MOTION-COMPENSATED INTERFRAME CODING

by

Robert James Moorhead II

and

Sarah A. Rajala

CENTER FOR COMMUNICATIONS AND SIGNAL PROCESSING

Electrical and Computer Engineering

North Carolina State University

Raleigh, NC

December, 1984

CCSP-TR-84/12

Abstract

This paper addresses various aspects of motion-compensated interframe coding. The image sequence data is investigated in an attempt to develop a more thorough analysis of the convergence requirements and the convergence rate. A new motion prediction technique is presented which increases the validity of the assumptions made in proving convergence and which decreases the total prediction error. Simulation results are presented to indicate the improvement of the proposed motion prediction scheme and to indicate the results of the convergence analysis.

Table of Contents

	Abstract	
I.	Introduction	1
II.	The Basic Algorithm.	4
III.	Image Sequence Model	7
IV.	Convergence Analysis	12
V.	Initial Displacement Estimate.	19
VI.	Simulations.	27
VII.	Conclusions and Further Research	35
VIII.	References	37

I. Introduction

This paper will focus on the problem of detecting the motion contained in a sequence of images -- a major problem which has not been adequately solved.

In transmitting or storing image sequences, one seeks to minimize the amount of information that must be transmitted or stored. One approach is to use previously transmitted frames to predict the state of the current frame and then transmit only the difference between the actual frame and the predicted frame. Previous work [1-12] has shown that motion-estimation techniques can be used to improve these predictions of the intensity values in the images. In other words, predicting the motion and accessing the intensity values from the previous frame (or field) results in a better prediction of the intensity values than trying to predict the intensity values solely from previous intensity values.

There have been two approaches to motion estimation: block-matching and pel-recursion [6,9,10]. In block-matching a block of intensity values in a frame is compared with blocks of intensity values in the previous frame until a best-match is determined. From this an interframe displacement vector (how much the block has moved between frames) for the whole block can be estimated for the frame being transmitted. Poor estimations result

if all pixels in the block do not move the same way. Using the pel-recursive approach, a displacement is determined for each pel value. This technique allows for a more exact estimation of the intensity value and has the ability to handle scale changes (zooming, movement perpendicular to the image plane).

In both block-matching and pel-recursion the prediction can be backwards or forwards, i.e., the displacement can be determined from previously transmitted information only (backwards) or from past values and the current value (forwards). Forward prediction requires explicit transmission of information about the displacement value; backwards does not. The advantage of the forward technique is that a better estimate of the displacement vector reduces the error in the intensity prediction. However, the majority of the previous approaches have used backwards prediction, implying backwards prediction leads to: 1) reduced bit rates, 2) lower computational requirements, and/or 3) faster prediction/estimation techniques.

Although motion-estimation techniques have existed for over ten years [13], they have not exhibited superior motion (or intensity) prediction characteristics. In the following sections, a way to improve the motion prediction and thus the data compression will be developed. It will be shown that although convergence of the basic algorithm

has been proven, this proof of convergence and the resulting algorithms that have been implemented do not coincide. The major contribution of this work will be twofold: 1) the image sequence data will be investigated in an attempt to develop a more thorough analysis of the convergence requirements and the convergence rate, and 2) a new motion prediction technique will be presented which will increase the validity of the assumptions made in proving convergence and which will decrease the total prediction error.

The basic algorithm is presented in Section II. In Section III we present our image sequence model. In Section IV we derive the constraints on ϵ , the convergence coefficient, for convergence of the algorithm at each pel. In Section V, we present a new initial displacement estimation approach. Simulations verifying the proposed improvements are in Section VI. Conclusions are drawn and areas needing further research are noted in Section VII.

II. The Basic Algorithm

Netravali and Robbins [6] were one of the first to develop a pel-recursive technique to estimate the displacement for a moving object in a sequence. The development here will be very similar to theirs.

The intensity values within a frame are represented by $I(\underline{z}, t)$, where $\underline{z}$ is a two-dimensional spatial vector and t is the frame at time t . When no ambiguity occurs "frame t " is used in lieu of "the frame at time t ." Although the $I(\underline{z}, t)$ function is a sampled discrete-valued function, we assume that by interpolating we can view $\underline{z}$ and $I(\underline{z}, t)$ as continuous.

If an object moves with purely translational motion, then for some $\underline{d}$, where $\underline{d}$ is the two-dimensional spatial translation displacement vector of the object point during the time interval $[t-\tau, t]$,

$$I(\underline{z}, t) = I(\underline{z} - \underline{d}, t - \tau) \quad (1)$$

We define a function called the displaced frame difference:

$$DFD(\underline{z}, \hat{\underline{d}}^i) = I(\underline{z}, t) - I(\underline{z} - \hat{\underline{d}}^i, t - \tau) \quad (2)$$

where $\hat{\underline{d}}^i$ is an estimate of the translation vector. The DFD converges to zero as $\hat{\underline{d}}^i$ converges to the actual dis-

placement, $\underline{d}$, of the object point. Thus what we seek is an iterative algorithm of the form

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i + \text{update-term}, \quad (3)$$

where for each step, the update-term seeks to improve the estimate of $\underline{d}$. Our end goal is minimization of the magnitude of the prediction error, DFD. If a pel at location $\underline{z}_a$ is predicted with $\hat{\underline{d}}^i$ to have intensity $I(\underline{z}-\hat{\underline{d}}^i, t-\tau)$, resulting in a prediction error of $\text{DFD}(\underline{z}, \hat{\underline{d}}^i)$, the predictor should attempt to create a new estimate, $\hat{\underline{d}}^{i+1}$ such that $|\text{DFD}(\underline{z}, \hat{\underline{d}}^{i+1})| \leq |\text{DFD}(\underline{z}, \hat{\underline{d}}^i)|$. To apply minization techniques, a function is needed which is minimized when the DFD is equal to zero. This can be accomplished by minimizing $[\text{DFD}(\underline{z}, \hat{\underline{d}})]^2$. The most common approach is a steepest descent or gradient method. Although other techniques may have a faster rate of convergence, they require more computation and are more likely to diverge [14].

