내용기반 비디오 검색을 위한 움직임 벡터 특징 추출 알고리즘

(Efficient Methods for Detecting Frame Characteristics and Objects in Video Sequences)

이 현 창 † 이 재 현 †† 장 옥 배 †††
(HyunChang Lee) (JaeHyun Lee) (OkBae Jang)

요 약 본 논문은 비디오의 효율적인 내용기반 검색을 지원하기 위해 움직임벡터의 특징을 검출하였다. 이를 위해 비디오의 현재 프레임을 일정한 크기의 블록으로 나누고 시간 축상 기준이 되는 프레임에서 각 블록의 움직임을 추정하는 블록정합 알고리즘을 이용하였다. 하지만 블록 정합법의 경우 여러 가지 제약 조건과 함께 블록 정합법에 의해 얻어지는 벡터가 실제 움직임과 상이한 경우도 발생한다. 이러한 문제점을 해결하기 위해서 전역탐색방식을 응용했으나 이 방법은 계산량이 많다는 단점이 있다. 그 대안으로 본 논문에서는 움직임 벡터의 시공간 상관성(MVSTC : Motion Vector Spatio-Temporal Correlations)의 시간적 공간적 특징을 추출하였다. 그 결과 본 논문에서는 인접 블록의 움직임 벡터를 이용하여 좀 더 정확한 움직임 벡터의 예측을 수행할 수 있었다. 하지만 참조되는 블록 벡터의 수가 여러개 발생되기 때문에 이러한 부가 정보를 수신단에 전송해야 하는 부담을 초래하게 된다. 따라서 각 블록의 움직임 특징을 예측하고 이에 알맞은 탐색 범위를 설정하는 문제도 고려해야 한다. 제시된 알고리즘을 바탕으로 움직임 보상을 위한 움직임 추정 기법을 고찰하고 이를 적용한 결과를 제시하고자 한다.

키워드 : 내용기반 검색, 멀티미디어, 에이치씨아이, 옵티컬 플로우, 비디오 인덱스, 비쥬얼 인터페이스

Abstract This paper detected the characteristics of motion vector to support efficient content-based video search of video. Traditionally, the present frame of a video was divided into blocks of equal size and BMA (block matching algorithm) was used, which predicts the motion of each block in the reference frame on the time axis. However, BMA has several restrictions and vectors obtained by BMA are sometimes different from actual motions. To solve this problem, the full search method was applied but this method is disadvantageous in that it has to make a large volume of calculation. Thus, as an alternative, the present study extracted the Spatio-Temporal characteristics of Motion Vector Spatio-Temporal Correlations (MVSTC). As a result, we could predict motion vectors more accurately using the motion vectors of neighboring blocks. However, because there are multiple reference block vectors, such additional information should be sent to the receiving end. Thus, we need to consider how to predict the motion characteristics of each block and how to define the appropriate scope of search. Based on the proposed algorithm, we examined motion prediction techniques for motion compensation and presented results of applying the techniques.

Key words : Content-Based Retrieval, Multimedia, HCI, Optical Flow, Video Index, Visual Interface

† 정 회 원 : 전북대학교 컴퓨터통계정보학과
hclee@jbbank.co.kr
†† 정 회 원 : 벽성대학교 컴퓨터학과 교수
jhlee@mail.byuksung.ac.kr
††† 종신회원 : 전북대학교 컴퓨터과학과 교수
okjang@moak.chonbuk.ac.kr
논문접수 : 2007년 8월 3일
심사완료 : 2007년 12월 11일

정보과학회논문지: 소프트웨어 및 응용 제35권 제1호(2008.1)

1. 서 론

최근 컴퓨터 발전에 따른 멀티미디어 컨텐츠의 내용기반(content-based) 검색에 대한 연구가 활발히 진행되고 있다. 내용기반 검색은 해석과 설명을 통해 텍스트가 표현할 수 없었던 다양한 효과를 얻을 수 있는 표현방법과 검색 시간을 단축시켜 다량의 정보를 효과적으로 제공받아 개인에 적합한 특정 정보를 제공할 수 있어야 한다. 이러한 요구에 현재 웹에서는 단순 속성(attribute)이나 수동식 기술(description), 또는 캡션 자료를 이용한 텍스트의 매칭에 의한 경우가 대부분이다. 그러나 멀티미디어 정보는 여러 가지 속성을 가지고 있으며, 모든 멀티미디어에 관한 기술을 사람이 수행해야 할 때는 엄청난 작업량이 수반 될 뿐만 아니라 동일한 데이타에 대한 기술이 주관에 따라 달라질 수도 있다. 때문에 여러 유형의 멀티미디어 컨텐츠에 대한 빠르고 효율적인 검색방법의 새로운 기술이 필요하다[1,2].

내용기반 검색을 위한 비디오 데이타베이스의 구성은 특징을 기반으로 비디오를 접근할 수 있도록 제공하며, 특징을 기반으로 데이타를 구성하거나 비디오를 인덱싱화하여 성취될 수 있을 것이다. 시간성을 갖는 멀티미디어 데이타를 정확히 표현하기 위해서는 멀티미디어 데이타의 시간성에 관한 정보를 유지, 관리하여야 한다[2-4]. 특징을 기반으로 하는 부분적인 인덱스[5]는 영상이나 비디오에 대부분 의존하는 하위 레벨의 기술로서, 어떤 하위 레벨의 측정을 유형적으로 추출하고 그리고 비디오를 순서화하기 위해 그러한 특징들을 인덱스 키로 이용한다[2,4]. 이러한 특징적인 인덱스 키로 사용되는 것으로서 유형적인 영상 열을 기초로 한 차 영상(difference image)이나 대표 컬러, 영상 움직임 벡터 등이 있다[6]. 본 논문에서 연구한 움직임 벡터와 관련된 기존의 연구는 내용기반 검색을 위해 주로 문자 대표영상 등을 인덱스로 사용하였으나[3,4] 이는 동영상의 고유한 특징인 움직임 활동 정도를 효과적으로 기술하지 못하는 단점이 있다. 동영상에서의 움직임 활동 특징을 기술하는 종래의 기술로는 카메라 움직임이나 영상 객체관련 궤적의 특징을 사용하는 방법이 있으나[7,8] 이는 동영상 화면 전체에 나타나는 전체적인 움직임 활동의 특징을 기술하지 못하는 단점이 있다. 또한 기존의 영상에서의 움직임의 크기 자체를 이용하여 움직임의 크기에 대한 성질을 나타낼 수 있었으나[2] 영상속 움직임관련 변화량의 개념인 움직임 활동성을 나타내는 특징이 제안되어 있지 않았다. 본 논문에서는 비디오 내용검색에서 사용자에게 보다 다양한 질의를 제공하기 위해서 움직임 벡터를 이용함으로서 얻어진 움직임 정보로부터 펜, 틸트, 줌 그리고 객체의 검출, 점진적인 전환과 카메라 컷 모두에 대한 정보를 얻어낼 수 있고, 연산시간의 감축과 검색시스템 설계시 연산의 단순화와 같은 이점을 얻을 수 있는 방법을 개발하여 인덱스 필드로 활용할 수 있도록 제안했다.

