COMBINATIVE MOTION ESTIMATION ALGORITHM AND THE CORRESPONDING ARCHITECTURE FOR COMPLEX MOTION PHENOMENON

Pei-Chuan Liu, Wen-Thong Chang and Wen-Zen Shen
Institute of Electronics
National Chiao Tung University
HsinChu, Taiwan, R.O.C.

Abstract

Combining the features of block matching algorithm and block recursive algorithm a new motion estimation method is proposed for complex motion phenomenon. With the full searching procedure used in block matching algorithm and the adaptive searching procedure used in block recursive algorithm, the proposed combinative algorithm can be used to estimate more complex motion parameters such as translation, zoom, rotation and image deformation and get good performance. To reduce the computational complexity, the corresponding architecture of the combinative algorithm is also proposed. With this architecture the computation amount can be largely reduced to make the proposed algorithm can be used for real time motion estimation.

I. Introduction

Block matching method [1-2] based on two-parameter motion model has been widely used for motion estimation. In this method the full searching procedure is used to find the best matched blocks between two iterative images in a predefined searching range. Using the full searching procedure the block matching method can get the global minimum solution. However, in many cases, pure translatory motion assumption is not sufficient to describe the image motion phenomenon. For this, the three-parameter motion model [3], the six-parameter affine transform and the eight-parameter bilinear transform [4] have been proposed to describe the more complex motion phenomenon. In these motion models block deformation on block size or block shape is seen. The lack of ability of block deformation makes block matching algorithm hard to be applied for the motion estimation with more complex motion phenomenon.

Block recursive method is an adaptive motion estimation algorithm that combines the block process concept [5] and the recursive searching procedure. The adaptive searching procedure makes the block recursive method to have the ability of block deformation and so it is suit for complex motion estimation. Based on the block recursive algorithm and the quantized state algorithm [6-7], a quantized block recursive method is proposed to reduce the computational amount with the least performance loss. In spite of having the ability of block deformation for complex motion estimation, because of using the recursive searching procedure, block recursive algorithm meets the local minimum problem [8] and may estimate to get the local minimum solution but miss the global minimum solution.

To solve the local minimum problem, a combinative motion estimation algorithm that combines the features of block matching method and block recursive method is proposed. In this combinative motion estimation method the recursive searching procedure is used for the estimation of complex motion parameters and the full searching procedure is used to solve the local minimum problem. So with the combinative estimation method the block deformation problem and the local minimum problem can both be solved.

Under the consideration of hardware implementation, the large computational amount makes the combinative algorithm hard to be implemented. For this, the architecture design of the proposed combinative algorithm is discussed. In this architecture the computation amount could be largely reduced according to the coordinate relationship. The rest of this paper is organized as follows. In section II we first briefly introduce the block recursive method for complex motion estimation. The quantized block recursive algorithm is also derived in this section. After this, the combinative motion estimation algorithm and the performance analysis is derived. For hardware implementation, the corresponding architecture

Contributed Paper
Manuscript received November 25, 1997

design of the proposed algorithm is introduced in section III. Finally, a conclusion is made in section IV.

II. The combinative motion estimation algorithm

To describe the different motion phenomenon of moving object some motion representation models are proposed. These motion models describe the coordinate relationship before and after deformation. In three-parameter motion model the transformation of the coordinate can be described as

$$(X_2, Y_2) = (c_1 X_1 + c_2, c_1 Y_1 + c_3) \quad (1)$$

where c_1 is the zooming motion parameter c_2 and c_3 are the panning motion parameters to be estimated. In six-parameter motion model the zoom, pan and rotate of object can be described by the affine transformation. The transformation of the coordinate can be described as

$$(X_2, Y_2) = (c_1 X_1 + c_2 Y_1 + c_3, c_4 X_1 + c_5 Y_1 + c_6) \quad (2)$$

In eight-parameter motion model the bilinear transformation of the coordinate can be described as

$$(X_2, Y_2) = (c_1 X_1 + c_2 Y_1 + c_3 X_1 Y_1 + c_4, c_5 X_1 + c_6 Y_1 + c_7 X_1 Y_1 + c_8) \quad (3)$$

Besides of the zoom, pan and rotate of object, the image distortion is also considered in the eight-parameter motion model. With this model, the displaced frame difference (DFD) can be described as

$$\begin{aligned} &DFD(x,y) \\ &= G_x[x(c_1 - a_1) + y(c_2 - a_2) + xy(c_3 - a_3) + (c_4 - a_4)] \\ &+ G_y[x(c_5 - a_5) + y(c_6 - a_6) + xy(c_7 - a_7) + (c_8 - a_8)] \end{aligned} \quad (4)$$

where a_1 to a_8 are the current estimates of the true motion parameters c_1 to c_8. G_x and G_y are the gradients in X and Y direction.

