Trie 구조를 이용한 비디오 인덱스 생성 및 검색

(Video Index Generation and Search using Trie Structure)

현 기 호 † 김 정 엽 †† 박 상 현 †††

(Ki-Ho Hyun) (Jeong-Yeop Kim) (Sanghyun Park)

요 약 비디오 데이타베이스에서 유사도 정합은 비디오 클러스터링과 비디오 라이브러리 등과 같은 많은 새로운 응용분야에서 중요성이 증가하고 있다. 대용량 데이타베이스에서 효과적인 접근을 제공하기 위하여 다양한 공간과 시간에 대한 특징치를 이용한 비디오 인덱싱 분야의 많은 연구노력이 있어왔다. 그러나 대부분의 기존 방법들은 순차적인 정합방법 또는 메모리 기반의 역 파일 기법 등에 의존하므로 대용량 데이타베이스에는 적합하지 않다. 이러한 문제를 해결하기 위하여 본 논문에서는 효과적이고 스케일 조정 가능한 인덱싱 기법을 제안하기 위하여 문자열 정합을 위해 제안된 trie를 인덱스 구조로 이용하였다. 인덱스 구성을 위하여 윈도우 순서 휴리스틱을 이용하여 각 프레임을 기호 시퀀스로 변환하고 기호 시퀀스의 집합으로부터 디스크 상주 trie를 구성하였다. 질의 처리를 위하여 trie 상에서 깊이-우선 검색과 시간 축 분할을 실시하였으며, 제안한 방법의 성능을 검증하기 위하여 실제와 합성 데이타 집합에 대한 실험을 수행하였다. 제안한 방법은 지속적으로 순차적 스캔 방법보다 우수한 성능을 보였고, 성능이득은 대용량 비디오 데이타베이스에서도 유지되었다.

키워드 : 인덱싱 기술, 유사도 정합, 비디오 데이타베이스

Abstract Similarity matching in video database is of growing importance in many new applications such as video clustering and digital video libraries. In order to provide efficient access to relevant data in large databases, there have been many research efforts in video indexing with diverse spatial and temporal features. However, most of the previous works relied on sequential matching methods or memory-based inverted file techniques, thus making them unsuitable for a large volume of video databases. In order to resolve this problem, this paper proposes an effective and scalable indexing technique using a trie, originally proposed for string matching, as an index structure. For building an index, we convert each frame into a symbol sequence using a window order heuristic and build a disk-resident trie from a set of symbol sequences. For query processing, we perform a depth-first search on the trie and execute a temporal segmentation. To verify the superiority of our approach, we perform several experiments with real and synthetic data sets. The results reveal that our approach consistently outperforms the sequential scan method, and the performance gain is maintained even with a large volume of video databases.

Key words : indexing technique, similarity matching, video database

1. 서 론

최근 멀티미디어 데이타베이스에서 비디오 데이타 양의 증가는 매우 크다. 비디오 데이타베이스의 유사도 정합은 비디오 클러스터링과 디지털 비디오 라이브러리 등과 같은 새로운 응용분야에서 중요성이 점차 증가하고 있다. 대용량 데이타베이스에서 관련 데이타에 접근하는 효과적이고 효율적인 방법을 제공하기 위하여 색상(color), 무늬(texture), 의미(semantic), 움직임(motion) 등의 특징치를 이용하여 비디오 데이타베이스를 인덱싱하는 많은 연구 노력이 행해져 왔다[1-8].

기존의 많은 비디오 인덱싱 방법들은 데이타베이스에

† 비 회 원 : 영산대학교 컴퓨터정보공학부 교수
khhyun@mail.ysu.ac.kr
†† 비 회 원 : 영산대학교 멀티미디어공학부 교수
neocopy@mail.ysu.ac.kr
††† 종신회원 : 포항공과대학교 컴퓨터공학과 교수
sanghyun@postech.ac.kr
논문접수 : 2002년 3월 22일
심사완1 : 2003년 3월 21일

서 유사한 비디오를 추출하기 위하여 순차적 스캔방법에 의존하였다. Mohan[9]과 Vailaya 등[10]은 영상내용과 영상 움직임을 결합하고, 행동 유사도를 이용한 비디오 시퀀스 정합방법을 제안하였다. Lienhard[11] 등과 Sanchez[12]등은 유사한 상업용 비디오 클립을 검출하기 위하여 색상 히스토그램의 주성분과 색상 일관성 벡터를 사용하였다. Adjeroh[13] 등은 비디오 시퀀스에 대한 vstring 표현을 채용하고, 유사도 표시인자로 vstring 편집거리를 소개하였다. Ardizzone[1] 등은 각 16x16 부분 이미지로부터 단일 MPEG 움직임벡터를 추출하였고, Meng과 Chang[5]은 카메라의 줌과 팬 등과 같은 저수준 움직임 특징치를 사용하였다. 앞에서 언급한 모든 접근방법은 순차적 스캔방법을 사용하였으며 데이타베이스의 용량이 커질수록 성능이 저하된다. Squire[14] 등은 특징치 기반 영상 추출을 위한 역 파일 기술의 사용을 제안하였고, Hampapur[3] 등은 역 파일 기술을 미디어 추적에 적용하였다. 두 방법 모두 역 파일을 질의 처리(query processing)시 주 기억장치에 보관하는 방법을 사용하나 대용량 비디오 데이타베이스에는 적합하지 않다.

