithm. A hierarchical approach consists of two stages. First, the computations are performed with a zoomed out version of the images, that is a coarser resolution version. Then the results at this stage are extrapolated to a finer resolution. In this way we have a coarse estimation of the optical flow at this new resolution. This can be done with the use of equations 2 and 3, for an image of size $m \times n$ for each $i, j / i \in [1, \frac{m}{2}] \wedge j \in [1, \frac{n}{2}]$ we have:

$$\left.\begin{array}{l} U_1((i-1)*2,(j-1)*2) \\ U_1((i-1)*2+1,(j-1)*2) \\ U_1((i-1)*2,(j-1)*2+1) \\ U_1((i-1)*2+1,(j-1)*2+1) \end{array}\right\} = 2 * U_2(i,j) \tag{2}$$

$$\left.\begin{array}{l} V_1((i-1)*2,(j-1)*2) \\ V_1((i-1)*2+1,(j-1)*2) \\ V_1((i-1)*2,(j-1)*2+1) \\ V_1((i-1)*2+1,(j-1)*2+1) \end{array}\right\} = 2 * V_2(i,j) \tag{3}$$

where U_2 and V_2 are the velocities obtained from the sequence with a given resolution and U_1 and V_1 are the coarse estimation of the velocities with twice the resolution. The key idea is that we can find a scale where the displacement is lower than the size of the neighborhood, so at that resolution the derivatives estimation can be performed without the commented inconsistency. This is, if we use a neighborhood of size $(2l+1) \times (2l+1)$ with l in $1, 2, 3, \ldots$ and the maximum displacement is Δ in a $m \times n$ image ($m = 2^a$ and $n = 2^b$ for simplicity) then there exists c such that $c < a$ and $c < b$, c in $1, 2, 3, \ldots$ such that if we reduce the size of the image to $\frac{m}{2^c} \times \frac{n}{2^c}$ then at that scale the maximum displacement is δ, with $\delta = \frac{\Delta}{2^c} < l$ and then, the mentioned inconsistency does not happen. With the full data at higher resolution the coarse estimation is refined, in order to make it a better approximation of the optical flow field. This is achieved using the coarse estimation to perform a better estimation of spatiotemporal derivatives, getting the equations 4,5,6, where (i, j, k) is the current pixel.The estimation is performed using a neighborhood of size $(2l + 1) \times (2l + 1)$. k stands for frame number and the true displacement of that pixel is (D_u, D_v) such that $D_u > l$ and $D_v > l$. The estimation at lower resolution is (d_u, d_v) and the coarse estimation of the movement of the pixel is $(2d_u, 2d_v)$. Let be (di_u, di_v) the integer part of the coarse estimation, this is the displacement that must be taken into account to compute the derivatives in equations 4,5 and 6. The relationship between the sign of x and y axis and the increment/decrement of rows and columns is as follows, x positive axis is in the direction of increasing columns and y positive axis is in the direction of decreasing rows.

$$\begin{array}{l}\frac{\partial I}{\partial x}(i,j,k) = \\ \frac{1}{\delta x}(I(i+1,j,k)+ \\ I(i+1-di_v, j+di_u, k+1)+ \\ I(i+1,j+1,k) + I(i+1-di_v, j+1+di_u, k+1))- \\ \frac{1}{\delta x}(I(i,j,k) + I(i-di_v, j+di_u, k+1)+ \\ I(i,j+1,k) + I(i-di_v, j+1+di_u, k+1))\end{array} \tag{4}$$

$$\begin{array}{l}\frac{\partial I}{\partial y}(i,j,k) = \\ \frac{1}{\delta y}(I(i,j+1,k)+ \\ I(i-di_v, j+1+di_u, k+1)+ \\ I(i+1,j+1,k) + I(i+1-di_v, j+1+di_u, k+1))- \\ \frac{1}{\delta y}(I(i,j,k) + I(i-di_v, j+di_u, k+1)+ \\ I(i+1,j,k) + I(i+1-di_v, j+di_u, k+1))\end{array} \tag{5}$$

$$\begin{array}{l}\frac{\partial I}{\partial t}(i,j,k) = \\ \frac{1}{\delta t}(I(i-di_v, j+di_u, k+1)+ \\ I(i-di_v, j+1+di_u, k+1)+ \\ I(i+1-di_v, j+di_u, k+1) + I(i+1-di_v, j+1+di_u, k+1))- \\ \frac{1}{\delta t}(I(i,j,k) + I(i,j+1,k)+ \\ I(i+1,j,k) + I(i+1,j+1,k))\end{array} \tag{6}$$

