ISTANBUL UNIVERSITY –
JOURNAL OF ELECTRICAL & ELECTRONICS ENGINEERING

YEAR : 2004 (1233-1242)
VOLUME : 4
NUMBER : 2

IMAGE SEQUENCE STABILIZATION BASED ON ADAPTIVE POLYNOMIAL FILTERING USING THE LMS AND RLS ALGORİTHMS

Fatma ÖZBEK[1] Sarp ERTÜRK[2]

[1,2] University of Kocaeli, Engineering Faculty,
Electronics and Telecommunications Eng. Dept.,
41040, İzmit, Kocaeli

[1]E-mail: fozbek@kou.edu.tr [2]E-mail: sertur@kou.edu.tr

ABSTRACT

A novel image sequence stabilization system based on adaptive polynomial filtering of absolute frame displacements for the removal of undesired translational fluctuations in image sequences is proposed in this paper. Image sequence stabilization is accomplished by shifting image frames into smoothened positions obtained through the adaptive polynomial filter using the LMS and RLS algorithms. A stabilization system has successfully been implemented for real time applications, removing high frequency fluctuations in absolute frame positions while preserving desired global camera displacements.

Keywords: *Image Sequence Stabilization, Frame Position Smoothing, Polynomial Filtering, LMS Algorithm, RLS Algorithm*

1. INTRODUCTION

Image sequence stabilization (ISS) aims to remove irregular image fluctuations (jitter) being caused by undesired global camera motions in a video sequence, such as sequences acquired by a camera mounted on a moving platform, by a mobile phone with video capabilities, and robot-camera applications. ISS is the process of removing undesired translation, rotation or zoom jitter of global motion, which can be accomplished by smoothing absolute frame displacements so as to preserve requisite gross camera displacements only. While undesired fluctuations can be translation, rotation or zoom based, translational jitter is the most commonly encountered case [1]. Visual quality is most severely degraded by translational jitter, and therefore most stabilization systems have focused on translational jitter only, commonly referred to as two dimensional (2-D) stabilization systems [2]. While decomposing the jitter element of the assessed global motion, it is required to protect intentional camera motions such as pan during stabilization in order to prevent picture loss. Translational jitter correction can be carried out by displacing each frame by an appropriate amount, namely the

Received Date : 02.08.2003
Accepted Date: 15.06.2004

correction vector, so as to have the sequence display gross camera movements only.

Image stabilization systems typically consist of a motion estimation part and a motion correction part. The motion estimation part aims to find global motions of frames in an image sequence. The motion correction part carries out stabilization by eliminating undesired motion effects and preserving intentional global camera movements. In other words, the motion estimation part resolves global motions contained in an image sequence while the correction part compensates for the jitter component.

Global motion estimation techniques can be categorized into block-based matching approaches, phase correlation based on FFT techniques, and feature-based matching approaches. These techniques have in common that global motion vectors of an image sequence are obtained by registering every image frame with respect to the preceding frame. The frame displacement vector that will be used at the motion correction part of the proposed stabilization system is defined as the absolute frame displacement with respect to the first frame and can be obtained from the accumulation of all former interframe differential global motion vectors [3]. Motion vector integration (MVI), frame position smoothing (FPS) in the frequency domain, FPS based on FIR filtering and IIR filtering, FPS based on Kalman filtering [1], and FPS based on Polynomial Filtering [4] has been proposed for the stabilization of translational jitter. Although successful stabilization is achieved by optimally preserved gross displacements with FPS based on Polynomial Filtering, the system requires off-line processing not suited for real-time applications as the entire absolute displacement signal an image sequence is required. This paper presents two-dimensional frame position smoothing based image stabilization using adaptive polynomial filtering with least-mean square (LMS) and recursive least-squares (RLS) algorithm adaptation, providing real-time stabilization with appropriately preserved gross displacements.

2. Polynomial Filter

Polynomials are the simplest class of mathematical functions. A polynomial has one independent variable h, and it produces an output value by forming a weighted sum of the positive, integer-valued powers of h [5]. An m th-degree polynomial sequence is given by

$$f_i = \sum_{j=0}^{m} a_j h_i^j = a_0 + a_1 h_i + ... + a_m h_i^m \quad (1)$$

where the polynomial coefficients, a_j, $j \in [0, m]$, are unknown real constants [5].