본 논문에서는 스트링 정합을 위하여 제안된 trie[15]를 인덱스 구조로 사용하여 대용량 비디오 데이타베이스의 유사도 정합을 위한 스케일 조정이 가능한 효과적인 인덱싱 기법을 제안하였다. 모든 비디오 프레임들은 동일한 숫자의 창으로 분할하고, 인덱스를 구축하기 위하여 다음과 같은 과정을 거친다: 1) 각 창마다 특징치를 추출하여 개별적인 기호로 변환하고, 2)각 프레임을 윈도우 순서 휴리스틱(window order heuristic)을 이용하여 기호 시퀀스로 변환하고, 3)기호 시퀀스 집합으로부터 디스크 상주 trie를 구축한다. 질의 처리를 위한 과정은 다음과 같다: 1)기호 시퀀스를 작성하기 위하여 인덱스 구축의 첫 두 단계를 프레임별로 실행하고, 2) trie 상에서 깊이-우선(depth-first) 검색을 시행하여 각 목표 프레임에 가장 가까운 데이타 프레임을 검출하고, 3)시간 축에 대한 분할을 실시하여 목표 비디오와 유사한 비디오 또는 비디오의 일부를 검출한다.

2. 비디오 정합을 위한 특징추출

비디오 검색 연구의 목적은 비디오 데이타베이스를 내용으로 검색을 할 수 있도록 하는데 있으며, 데이타베이스, 정보검색과 비전 이해(vision understanding)와 영상처리 분야까지 다양하게 확대되고 있다. 이는 대부분 특징기반의 비디오 검색 방법을 사용하며 데이타베이스분야는 질의를 정의하며 효율적인 검색 메카니즘을 연구하고, 비전 해석과 영상처리 분야는 원영상 및 비디오 처리와 관련된 양질의 특징을 추출하여 비디오 데이타베이스의 비디오 단편들을 특징벡터로 데이타베이스를 구성하고 비디오 인덱스로서 검색한다.

특히 비디오 인덱싱 분야의 연구는 정합의 정확성과 대용량 데이타베이스 처리를 만족해야 한다. 정합을 위해서는 비디오의 다양한 특징을 포함하는 다차원 인덱싱 분야가 활발히 연구되고 있으며, 본 논문에서는 특징기반의 인덱스로서 대용량 데이타베이스를 처리하기위한 방법을 제시하였다. 기존의 방법은 소량의 데이타베이스를 순차적 스캔 방법에 의해 유사도 정합을 하였다. 특히 대규모 데이타베이스와의 정합을 처리하는 문제는 현재 많이 연구되고 있다[20].

2.1 움직임 특징치

움직임은 비디오 시퀀스에서 필수적인 요소이며 이의 해석을 통하여 시간 축에 대한 독립적인 정보를 추출할 수 있다[16,18]. 움직이는 물체의 변위 벡터를 추정하는 알고리즘은 광 흐름(optical flow) 기반, 특징치 기반, 윈도우 정합 기반의 세 가지 형태로 분류된다. 본 논문에서는 제안한 인덱싱 방법에 적합한 윈도우 정합 기반 특징추출 방법을 채택하였다. 예를 들면, F_t와 F_{t+1}을 각각 현재와 연속되는 프레임으로 두고, 프레임들을 다수의 윈도우로 분할한 후, 각 윈도우별로 움직임 벡터를 추정한다. F_t의 한 지점(x,y)에 대한 F_{t+1}에서의 변위를 찾기 위해서는 지점을 둘러싸는 창과 검색 영역 내부의 화소들과의 비교가 필요하다. 변위 D는 현재프레임 t의 i번째 윈도우와 연속프레임 t+1의 j번째 윈도우와의 차이 값을 나타내며, 변위 D중 가장 작은 값을 가지는 i와 j를 구하면 식(1)과 같은 움직임의 방향을 계산할 수 있다.

$$D_{ij} = (W_{t,i} - W_{t+1,j})^2 \quad (1)$$

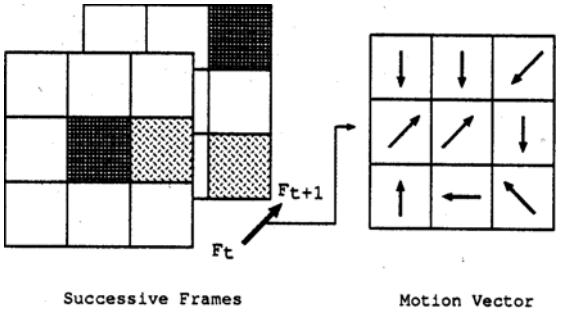
여기서, i=1~N, j=1~N이고, N은 전체 윈도우의 개수이다.

