# MCFI 구현을 위한 형태 기반 움직임 예측에 관한 연구

정회원 박 주 현*, 김 영 철*, 홍 성 훈*

# A Study on the Shape-Based Motion Estimation For MCFI

Ju Hyun Park, Young-Chul Kim*, Sung-Hoon Hong** *Regular Members*

요 약

DTV의 Full HD급이 보편화되면서 LCD(Liquid Crystal Display)기반 대형화와 고화질 요구가 급속도로 증대되고 있다. 이에 LCD의 잔상효과 제거와 격동적인 화면에서의 고화질 구현을 위해 수신 단에서 후처리 과정으로 움직임 보상 기반 프레임 보간법(MCFI)이 사용되고 있다. 이때 움직임 예측 시 구현의 용이성 때문에 블록정합 기법(BMA)이 가장 많이 사용되고 있지만 여러 가지 문제점을 보여주고 있다. 본 논문에서는 움직임 정보의 정확성을 높이고 움직임 예측의 계산량 감소를 위하여 움직임 형태 기반의 움직임 예측 기법을 제안 한다. 움직임 정보의 정확성 향상을 위하여 움직임 형태를 기반으로 움직임 예측을 수행한다. 그리고 계산양 감소를 위하여 움직임 영역에서만 예측을 수행하는데 제안한 방법은 전 영역 움직임 예측기의 평균 25% 계산양만으로 비슷한 성능의 결과를 보여주고 있으며 특히 빠른 움직임 영역에서는 다른 여러 가지 움직임 예측 기법들 보다 우월한 성능을 보여준다.

**Key Words** : MCFI, BMA, Shape-based Motion Estimation

ABSTRACT

Motion Compensated Frame Interpolation(MCFI) has been used to reduce motion jerkiness for dynamic scenes and motion blurriness for LCD-panel display as post processing for large screen and full HD(high definition) display. Conventionally, block matching algorithms (BMA) are widely used to do motion estimation for simplicity of implementation. However, there are still several drawbacks. So in this paper, we propose a novel shape-based ME algorithm to increase accuracy and reduce ME computational cost. To increase ME accuracy, we do motion estimation based on shape of moving objects. And only moving areas are included for motion estimation to reduce computational cost. The results show that the computational cost is 25 % lower than full search BMA, while the performance is similar or is better, especially in the fast moving region.

## Ⅰ. 서 론

현재 DTV는 HD급에서 Full HD급으로 변화하고 있으며 이와 함께 대형화와 고화질이 요구되고 있다. 특히 스포츠 같은 격동적인 화면에서의 고화질을 구현하기 위해 수신 단에서 후처리 과정으로 Frame Rate Up-Conversion (FRUC)이 사용되고 있다. 전통적인 FRUC는 가장 단순한 방법으로 프레임 반복이나 선형 프레임 보간법이 사용되었다. 그러나 이러한 방법은 움직임 정보를 고려하지 않으므로 블러링이나 고스트 현상을 피할 수 없다. 이에 움직임 정보를 고려한 움직임 보상 기반 프레임 보간(MCFI)기술이 제

※ 본 연구는 교육과학기술부와 한국산업기술진흥원의 지역혁신인력양성사업으로 수행된 연구결과임
* 전남대학교 전자컴퓨터공학부(svelo2000@yahoo.co.kr, hsh@chonnam.ac.kr)
논문번호 : KICS2009-11-542, 접수일자 : 2009년 11월 2일, 최종논문접수일자 : 2010년 3월 8일

안되었으며 이 기술은 지난 수년간 많은 주목을 받고 있다[1]. MCFI는 시간적인 관점에서 해상도를 60Hz에서 120Hz로 개선시키므로 LCD 디스플레이에서 움직임 번짐 현상(motion blurriness)이나 움직임 요동(motion jerkiness) 현상을 줄일 수 있다. 이러한 MCFI 구현을 위해서는 움직임 예측기를 이용하여 인접한 두 프레임 사이에서 움직임 정보 추출이 필요하다[3].

일반적인 MCFI 기법은 현재 프레임과 이전 프레임 사이에서 움직임 예측을 수행하고 얻어진 움직임 벡터의 중간 값을 양 방향 프레임에 적용하여 중간 프레임을 생성한다. 그러나 움직임 정보 개선 기능(refinement)이나 부드러움 기능(smoothing)과 같은 후처리 과정 없이 움직임 벡터를 바로 사용하는 것은 블록화 현상이나 고스트 현상과 같은 화질 열화현상을 초래한다. 그 결과 측정되어진 움직임 벡터의 정확성 개선을 위하여 많은 연구가 진행되고 있는데 이는 보간 될 프레임에 관한 화소 정보사용이 불가능하기 때문에 움직임 벡터가 정확하면 할수록 더 좋은 화질의 프레임 보간이 가능하기 때문이다.

일반적으로, 블록정합 알고리즘(BMA)은 하드웨어 구현의 용이성과 단순성 때문에 움직임 벡터 측정에 가장 널리 사용되고 있다. 그러나 블록 정합 알고리즘은 현재 프레임의 블록과 이전 프레임의 블록 간의 절대 차이 합(SAD)을 계산하여 SAD 값이 가장 적은 블록을 선택하고 그때의 변위를 움직임 벡터로 선택하기 때문에 실제 움직임 정보와 다를 수 있다. 그 결과 잘못 계산 된 움직임 벡터를 직접 사용하여 프레임을 보간 시 블록열화현상을 피할 수 없게 된다[2]. 이를 해결하기 위하여 3차원 회귀 움직임 예측(3-D recursive ME) 기법이나 계층적 블록정합 움직임 예측 기법(hierarchical block matching ME)들을 통하여 실제 움직임 벡터를 찾는 방법이 제안되었다[4,9].

그러나 위에서 제안된 알고리즘들이 실제 움직임을 예측한다고 보장할 수 없을 뿐만 아니라 움직임 예측 블록이 전체 계산양의 70% 이상을 요구하는 점을 감안할 때 복잡한 알고리즘의 사용이 자유롭지 않음을 예상할 수 있다.

