고속 움직임 추정 알고리즘에 적합한 VLSI 구조 연구

李 在 憲, 羅 鍾 範
한국 과학 기술원 전기 및 전자공학과
대전 광역시 유성구 구성동 373-1

A VLSI Architecture for Fast Motion Estimation Algorithm

Jachun Lee and Jong Beom Ra
Dept. of Electrical Engineering, KAIST
373-1 Kusongdong, Yusonggu, Taejon, 305-701, Korea

요약

In this paper, we propose a VLSI architecture for implementing a recently proposed fast block matching algorithm, which is called the HSBMA3S. The proposed architecture consists of a systolic array based basic unit and two shift register arrays. And it covers a search range of -32~+31. By using a basic unit repeatedly, we can reduce the number of gates. To implement the basic unit, we can select one among various conventional systolic arrays by trading-off between speed and hardware cost. In this paper, the architecture for the basic unit is selected so that the hardware cost can be minimized. The proposed architecture is fast enough for low bit-rate applications (frame size of 352x288, 30 frames/sec) and can be implemented by less than 20,000 gates. Moreover, by simply modifying the basic unit, the architecture can be used for the higher bit-rate application of the frame size of 720x480 and 30 frames/sec.

I. 서론

움직임 추정 기법은 동영상 부호화에서 전체적인 부호화 성능과 시간에 영향을 미치는 중요한 부분이다. 움직임 추정 기법으로는 블록 정합 기법(Block Matching Algorithm: BMA)이 주로 이용된다. 블록 정합 기법의 대표적인 방법으로는 전역 탐색 기법(Full Search BMA)이 있다. 전역 탐색 기법은 성능은 최적이지만 계산량이 많아서 하드웨어로 실시간 처리가 가능하도록 구현하고자 할 때 칩 사이즈가 커지는 단점이 있다. 이러한 단점을 극복하고자 그동안 여러 고속 알고리즘들과 그 알고리즘에 적합한 VLSI 구조 또한 제안되어 왔다[1-6]. 하지만 제안된 고속 알고리즘들은 계산량을 줄이는 데에만 초점을 두어 하드웨어로 구현하기에 부적합 하다거나[6], 전역 탐색에 비해 성능이 떨어진다는 단점을 가지고 있다[7]. 하드웨어로 구현하기 용이한 대표적인 고속 알고리즘인 TSS(Three Step Search)의 경우 PSNR 이 전역 탐색 기법에 비해 약 1dB 정도 떨어진다[7].

이런 가운데 최근 블록 정합 알고리즘의 하나인 HSBMA3S 라는 새로운 고속 알고리즘이 제안되었다[8]. HSBMA3S 는 기본적으로 계층적 구조를 지니고 있으며, 다수의 MV 후보와 MV 들의 공간적 상관 관계를 사용함으로써 기존의 여러 고속 알고리즘들이 가지고 있던 국소적인 최소치(local minimum) 문제를 해결하여 전역 탐색 기법에 비해 계산량이 훨씬 적으면서 전역 탐색 기법의 성능을 거의 그대로 유지하는 특징을 가진다. MPEG-2 비디오 코덱에 적용했을 때 전역 탐색 기법에 비해 최악의 경우에도 0.2dB 정도만 떨어지는 성능을 보인다[8].

본 논문에서는 HSBMA3S 에 적합한 VLSI 구조를 제안한다. 적용 범위는 저 전송률(352x288, 30Hz)이고 탐색 영역의 크기는 수평/수직 -32~+31 화소로써 이동단말기에 적용하는 것을 궁극적인 목표로 두고 있다.

본 논문의 구성은 다음과 같다. 2 장에서 HSBMA3S 알고리즘과 제안한 VLSI 구조에 대해 설명하고, 3 장에서는 제안된 구조를 구현하였을 때의 결과를 전역 탐색 기법과 TSS 를 구현한 여러 다른 구조들과 서로 비교한다. 마지막으로 4 장에서는 결론을 맺는다.

II. 본문

1.HSBMA3S 알고리즘

HSBMA3S 알고리즘은 그림 1 에서 보는 바와 같이, 상위 레벨, 중간 레벨, 하위 레벨의 3 단계로 나누어져 있다.