For image stabilization by polynomial filtering, the frame position vector obtained by the global motion estimation process is applied to a low-degree polynomial filter and a smoothened frame position vector for horizontal and vertical directions is obtained. Referring to Eqn. (1), when the number of frames of the video sequence is given as n; for $i \in [0, n-1]$, $j \in [0, m]$, h_i shows the frame number of the image sequence, and f_i states the output of the m th-degree polynomial filter with polynomial coefficients a_j for the i th-frame. In other words, f_i shows the stabilized absolute frame position values for the horizontal and vertical direction. The observation matrix H is the same for horizontal and vertical directions, as its element h_{ij} display j -valued powers of the i th-frame number as shown as in Eqn. (2).

$$h_{ij} = h_i^j = i^j \quad , \quad i = 0,1,..,n-1 \quad , \quad j = 0,1,..,m \quad (2)$$

The observation matrix H in the polynomial filter is defined as [6]

$$H = \begin{bmatrix} 1 & h_0 & h_0^2 \cdots\cdots h_0^m \\ 1 & h_1 & h_1^2 \cdots\cdots h_1^m \\ \vdots & & \\ 1 & h_{n-1} & h_{n-1}^2 \cdots\cdots h_{n-1}^m \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \cdots\cdots 0 \\ 1 & 1 & 1 \cdots\cdots 1 \\ \vdots & & \\ 1 & (n-1) & (n-1)^2 \cdots (n-1)^m \end{bmatrix} \quad (3)$$

Eqn. (1) can then be rewritten in vector form as $F_x = HA$, and $F_y = HB$, representing stabilized frame position values in horizontal and vertical directions, respectively.

Polynomial filtering uses the least chi-square plane method to obtain the coefficients in Eqn. (1), that is finding the optimum solutions, A and B, which minimize the quantities [4] can be formulated as

$$\mathbf{h}_x^2 = \sum_{i=0}^{n-1}\left(\frac{x_i - f_{xi}}{\sigma_i}\right)^2 = \sum_{i=0}^{n-1}\left(\frac{x_i - \sum_{j=0}^{m} a_j h_i^j}{\sigma_i}\right)^2 = \left|H_0 A - X_0\right|^2$$

$$\mathbf{h}_y^2 = \sum_{i=0}^{n-1}\left(\frac{y_i - f_{yi}}{\sigma_i}\right)^2 = \sum_{i=0}^{n-1}\left(\frac{y_i - \sum_{j=0}^{m} b_j h_i^j}{\sigma_i}\right)^2 = \left|H_0 B - Y_0\right|^2 \quad (4)$$

where $h_{0ij} = \dfrac{h_i^j}{\sigma_i}$, $x_{0i} = \dfrac{x_i}{\sigma_i}$, $y_{0i} = \dfrac{y_i}{\sigma_i}$

$i = 0,1,....,n-1$; $j = 0,1,...,m$

In these equations, the standard deviation σ is usually taken equal to one. Horizontal and vertical frame positions, denoted as X and Y respectively, are constructed as

$$X = [x(0) \quad x(1) \quad ..... \quad x(n-1)]^T$$
$$Y = [y(0) \quad y(1) \quad ..... \quad y(n-1)]^T \quad (5)$$

The preferred way to minimize $\mathbf{h}^2$ is to find the least square solutions of the equations given in Eqn. (6).

$$H_o A = X_o \;\; ; \;\; H_o B = Y_o \quad (6)$$

Therefore, the polynomial coefficients A and B, defining the polynomial filter coefficients for stabilized position values in the horizontal and vertical directions are found by

$$A = \left(H^T H\right)^{-1} H^T X$$
$$B = \left(H^T H\right)^{-1} H^T Y \quad (7)$$

3. Adaptation of the Polynomial Filter by LMS Algorithm

The least-mean square (LMS) algorithm is probably the most widely used adaptive filtering algorithm due to its low computational cost, robustness, and good tracking performance (which is comparable to that of the recursive least-squares algorithm, RLS) [7].

Taking into account an image sequence with n frames, of which the horizontal movement is represented by the displacement vector $\mathbf{X}(i)$ and the vertical movement is represented by the displacement vector $\mathbf{Y}(i)$, can be stabilized based on adaptive LMS algorithm, using a length-N polynomial filter with an m th-degree polynomial. The LMS algorithm can be formulated as follows. The input signal vectors at time-i, represented by $\mathbf{X}(i)$ and $\mathbf{Y}(i)$ are denoted as

$$\mathbf{X}(i) = \left[x(i) \quad x(i-1) \;\; .... \;\; x(i-N+1)\right]^T$$
$$\mathbf{Y}(i) = \left[y(i) \quad y(i-1) \;\; .... \;\; y(i-N+1)\right]^T \quad (8)$$

In the following equations the variable $\mathbf{u}(i)$ is used to represents either of $\mathbf{X}(i)$ or $\mathbf{Y}(i)$.