이 논문에서는 MCFI에 적용하기 위한 새로운 움직임 예측 방법인 움직임 형태 기반 움직임 예측(Shape-based ME) 기법을 제안한다. 제안된 기법은 움직임이 있는 물체의 형태를 효과적으로 추출하고 그 형태를 기반으로 움직임 예측을 수행 한다.

본 논문 구성은 다음과 같다. 2장에서는 본 논문에서 제안한 영역 구분 정의와 움직임 형태 기반의 움직임 예측 알고리즘을 설명하고 3장에서는 제안한 알고리즘을 다른 기법과 비교, 분석 후 4장에서 결론을 맺는다.

## Ⅱ. 제안된 움직임 형태 기반 움직임 예측 알고리즘

움직임 예측 기법은 비디오 처리 및 분석을 위한 중요한 기능 중 하나로 블록정합 알고리즘이 가장 널리 사용된다. 블록 기반의 움직임 보상기반 프레임 보간을 통해 삽입 될 프레임은 먼저 일정한 크기의 블록으로 분할되고 각 블록은 이전 프레임과 현재 프레임의 움직임 벡터를 이용하여 보간 되어 진다[6,7]. 블록정합 알고리즘에서 예측되어진 움직임 벡터는 절대 차이 값의 합(SAD) 중 가장 적은 값을 가진 블록을 선택하고 그때의 움직임 변위를 최종 움직임 벡터로 선택하게 되는데 식 1, 2와 같이 표현할 수 있다.

$$SAD(d\bar{x},d\bar{y})=\sum_{x=1}^{M}\sum_{y=1}^{N}\left|I_{t-1}(x+d\bar{x},y+d\bar{y})-I_t(x,y)\right| \quad (1)$$

여기에서 $I_{t-1}(x,y)$와 $I_t(x,y)$는 두 인접한 프레임 $f_{t-1}$과 $f_t$의 휘도 성분을 의미하고 t는 시간 지수, x, y는 공간지수를, $(d\bar{x},d\bar{y})$는 후보 움직임 벡터를 의미한다.

$$(\vec{V}_x,\vec{V}_y)=\arg\min_{(d\bar{x},d\bar{y})\in S}\{SAD(d\bar{x},d\bar{y})\} \quad (2)$$

그리고 식 2와 같이 이러한 후보 움직임 벡터들 중에서 가장 적은 SAD를 갖는 움직임 벡터를 최종 움직임 벡터로 선정한다. 식 2에서 S는 검색 영역을, $(\vec{V}_x,\vec{V}_y)$는 최종 움직임 벡터를 의미한다. 그러나 블록정합 알고리즘은 단지 시간적 대칭을 이용하여 움직임 벡터를 결정하기 때문에 만약 보간 될 프레임 기준에서 움직이는 물체들이 유사한 시간적 대칭성을 가지고 있다면 블록정합 알고리즘은 잘못된 움직임 벡터를 제공할 수 있으며 특히 폐색 영역에서 화질 열화를 초래한다[3].

이에 이 논문에서는 움직임 벡터의 정확성 향상과 계산양 감소를 위하여 움직임 형태 기반의 새로운 움직임 예측 알고리즘을 제안한다. 이를 위하여 먼저 4프레임간의 차를 구하고 그 결과를 바탕으로 영역을 구분 한다. 그리고 구분된 영역 중 움직임 영역을 기반으로 마스크 생성 후 이를 이용하여 움직임 예측을 수행하므로 움직임 벡터의 정확성 향상과 계산량을

줄이고자 한다.

### 2.1 다중 프레임 기반 움직임 검출

제안된 움직임 예측을 수행하기 위한 첫 단계는 그림 1과 같이 4개의 프레임을 이용하여 프레임 차 연산을 수행하는 것이다. 그리고 차 연산의 결과와 임계치 값($V_{threshold}$) 기준으로 4개의 영역으로 분류하는데 영역 정의는 다음과 같다.

정의 :

(1) 움직임 영역 : $Dn$ and $Dc$ and $Dp > V_{threshold}$.

(2) 정지 영역 : $Dc < V_{threshold}$.

(3) Covered 영역 : $Dn$ and $Dc > V_{threshold}$, $Dp < V_{threshold}$.

(4) Uncovered 영역 : $Dn < V_{threshold}$, $Dc$ and $Dp > V_{threshold}$.

여기서 $Dn$, $Dc$, $Dp$ 은 각각 현재 프레임과 다음 프레임간의 차이 값, 이전 프레임과 현재 프레임간의 차이 값, 그리고 이전 프레임과 이전이전 프레임과의 차이 값을 의미한다.

여기서 $V_{threshold}$ 값은 두개의 임계값을 사용하는데 먼저 '8' 이상일 경우와 '2' 에서 '7' 사이의 경우를 고려한다. '8'이상일 경우는 움직임 영역으로 구분하고 '2' 에서 '7' 사이의 경우 해당 화소 값들의 평균을 구하고 그 평균값을 임계값으로 움직임 영역을 구분한다. 이때 영역 구분을 위한 연산은 블록 기반으로 수행된다. 그림 2에서 $Diff$ 값은 프레임 차이 값을, $V_{average}$ 값은 2 ~ 7 사이 평균값을 의미한다.

