ELSEVIER

Signal Processing: *Image Communication* 16 (2001) 837–857

SIGNAL PROCESSING:
IMAGE
COMMUNICATION

www.elsevier.nl/locate/image

An illumination invariant algorithm for subpixel accuracy image stabilization and its effect on MPEG-2 video compression

Çiğdem Eroğlu Erdem[a,]*, A. Tanju Erdem[b]

[a] *Department of Electrical and Electronics Engineering, Boğaziçi University, İstanbul, Turkey*
[b] *Momentum Digital Media Technologies, İstanbul, Turkey*

Received 10 October 1999; received in revised form 23 February 2000; accepted 18 July 2000

Abstract

This paper offers a fast and simple near-closed-form solution for the least mean-squared-error (LMSE) estimation of the frame-to-frame global subpixel motion in an unsteady image sequence. The offered near-closed-form solution achieves unlimited subpixel accuracy by always employing a small and fixed number of computations, independent of the desired subpixel accuracy. The algorithm is designed so that it is insensitive to frame-to-frame intensity variations, which is a distinctive feature of the method. Experimental results demonstrate the superiority of the proposed method to the spatio-temporal differentiation and surface fitting algorithms under different illumination conditions. This paper furthermore discusses the effect of the proposed image stabilization algorithm on the performance of MPEG-2 video compression. We report that removal of global motion down to subpixel accuracy from an unsteady video improves MPEG-2 compression performance significantly (by at least 1 dB for all frame types), in spite of the fact that motion vectors are differentially encoded in MPEG-2. This result is supported via experimental results using the proposed subpixel registration algorithm and an analysis of the macroblock coding preferences accepted in MPEG-2. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Unsteadiness correction; Image registration; Image stabilization; MPEG-2; Preprocessing; Video coding; Motion estimation

1. Introduction

Image unsteadiness in a video or a film sequence may be caused by any unwanted or unpredictable relative movements of a camera and a scene during the recording of the scene, or that of a scanner and a motion picture film during the digitization of the film. In such applications, image stabilization problem refers to finding the global motion of each frame in the sequence with respect to a reference frame, and then correcting for each frame with the found motion parameters. With the ever increasing use of the MPEG-2 video compression for storing, broadcasting and communicating digital video data, the question of whether image

* Corresponding author.
E-mail addresses: erogluc@boun.edu.tr (Ç.E. Erdem), tanju@momentum-dmt.com (A.T. Erdem).

0923-5965/01/$ - see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0923-5965(00)00046-1

stabilization prior to MPEG-2 compression would increase the quality of the compressed video becomes very important.

The displacement (translational motion) causing the unsteadiness in image sequences will, in general, have a fractional (i.e., subpixel) part as well as an integer (i.e., pixel) part. The integer part of the displacement can be found using one of the well-known techniques in [1], such as the phase correlation technique [10]. In this paper, we are interested in estimating the subpixel part of the motion (translation and/or affine) given its pixel part even if there are illumination variations between the frames. It is indeed necessary to estimate the motion down to subpixel accuracy, because subpixel translations in a sequence may cause a disturbing jitter, especially in stationary scenes.

Subpixel image registration techniques for translational motion are discussed in [10,13,15], and references therein. These techniques can be broadly classified as those that are based on intensity matching using least mean-squared-error (LMSE) criterion, those that employ spatio-temporal differentiation, and those that fit a parametric surface to the cross correlation or phase correlation functions [15]. Although the differentiation and surface fitting approaches to subpixel motion estimation result in closed-form solutions [15], the LMSE algorithms proposed in the literature offer mostly search-based solutions, where bilinear interpolation and mean-squared error computations are repeated for every 2-D subpixel displacement being tested by the particular search algorithm. Search-based solutions require significantly more computational time than the closed-form solutions, and the computational time increases with the increased subpixel accuracy. There are also image registration techniques that do not fall into the above categories. A search-based image registration technique is presented by Thevenaz et al. [14], which is based on a modified iterative Marquardt–Levenberg algorithm for nonlinear least-squares optimization of the residue. Kim et al. [9] achieves subpixel accuracy image registration in the frequency domain via spectrum cancellation. Schultz and Alford [12] also addresses the problem of camera motion estimation in an iterative least-squares approach. Sawhney and Ayer [11] solve the problem of dominant motion estimation using an iterative least-squares procedure based on Gauss–Newton formulation. Finally, Zheng and Chelappa [17,18] try to estimate some camera motion parameters by estimating the illuminant direction. This approach assumes that the illuminant direction remains the same in an image sequence.

We propose in this paper, a fast and simple near-closed-form solution (i.e., no iterations and search is involved) for the LMSE estimation of the subpixel displacement between two frames of an unsteady image sequence. We do not assume that the illumination is the same in all the frames to be processed. Therefore, we extend our method to handle brightness and contrast variations between the frames. The method can be extended to estimate the affine motion parameters between two frames from three point correspondences. The performance of the method presented is independent of the motion content of the image if an appropriate background block is chosen for processing. Simulations on real and synthetic sequences show that the overall estimation accuracy is better than spatio-temporal differentiation, search-based and surface-fitting algorithms and it is faster than exhaustive search.

In this paper, we also present that removal of subpixel global camera motion (using the proposed method) prior to MPEG-2 coding of an unsteady video sequence increase the quality of the compressed video considerably (at least 1 dB improvement for all frame types). We provide detailed explanations and simulation results on real sequences as to why the MPEG-2 algorithm responds favorably to subpixel image stabilization.

The outline of the paper is as follows. The problem formulation and the notation that will be used in subsequent sections is introduced in Section 2. A near-closed-form solution to image stabilization that removes the computational burden on the LMSE estimation is introduced in Section 3. The extension of the method that is insensitive to intensity variations between frames, i.e. illumination effects, is proposed in Section 4. An application of the method for estimating the affine motion parameters is discussed in Section 5. Experimental results that demonstrate the performance of the proposed subpixel image stabilization algorithm on synthetic and real video sequences are presented in Section 6. The coding improvements in MPEG-2 achieved by using the proposed subpixel stabilization algorithm and an analysis of the MPEG-2 bit allocation scheme in order to explain the coding improvement due to image stabilization are discussed in Section 7. Conclusions are given in Section 8.

2. Problem formulation for image stabilization

Let $s_1(\ )$ denote the displaced frame and $s_2(\ )$ denote the unsteady frame after having been corrected for any integer pixel displacement, say, by using the phase correlation method. Then, $s_1(\ )$ and $s_2(\ )$ differ from each other only by a subpixel displacement (d_1, d_2) (assuming that the only cause of misregistration is a displacement), i.e.,

$$s_1(n_1, n_2) = s_2(n_1 + d_1, n_2 + d_2), \quad -1 < d_1, d_2 < 1. \tag{1}$$

In general, bilinear interpolation is employed to approximate the value of $s_2(n_1 + d_1, n_2 + d_2)$. That is,

$$s_2(n_1 + d_1, n_2 + d_2) \approx \tilde{s}_2^{(i)}(n_1, n_2;\ d_1, d_2) = S_0^{(i)} + S_1^{(i)} d_1 + S_2^{(i)} d_2 + S_3^{(i)} d_1 d_2, \quad (d_1, d_2) \in \mathscr{Q}^{(i)}, \tag{2}$$

where $\mathscr{Q}^{(i)}$, $i = 1, 2, 3, 4$, denote the four quadrants defined as

$$\begin{aligned} &\mathscr{Q}^{(1)} = \{(d_1, d_2){:}\, 0 \leqslant d_1, d_2 < 1\}, \qquad \mathscr{Q}^{(2)} = \{(d_1, d_2){:}\, 0 \leqslant d_1 < 1, -1 < d_2 < 0\}, \\ &\mathscr{Q}^{(3)} = \{(d_1, d_2){:}\, -1 < d_1 < 0, 0 \leqslant d_2 < 1\}, \qquad \mathscr{Q}^{(4)} = \{(d_1, d_2){:}\, -1 < d_1, d_2 < 0\}, \end{aligned} \tag{3}$$

and the coefficients $S_0^{(i)}, S_1^{(i)}, S_2^{(i)}, S_3^{(i)}$ are functions of the intensities at pixels neighboring to (n_1, n_2); and they are defined by

$$\begin{aligned} S_0^{(i)} &= s_2(n_1, n_2), \\ S_1^{(i)} &= k[s_2(n_1 + k, n_2) - s_2(n_1, n_2)], \\ S_2^{(i)} &= l[s_2(n_1, n_2 + l) - s_2(n_1, n_2)], \\ S_3^{(i)} &= kl[s_2(n_1 + k, n_2 + l) - s_2(n_1 + k, n_2) - s_2(n_1, n_2 + l) + s_2(n_1, n_2)], \end{aligned} \tag{4}$$

where

$$k = \begin{cases} 1 & \text{for } i = 1, 2 \\ -1 & \text{for } i = 3, 4 \end{cases} \quad \text{and} \quad l = \begin{cases} 1 & \text{for } i = 1, 3, \\ -1 & \text{for } i = 2, 4. \end{cases} \tag{5}$$

The MSE for each quadrant is then defined as

$$\mathrm{MSE}^{(i)}(d_1, d_2) = \frac{1}{N_1 N_2} \sum_{n_1, n_2 \in \mathscr{B}} [s_1(n_1, n_2) - \tilde{s}_2^{(i)}(n_1, n_2; d_1, d_2)]^2, \quad (d_1, d_2) \in \mathscr{Q}^{(i)},\ i = 1, 2, 3, 4, \tag{6}$$

where $\mathscr{B}$ denotes an $N_1 \times N_2$ block of pixels over which the MSE is computed. When we substitute (2) in Eq. (6), the expression inside the square brackets becomes

$$(s_1(n_1, n_2) - S_0^{(i)}) + (-S_1^{(i)}) d_1 + (-S_2^{(i)}) d_2 + (-S_3^{(i)}) d_1 d_2, \tag{7}$$

which is bilinear in d_1 and d_2.