이와 같이 구해진 i와 j의 값을 이용하여 움직임 벡터의 방향을 구하고, q 방향 또는 레벨로 양자화 한다. 창의 움직임 벡터의 크기가 0인 경우는 윈도우에 레벨 0을 할당한다. 움직임 추정 과정을 그림 1에 보였다.

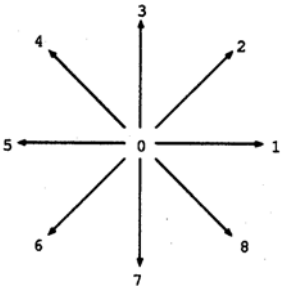
계산된 평균 명암도의 집합은 오름차순으로 정렬되고, 프레임의 각 윈도우에 등위가 할당된다. 그림 2에 오디널 기준 계산 예제를 보였다.

2.2 공간 특징치

비디오 시퀀스의 공간 정보를 추출하기 위하여 오디널 기준(ordinal measure)을 적용하였다[9,19]. 오디널 변수는 학교에서 성적 등급의 집합과 같은 이산적인 순서집합으로부터 얻어지고, 두 값의 비율은 중요하지 않



(a) 움직임 벡터추정



(b) 움직임 벡터의 양자화

그림 1 움직임 추정 과정

으며, 값들의 상대적인 순서만이 의미가 있다 값들 간의 상대적인 순서는 등위(rank)로 표시된다. 등위의 교환은 샘플들을 오름차순으로 정렬하여 얻어지고, [1,2,3,...m]과 같이 정수를 이용하여 번호를 붙이며, m은 샘플의 수가 된다. 오디널 기준은 화소단위의 변화에 덜 민감한 장점을 가지고 있으며, 본 논문에서는 각 윈도우의 평균 명암도 값을 오디널 변수로 사용하였다. 평균명암도 A_i는 윈도우 i에서의 명암도 값을 평균한 것이며 다음 식 (2)와 같이 계산된다.

$$A_i = \frac{1}{N*N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} L(m,n), \qquad (2)$$

여기서 i는 윈도우 번호이고, N은 윈도우의 크기이며 L은 해당화소에서의 밝기 값을 나타낸다

25.3	80.2	39.8
152.3	93.2	125.7
160.3	225.3	230.6

(a) 프레임에서 각 윈도우의 평균 명암도

1	3	2
6	4	5
7	8	9

(b) 프레임에서의 오디널 특징치

그림 2 오디널 기준의 예제

3. Trie 구조를 이용한 비디오 인덱싱 기법

3.1 순서 스캔 방법

기존의 순서 스캔방법(Sequential scan method)은 비디오를 V, 타겟 비디오를 V_t, 데이타(서브) 비디오를 V_s라고 할때 다음과 같은 방법으로 실행된다

1. Database에 저장된 각각의 비디오 V를 읽는다.
2. V로 부터 V_t와 동일한 길이를 가지는 각각의 서브 비디오 V_S을 추출한다.

   여기서 V의 길이가 n이고 V_t의 길이가 m이면 n-m+1개의 서브 비디오를 추출할 수 있다.
3. Distance=(V_t, V_s)를 계산한다.
4. 거리가 distance tolerance ε를 초과하지 않으면 V_s를 answer set에 포함시킨다.

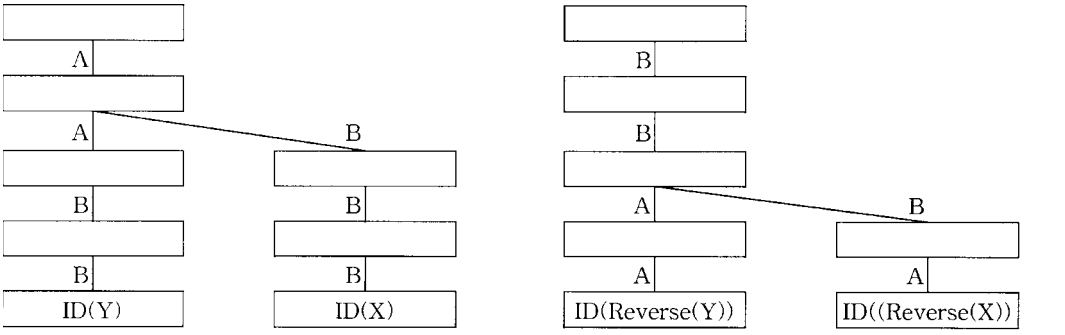
순서 스캔 방법은 데이타베이스에 저장된 모든 비디오에 대해 처리하므로 대용량 데이타베이스에 적합하지 않다. 기존의 비디오 검색은 소량의 데이타베이스를 순서 스캔방법에 의해 유사도 정합을 하였다

Ellis는 대규모 광고 데이타베이스와 방송 단편을 비교하기 위해 룩 업 테이블을 사용하였다 Sanchez등은 키 프레임의 칼라히스토그램의 주성분을 사용하여 순서 스캔 방법을 20여 가지의 광고 데이타베이스를 가지고 실험하였다[12].