The resulting iterative equation is:

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i - \frac{\epsilon}{2} \nabla_{\underline{d}} [\text{DFD}(\underline{z}_a, \hat{\underline{d}}^i)]^2, \quad (4)$$

where $\nabla_{\underline{d}}$ is the two-dimensional gradient operator with respect to displacement $\hat{\underline{d}}^i$ and ϵ is a positive scalar constant called the convergence coefficient or coefficient on the update term. This can be seen to simplify to

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i - \varepsilon \text{DFD}(\underline{z}_a, \hat{\underline{d}}^i) \nabla_{\underline{d}} \text{DFD}(\underline{z}_a, \hat{\underline{d}}^i) \quad (5)$$

We can evaluate $\nabla_{\underline{d}}$ by using the definition of DFD in eq. (2) and noting that

$$\nabla_{\underline{d}} \text{DFD}(\underline{z}_a, \hat{\underline{d}}^i) = \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (6)$$

where $\nabla_{\underline{z}}$ is the two-dimensional spatial gradient operator with respect to $\underline{z}$.

This gives us

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i - \varepsilon \text{DFD}(\underline{z}_a, \hat{\underline{d}}^i) \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (7)$$

Note that for nonintegral $\hat{\underline{d}}^i$, DFD and $\nabla_{\underline{z}} I$ can be obtained by interpolation.

III. Image Sequence Model

Two consecutive frames of a sequence we call bobsjob are shown in figure 1. Each frame is 282 lines by 448 pels/line. Four blocks of data taken from moving edges in figure 1a are shown in figure 2. The change in the value of the spatial gradient as the edges are traversed in either the x or y direction can readily be seen to vary. It would be advantageous for the analysis to be done in Section IV if the edges could be modeled by either a linear or parabolic function. However an arctan function appears to be a much better model. In calculating $\nabla_z I$ we actually calculate the components $\nabla_x I$ and $\nabla_y I$. Thus it is those values we want to model. The intensity function in one dimension and the derivatives are:

$$\begin{aligned}
 I &= (A)\arctan(Bx+C)+D \\
 \frac{dI}{dx} &= \frac{AB}{1+(Bx+C)^2} \\
 \frac{d^2I}{dx^2} &= \frac{-2AB^2(Bx+C)}{[1+(Bx+C)^2]^2} \\
 \frac{d^3I}{dx^3} &= -2AB^3 \frac{1-3(Bx+C)^2}{[1+(Bx+C)^2]^3}
 \end{aligned}$$

The x for which $Bx+C=0$ is the center of the edge. Table I shows the values of the derivatives and the ratio of derivatives for various values of u, where $u=Bx+C$ is a measure of how far we are from the center of the edge. Note that the magnitudes of the higher order derivatives

decrease faster than dI/dX as $|u|$ increases for $B \approx 1$. However, the magnitudes of the higher order derivatives are significant for values of u close to zero.

bobsjob, frame 20

Figure 1a

Figure 1a

bobsjob, frame 21

Figure 1b

[185,215] - [200,230] (right thumb) first field															
64	64	64	63	63	62	63	64	64	63	62	62	61	61	62	61
63	63	63	60	62	64	64	68	72	76	81	77	70	67	63	61
63	63	61	63	66	73	83	96	111	122	132	133	126	116	96	74
64	64	65	70	83	96	111	123	133	139	140	144	142	134	121	103
65	69	83	92	106	120	127	136	136	138	139	136	134	124	112	101
76	88	100	114	125	129	133	133	132	132	130	125	117	108	95	86
86	103	116	124	133	135	133	132	129	121	119	113	102	94	83	78
89	101	110	120	127	128	129	125	123	120	107	101	93	80	76	72

[185,215]		[200,230]		(right thumb)		second field									
69	68	66	67	64	64	66	63	63	64	63	63	62	61	61	62
66	67	64	62	63	62	63	63	61	64	66	66	67	66	64	63
66	64	64	61	62	63	61	64	67	75	87	98	107	111	110	100
64	63	62	60	63	65	70	77	88	104	117	128	136	139	140	138
64	62	62	64	69	76	89	102	112	122	131	135	138	139	137	133
61	62	67	73	86	100	113	121	125	130	132	133	133	131	126	118
63	66	72	88	104	115	126	129	130	131	129	127	121	120	112	102
66	70	77	92	102	112	121	126	130	127	124	122	115	108	98	90