The LMS Algorithm is briefly described by

$$y(i) = \mathbf{u}^T(i)\mathbf{W}(i-1)$$
$$e(i) = d(i) - y(i)$$
$$\mathbf{W}(i) = \mathbf{W}(i-1) + \mu\mathbf{u}(i)e(i) \quad (9)$$

where $\mathbf{W}(i)$ is the weight vector of the length-N adaptive filter at discrete time instance i, $\mathbf{u}(i)$ is a noisy, random input signal, in this case representing raw frame positions.

The purpose of this algorithm (as well as the RLS algorithm) is to adjust the weights so as to minimize the error which is defined as the difference between the desired signal, $d(i)$ and the output signal, $y(i)$. In the LMS algorithm, the step-size parameter μ is the factor that controls stability and the rate of convergence. The algorithm will only converge as long as μ is within the bounds of $0 \le \mu \le 2/\lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of the correlation matrix R. Larger values of the parameter μ lead to faster convergence; however, they cause increases in the residual error and may potentially render the algorithm unstable. In practice such problems are usually resolved with adaptive change of the adaptation step size [8], or with a conservative choice of a fixed step size [9]. For our applications, the initial value of μ was determined off-hand to ensure convergence

at a reasonable rate, then μ was adapted according to the reciprocal of $\lambda_{\max}$ at every frame. The other factor affecting performance of the filter is the initial value of the weight coefficient vector. $\mathbf{W}(0)$ is commonly preferred to be in the range of $0 \le \mathbf{W}(0) < 1$, in order to obtain a good performance, and in our case is selected off-hand by considering the length of the filter. The LMS algorithm is generally easy to implement as it does not require complex operations such as squaring, averaging or differentiation. Table 1 shows a step by step summary of the LMS algorithm.

Table 1. Summary of the LMS Algorithm.

Setting initial values	$\mu(0)$, $\mathbf{W}(0)$
Condition 1.	$\mu(0) = 0.0001$
Condition 2.	$\mathbf{W}(0)=0$ or $\mathbf{W}(0)=0.08$
Available at time i	$\mathbf{u}(i),\ \mathbf{W}(i-1),\ \mu(\mathrm{i})\ and\ d(i)$
Step 1.	$y(i) = \mathbf{u}^T(i)\mathbf{W}(i-1)$
Step 2.	$e(i) = d(i) - y(i)$
Step 3.	$\mathbf{W}(i) = \mathbf{W}(i-1) + \mu(\mathrm{i}-1)\mathbf{u}(i)e(i)$
Step 4.	$\mu(\mathrm{i}) = 1/\lambda_{\max}$

3.1. The Formation of the Desired Signal $d(i)$ of the Adaptive Filter

The desired signal $d(i)$ of the adaptive filter based on the LMS algorithm (and also the RLS algorithm) is obtained by the output of a small, length-N polynomial filter with the m-th degree. The desired signal $d(i)$ is the expectation of the product of the observation matrix and the polynomial coefficient matrix. The observation matrix H can be shown as

$$H = \begin{bmatrix} 1 & h_0 & h_0^2 \cdots\cdots h_0^m \\ 1 & h_1 & h_1^2 \cdots\cdots h_1^m \\ 1 & h_2 & h_2^2 \cdots\cdots h_2^m \\ \vdots & & \\ 1 & h_{N-1} & h_{N-1}^2 \cdots\cdots h_{N-1}^m \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots\cdots & 0 \\ 1 & 1 & 1 & \cdots\cdots & 1 \\ 1 & 2 & 4 & \cdots\cdots & 2^m \\ \vdots & & & & \\ 1 & (\mathrm{N}-1) & (\mathrm{N}-1)^2 & \cdots & (\mathrm{N}-1)^m \end{bmatrix} \quad (10)$$