다음 단계는 분류 된 결과를 기반으로 마스크를 생성하는데 이 단계에서 하나는 모든 움직임 영역을 포함하고 있는 마스크, 그리고 다른 하나는 움직임 예측 연산을 위한 마스크를 생성한다. 움직임 예측 연산을 위한 마스크에서는 다시 두 개의 마스크로 구분되는

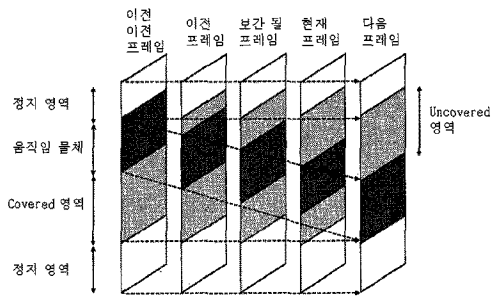



그림 1. 4프레임을 이용한 각 영역 구분

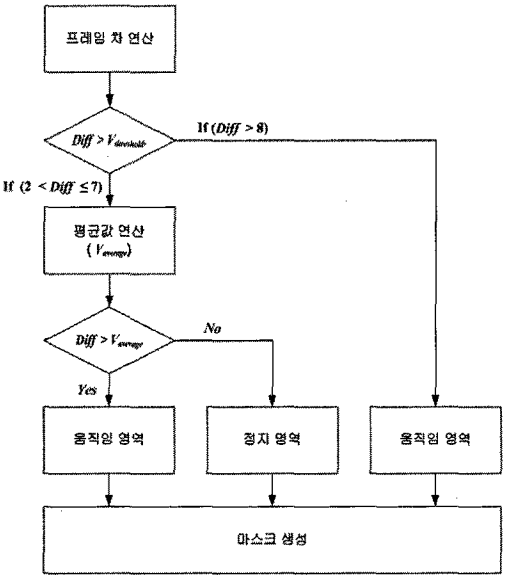



그림 2. 움직임 영역 검출 흐름도

데 그림 3에서처럼 움직임 영역과 covered 영역만을 포함하는 현재 움직임 마스크와 움직임 영역과 uncovered 영역만을 포함하고 있는 이전 움직임 마스크이다. 이러한 마스크를 생성하기 전 단계로 모폴로지 필터링 중 하나인 닫힘 연산을 수행하므로 잡음을 제거한다.

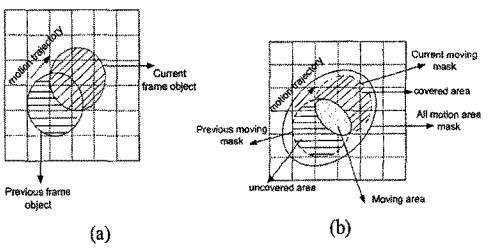



그림 3. 움직임 물체 궤적에 따른 영역 구분 및 제안된 영역 마스크

### 2.2 움직임 형태 기반 움직임 예측 알고리즘

제안된 움직임 예측은 앞에서 생성 된 현재 움직임 마스크와 이전 움직임 마스크를 기반으로 양방향으로 수행된다. 그림 4에서 보는바와 같이 움직임 예측의 경우 기존 블록정합 알고리즘처럼 블록 기반으로 움직임 예측을 수행한다.

그러나 기존의 블록 정합에 사용되는 블록은 모든 화소로 채워져 있는 반면, 제안한 방법에서의 블록은 검출된 움직임 영역만을 포함하고 있다. 즉, 임의의 블록이 움직임 영역을 포함하고 있지 않다면, 그 블록

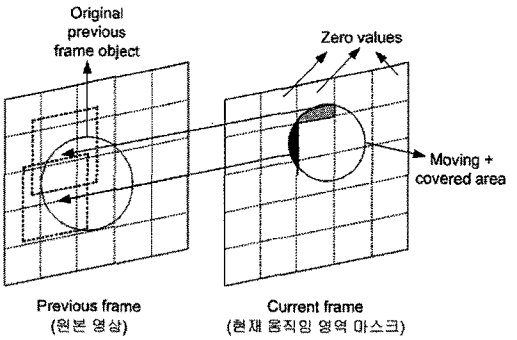



그림 4. 제안한 움직임 형태 기반 전 방향 움직임 예측

화소 값의 총 합은 '0'이 되고 움직임 예측에서 제외되는데 이를 통하여 움직임 예측을 위한 연산양은 감소된다. 한편, 블록 안에 움직임 영역과 정지영역이 동시에 존재 할 경우 임계값을 두어 임의의 블록이 움직임 예측에 참여하는데 제안을 두었다. 이를 통해 움직임 물체의 형태를 유지시키면서 움직임 형태에 기반 한 움직임 예측을 수행 가능하도록 한다.

일반적으로 SAD(Sum of Absolute Difference)는 움직임 예측을 위한 가장 효과적인 기준이 된다[8]. SAD 연산의 경우 현재 프레임과 이전 프레임 간 일정한 크기의 블록을 기반으로 연산을 수행하는데 이때 연산되는 블록들은 홀(화소 값이 '0'인 경우)을 포함하고 있지 않다. 그러나 제안한 움직임 예측 기법에서는 움직임 영역을 제외한 나머지 블록들은 모두 화소 값이 '0'이 된다. 또한 움직임 영역 안의 블록들도 '0' 값을 많이 포함하고 있기 때문에 일반적 SAD 연산은 적용 할 수 없다. 그러므로 식 3에서처럼 수정된 SAD 연산 식을 사용하여 움직임 예측을 수행한다.

식 3에서 New_Curr_Frame_Value는 현재 프레임에 현재 움직임 마스크가 적용된 화소 값을 의미하고 M 과 N은 블록 크기를 의미한다. 'sum' 은 움직임 영역을 포함하는 블록의 경우 움직임 예측에 참여 제한을 두기 위한 화소 수를 나타낸다. 이 논문에서는 'sum' 값이 8 이하일 땐 움직임 예측 참여를 제한하였다.

If New_Curr_Frame_value ≠ 0,

$$SAD(d\hat{x},d\hat{y})=\sum_{x=1}^{M}\sum_{y=1}^{N}\left|I_{t-1}(x+d\hat{x},y+d\hat{y})-I_t(x,y)\right| \quad (3)$$

*sum*++,

If Sum < 8, SAD = 255*M*N,

Else SAD = (SAD *M*N)/sum.

