

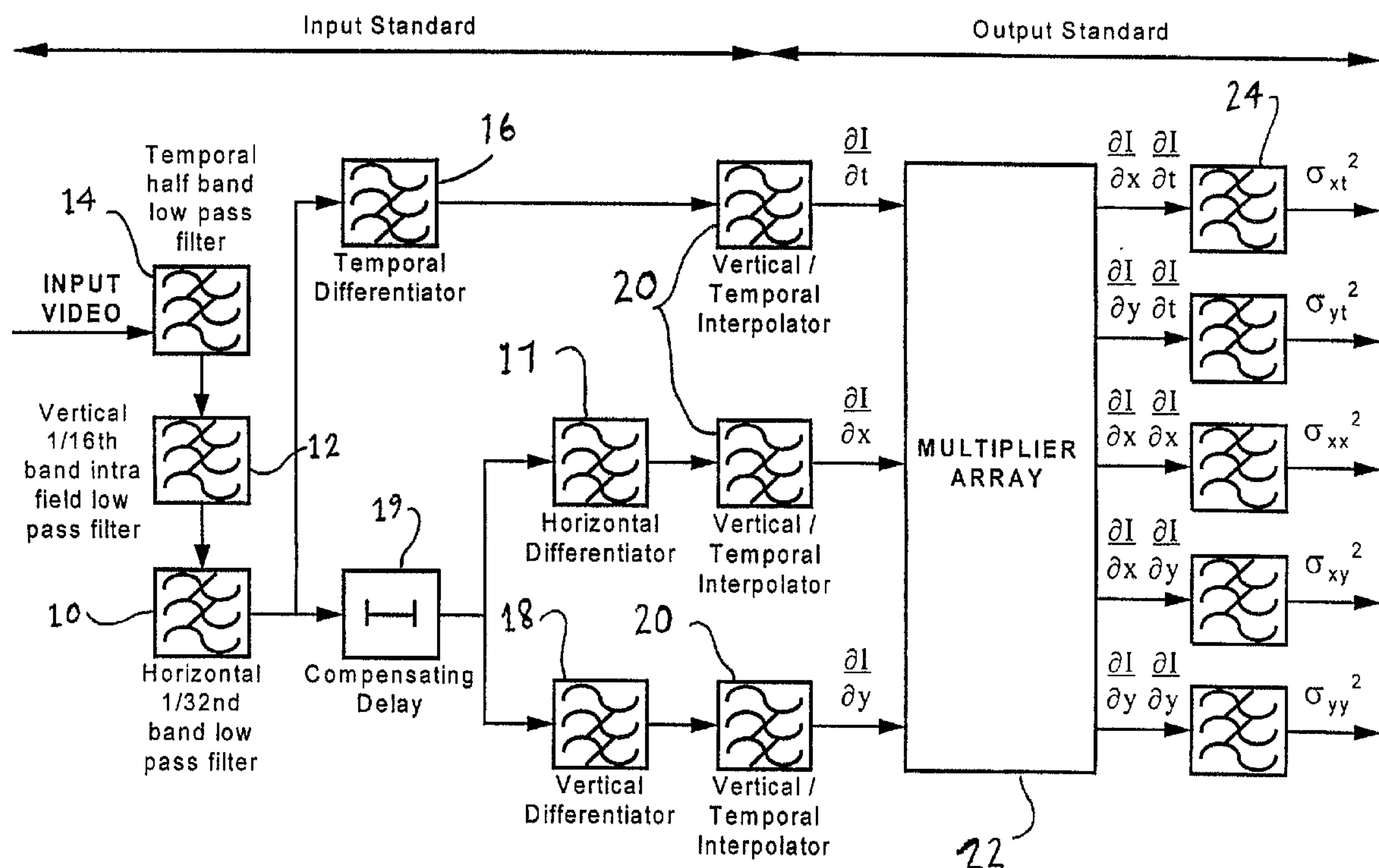


(86) Date de dépôt PCT/PCT Filing Date: 1997/03/03
(87) Date publication PCT/PCT Publication Date: 1997/09/18
(45) Date de délivrance/Issue Date: 2003/05/06
(85) Entrée phase nationale/National Entry: 1998/09/02
(86) N° demande PCT/PCT Application No.: EP 1997/001067
(87) N° publication PCT/PCT Publication No.: 1997/034417
(30) Priorité/Priority: 1996/03/13 (9605317.8) GB

(51) Cl.Int.⁶/Int.Cl.⁶ H04N 7/28, G06T 7/20
(72) Inventeur/Inventor:
BORER, TIMOTHY JOHN, GB
(73) Propriétaire/Owner:
LEITCH EUROPE LIMITED, GB
(74) Agent: DIMOCK STRATTON CLARIZIO LLP

(54) Titre : ESTIMATION DE MOUVEMENT EN FONCTION DU GRADIENT

(54) Title: GRADIENT BASED MOTION ESTIMATION



(57) Abrégé/Abstract:

A technique for generating motion vectors for applications requiring field or frame rate interpolation and especially in standards conversion. The image gradients are calculated on the same standard as the input video or film signal and then vertical/temporal interpolators (20) are used to convert to the output standard before determining the motion vectors. This allows motion vectors to be easily calculated on the output standard.

GRADIENT BASED MOTION ESTIMATION

The invention relates to a technique for estimating motion vectors for a video sequence and, in particular, to a motion estimator for use with a standards converter. The technique has application in any apparatus requiring field or frame rate interpolation, for example, slow motion display apparatus and conversion of film sequences to interlaced video sequences.

Gradient motion estimation is one of three or four fundamental motion estimation techniques and is well known in the literature (references 1 to 18). More correctly called 'constraint equation based motion estimation' it is based on a partial differential equation which relates the spatial and temporal image gradients to motion.

Gradient motion estimation is based on the constraint equation relating the image gradients to motion. The constraint equation is a direct consequence of motion in an image. Given an object, 'object(x, y)', which moves with a velocity (u, v) then the resulting moving image, I(x, y, t) is defined by Equation 1;

$$I(x, y, t) = \text{object}(x - ut, y - vt)$$

This leads directly to the constraint equation, Equation 2;

$$u \cdot \frac{\partial I(x, y, t)}{\partial x} + v \cdot \frac{\partial I(x, y, t)}{\partial y} + \frac{\partial I(x, y, t)}{\partial t} = \frac{\partial \text{object}(x, y)}{\partial t} = 0$$

where, provided the moving object does not change with time (perhaps due to changing lighting or distortion) then $\text{object}/t=0$. This equation is, perhaps, more easily understood by considering an example. Assume that vertical motion is zero, the horizontal gradient is +2 grey levels per pixel and the temporal gradient is -10 grey levels per field. Then the constraint equation says that the ratio of horizontal and temporal gradients implies a motion of 5 pixels/field. The relationship between spatial and temporal gradients is summarised by the constraint equation.

To use the constraint equation for motion estimation it is first necessary to estimate the image gradients; the spatial and temporal gradients of brightness. In principle these are easily calculated by applying straightforward linear horizontal, vertical and temporal filters to the image sequence. In practice, in the absence of additional processing, this can only really be done for the horizontal gradient. For the vertical gradient, calculation of the brightness gradient is confused by interlace which is typically used for television pictures; pseudo-interlaced signals from film do not suffer from this problem. Interlaced signals only contain alternate picture lines on each field. Effectively this is vertical sub-sampling resulting in vertical aliasing which confuses the vertical gradient estimate. Temporally the situation is even worse, if an object has moved by more than 1 pixel in consecutive fields, pixels in the same spatial location may be totally unrelated. This would render any gradient estimate meaningless. This is why gradient motion estimation cannot, in general, measure velocities greater than 1 pixel per field period (reference 8).

Prefiltering can be applied to the image sequence to avoid the problem of direct measurement of the image gradients. If spatial low pass filtering is applied to the sequence then the effective size of 'pixels' is increased. The brightness gradients at a particular spatial location are then related for a wider range of motion speeds. Hence spatial low pass filtering allows higher velocities to be measured, the highest measurable velocity being determined by the degree of filtering applied. Vertical low pass filtering also alleviates the problem of vertical aliasing caused by interlace. Alias components in the image tend to be more prevalent at higher frequencies. Hence, on average, low pass filtering disproportionately removes alias rather than true signal components. The more vertical filtering that is applied the less is the effect of aliasing. There are, however, some signals in which aliasing extends down to zero frequency. Filtering cannot remove all the aliasing from these

signals which will therefore result in erroneous vertical gradient estimates and, therefore, incorrect estimates of the motion vector.

Prefiltering an image sequence results in blurring. Hence small details in the image become lost. This has two consequences, firstly the velocity estimate becomes less accurate since there is less detail in the picture and secondly small objects cannot be seen in the prefiltered signal. To improve vector accuracy hierarchical techniques are sometimes used. This involves first calculating an initial, low accuracy, motion vector using heavy prefiltering, then refining this estimate to higher accuracy using less prefiltering. This does, indeed, improve vector accuracy but it does not overcome the other disadvantage of prefiltering, that is, that small objects cannot be seen in the prefiltered signal, hence their velocity cannot be measured. No amount of subsequent vector refinement, using hierarchical techniques, will recover the motion of small objects if they are not measured in the first stage. Prefiltering is only advisable in gradient motion estimation when it is only intended to provide low accuracy motion vectors of large objects.

Once the image gradients have been estimated the constraint equation is used to calculate the corresponding motion vector. Each pixel in the image gives rise to a separate linear equation relating the horizontal and vertical components of the motion vector and the image gradients. The image gradients for a single pixel do not provide enough information to determine the motion vector for that pixel. The gradients for at least two pixels are required. In order to minimise errors in estimating the motion vector it is better to use more than two pixels and find the vector which best fits the data from multiple pixels. Consider taking gradients from 3 pixels. Each pixel restricts the motion vector to a line in velocity space. With two pixels a single, unique, motion vector is determined by the intersection of the 2 lines. With 3 pixels there are 3 lines and, possibly, no unique solution. This is illustrated in figure 1. The vectors E_1 to E_3 are the error from the best fitting vector to the constraint line for each pixel.

One way to calculate the best fit motion vector for a group of neighbouring pixels is to use a least mean square method, that is minimising the sum of the squares of the lengths of the error vectors (E_1 to E_3 figure 1). The least mean square solution for a group of neighbouring pixels is given by the solution of Equation 3;

$$\begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = - \begin{bmatrix} \sigma_{xi}^2 \\ \sigma_{yi}^2 \end{bmatrix}$$

$$\text{where } \sigma_{xx}^2 = \sum \frac{\partial I}{\partial x} \cdot \frac{\partial I}{\partial x}, \quad \sigma_{xy}^2 = \sum \frac{\partial I}{\partial x} \cdot \frac{\partial I}{\partial y} \quad \text{etc}$$

where (u_0, v_0) is the best fit motion vector and the summations are over a suitable region. The (direct) solution of equation 3 is given by Equation 4;

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \frac{1}{\sigma_{xx}^2 \sigma_{yy}^2 - \sigma_{xy}^4} \begin{bmatrix} \sigma_{xy}^2 \sigma_{yi}^2 - \sigma_{yy}^2 \sigma_{xi}^2 \\ \sigma_{xy}^2 \sigma_{xi}^2 - \sigma_{xx}^2 \sigma_{yi}^2 \end{bmatrix}$$

Small regions produce detailed vector fields of low accuracy and vice versa for large regions. There is little point in choosing a region which is smaller than the size of the prefilter since the pixels within such a small region are not independent.

Typically, motion estimators generate motion vectors on the same standard as the input image sequence. For motion compensated standards converters, or other systems performing motion compensated temporal interpolation, it is desirable to generate motion vectors on the output image sequence standard. For example when converting between European and American television standards the input image sequence is 625 line 50Hz (interlaced) and the output standard is 525 line 60Hz (interlaced). A motion compensated standards converter operating on a European input is required to produce motion vectors on the American output television standard.

It is an object of the present invention to provide a method and apparatus capable of generating motion vectors on an output standard different from the input standard. This is achieved by first calculating image gradients on the input standard and then converting

these gradients to the output standard before implementing the rest of the motion estimation process.

The direct implementation of gradient motion estimation, discussed herein in relation to figures 2 and 3, can give wildly erroneous results. Such behaviour is extremely undesirable. These problems occur when there is insufficient information in a region of an image to make an accurate velocity estimate. This would typically arise when the analysis region contained no detail at all or only the edge of an object. In such circumstances it is either not possible to measure velocity or only possible to measure velocity normal to the edge. It is attempting to estimate the complete motion vector, when insufficient information is available, which causes problems. Numerically the problem is caused by the 2 terms in the denominator of equation 4 becoming very similar resulting in a numerically unstable solution for equation 3.