1.1 상위 레벨

상위 레벨에서는 입력 영상으로부터 필터링되고 sub-

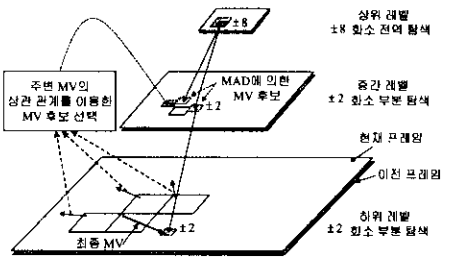


그림 1. HSBMA3S 개념도

sampling 된 가로, 세로 각각 1/4 크기인 영상을 이용하여 탐색을 수행한다. 가로, 세로 각각 1/4 씩 줄어든 4x4 크기의 블록 단위로 탐색 영역의 크기도 가로, 세로 각각 1/4 씩 줄인 수평/수직 ±8 화소 영역에 대해서 전역 탐색을 수행한다. 정합 기준으로는 MAD(Mean Absolute Difference)를 사용하며, 중간 레벨의 탐색을 위한 MV 후보로 MAD 가 최소인 두 점을 선택한다.

1.2 중간 레벨

중간 레벨에서는 가로, 세로 각각 입력 영상의 1/2 크기 영상을 이용하여 탐색을 수행한다. 중간 레벨에서는 총 3 개의 MV 후보에 대해서, 각각의 MV 후보를 초기점으로 8x8 크기의 블록 단위로 수평/수직 ±2 화소 크기의 부분 탐색을 수행한다.

여기서 사용되는 3 개의 MV 후보 중 두 개는 상위 레벨에서 MAD 를 정합 기준으로 하여 선택된 것들이고, 나머지 하나는 MV 들의 공간적 상관 관계를 이용하여 선택된 것이다. 3 개의 후보에 대해서 각각 탐색을 수행하여 그중에 최소 MAD 를 가지는 한 점만을 하위 레벨에서의 탐색 초기점으로 선택한다.

MV 들의 공간적 상관 관계를 이용하여 후보를 선택하는 방법으로는 하드웨어 구현이 용이하도록 HSBMA3S 에서 사용한 방법[8]과는 다른 방법을 사용한다. 그림 2 와 같이 시간적으로 우선 처리되어 이미 최종 MV 를 알고 있는 3 개의 MB 들의 MV 들의 x 성분과 y 성분을 각각 따로 중간값을 취해 하나의 후보를 결정한다. 수식으로 표현하면 다음과 같다.

$$MV_x = \text{Median}\{NMV0_x, NMV1_x, NMV2_x\}$$
$$MV_y = \text{Median}\{NMV0_y, NMV1_y, NMV2_y\}$$

이를 위해서 22 개의 shift register 와 comparator 가 필요하다. 이렇게 얻어진 MV 후보를 1/2 로 크기를 줄여 중간 레벨에서의 탐색 초기점 중 하나로 사용한다.

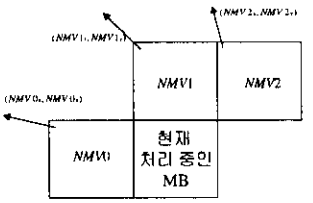


그림 2. 공간적 상관 관계를 이용한 MV 후보 선정에 사용되는 주변 MB 들

1.3 하위 레벨

하위 레벨에서는 입력 영상을 그대로 이용하여 중간 레벨에서 얻어진 점을 중심으로 16x16 크기의 MB 단위로 수평/수직 ±2 화소의 부분 탐색을 수행하여 최종 MV 를 하나 결정하게 된다.

2. 제안한 구조

다수의 PE(processing element)로 이루어진 systolic array 구조는 움직임 추정 알고리즘을 구현하는데 적합하다. 이 논문에서 제안한 구조도 systolic array 에 바탕을 두었다. Systolic array 를 사용할 경우 PE 의 개수는 일반적으로 MB 의 크기나 탐색 영역의 크기와 비례하게 된다.

HSBMA3S 는 3 개의 레벨이 있고 각각의 레벨마다 처리하는 블록의 크기와 탐색 영역의 크기가 다르므로 서로 다른 3 개의 systolic array 가 필요하다. 본 논문에서는 하나의 탐색 기본 단위를 두어서, 모든 레벨에서 순차적으로 사용할 수 있도록 하여 PE 의 개수를 최소화 하는데 중점을 두었다. 즉, 처리 블록의 크기는 각 레벨에서 처리하는 블록 크기 중 가장 작은 4x4 크기를 선택하고 탐색 영역의 크기는 각 레벨의 탐색 영역 중 가장 작은 수평/수직 ±2 화소(25 개의 탐색점)를 선택하여 이를 처리할 수 있는 하나의 systolic array 를 탐색 기본 단위로 두었다. 모든 레벨에서 이 탐색 기본 단위를 순차적으로 반복 사용함으로써 탐색을 수행할 수 있게 된다.