In order to obtain the desired signal $d(i)$, tap-input vectors consisting of the horizontal and vertical position data of the i-th frame and previous N-1 frames are defined as

$$\underline{\mathbf{X}}(i) = \left[x(i-N+1)\ \ x(i-N+2)\ \ldots.\ x(i)\right]^T$$

$$\underline{\mathbf{Y}}(i) = \left[y(i-N+1)\ \ \mathrm{y}(i-N+2)\ \ldots.\ \mathrm{y}(i)\right]^T \quad (11)$$

For convenience, the variable $\underline{\mathbf{u}}(i)$ is used to represent either $\underline{\mathbf{X}}(i)$ or $\underline{\mathbf{Y}}(i)$ for the signal $d(i)$ depending on the processed direction. Consequently the desired signal $d(i)$ of the horizontal and vertical position, which minimizes the mean-squared error (MSE) between the original frame displacements (i.e, the raw displacements of frames) and the desired signal of the adaptive filter (i.e, the output of polynomial filter), is expressed as

$$d(i) = E\left[H(H^T H)^{-1} H^T \underline{\mathbf{u}}(i)\right] \quad (12)$$

4. Adaptation of RLS Algorithm to Polynomial Filter

The RLS algorithm can be formulated as follows. The input signal vectors at time-i are represented as $\mathbf{X}(i)$ and $\mathbf{Y}(i)$, and are constructed as shown in Eqn. (8). Again, the variable $\mathbf{u}(i)$ is used for either $\mathbf{X}(i)$ or $\mathbf{Y}(i)$,

while stabilizing with a length-N and m-th degree polynomial filter.

In the RLS algorithm, the update of the weight vector $\mathbf{W}(i)$ is described by

$$\mathbf{W}(i) = \mathbf{W}(i-1) + \mathbf{k}(i)e(i) \quad (13)$$

where the error signal $e(i)$ and the Kalman gain vector $\mathbf{k}(i)$ are given by

$$e(i) = d(i) - \mathbf{W}^T(i)\mathbf{u}(i) \quad (14)$$

$$\mathbf{k}(i) = \frac{\lambda^{-1}\mathbf{P}(i-1)\mathbf{u}(i)}{1+\lambda^{-1}\mathbf{u}^T(i)\mathbf{P}(i-1)\mathbf{u}(i)} \quad (15)$$

In Eqn. (15), $\mathbf{P}(i)$ is the inverse of the correlation matrix and is recursively given by

$$\mathbf{P}(i) = \left(\sum_{j=0}^{i}\lambda^{i-j}\mathbf{u}(j)\mathbf{u}^T(j)\right)^{-1} = \lambda^{-1}\left[\mathbf{P}(i-1) - \mathbf{k}(i)\mathbf{u}^T(i)\mathbf{P}(i-1)\right] \quad (16)$$

where λ is the forgetting factor in the range of $0 < \lambda \leq 1$. The purpose of the factor λ is to weight the most recent data more heavily and thus allow the filter coefficients to adapt to time-varying statistical characteristics of the position data [10]. The initial value of the inverse of the correlation matrix, i.e. $\mathbf{R}^{-1}(0)$, is generally taken as $\mu\mathbf{I}$, and corresponds to the initial value of the step size μ of the LMS algorithm [11]. The desired signal $d(i)$ of the adaptive filter based on the RLS algorithm is computed in a similar fashion to the LMS algorithm:

$$d(i) = E\left[H\left(H^T H\right)^{-1} H^T \underline{\mathbf{u}}(i)\right] \quad (17)$$

Table 2. Summary of the RLS Algorithm.

Setting initial values	λ, $\mathbf{W}(0)$, $\mathbf{P}(0)$,
Condition 1.	$\lambda = 0.99$
Condition 2.	$\mathbf{W}(0)=0$
Condition 3.	$\mathbf{P}(0)=\mu\mathbf{I}$, $\mu=0.01$
Available at time I	$\mathbf{u}(i),\ \mathbf{W}(i-1),\ \mathbf{P}(i-1)\ and\ d(i)$
Step 1.	$y(i) = \mathbf{u}^T(i)\mathbf{W}(i-1)$
Step 2.	$\mathbf{k}(i) = \frac{\lambda^{-1}\mathbf{P}(i-1)\mathbf{u}(i)}{1+\lambda^{-1}\mathbf{u}^T(i)\mathbf{P}(i-1)\mathbf{u}(i)}$
Step 3.	$\mathbf{P}(i) = \lambda^{-1}\left[\mathbf{P}(i-1) - \mathbf{k}(i)\mathbf{u}^T(i)\mathbf{P}(i-1)\right]$
Step 4.	$e(i) = d(i) - y(i)$
Step 5.	$\mathbf{W}(i) = \mathbf{W}(i-1) + \mathbf{k}(i)e(i)$

