Optical Flow Estimation via Neural Singular Value Decomposition Learning

Simone Fiori[*], *Nicoletta Del Buono*[†], *Tiziano Politi*[‡]

Keywords:
Singular value decomposition; Orthogonal matrix group; Differential geometry; Optical flow.

Pages: 11, **Figures**: 3, **References**: 21.

To be presented at:
International Conference on Computational Science and its Applications – ICCSA*04

February 16, 2004

[*]Facoltà di Ingegneria, Università di Perugia, Polo Didattico e Scientifico del Ternano, Loc. Pentima bassa, 21, I-05100 Terni, Italy (Email: fiori@unipg.it)

[†]Dipartimento di Matematica, Università di Bari, Via Orabona 4, I-70125 Bari, Italy (Email: delbuono@dm.uniba.it)

[‡]Dipartimento di Matematica, Politecnico di Bari, Via Amendola 126/B, I-70126, Bari, Italy (Email: pptt@dm.uniba.it)

Optical Flow Estimation via Neural Singular Value Decomposition

Simone Fiori

Abstract

In the recent contribution [9], it was given a unified view of four neural-network-learning-based singular-value-decomposition algorithms, along with some analytical results that characterize their behavior. In the mentioned paper, no attention was paid to the specific integration of the learning equations which appear under the form of first-order matrix-type ordinary differential equations on the orthogonal group or on the Stiefel manifold. The aim of the present paper is to consider a suitable integration method, based on mathematical *geometric integration* theory. The obtained algorithm is applied to optical flow computation for motion estimation in image sequences.

1 Introduction

The computation of the singular value decomposition (SVD) of a non-square matrix [11, 12, 19], plays a central role in several signal/data automatic processing. It has found widespread applications e.g. in automatic control [15], digital circuit design [14], time-series prediction [18] and image processing [3, 16]. Recently, some efforts have been devoted to SVD computation by neural networks in the neural community [2, 17, 21].

The aim of this paper is to present some notes on neural SVD computation with application to optical flow estimation.

Optical-flow (OF) estimation is a well-known image-processing operation that allows estimating the motion of portions of an image over an image-sequence. It is closely related to motion estimation. Most of the OF estimation algorithms used in video encoding belong to either Block Matching Algorithms (BMAs) or Pixel Recursive Algorithms (PRAs) [13]: The majority of the current estimation methods employ a block-matching algorithm.

The BMA methods are based on the concept of template-matching: It is supposed that a single block in a time-frame has moved solidly to another location in a subsequent time-frame, so the image-block is regarded as a template to be searched for in the subsequent frame. The BMA methods try to achieve motion computation by reducing the number of search locations in the search range and/or by reducing the number of computations at each search location. Such algorithms are either *ad hoc* or are based on the assumption that the error increases monotonically from the best-match location. However, typically the error surface may exhibit local minima and the majority of the OF estimation methods get trapped in one of the local minima depending on the starting point and the search direction. Also, the matching algorithms aim at finding the best

match with respect to some selected mismatch (error) measure, but the best match may not represent the true motion [4].

Conversely, the standard PRAs try to estimate the motion at each pixel. In the method for OF estimation considered here, based on paper [4], a methodology similar to the PRA is employed but we operate on a pixel-block basis and find a single motion-vector for each block. In order to make the method robust in a noisy environment, we use the total least squares (TLS) estimation approach.

Through the paper we use the following notation. Symbol $I_{m,n}$ denotes the pseudo-identity matrix of size $m \times n$ and $I_m = I_{m,m}$ while symbol $0_{m,n}$ denotes a $m \times n$ all-zero matrix. Symbol X' denotes the transposition of the matrix X while X^* denotes Hermitian-transposition; symbol $\mathrm{tr}(X)$ denotes the trace of the square matrix X, i.e. the sum of its in-diagonal entries. The orthogonal group $\mathrm{O}(m, \mathbb{K}) \stackrel{\mathrm{def}}{=} \{X \in \mathbb{K}^{m \times m} | X^* X = I_m\}$, where the field $\mathbb{K}$ may be either $\mathbb{R}$ or $\mathbb{C}$. For details on this Lie group see e.g. [8]. Also, the Frobenius norm of a matrix $X \in \mathbb{K}^{n \times n}$ is defined as $\|X\|_{\mathrm{F}} \stackrel{\mathrm{def}}{=} \sqrt{\mathrm{tr}(X^* X)}$.