2.1 탐색 기본 단위

본 논문에서는 [9]에서 제안된 구조를 약간 변경하여 탐색 기본 단위의 구조를 결정하였다. 탐색 기본 단위는 그림 3 에서 보는 바와 같이 5 개의 PE 와 flip-flop, mux, 그리고 mux 를 제어하는 간단한 로직 등으로 이루어져 있다.

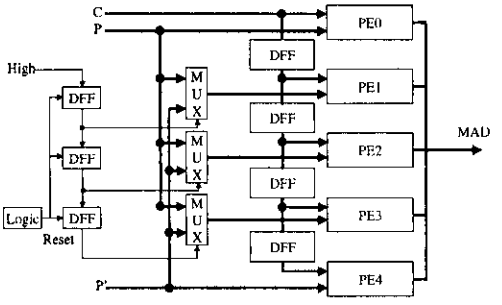


그림 3. 탐색 기본 단위의 구조

2.2 탐색 기본 단위의 적용

2.2.1 상위 레벨에의 적용

탐색 기본 단위의 효과적인 적용을 위해서 탐색 영역을 수평/수직 ±7 화소로 약간 수정하였다. 그림 4 에서와 같이 수평/수직 ±7 화소 225 개의 탐색점을 포함하

는 영역을 가로, 세로 각각 3 등분하여 총 9 개의 영역으로 나눈다. 각각의 나누어진 영역은 가로, 세로 5 개씩 총 25 개의 탐색점들을 갖게 되어 탐색 기본 단위로 한번에 처리할 수 있는 크기가 된다.

이렇게 나누어진 영역에 대해서 차례대로 탐색 기본 단위를 이용해서 9 번 처리하면 모든 탐색 영역내의 탐색점들에 대한 탐색을 수행할 수 있다. 탐색 기본 단위로 25 개의 탐색점들을 처리하는데 총 84 사이클이 소요되고 상위 레벨을 처리하는 데는 모두 724 사이클이 소요된다.

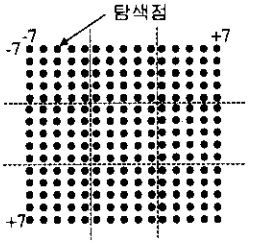


그림 4. 탐색 기본 단위의 상위 레벨에의 적용

2.2.2 중간 레벨에의 적용

그림 5 와 같이 8x8 크기의 블록을 4x4 크기의 블록으로 4 등분하여 탐색 기본 단위를 이용하여 4 번 나누어 처리함으로써 탐색을 마칠 수 있다. 이때 탐색 기본 단위에서 얻어지는 MAD 는 8x8 크기의 블록에 대한 완전한 MAD 가 아니라 일부분인 4x4 크기의 블록에 대한 부분 MAD 가 된다. 한 탐색점에 대한 MAD 는 탐색 기본 단위를 4 번 사용하여 얻은 부분 MAD 를 더하여 얻을 수 있다.

MV 후보가 모두 3 개이므로 탐색 기본 단위를 총 12 번 사용함으로써 탐색을 마치게 된다. 중간 레벨의 탐색을 수행하는 데는 총 964 사이클이 소요된다.

3 개의 후보 중 주변 MV 들의 상관 관계를 이용한 MV 후보에 대해 먼저 탐색을 수행하면 상위 레벨과 중간 레벨사이의 레벨간의 의존성(inter level dependency)에 의한 시간 지연을 피할 수 있다.

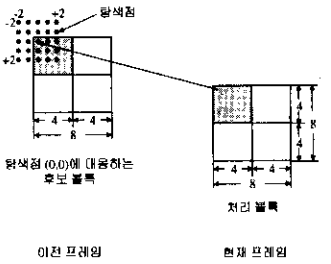


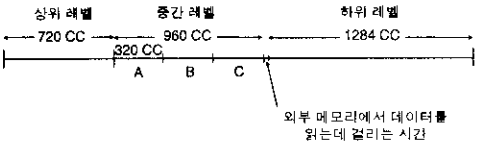
그림 5. 탐색 기본 단위의 중간 레벨에의 적용

2.2.3 하위 레벨에의 적용

중간 레벨과 동일한 방식으로 4x4 크기의 블록으로 모두 16 개의 블록으로 나누어 순차적으로 처리한다. 중간 레벨에서와 마찬가지로 탐색 기본 단위로부터 나오는 부분 MAD 들을 더하여 완전한 MAD 를 구하는 과정이 필요하다. 이를 위해서 25 개의 shift register 와 adder 가 필요하다. 하위 레벨에서 소요되는 시간은 총 1,284 사이클이다.