A solution to this problem of gradient motion estimation has been suggested by Martinez (references 11 and 12). The matrix in equation 3 (henceforth denoted 'M') may be analysed in terms of its eigenvectors and eigenvalues. There are 2 eigenvectors, one of which points parallel to the predominant edge in the analysis region and the other points normal to that edge. Each eigenvector has an associated eigenvalue which indicates how sharp the image is in the direction of the eigenvector. The eigenvectors and values are defined by Equation 5;

$$M \cdot e_i = \lambda_i e_i \quad i \in \{1, 2\}$$

$$\text{where; } M = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix}$$

The eigenvectors e_i are conventionally defined as having length 1, which convention is adhered to herein.

In plain areas of the image the eigenvectors have essentially random direction (there are no edges) and both eigenvalues are very small (there is no detail). In these circumstances the only sensible vector to assume is zero. In parts of the image which contain only an edge feature the eigenvectors point normal to the edge and parallel to the edge. The eigenvalue corresponding to the normal eigenvector is

(relatively) large and the other eigenvalue small. In this circumstance only the motion vector normal to the edge can be measured. In other circumstances, in detailed parts of the image where more information is available, the motion vector may be calculated using Equation 4.

The motion vector may be found, taking into account Martinez' ideas above, by using Equation 6;

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = - \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} \mathbf{e}_1 \mathbf{e}_1^t + \frac{\lambda_2}{\lambda_2^2 + n_2^2} \mathbf{e}_2 \mathbf{e}_2^t \right) \cdot \begin{bmatrix} \sigma_x^2 \\ \sigma_y^2 \end{bmatrix}$$

where superscript t represents the transpose operation. Here n_1 & n_2 are the computational or signal noise involved in calculating λ_1 & λ_2 respectively. In practice $n_1 \approx n_2$, both being determined by, and approximately equal to, the noise in the coefficients of M. When λ_1 & $\lambda_2 \ll n$ then the calculated motion vector is zero; as is appropriate for a plain region of the image. When $\lambda_1 \gg n$ and $\lambda_2 \ll n$ then the calculated motion vector is normal to the predominant edge in that part of the image. Finally if $\lambda_1, \lambda_2 \gg n$ then equation 6 becomes equivalent to equation 4. As signal noise, and hence n , decreases then equation 6 provides an increasingly more accurate estimate of the motion vectors as would be expected intuitively.

In practice calculating motion vectors using the Martinez technique involves replacing the apparatus of figure 3, below, with more complex circuitry. The direct solution of equation 6 would involve daunting computational and hardware complexity. It can, however, be implemented using only two-input, pre-calculated, look up tables and simple arithmetic operations. It is another object of the present invention to provide a streamlined implementation of the Martinez technique.

The invention provides motion vector estimation apparatus for use in video signal processing comprising means for calculating image gradients for each input sampling site of a picture sequence, the image gradients being calculated on the same

standard as the input signal, means for converting the image gradients from the first standard to a second, output standard, and means for generating a plurality of motion vectors from the image gradients, the apparatus being arranged to convert the image gradients from the input standard to the output standard before calculation of motion vectors thereby producing motion vectors on the desired output standard. The motion vectors are calculated on the output standard thereby avoiding the difficulties and inaccuracies involved in converting the signals to the output standard after calculation of the motion vectors.

The apparatus may comprise temporal and spatial low pass filters for prefiltering the input video signal. Prefiltering increases the maximum motion speed which can be measured and reduces the deleterious effects of vertical/temporal aliasing.

The means for calculating the image gradients may comprise temporal and spatial (horizontal and vertical) differentiators.

The means for converting the image gradients from the input standard to the output standard comprise vertical/temporal interpolators. For example a linear (polyphase) interpolator such as a bilinear interpolator.

The image gradients corresponding to a plurality of output sampling sites are used to calculate the motion vectors. The motion vectors may be calculated using a least mean square solution for a group of neighbouring output sampling sites.

In an embodiment the apparatus further comprises a multiplier array having as its inputs the image gradients previously calculated and converted to the output standard, and corresponding low pass filters for summing the image gradient products. The means for calculating the motion vectors utilises the sums of the image gradient products corresponding to a group of neighbouring output sampling sites to produce the best fit motion vector for the group of sampling sites. A different group of neighbouring sampling sites may be used to calculate each motion vector. The means for calculating

motion vectors, determines the best fit motion vector given by equation 4 or equation 6 as herein defined.

In an alternative embodiment the apparatus comprises rectangular to polar coordinate converter means having the spatial image gradients converted to the output standard as its inputs and the motion vectors are determined for a group of output sampling sites based on the angle and magnitude of the image gradients of each sampling site in said group. The motion vectors being calculated on the basis of equation 11 or 13 as herein defined.

The invention also provides a method of motion estimation in video or film signal processing comprising calculating image gradients for each input sampling site of a picture sequence, the image gradients being calculated on a first, input standard, generating a plurality of motion vectors from the image gradients, the image gradients being converted to a second, output standard before generating the motion vectors thereby generating motion vectors on the desired output standard.

The method may comprise a prefiltering step. The input video signal may be prefiltered for example using temporal and spatial lowpass filters.

The image gradients corresponding to a plurality of output sampling sites are used to calculate the motion vectors. The motion vectors may be calculated using a least mean square solution for a group of neighbouring sampling sites.

The step of generating motion vectors may comprise using the sums of the image gradient products corresponding to a group of neighbouring output sampling sites to produce the best fit motion vector for each said group. The motion vectors may be calculated using equation 4 or 6 as defined herein.

In an embodiment the step of generating motion vectors may comprise performing eigen-analyses on the sums of the image gradient products using the spatial image gradients converted to the output standard and assigning two eigenvectors and eigenvalues to each

output sampling site. The motion vector for each group of sampling sites is calculated by applying equation 6, as herein defined, to the results of the eigen analyses.

In another embodiment the step of generating motion vectors comprises transforming the spatial image gradient vectors on the output standard from rectangular to polar coordinates and the motion vectors are determined for a group of output sampling sites based on the angle and magnitude of the image gradients of each sampling site in said group. The motion vectors being calculated on the basis of equation 11 or 13 as herein defined.

The invention will now be described in more detail with reference to the accompanying drawings in which:

Figure 1 shows graphically the image gradient constraint lines for three pixels.

Figures 2 and 3 are a block diagram of a motion estimator according to an embodiment of the invention.

Figure 4 is a block diagram of apparatus for calculating motion vectors which can be substituted for the apparatus of fig. 3.

Figure 5 is a block diagram of apparatus for implementing the eigen analysis required in figure 4.

Figures 6 and 7 show another embodiment of the gradient motion estimation apparatus according to the invention.

Figure 8 shows graphically the distribution of errors in the case of a best fit motion vector.

Figures 9 and 10 are block diagrams of apparatus capable of providing an indication of the error of motion vectors in a motion estimation system.

A block diagram of a direct implementation of gradient motion estimation is shown in figures 2 & 3.

The apparatus shown schematically in Figure 2 performs filtering and calculation of gradient products and their summations. The apparatus of Figure 3 generates motion vectors from the sums of gradient products produced by the apparatus of figure 2. The horizontal and vertical low pass filters (10,12) in figure 2 perform

spatial prefiltering as discussed above. The cut-off frequencies of 1/32nd band horizontally and 1/16th band vertically allow motion speeds up to (at least) 32 pixels per field to be measured. Different cut-off frequencies could be used if a different range of speeds is required. The image gradients are calculated by three temporal and spatial differentiators (16,17,18).

The vertical/temporal interpolation filters (20) convert the image gradients, measured on the input standard, to the output standard. Typically the vertical/temporal interpolators (20) are bilinear interpolators or other polyphase linear interpolators. Thus the output motion vectors are also on the output standard. The interpolation filters are a novel feature which facilitates interfacing the motion estimator to a motion compensated temporal interpolator. Temporal low pass filtering is normally performed as part of (all 3 of) the interpolation filters. The temporal filter (14) has been re-positioned in the processing path so that only one rather than three filters are required. Note that the filters prior to the multiplier array can be implemented in any order because they are linear filters. The summation of gradient products, specified in equation 3, are implemented by the low pass filters (24) following the multiplier array (22). Typically these filters would be (spatial) running average filters, which give equal weight to each tap within their region of support. Other lowpass filters could also be used at the expense of more complex hardware. The size of these filters (24) determines the size of the neighbourhood used to calculate the best fitting motion vector. Examples of filter coefficients which may be used can be found in the example.

A block diagram of apparatus capable of implementing equation 6 and which replaces that of figure 3, is shown in figures 4 and 5.

Each of the 'eigen analysis' blocks (30), in figure 4, performs the analysis for one of the two eigenvectors. The output of the eigen-analysis is a vector (with x and y components) equal to

$$s_i = e_i \cdot \sqrt{\lambda_i / \lambda_i^2 + n^2}$$

These 's' vectors are combined with vector $(\sigma_{xt}^2, \sigma_{yt}^2)$ (denoted c in figure 4), according to equation 6, to give the motion vector

according to the Martinez technique.

The eigen analysis, illustrated in figure 5, has been carefully structured so that it can be implemented using lookup tables with no more than 2 inputs. This has been done since lookup tables with 3 or more inputs would be impracticably large using today's technology. The implementation of figure 5 is based on first normalising the matrix M by dividing all its elements by $(\sigma_{xx}^2 + \sigma_{yy}^2)$. This yields a new matrix, N, with the same eigenvectors (e_1 & e_2) and different (but related) eigenvalues (X_1 & X_2). The relationship between M, N and their eigenvectors and values is given by Equation 7;

$$N = \frac{1}{\sigma_{xx}^2 + \sigma_{yy}^2} M = \begin{bmatrix} \frac{\sigma_{xx}^2}{\sigma_{xx}^2 + \sigma_{yy}^2} & \frac{\sigma_{xy}^2}{\sigma_{xx}^2 + \sigma_{yy}^2} \\ \frac{\sigma_{xy}^2}{\sigma_{xx}^2 + \sigma_{yy}^2} & \frac{\sigma_{yy}^2}{\sigma_{xx}^2 + \sigma_{yy}^2} \end{bmatrix}$$

$$M \cdot e_i = \lambda_i \cdot e_i$$

$$N \cdot e_i = \chi_i \cdot e_i$$

$$\lambda_i = (\sigma_{xx}^2 + \sigma_{yy}^2) \chi_i$$

$$n_{\lambda} = (\sigma_{xx}^2 + \sigma_{yy}^2) n_{\chi}$$

Matrix N is simpler than M as it contains only two independent values, since the principle diagonal elements ($N_{1,1}$, $N_{2,2}$) sum to unity and the minor diagonal elements ($N_{1,2}$, $N_{2,1}$) are identical. The principal diagonal elements may be coded as $(\sigma_{xx}^2 - \sigma_{yy}^2) / (\sigma_{xx}^2 + \sigma_{yy}^2)$ since Equation 8;

$$N_{1,1} = \frac{1}{2} \left(1 + \left(\frac{\sigma_{xx}^2 - \sigma_{yy}^2}{\sigma_{xx}^2 + \sigma_{yy}^2} \right) \right)$$

$$N_{2,2} = \frac{1}{2} \left(1 - \left(\frac{\sigma_{xx}^2 - \sigma_{yy}^2}{\sigma_{xx}^2 + \sigma_{yy}^2} \right) \right)$$

Hence lookup tables 1 & 2 have all the information they require to find the eigenvalues and vectors of N using standard techniques. It is therefore straightforward to precalculate the contents of these lookup tables. Lookup table 3 simply implements the square root function. The key features of the apparatus shown in figure 5 are

that the eigen analysis is performed on the normalised matrix, N , using 2 input lookup tables (1 & 2) and the eigenvalue analysis (from table 2) is rescaled to the correct value using the output of table 3.

The gradient motion estimator described above is undesirably complex. The motion estimator is robust to images containing limited information but figures 4 and 5 show the considerable complexity involved. The situation is made worse by the fact that many of the signals have a very wide dynamic range making the functional blocks illustrated much more difficult to implement.

A technique which yields considerable simplifications without sacrificing performance. This is based on normalising the basic constraint equation (equation 2) to control the dynamic range of the signals. As well as reducing dynamic range this also makes other simplifications possible.

Dividing the constraint equation by the modulus of the gradient vector yields a normalised constraint equation i.e. Equation 9:

$$\frac{u \frac{\partial I}{\partial x} + v \frac{\partial I}{\partial y}}{|\nabla I|} = - \frac{\frac{\partial I}{\partial t}}{|\nabla I|}$$

$$\text{where: } \nabla I = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \quad \& \quad |\nabla I| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

The significance of this normalisation step becomes more apparent if equation 9 is rewritten as Equation 10;

$$u \cdot \cos(\theta) + v \cdot \sin(\theta) = vn$$

$$\text{where: } \cos(\theta) = \frac{\frac{\partial I}{\partial x}}{|\nabla I|}, \quad \sin(\theta) = \frac{\frac{\partial I}{\partial y}}{|\nabla I|}; \quad vn = - \frac{\frac{\partial I}{\partial t}}{|\nabla I|}$$

in which θ is the angle between the spatial image gradient vector (∇I) and the horizontal; vn is the motion speed in the direction of the image gradient vector, that is, normal to the predominant edge in the picture at that point. This seems a much more intuitive equation relating, as it does, the motion

vector to the image gradient and the motion speed in the direction of the image gradient. The coefficients of equation 10 ($\cos(\theta)$ & $\sin(\theta)$) have a well defined range (0 to 1) and, approximately the same dynamic range as the input signal (typically 8 bits). Similarly vn has a maximum (sensible) value determined by the desired motion vector measurement range. Values of vn greater than the maximum measurement range, which could result from either noise or 'cuts' in the input picture sequence, can reasonably be clipped to the maximum sensible motion speed.