2 Optical flow estimation by total-least-squares

Let us consider a sequence of gray-level images $\{\mathcal{I}(x, y, t)\}_t$, where $\mathcal{I}$ denotes the scalar image intensity, the pair (x, y) denotes the coordinate-pair of any pixel, and t denotes the frame index.

During motion, any pixel moves from frame t and position (x, y) to frame $t + \Delta t$ at position $(x + \Delta x, y + \Delta y)$. The fact that the pixel-intensity has moved over the image support may be formally expressed with the *optical-flow conservation* equation, namely:

$$\mathcal{I}(x, y, t) = \mathcal{I}(x + \Delta x, y + \Delta y, t + \Delta t) \ . \tag{1}$$

On the basis of the above conservation equation and on the knowledge of a sequence of two subsequent frames it is possible to estimate the motion of any pixel within the sequence $\{\mathcal{I}(x, y, t)\}_t$.

In fact, let us define $\Delta\mathcal{I}(x, y, t) \stackrel{\mathrm{def}}{=} \mathcal{I}(x, y, t + \Delta t) - \mathcal{I}(x, y, t)$. For this quantity we have:

$$\begin{aligned} \Delta\mathcal{I}(x, y, t) &= \mathcal{I}(x - \Delta x, y - \Delta y, t) - \mathcal{I}(x, y, t) \ , \\ &= -\mathcal{I}_x(x, y, t)\Delta x - \mathcal{I}_y(x, y, t)\Delta y + \text{h.o.t.} \ , \end{aligned}$$

where $\mathcal{I}_x$ and $\mathcal{I}_y$ denote the partial derivates of the image function, h.o.t. denotes the sum of higher-order terms in the Taylor expansion of the image function, and the vertical and horizontal displacements $(\Delta x, \Delta y)$ have been supposed small enough for the Taylor series to represent accurately the optical flow change. The latter hypothesis is equivalent to assuming slow motion or sufficiently high-rate image sampling.

As mentioned, we make the solid-block-motion assumption, thus the above equation holds true, with the same values of displacements, for a set of pixels

located within the square described by $x \in [x_1, x_{N_x}]$ and $y \in [y_1, y_{N_y}]$, where constants N_x and N_y denote the block-size. On the basis of these considerations, it is possible to write the resolving system for any block between frames t and $t + \Delta t$, that is:

$$\begin{aligned}
\mathcal{I}(x_1, y_1, t + \Delta t) - \mathcal{I}(x_1, y_1, t) &= -\mathcal{I}_x(x_1, y_1, t)\Delta x - \mathcal{I}_y(x_1, y_1, t)\Delta y \ , \\
\mathcal{I}(x_2, y_1, t + \Delta t) - \mathcal{I}(x_2, y_1, t) &= -\mathcal{I}_x(x_2, y_1, t)\Delta x - \mathcal{I}_y(x_2, y_1, t)\Delta y \ , \\
\mathcal{I}(x_3, y_1, t + \Delta t) - \mathcal{I}(x_3, y_1, t) &= -\mathcal{I}_x(x_3, y_1, t)\Delta x - \mathcal{I}_y(x_3, y_1, t)\Delta y \ , \\
&\vdots \\
\mathcal{I}(x_1, y_2, t + \Delta t) - \mathcal{I}(x_1, y_2, t) &= -\mathcal{I}_x(x_1, y_2, t)\Delta x - \mathcal{I}_y(x_1, y_2, t)\Delta y \ , \\
&\vdots \\
\mathcal{I}(x_{N_x}, y_{N_y}, t + \Delta t) - \mathcal{I}(x_{N_x}, y_{N_y}, t) &= -\mathcal{I}_x(x_{N_x}, y_{N_y}, t)\Delta x \\
&\quad -\mathcal{I}_y(x_{N_x}, y_{N_y}, t)\Delta y \ ,
\end{aligned}$$

where high-order terms have been neglected.