[210,400] - [225,415] (left elbow/bkg edge) first field															
53	54	54	56	63	75	108	136	143	144	143	144	142	143	146	145
52	51	52	55	53	54	61	79	113	135	142	143	139	146	145	144
54	53	53	55	53	54	55	55	60	87	124	138	142	145	144	147
53	53	51	51	53	52	53	53	54	58	73	109	136	140	141	144
52	53	53	51	50	52	50	51	54	53	54	66	97	130	140	140
51	52	54	54	51	52	50	49	51	50	51	54	61	84	121	142
49	48	50	51	52	53	50	49	50	48	47	50	53	60	81	120
50	48	48	49	49	50	50	50	49	47	49	46	44	52	59	81

[210,400]		[225,415]		(left elbow/bkg edge) second field															
52	52	55	54	55	60	70	98	131	143	143	142	142	144	146	145				
52	52	53	53	53	54	57	59	74	107	134	143	144	145	146	145				
54	53	54	55	55	54	53	55	56	61	88	124	141	141	144	145				
56	53	53	51	49	54	53	53	54	54	58	74	110	137	139	143				
52	53	54	53	54	51	50	53	51	51	52	55	65	96	130	141				
50	49	52	53	52	52	51	50	50	49	50	52	55	63	88	126				
51	49	49	49	51	52	51	50	50	49	47	47	49	55	61	86				
51	49	50	50	49	51	51	51	49	49	47	47	46	46	53	60				

10

TABLE I

u	$\frac{dI}{dx}$	$\frac{d^2I}{dx^2}$	$\frac{d^3I}{dx^3}$	$\frac{d^2I}{dx^2} / \frac{dI}{dx}$	$\frac{d^3I}{dx^3} / \frac{dI}{dx}$
-4	0.06AB	0.03AB ²	0.02AB ³	0.47B	0.33B ²
-3	0.10AB	0.06AB ²	0.05AB ³	0.60B	0.52B ²
-2	0.20AB	0.16AB ²	0.18AB ³	0.80B	0.88B ²
-1	0.50AB	0.50AB ²	0.50AB ³	1.00B	1.00B ²
-0.50	0.80AB	0.64AB ²	-0.26AB ³	0.80B	-0.33B ²
-0.33	0.90AB	0.54AB ²	-0.97AB ³	0.60B	-1.08B ²
-0.25	0.94AB	0.44AB ²	-1.35AB ³	0.47B	-1.44B ²
0	1.00AB	0	-2.00AB ³	0	-2.00B ²
0.25	0.94AB	-0.44AB ²	-1.35AB ³	-0.47B	-1.44B ²
0.33	0.90AB	-0.54AB ²	-0.97AB ³	-0.60B	-1.08B ²
0.50	0.80AB	-0.64AB ²	-0.26AB ³	-0.80B	-0.33B ²
1	0.50AB	-0.50AB ²	0.50AB ³	-1.00B	1.00B ²
2	0.20AB	-0.16AB ²	0.18AB ³	-0.80B	0.88B ²
3	0.10AB	-0.06AB ²	0.05AB ³	-0.60B	0.52B ²
4	0.06AB	-0.03AB ²	0.02AB ³	-0.47B	0.33B ²

IV. Convergence Analysis

Two of the important issues of this type of algorithm are to guarantee convergence and to determine the rate of convergence. Almost all previously presented pel-recursive techniques have used only one iteration per pel and assumed convergence over time [6-8,10-12]. We are interested in convergence at each pel for which the $|DFD| > T$. In other words, we want the algorithm to converge to the correct displacement at every moving pel, not just within the moving area.

We can prove that the gradient algorithm (eq. (7)) which attempts to minimize the DFD converges, under certain conditions, to the true displacement. The first requirement is that the area between $DFD(\underline{z}_a, \hat{\underline{d}}^0)$ and $DFD(\underline{z}_a, \underline{d})$ must be a concave surface, a minimum point of which is $DFD(\underline{z}_a, \underline{d})$. ($\hat{\underline{d}}^0$ is the initial estimate of the displacement and $\underline{d}$ is the actual displacement.) The proof procedure used is similar to one in [6]. The assumptions are that the motion is purely translational and that the uncovered background is neglected. Substitute equation (2) into (7) and obtain

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i - \epsilon \{I(\underline{z}_a, t) - I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau)\} \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (8)$$

Substituting from (1) for $I(\underline{z}_a, t)$,

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i - \varepsilon \{ I(\underline{z}_a - \underline{d}, t - \tau) - I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \} \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (9)$$

Expanding the term in braces using a Taylor series expansion, we obtain

$$\begin{aligned} I(\underline{z}_a - \underline{d}, t - \tau) - I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) = \\ (\hat{\underline{d}}^i - \underline{d})^T \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) + \\ \frac{1}{2} (\hat{\underline{d}}^i - \underline{d})^T \nabla_{\underline{z}}^2 I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) (\hat{\underline{d}}^i - \underline{d}) + \\ O(\hat{\underline{d}}^i - \underline{d})^3, \end{aligned} \quad (9a)$$

where $\nabla_{\underline{z}}^2 I()$ is the 2×2 matrix of the second partial derivatives of $I()$ and $O(\hat{\underline{d}}^i - \underline{d})^3$ represents the higher order terms in $(\hat{\underline{d}}^i - \underline{d})$.