The normalised constraint equation 10 can be solved to find the motion vector in the same way as the unnormalised constraint equation 2. With normalisation equation 3 becomes Equation 11;

$$\begin{bmatrix} \sum \cos^2(\theta) & \sum \cos(\theta).\sin(\theta) \\ \sum \cos(\theta).\sin(\theta) & \sum \sin^2(\theta) \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \begin{bmatrix} \sum vn.\cos(\theta) \\ \sum vn.\sin(\theta) \end{bmatrix}$$

or: $\Phi \cdot \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \psi$

In fact matrix (Φ) has only 2 independent elements, since $\cos^2(x) + \sin^2(x) = 1$. This is more clearly seen by rewriting $\cos^2(x)$ and $\sin^2(x)$ as $\frac{1}{2}(1 + \cos(2x))$ hence equation 11 becomes Equation 12;

$$\frac{1}{2} \left(N.I + \begin{bmatrix} \sum \cos(2\theta) & \sum \sin(2\theta) \\ \sum \sin(2\theta) & -\sum \cos(2\theta) \end{bmatrix} \right) \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \begin{bmatrix} \sum vn.\cos(\theta) \\ \sum vn.\sin(\theta) \end{bmatrix}$$

where I is the (2x2) identity matrix and N is the number of pixels included in the summations. Again the motion vector can be found using equation 13:

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} \mathbf{e}_1 \mathbf{e}_1^t + \frac{\lambda_2}{\lambda_2^2 + n_2^2} \mathbf{e}_2 \mathbf{e}_2^t \right) \begin{bmatrix} \sum vn.\cos(\theta) \\ \sum vn.\sin(\theta) \end{bmatrix}$$

where now $\mathbf{e}$ and λ are the eigenvectors and eigenvalues of Φ rather than M . Now, because Φ only has two independent elements, the eigen-analysis can now be performed using only three, two-input, lookup tables, furthermore the dynamic range of the elements of Φ

(equation 11) is much less than the elements of M thereby greatly simplifying the hardware complexity.

A block diagram of a gradient motion estimator using Martinez technique and based on the normalised constraint equation is shown in figures 6 & 7.

The apparatus of figure 6 performs the calculation of the normalised constraint equation (equation 10) for each pixel or data value. Obviously, if prefiltering is performed the number of independent pixel values is reduced, the effective pixel size is greater. The filtering in figure 6 is identical to that in figure 2. The spatial image gradients converted to the output standard are used as inputs for a rectangular to polar coordinate converter (32) which calculates the magnitude of the spatial image vector and the angle θ . A suitable converter can be obtained from Raytheon (Coordinate transformer, model TMC 2330). A lookup table (34) is used to avoid division by very small numbers when there is no detail in a region of the input image. The constant term, ' n ', used in the lookup table is the measurement noise in estimating $|\nabla I|$, which depends on the input signal to noise ratio and the prefiltering used. A limiter (36) has also been introduced to restrict the normal velocity, v_n , to its expected range (determined by the spatial prefilter). The normal velocity might, otherwise, exceed its expected range when the constraint equation is violated, for example at picture cuts. A key feature of figure 6 is that, due to the normalisation that has been performed, the two outputs, v_n & θ , have a much smaller dynamic range than the three image gradients in figure 2, thereby allowing a reduction in the hardware complexity.

In the apparatus of figure 6 the input video is first filtered using separate temporal, vertical and horizontal filters (10,12,14), the image gradients are calculated using three differentiating filters (16,18) and then converted, from the input lattice, to the output sampling lattice using three vertical/temporal interpolators (20), typically bilinear or other polyphase linear filters. For example, with a 625/50/2:1 input the image gradients are calculated on a 525/60/2:1 lattice.

The parameters of the normalised constraint equation, v_n & θ , are calculated as shown.

The apparatus of figure 7 calculates the best fitting motion vector, corresponding to a region of the input image, from the constraint equations for the pixels in that region. The summations specified in equation 12 are implemented by the lowpass filters (38) following the polar to rectangular coordinate converter (40) and lookuptables 1 & 2. Typically these filters (38) would be (spatial) running average filters, which give equal weight to each tap within their region of support. Other lowpass filters could also be used at the expense of more complex hardware. The size of these filters (38) determine the size of the neighbourhood used to calculate the best fitting motion vector. Lookup tables 1 & 2 are simply cosine and sine lookup tables. Lookup tables 3 to 5 contain precalculated values of matrix 'Z' defined by Equation 14;

$$Z = \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} e_1 e_1^t + \frac{\lambda_2}{\lambda_2^2 + n_2^2} e_2 e_2^t \right)$$

where e and λ are the eigenvectors and eigenvalues of Φ . Alternatively Z could be Φ^{-1} (ie. assuming no noise), but this would not apply the Martinez technique and would give inferior results. A key feature of figure 7 is that the elements of matrix Z are derived using 2 input lookuptables. Their inputs are the output from the two lowpass filters (39) which have a small dynamic range allowing the use of small lookup tables.

The implementations of the gradient motion techniques discussed above seek to find the 'best' motion vector for a region of the input picture. However it is only appropriate to use this motion vector, for motion compensated processing, if it is reasonably accurate. Whilst the determined motion vector is the 'best fit' this does not necessarily imply that it is also an accurate vector. The use of inaccurate motion vectors, in performing motion compensated temporal interpolation, results in objectionable impairments to the interpolated image. To avoid these impairments it is desirable to

revert to a non-motion compensated interpolation algorithm when the motion vector cannot be measured accurately. To do this it is necessary to know the accuracy of the estimated motion vectors. If a measure of vector accuracy is available then the interpolation method can be varied between 'full motion compensation' and no motion compensation depending on vector accuracy, a technique known as 'graceful fallback' described in references 4 & 16.

A technique for measuring the accuracy of motion vectors is based on the use of the constraint equation and hence is particularly suitable for use with gradient based motion estimation techniques as described above. The method, however, is more general than this and could also be used to estimate the accuracy of motion vectors measured in other ways. The measurement of the accuracy of motion vectors is a new technique. Most of the literature on motion estimation concentrates almost wholly on ways of determining the 'best' motion vector and pays scant regard to considering whether the resulting motion vectors are actually accurate. This may, in part, explain why motion compensated processing is, typically, unreliable for certain types of input image.

Once a motion vector has been estimated for a region of an image an error may be calculated for each pixel within that region. That error is an indication of how accurately the motion vector satisfies the constraint equation or the normalised constraint equation (equations 2 and 10 above respectively). The following discussion will use the normalised constraint equation as this seems a more objective choice but the unnormalised constraint equation could also be used with minor changes (the use of the unnormalised constraint equation amounts to giving greater prominence to pixels with larger image gradients). For the i^{th} pixel within the analysis region the error is given by Equation 15;

$$\text{error}_i = vn_i - u_0 \cos(\theta_i) - v_0 \sin(\theta_i)$$

(for all i when $1 \leq i \leq N$, where N is the number of pixels in the analysis region).

This error corresponds to the distance of the 'best' motion vector, (u_0, v_0) , from the constraint line for that pixel (see figure 1). Note that equation 11 above gives a motion vector which minimises the sum of the squares of these errors. Each error value is associated with the direction of the image gradient for that pixel. Hence the errors are better described as an error vector, E_i , illustrated in figure 1 and defined by Equation 16;

$$E_i^t = \text{error}_i \cdot [\cos(\theta), \sin(\theta)]$$

where superscript t represents the transpose operation.

The set of error vectors, $\{E_i\}$, form a two dimensional distribution of errors in motion vector space, illustrated in figure 8 below. This distribution of motion vector measurement errors would be expected to be a two dimensional Gaussian (or Normal) distribution. Conceptually the distribution occupies an elliptical region around the true motion vector. The ellipse defines the area in which most of the estimates of the motion vector would lie; the 'best' motion vector points to the centre of the ellipse. Figure 8 illustrates the 'best' motion vector, (u_0, v_0) , and 4 typical error vectors, E_1 to E_4 . The distribution of motion vector measurement errors is characterised by the orientation and length of the major and minor axes (σ_1, σ_2) of the ellipse. To calculate the characteristics of this distribution we must first form the $(N \times 2)$ matrix defined as Equation 17;

$$E = \begin{bmatrix} E'_1 \\ E'_2 \\ \vdots \\ E'_N \end{bmatrix} = \begin{bmatrix} \text{error}_1 \cdot \cos(\theta_1) & \text{error}_1 \cdot \sin(\theta_1) \\ \text{error}_2 \cdot \cos(\theta_2) & \text{error}_2 \cdot \sin(\theta_2) \\ \vdots & \vdots \\ \text{error}_N \cdot \cos(\theta_N) & \text{error}_N \cdot \sin(\theta_N) \end{bmatrix}$$

The length and orientation of the axes of the error distribution are given by eigenvector analysis of $E^t.E$; the eigenvectors point along the axes of the distribution and the eigenvalues, $N_{total}.\sigma_1^2$ & $N_{total}.\sigma_2^2$ (where N_{total} is the total number of pixels in the region used to estimate the errors), give their length (see figure 8) that is Equation 18;

$$(E^t.E).e_i = N_{total}.\sigma_i.e_i \quad \text{where } i=1 \text{ or } 2$$

The matrix $E^t.E$ (henceforth the 'error matrix' and denoted Q for brevity) can be expanded to give Equation 19;

$$E^t.E = \begin{bmatrix} \sum error^2.\cos^2(\theta) & \sum error^2.\cos(\theta).\sin(\theta) \\ \sum error^2.\cos(\theta).\sin(\theta) & \sum error^2.\sin^2(\theta) \end{bmatrix}$$

where the summation is over a region of the image.

The likely motion vector error depends on how the motion vector was measured. If the motion vector was calculated using, for example, block matching then the likely error would be approximately as determined by the above analysis. However it is quite likely that this error estimation technique of the invention would be applied to motion vectors calculated using gradient (constraint equation) based motion estimation. In this latter case the motion vector is, itself, effectively the 'average' of many measurements (i.e. 1 measurement per constraint equation used). Hence the error in the gradient based motion vector is less than the error estimated from the 'error matrix' above. This is an example of the well known effect of taking an average of many measurements to improve the accuracy. If larger picture regions are used for gradient motion estimation then more accurate motion vectors are obtained (at the expense, of course, of being unable to resolve small objects). By contrast taking larger regions in a block matching motion estimator does not necessarily increase the vector accuracy (assuming the selected vector is correct), it does however reduce the chance of measuring a 'spurious' vector.

The likely error in the motion vector may be less than the

'size' of the distribution of error vectors. The reduction is specified by a parameter $N_{\text{effective}}$ which depends on how the motion vector was measured. For block matching, $N_{\text{effective}}$ would be approximately 1. For gradient motion estimation $N_{\text{effective}}$ might be as high as the number of pixels used in the measurement. It is more likely, however, that $N_{\text{effective}}$ is less than the number of pixels due to the effects of prefiltering the video prior to motion estimation. Prefiltering effectively 'enlarges' the pixels (i.e. individual pixels are not longer independent) reducing the effective number of pixels ($N_{\text{effective}}$). Typically the region of the image used both to calculate the motion vector and estimate its error might be 3 times the 'size' (both horizontally and vertically) of the prefilter used. This would give a typical value for $N_{\text{effective}}$ of 3^2 . For a given value of $N_{\text{effective}}$ the size of the error distribution, calculated above, must be reduced by the square root of $N_{\text{effective}}$. This is the well known result for the reduction in error due to averaging $N_{\text{effective}}$ measurements. Thus, for a typical gradient based motion estimator in which $N_{\text{effective}}$ is 9, the likely error in the measured motion vector is 3 times less than the distribution of vector errors calculated above.

In an embodiment, the averaging filter is 95 pixels by 47 field lines so, the total number (N_{total} in figure 10) of pixels is 4465. The effective number of pixels ($N_{\text{effective}}$) used in error estimation will be less than the total number of pixels if prefiltering is performed. In the specification of the gradient motion estimator parameters in the example, the spatial pre-filter is $1/16^{\text{th}}$ band vertical intra-field and $1/32^{\text{nd}}$ band horizontal. The error estimation region is 3 times the effective size of the spatial pre-filters both horizontally and vertically, giving an effective number of pixels used in the selected error estimation region of 9.