본 논문에서는 내용기반 검색지원을 위한 연구로 비디오 시퀀스에서 움직임 객체를 검출하고 움직임 객체에 내재된 움직임벡터의 방향과 시간의 특징을 추출하여 이를 인덱스에 활용함으로서 궁극적으로 비디오 데이타베이스를 구축하고 멀티미디어 정보검색에 기반 기술이 될 수 있도록 하기 위함이다. 이를 위해 본 논문에서는 첫째, 블록 정합 알고리즘을 이용한 움직임 벡터의 연산에 관한 연구로 블록 정합 알고리즘은 현재 프레임을 일정한 크기의 블록으로 나누고 시간축상 기준이 되는 프레임에서 각 블록의 움직임을 추정하는 방식이다. 따라서 본 연구에서는 움직임 추정 방법 중 블록 단위로 움직임을 추정할 때 움직임 벡터의 시공간적 특성을 이용하였다. 둘째, 움직임의 궤적을 고려한 움직임 추정에관한 연구로 동영상이 내포하고 있는 각 구성요소들의 움직임은 정지해 있는 배경, 그리고 매우 느린 움직임으로부터 매우 빠른 움직임에 이르기까지 그 움직임의 크기에 있어서 많은 분포를 보인다. 따라서, 제한된 탐색영역을 유지한 채 움직임 추정을 행할 때, 만일 빠른 움직임이 발생할 경우 큰 예측 오차로 인한 오검출을 유발할 수 있다. 본 연구는 동영상 움직임 추정시 제한된 탐색 범위로 인해 발생되는 점들을 해결하여 움직임의 추정보상 효율을 향상시키고, 고속의 알고리즘을 제시함으로써 효과적인 움직임 추정 기법에 의한 움직임 벡터 속성 기술을 검출하고자 한다. 본 논문의 구성은 1장 문제의 정의 2장 서론 및 관련연구 3장 본 논문에서 제안한 방법 및 이론 4장 움직임 특징 추출 및 분석 5장 본 논문에서 제시한 알고리즘에 대한 실험 6장 향후 연구과제로 결론을 맺었다.

2. 디지털 비디오 움직임 추정 기법

2.1 블록 정합 알고리즘

연속하는 두 프레임 사이의 움직임 추정법은 여러 가지가 있으나 움직임을 추정하는 단위에 따라 크게 화소순환법(PRA)과 블록정합법(BMA:Block Matching Algorithm)으로 나뉜다[9]. 화소순환법은 화소 단위로 움직임을 추정하기 때문에 정확한 움직임 값을 얻을 수 있으나 반복적인 계산과 부화소 위치의 보간 과정이 필요하기 때문에 계산량이 많고 복잡하여 하드웨어 구현이 어렵다. 반면에 블록정합법은 2차원 영상을 N개의 블록으로 나누어 블록 단위로 물체의 움직임을 추정하는 방법이다. 블록정합법은 블록 단위로 움직임을 추정하기 때문에 화소순환법보다 정확도는 떨어지지만 알고리즘

이 단순하여 하드웨어 구현에 용이하다는 이점이 있다. 따라서 대부분의 시스템에서는 블록정합법을 사용한다. 본 논문에서는 현재 프레임을 일정한 크기의 블록으로 나누고 시간축상 기준이 되는 프레임(reference frame)에서 각 블록의 움직임을 추정하는 방식으로 보편적인 블록정합법을 사용한다.

일반적으로 움직이는 물체에는 회전 운동 또는 크기의 확대나 축소가 있을 수 있으나, BMA는 시간 흐름에 따른 물체의 변형이 많이 생기지 않고, 물체의 움직임은 그 물체를 구성하는 블록들의 움직임으로 근사화 할 수 있다는 전제하에 사용되는 움직임 추정 방식이다.

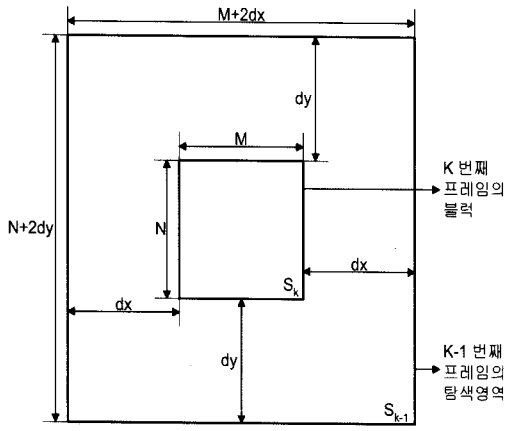


그림 1 블록 정합 알고리즘

그림 1에서와 같이 BMA에서의 움직임 추정은 $(K\text{-}1)$번째 프레임에서 $(M+2dx)\times(N+2dy)$의 탐색 영역(search region)을 정하고 K번째 프레임에서의 $M\times N$ 크기의 블록과 같은 크기의 블록을 $(K\text{-}1)$번째 프레임에서 탐색 영역을 벗어나지 않도록 하여 서로간의 유사도를 계산한 후 가장 알맞은 블록을 찾아 이 때의 변위를 계산하여 해당 블록의 움직임 벡터로 한다. 유사도를 측정하는 평가 함수(cost function)로는 식 (1), 식 (2), 식 (3) 등에 정의되는 평균 제곱 오차(mean square error: MSE), 평균 절대값 오차(mean absolute error: MAE), 정규화된 상호 상관 함수(normalizd cross correlation function: NCCF) 등이 주로 사용된다.

$$MSE=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}[S_k(m,n)-S_{k-1}(m,n)]^2 \quad (1)$$

$$MAE=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}|S_k(m,n)-S_{k-1}(m,n)| \quad (2)$$

$$NCCF=\frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}S_k(m,n)-S_{k-1}(m,n)}{\left\{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}S_k(m,n)^2\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}S_{k-1}(m,n)^2\right\}^{1/2}} \quad (3)$$

여기서 S_k와 S_{k-1}은 각각 K번째 프레임과 $(K\text{-}1)$번째 프레임이다.

블록 정합 알고리즘에는 탐색영역 내의 모든 위치들에 대해 움직임을 추정하는 전역 탐색(absolute force or full search) 방식과 전역 탐색에 비하여 성능은 떨어지지만 계산 시간에서 이득을 얻을 수 있는 고속 탐색 방식(fast algorithm), 그리고 계층적인 방법을 이용하는 계층적 블록 정합 알고리즘(hierarchical BMA) 등이 있다.

3. MVSTC에 기반한 특수효과 검출

카메라와 물체의 상대적 움직임에 의해 발생된 움직임 벡터는 동영상에서만의 특징으로 움직임벡터는 시간성과 공간성을 가지고 있다. 따라서 본 논문에서는 움직임 벡터의 시공간 상관성(MVSTC : Motion Vector Spation-Temporal Correlations)의 시간적 공간적 특징을 추출한 후 비디오의 씬에 포함된 특수효과를 검출하여 인덱스 정보에 활용한다.

