Robust 3D Camera Motion Parameter Estimation for Video Coding

Eung-Tae Kim,

Korea Polytechnic University, Siheung-City, Kyonggi-Do, Korea

Summary
In this paper, we present the estimation method of global motion parameters corresponding to 3D camera motion in the non-stationary noisy situation. Total least squares problem is first formulated to represent the global motion parameters estimation procedure from the noise-corrupted image coordinates. Then, a recursive total least squares (RTLS) algorithm is proposed to estimate 3D camera motion parameters in image sequences. The algorithm is proposed based on a five camera parameter model: zoom, focal length, pan, tilt, and swing. In the experimental results, the efficiency of the proposed RTLS algorithm is shown by comparing its MSE and PSNR with those of the conventional linear algorithms.

Key words:
Camera motion estimation, recursive total least squares algorithm, non-stationary noise, global motion compensation.

Introduction

Recently, the prominent trends in the video sequence coding are to achieve the various video services in Internet, multimedia communication, personal communications equipment [1]-[2]. Especially for the lower bitrate transmission, it is necessary to improve the motion estimation (ME) and motion compensation (MC) technique in video coding scheme.

To this aim, many researchers have studied the estimation of global motion parameters for video sequence coding [5]-[17]. They have shown that the global motion compensation can improve motion prediction and remove the motion side information greatly. In particular, in the model-based video coding systems [4], [5], it is essential to exactly model the camera system and to accurately extract all the parameters from image sequences.

Most of the existing camera motion estimations used for video coding applications can be divided into the nonlinear estimation [5], [15] and the linear estimation [6]-[11]. Among them, the linear estimation method is mostly used due to its simplicity and fast computation. This scheme is based on a simplification and approximation of a camera motion model under the assumption that the rotation angle is small enough and the focal length is sufficiently large. However, all of the existing linear methods suffer from the inevitable measurement errors such as spatial quantization errors, feature detector errors, and camera distortion. Furthermore, as the number of skip frames increases in low bitrate video coding, their performances decrease greatly since the simplified model is not appropriate for a large rotating images. To recover these problems, Kim and Kim [13] proposed mixed least squares - total least squares (MLS-TLS) method which can estimate the camera parameters such as the wide range of rotation and focal length.

The total least-squares (TLS) method to the linear regression problem has been shown to result in unbiased parameter estimates in which both the data matrix and the observation vector are corrupted by noise [20], [22]. In TLS problems, when large symmetric matrices, which usually have a dominant main diagonal and are sparse, are produced, one or more of their lowest eigenvalues and corresponding eigenvectors are required [28]. Specially, as the size of the matrices increases, TLS method needs an expensive computational load for computing the eigenvector and the large amount of memory. Therefore, it needs a fast algorithm for efficiently computing TLS solution.

Another serious problem in the global motion parameter estimation is the performance degradation due to the disturbance of independently moving objects. Several researchers assumed that local motion can be regarded as matching noise whose distribution is supposed to be Gaussian as a stationary noise [5], [13], [18]. However, if there exist local motions or the undesirable matching errors, these errors seem to be non-stationary noise. Therefore, unlike what the conventional methods assume, these errors introduce a bias to the estimated parameter [4], [17]. In that case, they do not guarantee a good estimation.

To overcome these problems, we propose a recursive total least squares (RTLS) algorithm which can accurately estimate the camera motion parameters in a non-stationary noise environment. In Section 2, we briefly present a linear motion parameter model which describes the camera motion such as zoom, focal length, and rotation. In Section 3, a recursive total least squares algorithm for estimating the camera motion parameters are described. In Section 4, the performance of the proposed RTLS algorithm is evaluated.

Manuscript received February 5, 2007
Manuscript revised February 25, 2007

2. Linear Motion Parameter Model

2.1 Camera Motion Model

Consider a situation where the features of an object are projected on the image plane and only these projections are known. Fig. 1 shows the geometry of camera motion in the 3D real world space.

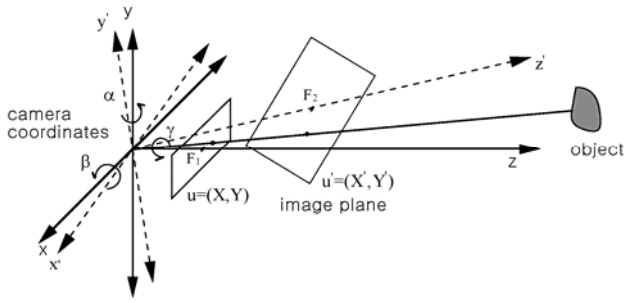


Fig. 1 Geometry of camera motion

The scene is stationary while the camera is zooming and rotating in the 3D space. In perspective imaging, the relationship between the image coordinate (*X, Y*) before the camera motion and the image coordinate (*X', Y'*) after the camera motion is described [12], [13] as

$$\begin{aligned} X' &= F_2\frac{r_{11}X + r_{12}Y + r_{13}F_1}{r_{31}X + r_{32}Y + r_{33}F_1} \\ Y' &= F_2\frac{r_{21}X + r_{22}Y + r_{23}F_1}{r_{31}X + r_{32}Y + r_{33}F_1}. \end{aligned} \quad (1)$$

where *F*1, *F*2 are the focal lengths of the camera before and after zoom *s* = *F*2=*F*1, respectively, and r_{ij} for *i, j* = 1, 2, 3 is an element of a 3D rotation matrix R which can be represented by successive rotations around the *y*-, *x*-, and *z*- axes of the camera coordinate system. The parameters of equation (1) are composed of five 3D camera motion parameters: zoom factor *s*, pan angle α, tilt angle β, swing angle γ, and focal length *F*1.