To calculate the distribution of motion vector measurement errors it is necessary to first calculate the elements of the error matrix, according to equation 19, then calculate its eigenvectors and eigenvalues. The elements of the error matrix may be calculated by

the apparatus of figure 9. Other implementations are possible, but figure 9 is straight forward and efficient. The inputs to figure 9, θ and v_n , may be derived as in figure 6. The motion vector input to figure 9, (u, v) , could be derived as in figure 7, however it could equally well come from any other source such as figure 3 or 4 or even a block matching motion estimator. The lookup tables (1 and 2) are simply cosine and sine tables and, as in figures 2 & 7, the required summations are performed using spatial lowpass filters (42) such as running average filters.

Once the error matrix has been calculated (e.g. as in figure 9) its eigenvalues and eigenvectors may be found using the implementation of figure 10 whose inputs are the elements of the error matrix, i.e. $\Sigma(\text{error}^2 \cdot \cos^2(\theta))$, $\Sigma(\text{error}^2 \cdot \cos(\theta) \cdot \sin(\theta))$ and $\Sigma(\text{error}^2 \cdot \sin^2(\theta))$, denoted Q_{11} , Q_{12} and Q_{22} respectively. Note that, as in figure 5, since there are two eigenvalues the implementation of figure 10 must be duplicated to generate both eigenvectors. As in figure 5, described previously, the implementation of figure 10 has been carefully structured so that it uses look up tables with no more than 2 inputs. In figure 10 the output of lookup table 1 is the angular orientation of an eigenvector, that is the orientation of one of the principle axes of the (2 dimensional) error distribution. The output of lookup table 2, once it has been rescaled by the output of lookup table 3, is inversely proportional to the corresponding eigenvalue. An alternative function of the eigenvalue (other than its inverse) may be used depending on the application of the motion vector error information.

The spread vector outputs of figure 10 (i.e. (Sx_i, Sy_i) $i=1, 2$) describe the likely motion vector measurement error for each motion vector in two dimensions. Since a video motion vector is a (2 dimensional) vector quantity, two vectors are required to describe the measurement error. In this implementation the spread vectors point along the principle axes of the distribution of vector measurement errors and their magnitude is the inverse of the standard deviation of measurement error along these axes. If we assume, for

example, that the measurement errors are distributed as a 2 dimensional Gaussian distribution, then the probability distribution of the motion vector, v , is given by equation 20;

$$P(v) = (|S_1| \cdot |S_2| / 2\pi) \cdot \exp(-(((v-v_m) \cdot S_1)^2 + ((v-v_m) \cdot S_2)^2))$$

where v_m is the measured motion vector and S_1 and S_2 are the two spread vectors. Of course, the motion vector measurement errors may not have a Gaussian distribution but the spread vectors, defined above, still provide a useful measure of the error distribution. For some applications it may be more convenient to define spread vectors whose magnitude is a different function of the error matrix eigenvalues.

An alternative, simplified, output of figure 10 is a scalar confidence signal rather than the spread vectors. This may be more convenient for some applications. Such a signal may be derived from, r_{error} , the product of the outputs of lookup tables 3 and 4 in figure 10, which provides a scalar indication of the motion vector measurement error.

The confidence signal may then be used to implement graceful fallback in a motion compensated image interpolator as described in reference 4. The r_{error} signal is a scalar, average, measure of motion vector error. It assumes that the error distribution is isotropic and, whilst this may not be justified in some situations, it allows a simple confidence measure to be generated. Note that the scalar vector error, r_{error} , is an objective function, of the video signal, whilst the derived confidence signal is an interpretation of it.

A confidence signal may be generated by assuming that there is a small range of vectors which shall be treated as correct. This predefined range of correct vectors will depend on the application. We may, for example, define motion vectors to be correct if they are within, say, 10% of the true motion vector. Outside the range of correct vectors we shall have decreasing confidence in the motion vector. The range of correct motion vectors is the confidence region

specified by $r_{\text{confident}}$ which might, typically, be defined according to equation 21;

$$r_{\text{confident}} = k \cdot |v| + r_0$$

where k is a small fraction (typically 10%) and r_0 is a small constant (typically 1 pixel/field) and $|v|$ is the measured motion speed. The parameters k and r_0 can be adjusted during testing to achieve best results. Hence the region of confidence is proportional to the measured motion speed except at low speeds when it is a small constant. The confidence value is then calculated, for each output motion vector, as the probability that the actual velocity is within the confidence radius, $r_{\text{confident}}$, of the measured velocity. This may be determined by assuming a Gaussian probability distribution:

$$\text{confidence} = \frac{1}{2\pi r_{\text{error}}^2} \int_0^{r_{\text{confident}}} 2\pi x \cdot \exp\left(-\frac{1}{2} \frac{x^2}{r_{\text{error}}^2}\right) dx$$

giving the following expression for vector confidence (equation 22):

$$\text{confidence} = 1 - \exp(-\frac{1}{2}(r_{\text{confidence}}^2/r_{\text{error}}^2))$$

An embodiment of apparatus for estimating vector error is shown in figures 6, 9 and 10. The apparatus of figure 9 calculates the error matrix using the outputs from the apparatus of figure 6, which were generated previously to estimate the motion vector. The error matrix input in figure , $E^t \cdot E$, is denoted Q to simplify the labelling. The content of lookup tables 1 & 2 in figure 10 are defined by;

Look Up Table 1 = $\text{angle}(2y, -(x \pm \sqrt{(x^2 + 4y^2)}))$

Look Up Table 2 = $1/\sqrt{(2(1 \pm \sqrt{(x^2 + 4y^2)}))}$

$$\text{Where; } x = \frac{Q_{1,1} - Q_{2,2}}{Q_{1,1} + Q_{2,2}} \quad \text{and } y = \frac{Q_{1,2}}{Q_{1,1} + Q_{2,2}}$$

where the 'angle(x, y)' function gives the angle between the x axis and point (x, y) and where the positive sign is taken for one of the

eigenanalysis units and the negative sign is taken for the other unit.

The input of lookup table 3 in figure 10 ($Q_{11} + Q_{22}$) is a dimensioned parameter (z) which describes the scale of the distribution of motion vector errors. The content of lookup table 3 is defined by $\sqrt{(z/N_{\text{total}} \cdot N_{\text{effective}})}$. The output of Lookup table 3 is a scaling factor which can be used to scale the output of lookup table 2 defined above. The input to the polar to rectangular coordinate converter is, therefore, related to the inverse of the length of each principle axis of the error distribution. Using different Lookup table it would be possible to calculate the spread vectors directly in cartesian co-ordinates.

The apparatus described in relation to figure 10, is capable of producing both the spread vectors and the scalar confidence signal. The present invention encompasses methods and apparatus which generate only one such parameter; either the confidence signal or the spread vectors. The eigen analyses performed by the apparatus of figure 10 must be performed twice to give both spread vectors for each principle axis of the error distribution; only one implementation of figure 10 is required to generate r_{error} and the derived confidence signal. The inputs to lookup table 4 are the same as for lookup table 1 (x and y). The content of Lookup table 4 is defined by $\sqrt[4]{(x(1-x^2)-y^2)}$. The output of lookup table 4 scaled by the output of lookup table 3 gives r_{error} a scalar (isotropic) vector error from which a confidence signal is generated in lookup table 5, the contents of which are defined by equation 22, for example. r_{error} is the geometric mean of the length of the major and minor axes of the error distribution, that is, $r_{\text{error}} = \sqrt{(\sigma_1 \cdot \sigma_2)}$.

In figures 7 and 9 picture resizing is allowed for using (intrafield) spatial interpolators (44) following the region averaging filters (38,39,42). Picture resizing is optional and is required for example for overscan and aspect ratio conversion. The apparatus of figure 6 generates its outputs on the nominal output standard, that is assuming no picture resizing. The conversion from

input to (nominal) output standard is achieved using (bilinear) vertical/temporal interpolators(20). Superficially it might appear that these interpolators (20) could also perform the picture stretching or shrinking required for resizing. However, if this were done the region averaging filters (38,42) in figures 7 and 9 would have to vary in size with the resizing factor. This would be very awkward for large picture expansions as very large region averaging filters (38,42) would be required. Picture resizing is therefore achieved after the region averaging filters using purely spatial (intrafield) interpolators (44), for example bilinear interpolators. In fact the function of the vertical/temporal filters (20) in figure 6 is, primarily, to interpolate to the output field rate. The only reason they also change the line rate is to maintain a constant data rate.

Experimental Results

Experiments were performed to simulate the basic motion estimation algorithm (figures 2 & 3), use of the normalised constraint equation (figures 6 & 7), the Martinez technique with the normalised constraint equation and estimation of vector measurement error (figures 9 & 5). In general these experiments confirmed the theory and techniques described above.

Simulations were performed using a synthetic panning sequence. This was done both for convenience and because it allowed a precisely known motion to be generated. Sixteen field long interlaced sequences were generated from an image for different motion speeds. The simulation suggests that the basic gradient motion estimation algorithm gives the correct motion vector with a (standard deviation) measurement error of about $\pm 1/4$ pixel/field. The measured velocity at the edge of the picture generally tends towards zero because the filters used are not wholly contained within the image. Occasionally unrealistically high velocities are generated at the edge of image. The use of the normalised constraint equation gave similar results to

the unnormalised equation. Use of the Martinez technique gave varying results depending on the level of noise assumed. This technique never made things worse and could significantly reduce worst case (and average) errors at the expense of biasing the measured velocity towards zero. The estimates of the motion vector error were consistent with the true (measured) error.

Example:

This example provides a brief specification for a gradient motion estimator for use in a motion compensated standards converter. The input for this gradient motion estimator is interlaced video in either 625/50/2:1 or 525/60/2:1 format. The motion estimator produces motion vectors on one of the two possible input standards and also an indication of the vector's accuracy on the same standard as the output motion vectors. The motion vector range is at least ± 32 pixels/field. The vector accuracy is output as both a 'spread vector' and a 'confidence signal'.

A gradient motion estimator is shown in block diagram form in figures 6 & 7 above. Determination of the measurement error, indicated by 'spread vectors' and 'confidence' are shown in figures 9 & 10. The characteristics of the functional blocks of these block diagrams is as follows:

Input Video:

4:2:2 raster scanned interlaced video.

luminance component only

Active field 720 pixel $\times$ 288 or 244 field lines depending on input standard.

Luminance coding 10 bit, unsigned binary representing the range 0 to $(2^{10}-1)$

Temporal Halfband Lowpass Filter (14):

Function: Temporal filter operating on luminance. Implemented as a vertical/temporal filter because the input is interlaced. The coefficients are defined by the following matrix in which columns represent fields and rows represent picture (not field) lines.

$$\text{Temporal Halfband filter coefficients} = 1/8 \begin{bmatrix} 1 & 0 & 1 \\ 0 & 4 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Input: 10 bit unsigned binary representing the range 0 to 1023(decimal).

Output: 12 bit unsigned binary representing the range 0 to 1023.75(decimal) with 2 fractional bits.

Vertical Lowpass Filter (12):

Function: Vertical intra field, $1/16^{\text{th}}$ band, lowpass, prefilter and anti-alias filter. Cascade of 3, vertical running sum filters with lengths 16, 12 and 5 field lines. The output of this cascade of running sums is divided by 1024 to give an overall D.C. gain of $15/16$. The overall length of the filter is 31 field lines.

Input: As Temporal Halfband Lowpass Filter output.

Output: As Temporal Halfband Lowpass Filter output.

Horizontal Lowpass Filter (10):

Function: Horizontal, $1/32^{\text{nd}}$ band, lowpass, prefilter. Cascade of 3, horizontal, running sum filters with lengths 32, 21 and 12 pixels. The output of this cascade is divided by 8192 to give an overall D.C. gain of $63/64$. The overall length of the filter is 63 pixels.

Input: As Vertical Lowpass Filter output.

Output: As Vertical Lowpass Filter output.

Temporal Differentiator (16):

Function: Temporal differentiation of prefiltered luminance signal. Implemented as a vertical/temporal filter for interlaced inputs.

$$\text{Temporal Differentiator coefficients} = 1/4 \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}$$

Input: As Horizontal Lowpass Filter output.

Output: 12 bit 2's complement binary representing the range -2^9 to $(+2^9 - 2^{-2})$.

Horizontal Differentiator (17):

Function: Horizontal differentiation of prefiltered luminance signal. 3 tap horizontal filter with coefficients $\frac{1}{2}(1, 0, -1)$ on consecutive pixels.

Input: As Horizontal Lowpass Filter output.

Output: 8 bit 2's complement binary representing the range -2^4 to $(+2^4 - 2^{-3})$.

Vertical Differentiator (18):

Function: Vertical differentiation of prefiltered luminance signal. 3 tap, intra-field, vertical filter with coefficients $\frac{1}{2}(1, 0, -1)$ on consecutive field lines.

Input: As Horizontal Lowpass Filter output.