3.2 인덱싱을 위한 Trie 구조

Trie는 키워드의 집합을 인덱싱하는데 사용되는 트리 구조이며 내부 노드는 비어있고, 경계에 기호가 저장된다. 루트(root)에서 단말노드(leaf)까지의 경로는 trie에 삽입된 기호 시퀀스를 나타낸다. 일반적으로 종단 노드는 기호 시퀀스의 식별자를 저장한다. trie는 원래 기호 시퀀스에 포함된 모든 정보를 갖고 있으나 공통 접두어는 한번만 저장된다. 그러므로 기호 시퀀스가 다수의 공통 접두어를 포함하고, 공통 접두어의 길이가 길 경우, 압축 가능성이 증가한다. 그림 3(a)는 두 개의 기호 시퀀스인 X=<A,B,B,B>와 Y=<A,A,B,B>로부터 구축된 trie를 나타내고, 그림 3(b)는 Reverse(X)=<B,B,B,A>와 Reverse(Y)=<B,B,A,A>로부터 구축된 trie를 나타낸다.

제안한 인덱스 구축방법은 그림 4에서 보이는 것과 같이 전처리, 기호 시퀀스 생성과 trie 구축의 3단계를



(a) X=<A,B,B,B>와 Y=<A,A,B,B>로부터 구해진 Trie (b) Reverse(X)=<A,B, B,B>와 Reverse(Y)=<A,A,B,B>의 Trie

그림 3 trie 예제

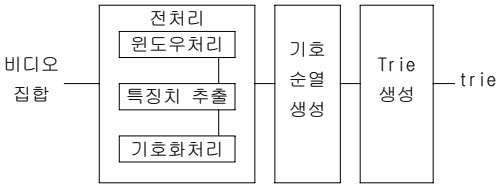


그림 4 인덱스 구축 단계

Window-1 A : 80%	Window-2 B : 71%	Window-3 B : 95%
Window-4 A : 64%	Window-5 A : 54%	Window-6 B : 67%
Window-7 C : 85%	Window-8 C : 43%	Window-9 C : 73%

(a)

3	5	1
7	8	6
2	9	4

(b)

그림 5 9개 윈도우들의 윈도우 읽기 순서

가진다. 인덱스 구축의 전처리는 다음과 같은 윈도우처리, 특징치 추출, 기호화의 과정을 거친다.

단계1 : (윈도우처리) 각 프레임을 지정된 숫자(예:225)의 윈도우로 분할함.

단계2 : (특징치 추출) 각 윈도우로부터 2장에 설명한 움직임 특징치 또는 오디널 특징치를 계산함

단계3 : (기호화) 각 움직임과 오디널 특징치를 해당 기호(corresponding symbol)로 변환함.

두 번째 단계인 기호 시퀀스 생성과정은 각 프레임별로 기호 시퀀스를 생성하기 위하여 프레임의 윈도우들을 미리 지정된 순서로 읽는다. 기호 시퀀스가 더욱 긴 공통 접두어를 포함하면 trie가 더욱 함축적이 된다. 예를 들면, 그림 3의 두 개 trie를 비교하였을 때, Reverse(X)와 Reverse(Y)의 공통 접두어가 X와 Y의 경우보다 길기 때문에 Reverse(X)와 Reverse(Y)의 trie가 더 적은 노드를 가진다. 공통 접두어의 수와 길이를 최대화하기 위하여 다음과 같은 과정을 처리한다 1) 각 윈도우마다 우세한 기호(dominant symbol)를 결정하고, 발생비율을 계산한다. 2) 우세한 기호의 발생비율에 따라 윈도우의 순위를 결정한 다음 이 순위에 의해서 윈도우 읽기 순서가 결정된다. 이것을 윈도우 순서 휴리스틱이라 부른다. 그림 5(a)는 우세한 기호와 발생 비율이 표시된 9개의 윈도우를, 그림 5(b)는 제안한 휴리스틱에 의해 결정된 윈도우 읽기 순서를 보여준다.

마지막 단계로서, Trie 구축을 위하여, 각 프레임에 대한 엔트리를 생성한 후, trie에 삽입한다. 비디오 Vi의 p번째 프레임에 대한 인덱스 엔트리는 (기호 시퀀스, i, p)로 표기한다. 모든 기호 시퀀스가 동일한 길이를 가지고 있으므로 디스크 기반 trie 구축 방법을 쉽게 사용할 수 있다. 인덱스 엔트리를 해당 기호 시퀀스의 순서에 따라 오름차순으로 정렬하고, 각 인덱스 엔트리를 구축중인 trie에 추가한다. 이 방법은 단지 인덱스 엔트리가 삽입되도록 유지하고, 메인 메모리 내부의 각 레벨의 마지막 노드가 제안한 방법이 대용량 비디오 데이타베이스에 적합하도록 만든다.

3.3 질의 처리

목표 비디오가 허용치 ε와 같이 제시되면, 그림 6에 제시된 4단계를 거쳐 가상 유사한 부분 비디오를 검색한다. 전처리, 기호 시퀀스 생성, 인덱스 검색과 시간 축 분할 등의 처리를 하고, 첫 번째 단계동안 각 목표 프레임을 지정된 숫자의 윈도우로 분할하여, 각 윈도우로부터 특징치를 계산하고, 해당기호로 변환한다.