2.2 Linear Motion Parameter Model

From (1), the observed image coordinate (X', Y') after camera motion is given [12], [13] as

$$\vec{U} = \mathbf{H}\vec{a} \quad (2)$$

Where $\vec{U} = (X', Y')^T$, $\vec{a}$ consists of the eight *motion parameters* which are the function of the unknown camera motion parameters,

$$\begin{aligned} &a_1 = s\cdot r_{11}/r_{33}, \quad a_2 = s\cdot r_{12}/r_{33}, \quad a_3 = s\cdot F_1 r_{13}/r_{33}, \\ &a_4 = s\cdot r_{21}/r_{33}, \quad a_5 = s\cdot r_{22}/r_{33}, \quad a_6 = s\cdot F_1 r_{23}/r_{33}, \\ &a_7 = r_{31}/(F_1 r_{33}), \quad a_8 = r_{32}/(F_1 r_{33}), \end{aligned} \quad (3)$$

and $\mathbf{H}$ is a 2x8 matrix whose entities are functions of image coordinates (X,Y) and (X',Y') as

$$H = \begin{bmatrix} X & Y & 1 & 0 & 0 & 0 & -X'X & -X'Y \\ 0 & 0 & 0 & X & Y & 1 & -Y'X & -Y'Y \end{bmatrix}. \quad (4)$$

From (3), it is shown in [13] that the actual 3D camera motion parameters can be obtained as follows,

$$\begin{aligned} &r_{33} = \frac{a_1}{a_5 - a_6 a_8},\ F_1 = \sqrt{\frac{a_3}{r_{33}(a_4 a_8 - a_5 a_7)}},\ s = \sqrt{r_{33}^2(a_1^2 + a_2^2 + \frac{a_3^2}{F_1^2})} \\ &\alpha = \tan^{-1}(a_7 F_1),\ \beta = -\sin^{-1}(a_8 F_1 r_{33}),\ \gamma = \tan^{-1}(\frac{a_2}{a_5}). \end{aligned} \quad (5)$$

2.3 Motion Parameter Estimation Using Recursive Total Least Squares Algorithm

In general, the coordinates of features can not be measured exactly in all practical situations [5], [13], [18]. The sources of errors in the image coordinates include spatial quantization errors, feature detector errors, and camera distortion. These errors result in the errors of the estimates of the motion parameters. In a system that is well calibrated so that systematic errors are negligible, errors in the image coordinates of a feature can be modeled by random variables.

Let a random variable $\vec{\delta} = (\delta_x, \delta_y)^T$ denotes the errors in the image coordinates measured from the previous procedure. First, we assume that the noise in the image coordinates has zero mean and known variance σ^2. For example, the spatial quantization noise can be well modeled by a uniform distribution with the range corresponding to the width of the pixels. The variance of the feature detector error can also be estimated empirically. We assume that the noises at the different points are uncorrelated, and the noises in the two components of the same coordinates are uncorrelated.

Let the 2-D image plane vectors denote $\vec{U}_i = (X_i', Y_i')^T$ and $\vec{V}_i = (X_i, Y_i)^T$, $\forall k$, respectively. If the measured points, $\hat{X}_i, \hat{Y}_i, \hat{X}_i'$ and $\hat{Y}_i'$ have additive errors $\delta_{x_i}, \delta_{y_i}, \delta_{x'_i}$, and $\delta_{y'_i}$, respectively, for i=1,…,N, (2) can be expressed as

$$\vec{U}_i + \Delta\vec{U}_i = \hat{\boldsymbol{H}}_i\vec{a}, \quad i = 1, \cdots, N \quad (6)$$

where $\hat{\mathbf{H}}_i = \mathbf{H}_i + \Delta\mathbf{H}_i$ .Here, $\Delta\mathbf{H}_i$ denotes the noise matrix of $\mathbf{H}_i$ which is given by

$$\Delta H_i = \begin{bmatrix} \delta_{x_i} & \delta_{y_i} & 0 & 0 & 0 & 0 & -(\delta_{x'_i}X_i + \delta_{x_i}X'_i + \delta_{x'_i}\delta_{x_i}) & -(\delta_{y_i}X'_i + \delta_{x'_i}Y_i + \delta_{y_i}\delta_{x'_i}) \\ 0 & 0 & 0 & \delta_{x_i} & \delta_{y_i} & 0 & -(\delta_{y'_i}X_i + \delta_{x_i}Y'_i + \delta_{y'_i}\delta_{x_i}) & -(\delta_{y'_i}Y_i + \delta_{y_i}Y'_i + \delta_{y'_i}\delta_{y_i}) \end{bmatrix} \quad (7)$$

and $\Delta\hat{U}_i$ denotes the noise vector of $\hat{U}_i$ which is given by

$$\Delta\vec{U}_i = (\delta_{x'_i}, \delta_{y'_i})^T. \quad (8)$$

Given k point correspondences $(\hat{X}_i, \hat{Y}_i)$ and $(\hat{X}'_i, \hat{Y}'_i)$, $i = 1,2,\cdots,k$, the equation (6) can be expressed as

$$\vec{U}_k + \Delta\vec{U}_k = [\mathcal{H}_k + \Delta\mathcal{H}_k]\vec{a} \quad (7)$$

Where

$$\begin{aligned} \vec{U}_k &= [\vec{U}_1^T, \cdots, \vec{U}_k^T]^T \\ \mathcal{H}_k &= [\boldsymbol{H}_1^T, \cdots, \boldsymbol{H}_k^T]^T \\ \Delta\mathcal{H}_k &= [\Delta\boldsymbol{H}_1^T, \cdots, \Delta\boldsymbol{H}_k^T]^T \\ \Delta\vec{U}_k &= [\Delta\vec{U}_1^T, \cdots, \Delta\vec{U}_k^T]^T \end{aligned}$$

(7) is regarded as *Total Least Squares* (TLS) problem [20], [22] which is formulated as

$$\min_{\Delta\vec{U}^*_k, \Delta\mathcal{H}^*_k} \|[\Delta\vec{U}_k, \Delta\mathcal{H}_k]\|_F$$
$$subject\ to\ \vec{U}_k + \Delta\vec{U}_k = (\mathcal{H}_k + \Delta\mathcal{H}_k)\vec{a} \quad (8)$$

where $\|.\|_F$ denotes the Frobenius norm [19]. In [13], MLS-TLS method is proposed to solve the multiple linear regression problems. In our approach, a recursive algorithm is proposed for the efficient computation of the TLS solution. In the field of adaptive FIR system identification and coordinate relaxation, several researchers have studied the similar problems [26]-[29].