The $O(\hat{\underline{d}}^i - \underline{d})^3$ terms cannot be expressed in matrix notation; an open form must be used. Let

$$(\hat{\underline{d}}^i - \underline{d}) = \begin{bmatrix} \Delta d_x^i \\ \Delta d_y^i \end{bmatrix} \quad (9b)$$

and

$$\nabla_{\underline{z}} I() = \left\{ \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} \right\} I().$$

The Taylor series expansion of the difference in the two intensity values can now be rewritten as

$$I(\underline{z}_a - \underline{d}, t - \tau) - I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) =$$

$$\sum_{j=1}^{\infty} \frac{1}{j!} \left\{ \Delta d_x^i \frac{\partial}{\partial x} + \Delta d_y^i \frac{\partial}{\partial y} \right\}^j I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (10)$$

where Δd_x^i is the error in the x displacement estimate at the i th iteration, where Δd_y^i is the error in the y displacement estimate at the i th iteration, and the quantity in braces is an operator on the intensity function.

The number of terms in the Taylor series that are required to obtain a good estimate of the difference in the two intensity values is dependent on two factors: 1) the error in the displacement estimation, and 2) the magnitude of the higher order derivatives of the intensity function. There is no reason to assume that a sufficient estimate is always obtained by retaining only the first term of the Taylor series expansion.

Substituting (10) into (9),

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i$$

$$- \left\{ \sum_{j=1}^{\infty} \frac{1}{j!} \left\{ \Delta d_x^i \frac{\partial}{\partial x} + \Delta d_y^i \frac{\partial}{\partial y} \right\}^j I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \right\} \nabla_z I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (11)$$

where we choose to use mixed notation for compactness. After some intermediate algebra and regrouping of factors, we obtain

$$\hat{\underline{d}}^{i+1} = \hat{\underline{d}}^i - \varepsilon \{ \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \underline{f} \} (\hat{\underline{d}}^i - \underline{d}) \quad (12)$$

where $\underline{f}$ is a 1×2 matrix defined as

$$\underline{f} = \sum_{j=1}^{\infty} \frac{1}{j!} \left\{ \frac{\partial}{\partial x} \quad \frac{\partial}{\partial y} \right\} \left\{ \Delta d_x^i \frac{\partial}{\partial x} + \Delta d_y^i \frac{\partial}{\partial y} \right\}^{j-1} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \quad (12a)$$

For example, the first two terms of $\underline{f}$ are:

$$\begin{aligned} j=1: \quad \underline{f}_1 &= \left\{ \frac{\partial}{\partial x} \quad \frac{\partial}{\partial y} \right\} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \\ &= \left\{ \frac{\partial}{\partial x} \quad \frac{\partial}{\partial y} \right\} \bigg|_{(\underline{z}_a - \hat{\underline{d}}^i, t - \tau)} \\ &= \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau)^T \\ j=2: \quad \underline{f}_2 &= \frac{1}{2} \left\{ \frac{\partial}{\partial x} \quad \frac{\partial}{\partial y} \right\} \left\{ \Delta d_x^i \frac{\partial}{\partial x} + \Delta d_y^i \frac{\partial}{\partial y} \right\} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \\ &= \frac{1}{2} \left\{ \Delta d_x^i \frac{\partial^2 I}{\partial x^2} + \Delta d_y^i \frac{\partial^2 I}{\partial y \partial x} \quad \Delta d_x^i \frac{\partial^2 I}{\partial x \partial y} + \Delta d_y^i \frac{\partial^2 I}{\partial y^2} \right\} \bigg|_{(\underline{z}_a - \hat{\underline{d}}^i, t - \tau)} \\ &= \frac{1}{2} (\hat{\underline{d}}^i - \underline{d})^T \nabla_{\underline{z}}^2 I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \end{aligned} \quad (12b)$$

Subtracting $\underline{d}$ from both sides of (12), we get

$$(\hat{\underline{d}}^{i+1} - \underline{d}) = (\hat{\underline{d}}^i - \underline{d}) - \varepsilon \{ \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \underline{f} \} (\hat{\underline{d}}^i - \underline{d}) \quad (13)$$

or

$$(\hat{\underline{d}}^{i+1} - \underline{d}) = [J - \varepsilon \{ \nabla_{\underline{z}} I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \underline{f} \}] (\hat{\underline{d}}^i - \underline{d}) \quad (14)$$

where J is the appropriate size identity matrix.

This equation is of the form $\mathbf{e}_{i+1} = \mathbf{A}_i \mathbf{e}_i$ where $\mathbf{e}$ is the 2×1 error vector which we desire to reduce as $i \rightarrow \infty$ and

A is a 2×2 matrix. This can be rewritten as $e_k = (\prod_{i=0}^k A_i) e_0$. If we can show that the $\|e_k\|$ tends to zero as $k \rightarrow \infty$, we have proved convergence. Assume $B = \prod_{i=0}^k A_i$ as $k \rightarrow \infty$. $Be_0 = 0$ if $e_0 \in N(B)$, the null space of B. Alternatively we can say that $Be_0 = 0$ if the spectral radius of B is less than 1. Unfortunately neither of these is necessarily the case.

Closer inspection of the matrix we have called A will reveal it can be rewritten as $[J - \varepsilon C]$, where C is rank deficient, independent of the number of terms retained in the Taylor series expansion. Thus the spectral radius of A is ≥ 1 depending on ε . If the spectral radius of A is 1, the algorithm could converge.