5. Image Stabilization by Adaptive Polynomial Filtering

A novel image sequence stabilization system based on adaptive polynomial filtering with the LMS and RLS algorithms for translational jitter correction is proposed in this paper. Global frame positions are smoothened with the adaptive polynomial filter, removing undesired translational jitter from the image sequence while preserving gross camera displacements. Provided that the approximation parameter μ is updated adaptively to enable tracking of changes in motion dynamics for the LMS algorithm, and the initial values of the weight coefficient vector are adjusted to be sufficiently small (typically in between 0 and 1), the adaptive polynomial filter with LMS and RLS algorithms is obtained to successively smoothen absolute frame positions. In this case, relatively high-frequency components caused by camera instability are removed, while low-frequency parts representing intentional camera displacements are retained. Stabilization of the video sequences is

accomplished as a result of bringing image frames into their correct (smoothened) positions by shifting each image frame by the correction vector. The correction vector for each frame is obtained by the difference of the original and filtered absolute frame positions, i.e. $\mathbf{V} = (\mathbf{X}, \mathbf{Y}) - (\mathbf{y}_x, \mathbf{y}_y)$.

6. Simulation Results

In this section, we demonstrate the performance of the adaptive polynomial filtering based on LMS and RLS algorithm for the image stabilization process. The stabilization intensity and gross displacement tracking performance of the adaptive polynomial filter are governed by the step size parameter and initial value of the weight coefficient vector, particularly for the LMS algorithm. Hence, initial conditions enabling successful stabilization results were determined experimentally taking a variety of test sequences into account. For the length N of the adaptive filter a value of 10 is selected to keep the computational cost low, and the polynomial degree m is selected to be 2 in order to avoid short-term variations in the stabilized frame positions [4].

The stabilization results with parameters μ=0.0001, and W(0)=0 for the LMS algorithm, and λ=0.99, W(0)=0, and μ=0.01 for the RLS algorithm are presented in Fig.1. This figure shows the original and adaptive polynomial filtered absolute frame positions for a sample sequence captured from a camera mounted on a moving motorcycle.

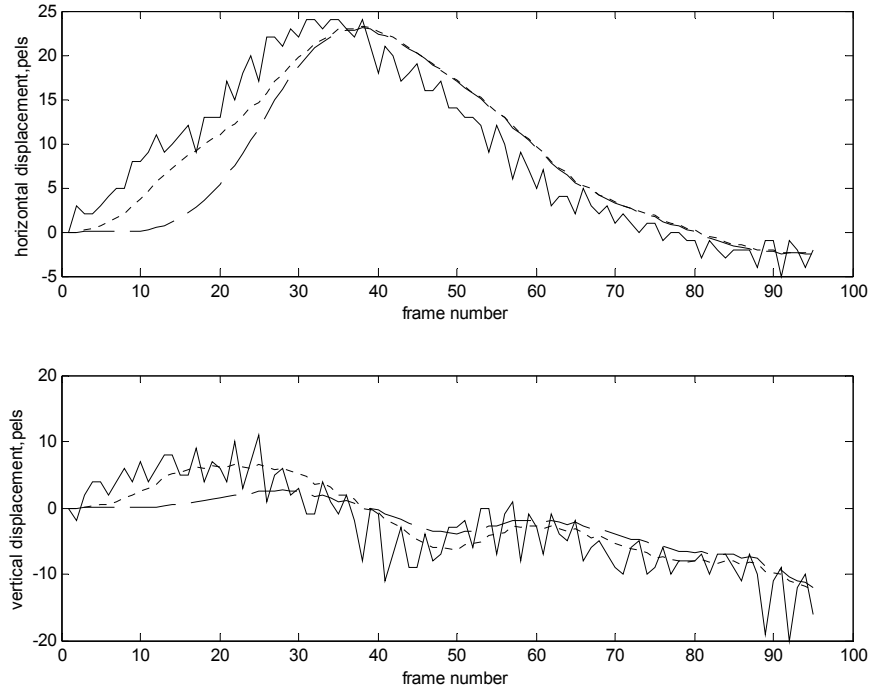

Fig. 1. Absolute frame positions: Original and filtered by adaptive polynomial filter with $W_{LMS}(0)=0$, $W_{RLS}(0)=0$.