Define

$$\begin{aligned} \Phi_k &\triangleq [\vec{U}_k, \mathcal{H}_k] \\ W_k &\triangleq [\Delta\vec{U}_k, \Delta\mathcal{H}_k] \end{aligned}$$

and

$$\vec{q}_k \triangleq \begin{pmatrix} 1 \\ -\vec{a} \end{pmatrix} = [1 - a(1), \cdots, -a(8)]^T.$$

The TLS problem in (8) can be rewritten as

$$\min \|W_k\|$$
$$subject\ to\ [\Phi_k + W_k]\vec{q}_k = 0. \quad (9)$$

As in [26],[29], the solution of (9)} can be regarded as solving the following generalized eigenvalue problem.

$$R_k\vec{q}^*_k = \lambda_{min} D\vec{q}^*_k \quad (10)$$

where *D* is a given symmetric nonnegative definite matrix as shown in *Appendix A*, and R_k satisfies the Hermitian matrix which is defined as

$$R_k \triangleq \Phi_k^T\Phi_k = \sum_{j=1}^{k} r_j r_j^T \quad (11)$$

where r_j is a 9x2 matrix defined as

$$r_j \triangleq [\vec{U}_j, \boldsymbol{H}_j]^T = [\vec{r}_{1j}, \vec{r}_{2j}]. \quad (12)$$

It is known in [21], [30] that the above constrained minimization problem can beassociated with the equivalent minimization problem of Rayleigh quotient.

$$\mu(\vec{q}_k) = \min\|W_k\|^2 = \min_{\vec{q}_k} \frac{\vec{q}_k^T\Phi_k^T\Phi_k\vec{q}_k}{\vec{q}_k^T D\vec{q}_k} = \min_{\vec{q}_k} \frac{\vec{q}_k^T R_k\vec{q}_k}{\vec{q}_k^T D\vec{q}_k} \quad (13)$$

As a result, the minimization problem in (13) corresponds to the finding of the eigenvector $\vec{q}_k$ associated with the smallest eigenvalue of R_k .

Given the previous eigenvector $\vec{q}_{k-1}$, we update it to obtain $\vec{q}_k$ from

$$\vec{q}_k = \vec{q}_{k-1} + \Psi_k\vec{\alpha} \quad (14)$$

where Ψ_k is a 9 x 2 correction matrix and $\vec{\alpha} = [\alpha_1, \alpha_2]^T$. In the gradient method, $\vec{\psi}_k$ is chosen as the gradient of $\mu(\vec{q}_k)$, which gives a poor convergence speed. In the Newton's method, R_k is chosen as a Hessian matrix of (13). But it is difficult to compute a second derivatives of (13) and guarantee positive definite of Hessian matrix in practical situations.

Ψ_k is chosen to be the Kalman gain matrix,

$$\Psi_k = \boldsymbol{K}_k = R_k^{-1} r_k = [\vec{\psi}_{k1}, \vec{\psi}_{k2}] \quad (15)$$

In *Appendix B*, we divide the input matrix of (12) into the separate input vectors, that is,

$$\vec{r}_k = \vec{r}_{mk}, \quad m = 1, 2 \quad (16)$$

First we select $\vec{r}_{1k}$. Then, (14) can be reduced into

$$\vec{q}_k = \vec{q}_{k-1} + \alpha\vec{\psi}_k \quad (17)$$

where $\vec{\psi}_k$ is the Kalman gain vector,

$$\vec{\psi}_k = K_k = R_k^{-1}\vec{r}_k \quad (18)$$

In order to reduce the computational complexity, we use the matrix inversion lemma [31] to update the Kalman gain.

$$R_k^{-1} = [R_{k-1} + \vec{r}_k\vec{r}_k^T]^{-1} = R_{k-1}^{-1} - \frac{R_{k-1}^{-1}\vec{r}_k\vec{r}_k^T R_{k-1}^{-1}}{1 + \vec{r}_k^T R_{k-1}^{-1}\vec{r}_k}$$

So, R_k^{-1} is simply updated from the previous value. In our approach, it is easy to see that R_k satisfies positive semidefinite because it is a real-valued symmetric matrix. In the presence of noise, the *rank* (R_k) is generally full since the independent noises are added to the coordinates which are the elements of R_k and each coordinates before camera motion is measured at the distinct points.

So, we assume that R_k is positive definite, that is, *rank* (R_k) is full.

Instead of (14), substituting (17) to (13) and differentiating with respect to a scalar α, the value of α can be found by minimizing the quadratic equation

$$a\alpha^2 + b\alpha + c = 0 \quad (19)$$

where

$$\begin{aligned} a &= \vec{q}_{k-1}^T R_k\vec{\psi}_k\vec{\psi}_k^T D\vec{\psi}_k - \vec{\psi}_k^T R_k\vec{\psi}_k\vec{q}_{k-1}^T D\vec{\psi}_k \\ b &= \vec{q}_{k-1}^T R_k\vec{q}_{k-1}\vec{\psi}_k^T D\vec{\psi}_k - \vec{\psi}_k^T R_k\vec{\psi}_k\vec{q}_{k-1}^T D\vec{q}_{k-1} \\ c &= \vec{q}_{k-1}^T R_k\vec{q}_{k-1}\vec{q}_{k-1}^T D\vec{\psi}_k - \vec{q}_{k-1}^T R_k\vec{\psi}_k\vec{q}_{k-1}^T D\vec{q}_{k-1} \end{aligned}$$

It has shown in [28], [30] that α is always real, that is, $b^2 \geq 4ac$. Among the coefficients a, b, c , the quadratic forms are computed efficiently by using $\vec{q}_{k-1}^T R_k\vec{\psi}_k = \vec{q}_{k-1}^T\vec{r}_k$ and $\vec{\psi}_k^T R_k\vec{\psi}_k = \vec{\psi}_k^T\vec{r}_k$. Also, $\vec{q}_{k-1}^T R_k\vec{q}_{k-1}$ can be computed simply as follows.

$$\begin{aligned} \vec{q}_{k-1}^T R_k\vec{q}_{k-1} &= \vec{q}_{k-1}^T R_{k-1}\vec{q}_{k-1} + \vec{q}_{k-1}^T\vec{r}_k\vec{r}_k^T\vec{q}_{k-1} \\ &= \lambda_{min}(k-1)\vec{q}_{k-1}^T D\vec{q}_{k-1} + (\vec{q}_{k-1}^T\vec{r}_k)^2 \end{aligned}$$