For N points of signal, Eqn.(4) can be written in a matrix form as

$$\mathbf{D} = \mathbf{G}(\mathbf{C} - \mathbf{A}^{(p)}) \quad (5)$$

with

$$\mathbf{D} = \begin{bmatrix} DFD(x_1^{(p)}, y_1^{(p)}) \\ DFD(x_2^{(p)}, y_2^{(p)}) \\ \vdots \\ \vdots \\ DFD(x_N^{(p)}, y_N^{(p)}) \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} G_{x_1}^{(p)}x_1 & G_{x_1}^{(p)}y_1 & G_{x_1}^{(p)}x_1y_1 & G_{x_1}^{(p)} & G_{y_1}^{(p)}x_1 & G_{y_1}^{(p)}y_1 & G_{y_1}^{(p)}x_1y_1 & G_{y_1}^{(p)} \\ G_{x_2}^{(p)}x_2 & G_{x_2}^{(p)}y_2 & G_{x_2}^{(p)}x_2y_2 & G_{x_2}^{(p)} & G_{y_2}^{(p)}x_2 & G_{y_2}^{(p)}y_2 & G_{y_2}^{(p)}x_2y_2 & G_{y_2}^{(p)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ G_{x_N}^{(p)}x_N & G_{x_N}^{(p)}y_N & G_{x_N}^{(p)}x_Ny_N & G_{x_N}^{(p)} & G_{y_N}^{(p)}x_N & G_{y_N}^{(p)}y_N & G_{y_N}^{(p)}x_Ny_N & G_{y_N}^{(p)} \end{bmatrix}$$

where

$\mathbf{C} = [c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8]^T$ is the true motion parameter vector.

$\mathbf{A}^{(p)} = [a_1^{(p)}, a_2^{(p)}, a_3^{(p)}, a_4^{(p)}, a_5^{(p)}, a_6^{(p)}, a_7^{(p)}, a_8^{(p)}]^T$ is the estimated motion parameter vector at iteration p.

The Block recursive algorithm is to estimate $\mathbf{A}^{(p)}$ such that $E\{\|DFD^{(p)}\|^2\}$ is minimized. Based on Eqn.(5) and the steepest descent adaptive algorithm, the block recursive algorithm is derived as

$$\mathbf{A}^{(p+1)} = \mathbf{A}^{(p)} - \varepsilon \mathbf{G}^T\mathbf{D} \quad (6)$$

Where ε is the convergence factor. To reduce the computational amount of block recursive algorithm, a quantized block recursive method is proposed. In this quantized algorithm the gradient value is quantized to be number of 2's power and so the multiplication with gradient could be replaced by the shift operation. Also, to increase the stability of algorithm, the compensation for pixel with large coordinate is made to replace the convergence factor ε with

$$\varepsilon = \begin{bmatrix} \dfrac{\varepsilon_1}{X_0^2 + Y_0^2} & 0 & 0 \\ 0 & \varepsilon_2 & 0 \\ 0 & 0 & \varepsilon_2 \end{bmatrix} \quad (7)$$

To reduce the computation amount the convergence factor is chosen to be number of 2's power. And so the multiplication with convergence factor could be replaced by shift operation.

