The INTERNATIONAL CONFERENCE ON ELECTRONICS & OIL: FROM THEORY TO APPLICATIONS
March 05-06, 2013, Ouargla, Algeria

Horn & Schunck Meets a Discrete Zhang Neural Networks for Computing 2D Optical Flow

F. Charif [(1)], A. Benchabane[(1)], N. Djedi[(2)], and A. Taleb-Ahmed[(3)]

[(1)] *Electrical Engineering Department, University of Ouargla, Algeria*
[(2)] *Department of Computer Science, LESIA Laboratory of Research University of BISKRA, Algeria*
[(3)] *LAMIH Laboratory, FRE CNRS 3304 UVHC, University of Valenciennes, France*
cherif.fe@univ-ouargla.dz
Benchabane.ab@univ-ouargla.dz
djedi_nour@yahoo.fr
Abdelmalik.Taleb-Ahmed@univ-valenciennes.fr

***Abstract*— In this paper, we develops a new fast on line algorithms for motion estimation. This new algorithm is based on the Horn & Schunck algorithm with a new kind of recurrent neural network called Discrete Zhang Neural Networks (DZNN) and Simoncelli's matched-pair 5 tap filters. We simulate the network on a sequential processor and compare its performance with a sequential algorithm based on the Jacobi method. Experimental results on synthetic and real image sequences for the method are given to demonstrate its fastness in comparison with Jacobi method.**

Key-Words— Motion estimation, Real-time applications, Horn & Schunck method, discrete Zhang neural networks.

I. INTRODUCTION

MOTION estimation is the process of determining motion vectors that describe the transformation from one 2D image to another; usually from adjacent frames in a video sequence. Motion estimation is known to be a fundamental step in many applications in image processing, pattern recognition, data compression, and biomedical technology [1].

A great number of approaches for Optical Flow estimation have been proposed in the literature, including differential, correlation-based, energy-based, and phase-based methods [1]. The benchmark of the based differential methods is certainly the one proposed by Horn & Schunck [2]. They imposed a global constraint of smoothness on the velocity field and the optical flow estimation is reduced to a minimization problem of a functional, where the solution obtained using iterative Gauss–Seidel method is high computational cost [3]. In recent years, a number of different schemes have been proposed to implement the Horn & Schunck optical flow algorithms in real-time [4-9] where the accuracy and the fastness depend largely on the number of iterations and the size of the image respectively.

Since March 2001, a special kind of recurrent neural networks called the Discrete Zhang Neural Network (DZNN) has been proposed by Zhang and co-workers for solving online time-varying and/or static problems [10-19], such as time-varying Sylvester equation solving, time-varying matrix inversion, and time-varying linear inequalities. All these models which are generally independent of each other prove an efficiency and superiority for the online application [20]. Motivating by the fastness and robustness of these methods versus the gradient based neural networks; we propose a parallel implementation of the Horn & Schunck motion estimation method in image sequences using DZNN for the online matrix inversion. The remainder of this paper is organized as follows: Section 2 presents the standard Horn & Schunck's method of optical flow computation. A real-time computation of dense optical flow for 2D images by the classical Horn & Schunck model by using discrete Zhang neural networks is described in section 3. Section 4 shows the experimental results obtained using our fast image motion estimation method. Finally, section 5 contains the conclusion.

II. HORN & SCHUNCK FORMULATION

The Horn & Schunck's differential method [2] is mainly based on optimizing an energy function (over a domain Ω):

$$\int_{\Omega}\left((I_t + uI_x + vI_y)^2 + \lambda^2(\|\nabla u\|^2 + \|\nabla v\|^2)\right)dxdy \qquad (1)$$

where $I = I(x,y,t)$ denotes image brightness function at time t, (u,v) designates optical velocity, $(\nabla u, \nabla v)$ its spatial derivatives, (I_x, I_y, I_t) the spatiotemporal image brightness derivatives, and λ is the weighting parameter.
We can rewrite the Euler-Lagrange equations as:

$$\begin{cases} I_x^2 u + I_x I_y v + I_x I_t = \lambda^2 \nabla^2 u \\ I_x I_y u + I_y^2 v + I_y I_t = \lambda^2 \nabla^2 v \end{cases} \qquad (2)$$