As we can repeat this procedure at any resolution, a recursive formulation of the algorithm is suggested, as it is shown in the following pseudo code:

```
hierarchical_of(sequence) is
if size (sequence) < threshold then
  of=of_simple(sequence)
```

```
else
   sequence2=reduce(sequence)
   of=hierarchical_of(sequence2)
   of=extended(of)
   of=refined(of)
end if
```

where `of_simple(sequence)` is a non hierarchical estimation of optical flow, `extended(of)` corresponds to equations 2 and 3 and `refined(of)` corresponds to an optical flow estimation using equations 4, 5 and 6 to compute the spatiotemporal derivatives. That is, the hierarchical optical flow calculation consists of the following: if the size of the image is smaller than a threshold, then a simple (one stage) calculation of the optical flow is performed. If the size of the image is bigger than the same threshold, then the sequence is reduced in size to half of the rows and columns. At this point, `hierarchical_of` is called recursively. When the algorithm comes back from this call, the optical flow obtained is extended to the actual size of the image using 2 and 3, and after that it is refined at this level of resolution using 4, 5 and 6.

4. RESULTS

4.1. Qualitative results

In top left corner of figure 6 it is shown the optical flow field (obtained using a least squares (L.S.) estimation over a $9x9$ neighborhood) from the sequence of images in figure 5, where two sinusoidal patterns move with velocities $(1, -1)$ (pattern in the left) and $(-1, 1)$ (pattern in the right).

There are errors and the velocities seem (erroneously) to change smoothly. The same occurs in the top and right corner of the figure, where the results of the approach in [11], OFC Parameter Clustering, are shown. In the left down corner of figure 6 the results for the algorithm in [14] (named Schunck) are shown. Finally in the right and down corner of the same figure the same experiment is performed with the algorithm presented here (named Kernel).

The boundaries of the movement are better preserved due to the robustness of the estimator and because the window used in the estimation can be made smaller than in the other techniques that perform the clustering with central pixel data for the same amount of measurements.

The estimation here presented fails in two cases. First when the central pixel of the neighborhood lies on a surface that moves in a different way than most of the other pixels in the neighborhood. This happens in 90 degrees corners, for instance. In this case the dominant cluster belongs to a different object than the central pixel. Second, if the central pixel of the neighborhood belongs to one surface in first frame and to another,

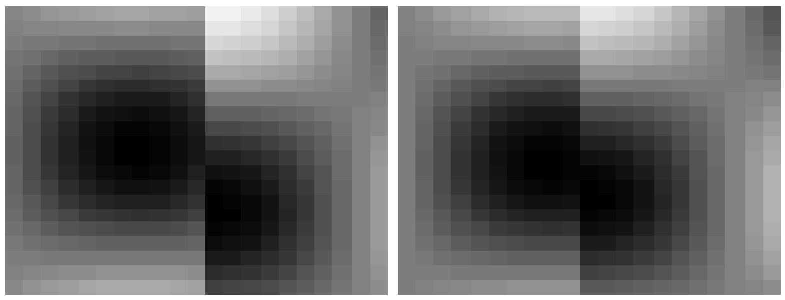

Fig. 5. Two frames of the test sequence used to show how our algorithm preserves the boundaries of the movement.

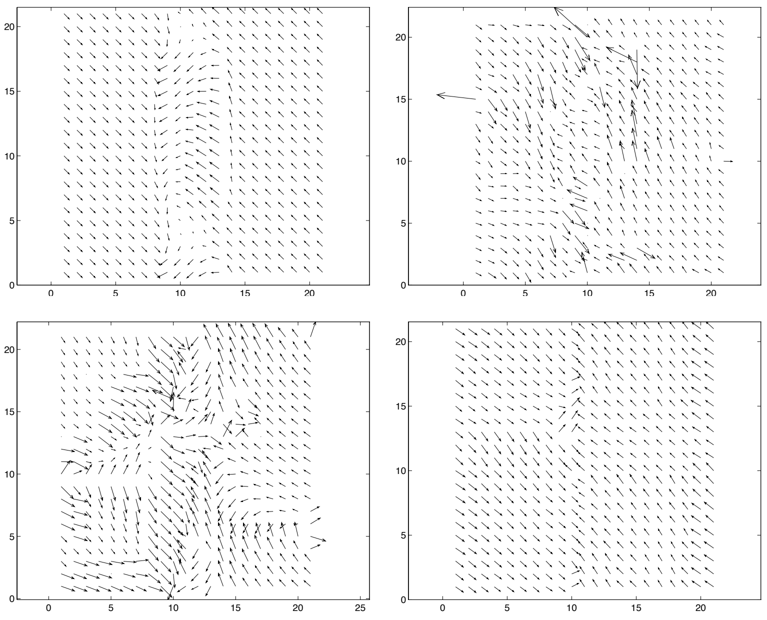

Fig. 6. In the right and down corner of this figure the optical flow obtained with our method is shown. From left to right and up to down are the results for L.S. estimation, the OFC Parameter Clustering algorithm and Schunck algorithm are shown. This test was especially designed to show how our approach preserves the boundaries of the movement.

occluded by the former, in the second frame.