Based on the block recursive algorithm in Eqn.(6)-(7) and the block matching algorithm, a combinative algorithm is proposed. In this algorithm, first the block recursive method is used to find the block deformation and the complex motion parameters of moving block and then the block matching method is used to find the global minimum solution. The estimation procedure of the combinative algorithm can be organized as following :

1. Use block recursive algorithm to find the complex motion parameters. In three-parameter motion model it is to estimate a_1, a_2 and a_3.
2. Use the complex motion parameters to find the shape and size of the deformed block. In three-parameter motion model it is to identify the corresponding points as $(x'_i, y'_i) = (a_1x_i+a_2, a_1y_i+a_3)$.
3. Use block matching algorithm to compare the displaced deformed block with the checking block and find the best matched motion vector dx and dy.
4. Add motion vector dx and dy to the estimated complex motion parameters. In three-parameter motion model it is to add dx and dy with a_2 and a_3 to get the new values of a_2 and a_3.

After proposed the combinative algorithm, the performance comparison of the proposed combinative algorithm with the individual algorithm is made. The image used for simulation is the image sequence "Football". The block size of 16×16 is chosen to be the size of basic block. The searching range of the block matching method is from –8 to 8 and the estimation iteration of the quantized block recursive method is 3. Because there is only zooming effect existing in the image sequence "Football", in our simulation the three-parameter motion model is used. To see the effect of zooming motion some slices of picture are skipped and then picture No.1 and No.5 are used as the reference pictures to get larger zooming effect. The mean absolute value of DFD is used for the judgement of performance. The simulation results are shown in Table.1. In Table.1 it shows the mean absolute value of DFD for blocks in different position with the combinative method, the block recursive method and the block matching method. The corresponding zooming motion parameters are also shown to see the zooming effect of different moving blocks. According to the simulation results it could be found that the performance of the block matching method is better than that of the block recursive method when the zooming effect is not so series. It is because the block matching method uses the full searching process and can get the global optimum solution. With the increase of zooming effect the performance of the block matching method gets worse because of the lack of ability of block deformation. The performance of the combinative algorithm is the best in each case. According to these results it could be found that the combinative algorithm is suit for the motion estimation with complex motion phenomenon and get the best output performance.

III. The architecture design

After made the performance analysis of the proposed combinative motion estimation algorithm, in this section the architecture design of the proposed algorithm is discussed. The architecture of the proposed algorithm is composed of two main parts : one part for block recursive algorithm and the other part for block matching algorithm. These two parts of architecture are discussed in the following.

3.1.1 The computation strategy of block recursive algorithm

Before discussing the architecture design first the computation procedure of the block recursive algorithm is discussed. In Eqn.(6) it can be found that the main computation of the block recursive algorithm is $\mathbf{G}^T\mathbf{D}$. In the quantized block recursive algorithm the multiplication with the quantized gradient could be replaced by the shift operation. So the rest of the multiplication needed to complete the calculation of $\mathbf{G}^T\mathbf{D}$ is the multiplication with coordinate x_i, y_i and x_iy_i. If we consider the relationship between the coordinates in a block, these multiplications can be largely reduced. Define the shift operation of the quantized gradient on DFD as

$$D_{xi}(p) = DFD(x_i(p), y_i(p))G_{xi}(p)$$
$$D_{yi}(p) = DFD(x_i(p), y_i(p))G_{yi}(p)$$

And then the calculation of $\mathbf{G}^T\mathbf{D} = \mathbf{U} = [u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8]^T$ will be

$$\begin{aligned}
u_1 &= \sum_{i=1}^{N} D_{xi}{}^{(p)} x_i \\
u_2 &= \sum_{i=1}^{N} D_{xi}{}^{(p)} y_i \\
u_3 &= \sum_{i=1}^{N} D_{xi}{}^{(p)} x_i y_i \\
u_4 &= \sum_{i=1}^{N} D_{xi}{}^{(p)} \\
u_5 &= \sum_{i=1}^{N} D_{yi}{}^{(p)} x_i \\
u_6 &= \sum_{i=1}^{N} D_{yi}{}^{(p)} y_i \\
u_7 &= \sum_{i=1}^{N} D_{yi}{}^{(p)} x_i y_i \\
u_8 &= \sum_{i=1}^{N} D_{yi}{}^{(p)}
\end{aligned} \tag{8}$$