Output: 8 bit 2's complement binary representing the range -2^4 to $(+2^4 - 2^{-3})$.

Compensating Delay (19):

Function: Delay of 1 input field.

Input & Output: As Horizontal Lowpass Filter output.

Vertical/Temporal Interpolators (20):

Function: Conversion between input and output scanning standards. Cascade of intra field, 2 field line linear interpolator and 2 field linear interpolator, i.e. a

vertical/temporal bi-linear interpolator. Interpolation accuracy to nearest $1/32^{\text{nd}}$ field line and nearest $1/16^{\text{th}}$ field period.

Inputs: as indicated in figure 6 and specified above.

Outputs: same precision as inputs.

θ : Orientation of spatial gradient vector of image brightness. 12 bit unipolar binary spanning the range 0 to 2π i.e. quantisation step is $2\pi/2^{12}$. This is the same as 2's complement binary spanning the range $-\pi$ to $+\pi$.

$|\nabla I|$: Magnitude of spatial gradient vector of image brightness. 12 bit unipolar binary spanning the range 0 to 16 (input grey levels /pixel) with 8 fractional bits.

n : Noise level of $|\nabla I|$ adjustable from 1 to 16 input grey levels / pixel.

vn : Motion vector of current pixel in direction of brightness gradient. 12 bit, 2's complement binary clipped to the range -2^6 to $(+2^6 - 2^{-5})$ pixels/field.

Polar to Rectangular Co-ordinate Converter (40):

Inputs: as vn & θ above

Outputs: 12 bit, 2's complement binary representing the range -2^6 to $(+2^6 - 2^{-5})$

Lookup Tables No.1 & No.2 (figure 7 and 9)

Function: Cosine and Sine lookup tables respectively.

Inputs: as θ above.

Outputs: 12 bit, 2's complement binary representing the range -1 to $(+1 - 2^{-11})$.

Region Averaging Filters (38,39,42):

Function: Averaging signals over a region of the image.
95 pixels by 47 field lines, intrafield, running average filter.

Inputs & Outputs: 12 bit 2's complement binary.

Spatial Interpolators (44):

Function: Converting spatial scanning to allow for picture resizing. Spatial, intrafield bilinear interpolator.
Interpolation accuracy to nearest 1/32nd field line and nearest 1/16th pixel.

Inputs: 12 bit 2's complement binary.

Outputs: 12 or 8/9 bit 2's complement binary.

Upper Interpolators feeding multipliers 12 bit.

Lower Interpolators feeding Lookup tables 8/9 bit (to ensure a practical size table).

Look Up Tables 3 to 5 (figure 7):

Function: Calculating matrix 'Z' defined in equation 14 above.

Parameters n_1 & n_2 adjust on test (approx. 2-5).

Inputs: 8/9 bit 2's complement binary representing -1 to (approx.) +1.

Outputs: 12 bit 2's complement binary representing the range 16 to (+16 - 2-5).

Multipliers & Accumulators:

Inputs & Outputs: 12 bit 2's complement binary.

Motion Vector Output:

Output of figure 7.

Motion vectors are measure in input picture lines (not field lines) or horizontal pixels per input field period.

Motion speeds are unlikely to exceed ± 48 pixels/field but an extra bit is provided for headroom.

Raster scanned interlaced fields.

Active field depends on output standard: 720 pixels $\times$ 288 or 244 field lines.

12 bit signal, 2's complement coding, 8 integer and 4 fractional bits representing the range -128 to $(+128-2^4)$

Spread Vectors S_1 and S_2 (Output of figure 10):

Spread vectors represent the measurement spread of the output motion vectors parallel and perpendicular to edges in the input image sequence.

The spread vectors are of magnitude σ^{-1} (where σ represents standard deviation) and point in the direction of the principle axes of the expected distribution of measurement error.

Each spread vector has two components each coded using two complement fractional binary representing the range -1 to $(+1-2^{-7})$.

Confidence Output:

Output of figure 10, derivation of confidence signal described above.

The confidence signal is an indication of the reliability of the 'Output Motion Vector'. Confidence of 1 represents high confidence, 0 represents no confidence.

The confidence signal uses 8 bit linear coding with 8 fractional bits representing the range 0 to $(1-2^{-8})$.

References

1. Aggarwal, J.K. & Nandhakumar, N. 1988. On the computation of motion from sequences of images - a review. Proc. IEEE, vol. 76, pp. 917-935, August 1988.
2. Bierling, M., Thoma, R. 1986. Motion compensating field interpolation using a hierarchically structured displacement estimator. Signal Processing, Volume 11, No. 4, December 1986, pp. 387-404. Elsevier Science publishers.
3. Borer, T.J., 1992. Television Standards Conversion. Ph.D. Thesis, Dept. Electronic & Electrical Engineering, University of Surrey, Guildford, Surrey, GU2 5XH, UK. October 1992.
4. Borer, T.J., 1995. Adaptive Motion trajectories for interpolating moving images. UK Patent Application No. 9519311.6, filed 21 St September 1995.
5. Cafforio, C., Rocca, F. 1983. The differential method for image motion estimation. Image sequence processing and dynamic scene analysis (ed. T.S. Huang). Springer-Verlag, pp 104-124, 1983.
6. Cafforo, C., Rocca, F., Tubaro, S., 1990. Motion Compensated Image Interpolation. IEEE Trans. on Comm. Vol. 38, No. 2, February 1990, pp215-222.
7. Dubois, E., Konrad, J., 1990. Review of techniques for motion estimation and motion compensation. Fourth International Colloquium on Advanced Television Systems, Ottawa, Canada, June 1990. Organised by CBC Engineering, Montreal, Quebec, Canada.
8. Fennema, C.L., Thompson, W.B., 1979. Velocity determination in scenes containing several moving objects. Computer Vision, Graphics and Image Processing, Vol. 9, pp 301-315, 1979.
9. Huahge, T.S., Tsai, R.Y., 1981. Image sequence analysis: Motion estimation. Image sequence analysis, T.S. Huang (editor), Springer-Verlag, Berlin, Germany, 1981, pp. 1-18.
10. Konrad, J., 1990. Issues of accuracy and complexity in motion compensation for ATV systems. Contribution to 'Les Assises Des Jeunes Chercheurs', CBC, Montreal, June 1990.

11. Lim, J.S., 1990. Two-dimensional signal and image processing. PrenticeHall 1990, ISBN 0-13-934563-9, pp 497-511.
12. Martinez, D.M. 1987. Model-based motion estimation and its application to restoration and interpolation of motion pictures. RLE Technical Report No.530. June 1987. Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139 USA.
13. Netravali, A.N., Robbins, J.D. 1979. Motion compensated television coding, Part 1. Bell Syst. Tech. J., vol. 58, pp 631-670, March 1979.
14. Paquin, R., Dubois, E., 1983. A spatio-temporal gradient method for estimating the displacement vector field in time-varying imagery. ComputerVision, Graphics and Image Processing, Vol. 21, 1983, pp 205-221.
15. Robert, P., Cafforio, C., Rocca, F., 1985. Time/Space recursion for differential motion estimation. Spie Symp., Cannes, France, November 1985.
16. Thomson, R. 1995. Problems of Estimation and Measurement of Motion in Television. I.E.E. Colloquium on motion reproduction in television. I.E.E Digest No: 1995/093, 3rd May 1995.
17. Vega-riveros, J.F., Jabbour, K. 1986. Review of motion analysis techniques. IEE Proceedings, Vol. 136, Pt I., No. 6, December 1989.
18. Wu, S.F., Kittler, J., 1990. A differential method for the simultaneous estimation of rotation, change of scale and translation. ImageCommunication, Vol. 2, No. 1, May 1990, pp 69-80.

WHAT IS CLAIMED IS:

1. A motion vector estimation apparatus for use in video signal processing which is adapted to generate motion vector sampled on an output sampling lattice, comprising:

a first means for spatially filtering an input signal sampled on an input lattice;

a second means for means operating on said input signal for calculating image gradients sampled on said input lattice;

a third means for converting the signal sampled on said input lattice to a signal sample on said output sampling lattice said first, second and third means operating on said input signal in any order and having a plurality of image gradients sampled on said output lattice as an output; and

a fourth means for calculating motion vectors,

wherein said fourth means for calculating motion vectors has as an input said plurality of image gradients sampled on said output sampling lattice.

2. A motion vector estimation apparatus as claimed in claim 1, wherein said first means for spatially filtering the signal includes spatial low pass filters.

3. A motion vector estimation apparatus as claimed in claim 1, wherein said second means for calculating the image gradients includes temporal and spatial differentiators.

4. A motion vector estimation apparatus as claimed in claim 1, wherein said third means for converting the input signal includes a vertical/temporal interpolator.

5. A motion vector estimation apparatus as claimed in claim 1, and further comprising a multiplier array for calculating image gradient products from a plurality of said image gradients generated on the output standard, filters for summing said plurality of image gradient products, wherein said second means for calculating the motion vectors utilizes the sums of a plurality of image gradient products to generate the best fit motion vector.

6. A motion vector estimation apparatus as claimed in claim 5 wherein said best fit motion vector is determined in accordance with the equation:

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = - \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} e_1 e'_1 + \frac{\lambda_2}{\lambda_2^2 + n_2^2} e_2 e'_2 \right) \cdot \begin{bmatrix} \sigma_{xx}^2 \\ \sigma_{yy}^2 \end{bmatrix}$$

where $[u_0 \ v_0]$ is the best fitting motion vector, σ represents the roots of sums of products of image gradients the subscript signifying the particular image gradients, and e and λ are the eigen vectors and eigen values of M , where

$$M = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix}$$

7. A motion vector estimation apparatus as claimed in claim 1, and further comprising means for calculating from a plurality of image gradients generated on said output sampling lattice, the spatial image gradient ($|\Delta I|$), the angle between the spatial image gradient and the horizontal (θ) and the motion speed (vn) in the direction of the image gradient vector.

8. A motion vector estimation apparatus as claimed in claim 7, wherein said means for calculating motion vectors calculates the best fitting motion vector in accordance with the equation:

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = - \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} e_1 e'_1 + \frac{\lambda_2}{\lambda_2^2 + n_2^2} e_2 e'_2 \right) \cdot \begin{bmatrix} \sum vn \cdot \cos(\theta) \\ \sum vn \cdot \sin(\theta) \end{bmatrix}$$

where e and λ are the eigen vectors and eigen values of:

$$\begin{bmatrix} \sum \cos^2(\theta) & \sum \cos(\theta) \cdot \sin(\theta) \\ \sum \cos(\theta) \cdot \sin(\theta) & \sum \sin^2(\theta) \end{bmatrix}$$

9. A method of estimating motion vectors on an output sampling lattice for use in video-signal processing, comprising the following steps:

- (a) spatially filtering a video signal,
- (b) converting the signal from an input sampling lattice to said output sampling lattice,
- (c) calculating a plurality of image gradients;

wherein steps a to c are carried out in any order, and calculating motion vectors on said output sampling lattice from said image gradients generated on said output sampling lattice.

10. A method of motion vector estimation as claimed in claim 9, wherein said step of spatially filtering the input signal includes filtering by spatial low pass filters.

11. A method of motion vector estimation as claimed in claim 9, wherein calculating said plurality of image gradients includes temporally and spatially differentiating the spatially filtered signal.

12. A method of motion vector estimation as claimed in claim 9, wherein converting the signal includes vertical and temporal interpolation.

13. A method of motion vector estimation as claimed in claim 9 and further comprising calculating image gradient products from a plurality of said image gradients generated on the output standard, summing said plurality of image gradient products, wherein calculating the motion vectors comprises utilizing the sums of a plurality of image gradient products to generate the best fit motion vector.

14. A method of motion vector estimation as claimed in claim 13, wherein said best fit motion vector is determined in accordance with the equation:

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = - \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} e_1 e_1' + \frac{\lambda_2}{\lambda_2^2 + n_2^2} e_2 e_2' \right) \cdot \begin{bmatrix} \sigma_{x'}^2 \\ \sigma_{y'}^2 \end{bmatrix}$$

where $[u_0 \ v_0]$ is the best fitting motion vector, σ represents the roots of sums of products of image gradients the subscript signifying the particular image gradients, and e and λ are the eigen vectors and eigen values of M , where

$$M = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix}$$

15. A method of motion vector estimation as claimed in claim 9, and further comprising calculating from a plurality of image gradients generated on said output sampling lattice, the spatial image gradient ($|\Delta I|$), the angle between the spatial image gradient and the horizontal (θ) and the motion speed (vn) in the direction of the image gradient vector.

