VSP International
Science Publishers
P.O. Box 346, 3700 AH Zeist
The Netherlands

Computing Letters
vol. 2, no. 1-2, 2006, pp. 93–106
ISSN 1574–0404

An Algorithm for Optical Flow Computation based on a Quasi-Interpolant Operator

Elisa Francomano [1], Adele Tortorici

Dipartimento di Ingegneria Informatica, Universitá degli Studi di Palermo, Viale delle Scienze, 90128, Palermo, Italy
ICAR, Istituto di Calcolo e Reti ad Alte Prestazioni, CNR, Viale delle Scienze, 90128, Palermo, Italy

Carmelo Lodato, Salvatore Lopes

ICAR, Istituto di Calcolo e Reti ad Alte Prestazioni, CNR, Viale delle Scienze, 90128, Palermo, Italy

Abstract: A fundamental problem in the processing of image sequences is the computation of the velocity field of the apparent motion of brightness patterns usually referred to optical flow. In this paper a novel optical flow estimator based on a bivariate quasi-interpolant operator is presented. Namely, a non linear minimizing technique has been employed to compute the velocity vectors by modeling the flow field with a 2D quasi-interpolant operator based on centered cardinal B-spline functions. In this way an efficient computational scheme for optical flow estimate is provided. In addition the large solving linear systems involved in the process are sparse. Experiments on several image sequences have been carried out in order to investigate the performance of the optical flow estimator.

Keywords: Optical flow, B-spline, Quasi-Interpolant operator, Levenberg-Marquardt method.

PACS: 07.05.P

Mathematics Subject Classification: 2.60.Ed, 47A58

1 Introduction

The recovery of motion information from visual input is an important task in the processing of image sequences. The standard approach to represent motion information is the motion field, i.e. the projection of the 3D point motion onto the image plane. Generally, computer vision techniques compute an estimate of the motion field from the spatial and temporal variations of image brightness. This field is known as optical flow [1, 2]. Although optical flow differs from motion field, it is usually adopted in many applications ranging from passive interpretation to autonomous active exploration [3, 4]. It has been used, quite successfully, as an intermediate step, in problems of 3D object reconstruction and 3D scene analysis for computing information such as depth, surface orientation and motion parameters [5, 6, 7]. Moreover, optical flow computation is significant to other applications such as in measuring fluid flow velocity [8], in gauging blood flow

[1]elisa@cere.pa.cnr.it

and heart-wall motion in medical imagery [9] and in estimating the amounts of growth in corn seedlings [10, 11].

In this paper an algorithm for computing the optical flow is presented. The novelty of the proposed approach lies in using a bivariate quasi-interpolant operator based on centered cardinal B-spline functions for modeling the optical flow. As it is well known, interpolatory approximation is unlikely to generate a model that accurately reflects the true interaction among data deriving from visual input [12]. Hence, local quasi-interpolant operators which are based on data but which are not interpolatory are preferred [13, 12]. Furthermore, these operators provide computationally efficient schemes for constructing cardinal spline approximants with the highest order of approximation [14].

In the proposed algorithm the optical field is evaluated fixing the velocity values of some image points and deriving all the others by applying a 2D quasi-interpolant operator. The computed values are further improved by searching for the maximum likelihood solution. To this end, a non-linear minimization technique based on the modified Levenberg-Marquardt algorithm [15] has been employed. This algorithm requires the solution of a large linear system at each step. However, by using the quasi-interpolant operator adopted a sparse multi-banded matrix is generated. Hence, a solver system based on a descendent directions method has been used.

The remainder of the paper is organized as follows. An overall description of the computational process is reported in section 2. Section 3 reviews the preliminaries of the quasi-interpolant operator based on centered cardinal B-spline functions and the construction of the bivariate quasi-interpolant operator adopted. Section 4 reports details of the computational process characterized by the quasi-interpolant operator previously described. Section 5 regards experiments performed on several image sequences.

2 The computational process

Given an image sequence, the function $I_t(x, y)$ is the image brightness over time at pixel (x, y). Optical flow is defined as the velocity field $(u_t(x, y), v_t(x, y))$ resulting from the apparent motion of brightness patterns in consecutive images. The aim is to recover, at each pixel, the components u_t and v_t.

A solution to this problem consists in finding the vector (u, v) between $I_0(x, y)$ and $I_1(x, y)$ at each location (x, y) by minimizing the Sum of Squared Differences (SSD) function [16, 17, 18]:

$$\sum_{j=-k}^{k} \sum_{i=-k}^{k} [I_1(x + i + u, y + j + v) - I_0(x + i, y + j)]^2 \tag{1}$$

In this SSD function the summation is performed in a window of size $(2k + 1, 2k + 1)$ centered at (x, y). Most correlation-based methods perform an extensive search for the velocity vector in a finite integer-pair set and find the pair with the smallest SSD value as a solution. Since the search region of the velocity vector is discretized for this extensive search, the accuracy of the computed optical flow is limited by the discretization itself. A more reliable flow estimate over the entire image domain can be obtained by minimizing the following SSD function:

$$E(\{u_{ij}, v_{ij}\}) = \sum_i \sum_j [I_1(x_i + u_{ij}, y_j + v_{ij}) - I_0(x_i, y_j)]^2 \tag{2}$$

where a 2D dimensional spline model is used for representing the velocity field $u_{ij} = u(x_i, y_j)$ and $v_{ij} = v(x_i, y_j)$, $i = 1, ..., r$, $j = 1, ..., c$ in all $r \cdot c$ pixels [19, 20]. The spline model provides an approximation for the velocity field working with patches of a preset size. The partitioning in patches can be avoided by using a quasi-interpolant operator which also provides significant advantages in the computations. Therefore, a quasi-interpolant operator has been developed for estimating the optical flow with computationally efficient schemes [14, 21]. For the sake of exposition, a detailed description of the quasi-interpolant operator developed is reported in section 3.