The procedure to find $-1 \leqslant d_1,\ d_2 \leqslant 1$ that results in the minimum MSE can then be stated as: Find $(\hat{d}_1^{(i)}, \hat{d}_2^{(i)})$ in $\mathscr{Q}^{(i)}$ that minimizes $\mathrm{MSE}^{(i)}(d_1, d_2)$, $i = 1, 2, 3, 4$. Then, pick the pair $(\hat{d}_1, \hat{d}_2)$ that results in the overall minimum MSE. That is,

$$(\hat{d}_1, \hat{d}_2) = \arg\min_i\ \mathrm{MSE}(\hat{d}_1^{(i)}, \hat{d}_1^{(i)}). \tag{8}$$

A straightforward approach taken to minimize (6) is to uniformly sample the set $\{(d_1, d_2): -1 \leqslant d_1, d_2 \leqslant 1\}$ at a desired accuracy, compute the MSE given in (6) for every sample pair (d_1, d_2), and pick the pair that minimizes the MSE. In exhaustive (full) search, all possible locations up to the desired accuracy are tested and the subpixel displacement which minimizes the MSE is chosen. Hence, if an accuracy of 2^{-n} pixels is desired, the exhaustive search requires the evaluation of (6) for $(2^{n+1}-1)^2$ different values of (d_1, d_2) pairs. This corresponds to N_1N_2 bilinear interpolations for each (d_1, d_2) pair, which results in a total of $9N_1N_2(2^{n+1}-1)^2$ multiplications and $6N_1N_2(2^{n+1}-1)^2$ summations. Since n appears as the power in these expressions, the number of multiplications and summations increase by approximately 16 times when n is doubled. This brings a large computational load which can be significantly reduced by using the logarithmic search technique, which requires $9N_1N_2[9+8(n-1)]$ multiplications and $6N_1N_2[9+8(n-1)]$ summations for an accuracy of 2^{-n} pixels. Hence, both exhaustive and logarithmic search techniques are time-consuming algorithms, and the computational time increases with the increased subpixel accuracy.

In the next section, we propose a near-closed-form solution to image stabilization that removes the computational burden on the LMSE estimation.

3. A near-closed-form solution for LMSE subpixel accuracy image stabilization

From (6) and (2) we obtain the following expression for $\mathrm{MSE}^{(i)}$ in terms of the subpixel shifts d_1 and d_2:

$$\mathrm{MSE}^{(i)}(d_1,d_2) = C_0^{(i)} + C_1^{(i)}d_1 + C_2^{(i)}d_2 + C_3^{(i)}d_1d_2 + C_4^{(i)}d_1^2 + C_5^{(i)}d_2^2 + C_6^{(i)}d_1^2d_2 + C_7^{(i)}d_1d_2^2 + C_8^{(i)}d_1^2d_2^2, \quad i = 1,2,3,4, \tag{9}$$

where (i) denotes one of the four quadrants in the Cartesian coordinates as defined in Section 2, and the coefficients $C_0^{(i)}, \ldots, C_8^{(i)}$ are computed over the two images using the basic summations as described in Appendices A and B.

In order to minimize $\mathrm{MSE}^{(i)}$ with respect to d_1 and d_2, we solve $\partial\mathrm{MSE}^{(i)}/\partial d_1 = 0$ and $\partial\mathrm{MSE}^{(i)}/\partial d_2 = 0$ simultaneously,

$$\frac{\partial\mathrm{MSE}^{(i)}}{\partial d_1} = C_1^{(i)} + C_3^{(i)}d_2 + 2C_4^{(i)}d_1 + 2C_6^{(i)}d_1d_2 + C_7^{(i)}d_2^2 + 2C_8^{(i)}d_1d_2^2 = 0, \tag{10}$$

$$\frac{\partial\mathrm{MSE}^{(i)}}{\partial d_2} = C_2^{(i)} + C_3^{(i)}d_1 + 2C_5^{(i)}d_2 + C_6^{(i)}d_1^2 + 2C_7^{(i)}d_1d_2 + 2C_8^{(i)}d_1^2d_2 = 0. \tag{11}$$

We note that Eq. (10) is linear in d_1. Thus we can express d_1 as a function of d_2 as

$$d_1 = -0.5\frac{C_1^{(i)} + C_3^{(i)}d_2 + C_7^{(i)}d_2^2}{C_4^{(i)} + C_6^{(i)}d_2 + C_8^{(i)}d_2^2}. \tag{12}$$

Then, we substitute (12) in Eq. (11), to obtain the following polynomial equation in d_2:

$$E_5d_2^5 + E_4d_2^4 + E_3d_2^3 + E_2d_2^2 + E_1d_2 + E_0 = 0, \tag{13}$$

where the coefficients $E_0, \ldots, E_5$ are defined in terms of $C_0, \ldots, C_8$. The definitions of $E_0, \ldots, E_5$ are given in Appendix C. Unfortunately, there does not exist an algebraic formula for the zeros of a fifth-degree polynomial. Thus, the zeros of (13) are obtained numerically using the Muller's method [2]. Once the solution for d_2 is obtained, d_1 is calculated from (12).

Since (13) is a fifth-degree polynomial, for each quadrant $\mathscr{Q}^{(i)}$, at least one of the roots will be real and the remaining two pairs may be complex conjugates of each other. Among the roots obtained for quadrant $\mathscr{Q}^{(i)}$, only the solutions (d_1, d_2) that are in $\mathscr{Q}^{(i)}$ are accepted. In the case there is more than one acceptable solution for (d_1, d_2) considering all quadrants, the solution with the minimum MSE is picked to be the actual subpixel displacement. On the other hand, when there is no acceptable solution at all – this actually happened very rarely in our experiments – the proposed algorithm defaults to an efficient exhaustive search method which uses (9) instead of (6) to find the subpixel displacement (hence the name *near*-closed-form solution).

Thus the steps of the proposed algorithm can then be summarized as follows:

1. Compute the basic summations $A_{0,0}$, $A_{0,0;0,0}$, $B_{0,0;i,j}$, $D_{i,j}$, $D_{i,j;k,l}$, given in Appendix A over a specified block of pixels. Note that only 49 basic summations are computed at this step.
2. Compute the MSE coefficients $C_0^{(i)}, \dots, C_8^{(i)}$, given in Appendix B for each quadrant, i.e., for each $i = 1, 2, 3, 4$.
3. Compute the coefficients $E_0, \dots, E_5$, of the fifth-degree polynomial as given in Appendix C for each quadrant.
4. Find the zeros of (13) for each quadrant. Among the acceptable ones, pick the one with the minimum MSE. That gives the near-closed-form solution. If there is no solution, find (d_1, d_2) which minimizes the MSE expression (9) using an efficient exhaustive search method [3].

The order of the computational complexity of the proposed method is approximately $5N_1N_2$ summations, $3N_1N_2$ multiplications plus the complexity of the root-finding algorithm. This number does not depend on the accuracy of the desired subpixel stabilization and varies linearly with the size of the chosen block.

In the next section, an extension of the method that is insensitive to intensity variations between frames is presented.

4. Accounting for intensity variations

In estimating the subpixel displacement of a frame with respect to the reference frame, it is important to account for intensity variations, that are due to illumination changes, between the current frame and the reference frame. We assume that the intensity I_c of a pixel in the current frame is related to the intensity I_r in the reference frame by

$$I_c = \gamma I_r + \eta, \tag{14}$$

where γ and η are called the contrast and brightness parameters, respectively. Thus, in order to account for intensity variations, we modify the mean squared error expression given in (6) as

$$\mathrm{MSE}^{(i)}(d_1, d_2) = \frac{1}{N_1 N_2} \sum_{n_1, n_2 \in \mathscr{B}} [\gamma s_1(n_1, n_2) + \eta - \tilde{s}_2^{(i)}(n_1, n_2; d_1, d_2)]^2, \quad (d_1, d_2) \in \mathscr{Q}^{(i)}, \ i = 1, 2, 3, 4. \tag{15}$$

We note that in addition to d_1 and d_2, two new parameters, namely γ and η, need to be determined for each frame.