마지막으로, 제안된 움직임 예측에서는 기준 영상과 참조 영상의 블록에 포함되는 화소 수가 다르기 때문에 'sum' 값을 사용하여 평균 SAD를 연산하고 이를 기반으로 움직임 예측을 수행한다.

### 2.3 움직임 보상 프레임 보간

보간 될 중간 프레임을 만들기 위하여 일반적으로 움직임 보상기반 프레임 보간은 식 4를 이용하여 양방향 프레임 보간 기법을 수행한다[10].

$$f_t(i,j)=\frac{1}{2}\cdot f_{t-1}(i+\frac{1}{2}v_x,j+\frac{1}{2}v_y)+\frac{1}{2}\cdot f_{t+1}(i-\frac{1}{2}v_x,j-\frac{1}{2}v_y), \quad (4)$$

이 논문에서 사용하는 양방향 프레임 보간은 첫 단계에서 얻어진 움직임 영역 마스크를 기반으로 화소 기반 프레임 보간을 수행한다. 이때 화소단위 프레임 보간은 처음 생성 된 영역 마스크와 현재·이전 움직임 영역 마스크를 이용하여 수행하며 각 영역별 수식은 다음과 같다.

(1) 움직임 영역 : 양방향 움직임 보상기반 프레임 보간

$$f_t(i,j)=\frac{1}{2}\cdot f_{pre}(i+\frac{1}{2}v_x,j+\frac{1}{2}v_y)+\frac{1}{2}\cdot f_{cur}(i-\frac{1}{2}v_x,j-\frac{1}{2}v_y). \quad (5)$$

(2) 정지 배경 영역 : 양방향 평균 프레임 보간

$$f_t(i,j)=\frac{1}{2}(f_{t-1}(i,j)+f_{t+1}(i,j)). \quad (6)$$

(3) Covered 배경 영역 : 이전 프레임 보간

$$f_t(i,j)=f_{t+1}(i,j). \quad (7)$$

(4) Uncovered 배경 영역 : 현재 프레임 보간

$$f_t(i,j)=f_{t-1}(i,j). \quad (8)$$

Covered/uncovered 배경 영역은 움직임 영역 보간 후 비어있을 수 있는 영역들 중 covered· uncovered 영역에 대한 보간이며 정지영역 보간과 함께 첫 단계에 생성 된 영역 마스크를 기반으로 수행된다.

## Ⅲ. 실험 결과

이 논문에서 실험은 표준 테스트 영상을 사용하였고 크기는 CIF(352 × 288) 형식을 사용하였다. 움직

임 예측과 움직임 보상에서의 블록 크기는 16 × 16을 사용하였고 검색 영역은 수평 및 수직 방향으로 각각 8개 화소 범위에서 수행하였다. 그리고 프레임 보간을 위한 입력 테스트 영상은 홀수 프레임을 제거하고 짝수 프레임만을 이용하였다.

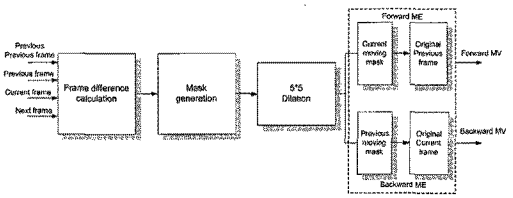



그림 5. 제안한 움직임 예측 블록 다이어그램

### 3.1 제안한 움직임형태기반 움직임 예측결과

영역 구분(그림 6(a)) 후 현재 및 이전 움직임 영역 마스크 결과는 그림 6(b), (c)와 같다. 영역 구분을 위한 임계치 값은 설명한 것처럼 '8'과 '2 ~ 7' 사이 평균값을 사용하였다. 59번째 프레임 보간 시 전 방향 움직임 예측은 그림 6(e)와 원본 영상 58번째 프레임, 후 방향 움직임 예측은 그림 6(d)와 원본영상 60번째 프레임 사이에서 수행한다. 그림 6(a)에서 검은색은 covered, 흰색은 uncovered, 밝은 회색은 움직임, 어두운 회색은 정지영역을 나타낸다.

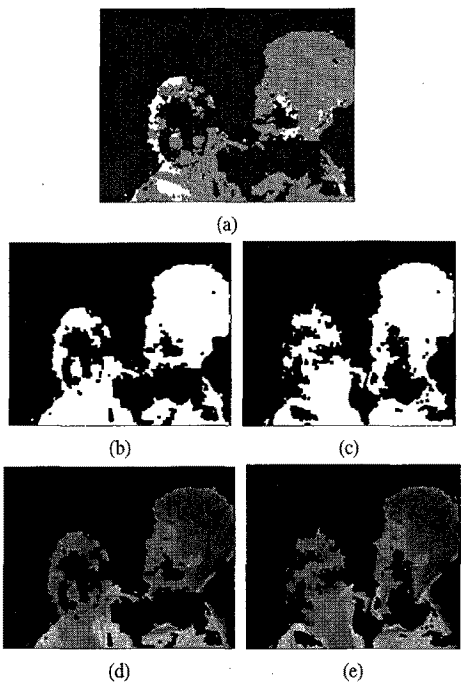



그림 6. 제안한 방법의 움직임 마스크(59th 프레임) : (a) 영역 마스크, (b) 이전 움직임 영역 마스크, (c) 현재 움직임 영역 마스크, (d)와(e) 이전·현재 움직임 영역 마스크에 원본영상 적용 결과

그림 7과 8은 프레임 보간 구현 알고리즘에서 가장 많이 사용되고 있는 움직임 예측 기법들과 제안한 기법의 결과를 비교한 것이다. 그림 7에서 (a)~(d)는 각각 전영역 검색 움직임 예측(FS : Full search), 빠른 전영역 움직임 예측(Fast FS : Fast full search), 양방향 움직임 예측(BD : Bidirectional)과 이 논문에서 제안한 움직임 예측 결과를 보여주고 있다.