The computational process starts by fixing some equally spaced points in which the initial velocity values $\{\hat{u}_{ij}, \hat{v}_{ij}\}_{i=i_1,...,i_{\hat{r}}\ j=j_1,...,j_{\hat{c}}}$, $\hat{r} < r$ and $\hat{c} < c$, are arbitrarily chosen. These points are the knots for the local approximation model. Naturally, the initial values set produces only a rough evaluation of the overall velocity field. Better optical flow estimates can be derived from a minimization process that improves the initial velocity values. To this end, the modified Levenberg-Marquardt iterative non-linear technique [15] has been employed to minimize Eq(2) in respect to the $\{\hat{u}_{ij}, \hat{v}_{ij}\}_{i=i_1,...,i_{\hat{r}}\ j=j_1,...,j_{\hat{c}}}$.

The computational process requires the generation of an hessian block matrix where the block entries are:

$$\begin{pmatrix} \frac{\delta^2 E}{\delta \hat{u}_{mn} \delta \hat{u}_{pq}} & \frac{\delta^2 E}{\delta \hat{u}_{mn} \delta \hat{v}_{pq}} \\ \frac{\delta^2 E}{\delta \hat{u}_{mn} \delta \hat{v}_{pq}} & \frac{\delta^2 E}{\delta \hat{v}_{mn} \delta \hat{v}_{pq}} \end{pmatrix}$$

$$m, p = i_1, ..., i_{\hat{r}} \;\; n, q = j_1, ..., j_{\hat{c}}$$

and a gradient block vector γ in which the components are:

$$\begin{pmatrix} \frac{\delta E}{\delta \hat{u}_{mn}} \\ \frac{\delta E}{\delta \hat{v}_{mn}} \end{pmatrix}$$

$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

At each step of the process the following linear system has to be solved:

$$(H + \omega D)\Delta\varepsilon = -\gamma \tag{3}$$

where ω is a stabilization term [15], $D = diag(H)$ and $\Delta\varepsilon$ is a block vector in which the entries are:

$$\Delta\varepsilon_{mn} = \begin{pmatrix} \varepsilon_{\hat{u}_{mn}} \\ \\ \varepsilon_{\hat{v}_{mn}} \end{pmatrix}$$

$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

The solution of the linear system Eq(3) is used for upgrading the current velocity values, i.e.:

$$\begin{aligned} \hat{u}^*_{mn} &= \hat{u}_{mn} + \varepsilon_{\hat{u}_{mn}} \\ \hat{v}^*_{mn} &= \hat{v}_{mn} + \varepsilon_{\hat{v}_{mn}} \end{aligned} \tag{4}$$

$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

After which the quasi-interpolant operator is applied on the upgraded values and the new velocity field $\{u^*_{ij}, v^*_{ij}\}_{i=1,...,r\ j=1,...,c}$ is generated. If the following relation:

$$E(\{u^*_{ij}, v^*_{ij}\}) < E(\{u_{ij}, v_{ij}\}) \tag{5}$$

$$i = 1, ..., r \;\; j = 1, ..., c$$

is verified, the current values are replaced according with Eq(4) and the parameter decreases. Otherwise, the previous values are preserved and the parameter ω is increased. The computational process is summarized in the following.

ALGORITHM

Given:
- $I_0(x, y)$ and $I_1(x, y)$
- a set of initial velocity values for the optical flow
- *maxit* and *toll* to stop the iterative process

Objective: Computation of the optical flow

Steps:

1. Compute the optical flow by applying the quasi-interpolant operator on the prefixed initial velocity values
2. Compute E_{Old} according with Eq(2)
3. Set the parameter ω
4. **Repeat**

   4.1 Compute the vector γ

   4.2 Compute the matrix H

   4.3 Generate the matrix $(H + \omega D)$

   4.4 Solve the linear system (3)

   4.5 Compute the optical flow by applying the quasi-interpolant operator on Eq(4)

   4.6 Compute E_{New} according with Eq(2)

   4.7 Check if $E_{New} < E_{Old}$ and update ω

   Until $(E_{New} - E_{Old}) < toll$ or $iterates > maxit$

3 Quasi-interpolant operator

As previously mentioned the optical flow has to be computed in all pixels starting from the initial values. In this paper a quasi-interpolant operator has been adopted to model the flow field and this section is devoted to its analysis.

3.1 Univariate formulation

When discrete data is uniformly distributed, an acceptable approximation can be obtained by using quasi-interpolant operators. These operators work with a truncated series of opportune locally supported functions [14].

A quasi-interpolant operator [14, 21] is defined by the following approximation scheme:

$$(Qf)(x) = \sum_{i=1}^{k} (\lambda_i f)(x)\phi_i(x) \tag{6}$$

where $\phi_1, \phi_2, ..., \phi_k$ are opportune functions and $\lambda_1, \lambda_2, ..., \lambda_k$ are appropriate linear functionals.