To get u_1 or u_2 the multiplication with x_i or y_i is needed. First the relationship of coordinates in a block is taken into consideration. In this discussion the 4×4 block is used as the basic block. A 16×16 block could be composed of sixteen 4×4 blocks. The coordinate relationship of a 4×4 block is shown in Fig.1. It can be seen that the pixels in a same column get the same X directional coordinate. Also the pixels in a same row get the same Y directional coordinate. So there are only four different X indices and four different Y indices for the 16 coordinates in a 4×4 block. Denoting the left-upper corner coordinate as (x, y), the rest of the element can be denoted as $(x+i, y-j)$ with i, j = 0, 1, 2 and 3. By defining

The ith column summation of $D_{xi}(p)$ is

$$C_{xi} = D_{x(0+i)}(p) + D_{x(4+i)}(p) + D_{x(8+i)}(p) + D_{x(12+i)}(p)$$

The ith row summation of $D_{xi}(p)$ is

$$R_{xi} = D_{x(4i-3)}(p) + D_{x(4i-2)}(p) + D_{x(4i-1)}(p) + D_{x(4i-0)}(p)$$

Then

$$\begin{aligned}
u_1 &= (\sum_{i=1}^{4} C_{xi})x + [(C_{x2} + C_{x4}) + 2(C_{x3} + C_{x4})] \\
u_2 &= (\sum_{i=1}^{4} R_{xi})y - [(R_{x2} + R_{x4}) + 2(R_{x3} + R_{x4})]
\end{aligned} \tag{9}$$

According to Eqn.(9) there are only 2 multiplications needed to complete the multiplication with x_i and y_i. To get u_3 the multiplication with x_iy_i is needed. In Fig.2 it shows the relationship of x_iy_i in a 4×4 block. According to this relationship it could be found that the multiplication with x_iy_i could be decomposed into three parts of calculation as shown in Fig.3. The calculation of part 1 is first to get signal u_1 (complete the multiplication with x_i) and then to multiply u_1 with y. The calculation of part 2 is the computation of $[(R_{x2}+R_{x4})+2(R_{x3}+R_{x4})](-x)$. As for the calculation of part 3, according to the relationship of coordinate in Fig.1, we define

$$O_1 = (D_{x6}(p) + D_{x8}(p)) + 2(D_{x7}(p) + D_{x8}(p))$$
$$O_2 = (D_{x10}(p) + D_{x12}(p)) + 2(D_{x11}(p) + D_{x12}(p))$$
$$O_3 = (D_{x14}(p) + D_{x16}(p)) + 2(D_{x15}(p) + D_{x16}(p))$$

Then the calculation of part 3 is equal to the calculation $(O_1+O_3)+2(O_2+O_3)$. According to these three parts of calculation there are only two multiplications needed to get u_3. The calculations for u_5 to u_8 are similar to the calculations for u_1 to u_4. The only different is to replace the input $D_{xi}(p)$ with $D_{yi}(p)$.

After this, the update term $\mathbf{U}$ is used to estimate the new values of motion parameter. Then the next step is to calculate the new corresponding coordinate of pixel. The new coordinate $(x_i(p), y_i(p))$ is estimated as

$$x_i(p) = a_1(p)\, x_i + a_2(p) y_i + a_3(p)\, x_i y_i + a_4(p)$$
$$y_i(p) = a_5(p)\, x_i + a_6(p) y_i + a_7(p)\, x_i y_i + a_8(p)$$

As mentioned in above paragraph, there are only four different X indices and four different Y indices for the 16 coordinates in a 4×4 block. Denoting the left-upper corner coordinates as (x, y), the rest of the element can be denoted as $(x+i, y-j)$ with i, j = 0, 1, 2 and 3. So to get $a_1(p)x_i$ it could use $a_1(p)x$ to add with 0, $a_1(p)$, $2\times a_1(p)$ or $3\times a_1(p)$. To get $a_2(p)y_i$ it could use $a_2(p)y$ to subtract with 0, $a_2(p)$, $2\times a_2(p)$ or $3\times a_2(p)$. The different in these two calculations is to add or subtract retrievals with different recursive cycle. For calculation of $a_1(p)x_i$ the retrieval 0, $a_1(p)$, $2\times a_1(p)$ or $3\times a_1(p)$ is changed every cycle and recursively used for every four cycles. For calculation of $a_2(p)y_i$ the retrieval 0, $a_2(p)$, $2\times a_2(p)$ or $3\times a_2(p)$ is changed every four cycles. To get $a_3(p)x_iy_i$ the relationship between x_iy_i can also be used to reduce the computational amount. The signal x_iy_i in a 4×4 block could be replaced by $(x+i)(y-j) = xy+yi-xj-ij$ and so there are only three multiplications of $a_3(p)xy$, $a_3(p)x$ and $a_3(p)y$ needed