Investigation of stochastic gradient algorithms shows that convergence properties are frequently determined by analyzing the behavior of the ensemble average [15-18]. Taking the expected value of both sides, assuming the two factors on the right-hand side are uncorrelated, and applying Schwartz inequality,

$$\overline{\|\mu_{\hat{d}}^{i+1} - \mu_d\|} \leq \overline{\|J - \varepsilon \nabla_z I(\underline{z}_a - \hat{\underline{d}}^i, t - \tau) \underline{f}\|} \cdot \overline{\|\mu_{\hat{d}}^i - \mu_d\|}, \quad (15)$$

where the overbar denotes expected value, μ_d is the expected value of $\underline{d}$, and $\mu_{\hat{d}}$ is the expected value of $\hat{\underline{d}}$.

This can be rewritten as

$$||\mu_{\hat{d}}^{i+1} - \mu_d|| \leq |1 - \varepsilon \lambda_{\max}| \cdot ||\mu_{\hat{d}}^i - \mu_d||, \quad (16)$$

where $\lambda_{\max}$ is the maximum eigenvalue of the positive semidefinite symmetric matrix

$$\nabla_z I(\underline{z}_a - \underline{\hat{d}}^i, t - \tau) \underline{f}$$

For convergence of the algorithm, we need

$$|1 - \varepsilon \lambda_{\max}| < 1, \quad (17)$$

which is equivalent to

$$\frac{2}{\lambda_{\max}} > \varepsilon > 0 \quad (18)$$

Note that since we only have two eigenvalues and the sum of the eigenvalues of any matrix equals the trace, $\lambda_{\max}$ is upper bounded by the trace and lower bounded by one half of the trace. Thus for convergence the following condition is sufficient:

$$4/\text{tr}[\nabla_z I(\underline{z}_a - \underline{\hat{d}}^i, t - \tau) \underline{f}] > \varepsilon > 0 \quad (19)$$

If $\underline{f}_1 \approx \underline{f}$ (this assumption is most valid at the edges of an edge), then eq. (19) can be rewritten as

$$\frac{4M}{\sum_{i=1}^M [\nabla_x I^2(\underline{z}_a - \underline{\hat{d}}^i, t - \tau) + \nabla_y I^2(\underline{z}_a - \underline{\hat{d}}^i, t - \tau)]} > \varepsilon > 0 \quad (20)$$

where M iterations are performed at each pel in the moving

area. Note that M can vary from pel to pel. $\nabla_x I$ and $\nabla_y I$ are the orthogonal components of $\nabla_z I$, i.e., the gradient vector elements. The ∇I^2 notation indicates the square of the real-valued gradient component. Note the difference in ∇I^2 in eq. (20) and $\nabla^2 I$ in eq. (9a) and (12b).

Netravali and Robbins [6] derived an equation very similar to eq. (20) for a slightly altered problem. Their proof assumed convergence over time, i.e. within the moving area, with only one iteration per pel, whereas we are seeking convergence at each and every pel in the moving area. They made no attempt to verify that the ϵ they used (.0625) met their required condition. The major conflict between analytical requirements and real-time implementation is that making ϵ small enough to insure convergence forces the convergence rate to be slow. In Section VI we will show that for our sequences, ϵ should be somewhere in the neighborhood of .0040 for low interframe motion and .0200 for high interframe motion.

V. Initial Displacement Estimate

There have been two predominant methods of initial displacement estimation: spatial and temporal. Most researchers have used a spatially adjacent displacement vector as an initial estimate [6-8,11,12,19]. Others, mostly from Bell Northern Research [1,10], proposed predicting the displacement along the temporal axis. We propose a third approach: project the motion estimation forward along the motion trajectory. This would have four advantages and require a minimal increase in computation and memory over the temporal projection procedure. First let us note the problems with the present schemes.

By always using a spatially adjacent displacement vector as an initial estimate for the displacement vector under consideration, an implicit assumption is being made that the displacement vectors always have a high spatial correlation. This is not what the original assumed model implies. The original model assumed that an object is moving over a fixed stationary background. Although the displacement vectors are highly correlated within the moving object and in the stationary background, at the edges of that moving object the displacement vectors are very uncorrelated. It has been shown that the edges of an object may be what contribute most to the estimation of the displacement vectors [20]. Let us look at a one-

dimensional example.

In figure 3 an edge has moved three units to the left between frames. As we scan left to right, we first encounter a nonzero DFD at point x_a . However $\hat{d}^0=0$ and the incorrect displacement estimate cannot be corrected since $\nabla_x I|_{t-\tau}=0$. It is not until we reach point x_b that we can correct the motion estimation (i.e., $\nabla_x I$ becomes nonzero). In other words no matter how many times we iterate at x_a , the correct displacement value cannot be found for $I(x_a, t)$. If the correct $\underline{d}$ has not been determined by the time we reach x_d in the scan line, it will not be corrected further until the spatial gradient becomes nonzero again, which may not occur until much later in the scan line.

Dubois et al [1,10] suggested using the temporally adjacent displacement vector as an initial estimate. By projecting the displacement vector estimates over time rather than space, the displacement estimates at the edges can exhibit a sharp discontinuity and this discontinuity can be sharpened over time. However, this approach does not fully solve the problem. It assumes that the location of the moving objects remain the same frame-to-frame. A better approach would be to assume the motion remained the same. Instead of projecting the motion estimations forward parallel to the temporal axis, project them forward

Figure 3

along the motion trajectory.