_____ Original position
_ _ _ _ position filtered by LMS algorithm
..... position filtered by RLS algorithm

From Fig. 1, it is seen that the adaptive polynomial filter with LMS and RLS algorithms successfully smoothens absolute frame positions, removing the relatively high-frequency jitter caused by camera instability during the ride, while retaining gross displacements. It is also seen that the RLS algorithm catches the motion dynamics faster than the LMS algorithm and the RLS algorithm tracks gross camera displacements more closely, particularly at the first 35 frames of the sequence. Hence, to improve the filtering performance of the LMS algorithm, the initial values of the weight coefficient vector are set to 0.08, that has been found to enable more successful tracking. The initial values of the weight coefficient vector are actually determined taking the length of the filter into account, utilizing $0.8/N$. The stabilization results with parameters μ =0.0001, and W(0)=0.08 for the LMS algorithm, and λ =0.99, W(0)=0, and μ =0.01 for the RLS algorithm are presented in Fig.2.

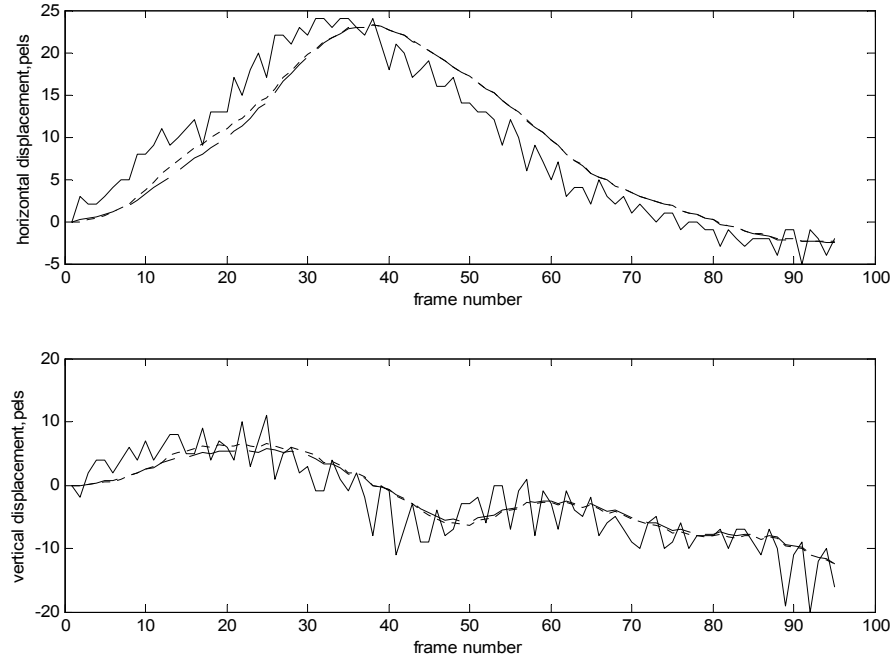

Fig. 2. Absolute frame positions: Original and filtered by adaptive polynomial filter with $W_{LMS}(0)=0.08$, $W_{RLS}(0)=0$.
_____ Original position
_ _ _ _ position filtered by LMS algorithm
..... position filtered by RLS algorithm

In the second case, the performance of the LMS algorithm is almost same as the RLS algorithm and the translational jitter is removed while preserving global frame displacements as shown in Fig. 2.

Stabilization Performance: Even in the second case, where $W_{LMS}(0)=0.08$, and $W_{RLS}(0)=0$, experimental results for various different image sequences show that the stabilization intensity and gross displacement preservation performance of RLS is superior to LMS. The main reason is that the RLS algorithm converges faster and in overall results in less error as shown in Fig.3.