We use the approach suggested in [5] to first find the optimal solution for γ and η in terms of d_1 and d_2 by setting $\partial\mathrm{MSE}/\partial\gamma = 0$ and $\partial\mathrm{MSE}/\partial\eta = 0$. That is,

$$\begin{aligned} \frac{\partial \mathrm{MSE}^{(i)}}{\partial \gamma} &= \sum_{n_1, n_2 \in \mathscr{B}} [\gamma s_1(n_1, n_2) + \eta - \tilde{s}_2^{(i)}(n_1, n_2; d_1, d_2)] s_1(n_1, n_2) = 0, \\ \frac{\partial \mathrm{MSE}^{(i)}}{\partial \eta} &= \sum_{n_1, n_2 \in \mathscr{B}} [\gamma s_1(n_1, n_2) + \eta - \tilde{s}_2^{(i)}(n_1, n_2; d_1, d_2)] = 0. \end{aligned} \tag{16}$$

Note that these equations are linear in γ and η,

$$\begin{bmatrix} \sum s_1^2(n_1,n_2) & \sum s_1(n_1,n_2) \\ \sum s_1(n_1,n_2) & N_1N_2 \end{bmatrix} \begin{bmatrix} \gamma \\ \eta \end{bmatrix} = \begin{bmatrix} \sum s_1(n_1,n_2)\tilde{s}_2^{(i)}(n_1,n_2;d_1,d_2) \\ \sum \tilde{s}_2^{(i)}(n_1,n_2;d_1,d_2) \end{bmatrix}. \tag{17}$$

If we substitute the bilinear interpolation expression given in (2) for $\tilde{s}_2^{(i)}(n_1,n_2;d_1,d_2)$ in (17), then the solution (γ^*,η^*) to (17) becomes a function of d_1 and d_2 in the following form:

$$\gamma^* = G_0^{(i)} + G_1^{(i)}d_1 + G_2^{(i)}d_2 + G_3^{(i)}d_1d_2, \qquad \eta^* = H_0^{(i)} + H_1^{(i)}d_1 + H_2^{(i)}d_2 + H_3^{(i)}d_1d_2, \tag{18}$$

where the coefficients $G_0^{(i)},\dots,H_3^{(i)}$ are given in terms of the basic summations that are defined in Appendix A. The actual expressions for $G_0^{(i)},\dots,H_3^{(i)}$ in terms of the basic summations are provided in Appendix D. We note that the optimal values γ^* and η^* given in (18) are bilinear in d_1 and d_2. Thus, when (2) and the expression for (γ^*,η^*) are substituted in (15), the expression within the square brackets in (15) will still be bilinear in d_1 and d_2,

$$\begin{aligned} &(G_0^{(i)}s_1(n_1,n_2) + H_0^{(i)} - S_0^{(i)}) + (G_1^{(i)}s_1(n_1,n_2) + H_1^{(i)} - S_1^{(i)})d_1 \\ &\quad + (G_2^{(i)}s_1(n_1,n_2) + H_2^{(i)} - S_2^{(i)})d_2 + (G_3^{(i)}s_1(n_1,n_2) + H_3^{(i)} - S_3^{(i)})d_1d_2. \end{aligned} \tag{19}$$

Comparing Eqs. (7) and (19), we can see that only the coefficients of the terms d_1, d_2, d_1d_2 and the constant term has changed. This observation is very important because we can say that when the intensity variations are incorporated, the new MSE$^{(i)}$ still has the same form as (9) in terms of d_1 and d_2,

$$\begin{aligned} \mathrm{MSE}^{(i)}(d_1,d_2) = {} & C_0^{(i)} + C_1^{(i)}d_1 + C_2^{(i)}d_2 + C_3^{(i)}d_1d_2 + C_4^{(i)}d_1^2 + C_5^{(i)}d_2^2 \\ & + C_6^{(i)}d_1^2d_2 + C_7^{(i)}d_1d_2^2 + C_8^{(i)}d_1^2d_2^2, \qquad i = 1,2,3,4. \end{aligned} \tag{20}$$

The expressions for the coefficients $C_0^{(i)},\dots,C_8^{(i)}$, in terms of the basic summations will now be different from those used in Section 3; however, they can be derived similarly. The full expressions for $C_0^{(i)},\dots,C_8^{(i)}$ can be found in [4].

In order to minimize (20) with respect to d_1 and d_2, we again solve $\partial\mathrm{MSE}/\partial d_1 = 0$ and $\partial\mathrm{MSE}/\partial d_2 = 0$ simultaneously. Carrying out the steps given in (10) and (11), we get the same expression for d_1 in terms of d_2 as given in (12). The expressions for the coefficients of the fifth-degree polynomial (13) in terms of new coefficients $C_0^{(i)},\dots,C_8^{(i)}$ will still be as given in Appendix C.

Thus, the incorporation of intensity variations into the near-closed-form solution is achieved by simply re-defining the coefficients $C_0^{(i)},\dots,C_8^{(i)}$, in terms of the basic summations. The result is a novel near-closed-form solution that is insensitive to intensity variations. The difference of the new algorithm with that of Section 3 is only in the computation of the coefficients $C_0^{(i)},\dots,C_8^{(i)}$; otherwise the two algorithms are exactly the same.

In particular, if frame-to-frame intensity variations are caused only by the changes in the brightness parameter (i.e., the contrast parameter is unchanged), then the MSE expression in (15) simplifies to

$$\mathrm{MSE}^{(i)}(d_1,d_2) = \frac{1}{N_1N_2}\sum_{n_1,n_2\in\mathscr{B}} [s_1(n_1,n_2) + \eta - \tilde{s}_2^{(i)}(n_1,n_2;d_1,d_2)]^2, \quad (d_1,d_2)\in\mathscr{Q}^{(i)},\ i = 1,2,3,4. \tag{21}$$

Then, using the procedure given above, the optimal brightness parameter η^* is found to be

$$\eta^* = \frac{1}{N_1N_2}\sum \tilde{s}_2^{(i)}(n_1,n_2;d_1,d_2) - \frac{1}{N_1N_2}\sum s_1(n_1,n_2). \tag{22}$$

Since the contrast parameter γ is assumed to be 1, we will have

$$G_0^{(i)} = 1 \quad \text{and} \quad G_1^{(i)} = G_2^{(i)} = G_3^{(i)} = 0. \tag{23}$$

The coefficients $H_1^{(i)}, H_2^{(i)}, H_3^{(i)}, H_4^{(i)}$ for the brightness parameters are given in Appendix E. The coefficients of the MSE expression (20) for this case can be obtained by substituting $H_1^{(i)}, H_2^{(i)}, H_3^{(i)}, H_4^{(i)}$ and (23) in the expressions for $C_0^{(i)}, \ldots, C_8^{(i)}$ given in [4].

On the other hand, if frame-to-frame intensity variations can be modeled using the contrast parameter only, the MSE expression in (15) simplifies to

$$\mathrm{MSE}^{(i)} = \frac{1}{N_1 N_2} \sum_{n_1, n_2 \in \mathscr{B}} [\gamma s_1(n_1, n_2) - \tilde{s}_2^{(i)}(n_1, n_2; d_1, d_2)]^2, \quad (d_1, d_2) \in \mathscr{Q}^{(i)}, \; i = 1, 2, 3, 4. \tag{24}$$

Then, using the procedure given above, the optimal contrast parameter γ^* is found to be

$$\gamma^* = \frac{\sum \tilde{s}_2^{(i)}(n_1, n_2; d_1, d_2) s_1(n_1, n_2)}{\sum s_1^2(n_1, n_2)}. \tag{25}$$

Since the contrast parameter η is assumed to be 0, we will have

$$H_0^{(i)} = H_1^{(i)} = H_2^{(i)} = H_3^{(i)} = 0. \tag{26}$$

The coefficients $G_1^{(i)}, G_2^{(i)}, G_3^{(i)}, G_4^{(i)}$ for the contrast parameters are given in Appendix F. The coefficients of the MSE expression (20) for this case can be obtained by substituting $G_1^{(i)}, G_2^{(i)}, G_3^{(i)}, G_4^{(i)}$ and (26) in the expressions for $C_0^{(i)}, \ldots, C_8^{(i)}$ given in [4].

Thus the steps of the proposed algorithm in case of intensity variations can be summarized as follows:

1. Compute the basic summations $A_{0,0}$, $A_{0,0;0,0}$, $B_{0,0;i,j}$, $D_{i,j}$, $D_{i,j;k,l}$, given in Appendix A over a specified block of pixels.
2. Compute the coefficients for contrast and brightness parameters, $G_0^{(i)}, \ldots, H_3^{(i)}$, given in Appendix D (or Appendix E or F depending on the intensity variation model) for each quadrant.
3. Compute the MSE coefficients $C_0^{(i)}, \ldots, C_8^{(i)}$ given in [4], for each quadrant, i.e., for each $i = 1, 2, 3, 4$.
4. Compute the coefficients $E_0, \ldots, E_5$, of the fifth-order polynomial as given in Appendix C for each quadrant.
5. Find the zeros of the fifth-order polynomial (13) for each quadrant. Among the acceptable ones, pick the one which gives the minimum MSE. That gives the near-closed-form solution. If there is no acceptable solution, find (d_1, d_2) that minimizes the MSE expression (20), using the efficient exhaustive search method [3] and the new definition for the coefficients $C_0^{(i)}, \ldots, C_8^{(i)}$ proposed in this section.

5. Finding the affine motion parameters

An affine motion includes a wide range of transformations which consists of translation, rotation, shear and scale operations. If there is an affine motion between two frames, we need to estimate six parameters to characterize the motion fully. These six parameters may be estimated using the correspondence of three noncollinear points in both images [16]. If the point (u_i, v_i) in the transformed image corresponds to the point (x_i, y_i) in the reference image for $i = 1, 2, 3$, then the relation between them can be written using a single matrix equation as

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} = \begin{bmatrix} u_1 & v_1 & 1 \\ u_2 & v_2 & 1 \\ u_3 & v_3 & 1 \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}. \tag{27}$$

Let us name the above matrices as X, U and T, successively. Then, in order to find the parameters of the affine motion, namely $t_{11}, t_{12}, t_{21}, t_{22}, t_{31}$ and t_{32}, we need to solve $X = UT$ for T.

The proposed subpixel displacement estimation algorithm can be used to find the corresponding point locations, i.e., the coordinates of the points (x_i, y_i) given (u_i, v_i). First, three different blocks of pixels are selected on the transformed image, and the centers of these blocks are chosen as the points $(u_1, v_1), (u_2, v_2)$, (u_3, v_3). Then, the proposed near-closed-form solution is used to estimate the translational motion between the same three blocks of the transformed and the reference images. In order to find the point (x_i, y_i) in the reference image that corresponds to the point (u_i, v_i) in the transformed image, we simply add to (u_i, v_i) the estimated displacement vector found for that block.