Let Ω be discretized via a unit-spacing grid D and the

grid points indexed by the integers $\{1,2,\ldots,N\}$ with $N = n \times m$ (is the number of discretization points). We take numbering to be top-down and left-to-right. For all grid point indices $i \in \{1,2,\ldots,N\}$, a discrete approximation of the Euler-Lagrange equations (2) is [3]:

$$\begin{cases} I_{xi}^2 u_i + I_{xi} I_{yi} v_i + I_{xi} I_{ti} - \lambda^2 \sum_{j \in N_i} (u_j - u_i) = 0 \\ I_{xi} I_{yi} u_i + I_{yi}^2 v_i + I_{yi} I_{ti} - \lambda^2 \sum_{j \in N_i} (v_j - v_i) = 0 \end{cases} \quad (3)$$

where $(u_i, v_i) = (u, v)_i$ is the optical velocity vector at grid point i; I_{xi}, I_{yi}, I_{ti} are the values at i of I_x, I_y, I_t, respectively, and N_i is the set of indices of the neighbors of i. For the 8-neighborhood, $card(N_i) = 8$ for points interior to the discrete image domain, and $card(N_i) < 8$ for boundary points. By rewriting (3), we can have the following system of linear equations [3]:

$$\begin{cases} (I_{xi}^2 + \lambda^2 c_i) u_i + I_{xi} I_{yi} v_i - \lambda^2 \sum_{j \in N_i} u_j = -I_{xi} I_{ti} \\ I_{xi} I_{yi} u_i + (I_{yi}^2 + \lambda^2 c_i) v_i - \lambda^2 \sum_{j \in N_i} v_j = -I_{yi} I_{ti} \end{cases} \quad (4)$$

where $c_i = card(N_i)$.

The minimization of the above energy leads to solving a linear system of equations:

$$Az = b \quad (5)$$

where $z = (z_1, \ldots, z_{2N})^T \in R^{2N}$ be the vector with coordinates $z_{2i-1} = u_i$, $z_{2i} = v_i$, $i \in \{1,\ldots,N\}$,

$b = (b_1, \ldots, b_{2N})^T \in R^{2N}$ be the vector with coordinates $b_{2i-1} = -I_{xi} I_{ti}$, $b_{2i} = -I_{yi} I_{ti}$, $i \in \{1,\ldots,N\}$

and the stiffness matrix $A \in R^{2N \times 2N}$ is symmetric positive-definite [3] with elements [3]:

$$\begin{cases} A_{2i-1,2i-1} = I_{xi}^2 + \lambda^2 c_i \\ A_{2i,2i} = I_{yi}^2 + \lambda^2 c_i \qquad \text{for all } i \in \{1,\ldots,N\} \\ A_{2i-1,2i} = A_{2i,2i-1} = I_{xi} I_{yi} \end{cases} \quad (6)$$

$A_{2i-1,2j-1} = A_{2i,2j} = -\lambda^2$ *for all* $i, j \in \{1,\ldots,N\}$, such that $j \in N_i$, all other elements being equal to zero.

System (5) is a large-scale sparse system of linear equations. The solution to equation (5) can be solved in closed form when A is a non-singular matrix. Mitiche and Mansouri [3] proved that the matrix A is symmetric positive-definitive. The solution to (5) is given by:

$$z = A^{-1} b \quad (7)$$

with the Jacobi method, the updates can be activated in parallel for all image points at each iteration and it can be written in matrix form as [3]:

$$z^{k+1} = Pz^k + d \quad (8)$$

where d is the $2N$ real vector with entries: $d_{2i-1} = -(I_{xi} I_{ti} / (I_{xi}^2 + I_{yi}^2 + \lambda^2 c_i))$

$d_{2i} = -(I_{yi} I_{ti} / (I_{xi} + I_{yi} + \lambda\, c_i))$, $i \in \{1,2,\ldots,N\}$;

and P is the $2N \times 2N$ Jacobi matrix corresponding to the 2×2 block division of A. The elements of P are:

$P_{2i,2i} = P_{2i-1,2i-1} = 0, \forall i \in \{1,\ldots,N\}$

$$\begin{cases} P_{2i,2j} = (I_{xi}^2 + \lambda^2 c_i) \,/\, c_i (I_{xi}^2 + I_{yi}^2) + \lambda^2 c_i^2 \\ P_{2i-1,2j} = P_{2i,2j-1} = (I_{xi} + I_{yi}) \,/\, c_i (I_{xi}^2 + I_{yi}^2) + \lambda^2 c_i^2 \\ P_{2i-1,2j-1} = (I_{yi}^2 + \lambda^2 c_i) \,/\, c_i (I_{xi}^2 + I_{yi}^2) + \lambda^2 c_i^2 \end{cases} \quad (9)$$

$\forall i,\ j \in \{1,\ldots,N\}, such\ that \cdots j \in N_i$

with all other elements equal to zero, including the diagonal entries.

III. SOLUTION VIA DISCRETE ZHANG NEURAL NETWORK

In many engineering applications, the online inversion of matrices is usually desired when the dimension of the matrix is large. Recently, a special kind of recurrent neural network for online matrix inversion has been proposed by Zhang et al.[14,16].It is well known [14] that DZNN method is preferable for solving large sparse systems which the rate of the convergence is high. To avoid the direct computation of the inverse matrix, we propose to use the discrete Zhang neural network for the online matrix inversion.

To solve for a matrix inverse, the DZNN system design is based on the equation, $AX(t) - I = 0$, with $A \in R^{2N \times 2N}$ and $I \in \Re^{2N \times 2N}$ denotes an identity matrix. We can define a matrix–valued error function such as [9] $\varepsilon(t) = AX(t) - I$. Then, we use the negative of the gradient as the descent direction:

$$\frac{d\varepsilon(t)}{dt} = -\Gamma f(\varepsilon(t)) \quad (9)$$

where the parameter $\Gamma \in \Re^{2N \times 2N}$ is a positive-definite matrix used to scale the convergence rate of the solution and $f(\cdot): \Re^{2N \times 2N} \to \Re^{2N \times 2N}$ $f(\cdot)$ denote the activation function-matrix. In general, any monotonically increasing odd activation function $f(\cdot)$ can be used for recurrent neural networks construction (such as the linear, sigmoid, power, and power–sigmoid functions) [14]. Expanding (9) leads to:

$$A\dot{X}(t) = -\Gamma f(AX(t) - I) \quad (10)$$

To make every entry converges to zero at the same rate and at the same time, we take $\Gamma = \eta I$ with $\eta > 0$ which leads to:

$$A\dot{X}(t) = -\eta f\left(AX(t) - I\right) \quad (11)$$

We note here to ensure that the matrix A is non-singular hence the ZNN model can be rewritten as [14]:

$$\dot{X}(t) = -\eta A^{-1} f\left(AX(t) - I\right) \quad (12)$$

as A^{-1} is unknown and $X(t)$ could be very close to A^{-1} after some number of iterations, the matrix A^{-1} in equation (12) could be replaced by $X(t)$. Then we can rewrite the equation (12) as:

$$\dot{X}(t) = -\eta X(t) f\left(AX(t) - I\right) \quad (13)$$

Finaly , the discrete-time update equation for the network to be simulated on a sequential computer is :

$$\dot{X}(t) = (X(k+1) - X(k))/h \quad (14)$$

where h denote the sampling time parameter. Then the discrete model of ZNN is then given by:

$$X(k+1) = X(k) - \tau X(k) f\left(AX(k) - I\right) \quad (15)$$

where $\tau = \eta h > 0$ is the step size.

Because the matrix A is positive-definitive, the initial state to start the DZNN can be chosen as:

$$X(0) = 2I / trace(A) \quad (16)$$

In order to implement the Zhang network, Eq. (15) has been transformed into a vectorial form based on the Kroneker product $\otimes$ and the vectorization operator $vec(.)$ [13].

The matrix-form equation (15) can be reformulated as the following vector-form equation:

$$vec(X(k+1)) = vec(X(k)) - \tau (I \otimes X(k)) \cdot f\left((I \otimes A)vec(X(k)) - vec(I)\right) \quad (17)$$

where activation-function mapping $f(\cdot)$ in (17) is defined the same as in (15) except that its dimensions are changed hereafter as $f(\cdot): \Re^{(2N)^2 \times 1} \to \Re^{(2N)^2 \times 1}$.