By defining the unknowns vector $v \stackrel{\text{def}}{=} [\Delta x \ \Delta y]'$ and properly defining a matrix $L \in \mathbb{R}^{N_x N_y \times 2}$ and a vector $c \in \mathbb{R}^{N_x N_y}$, the above system casts into $Lv = c$. This is an over-determined linear system of $N_x \cdot N_y$ equations in two unknowns which may be solved by the help of a total least squares technique [10]. The resulting algorithm is as follows:

1. Define the $N_x N_y \times 3$ matrix $Z = [L \ c]$;
2. Compute the SVD (U, D, V) of Z, with $V \in \mathrm{O}(3, \mathbb{R})$;
3. Define the partition $V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix}$ with $V_{12} \in \mathbb{R}^{2\times 1}$;
4. Estimate v as $-V_{12}V_{22}^{-1}$.

3 Neural SVD learning algorithm

Denoting as $Z \in \mathbb{C}^{m\times n}$ the matrix whose SVD is to be computed and as $r \leq \min\{m, n\}$ the rank of Z, its singular value decomposition writes $Z = UDV^*$, where $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ are unitary matrices and D is a pseudo-diagonal matrix that has all-zero values except for the first r diagonal entries, termed singular values. It is easily checked that the columns of U coincide with the eigenvectors of ZZ^* while V contains the eigenvectors of Z^*Z with the same eigenvalues.

Here we consider the Helmke and Moore (HM) algorithm [11], which was studied in details in [9]. This algorithm is utilized to train in an unsupervised way a neural network.

3.1 Learning differential equations

The HM dynamics arises from the maximization of a specific criterion $\Phi_W : \mathrm{O}(m,\mathbb{C})\times\mathrm{O}(n,\mathbb{C}) \to \mathbb{R}$ defined as:

$$\Phi_W(A,B) \stackrel{\mathrm{def}}{=} 2\mathrm{Re}\,\mathrm{tr}(WA^*ZB) \ , \tag{2}$$

where $W \in \mathbb{R}^{n\times m}$ is a weighting kernel and $Z \in \mathbb{C}^{m\times n}$ is the matrix whose (complex-valued) SVD is looked for, in the hypothesis that $m \geq n$. The dynamical system, derived as a Riemannian gradient flow on $\mathrm{O}(m,\mathbb{C})\times\mathrm{O}(n,\mathbb{C})$, reads:

$$\left\{ \begin{array}{l} \dot{A} = A(W^*B^*Z^*A - A^*ZBW) \ , \ A(0) = A_0 \in \mathrm{O}(m,\mathbb{C}) \ , \\ \dot{B} = B(WA^*ZB - B^*Z^*AW^*) \ , \ B(0) = B_0 \in \mathrm{O}(n,\mathbb{C}) \ . \end{array} \right. \tag{3}$$

By construction it holds $A(t) \in \mathrm{O}(m,\mathbb{C})$ as well as $B(t) \in \mathrm{O}(n,\mathbb{C})$. The weighting matrix W has the structure $[W_1 \ 0_{n,m-n}]$, where $W_1 \in \mathbb{R}^{n\times n}$ must be diagonal with, in general, unequal entries on the diagonal [9].

3.2 Learning algorithm through integration

Whereas the continuous-time versions of the learning algorithms leave the orthogonal-group invariant, this is not true for their discrete-time counterparts, which are obtained by employing a numerical integration scheme, unless a suitable integration method is selected.