for the estimate $a_3^{(p)}x_iy_i$. In Table.2 it shows the computational amount needed for the calculation of $\mathbf{G}^T\mathbf{D}$ and new corresponding coordinates in a 4×4 block with or without using the coordinate relationship. According to the result in Table.2 it can be seen that the computational amount can be largely reduced according to the relationship of coordinate.

3.1.2 the architecture of block recursive algorithm

In above paragraph the reduction of computation amount according to the coordinate relationship is introduced. In this paragraph the corresponding architecture design of the quantized block recursive algorithm is derived. First the architecture for the calculation of $\mathbf{G}^T\mathbf{D}$ is proposed. The architecture is shown in Fig.4 to Fig.5. In Fig.4 the architectures named "COLUMN" and "ROW" are to get the column summation and row summation of signals. The architectures named "SUM_L" and "SUM_R" are mirrored architecture for the multiplication with x and y. In this figure "FF1" means the flip-flop with clock cycle of one time unit and "FF4" means the flip-flop with clock cycle of four time units. By combining the architecture of "COLUMN" and "ROW" with "SUM_L" and "SUM_R", the architecture of "A1" and "B1" are used to calculate u_1, u_2 and u_4. In Fig.5 the architecture of "C1" is for the calculation of part 3 of u_3. Also in this figure number 1 in the name of module means the clock cycle of one time unit. The architecture "E1" is used for the calculations of u_1, u_2, u_3 and u_4. In this figure "ROM x" and "ROM y" are the memories used to store the X directional coordinate and Y directional coordinate on the left-upper corner for all 4×4 blocks in a image. The architecture of "C4" is same as "C1" but with clock cycle of four time units. Using the same architecture of "E1" with input $D_{yl}^{(p)}$ the update term u_5, u_6, u_7 and u_8 could be calculated.

After this, the architecture for the estimate of new motion parameters is composed of adders and registers. This part is very simple and is not discussed here. Then the new corresponding coordinate of pixel is estimated. The architecture design is shown in Fig.6. In this figure the architecture "SG1" is to store the value of 0, $a_i^{(p)}$, $2\times a_i^{(p)}$ or $3\times a_i^{(p)}$ in shift register and then shift the data every one clock cycle. In "SG4" the data in shift register is shifted every four clock cycles. In architecture "D1" the shift register operation is combined with the multiplication with the coordinate on left-upper corner to complete the multiplication with x_i, y_i or x_iy_i and so the new corresponding coordinate of pixel could be estimated.

3.2 the architecture of block matching algorithm

After derived the architecture of the block recursive algorithm, the architecture design for the block matching algorithm is discussed. In the proposed combinative algorithm the block matching method is used after the estimation of the block recursive method has completed. Because the deformed block is used to find the best matched block the distribution of pixels of block may be irregular. And so the data access procedure must be arranged with new method. The estimation step for block matching algorithm is described in the following :

Step1 : According the motion parameters estimated by block recursive algorithm, the corresponding coordinate of pixel is calculated. To calculate the corresponding coordinate architecture "D1" derived in Fig.6 could be used but with the different clock cycle time.

Step2 : Using the coordinate calculated in step.1 as center the luminance of pixels in a predefined searching range are downloaded from memory and saved in registers. In our proposed algorithm the searching range is -8 to 8. And so in this step the 17×17 luminance of pixels are downloaded from memory.

Step3 : Estimate all the corresponding DFD of the 17×17 pixel and then add the absolute value of DFD with the old value of sum of DFD to get the new value of the sum of DFD respectively. After this, the new value of the sum of DFD is stored in register.

Step4 : Repeat step1 to step3 until all the pixels in a 16×16 block have been searched and all the sums of DFD have been calculated. Then Compare all the sums of DFD and find the best matched block and the corresponding motion vector dx and dy.