The DZNN model for the online matrix inversion is show the figure (1) according to the equation (17).

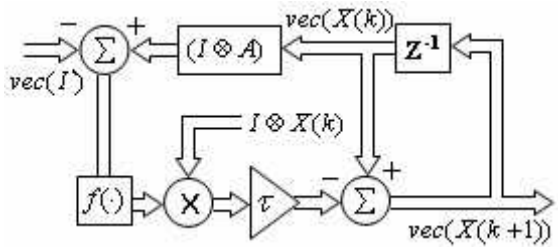


Fig. 1. Block diagram of the DZNN

However, with the DZNN method, the optical flow can be activated in parallel for all image points, to afford a significant gain in execution time.

IV. SIMULATION RESULTS

In this section the experimental results obtained by using the DZNN are presented and compared with those obtained by the Jacobi method. To demonstrate the performance of our algorithm, a variety of images have been tested, including synthetic images and real image. For all simulations, we estimate the spatial and temporal derivatives of images by Simoncelli's matched-pair 5 tap filters after smoothing by a simple averaging kernel (1/4,1/2,1/4) with superior accuracy to the Gaussian 1.5 filter [21-22] .

A. Synthetic Image Sequences

Tow synthetic image sequences were used to test our method quantitatively and compare it with other optical flow techniques. The Sinusoid-1 is the least complicated sequences to extract motion from. The images have no sharp changes in brightness, no occlusion boundaries, and a constant velocity in space and time. A 2-dimensional sinusoid texture translates with constant velocity to the top right of the screen with velocity $V = (1.585, 0.863)$ pixels/frame, Figure 2 shows the first sinusoid texture used and true flow field.

The Yosemite sequence is more complex with a wide range of velocities, it consists of fly-through of a virtual valley with a cloudy, since it not only has detailed textures, non-constant flow and sharp changes in brightness, but it also has occlusion between the top of the valley and the sky.

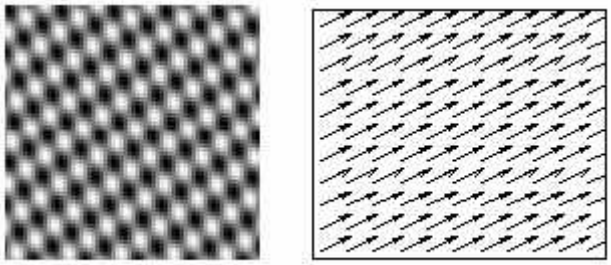


Fig. 2. Sinusoid-1 a) one frame from the sequence, b) the respective true flow field

In all the demonstrations, we have compared the estimated optical flow to the true optical flow in order to determine quantitative results. The angular error measure between the computed optical flow $\vec{v}_e$ and the true flow $\vec{v}_c$ is adopted in the experiments [1].

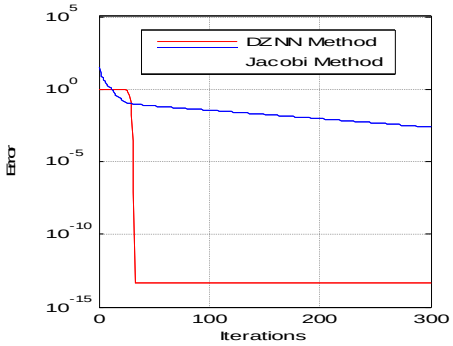
$$\Phi_E = ar\cos(\vec{v}_c \cdot \vec{v}_e) \quad (18)$$

The accuracy and the number of iterations obtained to compute the optical flow for the Sinusoid-1 image sequence are shown in Tables 1. For all these experiments, the accuracy is reported for a flow of 100% density computed using the DZNN method with pre-selected error 10^{-14}.

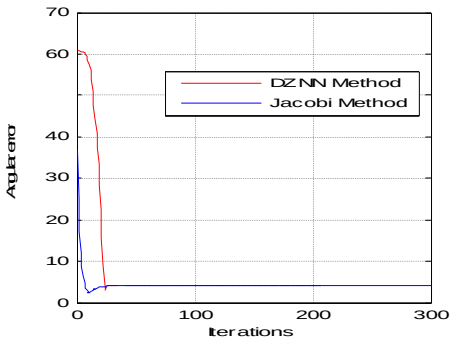
TABLE 1: Angular error and number of iterations obtained on the Sinusoid-1 sequence.