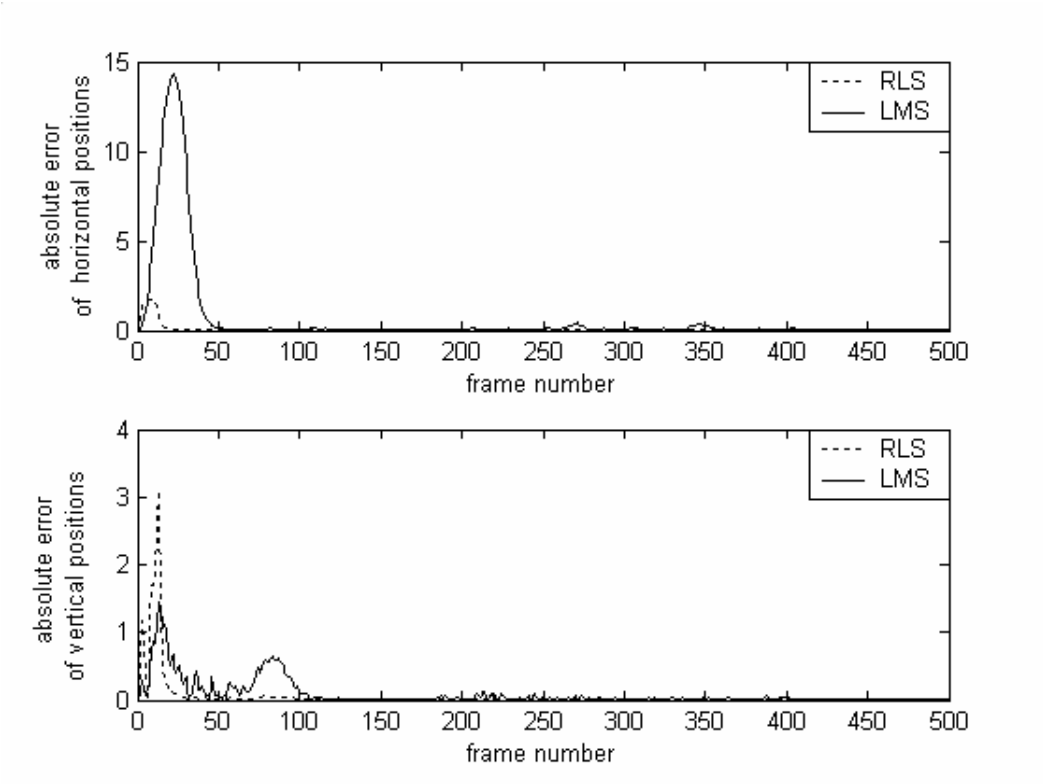

Fig. 3. Absolute error values computed for LMS and RLS algorithms, with $W_{LMS}(0)=0.08$, and $W_{RLS}(0)=0$, for a test image sequence.

Convergence Performance: A measure for the convergence of the algorithms is given in [12] and can be formulated as

$$\|V(k)\| = 10\log\frac{\sum_{i=1}^{k}\left(y(i)-d(i)\right)^2}{\sum_{i=1}^{k}\left(d(i)\right)^2} \quad , \quad k=1,.....,n \tag{18}$$

where n represents the number of frames of the image sequence. Fig. 4 displays a measure of the mean squared deviations of the adaptive polynomial filter outputs for a sample image sequence with 10000 frames. These results demonstrate that the RLS algorithm is clearly superior to the LMS adaptive polynomial filter in terms of the speed of convergence.