이상과 같은 방법으로 탐색 기본 단위를 적용하면 한 MB 을 처리하는데 총 2989 클럭 사이클이 소요된다.

그림 6 은 한 MB 을 처리하는 동안의 타이밍을 나타낸 것이다. 지금까지 살펴본 움직임 추정 칩의 개념적인 전체 블록 도는 그림 7 과 같다. 그림 8 은 탐색 기본 단위가 처리하는 기준 블록과 탐색 영역 데이터를 나타낸 것이고 표 2 는 PE 에서의 데이터 흐름도이다.



A. 주변 MV를 이용한 후보 처리
B. 최소 MAD를 가지는 후보 처리
C. 두 번째 최소 MAD를 가지는 후보 처리

그림 6. 한 MB 처리 시 타이밍도

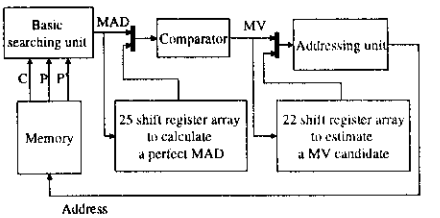


그림 7. 움직임 추정 칩의 개념적인 전체 블록도

III. 구현 결과 및 고찰

본 논문에서 제안한 구조를 VHDL 을 통하여 기술하고 시뮬레이션한 뒤 합성한 결과 20,000 게이트 이하로 구현 가능하였다. 표 1 은 제안한 구조와 다른 구조들을 비교한 것이다. 제안한 구조는 탐색 영역이 수평/수직 -32~+31 화소이고 다른 구조들은 모두 수평/수직 ±7 화소이다. 더 넓은 탐색 영역에 대해서 탐색을 수행함에도 불구하고, 전역 탐색에 비해서는 약 1/8 정도의 게이트 수를 가지며, 더 좋은 성능을 나타내면서도 TSS 보다는 더 적거나 비슷한 게이트 수를 가진다. ETSS[10]는 탐색점을 (0,0) 주위에 집중시켰기 때문에 움직임이 다소 큰 영상에 적용하기에는 부적합하다.

IV. 결론

본 논문에서는 탐색 기본 단위를 두고 이를 반복적으로 사용하는 구조로써 HSBMA3S 에 적합하며 systo-

표 1. 다른 구조들과의 비교(제안된 구조는 수평/수직 -32~+31 화소의 탐색 영역을 가지고 나머지는 수평/수직 ±7 화소의 탐색 영역을 가짐)

	2-D array FS[2]	TSS[3]	TSS[5]	ETSS[10]	Proposed
No. of PE's	64	48	9	3	5
Extended data access/block	768	2976	1280	1024	868
Clock cycles	1024	230	794	2304	2989
No. of Gates(K)	136.3	32.9	36.6	17.19	17.3

lic array의 크기를 줄일 수 있는 VLSI 구조를 제안하였다. 탐색 기본 단위로는 기존의 여러 전역 탐색을 수행하는 구조를 그대로 적용할 수 있으며, 구조의 선택에 따라 속도와 게이트 수 사이의 절충이 가능하다. 본 논문에서는 저 전송률의 영상에 적용하기 위해 5개의 PE로 이루어진 구조를 탐색 기본 단위로 선택하였다. 선택한 구조는 간단하고 PE의 효율이 거의 100%에 가깝고 데이터 I/O 핀 수가 적은 장점이 있다. 제안한 구조를 VHDL을 사용하여 기술하고 합성한 결과 총 20,000 게이트 이내로 움직임 추정 칩의 구현이 가능하였다. 제안된 구조는 352x288 크기의 영상, 탐색 영역 수평/수직 -32~+31 화소에 대해서 30Hz까지 실시간 처리가 가능하다.