IV. Conclusion

The combinative motion estimation algorithm is shown to be very effective for the motion parameter estimation with complex motion phenomenon. The improvement is seen to be very significant as compared with the conventional block matching method. In order to reduce the

additional computation amount for the block recursive algorithm the corresponding architecture is proposed. By using the coordinate relationship the computational amount can be largely reduced to meet the requirement of real time estimation.

Reference

[1] Arun N. Netravali, Barry G. Haskell, "Digital Picture Representation and Compression", AT&T Bell Laboratories, New York, 1988.

[2] Georg Musmann, Peter Pirsch, Hans-joachim Grallert, "Advances in Picture Coding", Proceedings of IEEE, vol.73, No.4, April 1985.

[3] M. Hotter, "Differential Estimation of The Global Motion Parameters Zoom and Pan", Signal Processing, vol.16, no.3, March 1989, pp.249-265.

[4] Y. Nakaya and H. Harashima, "Motion Compensation Based on Spatial Transformations", IEEE Trans. on CAS for Video Technology, vol.4, no.3, Jun. 1994, pp-339-356.

[5] G. A. Clark, S. K. Mitra, S. R. Parker, "Block Implementation of Adaptive Digital Filter", IEEE Trans. Circuits Syst., vol. CAS-28, June 1981, pp.584-592.

[6] A. Gersho, "Adaptive filtering with Binary Reinforcement", IEEE Trans. Inform. Theory, vol.IT-30, March. 1984.

[7] W. A. Sethares and Richard Johnsen, "A Comparison of two Quantized State Adaptive Algorithms", IEEE Trans. ASSP, vol.37, No.1, Jan. 1989, pp.138-143.

[8] David G. Luenberger, "Linear and Nonlinear Programming", Addison-Wesley publishing company, 1984.

Block center position	Combinative	Block recursive	Block matching	Zooming parameter
(-240, 24)	5.13	7.84	5.25	0.994
(-224, 24)	6.98	9.57	7.05	0.999
(-208, 24)	6.19	7.33	7.13	1.000
(-192, 24)	6.70	6.70	7.41	0.991
(-176, 24)	7.01	7.01	10.23	1.056
(-160, 24)	6.81	6.81	26.91	1.154
(-144, 24)	10.84	14.07	48.83	1.224
(-128, 24)	12.78	12.78	79.00	0.429
(-112, 24)	18.59	21.48	40.76	0.731
(-96, 24)	19.26	33.46	34.11	1.463
(-80, 24)	29.50	56.47	66.64	1.310

Table.1 The mean absolute values of DFD of different block with different estimation algorithms and the corresponding zooming motion parameters.

Calculation	Without coordinate relationship	With coordinate relationship
G^7D	128 mults + 120 adds	8 mults + 104 adds
New address	128 mults + 96 adds	10 mults + 124 adds

Table.2 The comparison of computational amount with or without using the coordinate relationship. (mult : multiplication, add : addition)

$(x1, y1)$ (x, y)	$(x2, y2)$ $(x+1, y)$	$(x3, y3)$ $(x+2, y)$	$(x4, y4)$ $(x+3, y)$
$(x5, y5)$ $(x, y-1)$	$(x6, y6)$ $(x+1, y-1)$	$(x7, y7)$ $(x+2, y-1)$	$(x8, y8)$ $(x+3, y-1)$
$(x9, y9)$ $(x, y-2)$	$(x10, y10)$ $(x+1, y-2)$	$(x11, y11)$ $(x+2, y-2)$	$(x12, y12)$ $(x+3, y-2)$
$(x13, y13)$ $(x, y-3)$	$(x14, y14)$ $(x+1, y-3)$	$(x15, y15)$ $(x+2, y-3)$	$(x16, y16)$ $(x+3, y-3)$

Fig.1 The x_i and y_i coordinate relationship in a 4×4 block.