Let N^d be the d^{th} order cardinal B-spline function [22] with $suppN^d = (0, d)$ and $(\lambda f)(l)$ a linear combination of data sequence $\{f_j\}_{j\in Z}$:

$$(\lambda f)(l) = \sum_{j\in Z} \lambda_{l-j} f_j$$

the quasi-interpolant operator :

$$(Qf)(x) = \sum_{l\in Z} (\lambda f)(l) N^d(x + \frac{d}{2} - l) \tag{7}$$

maps $C_b(R) = \{f \in C(R) : sup|f(x)| < \infty\}$ into the centered cardinal B-spline functions space V_0^d . By indicating with $\prod_{d-1}$, the algebraic polynomials of degree at most $d-1$, the operator Q verifies the following interesting result [14]:

$$(Qp)(x) = p(x) \qquad p(x) \in \Pi_{d-1}$$

This operator is local in the sense that $(Qf)(x)$ depends only on the values of f in a small neighborhood of x . Therefore, the quasi-interpolant operator Q provides computationally efficient schemes for constructing spline approximants providing the highest order of approximation. For instance, by considering the following linear functional:

$$(\lambda f)(l) = \frac{1}{6}[-f(l+1) + 8f(l) - f(l-1)] \tag{8}$$

the univariate quasi-interpolant operator with $d = 4$ is generated :

$$(Qf)(x) = \sum_{l\in Z} \frac{1}{6}[-f(l+1) + 8f(l) - f(l-1)] N^4(x + 2 - l) \tag{9}$$

Moreover, a multiresolution process can be carried out. Considering the data sequence $\{f_j\}_{j\in Z}$ in the space V_0^d, a data sampling in the space V_τ^d can be derived by fixing the sampling period as small as $h = 2^{-\tau}$. Thus, the quasi-interpolant operator Q^h gives the optimal approximation order going from the 0 resolution level, i.e. $x = q$, $q \in Z$, to the τ resolution level, i.e. $x = q * 2^{-\tau}$, by verifying the following result [14]:

For each $f \in C_b(R) \bigcap C^d(\Omega)$, where Ω is any given open set containing a compact K set in R there exists some positive constant α , depending only on f and K such that :

$$max_{x\in K} |(Q^h f - f)(x)| \leq \alpha h^d$$

for all sufficiently small $h > 0$.

3.2 Bivariate formulation

This section reports the bivariate quasi-interpolant operator which has been constructed and used in computing the optical flow. Taking into account the Eq(9) the following bivariate expression is generated:

$$
\begin{aligned}
(Qf)(x,y) = \sum_{l\in Z}\sum_{k\in Z} & [\frac{1}{36}f(l+1,k+1) - \frac{2}{9}f(l+1,k) + \frac{1}{36}f(l+1,k-1) + \\
& -\frac{2}{9}f(l,k+1) + \frac{16}{9}f(l,k) - \frac{2}{9}f(l,k-1) + \\
& +\frac{1}{36}f(l-1,k-1) - \frac{2}{9}f(l-1,k) + \frac{1}{36}f(l-1,k-1)] \\
& N^4(x+2-l)N^4(y+2-k)
\end{aligned} \tag{10}
$$

Fixing $x = q$ and $y = r$, $q, r \in Z$, the summation in Eq (10) consists only of the terms indexed by l and k values verifying the following two conditions:

$$
\begin{aligned}
q - 1 \leq l \leq q + 1 \\
r - 1 \leq k \leq r + 1
\end{aligned}
$$

The operator in Eq(10) can be rewritten by exclusively using even points. In this way the evaluation in odd points is obtained , i.e.:

$$
\begin{aligned}
(Qf)(x,y) = \sum_{l\in Z}\sum_{k\in Z} & [(\Lambda^{(1)}f)(l,k)N^4(2x-2l+2)N^4(2y-2k+2) + \\
& + (\Lambda^{(2)}f)(l,k)N^4(2x-2l+3)N^4(2y-2k+3) + \\
& + (\Lambda^{(3)}f)(l,k)N^4(2x-2l+2)N^4(2y-2k+3) + \\
& + (\Lambda^{(4)}f)(l,k)N^4(2x-2l+3)N^4(2y-2l+2)]
\end{aligned} \tag{11}
$$

where:

$$
\begin{aligned}
(\Lambda^{(1)}f)(l,k) &= \sum_{i=-2}^{1}\sum_{j=-2}^{1} c_{ij}f(i+l,j+k) \\
(\Lambda^{(2)}f)(l,k) &= \sum_{i=-2}^{1}\sum_{j=-2}^{1} d_{ij}f(i+l,j+k) \\
(\Lambda^{(3)}f)(l,k) &= \sum_{i=-2}^{1}\sum_{j=-2}^{1} g_{ij}f(i+l,j+k) \\
(\Lambda^{(4)}f)(l,k) &= \sum_{i=-2}^{1}\sum_{j=-2}^{1} h_{ij}f(i+l,j+k)
\end{aligned} \tag{12}
$$

The quantities $\{c_{ij}\}$, $\{d_{ij}\}$, $\{g_{ij}\}$ and $\{h_{ij}\}$, are computed so that the following relation is verified [20]:

$$(Qp)(q,r) = p(q,r) \qquad p \in \Pi_3^2 \;\; and \;\; q,r \in Z$$

The masks with the correspondent values are reported in fig.1.