3.1 움직임 벡터의 공간적 특성

영상 내에 움직이는 물체는 공간적으로 높은 상관도를 가지고 있다. 그림 2의 블록 O는 움직임에 있어서, 인접한 블록인 N1,N2,N3 그리고 N4와 서로 높은 상관성을 지니고 있다.

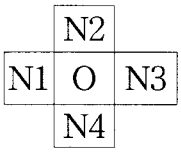


그림 2 공간적 4근방 블록

그림 2에서 움직임 벡터의 배경 영역에서는 거의 영(zero)의 값을 가지며 움직이는 물체 부분에서 공간적으로 인접 블록의 벡터와 거의 비슷하나, 많은 부분에서 인접 블록과 매우 다른 움직임 벡터를 나타내고 있음을 알 수 있다.

이러한 현상은 여러 가지 원인이 있을 수 있는데 첫째, 블록 정합법의 적용시 최소 정합 오차를 만족하기만 하면 정합 블록으로 선택되기 때문에 실제 움직임과 불일치하는 경우가 발생한다. 둘째, 서로 다르게 움직이는 물체 또는 물체와 배경 등이 동일한 블록에 속해져 있는 경우이다. 그 외 드러난 배경(uncovered background) 영역 등의 경우에도 인접 블록간의 움직임 상관도가 매우 떨어질 수 있다. 움직임 벡터의 크기와 편이량은 식 (4), 식 (5)에 의해서 구해진다.

$$\text{움직임 벡터의 크기} = \sqrt{v_x^2+v_y^2} \quad (4)$$

$$\text{움직임 벡터의 편이량} = \left|4v_x - \sum_{4}^{k=1} v_{xk}\right| + \left|4v_y - \sum_{4}^{k=1} v_{yk}\right| \quad (5)$$

여기서 Vx, Vy는 블록 벡터의 x성분, y성분을 그리고 Vxk, Vyk는 그것의 4근방 블록 벡터를 나타낸다.

3.2 움직임 벡터의 시간적 특성

움직이는 물체는 가속도에 의하여 그 다음의 움직임을 어느 정도 예측할 수 있다. 움직임이 불규칙하더라도 운동의 법칙에 의하면 움직임의 방향이 변하기 위해서는 속도가 0(zero)이 된 후에 다른 방향으로 가속이 이루어진다. 보통 처리하는 영상의 경우 프레임 사이의 간격이 짧기(초당 30-60 frame) 때문에 속도의 변화를 감지할 수 있고, 물체의 움직임이 있으면 프레임간의 시간간격이 짧기 때문에 그 블록의 속도를 알 수 있다. 속도를 안다면 현재 블록의 움직임 정도를 추정할 수 있다. 그러나 하나의 움직임 벡터로는 속도의 증감을 알 수 없으므로 두 개의 속도를 참조해야 한다. 그래서 두 프레임의 움직임 벡터를 필요로 한다. 탐색영역의 결정은 다음의 과정에 의해 처리된다.

a) 블록의 속도 v는 식 (6)과 식 (7)처럼 구할 수 있다.

$$\overline{v_{t-2}} = \frac{1_{t-2} - 1_{t-3}}{\triangle_t} = \frac{MV_{t-1}}{\triangle_t} \quad (6)$$

$$\overline{v_{t-1}} = \frac{1_{t-1} - 1_{t-2}}{\triangle_t} = \frac{MV_{t-1}}{\triangle_t} \quad (7)$$

b) 속도의 비율을 계산하면 다음과 같다.

$$dv = \left|\frac{v_{t-1}}{v_{t-2}}\right| \quad (8)$$

움직임의 추정은 움직임의 크기에 비례한다고 할 수 있으므로 식 (9)의 dv에 비례하여 다음과 같이 구해진다.

$$S = dv.MV_{t-1} \quad (9)$$

위의 과정을 수평 (x축), 수직(y축) 방향으로 향하여 Sx, Sy를 구한 후 탐색영역을 이동시킨다. 이렇게 정해진 탐색 영역을 바탕으로 움직임 추정이 행해진다.

4. 움직임 벡터를 이용한 비디오 특징 추출

4.1 움직임 영역 검출

움직임 벡터는 사용하는 목적에 따라 화소 단위, 블록 단위, 프레임단위로 구한다. 블록단위로 구할 때는 블록의 크기를 크게 하면 전송해야 하는 움직임 벡터의 비트율이 떨어지지만, 블록 안에 여러 가지 움직임이 있을 때는 움직임의 추정 신뢰도가 떨어진다. 반면 블록의 크기를 작게하면 움직임의 신뢰도가 높으나 전송시 비트율이 높아진다는 단점이 있다. 이런 점을 고려할 때 8×8 블록 또는 16×16 블록크기를 많이 사용한다. 탐색영역의 범위는 이론적으로 -∞에서 +∞까지 해야 하지만, 움직임을 추정하는데 소요되는 시간, 하드웨어의 복잡도, 인간이 따라갈 수 있는 움직임의 정도 등을 고려하여야 한다. 다음 그림 3은 움직임 검출 영역이동 알고리즘이다.

```
/* 입력 : Image */
if (움직임 블록인가를 판별)
{
 if ((Vt-1 Vt-2 < 0) and (|Vt-1 - Vt-2| > T1))
 {
   은위 방향 검출 (3×3);
   현재 블록의 MV를 우위 방향의 MV로 대체;
 }
 else if ((Vt-1 Vt-2 > 0) and (|Vt-1 / Vt-2| > T2))
 {
   이동량 제한;
 }
   이전 프레임의 MV로 현재 블록의 속도 추정;
   탐색 영역 이동;
}
움직임 추정;
전송;
```

그림 3 움직임 검출 영역이동 알고리즘

그림 3의 알고리즘에 의해 움직임 영역이동이 결정되면 움직임을 검출하는 방법으로는 블록 정합 방법을 사용하고 움직임 추정시 사용한 평가함수는 MAE이다. BMA의 여러 방법 중 본 연구에서는 전역 탐색법을 사용하고, 탐색 범위는 ±15로 제한하였으며 탐색은 정수 화소(integer pixel)까지 한다.

전역 탐색은 우선 현재 프레임 F_t를 M×N 크기의 블록으로 나눈다. F_t의 현재블럭 $B_t(i,j)$를 이전 프레임의 탐색영역 R_{t-1}에서 d_x, d_y 만큼 이동시킨 $B_{t-1}(i+d_x, j+d_y)$와의 평가치를 구한다. 여러 위치의 평가치중 최적의 블록을 선택한다. 즉 전역 탐색은 모든 탐색 영역에서 식 (10)의 평가 함수가 최소로 되는 위치를 찾아 그 위치를 움직임 벡터로 한다.

$$d(i,j) = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N} G[B_t(m,n) - R_{t-1}(m+d_x, n+d_y)]$$

$$-P \le d_x \le P, -P \le d_y \le P \quad (10)$$

여기서 G(·)는 예측 오차를 구하는 함수이고, B_t는 현재 프레임의 블록, R_{t-1}은 이전 프레임의 탐색영역, P는 최대 이동 변위를 나타낸다. 식 (10)에서 예측 오차가 최소가 되는 위치를 d_x, d_y라 할 때, 움직임 벡터 (V_x, V_y)는 $V_y = d_y$가 된다.