Apart from these situations, the estimation of the velocity is much better than the one obtained from L.S. estimation and other techniques based on a regularization constraint

[8]. The reason is that our approach discards the measures obtained from couples of pixels that have different velocities or have OFCs almost parallel (so the solution of the system is not reliable.)

4.2. Quantitative results, non hierarchical algorithm.

We will define the error as the sum of the modulus of the differences between correct and obtained velocities as in [13]. The test sequences used in these series of test were " Translating tree", "Diverging tree", "Yosemite flight through", "Rotating sphere", "Diverging office" and "Street". The first three are the well known sequences used in [3] to test the performance of various algorithms. The sequences are semi-synthetic, real images manipulated to give the illusion of movement in the scene or by the camera. The last three were proposed in [2] as an alternative to perform the same task. The "Rotating sphere" sequence shows a rotating sphere over a static sinusoidal pattern, the "Diverging office" shows a view of an office, the camera moves towards the monitor of a computer. The sequence "Street" shows an outdoor scene. We choose this sequences to perform the analysis published in [3]. The source code used in the test of Anandan's algorithm is the same used in [3]. To test Schunck algorithm [14] and OFC Parameter Clustering [11] we used our own implementation. In table 1 numerical values of the error are compared to Anandan algorithm [1], OFC Parameter Clustering [11] and Schunck algorithm [14]. We choose the first algorithm because in [3] it shows a good overall behavior compared with the other algorithms that compute the optical flow at each pixel (100% density) as our algorithm does. The other two algorithms are related to the one shown here, as explained in section 2. The neighborhood size was 5×5 both for Anandan's algorithm and for our approach. For the other approaches the size of the neighborhood was 15×15. The size of the neighborhoods was chosen in order to provide the algorithms a sufficient amount of data (comparable to the quantity used in our approach) to perform the estimation as proposed in OFC Parameter Clustering algorithm [11] and Schunck algorithm[14]. The estimator here presented performs better than Anandan's algorithm in the case of the "Translating tree", "Diverging tree" and "Diverging office" sequences. In the case of "Rotating sphere" our approach performs better than OFC's parameter clustering. In the case of "Yosemite flight through" and "Street", Anandan's algorithm performs slightly better. The K-neighbor approach is slightly worse than the Kernel estimator, but it is also more computationally efficient.

4.3. Quantitative results, hierarchical algorithm

In order to show how the hierarchical algorithm performs better than the non hierarchical one when there are large velocities in the sequences, the input sequence must contain objects moving in such way. This can be done easily with the usual sequences: discarding one frame from each group of three multiplies the velocities by two; discarding two frames

Tab. 1. The error obtained in the test by Kernel estimator and k-neighborhood, compared to Anandan's algorithm [1], OFC Parameter Clustering [11] and Schunck algorithm [14] divided by 10^4.

Sequence	Anandan	Schunck	OFC param. clustering	kernel	k-neigh.
Trans. tree	4.4438	3.9809	6.7663	2.3039	2.4032
Div. tree	1.5067	3.5310	2.8188	1.4246	1.5195
Yosemite	10.154	16.09	15.580	10.985	11.287
Rot. Sphere	2.4696	1.8011	1.0137	0.6612	0.7408
Office	2.1625	5.1679	3.8374	1.9104	2.1828
Street	1.8539	5.8890	4.7729	2.1046	2.4034

from each group of four multiplies the velocities by three, and so on. In table 2 the results of the experiments where one frame of each group of three was discarded are shown. As it can be seen, when the sequence is sampled in time with a bigger acquisition interval between frames, all the algorithms increase the error, but the increment of the error is bigger for Anandan's algorithm than for the Kernel approach. This becomes more evident if sequences with bigger velocities are used in the experiments. In table 3 the experiments where two of each group of four frames are discarded (so the velocities are multiplied by three) are shown. As it can be seen, in this case, the hierarchical algorithm performs better than Anandan's algorithm and even than the non hierarchical one in all the sequences except than "Rotating Sphere" sequence.

Tab. 2. The error obtained in the test by non hierarchical Kernel algorithm and hierarchical Kernel, compared to Anandan's algorithm [1], OFC Parameter Clustering [11] and Schunck algorithm [14] divided by 10^4. Experiments with velocities multiplied by two.