전처리 후의 단계인 인덱스 검색은 기호 시퀀스로 표

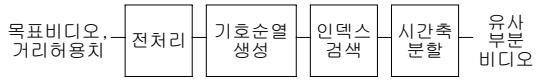


그림 6 질의 처리 단계

현되는 각 목표 프레임에 대하여 trie상에서 깊이-우선 검색을 실시하여, 목표 프레임과 기호의 정합수가 최대인 최적 정합을 찾는다. 두 기호간의 차이가 거리 허용치 ε이내이면 정합된 것이다. 인덱스 검색 알고리즘은 두 개의 전역변수를 가지는데 BestMatch와 MinNotMatch이다. BestMatch는 현재까지 발견된 최적 정합 프레임의 식별자를 저장하고, MinNotMatch는 BestMatch의 기호와 정합되지 않는 수를 저장한다. 노드 N에 이를 때마다 노드 N과 부모 노드의 기호를 검색한다. 그 기호가 목표 기호 시퀀스의 해당 기호로부터 거리 허용치 ε을 벗어나면 비정합 기호들의 숫자인 노드 N의 NotMatch(N)을 증가시킨다. NotMatch(N)이 MinNotMatch보다 작은 경우는 trie를 더 내려가서 N의 자식 노드를 검색하고, 그렇지 않으면 노드 N의 형제 노드로 이동한다. NotMatch(L)이 MinNotMatch보다 작은 종단 노드 L에 이르면 종단 노드 L에 저장된 식별자를 BestMatch로 수정하고, NotMatch(L)을 MinNotMatch로 수정한다. 종단 노드는 다수의 식별자를 저장할 수 있고, 다른 경로들은 동일한 수의 비 정합 기호를 가질 수 있으므로, 각 목표 프레임에 대하여 다수의 최적 정합을 가질 수 있다.

질의처리를 위한 마지막 단계인 시간 축 분할은 각 목표 프레임에 대하여 최적의 정합을 찾을 때마다 한 개 또는 그 이상의 스택에 비디오와 프레임 번호에 따라 저장으로 시작된다. 각 스택은 비디오 번호로 표기되어지고, 스택의 각 요소는 프레임 번호로 표기된다. $V_i[p]$에 대한 각 최적의 정합에 대하여 시간축 분할을 위하여 다음과 같은 처리를 한다. 1) i로 표기된 스택을 찾는다. 2) 최종 요소가 p보다 작은 스택의 집합에 p를 저장한다. 이 시간 축 분할을 처리한 후, 요소 수가 최대인 스택이 해답으로 판단된다. 목표 비디오 V_t가 다섯 프레임을 가지며, 최적 정합이 $V_t[1]$에 $V_1[1]$, $V_2[10]$, $V_t[2]$에 $V_2[5]$, $V_3[1]$, $V_t[3]$에 $V_2[7]$, $V_4[7]$, $V_t[4]$에 $V_2[8]$, $V_t[5]$에 $V_2[9]$인 것으로 가정한다. 그림 7은 시간 축 분할 후의 다섯 개 스택을 나타낸 것이며, 두 번째 스택이 다른 스택에 비하여 크기 때문에 목표 비디오와 가장 유사한 부분 비디오를 포함하고 있다. 부분 비디오의 시작과 끝 위치는 첫 번째와 마지막 요소 값으로부터 얻어지며, $V_2[5:9]$가 정합된 결과이다. 이상의 알고리즘은 그림 7과 8에 나타내었다.

```
START:
    x = set_target_frame(Vt) // Vt : target video
    get_video(Vi) // i=1~N
    S = create_stack(Vi) // i=1~N
    loop(Vi){
        Lp = get_last(Sj) // j:stack number
        if(x match Lp) push(j,x)
        }
    out_p = max_index(S1)
    loop(j){
        if(out_p > max_index(Sj)){
         out_p = max_index(Sj)
         out_video = j
        }
    }
    set_decision(out_video, out_p)
END:
```