The minimum value of $\mu(\vec{q}_k)$, $\lambda_{\min}(k)$ can be obtained [30], [26] by

$$\lambda_{min}(k) = \frac{\delta + a \cdot \alpha}{d} \quad (20)$$

where

$$d = \vec{\psi}_k^T D\vec{\psi}_k\vec{q}_{k-1}^T D\vec{q}_{k-1} - (\vec{q}_{k-1}^T D\vec{\psi}_k)^2$$
$$\delta = \vec{q}_{k-1}^T R_k\vec{q}_{k-1}\vec{\psi}_k^T D\vec{\psi}_k - \vec{q}_{k-1}^T R_k\vec{\psi}_k\vec{q}_{k-1}^T D\vec{\psi}_k.$$

As in [26], choosing the smallest root of (19), we obtain the update vector $\vec{q}_k$ for (17). Once more, $\vec{q}_k$ is updated by the above procedures for $\vec{r}_{2k}$ in (16). Finally, the solution of $\vec{a}$ can be obtained

$$a(i-1) = -\frac{q_k(i)}{q_k(1)}, \quad i = 2, \cdots, 9 \quad (21)$$

where $q_k(i)$ is the *i*th element of the vector $\vec{q}_k$. The proposed algorithm is summarized as follow.

Algorithm

[Step 1] Permutation : $\vec{a} \rightarrow \vec{q}$

[Step 2] Initialize $\vec{q}_0$ and $\vec{\psi}_0$

For *k=1,…,N*
For *m=1,2*

[Step 3] Select input vector $\vec{r}_{mk}$

[Step 4] Update $\vec{\psi}_k = R_k^{-1}\vec{r}_{mk}$ by using the matrix inversion lemma

[Step 5] Calculate α from (19)

$$\alpha = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

[Step 6] $\vec{q}_k = \vec{q}_k + \alpha\vec{\psi}_k$

End
End

[Step 7] Inverse permutation : $\vec{q} \rightarrow \vec{a}$

$$a(i-1) = -\frac{q_N(i)}{q_N(1)}, \quad i = 2, \cdots, 9$$

For good initial guess, least-squares method with *M(=8)* corresponding pairs is used to find the initial parameter

$$\vec{a}_0 = (\mathcal{H}^T\mathcal{H})^{-1}\mathcal{H}^T\vec{U} \quad (22)$$

where

$$\mathcal{H} = (H_1^T H_2^T \cdots H_M^T)^T$$

$$\vec{U} = (\vec{U}_1^T \vec{U}_2^T \cdots \vec{U}_M^T)^T$$

if $\det(\mathbf{H}^T\mathbf{H}) \neq 0$. The initial value of $\vec{q}_0$ is set to $q_0(1) = 1$ and $q_0(i+1) = -a_0(i), \text{for } i = 1, \cdots, 8$. This requires at least *M* input vectors until the rank of $\mathbf{H}^T\mathbf{H}$ is full.

3. Experimental Results

In this section, we show the experimental results on both the synthetic data and the real image sequences, respectively.

3.1 Experiments with synthetic data

The proposed RTLS algorithm is compared with the conventional linear least squares methods which are Y. T. Tse [7], A. Zakhor [6], and 6-parameter method [26, p.424], and with MLS-TLS algorithm [13]. The feature coordinates are randomly generated with a uniform distribution over the image plane coordinates. Random noise is added to each feature coordinate. Then, we have evaluated the average performance of the estimation method through 100 trials, where each has a new set of randomly generated feature points.

In this simulation, the image size of 480×704 is used, which is CCIR601 TV signal format. The focal length F_1 and F_2 are set to 100 and 95, respectively. The rotation angle (α, β, λ) is set to $(-0.1^o, 0.1^o, 0.0^o)$. Each feature points are contaminated with additive Gaussian noise with a mean of zero and a standard deviation of 0.5. To evaluate the robustness of the proposed algorithm, we generated 20% noisy motion fields, which represents the matching errors caused by local motion or undesirable observation, by corrupting the synthetic motion field with additive Gaussian noise which has mean, -2 and standard deviation, 5. This results in that in general, the matching noise caused by local motion may be biased. We estimated the global motion parameters for these motion fields with the proposed algorithm and the existing linear least squares algorithms.

To evaluate the performance of the parameter estimators, *mean square error* (MSE) is measured by

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \|\vec{u'}(\vec{u}_i, \vec{a}_e) - \vec{u'}(\vec{u}_i, \vec{a}_o)\|^2 \quad (23)$$

where $\vec{u}, \vec{u}'$ is a point before and after the camera motion, respectively, $\vec{a}_0$ denotes the true camera parameter vector, and $\vec{a}_e$ denotes the estimated one. The average mean squared errors of the estimation methods are shown in Fig. 2.

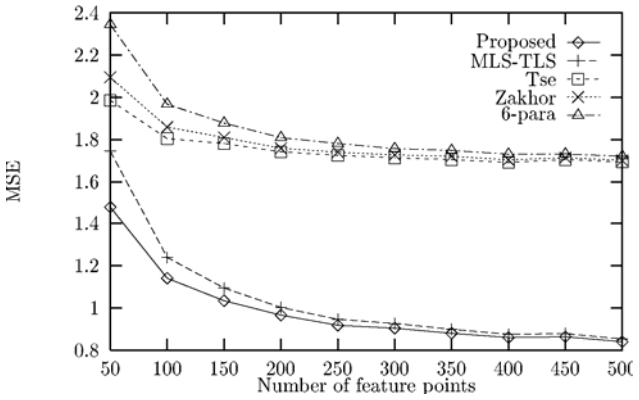


Fig. 2 *MSE* as the number of feature points varies

From this figure, the proposed RTLS algorithm produces smaller MSE than the existing least-squares algorithm when there exist the moving objects as well as feature detector errors.

Compared with MLS-TLS algorithm, the average performance of the proposed RTLS algorithm is shown in Fig. 3.