Sequence	Anandan	Schunck	OFC param. clustering	Non hierar.	Hierar.
Tras. tree	8.8364	11.451	10.041	5.6818	5.9323
Div. tree	3.3408	5.3881	5.6539	2.6279	2.5138
Yosemite	22.018	23.994	28.099	19.761	18.784
Rot. Sphere	4.0091	2.8263	5.2662	1.1153	1.1268
Office	4.1539	7.6367	9.0805	3.8541	3.4233
Street	6.9549	7.5094	10.079	5.7681	4.5005

Tab. 3. The error obtained in the test by non hierarchical Kernel algorithm and hierarchical Kernel, compared to Anandan's algorithm [1], OFC Parameter Clustering [11] and Schunck algorithm [14] divided by 10^4. Experiments with velocities multiplied by three.

Sequence	Anandan	Schunck	OFC param. clustering	Non Hierar.	Hierar.
Tras. tree	13.375	15.758	14.177	10.067	9.5772
Div. tree	5.8071	7.0964	7.5708	4.1595	3.8966
Yosemite	33.688	34.274	38.893	34.046	32.765
Rot. Sphere	6.1136	5.2483	7.3555	2.7218	2.2945
Office	7.3861	9.6053	11.4065	6.1911	5.3540
Street	11.424	8.9061	13.413	9.7437	8.8084

5. Conclusions and future work

In this paper a new approach to optical flow field computation is shown. The algorithm here presented is robust due to the use of the mode as an estimator of the velocity with more probability of being correct. In this way, the algorithm preserves the boundaries of the movement because the basic assumption holds when most of the pixels in the neighborhood move with similar velocity. This is done within a smaller neighborhood than other approaches. This assumption only fails in the situations explained in 4.1. The efficiency of the algorithm can be improved in two areas: first, many of the calculations performed with the data in a neighborhood can be reused when velocities are computed in adjacent pixels, because only one row (or column) of data changes. Second, as mentioned in 2, the velocities are sorted in an array to improve the efficiency in the computation of the mode. If the incoming data is sorted separately from the reused data, the efficiency of the sorting stage is increased too.

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion, International Journal of Computer Vision, 2:283–310, 1989.

[2] Brendan McCane, Ben Galvin and Kevin Novins, On The Evaluation of Optical Flow Algorithms, Fifth International Conference on Control, Automation, Robotics and Vision, Singapore 9-11 December 1998, pages 1563-1567.

[3] J. L. Barron, D.J. Fleet, and S. S. Beauchemin, Performance of optical flow techniques, International Journal of Computer Vision, vol. 12, no. 1, pages. 43-77, 1994.

[4] S. S. Beauchemin and J.L. Barron, The computation of Optical Flow, ACM Computing Surveys, 3(27), pages 433-467, 1996.

[5] R. Battiti and E. Amaldi and C. Koch, Computing optical flow across multiple scales: An adaptive coarse-to-fine strategy, International Journal of Computer Vision, 6(2), pages 133-145, 1991.

[6] W. Enkelmann, Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences, IEEE Proc. of Workshop on Motion: Representation and Analysis, pages 81-87, 1986.
[7] D. J. Heeger, Optical flow using spatiotemporal filters,International Journal of Computer Vision, pages 1988,279-302.
[8] Berthold K. P. Horn and Brian G. Schunk, Determining Optical Flow, MIT, Artificial Intelligence Laboratory, 1980, in Computer Vision Principles, pages 481-497.
[9] David W. Murray and Bernard F. Buxton, Experiments in the Machine Interpretation of Visual Motion, MIT Press, 1990.
[10] H. H. Nagel, Displacement Vectors Derived from Second Order Intensity Variations, Computer Vision, Graphics, and Image Processing, 21, 1983.
[11] P. Nesi, A. Del Bimbo, D, Ben Tzvi, A Robust Algorithm for Optical Flow Estimation", Journal on Computer Vision, Graphics and Image Processing: Image Understanding, CVGIP:IU, Academic Press, USA, ISSN 1077-3142, Vol.61, N.2, pages 59-68, 1995.
[12] A. Del Bimbo and P. Nesi, Real-time optical flow estimation, in Proc. of 1993 IEEE Systems, Man and Cybernetics Conference, Le Touquet, France, October 1993.
[13] M. Otte and H.-H. Nagel. Optical flow estimation: advances and comparisons. In Proc. Third European Conference on Computer Vision, pages 51–60. Stockholm, Sweden, May 2-6, J.O. Eklundh, Springer LNCS 800, 1994.
[14] Brian G. Schunck, Image Flow Segmentation and Estimation by Constraint Line and Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence, October, pages 1010-1027, 1989.