그림 7 시간축 분할을 위한 알고리즘

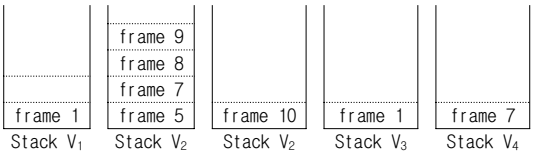


그림 8 시간 축 분할을 위한 다섯 개의 스택

4. 실험 및 고찰

4.1 실험환경

실험에는 합성과 실제 데이타 비디오의 두 종류가 사용되었다. 합성 비디오는 특징치가 정수 값의 유한한 집합으로부터 균일하게 선택된 합성 프레임들로 구성되었으며, 비디오의 수와 평균 프레임 수는 각 실험에 따라 가변적이다. 스타워즈 1 영화에서 처음 210600 프레임으로부터 39개의 실제 비디오를 생성하였고 각 비디오로부터 540개의 키 프레임을 추출하였다. 각 실험마다, 다음과 같이 생성된 목표 비디오와 100개의 질의를 실시하였다: 1) 데이타 비디오로부터 한 개를 임의로 선택한다. 2) 선택된 데이타 비디오로부터 부분 비디오를 추출한다. 3) 프레임의 각 윈도우로부터 적절한 범위의 임의 값을 선택한다. 4) 그 값을 대응하는 윈도우에 잡음으로 가산한다. 성능 단위로 100개의 질의를 처리하는 평균수행시간을 사용하였다. 실험에 사용된 하드웨어는 IBM PC 300PL-Pentium Ⅲ 662Mhz, 256MB RAM이며, 운영체제는 Windows NT 4.0이다. 성능 평가를 위

표 1 실험에 대한 매개변수 값

	exp 1	exp 2	exp 3	exp 4	exp 5	exp 6
data set	Star Wars I	Star Wars I	Synthetic	Synthetic	Synthetic	Synthetic
Feature	Ordinal	Motion	Uniform	Uniform	Uniform	Uniform
numWins	25	225	9-225	16	16	16
numSyms	25	9	8	4-32	8	8
distTol	0-4	0-3	1	1	1	1
numVids	39	39	100	100	100-400	100
lenVids	540	540	200	200	200	200-1000
lenQrys	30	30	20	20	20	20

하여 제안한 방법과 각 목표 프레임에 대한 최적 정합을 찾기 위하여 데이타 비디오를 순차적으로 읽은 다음 3장에서 설명한 시간축 분할을 시행한 순차적 스캔 방법을 비교하였다. 표 1은 각 실험에 사용된 매개변수를 설명한 것이며, 1) numWins는 한 개 프레임에서의 윈도우 개수, 2) numSyms는 한 개 윈도우 내에서의 양자화 기호의 전체 숫자 3) distTol은 사용자에 의해 제시되는 거리 허용치, 4) numVids는 데이타베이스에 저장된 비디오의 전체 숫자, 5) lenVids는 데이타 비디오에서의 평균 프레임의 수, lenQrys는 목표 비디오에서의 프레임 수를 의미한다.

4.2 결과 및 분석

스타워즈 1의 오디널 특징치를 사용하여 실험1은 제안한 방법과 순차적 스캔 방법의 수행시간을 거리 임계치 ε를 0에서 4로 가변하면서 비교하였다. 실험 2는 움직임 특징치를 사용하여 동일한 조건으로 시행하였다. 그림 9와 10에 보이는 것과 같이 제안한 방법이 오디널 특징치를 사용하였을 경우 두 배 정도 빠르고, 움직임 특징치를 사용하였을 때는 세 배 정도 빠른 것으로 나타났다. 제안한 방법은 허용치 ε이 클 때 좋은 성능을 보이는데, 큰 값의 ε은 MinNotMatch가 빨리 최종 값에 도달하게 하여 인덱스 검색에 의해 영향을 받는 trie의 부분을 감소시키기 때문이다.

합성 데이타 집합을 사용하여 실험3과 실험 4는 윈도우의 개수를 9에서 225까지 증가시키고, 양자화 기호의 수를 4에서 32로 증가시키면서 두가지 방법의 수행시간을 비교하였다. 그림 11에서 보는 바와 같이 두가지 방법은 윈도우의 숫자가 증가할수록 더 많은 질의 처리 시간을 필요로 하지만, 제안한 방법은 순차적 스캔방법에 비하여 기울기가 작게 나타났다. 그림 12에서 보는 바와 같이, 두 방법은 질의 처리 시간이 약간 증가함을 보이지만, 제안한 방법은 양자화 기호의 수가 5에서 10 사이일 때 급격히 증가하는 형태를 보인다. 이 현상은 양자화기호의 숫자가 증가함에따라 공통 부분열의 수와 그들의 평균 길이가 감소함을 보여주는 것이다. 그러나, 기울기는 기호의 수가 특정한 임계치를 지난 후 안정적으로 변한다.

합성 데이타 집합을 사용하여 실험5와 실험 6은 비디오의 숫자를 100에서 400까지 증가시키고, 비디오 프레임의 평균 숫자를 200에서 1000까지 증가시키며 두 방법의 수행 시간을 비교한다. 그림 13과 14는 실험적인 결과를 보여주며, 비디오 데이타베이스의 크기 증가로 인하여 두 가지 방법은 비디오의 숫자와 프레임의 평균 숫자 증가에 따라 실행이 느려지게 된다. 그러나, 제안

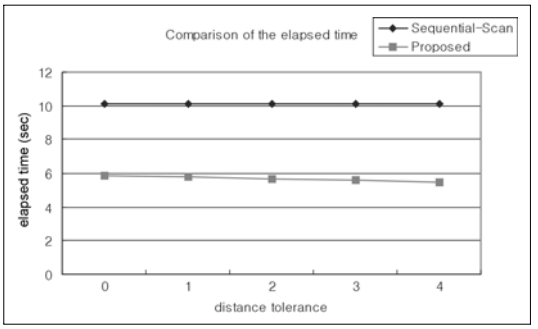


그림 9 거리 허용치의 증가에 따른 수행시간의 비교 (스타워즈 1, 오디널 특징치)