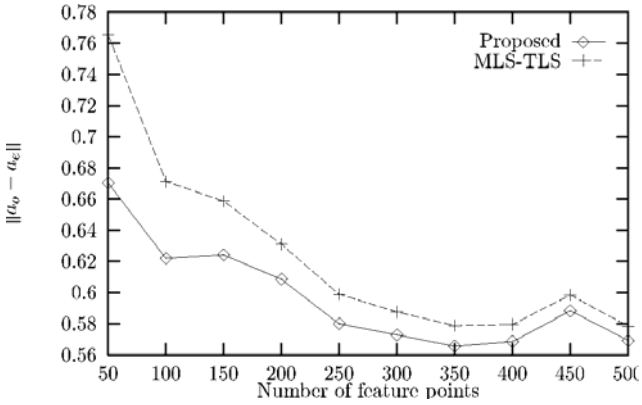


Fig. 3 Estimation accuracy

The proposed algorithm gives higher accuracy than MLS-TLS algorithm, and the proposed one is faster than MLS-TLS one as the number of feature points increases.

The computational complexity of the proposed algorithm is $m \cdot (5/2 \cdot n^2 + 6n)$ for $m = 2N, n = 9$. But the computational complexity of MLS-TLS algorithm is $2mn^2 - 2/3\,n^3 + 4mn_2^2 + 8n_2^3 + mn_1^2 + n_1^3/3$ for for $m = 2N, n = 9, n_1 = 2, n_2 = 7.$ Note that MLS-TLS algorithm consists of QR decomposition, TLS solution, and LS solution. Therefore, the computational complexity of MLS-TLS algorithm is larger than that of the proposed algorithm. In the aspects of memory cost, the proposed algorithm is more efficient than MLS-TLS algorithm.

3.2 Experiments with real image data

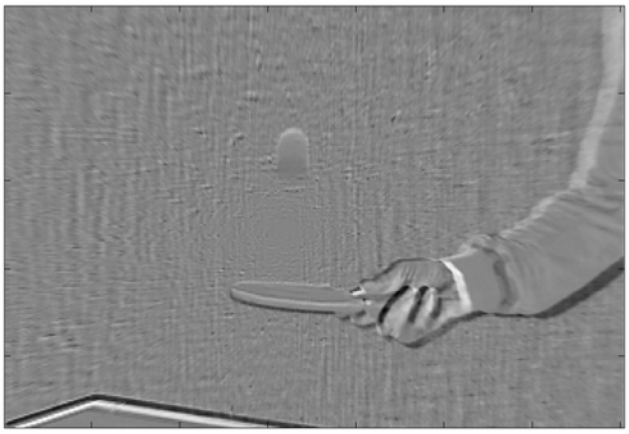
In this simulation, the proposed algorithm has been tested for real image data as shown in Fig. 4. The image size of 240 x 352 is used. The feature correspondence is established by using block matching algorithm whose block size and search range are set to (8, -15 ~ +15), respectively.

As shown in Fig. 5, the PSNR of the proposed algorithm is higher than that of the conventional algorithms.

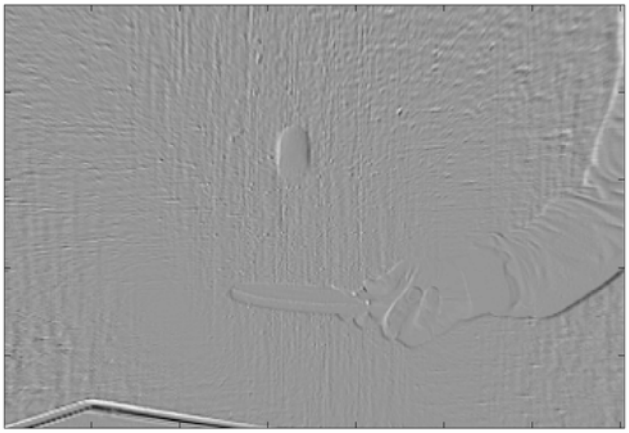
(a) Previous Image

(b) Previous Image

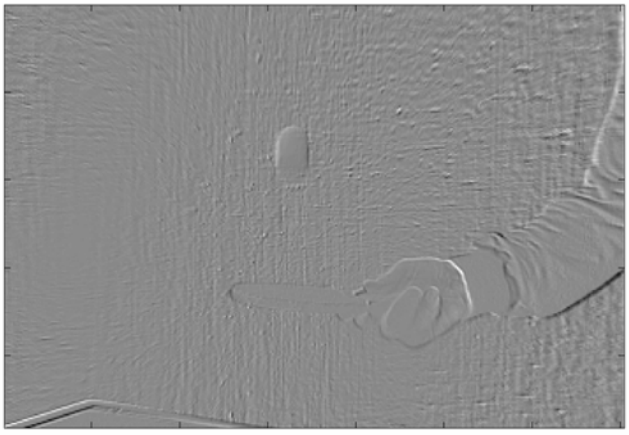
Fig. 4. *Table tennis* sequence with the camera motion parameters: $(\alpha, \beta, \gamma, F_1, s) = (-0.5^o, -0.25^o, 0.2^o, 100, 1.05)$



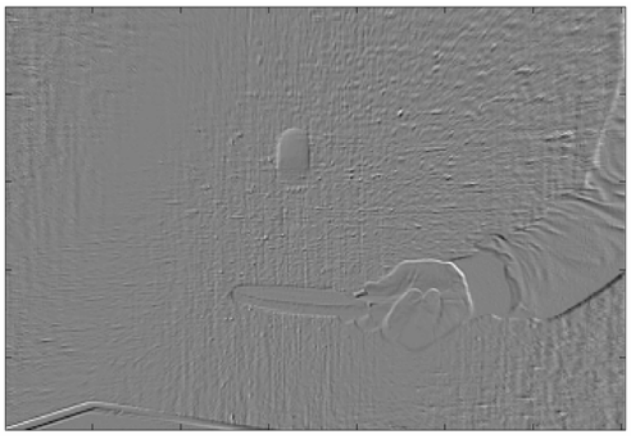
(a) Frame difference (PSNR = 20.5060dB)



(b) Tse's algorithm (PSNR = 30.4516dB)



(c) MLS-TLS algorithm (PSNR = 36.4962dB)



(d) Proposed algorithm (PSNR = 36.6730dB)
Fig. 5 Prediction error (x5)

3.3 Experiments in the video coding

In this simulation, the performance of the proposed algorithm is tested on the video sequences which contain global motion caused by camera motion and local motion caused by object motion. Therefore, a two-stage motion compensation (MC) technique is used in H.263 codec.

In the first stage, the proposed global MC (GMC) is used to compute camera motion parameters and to construct a globally motion compensated frame. In the second stage, a globally motion compensated frame is used as the reference frame in local MC (LMC). The succeeding LMC method predicts remaining object motion as well as "model failure" regions which are the luminance regions that cannot be successfully predicted, such as shadows or new objects appearing in the scene.