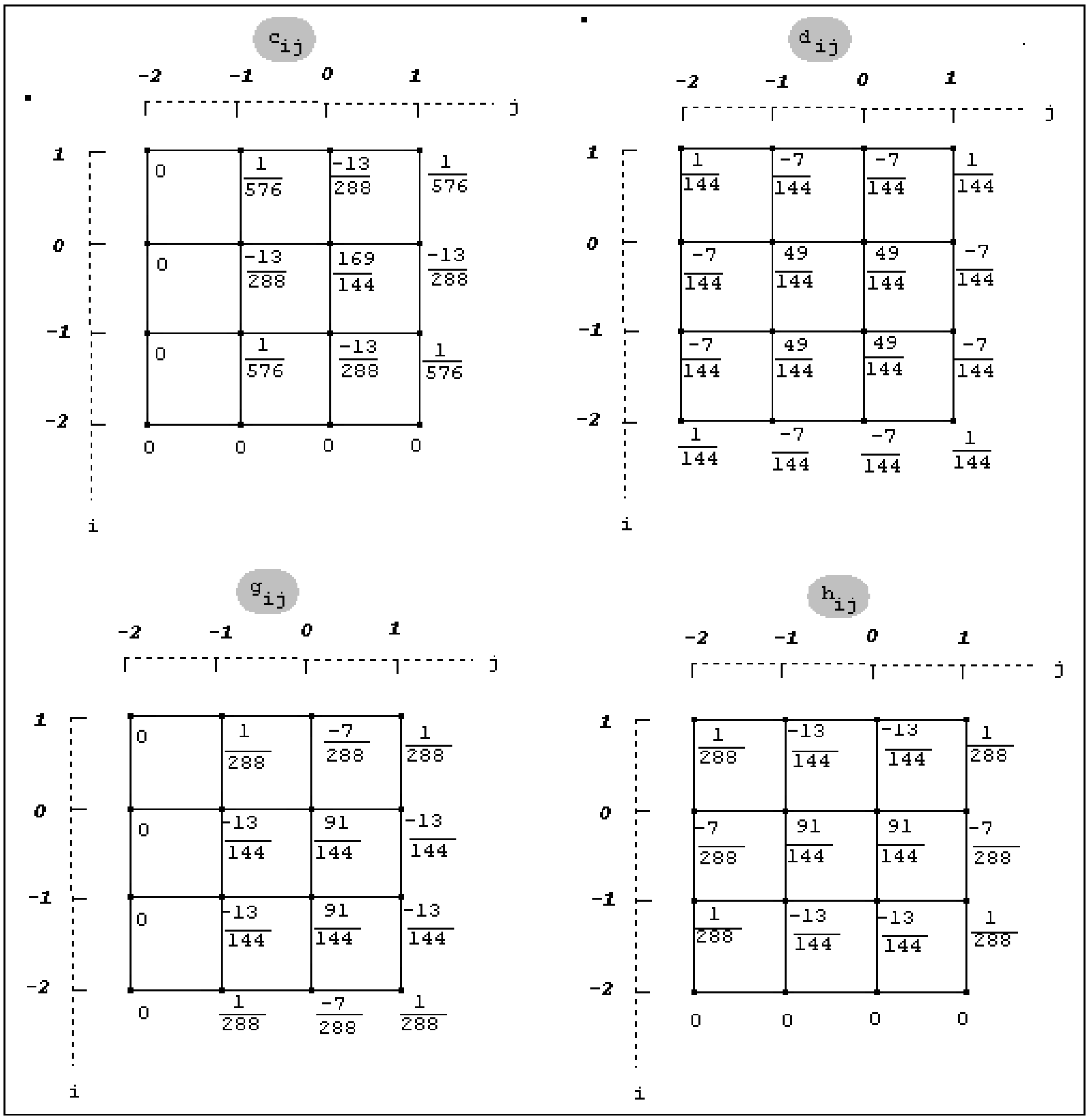


Figure 1: *Masks of the quasi-interpolant operator*

4 Quasi-interpolant motion model

Let $\{\hat{u}_{ij}, \hat{v}_{ij}\}_{i=i_1,...,i_{\hat{r}}\ j=j_1,...,j_{\hat{c}}}$ be the initial velocity values for the velocity field corresponding to the knots of the quasi-interpolant operator. A first approximation of the flow field can be expressed as:

$$u(x_i, y_j) \approx (Qu)(x_i, y_j) \tag{13}$$

$$v(x_i, y_j) \approx (Qv)(x_i, y_j) \tag{14}$$

$$i = 1, ..., r \;\; j = 1, ..., c$$

By indicating with:

$$e_{ij} = I_1(x_i + u_{ij}, y_j + v_{ij}) - I_0(x_i, y_j) \tag{15}$$

$$i = 1, ..., r \;\; j = 1, ..., c$$

and by referring to the quantities involved in section 2, the components of the vector γ are computed as follows:

$$\frac{\delta E}{\delta \hat{u}_{mn}} = 2 \sum_{i=1}^{r} \sum_{j=1}^{c} e_{ij} I_{ij}^{x} R_{ij}^{(mn)} \tag{16}$$

$$\frac{\delta E}{\delta \hat{v}_{mn}} = 2 \sum_{i=1}^{r} \sum_{j=1}^{c} e_{ij} I_{ij}^{y} R_{ij}^{(mn)} \tag{17}$$