양방향 움직임(BD) 예측은 초기 움직임 벡터를 가지고 양방향으로 동시에 움직임 예측을 수행한다. 그

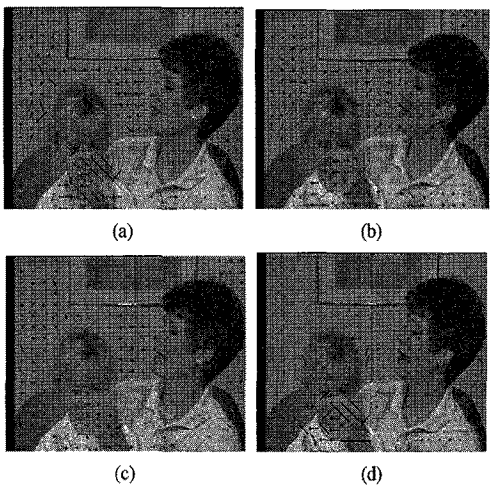



그림 7. 전 방향 움직임 예측 알고리즘 결과(mother_daughter, 59th 프레임) : (a) FS, (b) FFS, (c) BD, (d) 제안한 알고리즘

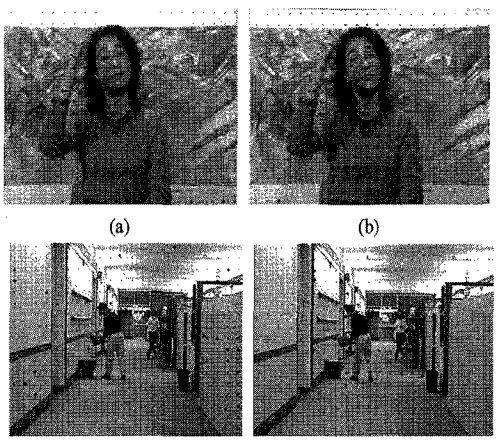



그림 8. 전방향 움직임 예측 알고리즘들의 실험 결과(silent 25th, hall 91th 프레임) : (a) 와 (c) FS, (b) 와 (d) 제안한 알고리즘

리고 제안한 움직임 예측 연산에서는 현재/이전 움직임 마스크에 8개 이하의 화소를 포함하고 있는 블록은 발산의 가능성이 있기 때문에 예측 연산에서 제외하였다.

그림 7과 8에서 보는바와 같이 제안한 알고리즘의 움직임 예측 결과는 움직임이 있는 손과 얼굴, 그리고 뒤 배경 중 그림자 움직임이 있는 곳에서만 움직임 벡터가 발생된 것을 볼 수 있다. 특히 그림 7(a),(b),(c)와 그림 8(c)에서 움직임이 없는 배경에서도 움직임 벡터가 발생된 것은 움직임 예측 시 시간적 대칭을 이용하여 SAD를 계산하기 때문에 유사한 시간적 대칭성에 기인한 움직임 벡터 발생이다. 이러한 움직임 벡터는 실제 움직임이 없는 영역에서 움직임 보상처리를 수행하여 블록 열화현상이나 화질열화 현상을 일으키는 원인이 되기도 한다.

표 1은 각 움직임 예측 알고리즘들의 수행 시간과 평균 수행시간을 비교한 결과이다. 움직임 예측 수행시간은 양방향 움직임 예측 수행시간을 나타내고 평균 수행시간은 총 100 프레임의 수행시간을 평균한 것이다. 표 1의 수행시간 측정은 Intel Dual Core 1.8 GHz 환경에서 측정한 결과인데 이는 상대적인 연산 수행 시간을 보여주기 위함이다. 빠른 전영역 검색의 수행 시간이 가장 빠르고 다음으로 제안한 알고리즘의 수행 시간이 빠름을 알 수 있다. 객관적인 비교를 위하여 표 2에서는 움직임 예측 시 평균 검색 화소수를 비교하였다.

그림 9에서는 움직임 예측기의 성능 비교를 위하여 평균 절대 차이 값의 합(SAD)를 비교하였다. 결과에서 보는 바와 같이 제안한 움직임 예측기는 표1에서

표 1. 각 움직임 예측 알고리즘 수행 시간 비교

| 입력영상 | 움직임 예측 알고리즘 | 수행 시간 (양방향) | 한 프레임당 평균 수행시간 |
|---|---|---|---|
| Silent (CIF : 352×288) | 전영역 검색(Full search) | 13.603 sec | 0.453 sec |
| | 빠른 전영역 검색(Fast full search) | 6.282 sec | 0.21 sec |
| | 양방향 검색(Bidirectional) | 23.297 sec | 0.77 sec |
| | 제안한 기법 | 9.704 sec | 0.31 sec |
| Mother_daughter (CIF : 352×288) | 전영역 검색(Full search) | 13.914 sec | 0.463 sec |
| | 빠른 전영역 검색(Fast full search) | 8.984 sec | 0.3 sec |
| | 양방향 검색(Bidirectional) | 22.766 sec | 0.76 sec |
| | 제안한 기법 | 10.141 sec | 0.33 sec |

표 2. 평균 검색 화소 수

| 알고리즘 | Silent | Mother_daughter |
|---|---|---|
| FFS(Diamond step) | 19 | 22 |
| TSS(Three Step Search) | 25 | 25 |
| Full search / Bidirectional | 255 | 255 |
| Proposed | 69 | 82 |