As the motion between corresponding blocks of the reference and transformed images are not purely translational, perfect correspondence between the points (x_i, y_i) and (u_i, v_i) may not be established by estimating the translational motion. As a consequence, the affine transformation matrix T obtained using the translation motion vectors may not be correct. The remedy for this problem is to apply translational motion estimation and affine parameter estimation over the same blocks iteratively, until the affine motion parameters between two consecutive iterations are sufficiently close to that of an identity transformation. In each iteration, the found T matrix is cascaded with the previously found matrices and the overall transformation matrix is applied to the original unsteady frame to prevent any accumulation of interpolation errors.

6. Experimental results for image stabilization

6.1. Synthetic sequences

We have tested the proposed near-closed-form method on three image sequences. One of these sequences is generated from a real CT image (shown in Fig. 1(a)). The other two sequences, namely the Text-1 and Text-2 sequences (Fig. 2), are generated from a synthetic image (shown in Fig. 1(b)). The subsequent 19 frames of all three sequences are generated by introducing random subpixel displacements to the first frames, i.e. the reference frames, shown in Figs. 1(a) and (b).

The CT sequence does not contain any intensity variations. On the other hand, we have simulated both contrast and brightness variations on the Text-1 sequence, and only contrast variations on the Text-2 sequence. In order to introduce the intensity variations to the subsequent frames of Text-1 sequence, the pixel intensities in the frames are first multiplied by γ, and η is added afterwards as given in (14). If we let k denote the frame number, the values of γ and η chosen for the kth frame are defined in our experiments as

$$\gamma(k) = 1 + \gamma_0 k, \tag{28}$$

$$\eta(k) = \eta_0 k. \tag{29}$$

For the Text-1 sequence, we have set $\gamma(k) = 1 - 0.02(k-1)$ and $\eta(k) = -2(k-1)$, whereas for the Text-2 sequence, we have set $\gamma(k) = 1 - 0.04(k-1)$ and $\eta(k) = 0$. In addition to intensity variations, a 10 dB white Gaussian noise is added to each frame in the Text-1 and Text-2 sequences to simulate any observation noise. The last frames of Text-1 and Text-2 sequences are given in Fig. 2. A 5×5 uniform blur is applied to each frame prior to subpixel displacement estimation to reduce the effects of bilinear interpolation and of any additive noise. The simulations are carried on a block of size approximately 100×100 pixels, that contains sufficient intensity variations. The motion compensation is done using cubic spline interpolation and the boundary pixels are copied from the previous frame if necessary.

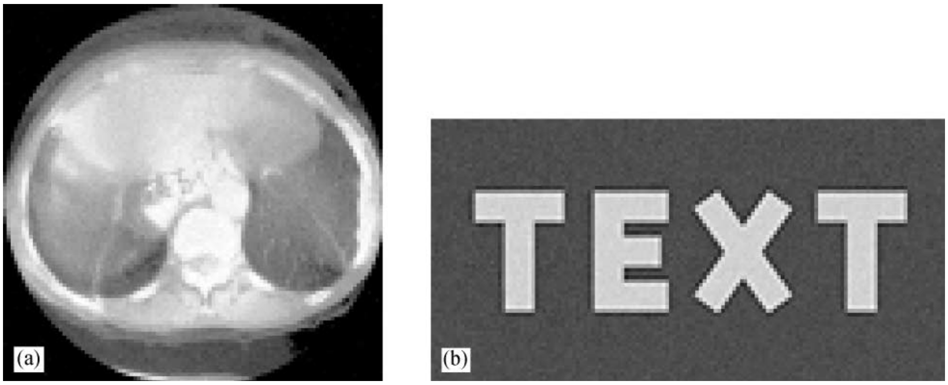


Fig. 1. (a) The first frame of the CT sequence, (b) the first frame of the Text-1 and Text-2 sequences.

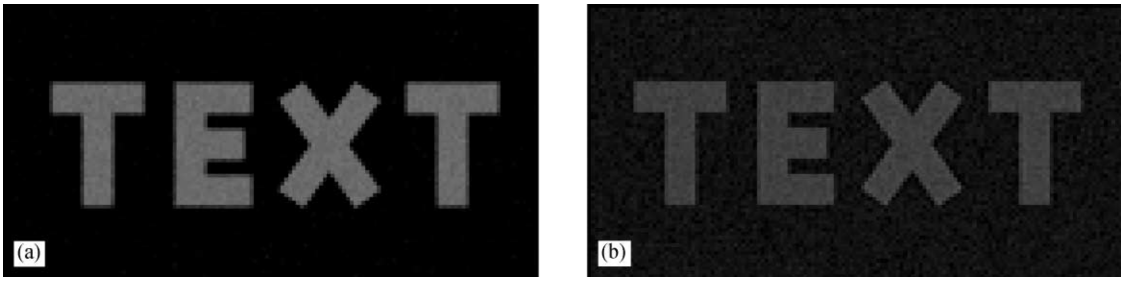


Fig. 2. (a) The last frame of the Text-1 sequence, (b) the last frame of the Text-2 sequence.

In Fig. 3, we compare the performance of the proposed near-closed-form solution to that of the exhaustive search method (the accuracy of the exhaustive search method is chosen to be 1/16 pixels). In the same figure, we also compare the performance of the proposed near-closed-form solution to that of the other closed-form solutions existing in the literature, namely, the spatio-temporal differentiation method, phase correlation surface interpolation method, and the cross correlation surface interpolation method. These techniques are discussed in detail in [10,13,15].

We observe that the cross correlation method performs nearly as well as the proposed near-closed-form solution for the Text-1 and Text-2 sequences. However, the performance of the cross correlation method degrades significantly for the CT sequence. On the other hand, the performance of the differentiation method is close to that of the proposed near-closed-form solution for the CT and the Text-1 sequences, while it degrades considerably for the Text-2 sequence. We conclude from Fig. 3 that the proposed near-closed-form solution consistently gives the best results for all the sequences considered and its performance is not affected by the severe brightness and contrast variations between the frames.

In Fig. 4, the plots of $\gamma(k)$ and $\eta(k)$ versus the frame number (k) are given for the Text-1 sequence. It can be seen that the near-closed-form solution is able to estimate the parameters $\gamma(k)$ and $\eta(k)$ for each frame (k) almost perfectly.

The CPU times for all of the methods compared in Fig. 3 are nearly the same, except for the exhaustive search method, which is significantly slower than the others.

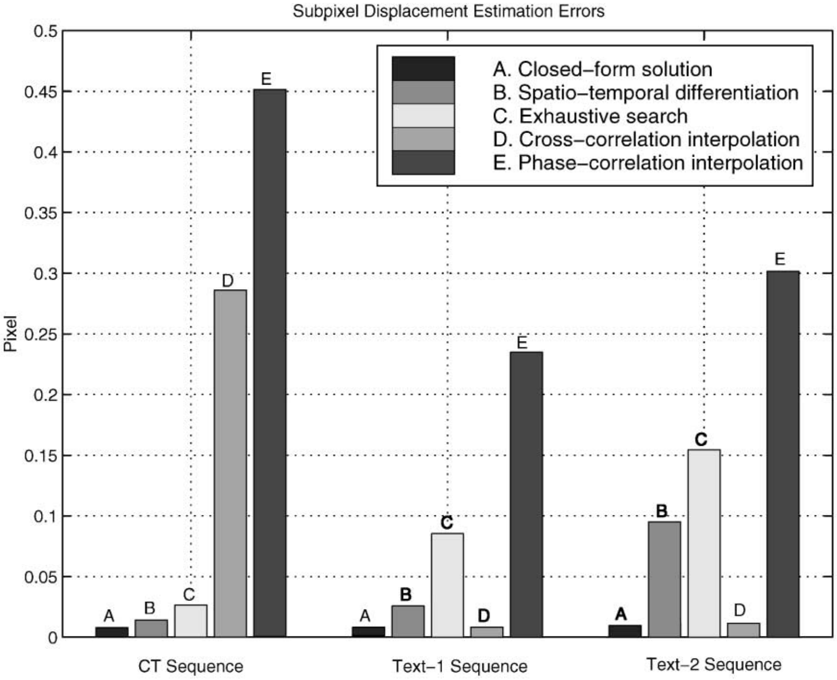


Fig. 3. The magnitude of the displacement estimation error vector ($\varepsilon(d_1)$, $\varepsilon(d_2)$), where $\varepsilon(d_1)$ is the error in d_1 and $\varepsilon(d_2)$ is the error in d_2.