$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

where:

$$(I_{ij}^{x}, I_{ij}^{y}) = \nabla I_1(x_i + u_{ij}, y_j + v_{ij}) \tag{18}$$

$$i = 1, ..., r \;\; j = 1, ..., c$$

and:

$$R_{ij}^{(mn)} = \Big(\frac{\delta(Qu)}{\delta \hat{u}_{mn}}\Big)_{ij} = \Big(\frac{\delta(Qv)}{\delta \hat{v}_{mn}}\Big)_{ij} \tag{19}$$

$$i = 1, ..., r \;\; j = 1, ..., c$$
$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

whilst the entries of the hessian matrix H require the following computations :

$$\frac{\delta^2 E}{\delta \hat{u}_{mn} \delta \hat{u}_{pq}} = 2 \sum_{i=1}^{r} \sum_{j=1}^{c} (I_{ij}^{x})^2 R_{ij}^{(mn)} R_{ij}^{(pq)}$$
$$\frac{\delta^2 E}{\delta \hat{v}_{mn} \delta \hat{u}_{pq}} = 2 \sum_{i=1}^{r} \sum_{j=1}^{c} I_{ij}^{x} I_{ij}^{y} R_{ij}^{(mn)} R_{ij}^{(pq)}$$
$$\frac{\delta^2 E}{\delta \hat{v}_{mn} \delta \hat{v}_{pq}} = 2 \sum_{i=1}^{r} \sum_{j=1}^{c} (I_{ij}^{y})^2 R_{ij}^{(mn)} R_{ij}^{(pq)} \tag{20}$$

$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

In order to compute the derivatives $R_{ij}^{(mn)}$ more easily, Eq(11) can be rewritten as follows:

$$(Qf)(x,y) = \sum_{l\in Z}\sum_{k\in Z}\sum_{i=-2}^{l}\sum_{i=-2}^{1} f(l+i,k+j)[c_{ij}N^4(2x-2l+2)N^4(2y-2k+2) + \\ +d_{ij}N^4(2x-2l+3)N^4(2y-2k+3) + \\ +g_{ij}N^4(2x-2l+2)N^4(2y-2k+3) + \\ +h_{ij}N^4(2x-2l+3)N^4(2y-2k+2)] \quad (21)$$

Considering that $suppN^4 = (0,4)$, the quantities $R_{ij}^{(mn)}$ satisfy the following relations:

if $i \in \{i_1,...,i_{\hat{r}}\}$ and $j \in \{i_1,...,i_{\hat{c}}\}$ **then**

$$R_{ij}^{(mn)} = \begin{cases} 1 & \text{if } (i,j) = (m,n) \\ 0 & \text{if } (i,j) \neq (m,n) \end{cases} \quad (22)$$

else

$$R_{ij}^{(mn)} \neq 0 \quad (m-2\rho) \leq i \leq (m+2\rho) \quad (n-2\rho) \leq j \leq (n+2\rho) \quad (23)$$

where ρ is the distance between two pixels in which the initial velocity values are fixed. The mask in fig.2 shows the $R_{ij}^{(mn)}$ values.

In computing the entries of the matrix H only some products $R_{ij}^{(mn)}R_{ij}^{(pq)}$ have to be calculated due to:

$$R_{ij}^{mn}R_{ij}^{pq} \neq 0 \quad if \ (m-4\rho) \leq p \leq (m+4\rho) \quad (n-4\rho) \leq q \leq (n+4\rho) \quad (24)$$

The matrix H structure strongly depends on the quasi-interpolant operator. The matrix size is equal to $2\hat{r}\hat{c}$ however the approximant operator adopted gives rise to a sparse matrix consisting of nine *18*-bandwidth bands. This feature is not dependent on the number of initial points, consequently the sparsity increases when more initial points are set. Therefore, the process developed provides a significant computational reduction compared to the spline approximant. This spline model generates a matrix with zero entries depending on the number of patches involved in the partitioning.

Finally, a preconditioned conjugate gradient method has been employed in solving the system Eq(3). A good convergence rate has been reached in the performed experiments by using a block diagonal matrix as a preconditioner. Each diagonal block is the inverse of the following matrix :

$$\begin{pmatrix} (1+\omega)\dfrac{\delta^2 E}{\delta^2 \hat{u}_{mn}} & \dfrac{\delta^2 E}{\delta \hat{u}_{mn}\delta \hat{v}_{mn}} \\ \dfrac{\delta^2 E}{\delta \hat{u}_{mn}\delta \hat{v}_{mn}} & (1+\omega)\dfrac{\delta^2 E}{\delta^2 \hat{v}_{mn}} \end{pmatrix}$$

