Comments on "Modified K-means algorithm for vector quantizer design"

Correspondence

Comments on "Modified K-Means Algorithm for Vector Quantizer Design"

Kuldip K. Paliwal and V. Ramasubramanian

***Abstract*—Recently a modified K-means algorithm for vector quantization design has been proposed where the codevector updating step is as follows: new codevector = current codevector + scale factor (new centroid — current codevector). This algorithm uses a fixed value for the scale factor. In this paper, we propose the use of a variable scale factor which is a function of the iteration number. For the vector quantization of image data, we show that it offers faster convergence than the modified K-means algorithm with a fixed scale factor, without affecting the optimality of the codebook.**

***Index Terms*—Faster convergence, K-means algorithm, vector quantization.**

I. Introduction

Vector quantization is a powerful data compression technique used in a number of signal processing applications including speech coding, image coding, and speech recognition [1]–[6]. Design of a vector quantizer is accomplished by generating a codebook from the training data using a distortion measure appropriate for the given application. The K-means clustering algorithm (or the generalized Lloyd algorithm) is usually used for this purpose [7], [8]. This algorithm is iterative in nature and requires a large amount of computation time for convergence. The computation time mainly depends on the amount of training data, codebook size, vector dimension, and distortion measure.

Recently, a modified K-means algorithm for vector quantization design was proposed by Lee *et al.* [9] for faster convergence. In this algorithm, the codevector updating step is as follows: new codevector = current codevector + scale factor (new centroid − current codevector). This algorithm uses a fixed value for the scale factor. In this paper, we propose the use of a variable scale factor which is a function of the iteration number. For vector quantization of image data, we show that it offers faster convergence than the modified K-means algorithm with a fixed scale factor, without affecting the optimality of the codebook.

II. Modified K-Means Algorithm

Recently, Lee *et al.* [9] proposed a modified K-means algorithm which provides faster convergence rate than the conventional K-means algorithm [7]. This algorithm achieves acceleration in the convergence by using a "scaled" updating scheme where a codevector is updated along the direction of the local gradient by a step-size larger than that used by the centroid update of the conventional K-means algorithm. First, we briefly describe this "scaled" update scheme with respect to the conventional K-means algorithm.

Manuscript received December 3, 1997; revised June 5, 2000. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Antonio Ortega.

K. K. Paliwal is with the School of Microelectronic Engineering, Griffith University, Brisbane, QLD 4111, Australia (e-mail: K.Paliwal@me.gu.edu.au).

V. Ramasubramanian is with the Tata Institute of Fundamental Research, Bombay 400005, India.

Publisher Item Identifier S 1057-7149(00)09392-1.

The conventional K-means algorithm [7] for the design of a codebook $\mathbf{C} = \{\mathbf{c}_i,\ i = 1, \cdots, N\}$ of size N from the training data $T = \{\mathbf{x}_m,\ m = 1, \cdots, M\}$ is as follows.

1) *Initialization:* Iteration number $n = 0$; codebook at iteration n, $\mathbf{C}^n = \{\mathbf{c}_i^n,\ i = 1, \cdots, N\}$; convergence threshold ϵ.
2) *Partitioning:* Find the nearest-neighbor partition $V(\mathbf{c}_j^n) = \{\mathbf{x}_m \in T:\ Q(\mathbf{x}_m) = \mathbf{c}_j^n\},\ j = 1, \cdots, N$. Here, Q denotes vector quantization operation and is defined as follows: $Q(\mathbf{x}) = \mathbf{c}_j^n$ if $d(\mathbf{x}, \mathbf{c}_j^n) \le d(\mathbf{x}, \mathbf{c}_i^n),\ i = 1, \cdots, N$ under some distance measure $d(\mathbf{x}, \mathbf{y}), \mathbf{x}, \mathbf{y} \in R^K$.
3) *Codebook Update:* Update codevectors $\mathbf{C}^n = \{\mathbf{c}_j^n,\ j = 1, \cdots, N\}$ to $\mathbf{C}^{n+1} = \{\mathbf{c}_j^{n+1},\ j = 1, \cdots, N\}$ as
$$\mathbf{c}_j^{n+1} = \mathcal{C}(V(\mathbf{c}_j^n)) \tag{1}$$
where $\mathcal{C}(V(\mathbf{c}_j^n))$ is the centroid of the partition $V(\mathbf{c}_j^n)$ under the given distance measure $d(\mathbf{x}, \mathbf{y})$.
4) *Convergence Check:* Stop if $|d_n - d_{n-1}|/d_n \le \epsilon$, where $d_n = (1/M)\sum_{m=1}^{M} d(\mathbf{x}_m, Q(\mathbf{x}_m))$. Otherwise, replace n by $n + 1$ and go to Step 2.

