연속제거알고리즘 기반의 고속 움직임 추정 프로그램 성능평가

김경현, 손승일

한신대학교 정보통신학과

Implementation of Fast Motion Estimation Program Based on Successive Elimination Algorithm

Kyung Hyun Kim, Seung Il Sonh

Dept. of Information and Communication HanShin University

E-mail : 52103kkh@hanmail.net

요 약

오늘날 컴퓨터와 데이터 통신의 급속한 발달로 인해 멀티미디어 정보통신 기술이 비약적으로 발전하고 있다. 이러한 멀티미디어 데이터 중에서 동영상은 다른 데이터 형태에 비해 정보량이 매우 방대하다. 따라서 동영상을 처리하는 시스템에서는 압축 기법이 매우 중요한 역할을 차지한다. 이에 본 논문은 연속제거 알고리즘을 기반으로 이전블록 초기 움직임 벡터 사용 및 strip단위 블록 합을 통하여 고속의 움직임 추정을 통해 영상을 복호화 하였고, 기존의 완전탐색 블록 정합방식과 영상 복원 능력 및 연산량을 비교 평가하였다. 뿐만 아니라 이후 이를 바탕으로 고속 움직임 추정 모듈을 VHDL로 구현하여 본 논문의 프로그램을 성능평가의 기준으로 이용할 것이다.

키워드

SEA, 움직임 추정, Strip

Ⅰ. 서 론

오늘날 컴퓨터와 데이터 통신의 급속한 발달로 인해 멀티미디어 정보통신 기술이 비약적으로 발전하고 있다. 이러한 멀티미디어 데이터 중에서 동영상은 다른 데이터 형태에 비해 정보량이 매우 방대하다. 따라서 동영상을 처리하는 시스템에서는 압축 기법이 매우 중요한 역할을 차지한다. 압축방식 중 손실압축은 사람이 직관적으로 느낄 수 있는 화질에 큰 영향을 주지 않으면서 시간적 및 공간적 중복성을 제거하는 방법이다. 기존의 H.264의 동영상 압축 방식에서는 완전탐색 블록 정합 알고리즘을 통하여 인코딩 한다. 이러한 과정은 최적의 움직임 값을 얻을 수 있지만 구현시 많은 연산과정과 시간의 지연이 생기게 된다. 이에 본 논문은 움직임 추정시 공간적으로 인접해 있는 블록간 높은 상관관계가 존재함을 바탕으로 효과적인 움직임 추정 및 완전탐색 블록정합 알고리즘의 계산적 변형을 통해 연산량을 감소시키는 연속제거 알고리즘 기반의 프로그램을 성능평가 한다. 뿐만 아니라 이후 이를 바탕으로 고속 움직임 추정 VHDL 모듈을 구현하여 본 논문의 프로그램을 성능평가 기준으로 이용할 것이다.

Ⅱ. 알고리즘

2-1. 고속 움직임 추정

고속 움직임 추정을 위한 방법은 크게 두 가지 방식으로 나뉜다. 첫째로 움직임 추정시 사용되는 후보벡터의 수를 줄임으로써 계산량을 줄이는 방법으로 이러한 방식을 택한 알고리즘으로 2-D Logarithm 탐색법, 3단계 탐색법, conjugate direct 탐색법, cross 탐색법, 4단계 탐색법, Diamond 탐색법이 있다. 두 번째로 모든 후보벡터는 검색을 하되 SAD계산 자체의 계산량을 줄이는 방법으로 SEA(Successive Elimination Algorithm), MSEA (Multi-level SEA), PDE (Partial Difference Elimination algorithm)의 알고리즘이 제안되었다. 두 가지 고속 움직임 추정 방식 중 첫 번째 방식은 단순히 확률적 예측에 따라 탐색지점의 수를 줄이기 때문에 움직임 예측

의 영상 복원력이 기존방식에 비해 현저히 떨어지게 되었다. 반면 두 번째 방식은 모든 후보영역에 대한 탐색을 수행하며 이때 발생되는 중복 연산을 제거함으로써 기존의 방식과 거의 동일한 영상 복원력을 발휘한다[2][3].

2-2. 연속제거 알고리즘

간단히 SEA이라 부르며 고속움직임 추정방식 중 SAD의 계산량을 줄인 방식의 알고리즘들의 기본이 되는 원형 알고리즘이다. MSEA 나 PDE 방식에 비해 성능 면에서 약간 뒤떨어지는 단점을 가지고 있기는 하지만 이러한 방식의 알고리즘 중 비교적 간단하고 규칙적 연산의 알고리즘 방식으로 하드웨어 설계에 적합하다 판단하여 이번 논문에서 채택한 방법이다[4][5].