As one example, look again at figure 3. We have the same problem as with spatially adjacent estimation: $\hat{\underline{d}}^0 = 0$ at x_a . There is no way to converge to $\underline{d}$ at x_a . The improvement of temporal prediction occurs at x_b where $\hat{\underline{d}}^0$ is not necessarily zero. This improvement in effect costs an extra frame buffer to store the $\hat{\underline{d}}^0$ from frame to frame.

As another example, consider an object moving to the left in the plane of view with a constant translational velocity. (See figure 4.) If the displacement vectors are projected forward parallel to the temporal axis, then there will be errors associated with both the leading and the trailing edge. The intensities along the leading edge (area L in figure 4) will not be predicted correctly since in the previous frame (at time $t-\tau$), nothing was moving in those pixel locations into which the leading edge has now moved. The trailing edge (area T in figure 4) on the other hand has left some pixel locations between time $t-\tau$ and t . The intensities at these pixel locations at time t constitute newly uncovered background. The algorithm will try to predict the intensities for these pixels from displaced intensities in the previous frame. The goodness of this prediction will depend on the correlation between the intensity values in the displaced region in the previous

Figure 4

frame and the intensity values in the newly uncovered background region in the present frame (at time t).

If the object has a constant velocity frame-to-frame, projecting the displacement vectors forward in the direction of motion will correctly predict the leading edge values. Also, those areas of the image which contain newly uncovered background can be detected easily.

By projecting the motion vectors forward in the direction of motion, we have solved a problem that has existed in the implementation of the algorithm. In proving convergence we noted that the uncovered background was neglected. Yet most algorithms [1,5-8,10,11,19] attempt to determine the intensity values for the newly uncovered background at time t using intensities in the frame at time $t-\tau$. The structure of the algorithm is at fault. By obtaining the initial estimates for the displacement vector from spatially or temporally adjacent pels there is no way to detect what regions are newly uncovered background. By predicting the motion vectors forward in the direction of motion, the uncovered background will have no displacement values predicted for it. The uncovered background is then easily detected, allowing a better predictor to be used for it and allowing the implementation to be a true implementation of the algorithm which was proved to converge.

To reiterate and summarize, by projecting the displacement estimates forward along the motion trajectory we obtain four improvements:

- 1) With respect to spatial prediction, sharp discontinuities can exist at the boundaries between moving objects and the background.
- 2) With respect to temporal prediction, the actual displacement of the object point can be found more often since the motion, not the location, of the moving area is assumed constant.
- 3) The number of iterations required for convergence at other points will be decreased due to a better initial estimate. Also a smaller displacement prediction error allows a larger ϵ which increases the convergence rate.
- 4) A substantial portion of the uncovered background is detectable and can be segmented out.

Computation requirements increase only slightly (mostly logical) and the memory requirements are only slightly greater than with temporal prediction. Either constant interframe velocity or constant interframe acceleration can be assumed.

A quantitative comparison of the three displacement prediction schemes (spatial, temporal, and "projection-along-the-motion-trajectory") is difficult. The number of pels for which the $\underline{d}$ can be found using "projection-along-the-motion-trajectory", but which cannot be found using spatial prediction is simply the number of pels in frame t with $|DFD| > T$ and $\nabla_z I = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, assuming that the motion is constant frame-to-frame. Using temporal prediction instead of spatial prediction the same holds true assuming the "correct" $\underline{d}$ was found for all pels in the moving area in the previous frame. Note that these assumptions are rather stringent. The reduction in the number of iterations required is much more difficult to ascertain, as is the relative improvement in being able to identify the uncovered background portion of a frame.

VI. Simulations

We have obtained a set of image sequences exemplary of sequences which might be transmitted in a video-teleconferencing environment. The major objective in capturing these scenes was to capture a few short sequences (approximately 5 seconds or 150 frames) which would be the best representative of the scenes that might be transmitted in a video-teleconferencing environment; the major objective was not to find some sequences which would make the proposed algorithm look best. The following simulation runs were done on one 60-frame sequence (2 seconds) in which a seated speaker gestures with his arms. The percent motion in the sequence varied from 2% to 20%.

We have chosen to use a noise reduction pre-filtering procedure in all our simulations to improve both picture quality and motion estimation. The procedure is a point operation followed by a local operation involving 5 pel values from each of two consecutive frames. In hardware it could be easily implemented as a pipelined unit preceding the prediction circuit. Many other researchers in simulating their algorithm have utilized a similar type of pre-filter [6-8,19,21,22].

Since we are only able to obtain an estimate of the gradient due to sampling and since the noise reduction scheme does not remove any noise in the moving area, the

displacement updates need to be clipped to prevent noisy estimates and unnecessary iterations [6-8,10,11,19,23-25].

Transmission of the sequence was simulated using the following procedure:

1) Using an initial estimate of the displacement vector, $\hat{\underline{d}}^0$, predict the intensity of the current pel by obtaining an intensity value from the previous frame at the offset $\hat{\underline{d}}^0$ from the current pel location $\underline{z}$. If $(\underline{z} - \hat{\underline{d}}^0)$ is not an integer multiple of both spatial sampling intervals, then bilinear interpolation is used to obtain the predicted intensity value.

2) Find $\text{DFD}(\underline{z}, \hat{\underline{d}}^0)$ by taking the difference of the actual current pel value and its predicted value.