The coding structure used is IPPPP…, and the PB-frames mode is not employed. Three-dimensional VLC in H.263 is used to encode indexes of quantized DCT coefficients. Quantization step size of the DCT coefficients is set to 15. VLC for motion vector is the same as in H.263 except that half-pel motion estimation is not used.

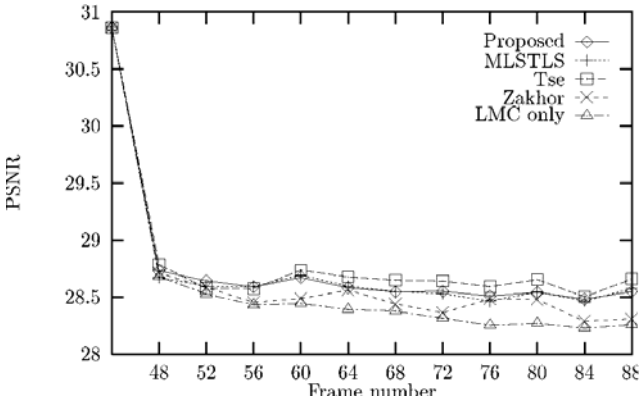
For GMC, block size and search range are set to (8, -15 ~ +15) and (16, -7 ~ +7), respectively. For LMC, block size and search range are set to 16, -7 ~ +7, respectively. In case of H.263, only LMC is used. The initial camera parameters can be obtained by the MLS-TLS algorithm [13] in first frame. For initial guess in first frame, all the rotational angles are set to zeros and zoom factor is set to 1.

The coding efficiency of the proposed MC method is compared with that of the MLS-TLS method and that of H.263. The test image sequences is 240 x352 SIF "Flower garden" sequence (44-90frames) whose frame rates are 15 fps.

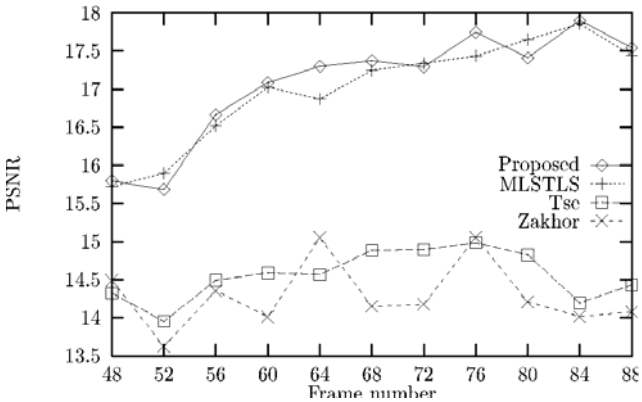
Fig. 6 show the peak-to-peak signal-to-noise ratio (PSNR) of decoded pictures, the PSNR of globally motion compensated (GMC) pictures, the total bitrates of coded pictures, and the bitrates of motion vector only, and the number of iteration in estimation procedure.

As shown in Fig. 6, the overall performance of the proposed method becomes considerably better than that of the conventional method [6,7] as well as that of H.263 (LMC only) even in a large 3D rotating image. The total PSNR and bitrates of the proposed method are slightly better than those of the MLS-TLS method.

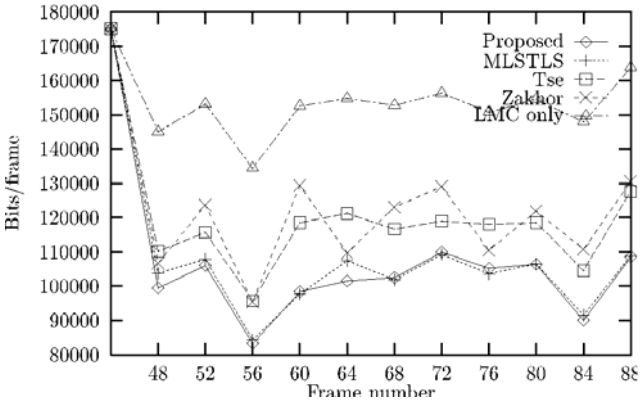
In the view of GMC, the proposed GMC method has better performance in Fig. 6 (b). In the aspect of computational time, the proposed method significantly outperforms the MLS-TLS method as shown in Fig. 6 (d). The average number of iteration of the proposed method is 1/3 of the MLS-TLS method. This implies that the proposed algorithm can accurately estimate the camera motion parameters by means of fewer iterations. Thus, the computational effort can be reduced greatly in the estimation procedure.



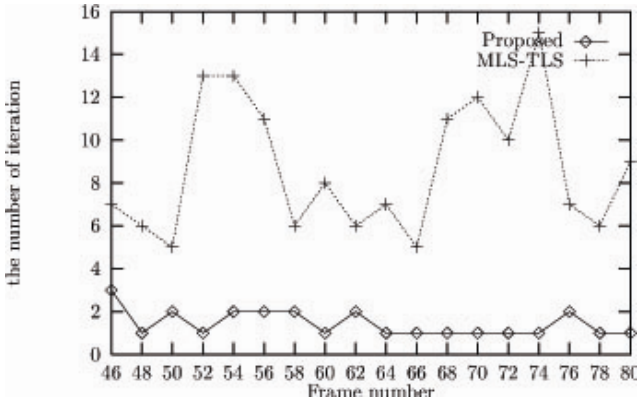
(a) PSNR of decoded pictures



(b) PSNR of GMC pictures



(c) total bitrates



(d) the number of iteration

Fig. 6 The performance of the proposed method in the flower garden sequence

4. Conclusions

In this paper, we have described a recursive total least squares algorithm for estimating 3D camera motion in image sequences. The proposed RTLS algorithm is based on a five camera parameter model: zoom, focal length, pan, tilt, and swing. Specially, we have shown that the recursive total least squares algorithm for MLS-TLS problem can be applied to estimate the motion parameters. RTLS algorithm has an advantage in efficiently computing an eigenvector associated with TLS solution. The parameter estimation using the RTLS algorithm reduces the effect of the non-stationary noises greatly. It has been shown in the simulation that the proposed algorithm outperforms MLS-TLS algorithm as well as the existing linear least squares algorithms in the presence of measurement errors.