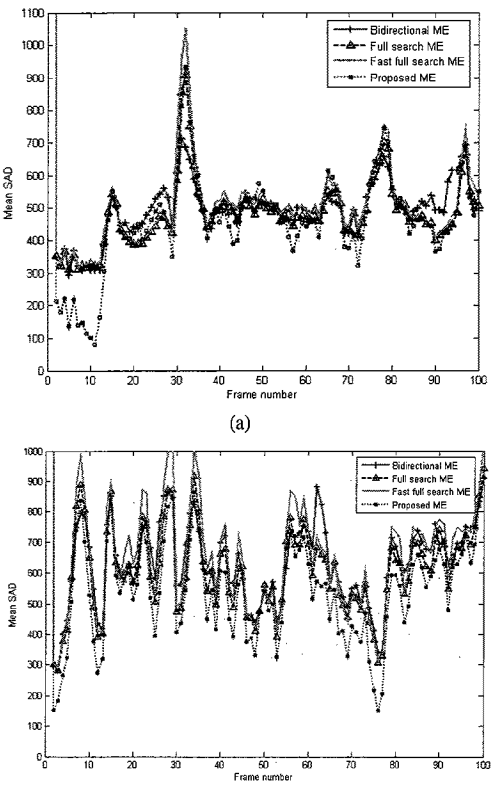



그림 9. 평균 절대 차이 값의 합(SAD) : (a) Mother_daughter 영상, (b) Silent 영상

처럼 연산 시간은 줄어드는 반면 전 영역 검색 움직임 예측기(Full search BMA)의 성능과 유사하거나 향상됨을 확인 할 수 있다.

영역 구분을 위한 프레임 차 연산은 화소의 밝기 변화에 기반 한다. 때문에 밝기 변화가 큰 Crew 영상 같은 경우 그림 10(a)처럼 거의 전 영역이 움직임 영역으로 정의된다. 그리고 이를 기반으로 현재 및 이전 움직임 영역 마스크 결과는 그림 10(b),(c)와 같다. 이와 같이 영상 전체에 밝기 변화가 큰 경우 전 영역이 움직임 영역으로 정의되기 때문에 그림 10(d),(e)를 이용한 움직임 예측은 전 영역 검색 움직임 예측(Full search BMA)과 거의 동일한 움직임 예측을 수행하게 된다. 그러므로 최악의 경우 검색 화소 수는 전 영역 검색 움직임 예측과 같은 255개의 화소가 된다.

그림 11에서는 전영역 검색 움직임 벡터와 제안한 방법의 움직임 벡터 결과를 비교하였다. 그림 11에서 (a)와(c)의 결과를 비교하였을 때 제안한 움직임 예측 방법은 밝기 변화가 큰 영상에서 전 영역 검색 움직임 벡터 결과와 유사함을 확인 할 수 있다.

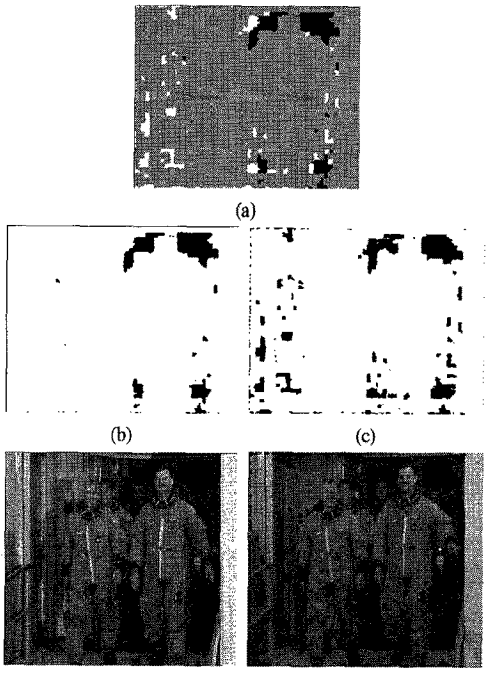


그림 10. Crew 영상의 제안한 움직임 마스크(61st 프레임) : (a) 영역 마스크, (b) 이전 움직임 영역 마스크, (c) 현재 움직임 영역 마스크, (d)와(e) 이전·현재 움직임 영역 마스크에 원본영상 적용 결과

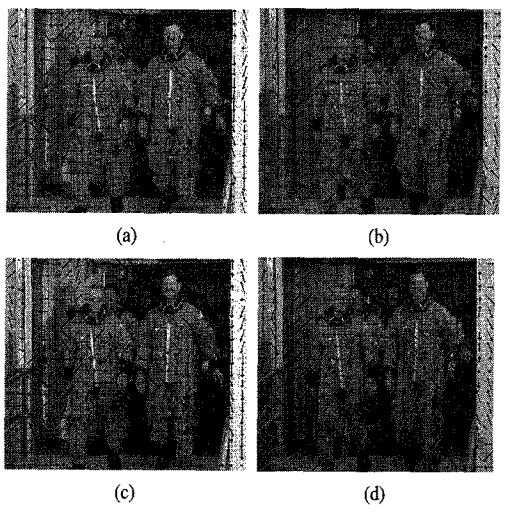


그림 11. Crew 영상의 전방향 움직임 벡터(59th, 61st 프레임) : (a)와(b) 전 영역 검색 움직임 벡터(59th, 61st 프레임), (c)와(d) 제안한 알고리즘 움직임 벡터(59th, 61st 프레임)

### 3.2 움직임 보상기반 프레임 보간 결과

그림 12는 3-D recursive 움직임 예측·보간과 양방향(Bilateral) 움직임 예측·보간[8], 그리고 제안한 움직임 예측·보간 결과를 보여준다. 그림 12(a),(d),(g)는

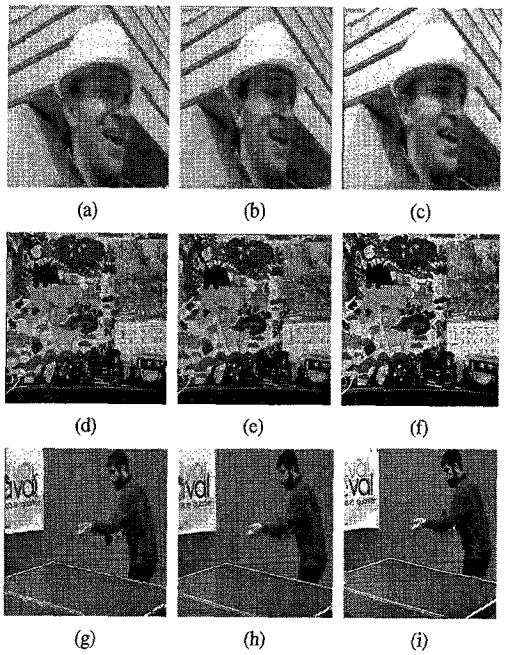


그림 12. 각 움직임 예측 결과를 이용한 프레임 보간 결과 : (a),(d),(g) 3-D recursive, (b),(e),(h) [8] 알고리즘, (c),(f),(i) 제안한 알고리즘