The modified K-means algorithm proposed by Lee *et al.* [9] updates the current codevector $\mathbf{c}_j^n$ (at iteration n) to the new codevector $\mathbf{c}_j^{n+1}$ to be used at iteration $n + 1$ as
$$\mathbf{c}_j^{n+1} = \mathbf{c}_j^n + s(\mathcal{C}(V(\mathbf{c}_j^n)) - \mathbf{c}_j^n). \tag{2}$$
Under a squared-error distance measure, Lee *et al.* [9] have found experimentally that the modified K-means algorithm converges slower in comparison to the conventional K-means algorithm when $s < 1$. When $1 < s < 2$, it converges faster and results in better performance in terms of mean-squared error. When $s > 2$, the algorithm either does not converge, or converges very slowly with poor performance. The modified K-means algorithm gives the best results when the scale factor is set to a fixed value of $s = 1.8$. It may be noted that for $s = 1$, the modified K-means algorithm becomes the same as the conventional K-means algorithm with centroid-update as given by (1).

III. Variable Scale Update

The codevector updates obtained by the conventional centroid-update [see (1)] have progressively decreasing step-sizes [given by $d(\mathbf{c}_j^n, \mathbf{c}_j^{n+1})$] as the codevectors encode the training data with decreasing mean squared error at each iteration until convergence when the codevectors do not have any appreciable update. Therefore, while the use of a scaled-update as in (2) can accelerate the convergence, use of a "fixed" scaling for the entire range of iterations results in the use of step sizes larger than the corresponding centroid-update at iterations closer to convergence and causes undesirably high perturbations of the codevectors which are otherwise converging to some optimal configuration. This in turn has the effect of increasing the number of iterations required to converge as well as perturbing the codebook convergence to a poorer local optimum. This can also be seen from the convergence characteristics observed in [9] and [10] for fixed scale values larger than 2, where either the convergence is very slow or nonexistent due to the step-size being considerably larger than the conventional centroid-update.

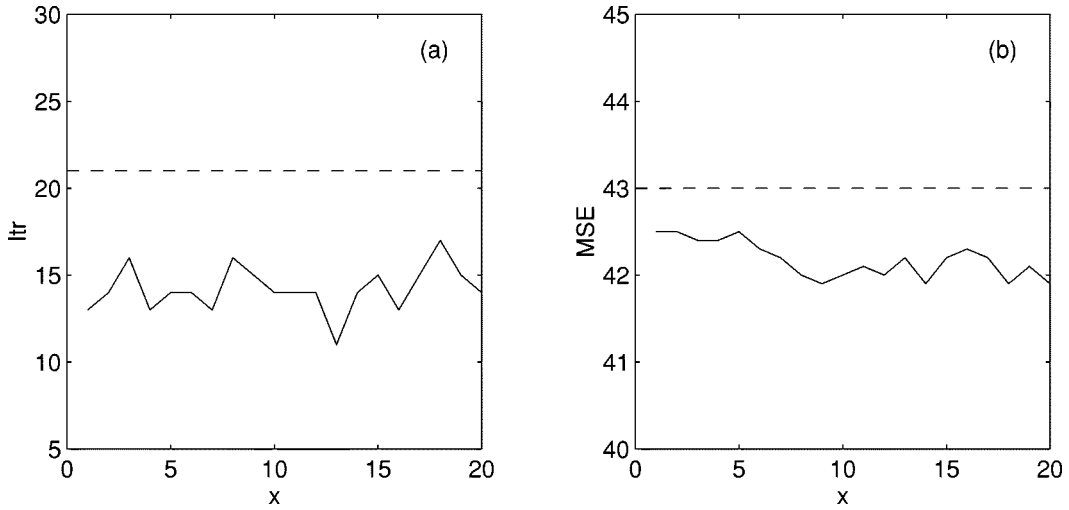

Fig. 1. (a) Number of iterations (Itr) required to reach convergence and (b) MSE at convergence for the NEW algorithm as a function of x (shown in solid line) on the Lena image data with dimension $K = 16$ and codebook size $N = 256$. Dashed horizontal lines show Itr and MSE for the CONV algorithm.

