비디오 검색 시스템을 위한 데이터 시퀀스 패턴 유사성 검색

이 석 룡†

요 약

비디오 스트림은 다차원 공간에서 데이터 포인트의 시퀀스로 표현될 수 있다. 본 논문에서는 시퀀스 내의 데이터 포인트들의 값들의 근사치에 대한 정보와 시퀀스 내의 포인트들의 방향성에 대한 정보를 내포하고 있는 트랜드 벡터(trend vector)에 대한 소개와 이 벡터를 이용하여 데이터 시퀀스를 위한 유사 패턴 검색 기법을 제안한다. 시퀀스는 복수 개의 세그먼트로 분할되며 각 세그먼트는 트랜드 벡터로 표현된다. 질의 처리는 시퀀스 내의 각각의 포인트들에 대하여 수행되는 대신, 트랜드 벡터들에 대하여 처리된다. 제안한 기법은 이 벡터를 사용하여 질의와 무관한 데이터 시퀀스들을 데이터베이스로부터 여과하고 질의 시퀀스와 유사한 시퀀스들을 검색하도록 설계되었다. 제안한 기법을 검증하기 위하여 비디오 스트림과 가상으로 생성된 데이터에 관하여 실험을 수행하였으며, 실험 결과 제안한 기법의 정밀도(precision)는 기존의 방법에 비하여 2.1배까지 향상되었으며 처리시간은 45%까지 감소되었음을 보여주고 있다.

키워드 : 비디오 스트림, 데이터 시퀀스, 정보 검색, 패턴 유사성 검색

Pattern Similarity Retrieval of Data Sequences for Video Retrieval System

Seok-Lyong Lee†

ABSTRACT

A video stream can be represented by a sequence of data points in a multidimensional space. In this paper, we introduce a trend vector that approximates values of data points in a sequence and represents the moving trend of points in the sequence, and present a pattern similarity matching method for data sequences using the trend vector. A sequence is partitioned into multiple segments, each of which is represented by a trend vector. The query processing is based on the comparison of these vectors instead of scanning data elements of entire sequences. Using the trend vector, our method is designed to filter out irrelevant sequences from a database and to find similar sequences with respect to a query. We have performed an extensive experiment on synthetic sequences as well as video streams. Experimental results show that the precision of our method is up to 2.1 times higher and the processing time is up to 45% reduced, compared with an existing method.

Key Words : Video Stream, Data Sequence, Information Retrieval, Pattern Similarity Retrieval

1. 서 론

비디오 스트림은 연속된 프레임들로 구성되며 각각의 프레임은 색상(color), 질감(texture), 모양(shape)과 같은 다양한 특징(feature)들로 표현된다. 하나의 프레임은 특징 공간(feature space)에서 데이터 포인트로 표현될 수 있으며, 따라서 비디오 스트림은 각 특징이 하나의 차원을 형성하게 함으로써 다차원 데이터 공간에서 데이터 포인트들의 궤적으로 모형화할 수 있다. 다차원 시퀀스(multidimensional sequence)라 불리는 이 포인트들의 궤적은 시간의 흐름에 따라 연속된 데이터 요소들의 집합으로써, 각 요소는 다차원 벡터에 의해 표현된다. [12]에서는 d-차원 공간에서 K개의 포인트들을 가지는 다차원 시퀀스 S를 다음과 같이 연속된 요소 벡터들로 정형화하였다: $S = <S[0], S[1], \cdots, S[K-1]>$, 여기에서 각 벡터 $S[i]$ $(0 \le i \le K-1)$ 는 d 개의 스칼라 요소들로 구성된다. 즉, $S[i] = (S^1[i], S^2[i], ..., S^d[i])$ 이다. 특정 시점에서의 값을 나타내는 일차원 실수 값들의 시퀀스인 시계열 데이터는 다차원 시퀀스의 각 백터 요소를 스칼라 값으로 치환함으로써 모형화할 수 있다.