	Image size	DZNN method		Jacobi Method	
		$\lambda=10^{2}$	$\lambda=10^{-3}$	$\lambda=10^{2}$	$\lambda=10^{-3}$
angular error[°]	16x16	14.85	14.72	14.85	14.72
	32x32	8.90	8.89	8.90	8.89
iterations	16x16	29	58	552	532
	32x32	33	61	1227	1209

From Table 1, we can see that the proposed method requires 33 iterations to reach an average error of 8.9098°, whereas the numerical scheme of Jacobi Method takes 1227 iterations to converge to a same average error. This proves that the proposed method offers a faster convergence speed.



(a) Convergence curves of the two methods



(b) Impact of the two methods on the average Error

Fig. 3. Sinusoid-1 : Impact of our method and Jacobi method on the convergence error and the average error for $\lambda = 100$.

From the table 1, and the figure (3), we note that the results of the Jacobi method are insensitive to the regularization parameter λ, whereas the results of our method are sensitive to the regularization parameter λ, for example we found that a gain of number of iteration is about 50% is obtained by passing from $\lambda = 0.001$ to $\lambda = 100$ .The optical flow obtained for $\lambda = 100$ is shown in figure 4.

In order to represent the computed optical flow, each velocity vector is encoded according to the color map; witch gives the angle while the brightness represents the norm of the considered displacement.

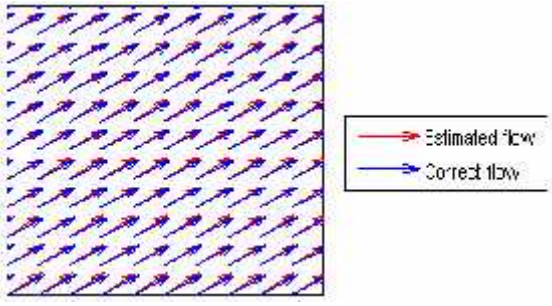
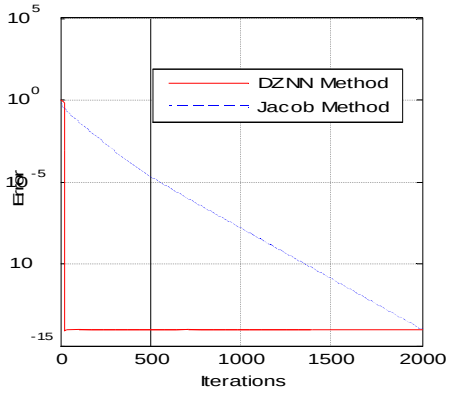
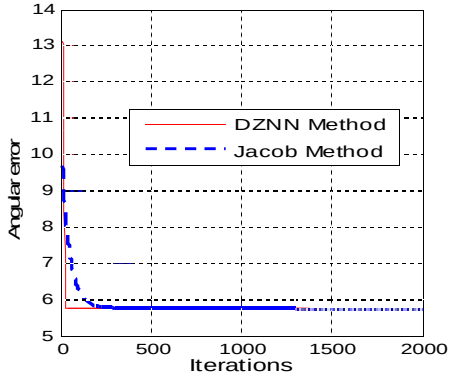


Fig. 4. Optical flow of the sinusoid-1 computed with $\lambda = 100$ using DZNN or Jacobi method

In the second experiment, we use the famous Yosemite sequence with clouds. The accuracy in the computed optical flow for this sequence is shows in figure 5. We note, that the convergence error and the angular error decrease in the same manner for the two methods, but our method reached the minimum error just after a few iterations which make it faster and stable than the Jacobi method.



(a) Convergence curves of the two methods.



(b) Impact of the two methods on the average Error.

Fig.5. Yosemite: Impact of our method and Jacobi method on the convergence error and the average Error.