In the present case, we may employ a convenient Lie integration method drawn from geometric integration theory (see e.g. the recent contribution [1] and the previous reviews in [5, 6, 7]).

First, in the present case, we only consider learning over the real orthogonal group, therefore the learning equations for computing the SVD of the matrix P simply write as:

$$\left\{ \begin{array}{l} H = A'PB \ , \\ \dot{A} = A(W'H' - HW) \ , \ A(0) = A_0 \in \mathrm{O}(N_xN_y,\mathbb{R}) \ , \\ \dot{B} = B(WH - H'W') \ , \ B(0) = B_0 \in \mathrm{O}(3,\mathbb{R}) \ . \end{array} \right. \tag{4}$$

Also, we note that the product WH may be simplified as $W_1H(1:3,:)$ (using standard MATLAB notation), while the product HW may be computed as the composite matrix $[HW_1 \ 0_{NxNy,NxNy-3}]$.

If we denote by s the generic learning step index, the learning algorithm may thus be written as:

$$\left\{ \begin{array}{l} H_s = A_s'PB_s \ , \\ (HW)_s \stackrel{\mathrm{def}}{=} [H_sW_1 \ 0_{NxNy,NxNy-3}] \ , \ (T_a)_s \stackrel{\mathrm{def}}{=} (HW)_s' - (HW)_s \ , \\ A_{s+1} = A_s \exp(\eta(T_a)_s) \ , \ A_0 \in \mathrm{O}(N_xN_y,\mathbb{R}) \ , \\ (WH)_s \stackrel{\mathrm{def}}{=} W_1H_s(1:3,:) \ , \ (T_b)_s \stackrel{\mathrm{def}}{=} (WH)_s - (WH)_s' \ , \\ B_{s+1} = B_s \exp(\eta(T_b)_s) \ , \ B_0 \in \mathrm{O}(3,\mathbb{R}) \ , \end{array} \right. \tag{5}$$

with η being a convenient learning step-size (or integration step).

The exponential map 'exp' in this case lifts the Lie algebra of the skew-symmetric matrices to the associated orthogonal group and the above expressions ensure that the state-matrices A and B keep within the respective orthogonal groups up to machine precision. The efficient computation of the matrix exponential is a sensitive point in geometric integration and there exist many different ways of performing exponentiation, which differ by their computational burden [1, 7]. The selection of an efficient exponentiation method, tailored to the considered problem, is an interesting topic, that, however, falls outside the scope of the present contribution.

In the present contribution, as it is desirable that the matrix B is computed accurately for optical flow estimation purposes, for the second of equations (5) we relied on MATLAB's expm primitive; conversely, as the matrix A do not need to be computed accurately and the matrix T_a is generally very large, in the first of equations (5) we invoked the (rather coarse) Taylor approximation truncated to first order, namely we used:

$$\exp(\eta(T_a)_s) \approx I_{Nx \cdot Ny} + \eta(T_a)_s \ . \tag{6}$$

4 Numerical experiments

A useful performance measure for the numerical algorithm just explained is the norm of the off-diagonal part of the argument-matrix of the criterion (2), particularized to the problem at hand. Namely, we may define an index as:

$$\delta(A, B) \stackrel{\text{def}}{=} \|\text{offdiag}(WA'ZB)\|_{\text{F}} \ , \tag{7}$$

which may be computed at any step s. It is interesting to note that the above-defined measure is 'blind' in the sense that it is able to measure how far the algorithm is from the sought solution without actually knowing it.

Furthermore, the course of the learning criterion function is by itself a good indicator of the network's internal state of activation.