TABLE I
PERFORMANCE COMPARISON OF THE ALGORITHMS (CONV, MODI, AND NEW) FOR THE IMAGE DATA SET I

Initialization by minmax method							
Image	Alg	Codebook Size (N)					
		256		512		1024	
		Itr	MSE	Itr	MSE	Itr	MSE
Lena	CONV	25	44.4	23	33.1	15	24.6
	MODI	18	42.8	17	32.1	16	23.1
	NEW	18	42.9	15	32.4	12	23.2
Pepper	CONV	23	49.4	17	36.3	12	27.3
	MODI	18	47.7	16	35.1	16	25.8
	NEW	14	48.1	14	35.1	11	25.8
Baboon	CONV	21	250.5	16	211.4	13	173.6
	MODI	18	247.1	16	207.7	15	168.9
	NEW	19	248.2	12	209.0	11	168.8
Initialization by splitting method							
Image	Alg	Codebook Size (N)					
		256		512		1024	
		Itr	MSE	Itr	MSE	Itr	MSE
Lena	CONV	21	43.0	15	33.2	15	24.7
	MODI	19	42.2	14	32.2	15	23.8
	NEW	15	41.9	12	32.3	12	23.7
Pepper	CONV	16	46.9	14	36.3	14	27.4
	MODI	14	46.1	15	35.4	16	26.5
	NEW	12	46.0	10	35.9	11	26.3
Baboon	CONV	17	250.6	14	213.4	12	176.4
	MODI	15	248.5	14	208.9	13	171.0
	NEW	12	248.1	11	210.0	11	171.6

TABLE II
PERFORMANCE COMPARISON OF THE ALGORITHMS (CONV, MODI, AND NEW) FOR THE IMAGE DATA SET II

Initialization by minmax method									
Alg	Codebook Size (N)								
	256		512		1024		2048		4096	
	Itr	MSE	Itr	MSE	Itr	MSE	Itr	MSE	Itr	MSE
CONV	31	149.9	29	126.1	23	104.6	20	85.8	16	68.1
MODI	19	148.4	24	123.3	22	101.8	18	83.2	15	65.8
NEW	20	148.5	20	123.9	16	103.1	20	83.4	13	65.9
Initialization by splitting method										
Alg	Codebook Size (N)									
	256		512		1024		2048		4096	
	Itr	MSE	Itr	MSE	Itr	MSE	Itr	MSE	Itr	MSE
CONV	20	148.1	25	123.7	18	103.1	15	84.9	14	68.5
MODI	21	146.9	19	122.4	18	101.5	15	83.2	14	66.6
NEW	17	148.1	19	123.2	14	102.2	12	83.6	12	66.9

Therefore, in order to obtain codevector updates for iteration $n + 1$, which are reliable look-ahead estimates of conventional codevector updates corresponding to iterations $n + 2$, $n + 3$, $\cdots$, the scale factor in (2) should decrease with increase in iteration. Here, we propose the use of a scale factor s in the codevector update in (2) which varies as a function of the iteration n and is inversely proportional to n. This scale factor should satisfy the following two conditions. 1) It should be greater than 1 to ensure a faster convergence than the conventional K-means algorithm [9] and 2) it should be less than 2 to avoid a very slow or nonexistent convergence [9], [10]. A functional form (s as a function of n) that meets these requirements is as follows:

$$s = 1 + \frac{x}{(x+n)} \tag{3}$$

where $x > 0$. In this equation, $s = 2$ when $n = 0$, and $s = 1$ when $n = \infty$; thus, it satisfies the aforementioned conditions. Note that this type of functional dependence of the scale factor on the iteration number has been used earlier in a number of algorithms reported in the literature; e.g., algorithms used for stochastic approximation [11], [12], self-organizing feature map generation [13], simulated annealing [14], etc. Also, note that this is not the only functional form that satisfies the above-mentioned requirements. Other functional forms are also possible and may be investigated in future.