III. 매크로 블록의 고속 움직임 추정

3-1 효율적 블록합의 처리

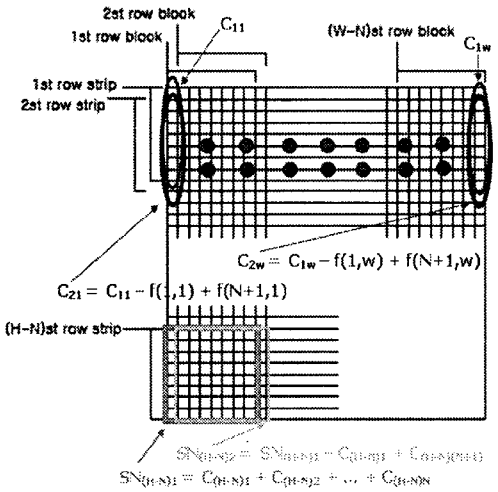

그림 1. strip단위 블록합

하나의 블록 또는 하나의 영상의 움직임을 추정하는 데는 영상 내에서 (N*N) 크기의 블록으로 나누고 블록 내에서도 픽셀단위로 나누어 처리하게 된다. 이때 현재 프레임의 블록과 이전 프레임의 후보블록간의 정합도를 찾는 과정에서 최초 후보블록의 데이터를 얻어오는 과정을 제외하고 나머지 후보블록은 규칙적으로 위치를 이동시키며 모든 탐색 범위에 대하여 정합도를 조사하게 되는데 이때 중복되는 부분의 데이터를 매번 새로 얻어오는 일은 시스템의 성능을 저하시키는 요인이 된다. 이에 그림 1과 같이 strip 단위의 블록 합을 구하는 방법을 사용하여 픽셀 값을 row strip 단위로 묶어 각각의 column을 계산하고, C_{11}, C_{12}, …, C_{1w}로 저장한다. 두 번째 row strip은 기존의 row strip 데이터에서 새로 추가된 f(N+1,w)값은 더하고 제외된 f(1,w)를 뺌으로써 규칙적인 이동에서 몇 개의 데이터 이동만으로 재사용이 가능하다. 이러한 row strip의 묶음은 하나의 블록을 구성하여 블록 전체의 연산에 대하여 효율적으로 사용된다[5][6][7].

3-2 고속 SAD 계산

최초 블록을 제외한 나머지 블록은 기준이 되는 norm블록의 값을 기준으로 탐색을 시작한다. norm 블록은 SEA의 성능을 좌우하는 가장 중요한 값으로 기존의 SAD를 구하는 방식과 달리 (식 1)은 현재 프레임의 $F_{S\times T}$블록의 합계와 미리 계산된 (m,n)의 초기 움직임 벡터에서의 $SAD_{S\times T}$ (m,n)값인 norm 값의 절대치 합의 범위 내에 후보블록 $R_{S\times T}(x,y)$벡터에서의 블록합계 값이 존재할 경우에만 계산이 이루어져 새로운 $SAD_{S\times T}(x,y)$ 의 값으로 대체되게 된다[8][9]

$$F_{S\times T} - SAD_{S\times T}(m,n) \le R_{s\times T}(x,y) \le F_{S\times T} + SAD_{S\times T}(m,n) \quad \text{(식 1)}$$

3-3 초기 움직임 벡터

고속 SAD를 가능하도록 할 수 있는 것은 얼마큼 정합도가 높은 초기 움직임 벡터를 택하는가에 달려있다. 극단적인 예로 가장 최적의 움직임 벡터를 찾아냈을 경우 norm 블록은 최소의 SAD값을 가지므로 새로운 값으로 대치될 때 이루어지는 연산이 불필요 하게 된다. 반대로 최악의 위치를 찾아냈을 경우 완전탐색 블록정합과 마찬가지의 연산이 이루어지게 된다. 기존의 완전탐색 블록정합이 초기 움직임 벡터를 (0,0)으로 놓는 이유는 움직임 벡터의 일반적인 특성상 이 벡터 값에 분포할 확률이 높기 때문에 나온 방식이지만 이는 움직임이 큰 동영상 시퀀스에는 적합하지 않는 방식이다. 이에 본 논문에서는 인접 블록 간에 움직임은 서로 상관관계가 높다는 점을 착안 그림 2와 같이 이전 블록의 움직임 벡터를 다음 블록의 움직임 벡터로 사용 하는 방식을 택하였다[9][10].