4.2 움직임 벡터 연산 알고리즘

본 논문의 알고리즘 그림 4는 연속되는 비디오 열의 프레임을 입력받아 그 프레임을 일정한 블럭(8×8)로 나누어서 n-1번째 프레임에서 블럭이 이동할 수 있는 영역(15×15)안에서 가장 근사한 블록을 찾는 알고리즘이다. 찾아진 가장 근사한 블럭이 n-1번째 프레임에서 n번째 프레임으로 시간의 경과에 따라 블럭이 이동한 움

```
/* 입력 : n-1, n 프레임(x_size, y_size)
   출력 : 각 블럭당 모션벡터 */
for(이미지를 블럭(8×8)의 크기씩)
{
   for(n번째 프레임의 지정한 블럭과 n-1번째 프레임 중 움직임 추정블럭 비교) {
      sum = 0;
      if(비교 블럭이 이미지 범위내에 있으면)
      {
         for(블럭 내의 각 픽셀에 대해서)
         {
            sum += MSE;
            if(sum이 최소값보다 크면) break;
         }
         if(sum <= 최소값)
         {
            x축 motion vector = ux;
            y축 motiom vector = uy;
            최소값 = sum;
         }
      }
   }
}
```

그림 4 움직임 벡터 연산 알고리즘

직임 벡터이다. 가장 근사한 블럭을 찾는 키는 앞에서 제시한 MSE를 사용하고 블럭들의 비교에서 최소값을 찾는 방법을 사용하여 움직임 벡터를 구한다.

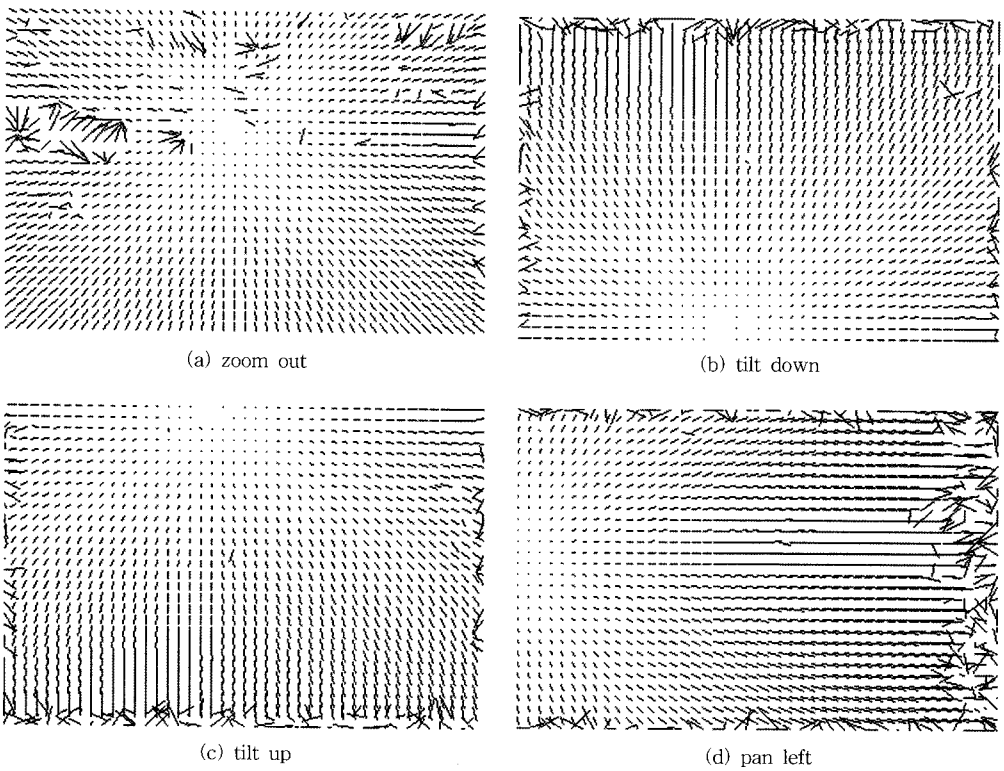
4.3 움직임 벡터의 카메라 동작의 추정

카메라 동작을 검출하기 위해 제공되는 특별한 특징은 컴퓨터 영상에 근원을 둔 빛의 흐름이다. 따라서 영상 움직임은 일정한 방향을 가지게 된다. 카메라 동작에 기인한 움직임 벡터는 공간적으로 나타나며 한 쌍의 프레임에 대한 각 블록의 좌표치가 계산된다. 계산된 공간적 패턴의 특징을 보면 움직이는 물체를 제외한 모든 물체의 방향이 어느 일정한 방향으로 향한다는 것을 볼 수 있다. 예를 들면 그림 5(d)에서 처럼 패닝이나 트래킹의 경우 빛 흐름의 패턴에서의 움직임 벡터는 왼쪽이나 오른쪽으로 향하며, 그림 (b)의 틸팅과 부밍은 위쪽이나 아래쪽으로 향한다. 또한 그림 (a)의 주밍의 경우 움직임 벡터는 중심을 향하거나 중심 바깥쪽으로 향한다. 즉, 패닝 및 트래킹하는 동안 벡터의 방향은 일정하며, 움직임 벡터의 대부분은 대표 벡터(modal vector)에 평행하다.

패닝이나 트래킹을 추정하기 위한 식은 다음과 같이 표현된다.

$$\sum_{k}^{N} | \theta_k - \theta_m | \leq \theta_p \qquad (11)$$

여기서 θ_k는 움직임 벡터 k의 방향이고 θ_m 대표 벡터의 방향이다. 그리고 N은 프레임에서의 움직임 벡터



(a) zoom out (b) tilt down

(c) tilt up (d) pan left

그림 5 카메라동작 결과

의 총 개수이다. $|\theta_k - \theta_m|$는 두 벡터가 정확히 평행일 때 0이 된다. 그러므로 식 (11)을 이용하여 대표 벡터 방향으로부터 모든 움직임 벡터의 변화량을 계산한다. 만약 이러한 변화량이 θ_p보다 작거나 같으면 카메라는 패닝 또는 트래킹이다.