IV. SIMULATION RESULTS

We compare the convergence characteristics of the three K-means clustering algorithms, namely

1) CONV, which uses the conventional update given by (1);
2) MODI, which uses the modified update given by (2) with a fixed scale value of $s = 1.8$;
3) NEW, the algorithm proposed here, which uses a variable scale factor $s = 1 + x/(x + n)$ (where n is the iteration number).

The three K-means algorithms are also compared with two initializations as in [9]: 1) maximum distance initialization using the minmax method [15] and 2) splitting initialization [7].

The performance of the K-means algorithms are studied here under squared-error distance measure for the vector quantization for image

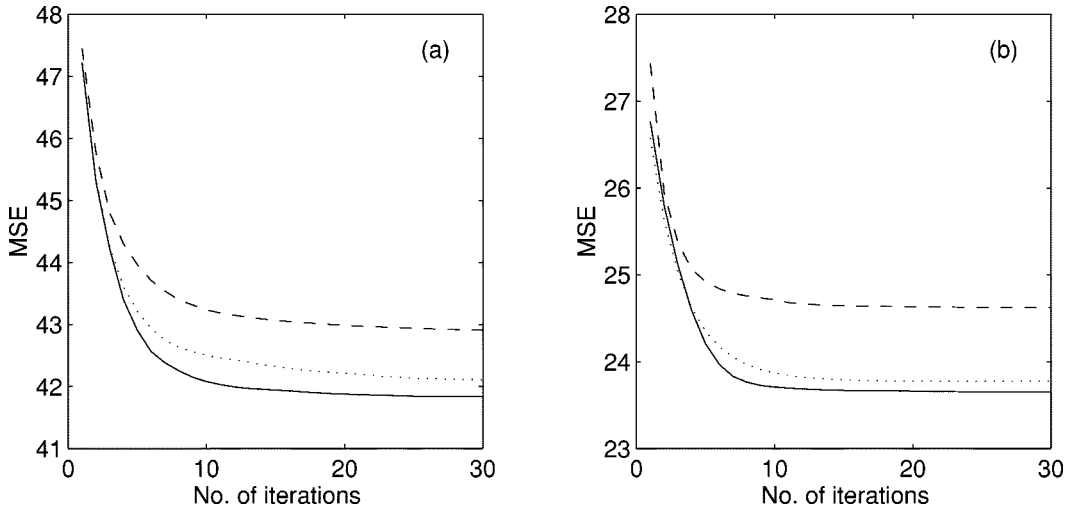

Fig. 2. MSE as a function of number of iterations for the CONV algorithm (dashed line), the MODI algorithm (dotted line) and the NEW algorithm (solid line) on the Lena image data with dimension $K = 16$. (a) Codebook size $N = 256$ and (b) Codebook size $N = 1024$.

data. For this we have used two sets of image data: 1) Set-I: Lena, Baboon and Pepper used separately as in [9] and 2) Set-II: A single data set consisting of the images (6386, baboon, bank, c43m, face, hat, vegas). Each of these monochrome images has 512×512 pixels with 256 gray levels.

We divide the images into a number of disjoint blocks, each of size 4×4 pixels and each resulting in a 16-dimensional vector. Different clustering algorithms are used to generate codebooks of sizes 256, 512, 1024, 2048, and 4096. These codebooks are then used to vector quantize the images. The quality of the coded image is evaluated in terms of mean-squared error (MSE) which is defined as

$$\text{MSE} = \frac{1}{IJ} \sum_{i=1}^{I} \sum_{j=1}^{J} (x_{ij} - y_{ij})^2, \tag{4}$$

where $I \times J$ is the size of the image and x_{ij} and y_{ij} are the pixel values of the original and coded images, respectively, at the coordinate (i, j). As done in [9], we also stop the algorithm and consider the convergence to be reached when the ratio of the MSE difference between the current and the preceding iterations and the MSE of the current iteration is less than 0.0005 or 0.05%.

Since the computation time for each iteration is almost the same for all the three algorithms, we use the number of iterations (Itr) needed to reach the convergence as a measure of the computational cost of an algorithm. The value of MSE at convergence is used as a measure of the optimality of the designed codebook. The same measures have been used by Lee *et al.* [9] to characterize the performance of the K-means algorithms.