The proposed method can improve the motion estimation accuracy in the video coding. The simulation result shows that the overall performance of the proposed method is considerably better than that of the conventional method.

Appendix A

We explain how to get *D*. In (7), the matrix $\hat{\mathbf{H}} = [\mathbf{H}_1, \hat{\mathbf{H}}_2]$ can be divided into the two noise-free columns $\mathbf{H}_1$ and the other noise-corrupted columns $\hat{\mathbf{H}}_2$. It is easy to see that 3rd, 6th columns of $\Delta\mathbf{H}$ are all zeros. In some applications, the similar problems are solved by modified TLS method when some of the columns of the data matrix may be known exactly [23]-[24]. In mixed LS-TLS problem, the closed-form solution can be interpreted as

$$\vec{a}^* = (\hat{\mathcal{H}}^T \hat{\mathcal{H}} - \sigma^* \Gamma)^{-1} \hat{\mathcal{H}}^T \hat{\vec{U}} \tag{24}$$

as in [25], where

$$\hat{\mathcal{H}}\vec{a} = \mathcal{H}_1 \vec{a}_1 + \hat{\mathcal{H}}_2 \vec{a}_2 = \hat{\vec{U}},$$

σ^* is a smallest singular value of $[R_{22}, R_{2b}]$ [13], and

$$\Gamma = \begin{bmatrix} \mathbf{0}_{2\times 2} & \\ & \boldsymbol{I}_{6\times 6} \end{bmatrix}. \tag{25}$$

Remind that the diagonal terms of Γ, correspond to the exactly known columns of $\hat{\mathbf{H}}$, are zeros. In (11), we get 9 x 9 matrix, $R = \Phi^T \Phi$. $\Phi = [\vec{U}, \mathbf{H}]$ has the exactly known columns, 4th and 7th columns. Using the relationship between (10) and the above properties, it is easy to see that

$$D = \begin{bmatrix} \boldsymbol{I}_{3\times 3} & & & & \\ & 0_{1\times 1} & & & \\ & & \boldsymbol{I}_{2\times 2} & & \\ & & & 0_{1\times 1} & \\ & & & & \boldsymbol{I}_{2\times 2} \end{bmatrix} \tag{26}$$

and *D* is a 9 x 9 symmetric nonnegative matrix.

Appendix B

We find $\vec{\alpha}$ by substituting (14) to (13) and differentiating with respect to α_1, α_2, respectively. From (14), $\Psi_k \vec{\alpha} = \alpha_1 \vec{\psi}_{k1} + \alpha_2 \vec{\psi}_{k2}$. First, let

$\vec{\phi}_{k2} = \vec{q}_{k-1} + \alpha_2 \vec{\psi}_{k2}$ and, for fixed α_2, minimizing (13) requires setting its derivative with respect to α_1 equal to zero. We get

$$a_1\alpha_1^2 + b_1\alpha_1 + c_1 = 0 \quad (27)$$

where

$$\begin{aligned} a_1 &= \vec{\phi}_{k2}^T R_k \vec{\psi}_k \vec{\psi}_k^T D\vec{\psi}_k - \vec{\psi}_k^T R_k \vec{\psi}_k \vec{\phi}_{k2}^T D\vec{\psi}_k \\ b_1 &= \vec{\phi}_{k2}^T R_k \vec{\phi}_{k2} \vec{\psi}_k^T D\vec{\psi}_k - \vec{\psi}_k^T R_k \vec{\psi}_k \vec{\phi}_{k2}^T D\vec{\phi}_{k2} \\ c_1 &= \vec{\phi}_{k2}^T R_k \vec{\phi}_{k2} \vec{\phi}_{k2}^T D\vec{\psi}_k - \vec{\phi}_{k2}^T R_k \vec{\psi}_k \vec{\phi}_{k2}^T D\vec{\phi}_{k2} \end{aligned}$$

Then, for fixed α_1 and $\vec{\phi}_{k1} = \vec{q}_{k-1} + \alpha_1 \vec{\psi}_{k1}$, we get the following equation with respect to α_2

$$a_2\alpha_2^2 + b_2\alpha_2 + c_2 = 0 \quad (28)$$

where

$$\begin{aligned} a_2 &= \vec{\phi}_{k1}^T R_k \vec{\psi}_k \vec{\psi}_k^T D\vec{\psi}_k - \vec{\psi}_k^T R_k \vec{\psi}_k \vec{\phi}_{k1}^T D\vec{\psi}_k \\ b_2 &= \vec{\phi}_{k1}^T R_k \vec{\phi}_{k1} \vec{\psi}_k^T D\vec{\psi}_k - \vec{\psi}_k^T R_k \vec{\psi}_k \vec{\phi}_{k1}^T D\vec{\phi}_{k1} \\ c_2 &= \vec{\phi}_{k1}^T R_k \vec{\phi}_{k1} \vec{\phi}_{k1}^T D\vec{\psi}_k - \vec{\phi}_{k1}^T R_k \vec{\psi}_k \vec{\phi}_{k1}^T D\vec{\phi}_{k1} \end{aligned}$$

As in [26], we choose the smallest root of (27) and (28) as

$$\begin{aligned} \alpha 1 &= \frac{-b_1 - \sqrt{b_1^2 - 4a_1c_1}}{2a_1} \\ \alpha 2 &= \frac{-b_2 - \sqrt{b_2^2 - 4a_2c_2}}{2a_2} \end{aligned} \quad (29)$$

From upper equations, it is difficult to solve α_1 and α_2 simultaneously. So, we can solve them alternately after one value is fixed. But it needs an expensive computational load. Therefore, we first solve α_1 for fixed $\alpha_2 = 0$, and then solve α_2 finally from the estimated α_1. This procedure means that if \$\alpha_2=0\$, $\vec{r}_{2k}$, the second column of the input matrix in (12), is removed. In other words, it separates input matrix into each column vector and update the parameter $\vec{q}_k$ successively. As a result, the column vectors, consisting of the input matrix, are considered as each input vector, separately.

Acknowledgments

This work is financially supported by SMBA(small and medium business administration) through the project of the UniBiz (University for Inno-BIZ).

References

[1] H. Li, A. Lundmark, and R. Forchheimer, "Image sequence coding at very low bitrates: a review," *IEEE Trans. on Image Processing***.** vol. 3, no. 5, pp. 589-608, September 1994.