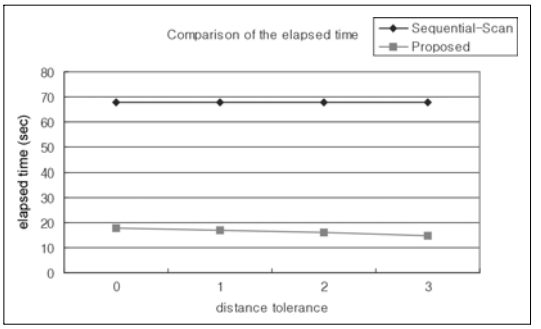


그림 10 거리 허용치의 증가에 따른 수행시간의 비교 (스타워즈 1, 움직임 특징치)

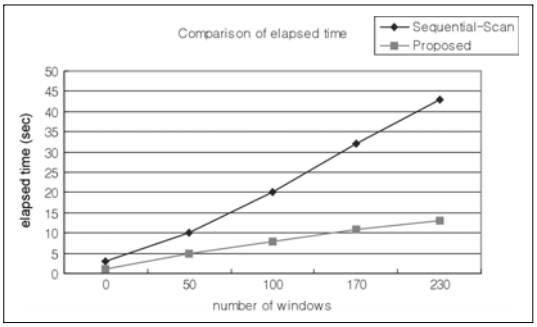


그림 11 윈도우 숫자의 증가에 따른 수행시간의 비교 (합성 데이타 집합)

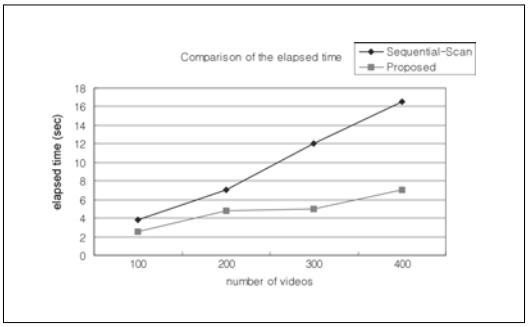


그림 13 비디오 숫자의 증가에 따른 수행시간의 비교 (합성 자료 집합)

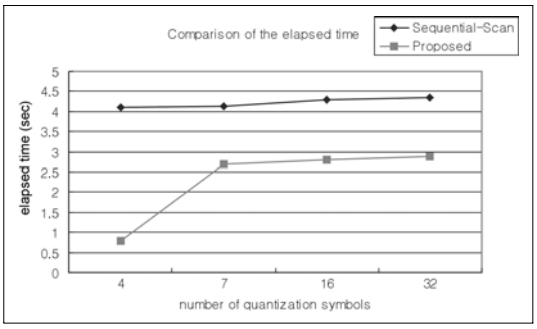


그림 12 양자화기호의 숫자 증가에 따른 수행시간의 비교 (합성 데이타 집합)

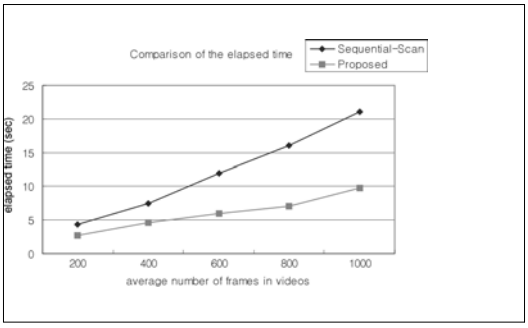


그림 14 비디오 프레임의 평균 숫자 증가에 따른 수행시간의 비교(합성 자료 집합)

한 방법의 기울기는 순차적 스캔방법보다 작다. 그러므로, 제안한 방법이 스케일 조정이 가능하고, 대용량 비디오 데이타베이스에 더욱 적합하다.

5. 결 론

비디오 데이타베이스에서의 유사도 정합은 새로운 많은 응용 프로그램에서 중요성이 커지고 있다. 비디오 데이타베이스에서 상대적인 자료에 대한 효과적인 접근을 제공하는 많은 연구 노력이 있어 왔으나, 기존의 방법들은 순차적 스캔 또는 메모리-기반의 반전된 파일 등에 의존하므로 대용량의 비디오 데이타베이스에 적합하지 않다. 이 문제를 해결하기 위하여, 본 논문에서는 효율적이고 스케일 조정 가능한 trie를 인덱스 구조로 사용한 인덱싱 기술을 제안하였다. 제안한 내용은 1) trie를 비디오 인덱싱에 적용함과 2) 검색성능의 향상과 인덱스 크기의 감소를 위한 윈도우 순서 휴리스틱의 도입이다. 제안한 방법의 효율성과 스케일 조정 가능을 확인하기 위하여 실제와 합성 비디오 데이타베이스를 이용한 실험을 하였으며, 제안한 방법이 일관성 있게 순차적인 검색 방법을 능가하였고, 성능 이득은 대용량 비디오 데이타에서도 유지되는 결과를 얻었다.