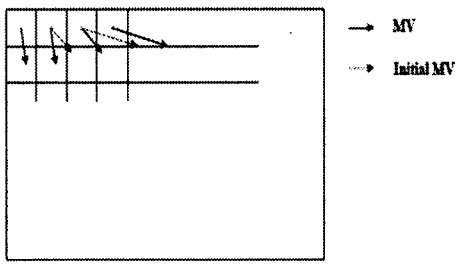

그림 2. 초기 움직임벡터 추출

IV. 실험 결과 및 분석

4-1 프로그램 순서도

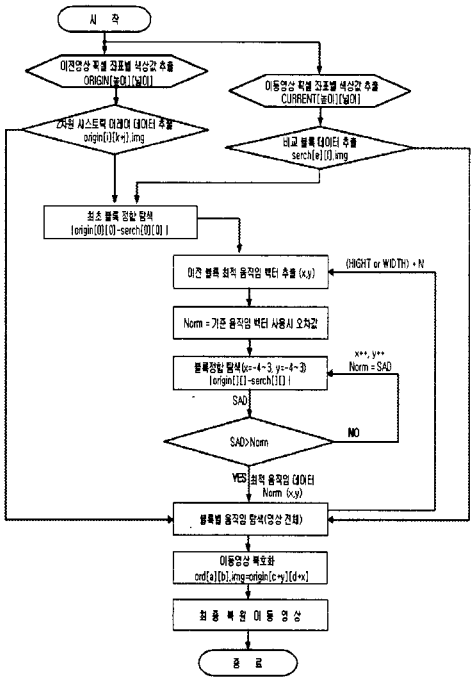

그림 3. 연속제거 알고리즘 프로그램 순서도

그림 3은 연속제거 알고리즘을 기반으로 한 프로그램을 통하여 영상을 복원하는 순서도로써 최초 두 영상을 입력 받아 각각 후보 블록과 현재 블록의 데이터를 추출하여 최초블록은 일반 블록 정합 방식과 동일하게 최적의 움직임 벡터를 탐색하게 되고, 이후 블록 연산의 초기 움직임 벡터로 사용된다. 초기 움직임 벡터를 통해 기준 norm 값을 도출하고 이때 얻어지는 값과 연산되어지는 후보블록의 SAD값과의 크기를 비교 최적의 움직임을 찾는 방식으로 모든 영상 내 블록에 대한 탐색을 거친다. 새로운 복원 영상은 이전 영상에서 블록별 벡터 값에 따른 이동만을 통하여 새로운 영상을 복원한다.

4-2 영상 복원력

그림 4는 기존의 완전탐색 블록정합 방식을 이용하여 복원한 영상이고 그림 5는 본 논문에서 제안한 연속제거 알고리즘 기반의 프로그램을 통하여 복원된 영상이다. 이 두 영상은 육안으로 쉽게 구분될 정도의 영상의 차이는 보이지 않았고 프레임에 따라 정합도가 낮은 벡터 값을 선택함으로써 특정 블록에 잘못된 복호화가 되는 경우가 있었다.