Also, as a general-purpose quality measure we may consider two indices that take into account the loss of orthogonality of the matrices A and B, namely:

$$n(A) \stackrel{\text{def}}{=} \|A'A - I_m\|_{\text{F}} \ , \ n(B) \stackrel{\text{def}}{=} \|B'B - I_n\|_{\text{F}} \ . \tag{8}$$

As a toy example, which mainly aims at verifying the effect of the considered numerical integration method, we considered a real-valued randomly-generated matrix Z having $m = 100$ and $n = 3$. In this case we know in advance the true SVD-pair (U, V) and may therefore compare the results provided by the neural network with the exact result. In this special case, if A_n, B_n, U_n and V_n denote the sub-matrices formed by the first n columns of the SVD and network matrices, it is known that the columns of A_n should tend to the columns of U_n, while the columns of B_n should tend to the columns of B_n, ordered in the

same way but with a possible sign switch for every column; therefore, a proper measure of (A, B) convergence is:

$$\epsilon(A_n) \stackrel{\text{def}}{=} \||U_n| - |A_n|\|_{\text{F}} \ , \ \epsilon(B_n) \stackrel{\text{def}}{=} \||V_n| - |B_n|\|_{\text{F}} \ , \tag{9}$$

where $|X|$ stands for component-wise absolute-value extraction. These are termed 'subspace errors'.

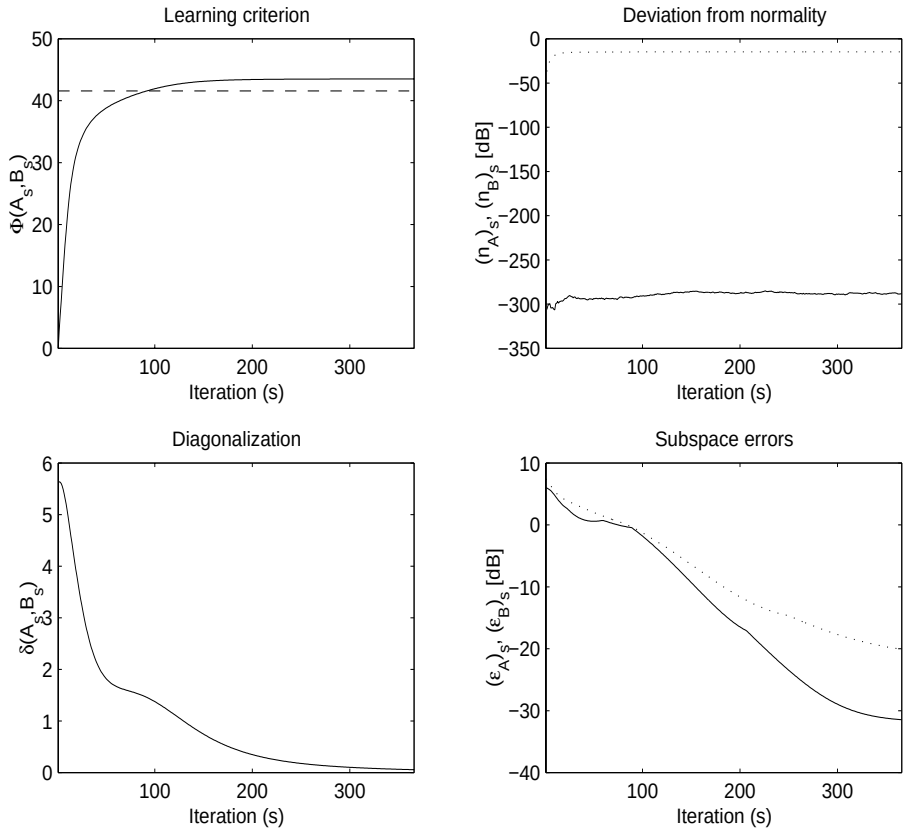

Figure 1: Results of toy experiment with a randomly-generated 100×3 matrix. (The values on the vertical axis of the right-hand panels are expressed in decibels, that is $20 \log_{10}(\cdot)$.)