We have also carried out experiments to test the performance of the proposed affine motion estimation algorithm on the CT image shown in Fig. 1(a). The transformed image is the counter-clockwise rotated version of this reference image by 10 degrees. The sizes of the three blocks are chosen as 79 × 79. The correct affine motion parameters between the rotated and reference images are given by

$$T^* = \begin{bmatrix} 0.9848 & -0.1736 & 0 \\ 0.1736 & 0.9848 & 0 \\ -40.4787 & 48.2391 & 1 \end{bmatrix}. \tag{30}$$

The affine parameter estimation algorithm is executed five times and the overall affine motion matrix T_k obtained after iteration k is

$$T_1 = \begin{bmatrix} 0.9790 & -0.1751 & 0 \\ 0.1926 & 1.0124 & 0 \\ -39.2509 & 43.6864 & 1 \end{bmatrix}, \quad T_5 = \begin{bmatrix} 0.98497 & -0.1737 & 0 \\ 0.1734 & 0.9850 & 0 \\ -39.4734 & 48.0752 & 1 \end{bmatrix}. \tag{31}$$

It can be seen from the above matrices that, as the iteration number k increases, the matrix T_k becomes a closer estimate of the correct matrix T^*. The incremental matrices A_k's after the kth iteration are

$$A_2 = \begin{bmatrix} 1.0036 & -0.0004 & 0 \\ -0.0152 & 0.9706 & 0 \\ -0.3415 & 4.9069 & 1 \end{bmatrix}, \quad A_5 = \begin{bmatrix} 1.0001 & 0.0001 & 0 \\ 0.0002 & 1.0001 & 0 \\ -0.0449 & -0.0568 & 1 \end{bmatrix}. \tag{32}$$

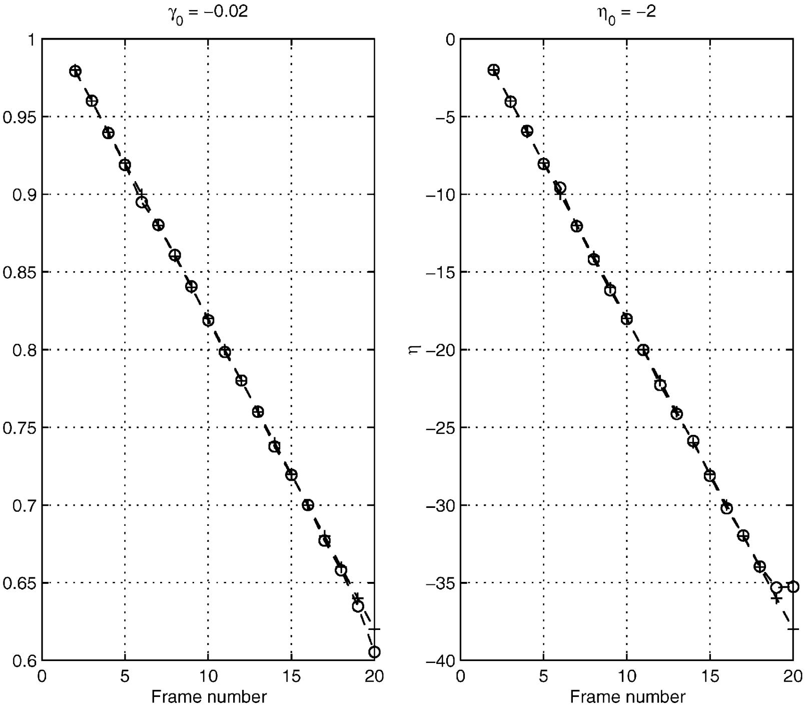


Fig. 4. The γ and η values found for the Text-1 sequence in which $(\gamma_0, \eta_0) = (-0.02, -2)$. The "o" and " + " signs denote the estimated and true values for γ (on the left) and η (on the right), respectively.

Note that $T_k = A_k A_{k-1} \ldots A_2 T_1$, for $k \geqslant 2$. It can be observed that the off-diagonal parameters of the incremental matrices A_k monotonically get smaller after each iteration and hence A_k approaches to the identity matrix. This shows that the proposed algorithm converges to the correct affine motion parameters.

6.2. Real sequences

In addition to the synthetically generated sequences, we tested the algorithm with two real unsteady sequences. The first real sequence, namely the Bilkent sequence, was recorded using a hand-held camcorder (SONY video-8). Unknown displacements occur between the frames of the Bilkent sequence due to the movements of the hand of the person holding the camcorder during recording. These displacements contain large pixel parts (between 8 and 16 pixels) as well as subpixel components. The first and the tenth frames of the unregistered Bilkent sequence are given in Figs. 5(a) and (b), respectively. Note that there is also rotational motion in the Bilkent sequence. The tenth frame of the Bilkent sequence after the correction of the rotation and the subpixel motion with respect to the first frame is given in Fig. 5(c). We observe that the proposed algorithm is successful in eliminating the global affine and subpixel motion in the sequence. In order to quantify the success of the subpixel stabilization, the MSE between the luminance components of the first frame and the subsequent frames are given in Fig. 6.

The second real test sequence, namely the Bilkent-2 sequence, is similar to the first one, except that it is recorded under changing illumination conditions. The pixel part of the estimated motion with respect to the

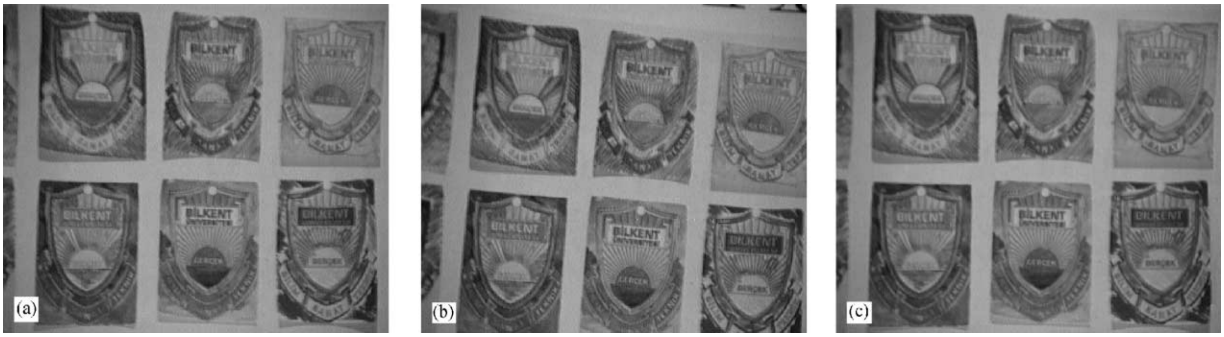


Fig. 5. (a) The first frame of the Bilkent sequence, (b) the tenth frame of the Bilkent sequence, (c) the tenth frame of the Bilkent sequence after stabilization.

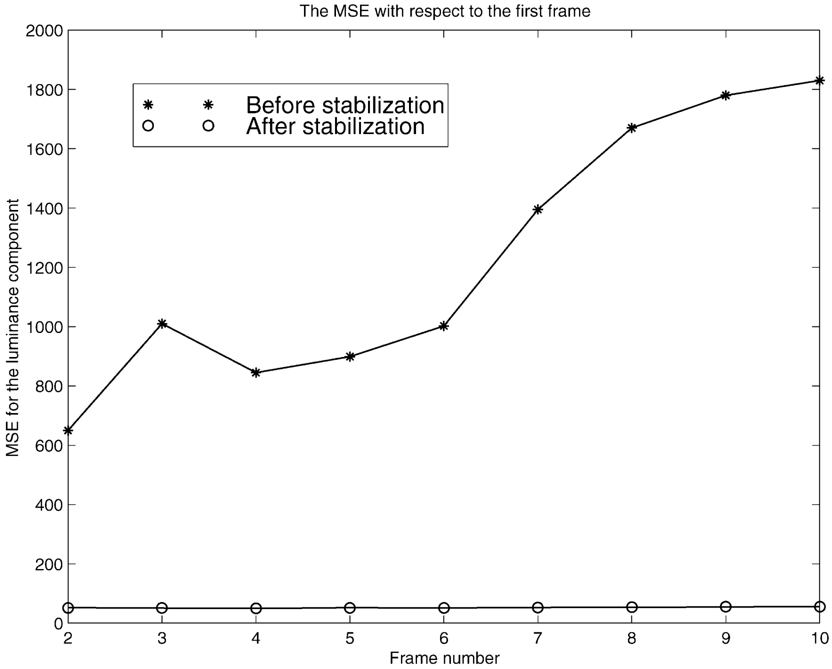


Fig. 6. The MSE between the luminance components of the first frame and the subsequent frames of the Bilkent sequence before (∗) and after (○) stabilization.

first frame changes between 2 and 16 pixels. The first and seventeenth frames of the Bilkent-2 sequence are given in Figs. 7(a) and (b), respectively. The seventeenth frame obtained after stabilization is given in Fig. 7(c). Note that in Fig. 7(c), the pixels on the right edge of the stabilized seventeenth frame are copied from the first frame as they are not visible in the unsteady seventeenth frame. The estimated contrast and brightness parameters of each frame with respect to the first frame of the Bilkent-2 sequence are presented in Fig. 8. Comparing Figs. 7(a) and (c), we conclude that the proposed stabilization algorithm is robust to varying illumination conditions.