그림 4. 완전탐색 정합 복원영상

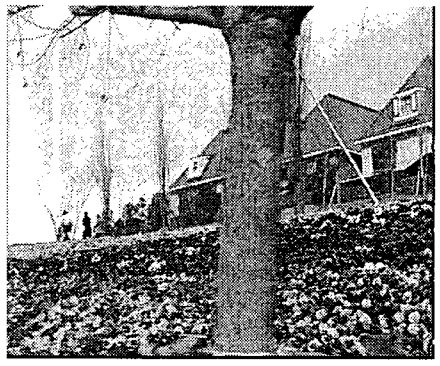

그림 5. 연속제거 알고리즘 복원영상

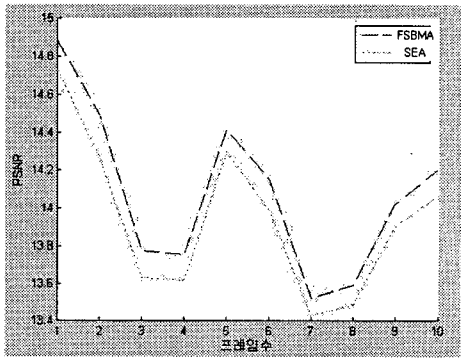

그림 6. PSNR

영상간의 정확한 객관적 비교를 위하여 그림 6에서와 같이 PSNR값을 통하여 10개 프레임 영상에 대하여 비교한 결과 약 10~15%가량의 영상 복원력의 정확도가 떨어지게 되었다. 이 결과는 기존의 SEA알고리즘에서 제시된 90%의 정확도와 유사한 수치를 나타내는 것으로 오차가 발생되는 영역의 공통점은 협소한 영역의 개별 움직임이나 기존에 없었던 영상의 영역 복원시 나타나는 오차였다. 이는 이전블록의 최적의 정합도를 가진 것으로 판단된 벡터 값을 다음 블록에서 초기 움직임 벡터로 사용함에 따라 발생된 오차이다.

4-3 연산 횟수 비교

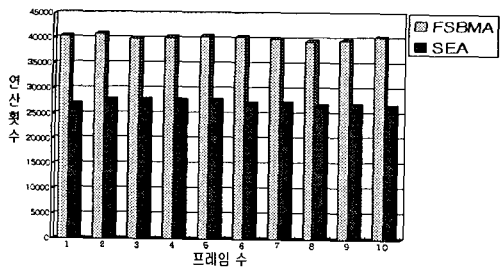

그림 7. 연산횟수 비교

그림 7은 최초의 SAD값에서 새로운 SAD값으로 대체되는 연산의 횟수를 비교한 것으로 기존의 완전탐색 블록정합방식에 비하여 연속제거 알고리즘이 연산의 능률면에서 약 30~35% 가량의 감소효과를 가져오게 되었다. 이는 이전 블록 움직임 벡터를 기준으로 한 norm값을 기준으로 하여 연산을 함에 따라 후보블록의 탐색점의 수를 줄이게 되는 효과를 거두었다.

V. 결론 및 향후 연구과제

연속제거 알고리즘 기반의 고속 움직임 추정을 프로그램으로 구현함에 따라 기존의 완전탐색 블록 정합 방식으로 영상을 복호화 하는 방식에 비해 약10~15% 가량의 영상 복원 능력은 감소하게 되었지만 30~35%가량의 연산량의 감소를 가져오게 되는 결과를 얻게 되었다. 이는 H.264가 상용화 된 DMB영상에서 문제점으로 지적되고 있는 방대한 연산에 따른 인코딩 시간의 지연에 대한 해결 방안을 제시하게 되었을 뿐 아니라 저전력 하드웨어 설계에 적합하다. 현재 DMB방송은 핸드폰이나 PMP 등의 다른 영상 매체에 비하여 비교적 작은 영상 크기의 화면을 통하여 제공되고 있어 어느정도의 화질저하는 감소하고 보다 많은 연산량 감소를 통해 한번에 더욱 많은 데이터 정보를 보내는 것을 목표로 하고 있다. 하지만 향후 화질의 저하에 대한 문제는 좀 더 연구를 통한 성능 향상이 필요로 하다. 또한 본 논문의 결과가 소프트웨어로써만 제안되고 연구가 중단되는 것이 아닌 이를 바탕으로 고속 움직임 예측 모듈을 VHDL로 구현하여 본 논문의 프로그램을 성능평가의 기준으로 사용할 것이다.

참고문헌

[1] 후지와라 히로시, 정제창 역, "최신 MPEG", 교보문고 1995.

[2] Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, JVT-G050r1, 2003.

[3] 이호석 · 김준기, "알기 쉬운 MPEG-2: 소스코드 해설", 홍릉과학 출판사, 2002.

[4] Iain E.G. Richardson, "H.264 and MPEG-4", WILEY, 2003.

[5] 김영문, "Fast Motion Estimation Algorithm for Multiple Reference Images Based on Elimination Algorithm", 중앙대 첨단영상대학원, 2004.

[6] Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, JVT-I020, 2003.

[8] Fernando Pereira·Touradj Ebrahimi, The MPEG-4 Book, 2002.

[9] John Watkinson, The MPEG Handbook, Focal Press, 2001.

[10] Hee-Soon Kim, Seunghwan Kim, and Yo-Sung Ho, "Fast Mode Decision Algorithm Using Mode Classification for H.264", ICT 2004, No. 18-19, 2004