3) If $|\text{DFD}(\underline{z}, \hat{\underline{d}}^0)| \geq T$, transmit a quantized value of the DFD to the receiver, preceded by the relative address from the previous point at which $|\text{DFD}(\underline{z}, \hat{\underline{d}}^0)| \geq T$.

4) a. If $|\text{DFD}(\underline{z}, \hat{\underline{d}}^0)| < T$, set $\hat{\underline{d}} = \hat{\underline{d}}^0$. (The displacement vector was predicted correctly.)

b. Else if $\sum_{\text{area}} \text{FD} \leq \sum_{\text{area}} \text{DFD}$, set $\hat{\underline{d}} = 0$.

c. Else use the reconstructed value of the

current pel to find a $\hat{\underline{d}}^{i-1}$, until a $\hat{\underline{d}}$ is obtained for which $|\text{DFD}(\underline{z}, \hat{\underline{d}})| < T$ or until the maximum number of allowed iterations is exceeded.

5) Go to next pel.

The first pair of simulation runs was intended to show the advantage of projecting the initial displacement along the motion trajectory. Constant velocity between frames was assumed. A 35-level symmetric quantizer was used and the motion-update algorithm allowed to iterate at most one time. Since the update algorithm was only allowed to iterate once, the update was clipped at 1/16 of a sampling interval. This has been shown experimentally to be the best value for at least one set of sequences [6-8] and our preliminary experiments indicated it was nearly optimal for ours.

The two runs vary in only one point: how the initial estimate of the motion vector is obtained. Run I used the $\hat{\underline{d}}$ of the (x-1,y-1) pel as an initial estimate of the displacement vector. In Run II the initial estimates are projected forward along the path of the motion trajectory from frame to frame.

Figure 5 is a plot of the entropy (bit rate) for both runs. Using Run I as a baseline, Run II shows a 36%

Figure 5

reduction in maximum bit rate for a frame and a 30% reduction in total bit rate. The maximum bit rate for a frame has great hardware-implementation implications since it determines the size of the transmission buffer which is required to maintain a constant bit rate.

With respect to signal-to-noise ratio (SNR) there was minimal difference in values between runs (Figure 6). The SNR for the predicted frame, however, was highly dependent on frame-to-frame motion. For 2% motion, the SNR for both runs was on the order of 58-62 dB; for 18% motion, the SNR for both runs was about 28 dB. For the 60 frames, using Run I as a baseline, the predicted frames using Run II were about 0.7 dB better.

The second set of runs was intended to show the improvement when the displacement estimation algorithm was allowed to converge at each and every pel. Preliminary runs using $\epsilon = .0010$ indicated that $\sqrt{\frac{2}{N_x}}I + \sqrt{\frac{2}{N_y}}I$ varied between 1000 for low interframe motion and 200 for high interframe motion. Substituting these values in eq. (20) indicates ϵ must be less than .0040 for low interframe motion and less than .0200 for high interframe motion. Therefore with $\epsilon = .0010$ the displacement estimation algorithm should converge at every pel if it is allowed to iterate long enough.

Figure 6

Given the improvement of Run II over Run I, a run similar to Run II was chosen to be the baseline for the this pair of runs. Run III is the same as Run II except a 33-level quantizer is used. Run IV is the same as Run III except ϵ is reduced from .0625 to .0010 and 30 iterations were allowed instead of just one.

Figure 7 is a plot of the entropy (bit-rate) for both runs. From previous simulation runs with the bobsjob sequence, it was determined that 30 frames (1 sec) were sufficient to compare the two approaches. Using Run III as a baseline, Run IV shows a 11% increase in both maximum bit rate for a frame and in total bit rate for the 30 frame sequence. This increase in bit-rate seems to indicate that 30 iterations were insufficient for the algorithm to converge when ϵ is reduced to .0010. The difference in SNR between the two runs was again minuscule.

Figure 7

VII. Conclusions and Further Research

It appears motion correction updates are being attempted based on noise as well as real motion. This seems the most plausible explanation for the noisy predicted images we have seen. There is also a disagreement in the theoretical and experimental "optimal" ϵ 's. One technique that might reduce the noise and the number of unnecessary iterations is to clip the maximum update pel-to-pel instead of (or as well as) at each iteration [10]. Another technique that could reduce the unnecessary iterations is a post-processing filter to smooth the displacements within the moving areas and within the background. Predicting field-to-field motion instead of frame-to-frame motion [6-8,10] also might help.

The improvement obtained by projecting the initial estimate of the motion vector forward along the motion trajectory is excellent even when performing only one iteration per pel.

A potential solution to the inability to correct the displacement vector could be averaging the spatial gradients in the two frames with a 2:1 weighting. (Less weight is given to the values obtained from the frame being transmitted since backwards differences must be used.) At least this would yield a nonzero spatial gradient in the correct direction without putting too much

weight on a noisy estimate. Alternatively one could first see if the $\nabla_z I|_{t-\tau} = 0$ when $|DFD| > T$ and then (and only then) use $\nabla_z I|_t$.

We realize the need to verify these preliminary results with more simulation runs and intend to simulate the algorithms on at least 2 more two second (60 frame) sequences.

Two questions remain about the analysis:

1) Can we assume that the mixed partial derivatives with respect to the intensity function are zero? If so the analysis of ϵ for convergence is much easier.