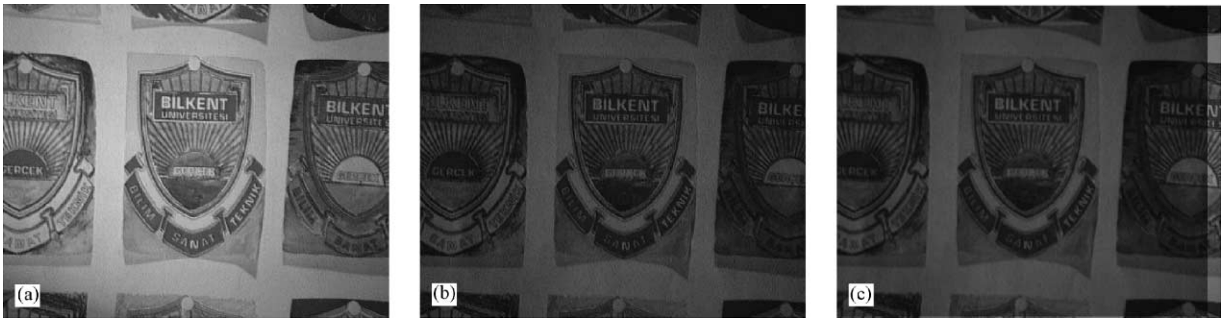
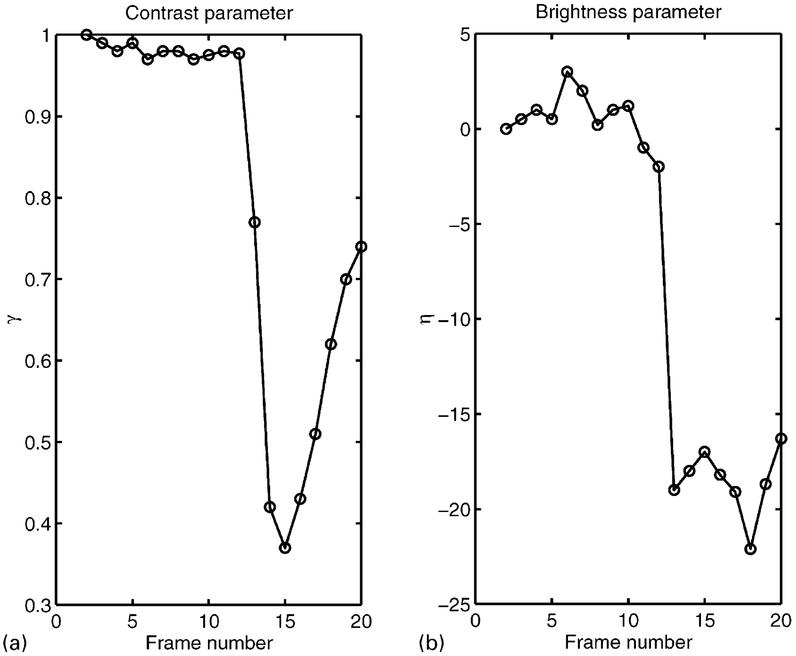


Fig. 7. (a) The first frame of the Bilkent-2 sequence, (b) the seventeenth frame of the Bilkent-2 sequence, (c) the seventeenth frame of the Bilkent-2 sequence after stabilization.


Contrast parameter

Brightness parameter

Fig. 8. (a) The estimated contrast (γ) parameters with respect to the first frame of Bilkent-2 sequence, (b) the estimated brightness (η) parameters with respect to the first frame of Bilkent-2 sequence.

7. The effect of image stabilization on the performance of the MPEG-2 video coding algorithm

At first, it may seem that a global translational motion among the frames in an unsteady video sequence would not affect the performance of the encoder noticeable because the motion vectors are differentially encoded in MPEG-2. However, in this and the next section, we show by means of experimental results that image stabilization prior to MPEG-2 coding of an unsteady video sequence does indeed increase the quality of the compressed video considerably. We also provide explanations as to why the MPEG-2 algorithm responds favorably to image stabilization.

7.1. Method of evaluation

In order to evaluate the response of the MPEG-2 video coding algorithm to image stabilization, three types of statistics are computed. These statistics are (i) the change in the allocation of the total bit budget by the encoder among the DCT data, the motion vectors and the header information in I, P and B frames; (ii) the distribution of macroblock types as intra-coded or non-intra-coded with various types of motion compensation schemes for P and B frames; and (iii) the PSNR for I, P and B frames. (For an overview of the MPEG-2 standard please refer to [6].) The header information in (i) refers to the information in the collection of all headers in the coded bitstream, i.e. the sequence, GOP, picture, slice and macroblock headers. (For the definition the header data in MPEG-2, please refer to [6] or [8].) In all our experiments, the change in the total bit count for the header information was almost completely determined by the macroblock header.

We carried out experiments using three video sequences that had unsteadiness as a result of misregistration of the frames during the scanning process. The test sequences had limited local motion and some frame-to-frame intensity variations as well. For image stabilization, we used the phase correlation [10] to estimate the pixel accurate part and the intensity-matching-based fast algorithm proposed in previous sections of this paper to estimate the subpixel accurate part of the motion. The estimated displacement vectors showing the amount of unsteadiness in the test sequences are given in Fig. 9. The absolute mean and

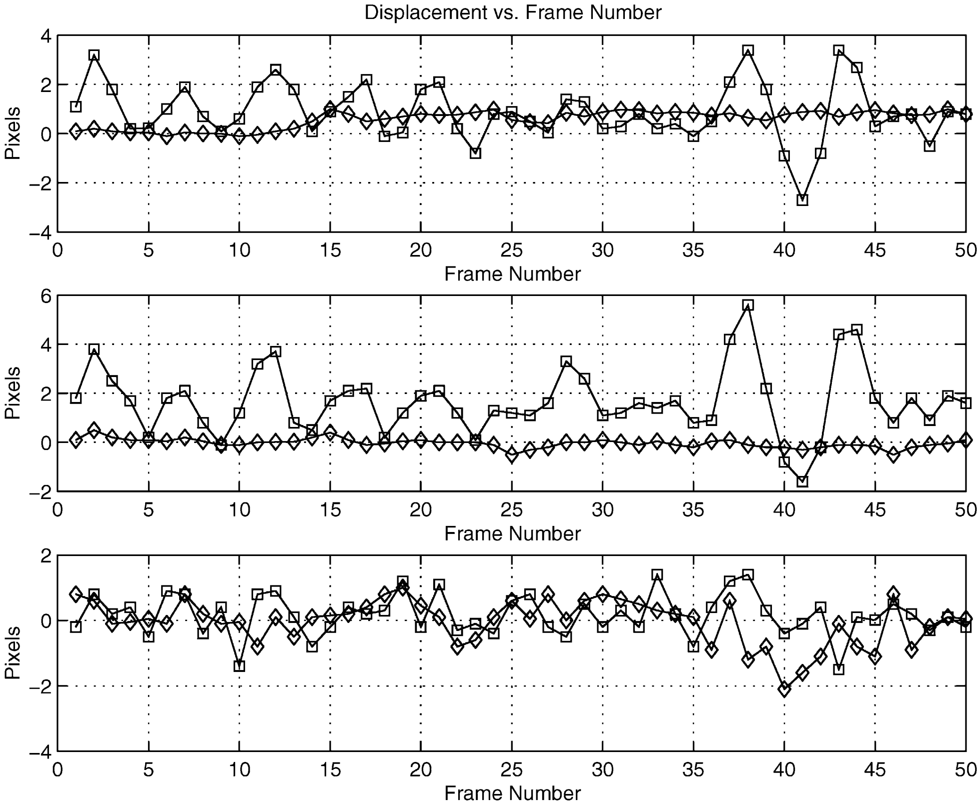


Fig. 9. The estimated horizontal (diamonds) and vertical (squares) jitter in the three film-originated image sequences used in the experiments.

Table 1
The absolute mean and variance of the horizontal and vertical components of the estimated jitter for the tree sequences

	Mean (pixels)	Variance (pixels)
Sequence 1	0.8586	0.5643
Sequence 2	0.9578	1.4364
Sequence 3	0.5396	0.1834

variations of the horizontal and vertical jitter are tabulated in Table 1. The estimation of the jitter is not affected by the amount of local motion in the scene since an appropriate block of the background is selected for estimation of the global unsteadiness.

We compressed both the original (unsteady) and stabilized (steady) image sequences at 4 Mb/s. The scan format used is 720×480 pixels per frame, 30 frame/s, 8-bit pixel depth and 4 : 2 : 0 chroma format (i.e., both chroma channels are horizontally and vertically subsampled). The encoder employed the TM-5 rate control [7]. We picked $N = 6$ and $M = 2$, i.e., each GOP had 6 frames: 1 I frame, 2 P frames and 3 B frames. Thus, in display order, the GOP definition was I B P B P B. This structure is chosen since it contains all three types of frames I, B and P. In fact, the experimental results do not change much for different compositions of the GOP. As the same bit-rate was used for both the original and the stabilized sequences, the total number of bits spent for a GOP was the same on the average for each sequence. What changed in the case of the stabilized sequence was the distribution of these bits among the I, P and B frames; and the allocation of the bits in each frame among the DCT data, the motion vectors and the header information.

7.2. Experimental results

We summarize the results in Figs. 10 and 11. In the following, we first describe how to read these plots, and then use them to explain the effect of stabilization on the performance of the MPEG-2 algorithm. The numbers in all the plots are sequence averages for each frame type.

Figs. 10(a)–(c) show the difference in the number of bits allocated for a particular type of information in the encoding process; a positive difference implies that more bits were spent by the encoder in the case of the stabilized sequence, and vice versa. In Figs. 10(a)–(c), two vertical axes are used to quantify the plotted data. On the left axis are the differences in the number of bits allocated for a particular type of frame in a GOP. Note that there are 2 P frames and 3 B frames in a GOP. The numbers on the right axes, on the other hand, give the difference in the number of bits on a per macroblock per frame basis. Hence, the scale of the right axis varies from plot to plot based on the number of times a particular frame type is repeated in a GOP. Fig. 11(b) shows the distribution of macroblock types in P frames as Intra (i.e., intra-coded), Zero (i.e., motion compensated with a zero motion vector), and non-zero (i.e., motion compensated with a non-zero motion vector). Fig. 11(c) shows the distribution of macroblock types in B frames as Intra, Fwd./Bwd. (i.e., forward or backward motion compensated) and Interp. (i.e., bidirectrionally motion compensated). Note that MPEG-2 distinguishes between the motion compensated macroblocks with zero vectors and those with non-zero vectors in P frames, whereas no such distinction is made for non-intra-coded macroblocks in B frames [8].

Fig. 11(a) gives the actual PSNR values for each frame type in the original and the stabilized sequences. The numbers in this plot show that *at least 1 dB improvement is achieved for all frame types* as a result of pre-processing the original sequence to remove the unsteadiness.