$$m = i_1,...,i_{\hat{r}} \ \ n = j_1,...,j_{\hat{c}}$$

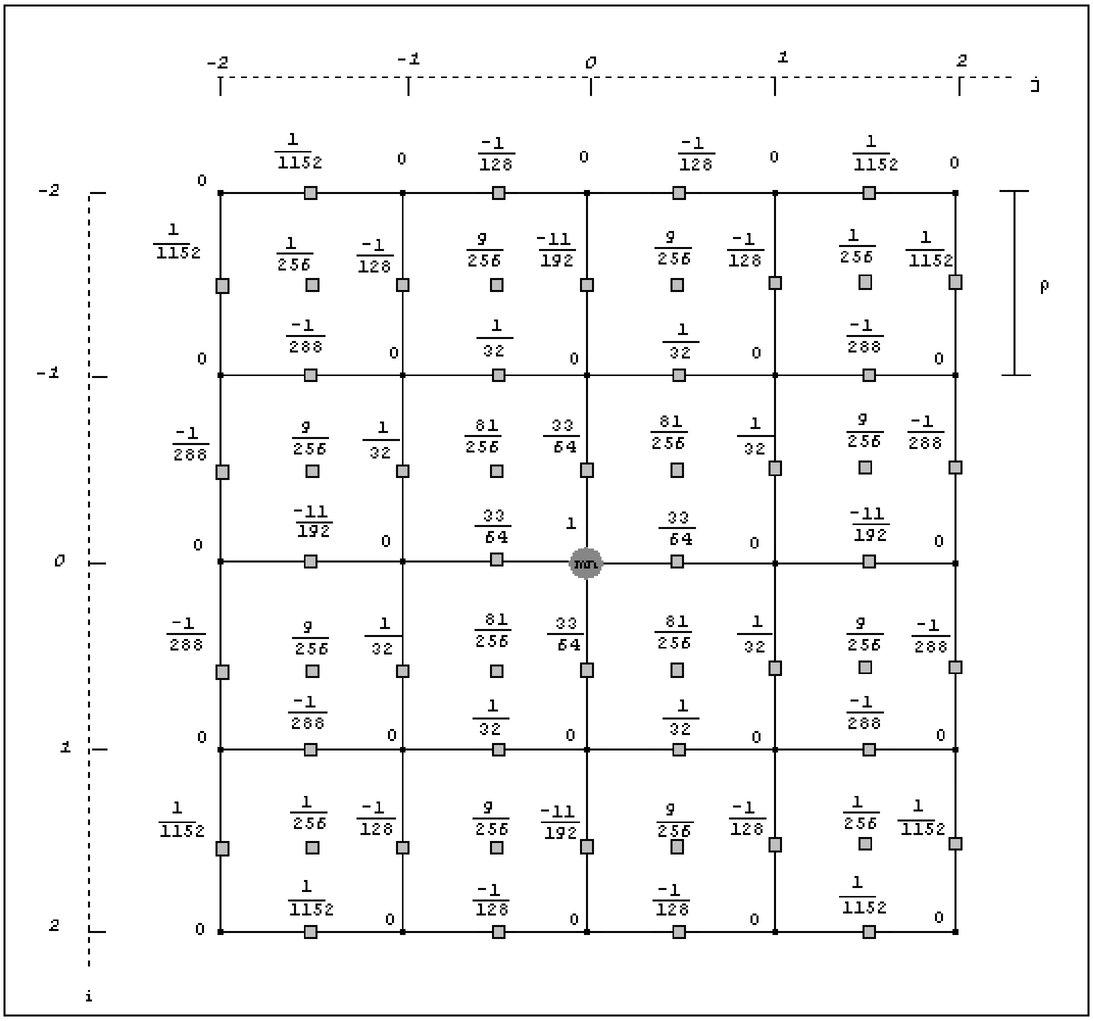


Figure 2: *Masks of the* $R_{ij}^{(mn)}$

5 Experimental results

In order to investigate the performance of the process developed, tests have been carried out on several image sequences. Quantitative results are reported in terms of optical flow and relative error. The results have been obtained without any pre- and post-processing techniques, i.e. neither presmoothing nor thresholding have been applied. The relative error, between the correct velocity vector and the estimated one, has been computed in each pixel. A gray-level scale has been adopted to give a graphic representation of the computed error. As previously mentioned, the initial points value can be arbitrarily fixed. In the conducted experiments, the following relations have been adopted:

$$\hat{u}_{mn} = \hat{v}_{mn} = k[I_0(x_m, y_n) - I_1(x_m, y_n)] \tag{25}$$

$$m = i_1, ..., i_{\hat{r}} \;\; n = j_1, ..., j_{\hat{c}}$$

where k is prefixed so that $u, v \leq 10^{-3}$.

Sequence 1
In this sequence two squares of a *11*-pixel width shift with a velocity of *(1;-1)* and *(-1;1)* pixels/frame respectively. The squares and the background are characterized by an uniformly varying brightness, which is not immediately evident due to the subtle variations in brightness. The first image of the sequence is reported in fig.3.

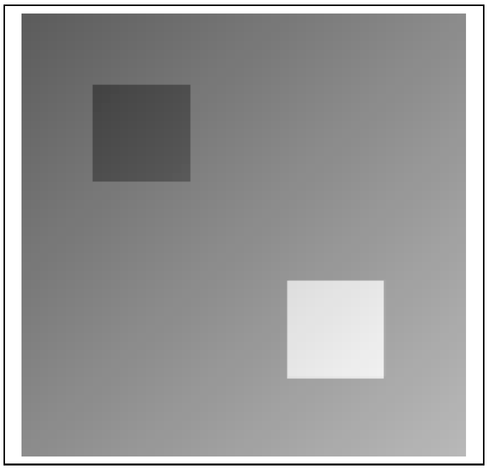


Figure 3: *One frame of the sequence 1*

The results obtained are quite satisfactory. The velocity field is well recovered as shown in fig.4(a). The error is located in correspondence to the edges of the squares, where it is about 30% and reaches the maximum value of 61% on a vertex (fig.4(b)).