xy	$(x+1)y$	$(x+2)y$	$(x+3)y$
$x(y-1)$ $= xy$ $-x$	$(x+1)(y-1)$ $= (x+1)y$ $-x$ -1	$(x+2)(y-1)$ $= (x+2)y$ $-x$ -2	$(x+3)(y-1)$ $= (x+3)y$ $-x$ -3
$x(y-2)$ $= xy$ $-2x$	$(x+1)(y-2)$ $= (x+1)y$ $-2x$ -2	$(x+2)(y-2)$ $= (x+2)y$ $-2x$ -4	$(x+3)(y-2)$ $= (x+3)y$ $-2x$ -6
$x(y-3)$ $= xy$ $-3x$	$(x+1)(y-3)$ $= (x+1)y$ $-3x$ -3	$(x+2)(y-3)$ $= (x+2)y$ $-3x$ -6	$(x+3)(y-3)$ $= (x+3)y$ $-3x$ -9

Fig.2 The x_iy_i coordinate relationship in a 4×4 block.

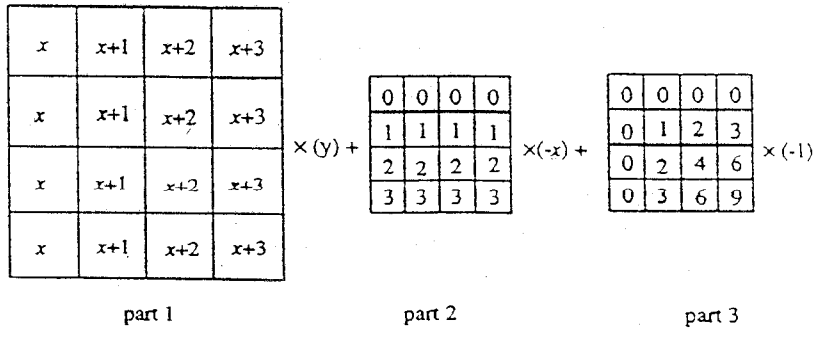

Fig.3 The three parts of decomposition of x_iy_i coordinate relationship.

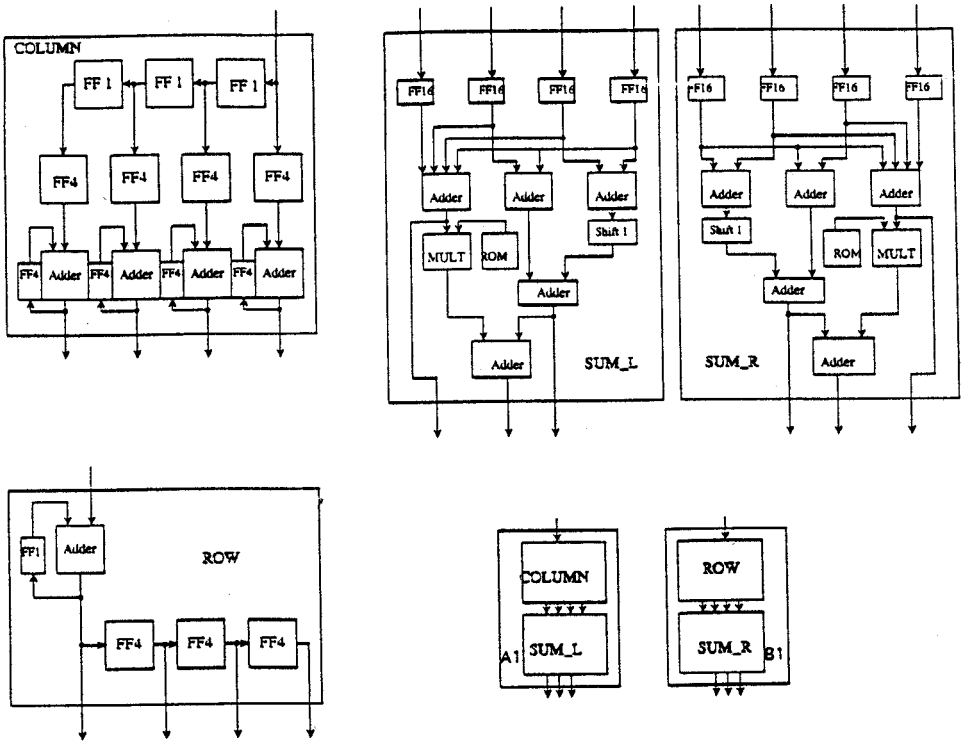

Fig.4 The architecture of module "COLUMN", "ROW", "SUM_L", "SUL_R", "A1" and "B1" for the calculation of u_1, u_3 and u_4.