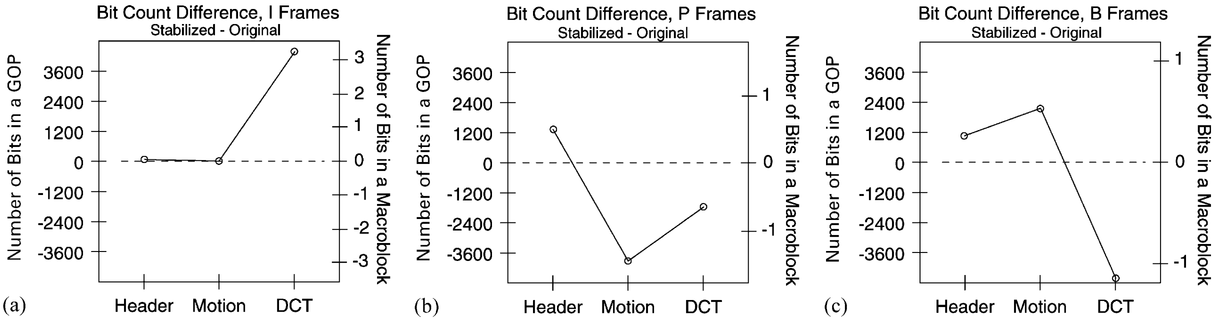


Fig. 10. The difference in the distribution of bits among the macroblock header, motion vectors, and DCT data for (a) I frames, (b) P frames and (c) B frames of the stabilized sequence versus the original sequence.

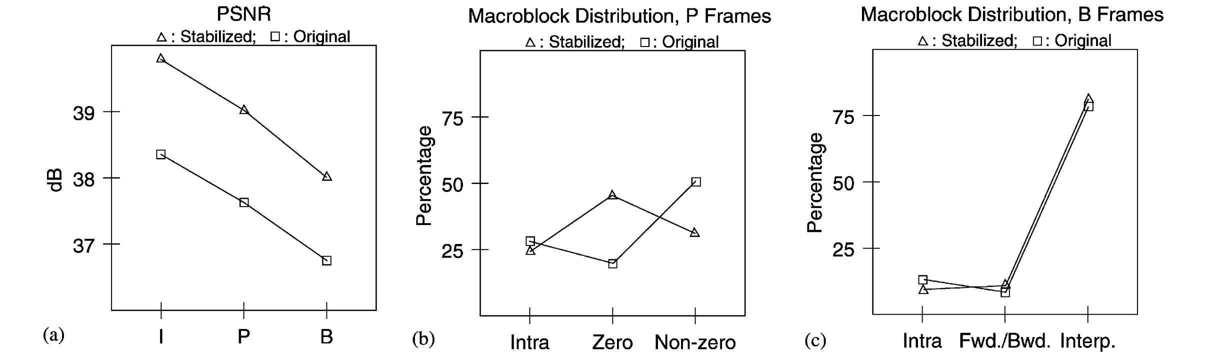


Fig. 11. (a) The comparison of PSNR values for the I, P and B frames in the stabilized and the original sequences; and the comparison of the distribution of macroblock types in (b) P frames and (c) B frames in the stabilized and the original sequences.

7.3. Analysis of the MPEG-2 coding results

In the following, we provide an analysis of the quality improvements in I, P and B frames brought by image stabilization in the light of the macroblock coding preferences employed in the MPEG-2 algorithm.

7.3.1. I frames

As seen in Fig. 10(a), only the number of bits for the DCT data is increased for the I frames. The header information does not increase or decrease noticeably for the stabilized sequence, and there is, of course, no motion data for I frames. Fig. 10(a) shows that the number of bits for the coded DCT data is increased by about 3.2 bits per macroblock for the stabilized sequence. This increase is the reason for the more than 1 dB improvement in the PSNR seen in Fig. 11(a).

7.3.2. P frames

Fig. 10(b) shows that the number of bits for both the DCT and the motion data are decreased for the P frames, while that for the header information is increased. The decrease for the motion data is about 1.5 bits per macroblock, while the increase for the header information is about 0.5 bits per macroblock. This behaviour of the encoder is explained by the results given in Fig. 11(b). Fig. 11(b) shows that in the stabilized image sequence, the number of non-intra macroblocks with zero motion vectors is increased by about 25% of the total number of macroblocks, and those with non-zero motion vectors is reduced by about the same amount. The number of intra-coded macroblocks is slightly decreased to make up for the difference.

In MPEG-2 (and MPEG-1), the fact that a particular macroblock has a zero motion vector is indicated in the macroblock header (the zero motion vector is not coded). This requires not more than 1 additional bit per macroblock [8]. On the other hand, if at least one component of the motion vector is non-zero, the macroblock is labeled as non-zero and its motion vector is coded. However, when the frame-to-frame displacement is global, the differential motion vector for such a macroblock will be zero. In MPEG-2 (and MPEG-1), coding a zero differential vector requires at least 2 bits per macroblock [8]. Thus, *in MPEG-2, an actual zero motion vector is favored over a zero differential motion vector*. Hence, the savings in the number of bits used for the combined motion and header information in P frames. This is the main reason why image stabilization improves the quality of the MPEG-2 compressed video.

We observe that although the number of bits used for coding the DCT data is decreased by more than 0.5 per macroblock, the PSNR is increased by at least 1 dB. This outcome is explained by the fact that the temporal prediction of the macroblocks in P frames is improved in the stabilized sequence because (i) the quality of the compressed I frames, which are used as anchor frames for prediction, is improved; and (ii) the proposed image stabilization algorithm can compensate for translational motion with better than 1/2 pixel accuracy (MPEG-2 has only 1/2 pixel accuracy), which results in smaller residual errors to be coded.

7.3.3. B frames

Fig. 10(c) shows that the number of bits for the DCT data is decreased by more than 1 bit per macroblock, while the number of bits for the motion and header information is only slightly increased. There is also only a slight change in the distribution of the macroblocks as intra-coded, forward or backward motion compensated, and interpolated (i.e., bidirectionally motion compensated). The fact that the PSNR is increased by more than 1 dB, while the number of bits for the DCT data is decreased, is again explained by two factors: the quality of the compressed I and P frames, which are used as anchor frames for predicting the B frames, is improved; and the image stabilization algorithm can compensate for translation motion with better than 1/2 pixel accuracy.

8. Conclusion

In this paper, a novel near-closed-form solution is proposed for the LMSE estimation of frame-to-frame subpixel displacements in an unsteady image sequence. It is shown that the proposed method provides significant improvements in terms of both accuracy and speed when compared to the existing search-based LMSE solutions. An extension of the method that is robust to frame-to-frame intensity variations is also developed by modeling such variations using contrast and brightness parameters. The performance of the proposed method is compared with the leading closed-form solutions for subpixel displacement estimation, namely the spatio-temporal differentiation method, the phase correlation surface interpolation method, and the cross correlation surface interpolation method, with and without the presence of intensity variations. It is shown that the near-closed-form solution outperforms the aforementioned methods in terms of the accuracy of displacement estimation. Finally, it is shown how the proposed near-closed-form solution can be utilized for estimating the affine motion when the frame-to-frame unsteadiness cannot be modeled by a pure displacement. The proposed method can be successfully used for jitter removal. However, some difficulties may arise when there is intentional camera motion in the sequence.

It is also demonstrated that the removal of the jitter prior to MPEG-2 video compression improves the quality of the compressed video significantly (by at least 1 dB for all frames). This result is explained by the fact that in MPEG-2, an actual zero motion vector is favored over a zero differential motion vector in P frames. Thus, in the case of image stabilization, the bits saved during the coding of the motion data in P frames are utilized during the coding of the DCT data in all frames resulting in reduced quantization error and hence increased PSNR. Furthermore, the accuracy of motion compensated prediction in MPEG-2 is limited to 1/2 pixels, while subpixel accurate image stabilization can compensate for translation motion with unlimited subpixel accuracy. The increased motion compensation accuracy of the proposed stabilization scheme helps reduce the error residuals left to be coded, causing an increase in the overall PSNR.

Appendix A. Basic summations

In the following, the summations are over $n_1, n_2 \in \mathscr{B}$ and the results of the summations are normalized by $N_1 N_2$. Thus, for example,

$$\sum s_1(n_1, n_2) \doteq \frac{1}{N_1 N_2} \sum_{n_1, n_2 \in \mathscr{B}} s_1(n_1, n_2).$$

The definition of the basic summations are now given as follows:

$$A_{0,0} = \sum s_1(n_1, n_2),$$

$$A_{0,0;\,0,0} = \sum s_1^2(n_1, n_2),$$

$$B_{0,0;\,i,j} = \sum s_1(n_1, n_2) s_2(n_1 + i, n_2 + j), \quad i,j = -1, 0, 1,$$

$$D_{i,j} = \sum s_2(n_1 + i, n_2 + j), \quad i,j = -1, 0, 1,$$

$$D_{i,j;\,k,l} = \sum s_2(n_1 + i, n_2 + j) s_2(n_1 + k, n_2 + l), \quad i,j,k,l = -1, 0, 1, (i,k), \quad (j,l) \neq (\pm 1, \mp 1).$$

We note that the total number of distinct basic summations is 49 (1 + 1 + 9 + 9 + 29).