주밍은 인이든 아웃이든 초점을 가지며, 만약 초점이 프레임의 중심에 배치해 있고 물체의 움직임이 없다면 모든 움직임 벡터의 평균은 0이 될 것이다. 그러나 프레임의 한가운데에 초점을 배치하는 것이 그리 쉽지 않은 일이다. 다른 동작과 복합되지 않는 한 영상의 중앙을 기준으로 상하좌우의 움직임 벡터의 방향은 대칭이다. 즉, 모든 행에서 수직요소의 차의 크기는 언제나 개별 요소의 크기를 초과할 것이다. 좌단 또는 우단행에서 움직임 벡터의 수평 요소 또한 마찬가지로 해석할 수 있다. 대부분의 움직임 벡터가 식 (12)와 식 (13)의 조건을 만족할 때 주밍이다.

$$| v_k^{top} - v_k^{bottom} | \geq \max(|v_k^{top}|, |v_k^{bottom}|) \quad (12)$$

$$| u_k^{top} - u_k^{bottom} | \geq \max(|u_k^{top}|, |u_k^{bottom}|) \quad (13)$$

여기에서 v_k^{top}은 k번째 수직요소의 첫 번째 행 값이고, u_k^{top}은 k번째 수평요소의 첫 번째 열 값이다.

4.4 움직임 블록과 배경 블록의 구분 알고리즘

영상의 움직임 벡터를 살펴보면 배경영역에서도 큰 움직임 벡터 영역을 갖는 블록이 존재한다. 그러나 배경영역은 조명의 변화와 카메라의 잡음 등에 의하여 변화할 뿐 움직임이 없기 때문에 움직임 벡터의 특징에 따라 카메라 동작에 의한 배경 영역의 이동인지, 아니면 물체의 움직임인지, 또는 특수 효과에 의한 움직임 벡터의 변화인지에 대한 명확한 구분이 컷 검출 및 인덱스를 위한 특징추출에 대한 오검출을 줄일 수 있으며, 본 논문에서 필요로하는 대표프레임 검출 또는 부대표프레임 검출, 특징추출등 필요한 데이타를 인덱스 필드로 사용할 수 있다.

배경 블록의 결정은 이전 프레임에서 같은 위치의 블록이 움직임 벡터와 주변 8개 블록의 움직임 벡터를 조사하여 결정한다. 즉 이전 프레임 블록에 대한 식 (14)와 같이 9개 블록의 움직임 벡터의 평균을 구한다.

$$M = \frac{1}{3}\frac{1}{3}\sum_{m=-1}^{1}\sum_{n=-1}^{1}|MV_{t-1}(i-m, j-n)| \quad (14)$$

여기서, $MV_{t-1}(i,j)$는 t-1 프레임의 i,j 위치 블록의 움직임 벡터이다. 식 (14)에서 구해진 값을 식 (15) 처럼 임계치와 비교하여 배경 블록과 움직임 블록을 구분한다.

$$\begin{aligned} &M \geq T : \text{움직임이 있는 블록} \\ &M \leq T : \text{배경 영역의 블록} \end{aligned} \quad (15)$$

그러나 배경 영역에서 움직임 블록이 단독으로 존재하는 경우가 발생할 수 있으므로 식 (16)처럼 이전 프레임에서 같은 위치에 있는 블록의 움직임 벡터에 같은 임계치를 적용하고 배경영역에 존재하는 움직이는 블록을 고려한다. 따라서 문턱치보다 크면 움직임이 있는 것으로 간주한다.

$$MV_{t-1(i,j)} \geq T : \text{움직임 블록} \quad (16)$$

움직임이 있다고 판별된 블록은 다음의 과정에 의하여 탐색 영역의 추정이 이루어지고, 그 탐색 영역을 기준으로 현재 블록에 대해 움직임 추정이 행해진다. 배경 블록은 탐색 영역의 이동없이 현재 위치를 기준으로 하여 움직임 추정이 행해진다. 즉, 배경이라고 판단된 블록은 BMA 방법으로 움직임 추정을 한다.

5. 실험 및 검토

5.1 실험환경

본 연구에서 컴퓨터 시뮬레이션에는 CCIR 601 테스트 시퀀스(sequence)인 'table tennis', 'mobile', 'Flower Garden', 'Football' 영상을 대상영상으로 사용한다. 'Table tennis'는 탁구 치는 모습이며 탁구공을 따라 카메라가 확대하는(zoom in)모습이다. 'Flower Garden' 영상은 카메라의 트래킹(tracking)현상이 있는 영상으로 꽃이 핀 정원의 모습이다. 'mobile' 영상은 장난감이 움직이는 영상으로 카메라는 고정이다. 'Football' 영상은 미식축구를 하는 운동선수들로 구성되어 움직임이 비교적 크고 빠르며 배경은 고주파 성분이 많으나 정적이다. 그리고 대상영상신호들은 모두 720 × 480 크기의 4:2:2 형식의 디지털 신호로 휘도 신호 Y와 색 신호 Cb, Cr로 구성되어 있다. 본 연구에서는 휘도 신호만을 대상으로 한다. 세 영상 모두 비월 주사(interacted scanning)된 신호이기 때문에 홀수 번째 필드만 세로축으로 보간하여 순차 주사(progressive scanning)된 형태로 바꾸어 그것을 한 프레임으로 하였다. 보간에 사용된 필터는 MPEG에서 영상의 해상도 변화에 사용되는 것을 사용했다. 본 시뮬레이션에 사용된 필터의 계수는 그림 6에 보인다. 그림 6의 "//"연산은 정수 나눗셈 후 반올림함을 나타낸다. 이 필터는 표준화된 필터가 아니며 다른 필터가 사용될 수 있다. 본 연구에서는 각 영상을 1 GOP(group of picture)개념으로 15 프레임씩 처리하였으며, Win2000 Server에서 수행되었다.

또한 시뮬레이션에 사용된 비디오 구성은 lin=lout, tin=tout인 경우이다. 구체적으로 그림 7과 같이 'mobile'

-29	0	88	138	88	0	-29	// 256

그림 6 보간 필터

의 후미 9프레임과 'flower garden' 의 선두 9프레임을 오버랩 시켰다. 'table tennis'와 'mobile' 사이 'flower garden'과 'football' 사이에는 단순 비디오 컷이 존재한다(그림 7).

5.2 움직임 벡터 검출

본 실험은 위에서 제시한 실험환경에 의한 것이며, 본 논문에서 제시한 "움직임 영역검출"과 "움직임 벡터 연산 알고리즘"에 의한 것이다.