2) Is our analysis of $\prod_{i=0}^k A_i$ the best approach? It is the approach which has been used when only one iteration is performed at each sample point and convergence over time is sought.

VIII. References

- [1] E. Dubois, B. Prasada, and M.S. Sabri, "Image Sequence Coding," in Image Sequence Analysis, (T. S. Huang, Ed.), pp. 229-287, Springer-Verlag, Berlin, 1981.
- [2] J. O. Limb and J. A. Murphy, "Measuring the Speed of Moving Objects from Television Signals," IEEE Trans. on Communications, Vol. COM-23, No. 4, pp. 474-478, April 1975.
- [3] J. O. Limb and J. A. Murphy, "Estimating the Velocity of Moving Images in Television Signals," Computer Graphics and Image Processing, Vol. 4, pp. 311-327, 1975.
- [4] C. Cafforio and F. Rocca, "Methods for Measuring Small Displacements of Television Images," IEEE Trans. on Inform. Theory, Vol. IT-22, No. 5, pp. 573-579, Sept. 1976.
- [5] C. Cafforio and F. Rocca, "Tracking Moving Objects in Television Images," Signal Processing, Vol. 1, pp. 133-140, 1979.
- [6] A. N. Netravali and J. D. Robbins, "Motion Compensated Television Coding, Part I," BSTJ, Vol. 58, No. 3, pp. 631-670, March 1979.
- [7] J. D. Robbins and A. N. Netravali, "Interframe Television Coding using Movement Compensation," Proc. Int. Conf. Communications, pp. 23.4.1-23.4.5, 1979.
- [8] A. N. Netravali and J. D. Robbins, "Motion-Compensated Coding: Some New Results," BSTJ, Vol. 59, No. 9, pp. 1735-1745, Nov. 1980.
- [9] T. Ishiguro and K. Iinuma, "Television Bandwidth Compression Transmission by Motion-Compensated Coding," Communications, Vol. 20, No. 6, pp. 24-30, Nov. 1982.
- [10] R. Paquin and E. Dubois, "A Spatio-Temporal Gradient Method for Estimating the Displacement Field in Time-Varying Imagery," Computer Vision, Graphics, and Image Processing, Vol. 21, pp. 205-221, 1983.
- [11] D.R. Walker and K.R. Rao, "New Techniques in Pel-Recursive Motion Compensation," Proc. Int. Conf. Communications, pp. 703-706, May 1984.
- [12] D.R. Walker and K.R. Rao, "Improved Pel-Recursive

Motion Compensation," IEEE Trans. on Communications, Vol. COM-32, No. 11, pp. 1128-1134, Oct. 1984.

[13] B.G. Haskell and J.O. Limb, "Predictive Video Encoding Using Subject Velocity," U.S. Patent 3,632,865, Jan. 1972.

[14] D.G. Luenberger, Introduction to Linear and Nonlinear Programming, Addison-Wesley, Reading, MA, 1973.

[15] B. Widrow, J. Glover, J. McCool, et. al., "Adaptive Noise Canceling: Principles and Applications," Proc. IEEE, Vol. 63, No. 12, pp. 1692-1716, Dec. 1975.

[16] L. Ljung, "Analysis of Recursive Stochastic Algorithms," IEEE Trans. on Automatic Control, Vol. AC-22, No. 4, pp. 551-575, Aug. 1977.

[17] B. Widrow and J. M. McCool, "A Comparison of Adaptive Algorithms Based on the Methods of Steepest Descent and Random Search," IEEE Trans. Antennas and Propagat., Vol. AP-24, pp. 615-637, Sept. 1976.

[18] S.T. Alexander, "Adaptive Image Compression Using the Least Mean Square (LMS) Algorithm," Ph.D. Dissertation, North Carolina State University, 1982.

[19] J. D. Robbins and A. N. Netravali, "Spatial Subsampling in Motion-Compensated Television Coders," BSTJ, Vol. 61, No. 8, pp. 1895-1910, Oct. 1982.

[20] W. E. Snyder, S. A. Rajala, and G. Hirzinger, "Image Modeling: The Continuity Assumption and Tracking," Proc. 1980 Int. Joint Conf. Pattern Recognition, pp. 1111-1114, 1980.

[21] B. G. Haskell, P. L. Gordon, R. L. Schmidt, and J. V. Scattaglia, "Interframe Coding of 525 line, Monochrome Television at 1.5 Mbits/s," IEEE Trans. on Communications, Vol. COM-25, No. 11, pp. 1339-1348, Nov. 1977.

[22] H. Kuroda, N. Mukawa, T. Matsuoka, and S. Okudo, "1.5 Mbit/s Interframe Codec for Video Teleconferencing Signals," 1982 IEEE Globecon, pp. E2.5.1-E2.5.5, Nov. 1982.

[23] C. Cafforio and F. Rocca, "The Differential Method for Motion Estimation," NATO ASI Series, Image Sequence Processing and Dynamic Scene Analysis, (T.S. Huang, Ed.), Vol. 2, pp. 104-124, Springer-Verlag, 1983.

[24] C. Cafforio, "Remarks on the differential method for

the estimation of movement in television images," Signal Processing, Vol. 4, pp. 45-52, 1982.

[25] J. A. Stuller, A. N. Netravali, and J. D. Robbins, "Interframe Television Coding Using Gain and Displacement Compensation," BSTJ, Vol. 59, No. 7, pp. 1227-1240, Sept. 1980.