The main advantage of this representation comes from the possibility of drawing a dense optical flow, whereas a vector field representation can mask or highlights some absurd

points due to its sampling. Thus, Figure. 6 illustrates the motion field performs on the Yosemite sequence with clouds using DZNN and Jacobi methods. The ground truth flow and the error distribution are also shown. As seen in Figure 6.(d-f), the Yosemite sequence results with Jacob method have many errors obtained after 27 iterations. The average angular error is 8.384°. Much of this error occurs in the sky and in the left side of the valley. The valley in the middle and right side has a good texture and almost no occlusion, giving the good results seen. Furthermore, our method (DZNN) has the best performance with an average angular =5.763° obtained after 27 iterations (in figure 6(g-i)), while the Jacobi method requires 446 iterations to obtain this accuracy. Much of this error occurs in the sky. In figure 6(e) and 6(h), small angular error are observed in blue.

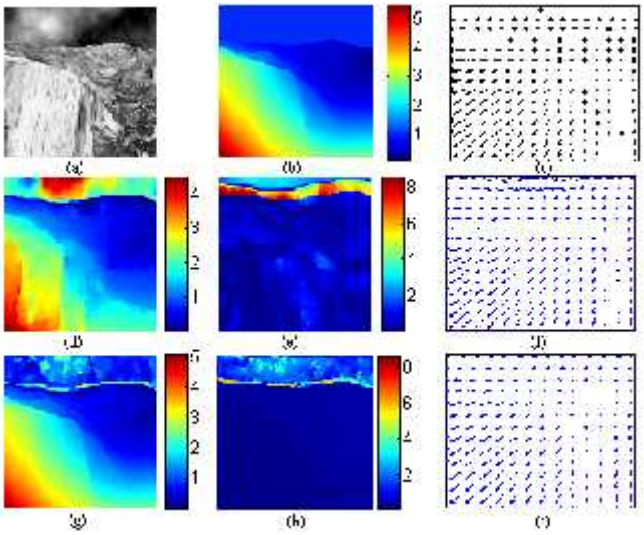


Fig. 6 Impact of DZNN method and Jacobi method for the Yosemite sequence with clouds

a) Synthetic sequence, (b) Ground truth (Color map), (c) Ground truth (Vector field), (d) Color map result with Jacob method after 27 iterations, (e) Angle Error map result with Jacob method , (f) Computed vector field with Jacob method , (g) Color map result with DZNN after 27 iterations,(h) Angle Error map result with DZNN and (i) Computed vector field with DZNN

B. Real Image Sequence

Our method was also tested on the Hamburg Taxi sequences , this real sequence contains three principal moving objects : the taxi turning around the corner (1 pixel/frame), a car in the lower left , driving from right (3 pixels/frame) and a van in lower right driving right to left (3 pixels/frame). Figure 7 shows one frame from the Hamburg Taxi sequence and the zoom in at the taxi . The regularization parameter λ , is empirically chosen to be 100 for the implementation of our and the Jacobi methods.

Since it is very difficult to determine the true optical flow for real images, usually only qualitative testing is performed. The validity of motion estimates for this real sequence can be verified by using these estimates in a motion-compensated interpolation. For this task, a bilinear interpolation is used. The impact of the proposed and Jacobi methods on motion –compensated prediction and prediction error- are show in Figure.8 and 9.

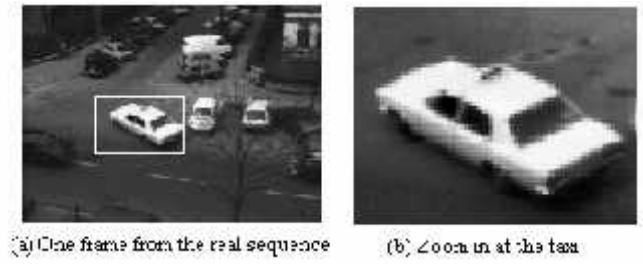
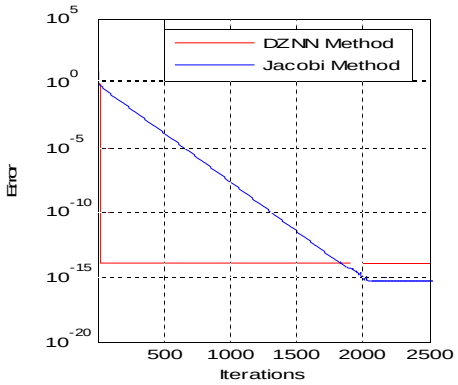
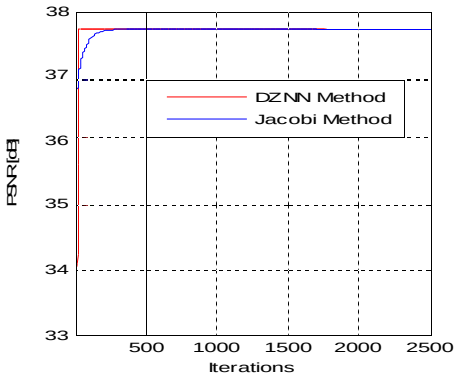


Fig.7. One frame from the Hamburg Taxi sequence and the zoom in at the taxi.