그림 8의 (a)(b)(c)는 원 영상에 보간 필터를 처리한 그림이다. 그림 8의 (a)는 시뮬레이션 데이타의 프레임 1에서 프레임 3으로의 움직임 벡터로 이 그림에서는 도트의 상태로 보아 미세하게 우측으로 그어져 있으므로 카메라가 우측으로 이동(패닝)되고 있음을 알 수 있다. 그림 8의 (b)는 프레임 30-32로의 움직임 벡터로 도트의 상태로 보아 카메라는 그림 8의 (a)보다 빠르게 이동(패닝)되고 빠른 물체의 움직임만 있다는 것을 알 수 있

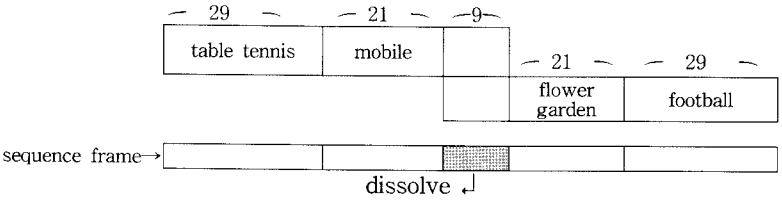
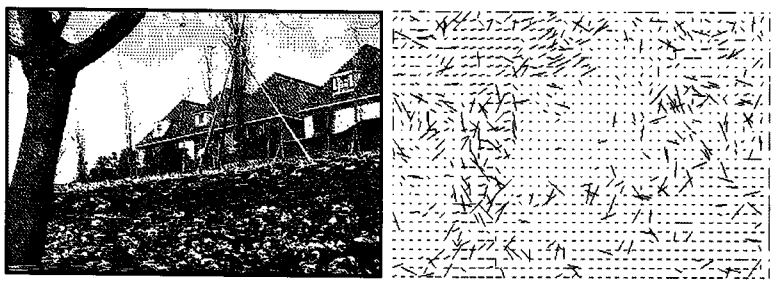
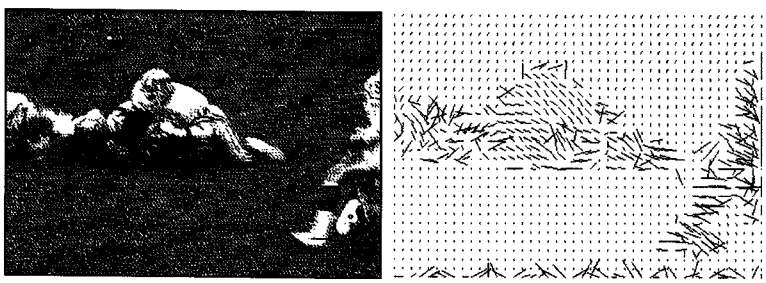


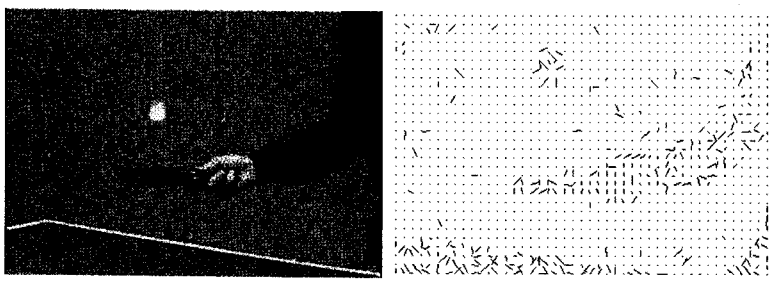
그림 7 특수효과 검출을 위해 재구성한 비디오 시퀀스



(a) 영상 "플라우어 가든"과 카메라 동작(camera panning)



(b) 영상 "미식축구"와 카메라 동작(camera fix)



(c) 영상 "탁구게임"과 카메라 동작(camera zoom-in)

그림 8 보간 필터 처리 후의 영상과 움직임 벡터

다. 그림 8의 (c)는 시뮬레이션 데이타의 프레임 62-64에서 프레임의 움직임 벡터를 나타낸다. 이 그림에서는 도트의 상태로 보아 프레임의 중심으로 몰려(zoom in) 있기 때문에 카메라 동작을 쉽게 판별할 수가 있고 부분적인 움직임이 발생되고 있다고 알 수 있다. 이 밖에 움직임 벡터를 이용한 객체의 검출, 컷 검출, 카메라 동작 검출(패닝, 트래킹, 디졸부, 페이드등)을 검출할 수 있다.

5.3 인덱스를 위한 움직임벡터 매개변수

표 1은 실험한 결과에서 나온 매개변수 값들을 나타낸 것이다. 여기에서 count는 패닝/트래킹을 판별하기 위한 값으로 임계치를 넘는 움직임 벡터의 수이다. Vcount와 Ucount는 주밍을 판별하기 위한 값으로 식(6~8)결과로 얻어진 것이다. 특이할 만한 것은 패닝/트래킹과 주밍을 판별하기 위한 변수들이 그 자체만으로는 완전한 판별이 어려우며 표 1에서 보이는 변수들을 이용해야 완전한 판별이 가능했다. 결과를 보면, football에서는 카메라 동작이 없었으며, flower garden에서는 트래킹이 사용되었고, table tennis에서는 주밍이 사용되었다는 것을 인식해낼 수 있었다. 여기에서 나온 결과의 브라우징을 인덱스로 활용할 수 있도록 했다. 표 1은 본 논문에서 제시한 알고리즘에 의해 산출된 컷이 발생된 부분의 수치 데이타 값이다. X, Y축으로 8×8블록으로 계산된 좌표값으로 소숫점 이하는 생략했다.

5.4 움직임 벡터를 기반으로 한 점진적 씬 전환의 검출

실험에 사용된 비디오 시퀀스는 실험데이타 그림 7의 구성에 의한 것이다. 그림의 특징은 29-30프레임에서 장면전환이 발생하고 50-59프레임 사이에서는 특수효과(디졸브)가 발생하고, 79-80프레임에서는 장면전환이 발생하는 실험 데이타이다.

그림 9는 연속되는 프레임 화소단위 비교 프로그램으로서 프레임 29와 프레임 79에서 피크를 기록했다. 그러나 22번 프레임과 30번 프레임 사이의 특수효과(디졸브)는 검출하지 못했다.

그림 10은 세기 총합에 의한 유사도 측정의 결과 값이다. 마찬가지로 특수효과(디졸브)는 검출하지 못했다.

그림 11은 빛의 세기에 연속하는 2프레임의 히스토그램 차를 계산한 결과 값인데 역시 29번과 79번에서 피크를 보였고 특수효과(디졸브)가 검출되는 것처럼 보였지만 임계치 적용에는 어려움이 있어 보인다.

그림 12는 이중비교법에 의한 컷 검출의 결과를 보인 것이다. 이 그림에서 특수효과(디졸브)를 정확히 검출해냈다. 하지만 그림에서와 같이 특수효과 부분에서의 정확도는 미흡하다는 것을 알 수 있다.