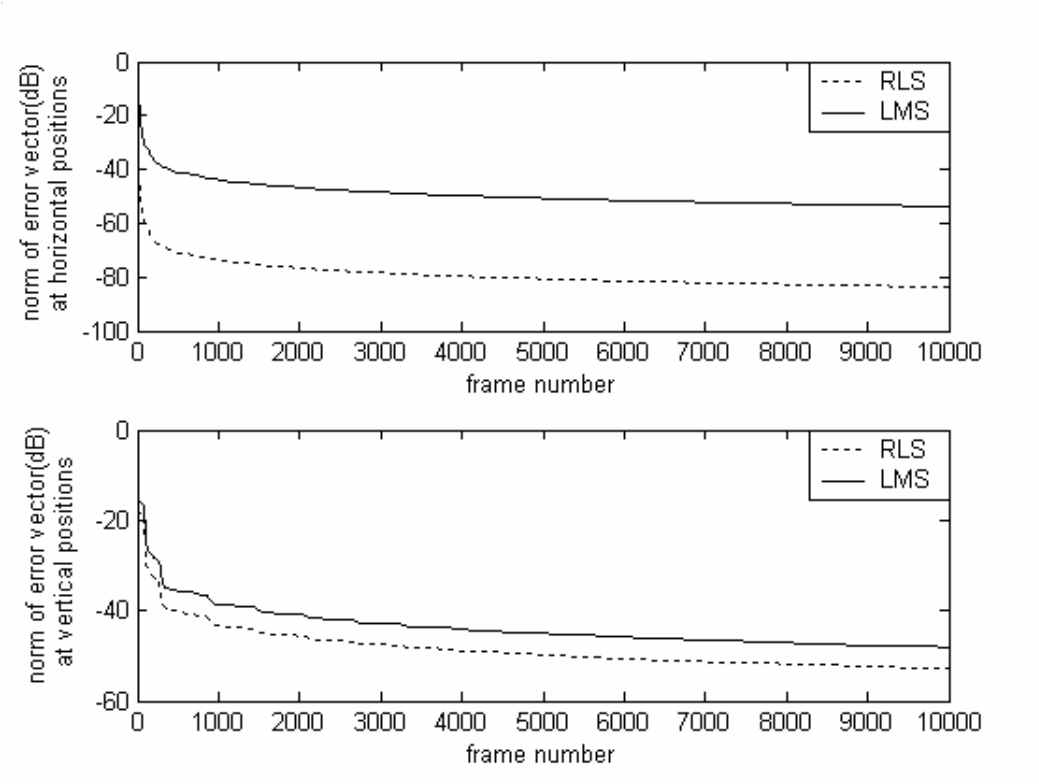

Fig. 4. Performance curve comparing the convergence speed of the RLS and LMS adaptive polynomial filters with $W_{LMS}(0)=0.08$, and $W_{RLS}(0)=0$.

7. Conclusions

An image sequence stabilization system with absolute frame displacement filtering, based on adaptive polynomial filtering with LMS and RLS algorithms has been presented in this paper. Image frames are shifted by correction vectors obtained from the difference of filtered versions of the absolute displacement signal using adaptive polynomial filters and original displacement values. Comparative results for the characteristics of two adaptive polynomial filters based on LMS and RLS algorithms respectively have been presented. Although the adaptive polynomial filter based on RLS algorithm has shown a better performance, it has a larger computational complexity. The proposed stabilizer combines real-time operation with reasonable stabilization and gross displacement preservation performance.

REFERENCES:

[1] S. Ertürk, "Image Sequence Stabilisation Based on Kalman Filtering of Frame Positions", Electronics Letters, 37, (20), 1217-1219, 2001.

[2] E. Yaman, S. Ertürk, "Image Stabilisation by Kalman Filtering Using a Constant Velocity Camera Model with Adaptive Process Noise", Proc. of ELECO'2001, November 2001, 152-155, Bursa, Turkey, 2001.

[3] S. Ertürk, T.J. Dennis, "Image Sequence Stabilisation Based on DFT Filtering", IEE Proceedings on Vision, Image, Signal Processing, 147, (2), 95-102, 2000.

[4] F. Özbek, S. Ertürk, "Polinom Filtresi ile Görüntü Stabilizasyonu", 10. Sinyal İşleme ve İletişim Uygulamaları Kurultayı, Pamukkale, 806-811, 2002.

[5] S. Valiviita, S. J. Ovaska, O. Vainio, "Polynomial Predictive Filtering in Control Instrumentation: A Review", IEEE Trans. on Ind. Electronics, Vol. 46, No: 5, 876-888, 1999.

[6] National Instruments™, LabVIEW™, User Manuel.

[7] V.H. Nascimento, "Improving the Initial Convergence of Adaptive Filters: Variable-Length LMS Algorithms", DSP 2002 14th IEEE Int. Conf. on DSP, 667-669, July 2002.

[8] M. Milisavljevic, "Delayed Error and Update Scheduling of LMS Algorithms", ICASSP 2002 Int. Con. On Acous. Speech and Sig. Proc., 1409-1412, May 2002.

[9] S. Haykin, "Adaptive Filter Theory", 3rd Edition, New Jersey, Prentice-Hall Inc., 1996.

[10] A.H. Abdullah, M.I. Yusof, S.R.M. Baki, "Adaptive Noise Cancellation: A Practical Study of the Least-Mean Square (LMS) Over Recursive Least-Square (RLS) Algorithm", IEEE Stud. Conf. on Res. And Dev. Proc., 448-452, March 2002.

[11] A.K. Chaturvedi, G. Sharma, "A New Family of Concurrent Algorithms for Adaptive Volterra and Linear Filters", IEEE Trans. on Signal Processing, Vol. 47, No: 9, 2547-2551, September 1999.

[12] V.J. Mathews, "Adaptive Polynomial Filters", IEEE Signal Processing Magazine, Vol. 8, No 3, pp. 10-26, July 1991.

ACKNOWLEDGEMENT

This work was supported by the Turkish Scientific and Technical Research Council, under grant EEEAG/101E006.