The result of this toy experiment is illustrated in the Figure 1. As it is readily seen, the singular values of Z are slightly over-estimated. However, the matrix B is rather well-adherent to the true value and orthogonal to machine precision. The iteration stopped when the ratio $\delta(A_s, B_s)/\delta(A_0, B_0) < 0.01$, namely when the residual off-diagonality is less than 1% of the initial off-diagonality.

In the experiment with real-world data, we consider a sequence of two images, drawn from the public database [20]. The image support has dimension $256 \times$

256. The images are represented with $0 \div 255$-range integer numbers capturing the local optical intensity, which was preventively scaled down of a factor 50 for numerical purposes. The sequence pertains to Walter Cronkite's screenplay in which he is rotating his head to his right. Note that the images were digitized from an original low-quality support, therefore they are quite noisy: This makes the optical flow estimation a rather difficult task.

Here we illustrate the behavior of the algorithm by showing two subsequent images with superimposed motion vectors on the first one, and the off-diagonality index pertaining to integrated HM-run illustrated for two image-blocks; the block-size was $N_x = N_y = 16$ pixels.

The Figure 2 shows the courses of the performance indices $\delta(A_s, B_s)$, $n(A_s)$ and $n(B_s)$ as well as $\Phi_W(A_s, B_s)$ at every iteration-step s. The results pertain to the step-size value $\eta = -0.01$. In this case, the iteration stop criterion was $\delta(A_s, B_s)/\delta(A_0, B_0) < 0.05$.

The Figure 3 shows two considered frames and the estimated motion vectors on the three blocks. (The arrows have been scaled to exhibit the same length for illustrative purposes only.)

The obtained results are interesting and confirm the good SVD-based OF-estimation ability of the discussed PRA approach: The estimated displacement vectors look locally consistent with the platform motion, and the learning algorithm behaved properly, as confirmed by the relatively low values of the off-diagonality measure.

5 Conclusions

In the contribution [9], a unified view of four neural-network-learning-based singular-value-decomposition algorithms was considered. No attention was paid to the specific integration method of the involved learning equations, which appear under the form of first-order matrix-type ordinary differential equations on the orthogonal group or on the Stiefel manifold. In the present paper, we considered a suitable integration method, based on mathematical geometric integration theory.

The discussed algorithm was applied to optical flow computation for motion estimation in image sequences. The obtained results confirmed the quality of the mentioned approach.

Future work could be directed along a) the search of an integration method specifically tailored to the solution of the learning differential equations defined over the Cartesian product of two orthogonal groups of different dimensions and b) the search of a partial learning algorithm that allows to extend an already learnt solution for a frame-pair to the subsequent frame(s) through partial updating.

Acknowledgment

The present work has been completed while the author SF was a short-term visitor of Professor Amari's Laboratory for Mathematical Neuroscience at the Brain Science Institute (RIKEN, Japan).

References

[1] E. Celledoni and S. Fiori, *Neural Learning by Geometric Integration of Reduced 'Rigid-Body' Equations*, Journal of Computational and Applied Mathematics. Accepted for publication

[2] A. Cichocki and R. Unbehauen, *Neural networks for computing eigenvalues and eigenvectors*, Biological Cybernetics, Vol. 68, pp. 155 – 164, 1992

[3] S. Costa and S. Fiori, *Image Compression Using Principal Component Neural Networks*, Image and Vision Computing Journal (special issue on "Artificial Neural Network for Image Analysis and Computer Vision"), Vol. 19, No. 9-10, pp. 649 – 668, Aug. 2001

[4] S.G. Deshpande and J.-N. Hwang, *Fast Motion Estimation Based on Total Least Squares for Video Encoding*, Proc. of the International Symposium on Circuits and Systems, Vol. 4, pp. 114 – 117, 1998

[5] S. Fiori, *A theory for learning by weight flow on Stiefel-Grassman manifold*, Neural Computation, Vol. 13, No. 7, pp. 1625 – 1647, July 2001