3-D recursive 블록정합을 이용하여 움직임 예측 수행 후 홀과 중첩영역 처리를 위하여 [11, 12]의 방법을 사용하였다[8]. 그림 12(b),(e),(h) 경우 Bilateral 움직임 예측 후 VS-BMC(가변 크기 블록 움직임 보상), AOBMC(적응적 중첩블록 움직임 보상 보간)을 이용하여 프레임 보간을 수행하였고[8], 그림 12(c),(f),(i)는 제안한 움직임 예측과 벡터 미디언 필터 적용 후 화소 기반 프레임 보간을 수행한 결과이다. 그림 12(a),(g)에서처럼 3-D recursive 방법의 경우 얼굴 주변과 테이블 경계면에 블록화 현상을 볼 수 있는 반면 제안된 방법에서는 블록화 현상이 현저하게 줄어듦을 볼 수 있다. 특히 그림 12(d),(e),(f)의 보간 결과에서는 제안된 방법이 달력의 숫자 부근에서 훨씬 더 선명함을 볼 수 있다.

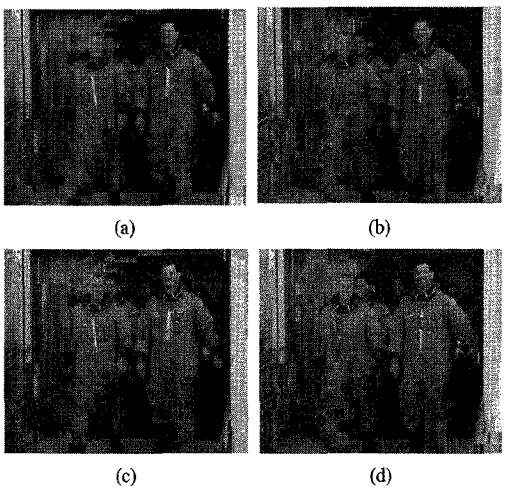
표 3은 각 테스트 영상들에 대해 보간 된 100 프레임의 평균 PSNR 결과와 구조 유사성(Structural similarity, SSIM)[13] 결과를 비교하였다. 표 3에서 Foreman과 Table 영상의 경우 제안한 방법의 평균 PSNR이 [8]의 결과와 비슷하거나 약간 낮음을 볼 수 있다. 그러나 구조의 유사성 비교에서는 제안한 방법이 높음을 알 수 있는데 이는 움직임 형태 기반의 움직임 예측으로 인해 보간 된 움직임 물체의 구조가 다른 두 방법에 비해 원본 영상의 움직임 물체와 더 유

표 3. 평균 PSNR 및 SSIM 비교

| 알고리즘 | Mobile | Foreman | Table |
|---|---|---|---|
| 3-D recursive MCI | 22.34 dB<br>85.21 % | 30.54 dB<br>91.40 % | 26.75 dB<br>86.56 % |
| [8] MCI | 24.32 dB<br>86.54 % | 33.02 dB<br>92.22 % | 32.43 dB<br>91.31 % |
| 제안한 알고리즘 | 26.61 dB<br>87.51% | 32.13 dB<br>92.48 % | 32.18 dB<br>91.52 % |

사하기 때문이다.

그림 13에서는 밝기 변화가 큰 Crew 영상의 움직임 보간 결과를 비교하였다. 그림 13(a), (b) 와 13(c), (d)는 각각 전영역 검색 움직임 예측과 제안한 움직임 예측 후 벡터 미디언 필터, 양방향 프레임 보간 방법을 사용하였다. PSNR 비교결과에서 볼 수 있듯이 밝기 변화가 큰 영상인 경우 제안한 방법이 전 영역 검색 움직임 예측(Full search BMA) 결과와 유사함을 확인 할 수 있다.



(a) (b) (c) (d)

그림 13. Crew 영상의 움직임 보상 프레임 보간 ($59^{th}$, $61^{st}$ 프레임) : (a)와(b) 전 영역 움직임 예측 후 움직임 보상, PSNR = 24.42 dB, 24.08 dB, (c)와(d) 제안한 움직임 예측 후 움직임 보상, PSNR = 24.54 dB, 24.06 dB

## V. 결 론

이 논문에서 우리는 움직임 벡터의 정확성 향상과 계산양 감소를 위하여 움직임 형태 기반의 움직임 예측 기법을 제안하였다. 제안된 움직임 예측 기법의 특징은 다음과 같다.

첫 번째로, 움직임 예측을 위해 현재 움직임 영역 마스크와 이전 움직임 영역 마스크로 구분 하였는데 이는 영상 이미지에서 움직임이 있는 물체의 모양을 추출하여 그 결과를 기반으로 움직임 예측 수행을 목표로 하였고 이를 통하여 움직임 예측의 정확성을 높이고자 하였다. 이 두 마스크는 4개의 프레임을 이용하여 차 연산을 수행하고 움직임 영역, covered /uncovered 영역 그리고 정지 배경영역으로 구분 후 움직임 영역과 covered 영역을 현재 움직임 영역으로 움직임 영역과 uncovered 영역을 이전 움직임 영역으로 구분하여 마스크를 생성하였다.

두 번째로, 제안된 움직임 예측을 위하여 기존 SAD 연산을 수정하였다. 기존의 움직임 예측 SAD 연산에서는 해당 블록 간 같은 화소수를 기반으로 연산을 수행하였지만 제안된 기법에서는 SAD 연산 시 화소 수가 같지 않는 경우가 발생한다. 이를 해결하기 위하여 평균값 기반 SAD 연산을 제안하고 수행하였다.