참 고 문 헌

[1] E. Ardizzone, M. La Cascia, A. Avanzato, and A. Bruna, "Video Indexing Using MPEG Motion Compensation Vectors," Proc. IEEE Int. Conf. on multimedia Computing System, 1999.

[2] S. Dagtas and A. Ghafoor, "Indexing and Retrieval of Video based on Spatial Relation Sequences," Proc. of ACM multimedia 1999, 1999.

[3] A. Hampapur and R. bolle, "Feature Based Indexing for Media Tracking," Proc. Int'l Conf. on Multimedia and Expo, pp.67-70, Aug. 2000.

[4] V. Kobla, D.S. Doermann, K-I. Kin, and C. Flaoutsos, "Compressed domain video indexing techniques using DCT and motion vector information in MPEG video," Proc. SPIE Conf. on Storage and Retrieval for Image and Video Databases, 1997.

[5] J. Meng and S. -F Chang, "CVEPS - A Compressed Video Editing and Parsing System,"

Proc. of ACM Multimedia 1996, 1996.
[6] Emile Sahouria, and Avideh Zakhor, "Motion Indexing of Video," Proc. International Conference on Image Processing, 1997.
[7] Roland Tusch, Harald, Kosch, and Laszlo Boszormeny, "VIDEX: An Integrated Generic Video Indexing Approach," Proc. of ACM Multimedia 2000, 2000.
[8] J. Wei, Z.N. Li, and I. Gertner, "A novel motion-based active video indexing method," Proc. IEEE int. Conf. on Multimedia Computing System, 1999.
[9] R. Mohan, "Video Sequence Matching," Proc. Int'l Conf. on Audio, Speech and signal processing, 1998.
[10] A. Vailaya, W. Xiong, and A. K. Jain, "Querty by Video Clip," Proc. Int'l Conf. on Pattern Recognition, pp.909-911, 1998.
[11] R. Lienhart, C. Kuhmunch, and W. Effelsberg, "On the Detection and REcognition of Television Commercials," Proc. IEEE Conf. on Multimedia Computing and Systems, 1997.
[12] J. M. Sanchez, X. Binefa, J. Vitria, and P. Radeva, "Local Color Analysis for Scene Break Detection Applied to TV commercials Recognition," Proc. Visual 99, pp.237-244, June 1999.
[13] D. A. Adjeroh, M. C. Lee, and I. King, "A Distance Measure for Video Sequence Similarity Matching," Proc. Int'l Workshop on Multi-Media Database Management Systems, 1998.
[14] D. M. Squire, H. Muller, and W. muller, "Improving Response Time by Search Pruning in Content Based Image Retrieval System, Using Inverted File Techniques," Proc. IEEE Workshop on Content Based Image and Video Libraries, pp.45-50, 1990.
[15] G. A. Stephen, String Searching Algorithms, world Scientific Publishing, 1994.
[16] A. Hampapur, K. Hyun and R. Bolle, "Comparison of Sequence Matching Techinques for Video Copy Detection," Proc. SPIE Conf. on storage and Retrieval for Media Databases, 2002.(accepted to appear)
[17] M Ellis, Method and systems for recognition of broadcast segments. In US5504518. United State Patent Office, 1996.
[18] M. Ioka and M. Kurokawa, "A Method for Retrieving Sequences of Images on the basis of Motion Analysis," Proc. SPIE Conf. on Image Storage and Retrieval Systems, 1994.
[19] D. N. Bhat and S. K. Nayar, "Ordinal Measures for Image Correspondence," IEEE Trans. on PAMI, 20(4), pp. 415-423, 1998.
[20] http://www.dlib.org/dlib/february97/columbia/02chang.html#art

현 기 호
정보과학회논문지 : 소프트웨어 및 응용
제 30 권 제 6 호 참조

김 정 엽
1990년 2월 경북대학교 전자공학과 졸업(공학사). 1992년 2월 경북대학교 대학원 전자공학과(공학석사). 1992년 3월~2001년 2월 경북대학교 대학원 전자공학과(공학박사). 1994년 6월~2001년 2월 삼성종합기술원 전문연구원. 2001년 3월~현재 영산대학교 멀티미디어공학부 전임강사. 주관심 분야는 영상처리, 광원추정, 디지털 하프토닝 비디오 인덱싱 등임

박 상 현
1989년 2월 서울대학교 컴퓨터공학과 졸업(학사). 1991년 2월 서울대학교 컴퓨터공학과 졸업(석사). 2001년 3월 UCLA대학교 전산학과 졸업(박사). 1991년 3월~1996년 7월 대우통신 연구원. 2001년 2월~2002년 6월 IBM T.J. Watson Research Center Post-Doctoral Fellow. 2002년 8월~현재 포항공과대학교 컴퓨터공학과 교수. 관심분야는 시공간 데이타베이스, 멀티미디어 데이타베이스, 데이타 마이닝, 데이타 웨어하우징, XML, 분산 데이타베이스