Appendix B. MSE coefficients

In the following, we express $C_0^{(i)}, \ldots, C_8^{(i)}$, in terms of the basic summations. The scalars k and l are determined by the quadrant number as defined in Eq. (5).

$$C_0^{(i)} = A_{0,0;\,0,0} - 2B_{0,0;\,0,0} + D_{0,0;\,0,0},$$

$$C_1^{(i)} = 2k(B_{0,0;\,0,0} - B_{0,0;\,k,0} - D_{0,0;\,0,0} + D_{0,0;\,k,0}),$$

$$C_2^{(i)} = 2l(B_{0,0;\,0,0} - B_{0,0;\,0,l} - D_{0,0;\,0,0} + D_{0,0;\,0,l}),$$

$$\begin{aligned} C_3^{(i)} = 2kl(&-B_{0,0;\,0,0} + B_{0,0;\,k,0} + B_{0,0;\,0,l} - B_{0,0;\,k,l} + 2D_{0,0;\,0,0} - 2D_{0,0;\,k,0} \\ &- 2D_{0,0;\,0,l} + D_{0,0;\,k,l} + D_{k,0;\,0,l}), \end{aligned}$$

$$C_4^{(i)} = D_{0,0;\,0,0} - 2D_{0,0;\,k,0} + D_{k,0;\,k,0},$$

$$C_5^{(i)} = D_{0,0;\,0,0} - 2D_{0,0;\,0,l} + D_{0,l;\,0,l},$$

$$C_6^{(i)} = 2l(-D_{0,0;\,0,0} + 2D_{0,0;\,k,0} + D_{0,0;\,0,l} - D_{0,0;\,k,l} - D_{k,0;\,k,0} - D_{k,0;\,0,l} + D_{k,0;\,k,l}),$$

$$C_7^{(i)} = 2k(-D_{0,0;\,0,0} + D_{0,0;\,k,0} + 2D_{0,0;\,0,l} - D_{0,0;\,k,l} - D_{k,0;\,0,l} - D_{0,l;\,0,l} + D_{0,l;\,k,l}),$$

$$C_8^{(i)} = D_{0,0;\,0,0} - 2D_{0,0;\,k,0} - 2D_{0,0;\,0,l} + 2D_{0,0;\,k,l} + D_{k,0;\,k,0} + 2D_{k,0;\,0,l}$$
$$- 2D_{k,0;\,k,l} + D_{0,l;\,0,l} - 2D_{0,l;\,k,l} + D_{k,l;\,k,l}.$$

Appendix C. The coefficients of the fifth-order polynomial

In the following, we give the expressions for $E_0^{(i)}, \ldots, E_5^{(i)}$ in terms of $C_1^{(i)}, \ldots, C_8^{(i)}$. For notational simplicity, we omit the superscript $^{(i)}$ in the following equations as the expressions are the same for each $i = 1,2,3,4$.

$$E_0 = -4C_2C_4^2 + 2C_1C_3C_4 - C_1^2C_6,$$

$$E_1 = -8C_4^2C_5 + 4C_1C_4C_7 + 2C_3^2C_4 - 8C_2C_4C_6 - 2C_1^2C_8,$$

$$E_2 = 2C_1C_6C_7 - 4C_2C_6^2 + C_3^2C_6 - 8C_2C_4C_8 - 16C_4C_5C_6 - 2C_1C_3C_8 + 6C_3C_4C_7,$$

$$E_3 = -8C_2C_6C_8 + 4C_4C_7^2 - 16C_4C_5C_8 + 4C_3C_6C_7 - 8C_5C_6^2,$$

$$E_4 = 3C_6C_7^2 - 4C_2C_8^2 - 16C_5C_6C_8 + 2C_3C_7C_8,$$
$$E_5 = 2C_7^2C_8 - 8C_5C_8^2.$$

Appendix D. The coefficients for contrast and brightness parameters

Let

$$\Delta = \frac{1}{A_{0,0;\,0,0} - A_{0,0}^2}.$$

Then

$$G_0^{(i)} = \Delta[B_{0,0;\,0,0} - A_{0,0}D_{0,0}],$$

$$G_1^{(i)} = k\Delta[B_{0,0;\,k,0} - B_{0,0;\,0,0} - A_{0,0}(D_{k,0} - D_{0,0})],$$

$$G_2^{(i)} = l\Delta[B_{0,0;\,0,l} - B_{0,0;\,0,0} - A_{0,0}(D_{0,l} - D_{0,0})],$$

$$G_3^{(i)} = kl\Delta[B_{0,0;\,0,0} - B_{0,0;\,k,0} - B_{0,0;\,0,l} + B_{0,0;\,k,l} - A_{0,0}(D_{0,0} - D_{k,0} - D_{0,l} + D_{k,l})]$$

and

$$H_0^{(i)} = \Delta[A_{0,0;\,0,0}D_{0,0} - A_{0,0}B_{0,0;\,0,0}],$$

$$H_1^{(i)} = k\Delta[A_{0,0;\,0,0}(D_{k,0} - D_{0,0}) - A_{0,0}(B_{0,0;\,k,0} - B_{0,0;\,0,0})],$$

$$H_2^{(i)} = l\Delta[A_{0,0;\,0,0}(D_{0,l} - D_{0,0}) - A_{0,0}(B_{0,0;\,0,l} - B_{0,0;\,0,0})],$$

$$H_3^{(i)} = kl\Delta[A_{0,0;\,0,0}(D_{0,0} - D_{k,0} - D_{0,l} + D_{k,l}) - A_{0,0}(B_{0,0;\,0,0} - B_{0,0;\,k,0} - B_{0,0;\,0,l} + B_{0,0;\,k,l})].$$

In the above, the scalars k and ℓ are again determined by the quadrant number i as given in Eq. (5).

Appendix E. The coefficients for the brightness parameter

$$H_0^{(i)} = D_{0,0} - A_{0,0},$$

$$H_1^{(i)} = k(D_{k,0} - D_{0,0}),$$

$$H_2^{(i)} = l(D_{0,l} - D_{0,0}),$$

$$H_3^{(i)} = k\,l(D_{0,0} - D_{k,0} - D_{0,l} + D_{k,l}).$$

In the above, the scalars k and l are again determined by the quadrant number i as given in Section 2.

Appendix F. The coefficients for the contrast parameter

$$G_0^{(i)} = B_{0,0;\,0,0}/A_{0,0;\,0,0},$$

$$G_1^{(i)} = k(B_{0,0;\,k,0} - B_{0,0;\,0,0})/A_{0,0;\,0,0},$$

$$G_2^{(i)} = l(B_{0,0;\,0,l} - B_{0,0;\,0,0})/A_{0,0;\,0,0},$$

$$G_3^{(i)} = k\,l(B_{0,0;\,0,0} - B_{0,0;\,k,0} - B_{0,0;\,0,l} + B_{0,0;\,k,l}),/A_{0,0;\,0,0}.$$

In the above, the scalars k and l are again determined by the quadrant number i as given in Eq. (5).

References

[1] L.G. Brown, A survey of image registration techniques, ACM Comput. Surv. 24 (4) (December 1992) 325–376.

[2] S.D. Conte, C. De Boor, Elementary Numerical Analysis — An Algorithmic Approach, McGraw-Hill, New York, 1980.

[3] Ç. Eroğlu, Subpixel accuracy image registration with application to unsteadiness correction, Master's Thesis, Bilkent University, Ankara, Turkey, 1997.

[4] Ç. Eroğlu, A. Tanju Erdem, A fast algorithm for subpixel accuracy image stabilization for digital film and video, in: Proceedings of International Conference on Visual Communications Image Processing, San Jose, CA, 28–30 January 1998, SPIE Vol. 3309, pp. 786–797.

[5] C.S. Fuh, P. Maragos, Affine models for image matching and motion detection, in: Proceedings of IEEE International Conference on Acoustics, Speech, Signal Processing, 1991, pp. 2409–2412.

[6] B.G. Haskell, A. Puri, A.N. Netravali, Digital Video: An Introduction to MPEG-2, Chapman & Hall, London, 1997.

[7] ISO/IEC-JTC1/SC29/WG11 N0400: MPEG video test model 5 (TM5), April 1993.

[8] ISO/IEC 13818-2: Generic coding of moving pictures and associated audio — Part 2: Video, 1996.

[9] S.P. Kim, W. Su, Subpixel accuracy image registration by spectrum cancellation, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, 1993, Vol. V, pp. 153–156.

[10] C.D. Kuglin, D.C. Hines, The phase correlation image alignment method, in: Proceedings of IEEE International Conference on Cybernetics Society, 1975, pp. 163–165.

[11] H.S. Sawhney, S. Ayer, Compact representations of videos through dominant and multiple motion estimation, IEEE Trans. Pattern Anal. Mach. Intell. 18 (8) (August 1996) 814–830.

[12] R. Schultz, M.G. Alford, Multiframe integration via the projective transformation with automated block mathing feature point selection, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, 1999.

[13] A.M. Tekalp, Digital Video Processing, Prentice-Hall, New Jersey, 1995.

[14] P. Thevenaz, U.E. Ruttimann, M. Unser, A pyramid approach to subpixel registration based on intensity, IEEE Trans. Image Process. 7 (1) (January 1998) 27–41.

[15] Q. Tian, M.N. Huhns, Algorithms for subpixel registration, Comput. Visual Graphics Image Process. (CVGIP) 35 (1986) 220–233.

[16] G. Woolberg, Digital Image Warping, IEEE Computer Society Press, Los Alamitos, CA, 1990.

[17] Q. Zheng, R. Chelappa, A computational vision approach to image registration, IEEE Trans. Image Process. 2 (3) (July 1993) 311–326.

[18] Q. Zheng, R. Chelappa, Automatic feature point extraction and tracking in image sequences for arbitrary camera motion, Int. J. Comput. Vision 15 (1995) 31–76.