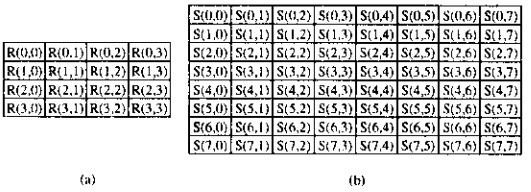


그림 8. (a) 기준 블록 데이터 (b) 탐색 영역 데이터

표 2. PE의 데이터 흐름도

	PE0		PE1		PE2		PE3		PE4	
	기준 블록	탐색 영역	기준 블록	탐색 영역	기준 블록	탐색 영역	기준 블록	탐색 영역	기준 블록	탐색 영역
0	R(0,0)	S(0,0)								
1	R(0,1)	S(0,1)	R(0,0)	S(0,1)						
2	R(0,2)	S(0,2)	R(0,1)	S(0,2)	R(0,0)	S(0,2)				
3	R(0,3)	S(0,3)	R(0,2)	S(0,3)	R(0,1)	S(0,3)	R(0,0)	S(0,3)		
4	R(1,0)	S(1,0)	R(0,3)	S(0,4)	R(0,2)	S(0,4)	R(0,1)	S(0,4)	R(0,0)	S(0,4)
5	R(1,1)	S(1,1)	R(1,0)	S(1,1)	R(0,3)	S(0,5)	R(0,2)	S(0,5)	R(0,1)	S(0,5)
6	R(1,2)	S(1,2)	R(1,1)	S(1,2)	R(1,0)	S(1,2)	R(0,3)	S(0,6)	R(0,2)	S(0,6)
7	R(1,3)	S(1,3)	R(1,2)	S(1,3)	R(1,1)	S(1,3)	R(1,0)	S(1,3)	R(0,3)	S(0,7)
8	R(2,0)	S(2,0)	R(1,3)	S(1,4)	R(1,2)	S(1,4)	R(1,1)	S(1,4)	R(1,0)	S(1,4)
9	R(2,1)	S(2,1)	R(2,0)	S(2,1)	R(1,3)	S(1,5)	R(1,2)	S(1,5)	R(1,1)	S(1,5)

참고 문헌

[1] B.M. Wang, J.C. Yen, and S. Chang, "Zero Waiting-Cycle Hierarchical Block Matching Algorithm and its Array Architectures," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 4, no. 1, pp. 18-27, Feb. 1994.

[2] L.D. Vos, "VLSI-architectures for the hierarchical block-matching algorithm for HDTV applications," *in SPIE*, vol. 1360 (*Visual Communications and Image Processing*), pp. 398-409, 1990.

[3] Y.S. Jehng, L.G. Chen, and T.D. Chiueh, "An Efficient and Simple VLSI Tree Architecture for Motion Estimation Algorithms," *IEEE Trans. on Signal Processing*, vol. 41, no. 2, pp. 889-900, Feb. 1993.

[4] G. Gupta and C. Chakrabarti, "Architectures for Hierarchical and Other Block Matching Algorithms," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 477-489, Dec. 1995.

[5] H. M. Jong, L. G. Chen, and T. D. Chiueh, "Parallel Architectures for 3-Step Hierarchical Search Block-Matching Algorithm," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 407-415, Aug. 1994.

[6] J. Chalidabhongse and C.-C. J. Kuo, "Fast Motion Vector Estimation Using Multiresolution-Spatio-Temporal Correlations," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 7, no. 3, pp. 477-488, June 1997.

[7] S.C. Cheng and H.M. Hang, "A Comparison of Block-Matching Algorithms Mapped to Systolic-Array Implementation," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 7, no. 5, pp. 741-757, Oct. 1997.

[8] K.W. Lim and J.B. Ra, "Improved Hierarchical Search Block Matching Algorithm by Using Multiple Motion Vector Candidates," *IEE Electronics letters*, pp. 1771-1772, Oct. 1997

[9] K.M. Yang, M.T. Sun and L. Wu, "A Family of VLSI Designs for the Motion Compensation Block-Matching Algorithm," *IEEE Trans. on Circuits and Systems*, vol. 36, no. 10, pp. 1317-1325, Oct. 1989

[10] P. Lakamsani, B. Zeng, and M. Liou, "An Enhanced Three Step Search Motion Estimation Method and its VLSI Architecture," *IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 754-757, May 1996.

[11] C.H. Hsieh and T.P. Lin, "VLSI Architecture for Block-Matching Motion Estimation Algorithm," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 2, no. 2, pp. 169-175, June 1992.

[12] C.Y. Lee, "A Novel VLSI Architecture for Block Matching Algorithms," *in Proc. of ICASSP'95*, vol. 3, no. 5, pp. 639-651, Sept. 1994.