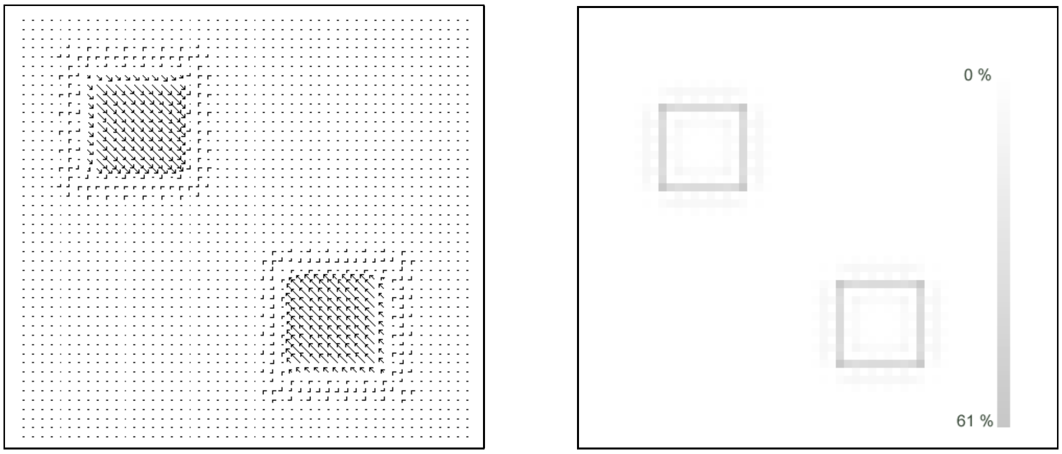
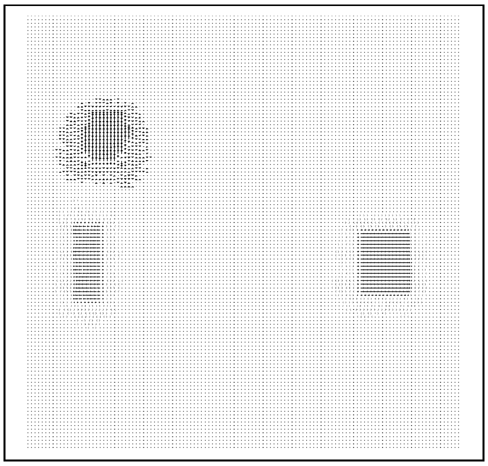


Figure 4: *(a)Optical flow* *(b)Relative error*

Sequence 2
This sequence has been generated from a real scene (fig.5) applying movements on three distinct areas. Namely, the hat moves up with a velocity of *(0;1)* pixels/frame. The dark wood panel, placed between the bookcase and the wall, shifts to the left with a velocity of *(-0.8;0)* pixels/frame. The screen of the monitor shifts towards the right with a velocity of *(1;0)* pixels/frame.

Figure 5: *One frame of the sequence 2*

The overall optical flow is shown in fig.6(a) where the three movements are well detected. The relative error is 1.5 % on average. The maximum error values, also in this case, are localized on the edges of the moving areas and are about 50% on average.



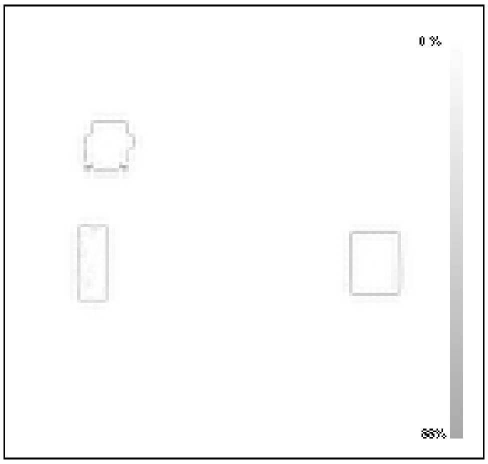


Figure 6: *(a)Optical flow* *(b)Relative error*

The peak of the error is of 88% and is reached near the lower part of the hat (fig.6(b)). This is highlighted in the particulars shown in the figs.7 - 8.