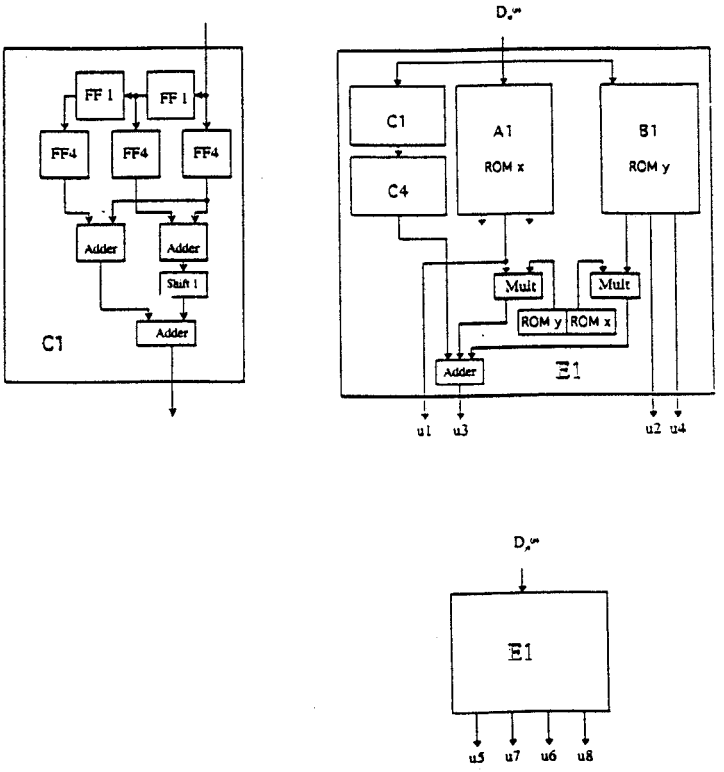

Fig.5 The architecture of module "C1" and "E1" for the calculation of $G^T D$.

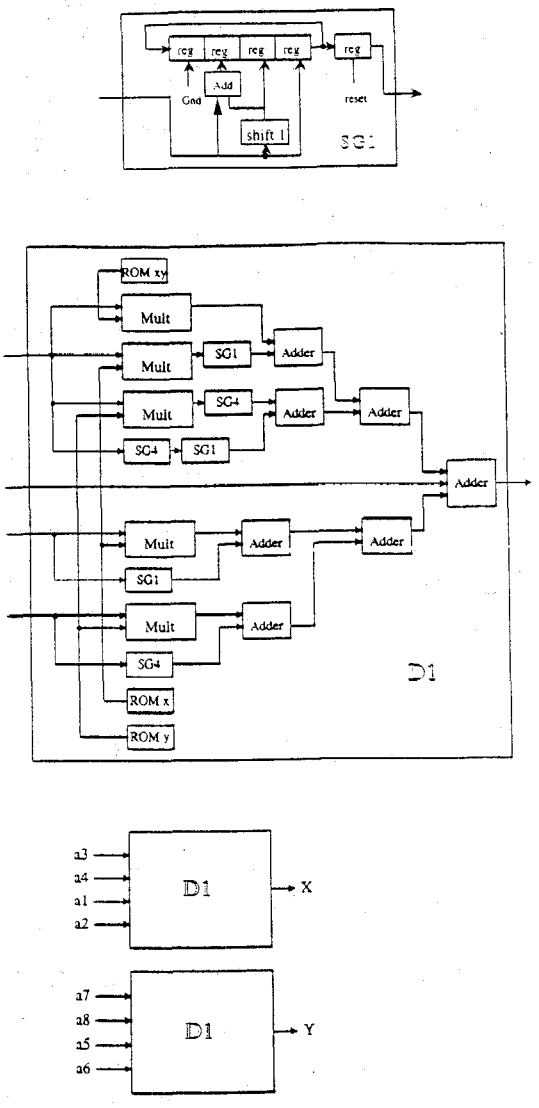

Fig.6 The architecture of module "SG1" and "D1" for the calculation of new corresponding coordinate.