16. A method of motion vector estimation as claimed in claim 15, wherein said means for calculating motion vectors calculates the best fitting motion vector in accordance with the equation:

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = - \left(\frac{\lambda_1}{\lambda_1^2 + n_1^2} e_1 e_1' + \frac{\lambda_2}{\lambda_2^2 + n_2^2} e_2 e_2' \right) \cdot \begin{bmatrix} \sum vn \cdot \cos(\theta) \\ \sum vn \cdot \sin(\theta) \end{bmatrix}$$

where e and λ are the eigen vectors and eigen values of:

$$\begin{bmatrix} \sum \cos^2(\theta) & \sum \cos(\theta) \cdot \sin(\theta) \\ \sum \cos(\theta) \cdot \sin(\theta) & \sum \sin^2(\theta) \end{bmatrix}$$

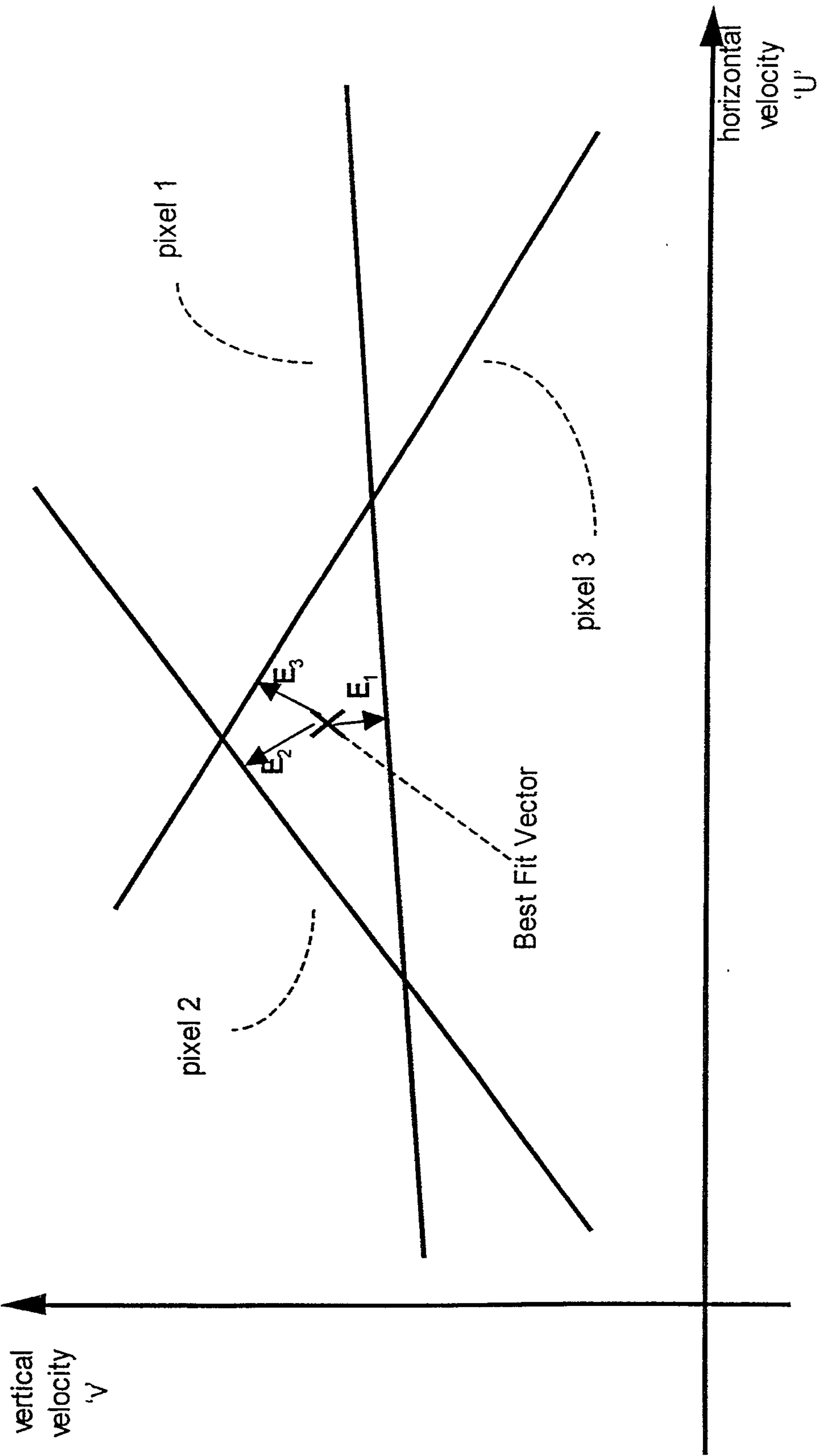


Figure 1

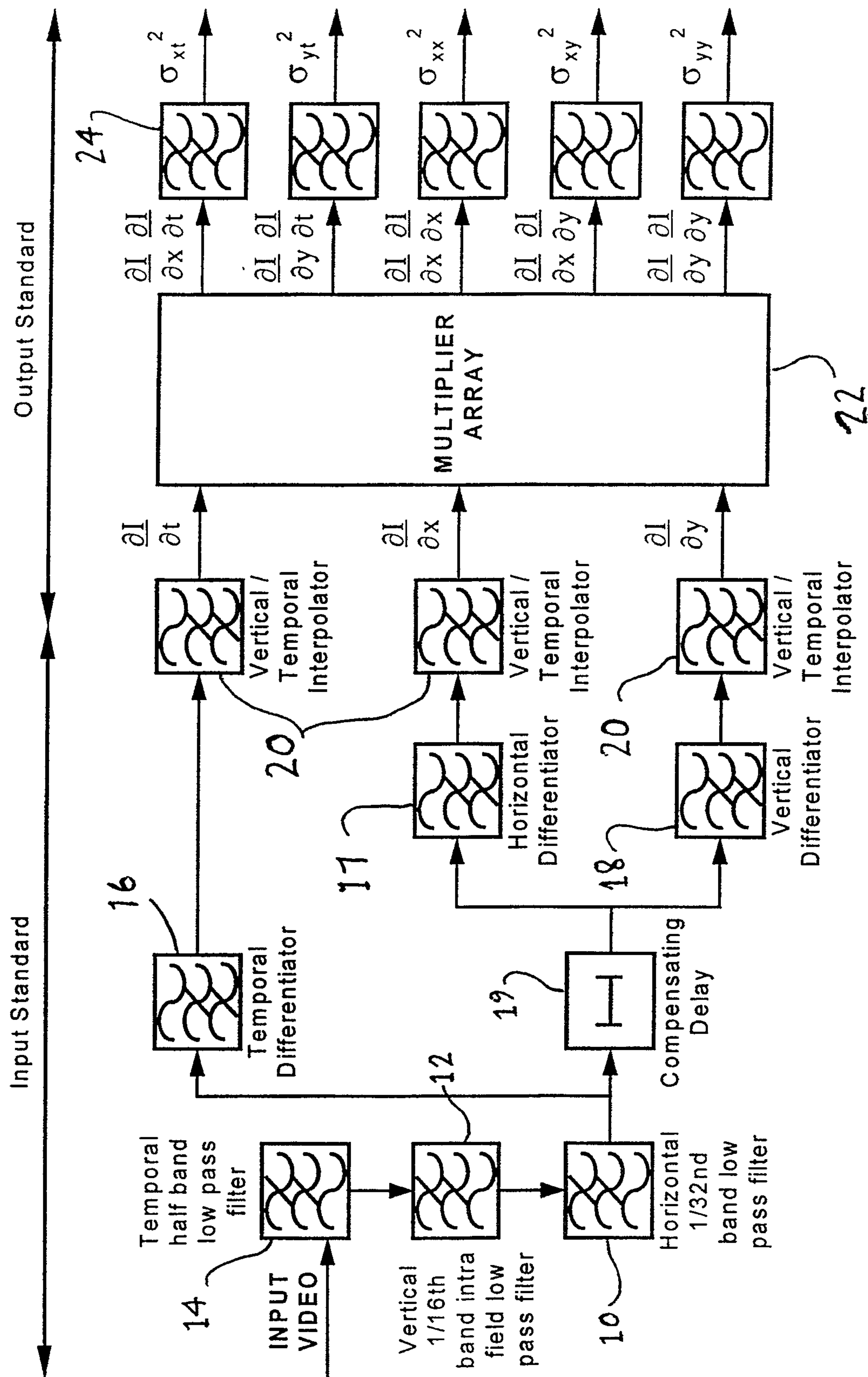


Figure 2

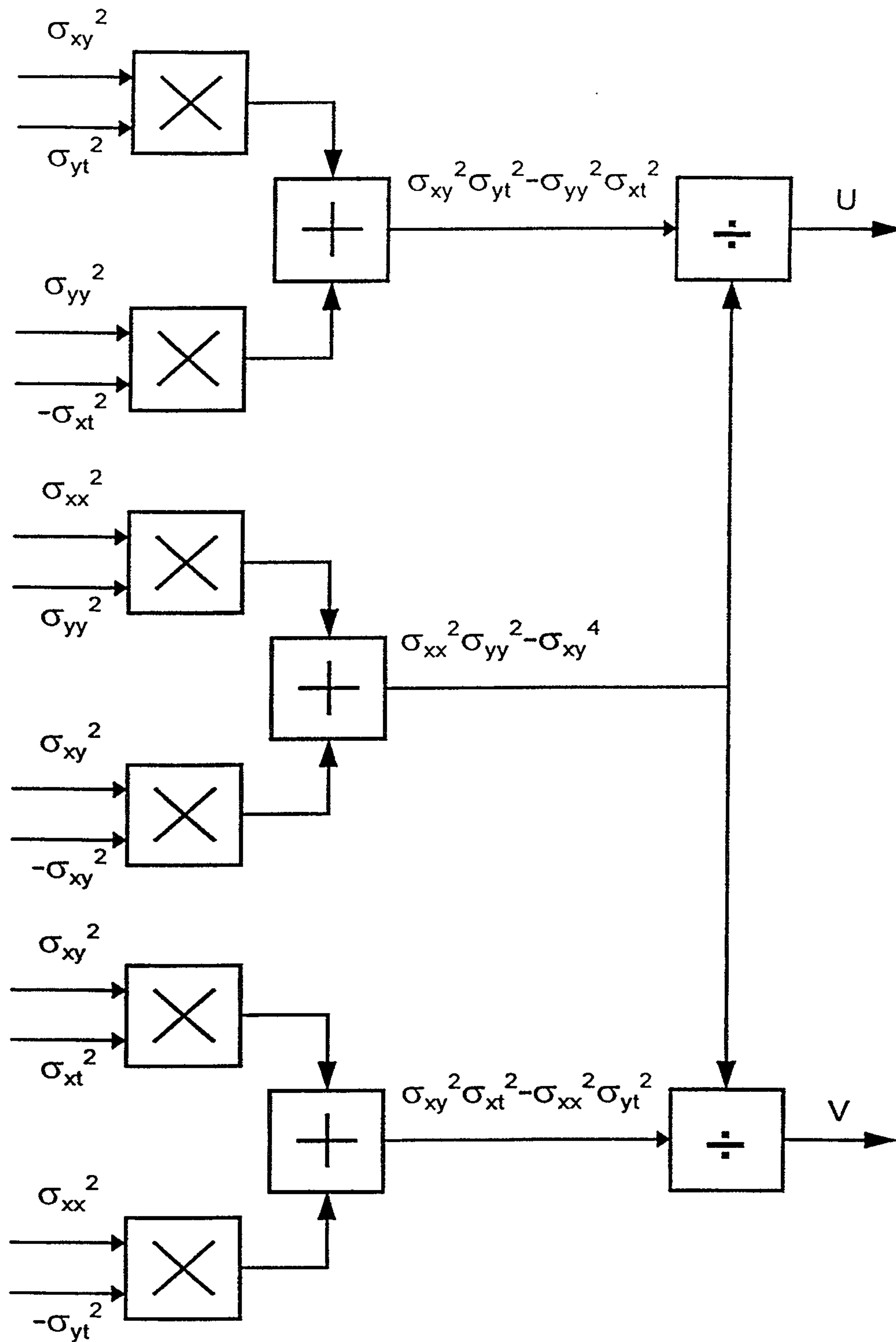