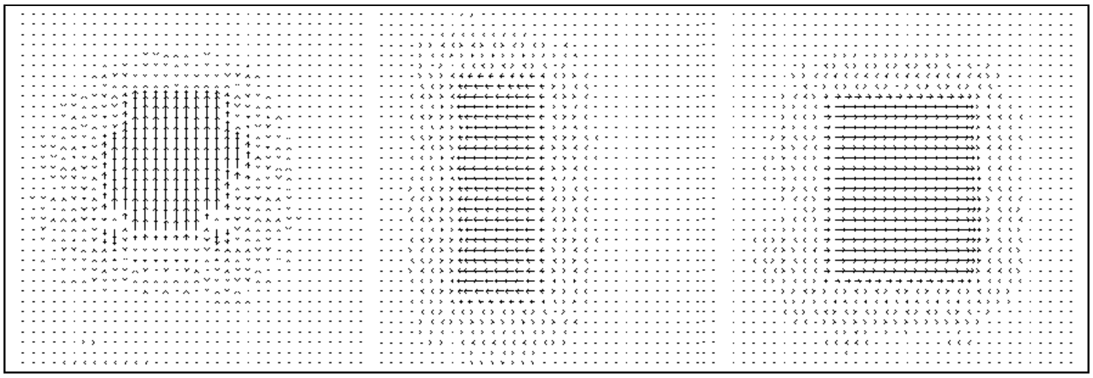


Figure 7: *Optical flow on the moving areas*

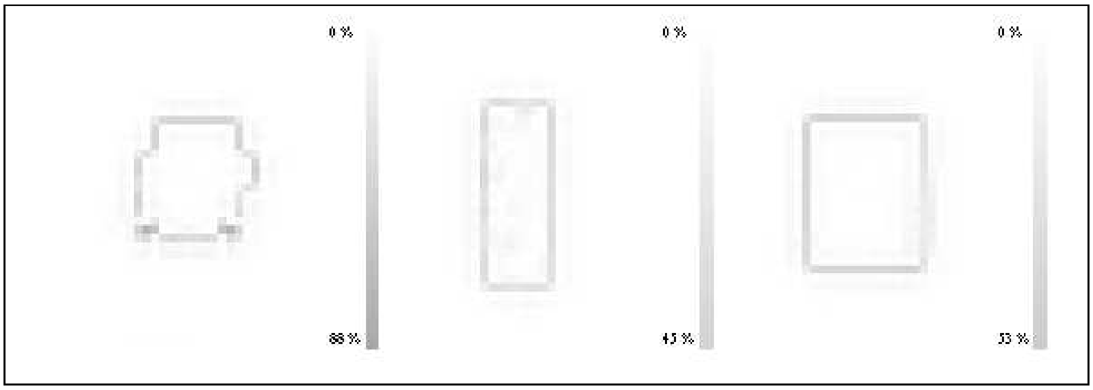


Figure 8: *Relative error on the moving areas*

Sequence 3

In this experiment the previous sequence has been modified by applying a random noise of 10% at the current brightness of each pixel. The first frame of the sequence is depicted in fig.9(a). As shown in fig.9(b) the resulting flow is still appreciable considering that all the significant movements are still recognized. The relative error is 6.7% on average.

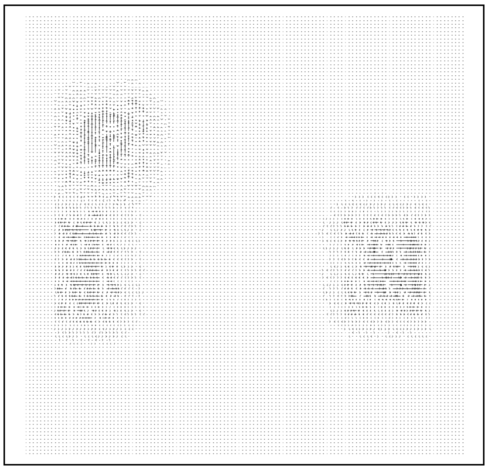


Figure 9: *(a)One frame of the sequence 3* *(b)Optical flow*

6 Conclusion

This paper proposes a novel algorithm to compute optical flow. The model has been developed by using a bivariate quasi-interpolant operator to represent the flow field and by involving a non-linear minimizing technique. The computational process is characterized by an efficient solving scheme working with sparse coefficient matrices. The sparsity strongly depends on the quasi-interpolant operator adopted, based on centered cardinal B-spline functions of order 4.

As shown in the performed experiments, the proposed algorithm seems promising. However, improvements are expected by the employment of a multi-frames sequence. In addition, considering that the motion in real image sequences involves a large amount of data and that the computational model developed is suitable for non-conventional platforms, studies on parallel and distributed environments are already in progress.

References

[1] K. Horn, *Robot vision*, MA MIT Press, Cambridge, 1986.

[2] G. Schunck, *Image flow: fundamentals and algorithms*, W. Martin and J.K.Aggarwal, Eds. Boston Kluwer, 1988.

[3] P. Boutemy and A. Mitiche, Computation and analysis of image motion : a synopsis of current problems and methods, *International Journal of Computer Vision* **19**, 29-55(1996).

[4] J.L. Barron, J. Fleet and S.S. Beauchemin, Systems and Experiment-Performance of optical flow techniques, *International Journal of Computer Vision* **12**, 43-77(1994).

[5] G. Adiv, Inherent ambiguities in recovering 3-D motion and structure from noisy flow field, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**, 477-489(1999).

[6] S. Negahdaripour and S. Lee, Motion recovery from image sequences using only first order optical flow, *International Journal of Computer Vision* **9(3)**, 163-184(1992).

[7] K. Prazdny, On the information in optical flow, *Computer Vision Graphics* **23**, 239-259 (1983).

[8] H. Quent, J. Pakleza and T.A. Kowalewsi, Particle image velocimetry with optical flow, *Experiments in fluids* **25**, 177-189 (1998).

[9] J.L. Price and E.R. McVeigh, Motion estimation from tagged mr image sequences, *IEEE Trans. On Medical Images* **11(2)**, 238-249 (1992).

[10] J.L. Barron and A. Liptay, Optical flow to measure minute increments in plant growth, *BioImaging* **2(1)**, 57-61(1994).

[11] J.L. Barron, A. Liptay, T. Jewett and I. van Wesenbeeck, Oscillations in Corn Seedling Growth as Measured by Optical Flow, *Journal of American Society for Horticultural Science* **120(3)**, 379-385 (1995).

[12] H.N. Mhaskar, F.J. Narcowich and J.D. Ward, Quasi-interpolation in shift invariant spaces, *J. Math. Anal. Appl.* **251**, 356-363(2000).

[13] C. de Boor, On uniform approximation by splines, *Journal of Approximation Theory*, 219-235 (1968).

[14] C.K. Chui, *An introduction to wavelets*, Academic Press, 1992.

[15] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge Univ. Press, 1992.

[16] P. Anandan, A computational framework and an algorithm for the measurement of visual motion, *International Journal of Computer Vision* **2(3)**, 283-310(1989).

[17] H. Lai and B.C. Vemuri, Reliable and efficient computation of optical flow, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **5(4)**, 598-610(1996).

[18] R. Szeliski and H. Shum, Motion estimation with quadtree splines, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **18(12)**, 1199-1210(1996).

[19] R. Szeliski, Fast surface interpolation using hierarchical basis functions, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12(6)**, 513-528(1990).

[20] V. Calderone, E. Francomano and A. Tortorici, Optical flow computation in a multi-scale framework, *Applicable Mathematics its perspective and Challenges*, 153-161(2000).

[21] C.K. Chui, *Multivariate splines*, CBMS-NSF Regional Conference Series in Applied SIAM, 1988.

[22] L. Schumaker, *Spline functions: Basic theory*, J. Wiley & Sons, 1981.