In order to see the effect of variable x used in the scale factor equation $s = 1 + x/(x + n)$, we have studied the NEW algorithm with various values of x. Fig. 1 shows number of iterations (Itr) needed to reach convergence and the value of MSE at convergence as a function of x. As a reference, we also show in this figure by dashed horizontal lines the values of Itr and MSE for the CONV algorithm. It can be seen that the NEW algorithm performs better than the CONV algorithm for all the values of x. This is consistent with the results obtained by Lee *et al.* [9], as the scale factor in the NEW algorithm always satisfies the condition $1 < s < 2$ for all x. Also, note that the NEW algorithm works equally well for a wide range of x values. For the results reported hereafter in this paper, we use $x = 9$.

In Table I, we show the number of iterations (Itr) required for convergence and MSE for images from Set-I for codebook sizes $N = 256, 512, 1024$ and dimension $K = 16$ (using blocks of 4×4 pixels) using training data of 16 384 vectors. In Table II, we have used data from set-II for codebook sizes $N = 256, 512, 1024, 2048$ and 4096 and dimension $K = 16$ with 114 688 vectors. In order to illustrate the convergence behavior of the three algorithms, we show in Fig. 2 the MSE as a function of number of iterations for the Lena image with dimension $K = 16$ and codebook sizes $N = 256$ and 1024.

From these two tables and the figure, we can make the following observations: 1) the modified K-means algorithm (MODI), which uses a fixed scale value of 1.8, provides most of the times (but not always) faster convergence than the conventional K-means algorithm (CONV) and 2) the K-means algorithm proposed here, NEW, which updates codevectors using a variable scale function, always provides a faster convergence than the CONV algorithm. Also note that the NEW algorithm does not sacrifice the optimality of the codebook with respect to the CONV algorithm in the sense that it does not increase the MSE. In fact, it reduces the MSE compared to the conventional K-means algorithm, though this reduction may not be significant. It is important to note here that the modification proposed in this paper is very simple and easy to implement.

We have reported here results for dimension $K = 16$. However, we have also carried out experiments for dimensions $K = 4$ and 64. These experiments show results similar to those described above for the three K-means algorithms.

V. Conclusions

In this paper, we have proposed the use of a modified update step based on a variable scale size which is a function of the iteration and shown that it offers faster convergence than the modified K-means algorithm with a fixed scale update without effecting the optimality of the codebook. Though the scale factor in the proposed algorithm varies as a function of iteration number, it is same for all the components of a vector in a given iteration. In principle, it is possible to assign different scale values for different components of a vector according to some criterion which might improve the convergence behavior further. However, this has to be investigated in the future.

References

[1] H. Abut, Ed., *Vector Quantization*. Piscataway, NJ: IEEE Press, May 1990.
[2] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.
[3] W. B. Kleijn and K. K. Paliwal, Eds., *Speech Coding and Synthesis*. Amsterdam, The Netherlands: Elsevier, 1995.

[4] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," *IEEE Trans. Speech Audio Processing*, vol. 1, pp. 3–14, Jan. 1993.
[5] M. Barlaud *et al.*, "Guest editorial: Special issue on vector quantization," *IEEE Trans. Image Processing*, vol. 5, pp. 197–201, Feb. 1996.
[6] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Trans. Commun.*, vol. 36, pp. 957–971, Aug. 1988.
[7] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantization design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1980.
[8] S. P. Lloyd, "Least-squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 129–137, Mar. 1982.
[9] D. Lee, S. Baek, and K. Sung, "Modified k-means algorithm for vector quantizer design," *IEEE Signal Processing Lett.*, vol. 4, pp. 2–4, Jan. 1997.
[10] M. R. Anderberg, *Cluster Analysis for Applications*. New York: Academic, 1973.
[11] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, pp. 400–407, 1951.
[12] A. Benveniste, M. Metivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximations*. Berlin, Germany: Springer-Verlag, 1987.
[13] T. Kohonen, *Self-Organization and Associative Memory*. Berlin, Germany: Springer-Verlag, 1984.
[14] K. Zeger, J. Vaisy, and A. Gersho, "Globally optimal vector quantizer design by stochastic relaxation," *IEEE Trans. Signal Processing*, vol. 40, pp. 310–322, Feb. 1992.
[15] I. Katsavounidis, C. C. J. Kuo, and Z. Zhang, "A new initialization technique for generalized Lloyd iteration," *IEEE Signal Processing Lett.*, vol. 1, pp. 144–146, Oct. 1994.