(a) Convergence curves of the two methods



(b) Impact of the two methods on the PSNR

Fig. 8. Impact of our method and Jacobi method on the convergence error and the PSNR for the Hamburg Taxi sequence.

Figure 8 shows the dynamic of the Jacobi method and DZNN method and the respect PSNR rate, we note that the PSNR rate increase in the same manner for the two methods, but our method reached the maximum PSNR just after a few iterations method with make it faster than the Jacobi method.

The motion in the zoom in at the taxi (figure 9-a) is shows in figure 9-b. From the zoom in at the taxi , we can see that moving object was clearly captured and located by

our method with few iterations (35) , while the Jacobi method requires 2040 iterations with a SNR=37.34dB.

V. CONCLUSION

In this paper we have developed a new fast on line algorithms for the estimation of optical flow for image sequences. This new algorithm was based on the Horn and Schunck algorithm with DZNN method and Simoncelli's matched-pair 5 tap filters. Experimental results on synthetic and real image sequences for our method are given to demonstrate its fastness in comparison with Jacobi method. There are several basic advantages of our algorithm. It is well known that the DZNN is a parallel algorithm. It has high parallel efficiency when parallel computers are used [12,23-24]. The DZNN algorithm is also very efficient when the problem under consideration is too large. Furthermore, our method is faster than the (sequential) Jacobi method.

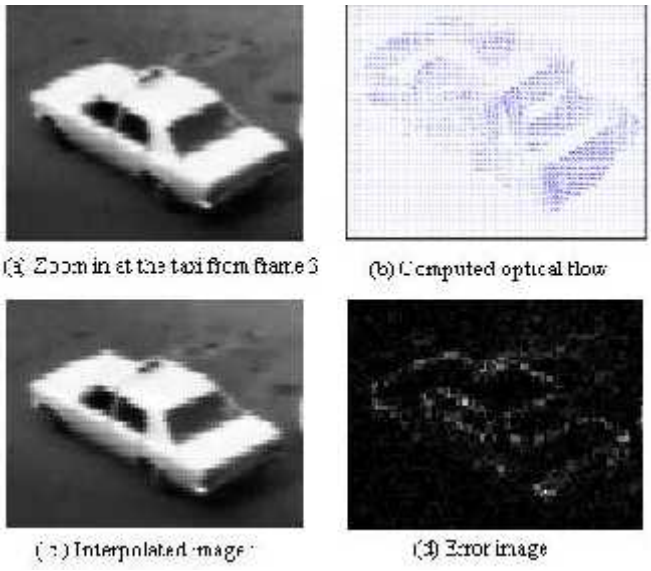


Fig. 9 Results of zoom in the taxi for the Hamburg Taxi sequence.

REFERENCES

[1] Barron .L, Fleet .J, S.S.Beauchemin., Performance of optical flow techniques. International Journal of Computer Vision, vol. 12(1), 1994, 43–77.

[2] Horn.K.P, Schunck. B.G., Determing optical flow. Artificial Intelligence,vol.17,185–204,1981.

[3] A.Mitiche, A.R.Mansouri., on convergence of the Horn and Schunck optical-flow computation." IEEE, Image Processing,vol. 13, 2004, 848–852.

[4] Martín.J. L, Zuloaga.A, Cuadrado.C, Lázaro. J, Bidarte. U., Hardware implementation of optical flow constraint equation using FPGAs. Computer Vision and Image Understanding, vol. 98, 2005, 462-490.

[5] Zuloaga .A, Martín. J. L, Ezquerra. J., Hardware architecture for optical flow estimation in real time, Proc.ICIP,vol. 3, 1998, 972-976.