그림 13은 본 논문에서 제시한 움직임벡터의 시간성과 공간성을 동시에 적용한 결과를 보여준다. 본 논문에서는 연속된 프레임에서 움직임벡터의 방향성과 시간성

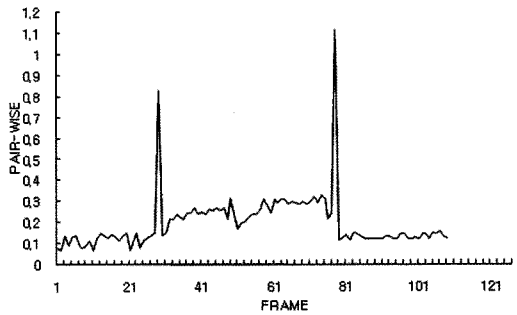


그림 9 화소단위 비교

표 1 움직임 벡터의 실험결과

parameter / laplace data	Average Orientation	Average Deviation of Orientation	Modal Orientation	count	Vcount	Ucount
data 1	-8.084444	38.407486	0	749	33	15
data 10	1.497778	43.625495	0	713	44	17
data 20	23.664444	54.372777	0	703	45	20
data 30	15.971111	44.237587	0	832	45	18
data 40	11.722963	45.942398	0	764	44	17
data 50	-2.580000	92.228148	135	88	22	22
data 60	82.424444	19.426301	90	1022	6	4
data 70	83.772593	19.743608	90	992	8	6
data 80	82.467407	19.085961	90	1083	7	0
data 90	82.966667	19.025975	90	1030	8	5
data 100	2.914074	90.224213	45	86	24	17
data 110	17.292593	38.333394	0	1118	45	17
data 120	0.000000	0.000000	0	1350	45	30
data 130	0.000000	0.000000	0	1350	45	30
data 140	8.329630	94.352027	-90	133	17	11
data 149	2.573333	79.138528	0	198	17	15

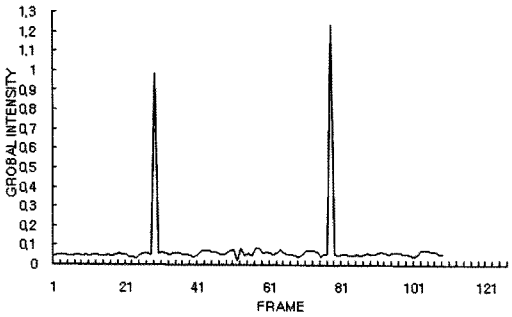


그림 10 세기총합 유사도

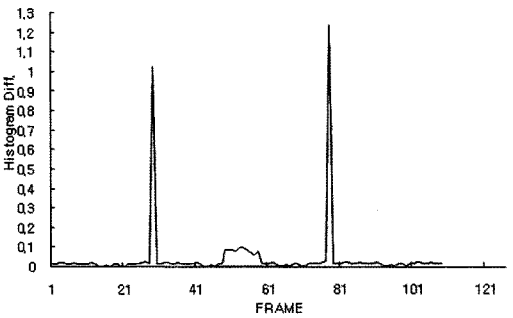


그림 11 히스토그램 비교법

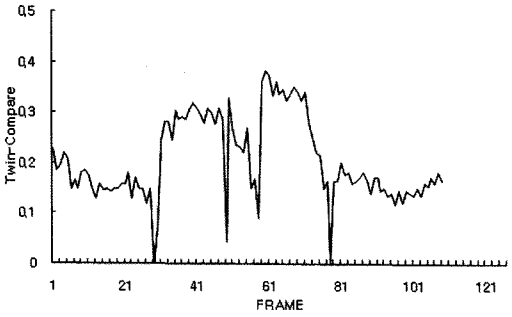


그림 12 이중비교법

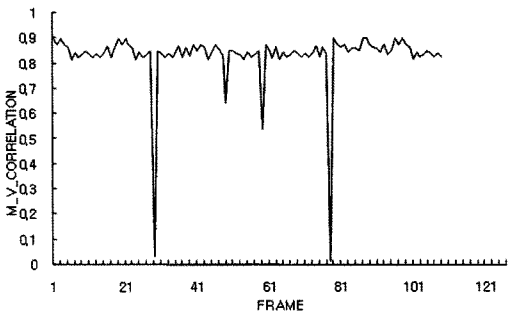


그림 13 움직임벡터의 시공간상관도

(크기)를 동시에 적용한 후 상관도를 이용한 히스토그램이다. 결과 컷 검출과 특수효과(디졸브) 검출도 우수함을 보였다.

본 논문에서의 실험은 디졸브가 들어있는 비디오에 대한 점진적 씬 전환의 검출을 위한 측정치로 종래에는 화면의 밝기를 사용했지만 본 논문에서는 움직임 벡터를 사용하였다. 실험 과정은 다음과 같다.

첫째: 움직임 벡터는 한 프레임당 16*16 블럭으로 분할하여 움직임 벡터를 탐색하고 이를 조합하여 전체를 계산했다. 또한 움직임 벡터는 방향과 크기로 구성하였다.

두 번째로: 점진적 씬 전환의 검출은 각 프레임에 대한 방향과 크기를 기반으로 교차율과 상관계수를 적용하여 히스토그램으로 변환하여 검출했다.

5.5 기존의 방법과 비교

이상의 실험에서 본 논문에서 제안한 움직임 벡터 비교에 의한 점진적 씬 전환 검출 방법이 밝기에 기반을 둔 방법보다 그 검출 능력이 떨어지지 않는다는 것을 보여주었다(표 2 참조). 따라서 굳이 점진적인 씬 전환의 검출을 위해 이중검출법과 같은 방법을 사용하고 또 움직임 정보를 얻기 위해 다시 움직임 벡터를 연산하는 비효율적인 방식을 사용할 필요가 없음을 알 수 있다.

본 논문에서 제안된 방법은 움직임의 특징을 추출할 수 있었으며 일반 컷은 물론 점진적 컷도 찾아낼 수 있었고 대표프레임의 검출도 가능함을 알 수 있다. 또한 압축알고리즘 H.264 등에 응용할 수 있음을 알 수 있었다.

그림 14는 본 논문에서 제시한 알고리즘으로 1) 검출 2) 대표프레임 검출 3) 내용기반 검색 시뮬레이션의 기능을 보이는 실험을 했다. 그림에서 위쪽은 대표프레임 검출이고 아래 부분과 히스토그램 중간 부분은 대표프레임에 따른 씬(scene)이다.

표 2 실험영상에 대한 기존의 방법과 성능비교

Ability / Methods	General Cut	Gradual Transition	Ability of Identification	Processing/sec	USE for MPEG
Pair-wise	2/2	0	×	9.132	×
Intensity	2	0	×	7.475	×
Histogram Diff	2	0	×	7.016	×
Twin-compare	2	1	○	13.502	×
Our method	2	1	○	12.156	○

표 3 영화 "쥬라기 공원"에 대한 기존의 방법과 성능 비교(9562 frame)

Ability / Methods	General Gut	Gradual Transition	Ability of Identification	Processing/sec
Pair-wise	12/12	1	×	326.25
Intensity	12/12	0	×	245.21
Histogram Diff	12/12	0	×	223.51
Twin-compare	12/12	3	O	525.62
Our method	12/12	3	O	485.73