A Least-Squares-Based 2-D Filtering Scheme for Stereo Image Compression

Sang-Hoon Seo, Mahmood R. Azimi-Sadjadi, and Bin Tian

***Abstract*—A two-dimensional (2-D) least squares (LS)-based filtering scheme for high fidelity stereo image compression applications is introduced in this correspondence. This method removes the effects of mismatching in a stereo image pair by applying the left image as the reference input to a 2-D transversal filter while the right image is used as the desired output. The weights of the filter are computed using a block-based LS method. A reduced order filtering scheme is also introduced to find the optimum number of filter coefficients. The principal coefficients and the disparity vectors are used together with left image to reconstruct the right image at the receiver. The proposed schemes were examined on a real stereo image pair for 3DTV applications and the results were benchmarked against those of the block-matching method.**

***Index Terms*—Least squares, stereo image compression, 2-D adaptive filtering.**

I. Introduction

Three-dimensional (3-D) video imaging has found applications in numerous areas such as 3DTV, computer games, augmented reality and surgical environments due to its capability in providing stereoscopic pictures with high resolution and great sensation of reality [1]. The compression schemes for stereo image sequences generally utilize the characteristics that there are strong spatial correlations between the right and left images as well as between the current and previous frames. The former correlation is exploited to reproduce pictures by using either the right or left image and a small amount of information that corresponds to the binocular parallax. This operation, which is similar to motion compensation generally used to predict motion in a sequence of images, is known as disparity estimation [2]–[9]. The disparity (mostly limited to the horizontal direction) vector, which requires considerably shorter length codes, will then be encoded. The decoded disparity vector is then used in conjunction with one image to generate the other one at the receiver.

Traditionally, motion estimation algorithms have been applied for disparity estimation. However, the estimation of disparity vectors requires greater accuracy compared with the estimation of motion vectors, since human eyes can see still objects sharper than moving ones and have higher resolution with 3-D images than with two-dimensional (2-D) images. In addition, there are mismatching problems between the left and right images that are caused by reflectivity/illumination differences, object occlusion, deformation,and noise that need to be compensated for.

Block-matching method is used for both disparity estimation [2], [3] as well as motion estimation [10], [11] due to its simplicity and low encoding overhead requirements. However, for disparity estimation this method has several shortcomings including blocking artifacts and lack of compensation ability for the mismatched areas. Several block-based methods were proposed [4]–[9] to provide better compensation ability with accurate disparity and/or motion estimation. The generalized block-matching method [4], [5] provides models of rotations and deformations of blocks between two images by employing the generalized spatial transformations such as affine, perspective and bilinear coordinate transformations. The phase-based methods [6], [7] use the characteristic that the phase difference in the phase domain can be related to the displacement vector between two blocks or pixels. Unlike the generalized block-matching method, these schemes are relatively insensitive to changes in the intensity and can represent displacement vector by subpixel accuracy using continuous phase information. However, these methods can not compensate for the intensity mismatching between two blocks. Baysian method [8], [9] attempts to model motion or disparity map using Markov random fields with the smoothness assumption among neighboring displacement vectors. To take care of the discontinuity of displacement vector at the object boundaries and occlusion, this method needs some *a priori* knowledge about these regions. This method does not provide compensation ability for intensity differences and further the computational effort to detect the depth discontinuity is very high.

In this correspondence, a new scheme using 2-D filtering is proposed which can be viewed as a modified version of the block-matching scheme utilizing a 2-D transversal filter to represent the effects of mismatching in stereo image pairs prior to disparity estimation. To minimize the number of filter coefficients for reconstructing the blocks, a reduced order filtering scheme is also proposed. Simulation results are presented which attest to the effectiveness of the proposed scheme in compensating for the mismatching effects when compared with the results of the standard block matching method.

II. Two-Dimensional Filtering Scheme for Disparity Estimation

The block diagram of the proposed system is shown in Fig. 1. In this scheme, the right image is considered as the desired image and the left image as the input image, X, to the 2-D transversal filter. The function of the filter is to represent the effects of mismatching between

Manuscript received June 19, 1998; revised May 22, 2000. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Rashid Ansari.

The authors are with the Signal/Image Processing Laboratory, Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523 USA (e-mail: azimi@engr.colostate.edu).

Publisher Item Identifier S 1057-7149(00)09397-0.