마지막으로, 움직임 보상 기반 프레임 보간을 움직임 영역 마스크와 영역 마스크 기반으로 화소 단위로 수행하였다.

결과적으로 제안한 알고리즘을 통하여 전 영역 움직임 예측 기법 연산양의 평균 25% 연산양을 가지고 움직임 예측 수행이 가능하였다. 비록 FSS, TSS의 평균 연산양에 비해 다소 높긴 하나 첫째로 빠른 움직임 영역에서의 움직임 정보 정확성 향상과 둘째로 하드웨어 구현 측면에서 전 영역 움직임 예측기와 비슷한 구조를 가지고 있어 구현용이 장점을 가지고 있다.

이 논문에서 사용된 테스트 영상은 카메라 회전이나 줌과 같은 큰 전역(global) 움직임이 포함되어 있지 않은 영상이다. 만약 큰 전역 움직임이 있는 테스트 영상을 사용한다면 첫 단계인 프레임 차를 계산할 때 정지 배경영역 마저도 움직임이 있는 영역으로 인식하여 연산양 증가를 초래한다. 이를 해결하기 위하여 향후 전처리 과정으로 전역 움직임 검출 및 보상 알고리즘 개발이 필요하다.

## 참 고 문 헌


[1] C.K. Wong and O.C. Au, "Fast motion compensated temporal interpolation for video", Proc. SPIE: Visual Communications and Image Processing '95, pp.1108-1118, May 1995

[2] Ravi Krishnamurthy, J.W. Woods, "Frame Interpolation and Bidirectional Prediction of Video Using Compactly Encoded Optical-Flow Fields and Label Fields", in IEEE Transaction on Circuits and Systems for Video Technology, Vol.9, No.5, 1999

[3] B.T. Choi, S.H. Lee, "New Frame Rate Up-Conversion Using Bi-directional Motion Estimation", in IEEE Transaction on Consumer Electronics, Vol.46, No.3, 2000

[4] Taehyeun Ha and S.J. Lee, " Motion Compensated Frame Interpolation by new Block-based Motion Estimation Algorithm", in IEEE Transaction on Consumer Electronics, Vol.5, No.2, 2004

[5] A. Huang and T. Nguyen, "A novel motion compensated frame interpolation based on block merging and residual energy", in Proc. Multimedia Signal Processing Workshop, Sep. 2006, Vol.4, pp. 353-356

[6] Dane, G. T. Nguyen, "Optimal temporal interpolation filter for motion-compensated frame rate up conversion", in IEEE Transaction on Image processing, Vol.15. Issue 4, 2006

[7] Ya-Ting Yang, Ja-Ling Wu, " Quality Enhancement of Frame Rate Up-Converted Video by Adaptive Frame Skip and Reliable Motion Extraction", in IEEE Transaction on Circuits and Systems for Video Technology, Vol.17, No.12, 2007

[8] Bayeon-Doo Choi, Chang-Su Kim, Sung-Jae Ko, " Motion-Compensated Frame Interpolation Using Bilateral Motion Estimation and Adaptive Over-lapped Block Motion Compensation", in IEEE Transaction on Circuits and Systems for Video Technology, Vol.17, No.4, 2007

[9] Ai-Mei Huang, T. Nguyen, " A Multistage Motion Vector Processing Method for Motion Compensated Frame Interpolation", in IEEE Transaction on Image Processing, Vol.7, No.5, 2008

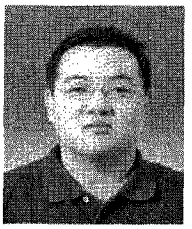
[10] Suk-Ju Kang, Dong-Gon Yoo, Sung-Kyu Lee, "Multiframe-based Bilateral Motion Estimation with Emphasis on Stationary Caption Processing for Frame Rate Up-Conversion", in IEEE Transaction on Consumer Electronics, Vol.54, No.4, 2008

[11] P. Csillag and L. Boroczky, "Enhancement of video data using MC postprocessing techniques," in *Proc. ICASSP*, Apr. 1997, Vol.4, pp.2897~2900.

[12] D. W. Kim, J. T. Kim, and I. H. Ra, "A new video interpolation technique based on motion-adaptive subsampling," *IEEE Trans. Consum. Electron.*, Vol.45, No.3, pp. 782-786, Aug. 1999.

[13] Zhou Wang, Alan Conrad Bovik, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, Vol.13, No.4, pp.600-612, Apr. 2004.

**박 주 현** (Ju Hyun Park) 정회원



1992년 2월 조선대학교 전자공학과

2002년 2월 전남대학교 전자공학과 석사

2006년 3월~현재 전남대학교 전자컴퓨터공학과 박사과정

<관심분야> 임베디드 SoC 설계, 저전력 설계, 영상 관련 SoC 및 VDP 설계

**김 영 철** (Young-Chul Kim) 정회원

1981년 2월 한양대학교 전자공학과

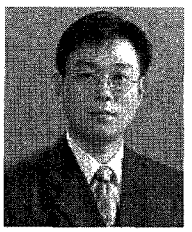
1987년 2월 Univeristy of Detroit 전자 공학과 석사

1993년 2월 Michigan state University 전자공학과 박사

1993년~현재 전남대학교 전자컴퓨터공학부 교수

<관심분야> 임베디드 SoC 설계, 저전력 설계, 영상 관련 SoC 및 VDP 설계

**홍 성 훈** (Sung-Hoon Hong) 정회원



1988년 2월 영남대학교 전자공학과

1991년 2월 한국과학기술원 전기 및 전자 공학과 석사

1991년~2000년 7월 LG전자 DTV 연구소 책임연구원

2000년 7월~현재 전남대학교 전자컴퓨터공학부 부교수

<관심분야> 영상통신시스템, SVC, 객체분할, 영상 편집 시스템