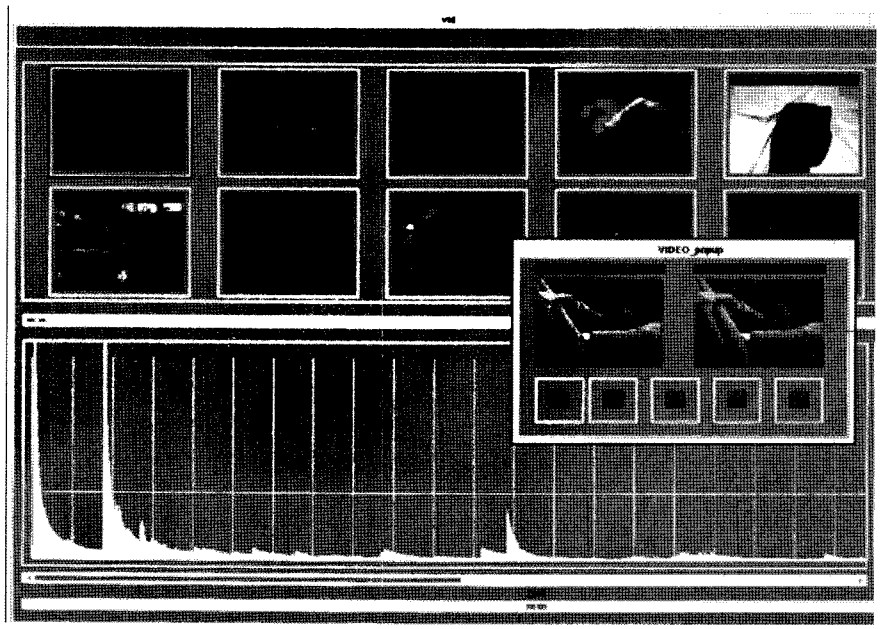


그림 14 영화 "쥬라기 공원" 모의실험 결과

6. 결 론

본 연구에서는 움직임 추정 방법 중 블록 단위로 움직임을 추정하는 기존의 여러 가지 기법과 관련하여 전역탐색기법에 대해 살펴보고 이를 응용한 새로운 알고리즘을 개발하였으며 실험대상으로 사용한 영상들과 실험 환경에 대하여 서술하였다. 동영상 움직임의 궤적을 고려한 움직임 추정에 관한 연구로 동영상이 내포하고 있는 각 구성요소들의 움직임은 정지해 있는 배경, 그리고 매우 느린 움직임으로부터 매우 빠른 움직임에 이르기까지 그 움직임의 크기에 있어서 많은 분포를 보인다. 따라서, 제한된 탐색영역을 유지한 채 움직임 추정을 행할 때, 만일 빠른 움직임이 발생할 경우 큰 예측 오차로 인한 오검출을 유발할 수 있다. 본 연구는 움직임 추정 시 제한된 탐색 범위로 인해 발생되는 점들을 해결하여 움직임의 추정보상 효율을 향상시키고, 고속의 알고리즘을 제시함으로써 효과적인 움직임 추정 기법을 제시했다. 또한, 블록 정합법의 경우 여러 가지 제약 조건과 함께 블록 정합법에 의해 얻어지는 벡터가 실제 움직임과 상이한 경우도 발생한다. 본 논문에서는 그 대안으로서 시간적, 공간적 인접 블록의 움직임 벡터를 이용하여 좀 더 정확한 움직임 벡터의 예측하는 기법을 제안하였고, 제시된 알고리즘을 바탕으로 움직임 보상을 위한 움직임 추정 기법을 고찰하고 이를 적용한 결과를 알아보았다.

향후로는 본 논문에서 연구된 결과물을 이용하여 압축과 전송에 응용되는 장면전환에 효율적인 비트율 제어 기법에 관한 연구를 하겠다.

참 고 문 헌

[1] Aigrain, P., and Joly, P., "The automatic real-time analysis of film editing and transition effects its applications," Computer & Graphics 18, 1, 1994. pp. 93-103.

[2] S. W. Smoliar, and H. Zhang, "Content-Based Video Indexing and Retrival," IEEE Multimedia 1994 summer, pp. 63-72.

[3] A. Hampapur, R. Jain, T. E. Weymouth, "Indexing in Video Databases," SPIE, Vol. 2420, 1995, pp. 292-306.
[4] Jae-Hyun Lee, Yeun-Sung Choi and Ok-Bae Jang, "Gradual Cut Detection Low Level Vison for Digital Video," SPIE, Vol. 2952, pp. 683-688, 1996.
[5] A. Akutsu et al., "Video Indexing Using Motion Vectors," Proc. SPIE Visual Comm. and image Processing 92, SPIE, Bellingham, Wash., 1992, pp. 1522-1530.
[6] F. Arman, R. Depommier, A. Hsu and M. Y. Chiu, "Content-based browsing of video sequences," In Proc. ACM Multimedia, '94, pp. 97-103.
[7] D. Manoranjan and V. V. Vinod, "Video Segment Activity," ISO/IEC JTCI/SC29/WG11, Lancaster, UK Jan. 1999, p. 627.
[8] V.V Vinod and H. Murase, "Video Shot Analysis Using Efficient Object Tracking," Proceedings of IEEE Conf on Multimedia Computing Systems, Jun, 1997, pp. 501-508.
[9] Akutsu, A., and Tonomura, Y. Video topography: an efficient method for camerawork extraction and motion analysis, In proc. ACM Multimedia, 94, ACM press, pp. 349-356, 1994.
[10] M, La Cascia, E. Ardzzone; JACOB; "Just a content-based query system for video databases," Proc. of ICASSP, '96, May 1996, pp. 7-10.
[11] G. Ahanger, T. Little, "Data Semantics for Improving Retrieval Performance of Digital News Video Systems," Proc. of IEEE Transactions on Knowledge and Data Engineering, Vol. 13, 2001, pp. 352-360.
[12] K. Otsuji, Y. Tonomura and Y. Ohba, "Video Browsing Using Brightness Data," Proc. SPIE Visual Comm. and Image Processing 91, SPIE, Bellingham, Wash., Vol. 1606, 1991, pp. 980-989.

이 현 창

1994년 전북대학교 전자계산학과 졸업(이학사). 1997년 전북대학교 대학원 정보과학과 졸업(이학석사). 1994년~2002년 전북은행 입행 및 전산정보팀 근무. 2001년~현재 전북대학교 대학원 컴퓨터통계정보학과 박사과정. 2002년~현재 전북은행 호성동지점 근무. 관심분야는 멀티미디어 정보검색, HCI, 멀티미디어 전송, 이미지프로세싱 등

이 재 현

1986년 호원대학교 전자계산학과 졸업(이학사). 1988년 조선대학교 컴퓨터공학과 졸업(공학석사). 1997년 전북대학교 컴퓨터과학과 졸업(이학박사). 1997년~현재 벽성대학교 컴퓨터경영정보과 교수 관심분야는 멀티미디어 정보검색, HCI 멀티미디어 데이타베이스, 멀티미디어 전송, 임베디드 멀티미디어 시스템 등

장 옥 배

1973년 고려대학교 졸업(학사, 석사). 1988년 산타바바라대 대학원(Ph. D.). 1974년~1980년 죠지아 주립대. 오하이오 주립대 박사과정 수료. 1980년~현재 전북대학교 자연과학대학 전자정보공학부 교수. 관심분야는 소프트웨어 공학, 전산교육, 수치해석, 인공지능 등