[6] Cobos.P, Monasterio.F., FPGA implementation of the Horn & Schunk Optical Flow Algorithm for Motion Detection in real time Images, Dcis'98 Proceedings, XIII Design of circuits and systems conference, 1998, 616-621.

[7] Cobos.P, Monasterio.F., FPGA implementation of Camus correlation Optical Flow Algorithm for real time images, 14th International Conference on Vision Interface Proceedings, 2001, 32-38.

[8] Bruhn.A , Weickert.J , Feddern.C, Kohlberger .T, and Schnorr.C. ,Variational optical flow computation in real time. IEEE Transactions on Image Processing 14(5), 2005, 608–615.

[9] Bruhn.A , Weickert.J ,Kohlberger.T, and Schnorr.C., A Multigrid Platform for Real-Time Motion Computation with Discontinuity-Preserving Variational Methods.Int. J. Computer Vision, 70(3), 2006, 257-277.

[10] Zhang. Y, Jiang.D. & Wang.J., A recurrent neural network for solving Sylvester equation with time-varying coefficients. IEEE Transactions on Neural Networks, 13(5), 2002, 1053-1063.

[11] Zhang Y. & Ge. S. S., Design and analysis of a general recurrent neural network model for time-varying matrix inversion. IEEE Transaction on Neural Networks, 16(6), 2005, 1477-1490.

[12] Y. Zhang.,Revisit the analog computer and gradient-based neural system for matrix inversion, in Proc. IEEE Int. Symp. Intell. Control,Limassol, Cyprus, 2005, 1411–1416.

[13] Zhang. Y, Chen.K, Ma.W, and Xiao.L., MATLAB Simulation of Gradient-Based Neural Network for Online Matrix Inversion. In: Huang, D. S., Heute, L. & Loog, M. (eds.), ICIC 2007, LNCS(LNAI), 4682, 2007,98-109, Springer, Heidelberg.

[14] Zhang. Y, Ma.W & Yi.C., The link between Newton iteration for matrix inversion and Zhang neural network (ZNN). Proceeding of IEEE International Conference on Industrial Technology, Chengdu, China.2008.

[15] Zhang.Y, Cai.B, Liang.M & Ma.W., On the Variable Step-Size of Discrete-Time Zhang Neural Network and Newton Iteration for Constant Matrix Inversion. Second International Symposium on Intelligent Information Technology Application.2008.

[16] Zhang.Y, Ma.W & Cai. B., From Zhang Neural Network to Newton Iteration for Matrix Inversion. IEEE Transactions on Circuits and Systems I. 56(7), 2009, 1405-1415.

[17] Zhang.Y. & Li.Z., Zhang neural network for online solution of time-varying convex quadratic program subject to time-varying linear equality constraints. Physics Letters A, 373(18-19), 2009, 1639-1643.

[18] Zhang. Y, Xiao. L, Ruan. G & Li. Z., Continuous and discrete time Zhang dynamics for time-varying 4th root finding. Numerical Algorithms, 57(1), 2011, 35-51.

[19] Xiao. L, & Zhang. Y., Zhang neural network versus Gradient neural network for solving time-varying linear inequalities. IEEE Transactions on Neural Networks, 22(10), 2011, 1676-1684.

[20] Benchabane.A, Bennia.A, Charif. F. and Taleb-Ahmed.A., Multi-dimensional Capon spectral estimation using discrete Zhang neural networks , Multidimensional Systems and Signal Processing 2012 .

[21] Simoncelli.E.P., Design of multi-dimensional derivative filters, IEEE Int. Conf. Image Processing,vol. 1, 1994, 790-793.

[22] Charif .F. , Baarir.Z-E., A Fast Modified Horn & Schunck Method, IEEE International Symposium on Industrial Electronics, vol. 1, 2006, 526-531.

[23] Syed Atiqur. R, Ansari. M. S., A neural circuit with transcendental energy function for solving system of linear equations. Analog Integrated Circuit and Signal Processing, 2011.

[24] Eberhardt.S.P,Tawel.R, Brown.T.X , Daud.T ,Thakoor.AP., Analog VLSI neural networks: implementation issues and examples in optimization and supervised learning. IEEE Transactions on Industrial Electronics, vol. 39(6), 552-564, 1992.