Figure 3

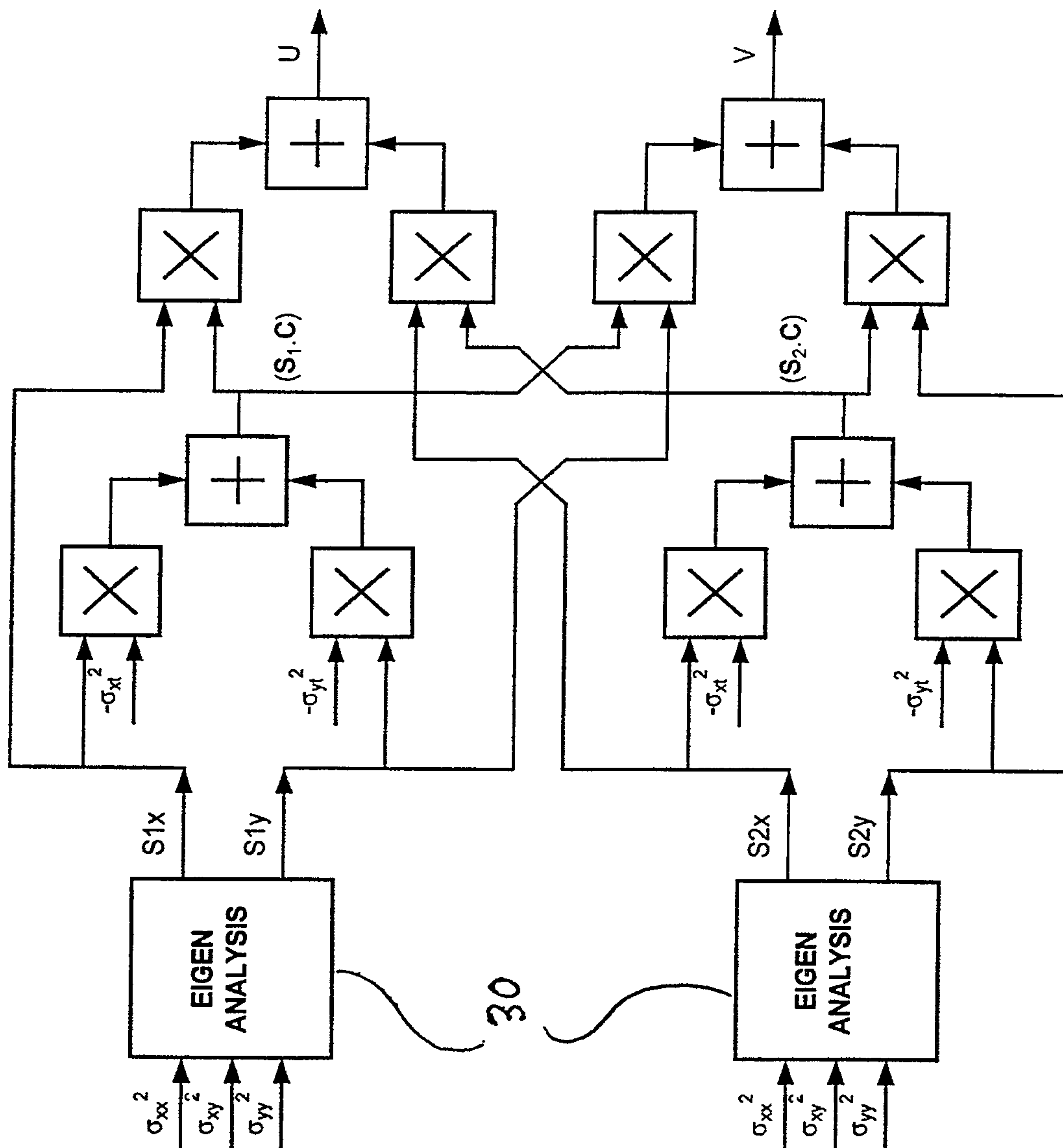


Figure 4

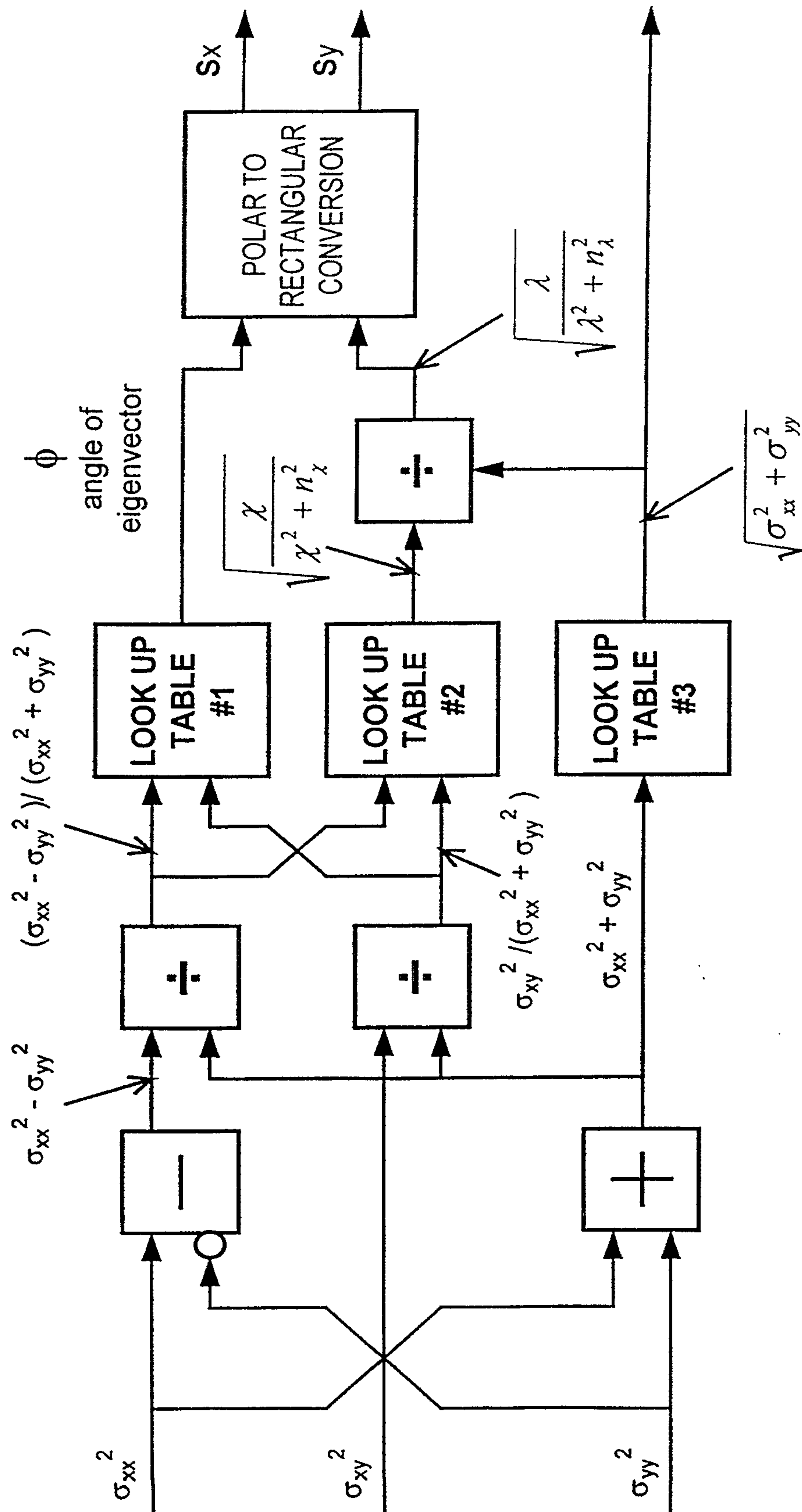


Figure 5

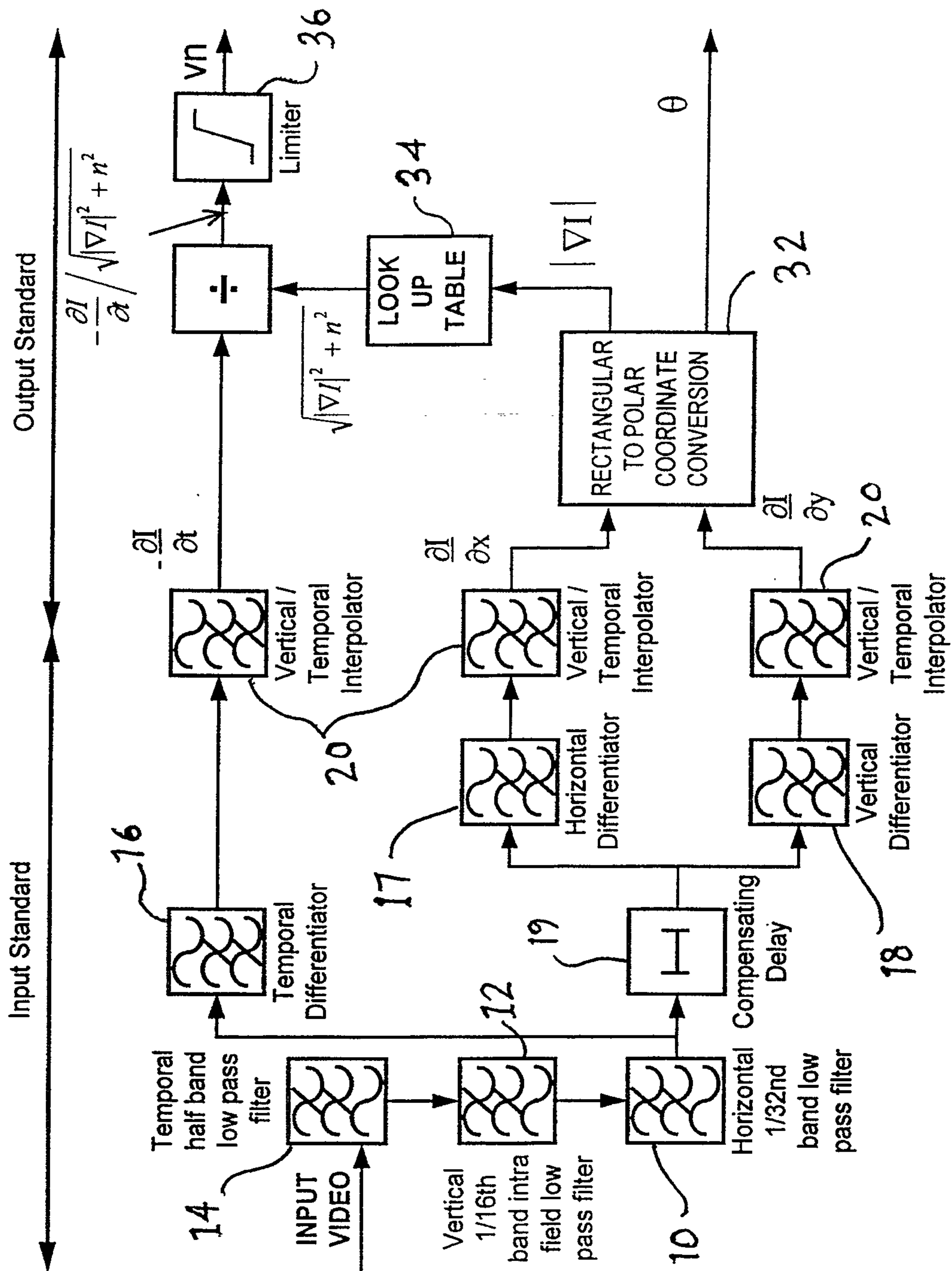


Figure 6

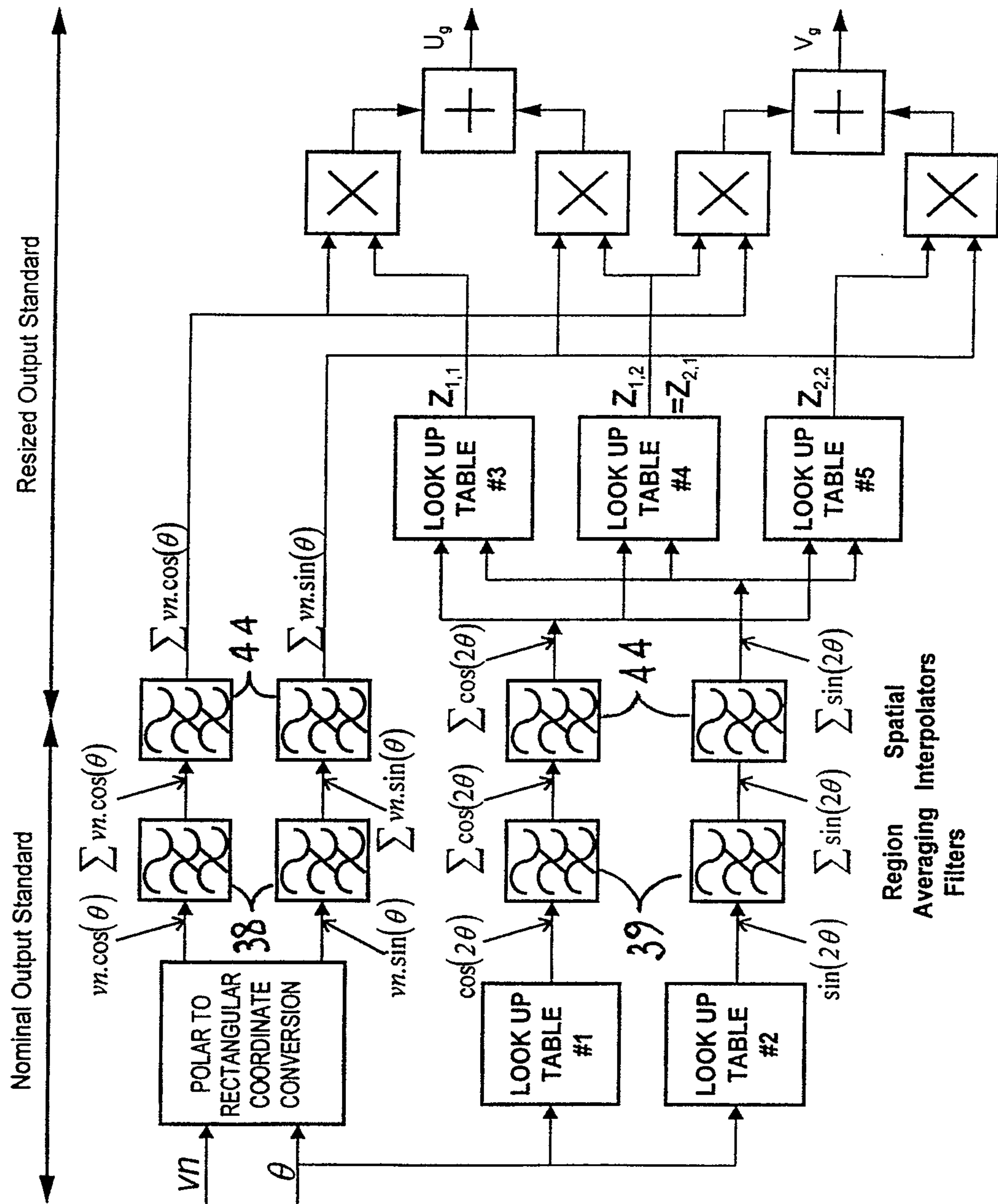


Figure 7

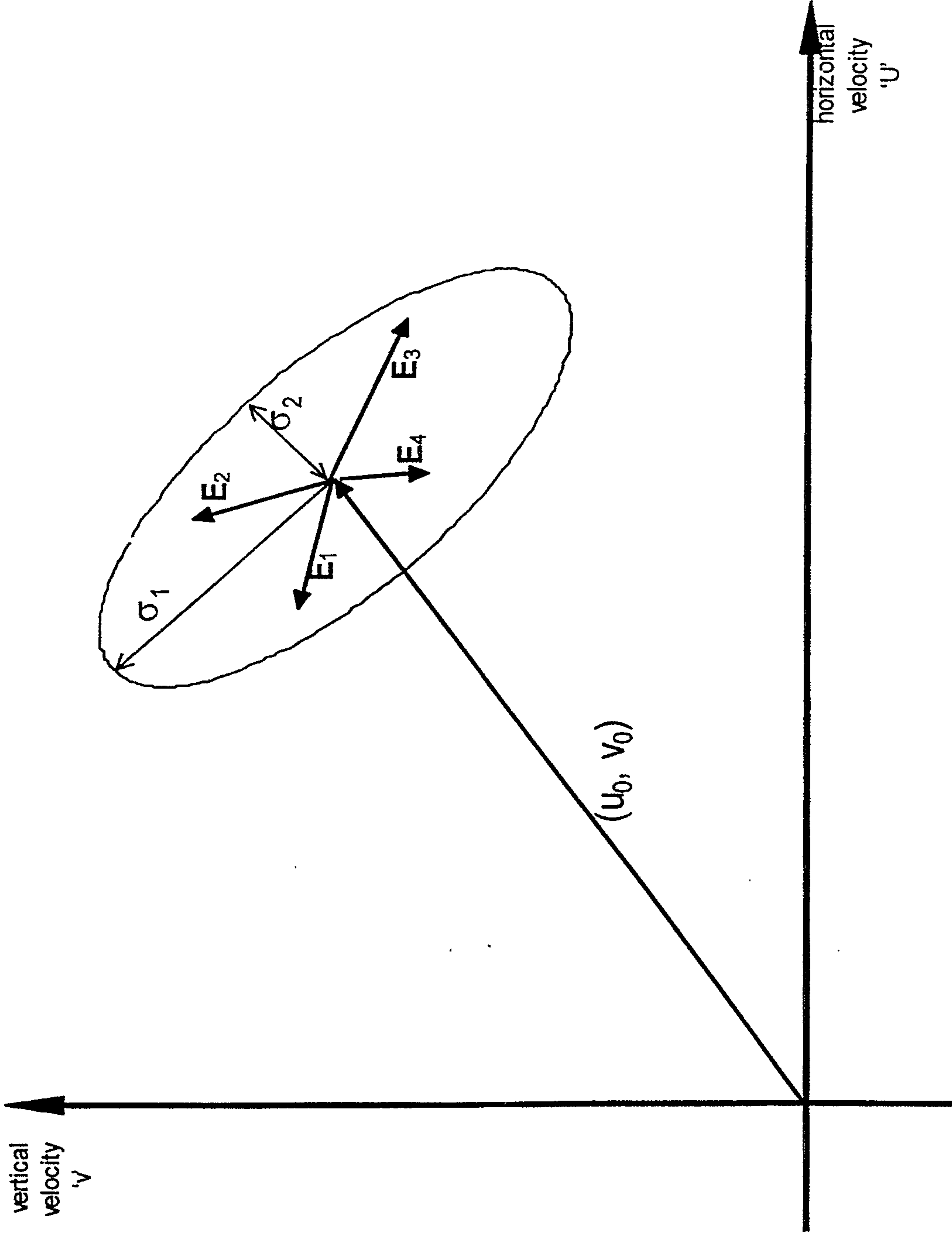


Figure 8

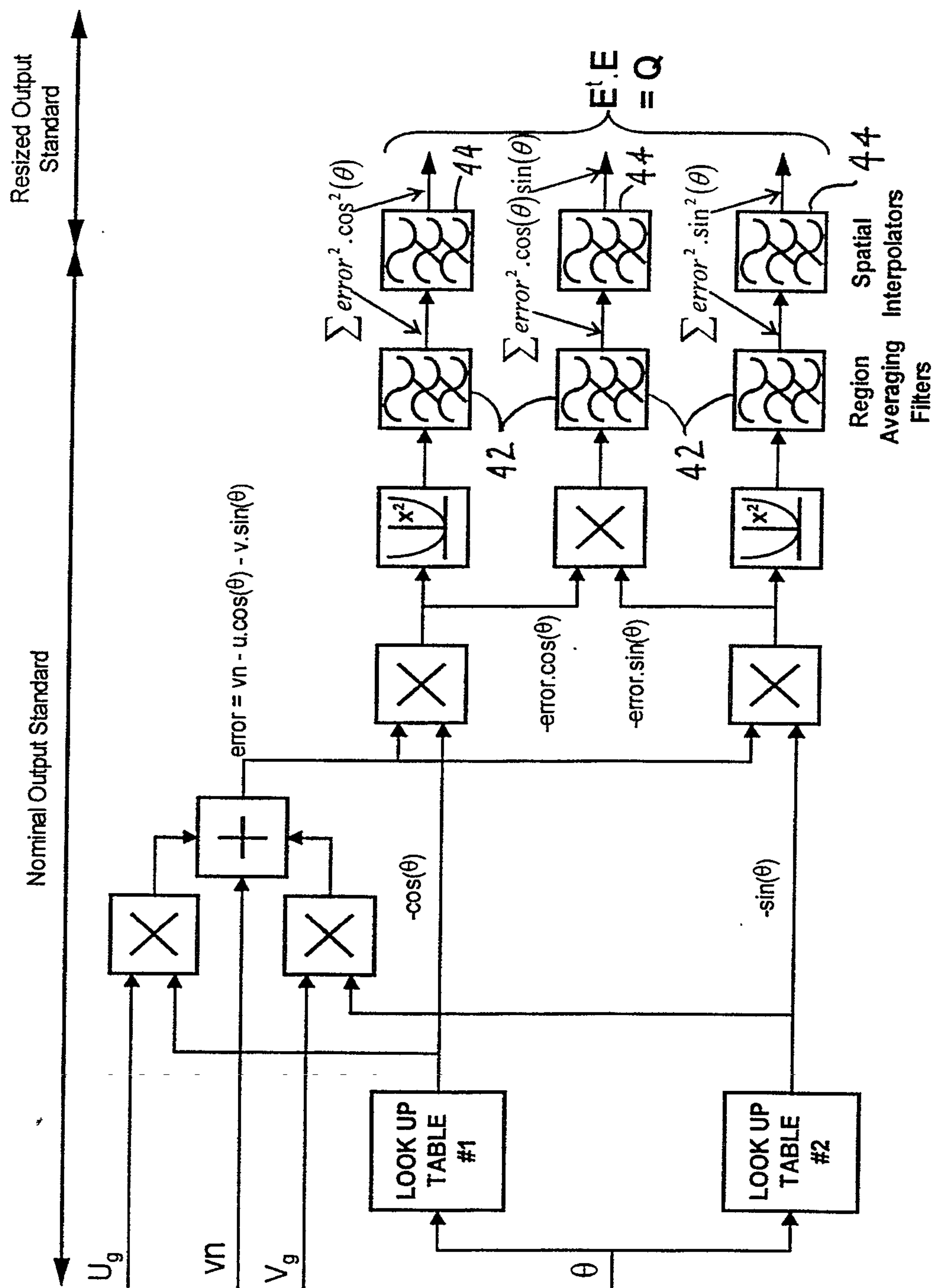


Figure 9

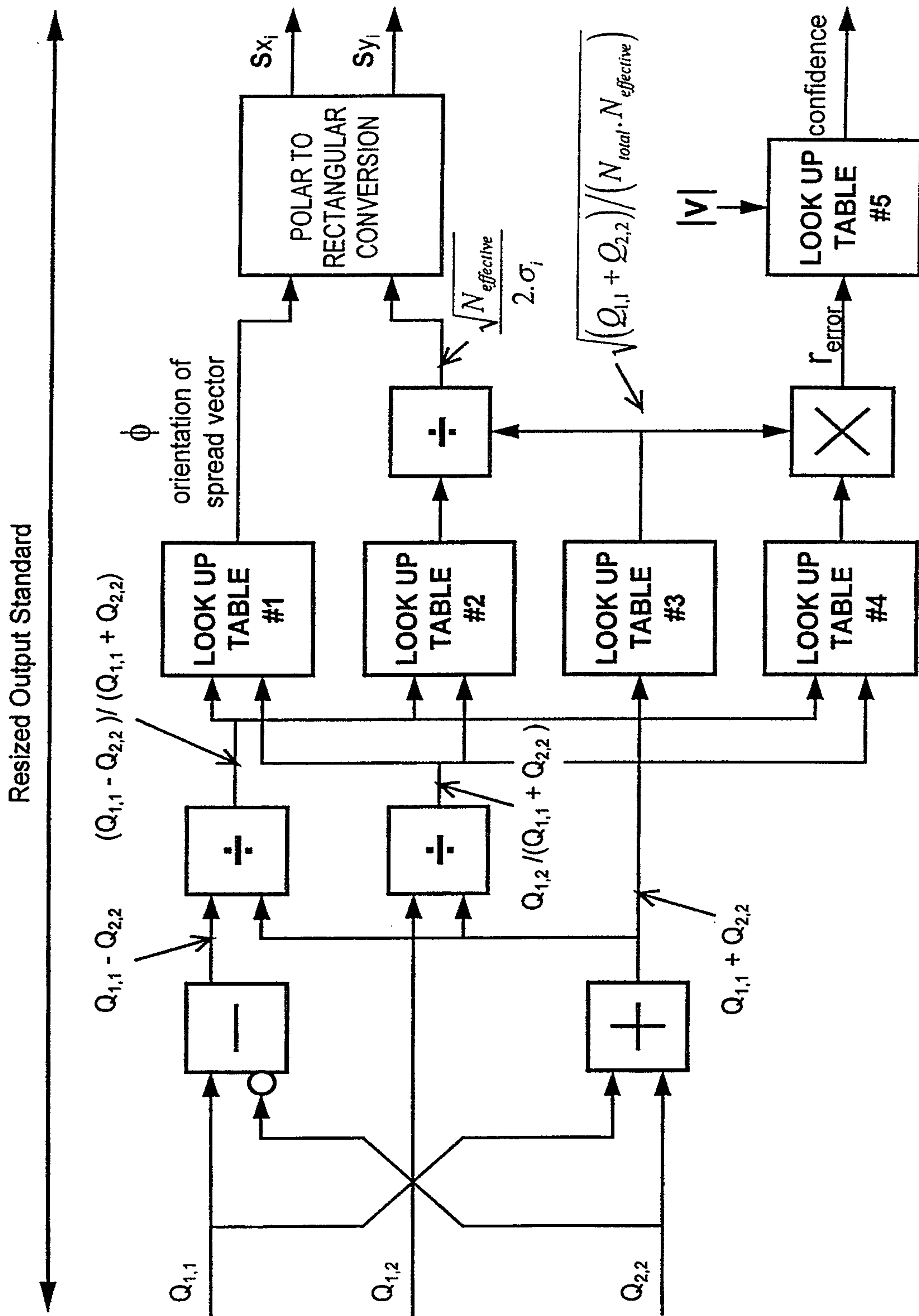


Figure 10

Input Standard

Output Standard