[2] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. of the IEEE*. vol. 83, no. 6, pp. 858-875, June 1995.

[3] Draft ITU-T Recommendation H.263, "Video coding for low bitrate communication," December 1995.

[4] M. I. Sezan and R. L. Lagendijk, *Motion analysis and image sequence processing*, Kluwer Academic Pub., 1993.

[5] J. Park, N. Yagi, K. Enami, K. Aizawa, and M. Hatori, "Estimation of camera parameters from image sequence for model-based video coding," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 4, no. 3, pp. 288-296, June 1994.

[6] A. Zakhor and F. Lari, "Edge-based 3-d camera motion estimation with application to video coding," *IEEE Trans. on Image Processing*, vol. 2, no. 4, pp. 481-498, October 1993.

[7] Y. T. Tse and R. L. Baker, "Global zoom/pan estimation and compensation for video compression," *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 2725-2728, 1991.

[8] D. Adolph and R. Buschmann, "1.15 Mbit/s coding of video signals including global motion compensation," *Signal Processing: Image Communication*, vol. 3, no. 4, pp. 259-274, 1991.

[9] M. Hotter, "Differential estimation of the global motion parameters: Zoom and pan," *Signal Processing*, vol. 16, no. 3, pp. 249-265, March 1989.

[10] S. F. Wu and J. Kittler, "A differential method for simultaneous estimation of rotation, change of scale and translation," *Signal Processing: Image Communication*, vol. 2, no. 1, pp. 69-80, May 1990.

[11] G. Keesman, "Motion estimation based on a motion model incorporating translation, rotation and zoom," *Signal Processing IV: Theories and Applications*, pp. 31-34, 1988.

[12] E. T. Kim, J. K. Han, and H. M. Kim, "A Kalman-filtering method for 3D camera motion estimation from image sequences," *Int. Conf. on Image Processing*, vol. 3, pp. 641-644, October 1997.

[13] E. T. Kim and H. M. Kim, "Efficient linear 3D camera motion estimation method with applications to video coding," *Optical Engineering*, vol. 37, no. 3, pp. 1065-1077, March 1998.

[14] E. T. Kim, Y. J. Choi, and H. M. Kim, "A new 3-D camera motion estimation using displacement field smoothing for video coding," *Fourth Int. Symposium on Commun. Theory and Applications*, pp. 309-314, July 1997.

[15] C. A. Papadopoulos and T. G. Clarkson, "Motion compensation using second-order geometric

transformations," *IEEE Trans. on Circuits and Systems for Video Technology*, vol.5, no.4, pp. 319-331, August 1995.
[16] Y. Nakaya and H. Harashima, "Motion compensation based on spatial transformations," *IEEE Trans. on Circuits and Systems for Video Technology*, vol.4, no.3, pp. 339-356, June 1994.
[17] D. Wang and L. Wang, "Global motion parameter estimation using a fast and robust algorithm," *IEEE Trans. on Circuits and Systems for Video Technology*, vol.7, no.5, pp. 823-826, October 1997.
[18] T. S. Huang and A. N. Netravali, "Motion and structure from feature correspondences: A review," *Proc. of the IEEE*, vol. 82, no. 2, pp. 252-268, February 1994.
[19] G. H. Golub and C. F. Van Loan, *Matrix computations*, The Johns Hopkins university press, 1989.
[20] G. H. Golub and C. F. Van Loan, "An analysis of the total least squares problem," *SIAM Journal. Numerical. Analysis*, vol.17, no.6, pp. 883-893, December 1980.
[21] G. H. Golub and C. F. Van Loan, "Some modified matrix eigenvalue problem," *SIAM, Rev*, vol.15, pp. 318-334, 1973.
[22] S. V. Huffel and J. Vandewalle, *The total least squares problem: Computational aspects and analysis*, SIAM, 1991.
[23] L. J. Gleser, "Estimation in a multivariate "errors in variables" regression model: Large sample results," *The Annals of Statistics*, vol. 9, no. 1, pp. 24-44, 1981.
[24] S. V. Huffel and J. Vandewalle, "Comparison of total least squares and instrumental variable methods for parameter estimation of transfer function models," *Int. J. Control*, vol. 50, no. 4, pp. 1039-1056, 1989.
[25] S. V. Huffel and J. Vandewalle, "Analysis and properties of the generalized total least squares problem $A X \approx B$ when some or all columns of A are subject to errors," *SIAM J. Matrix Anal. Appl.*, vol. 10, pp. 294-315, 1989.
[26] C. E. Davila, "Recursive total least squares algorithm for adaptive filtering," *IEEE Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 1853-1856, May 1991.
[27] N. K. Bose, H. C. Kim, and H. M. Valenzuela, "Recursive total least squares algorithm for image reconstruction from noisy, undersampled frames," *Multidimensional Systems and Signal Processing*, vol. 4, pp. 253-268, 1993.
[28] I. Shavitt, C. F. Bender, A. Pipano, and R. P. Hosteny, "The iterative calculation of several of the lowest or highest eigenvalues and corresponding eigenvectors of very large symmetric matrices," Journal of Computational Physics, vol. 11, pp. 90-108, 1973.
[29] M. J. Levin, "Estimation of a system pulse transfer function in the presence of noise," *IEEE Trans. on Automatic Control*, vol. 9, no. 7, pp. 229-235, July 1964.
[30] D. K. Faddeev and V. N. Faddeeva, *Computational methods of linear algebra*, San Fransisco:Freeman, 1963.
[31] P. M. Clarkson, *Optimal and adaptive signal processing*, CRC Press, Inc., 1993.

Eung-Tae Kim received the B.S. degree (summa cum laude) in Electronics Engineering from Inha University, Inchon, Korea, in 1991, and the M.S. degree and the Ph.D. degree in electrical & electronics engineering from Korea Advanced Institute of Science and Technology (KAIST), Taejon, Korea, in 1993 and 1999, respectively. From 1998 to 2004, he was a senior researcher at the Digital TV Lab. of LG Electronics Co. Ltd. Since 2004, he has been with the Department of Electronics Engineering at Korea Polytechnic University. His research interests include low bit-rate video transmission techniques, MPEG system and video, wireless multimedia transmission, and DTV/DMB/DVR system.