[6] S. Fiori, *A Theory for Learning Based on Rigid Bodies Dynamics*, IEEE Trans. on Neural Networks, Vol. 13, No. 3, pp. 521 – 531, May 2002

[7] S. Fiori, *Fixed-Point Neural Independent Component Analysis Algorithms on the Orthogonal Group*, Future Generation Computer Systems. Accepted for publication

[8] S. Fiori, *Unsupervised Neural Learning on Lie Group*, International Journal of Neural Systems, Vol. 12, No.s 3 & 4, pp. 219 – 246, 2002

[9] S. Fiori, *Singular Value Decomposition Learning on Double Stiefel Manifold*, International Journal of Neural Systems, Vol. 13, No. 2, pp. 155 – 170, June 2003

[10] G.H. Golub and C.F. van Loan, *Matrix Computations*, The John Hopkins University Press, Third Edition, 1996

[11] U. Helmke and J. B. Moore, *Singular value decomposition via gradient and self-equivalent flows*, Linear Algebra and its Applications, Vol. 169, pp. 223 – 248, 1992

[12] G. Hori, *A general framework for SVD flows and joint SVD flows*, Proc. International Conference on Acoustics, Speech and Signal Processing, Vol. II, pp. 693 – 696, 2003

[13] L.-K. Liu and E. Feig, *A Block-Based Gradient Descent Search Algorithm for Block Motion Estimation in Video Coding*, IEEE Trans. on Circuits and Systems for Video Technology, Vol. 6, No. 4, pp. 419 – 422, Aug. 1996

[14] W.-S. Lu, H.-P. Wang and A. Antoniou, *Design of two-dimensional FIR digital filters by using the singular value decomposition*, IEEE Trans. on Circuits and Systems, Vol. CAS-37, pp. 35 – 46, Jan. 1990

[15] B.C. Moore, *Principal component analysis in linear systems: Controllability, observability and model reduction*, IEEE Trans. on Automatic Control, Vol. AC-26, No. 1, pp. 17 – 31, 1981

[16] O. Nestares and R. Navarro, *Probabilistic estimation of optical flow in multiple band-pass directional channels*, Image and Vision Computing journal, Vol. 19, No. 6, pp. 339 – 351, Apr. 2001

[17] T.D. Sanger, *Two iterative algorithms for computing the singular value decomposition from input/output samples*, in J.D. Cowan, G. Tesauro and J. Alspector (Ed.s), Advances in Neural Processing Systems, Vol. 6, pp. 1441 – 151, Morgan-Kauffman Publishers, 1994

[18] M. Salmeron, J. Ortega, C.G. Puntonet and A. Prieto, *Improved RAN sequential prediction using orthogonal techniques*, Neurocomputing, Vol. 41, No. 1-4, pp. 153 – 172, Oct. 2001

[19] S.T. Smith, *Dynamic system that perform the SVD*, Systems and Control Letters, Vol. 15, pp. 319 – 327, 1991

[20] USC-SPC Image Database, *Sequences: Walter Cronkite*, Signal & Image Processing Institute, Electrical Engineering Department, University of Southern California. URL: http://sipi.usc.edu/service/database/Database.html

[21] A. Weingessel, *An Analysis of Learning Algorithms in PCA and SVD Neural Networks*, Ph.D. Dissertation, Technical University of Wien, Austria, Feb. 1999

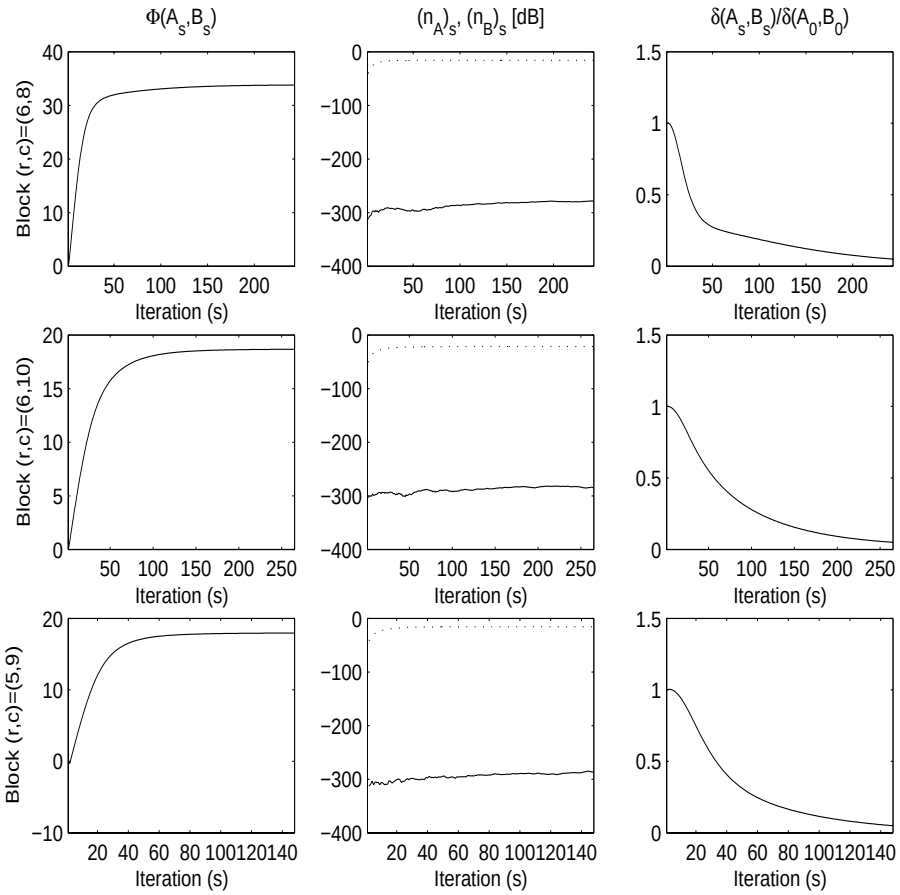

Figure 2: Courses of the SVD-neural-computation task performance indices on 'Walter Cronkite' sequence for three selected blocks. The indicated blocks' coordinates (r, c) denote the centers of the blocks in N_x and N_y units. (The values on the vertical axis of the central panels are expressed in decibels, that is $20 \log_{10}(\cdot)$.)

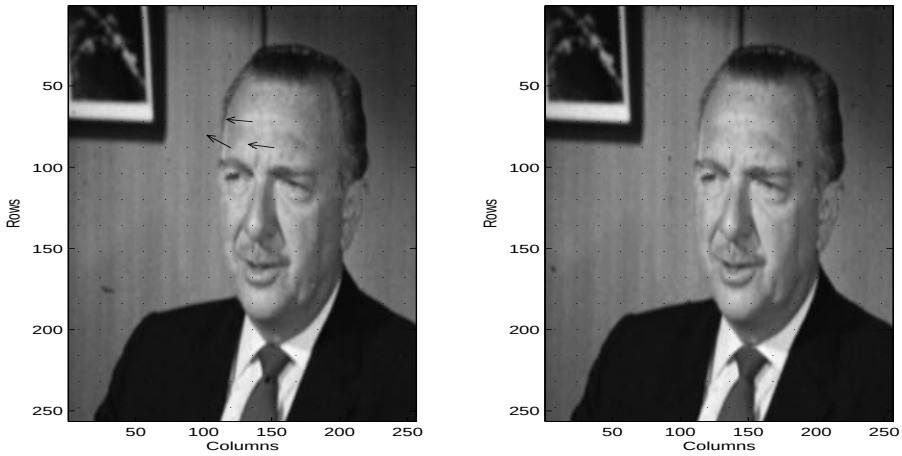

Figure 3: Two subsequent frames on a sequence of images. The arrows on the earlier image denote the estimated motion-vectors.