본 논문에서는 다차원 공간에서 시퀀스를 위한 효과적인

※ 이 논문은 2003년도 한국학술진흥재단의 지원에 의하여 연구되었음 (KRF-2003-041-D00628).
† 정 회 원 : 한국외국어대학교 산업정보시스템공학부 부교수
논문접수 : 2005년 7월 28일, 심사완료 : 2006년 5월 18일

유사성(similarity) 검색 기법을 제안한다. 데이터 시퀀스는 같은 길이 혹은 다른 길이의 세그먼트로 분할되고 각 세그먼트는 트랜드 벡터로 표현된다. 트랜드 벡터는 세그먼트 내의 포인트들의 움직임의 경향에 관한 중요한 정보를 내포하고 있다. 시퀀스 간의 유사성 척도는 각 시퀀스에 포함되어 있는 세그먼트들 간의 유사성을 기초로 하여 정의된다. 즉, 시퀀스 간의 질의 처리는 시퀀스 내의 모든 데이터 포인트에 대하여 수행되는 대신에 각 시퀀스를 이루고 있는 세그먼트들의 트랜드 벡터들을 비교함으로써 효율적으로 수행된다. 제안한 기법은 트랜드 벡터를 이용하여 데이터베이스로부터 질의와 무관한 시퀀스들을 여과하여 후보 시퀀스들을 추출하고, 이로부터 질의에 유사한 시퀀스들을 검색하도록 설계되었다. 세그먼트 사이의 유사성 척도 및 시퀀스 사이의 유사성 척도에 기초한 시퀀스의 패턴 유사성 검색 문제는 다음과 같이 정형화하여 표현할 수 있다:

Given: 질의 시퀀스 Q, 데이터 시퀀스의 집합 S_Set 및 유사성 임계 값 $\zeta(0 \le \zeta \le 1)$

Goal: 데이터 시퀀스 S ($S \in S\_Set$)와 Q의 유사성이 ζ 보다 크거나 같은 S를 S_Set 으로부터 검색

일반적으로 유사성 임계 값은 전통적인 기법들에서는 개체 간의 '유클리디안 거리(Euclidean distance)'에 의하여 정의된다. 그러나 사용자들이 이러한 값을 결정하는 것이 쉽지 않는 데, 그 이유는 사용자들이 거리와 유사성과의 상관관계에 대한 직관을 가지기가 어렵기 때문이다. 이러한 모호성을 피하기 위하여 제안한 기법은 사용자가 직관적으로 이해할 수 있고 정량화할 수 있는 '유사성(similarity)'을 사용한다. 이것은 0과 1사이의 값으로 여기에서 '1' 은 정확하게 같다는 것을 의미하고 '0' 은 완전히 다르다는 것을 의미한다. 제안한 기법의 주요한 공헌은 다음과 같이 요약될 수 있다:

- 제안한 기법은 일차원 시계열 데이터에서의 전통적인 유사성 검색을 확장하여, 다차원 데이터 시퀀스, 즉 더욱 일반화된 형태의 데이터 시퀀스에 대한 유사성 검색을 제안하고 있다.
- 트랜드 벡터에 의한 검색은 시퀀스의 거리뿐만 아니라 시퀀스 내의 포인트들의 이동 방향을 고려하여 이루어진다. 실험 결과에 의하면 제안한 기법이 기존의 방법(다차원 데이터시퀀스를 수용하도록 약간 수정됨) 에 비하여 2.1배까지 높은 정밀도(precision)를 보여주고 있고, 처리시간은 45% 까지 향상되었으며, 리콜(recall)에서는 비슷한 결과를 보여주고 있다.
- 제안한 기법의 트랜드 벡터 표현은 세그먼트의 포인트들의 원래 값에 대하여 매우 뛰어난 근사 방법을 제공하므로, 원래의 시퀀스에 대해 매우 낮은 복원 오차(reconstruction error)를 가진다.
- 제안한 기법은 검색을 위하여 사용할 인덱스 구조의 종류를 제한하지 않는다. 따라서 R*-트리와 X-트리 등의 다양한 인덱스 구조를 사용할 수 있다.

본 논문의 구성은 다음과 같다: 2절은 기존의 시퀀스 데이터의 유사성 검색 기법에 관한 관련 연구를 제공한다. 3절에서는 트랜드 벡터의 소개와 표현 방법을 제시하고, 4절에서는 유사성 함수 및 트랜드 벡터를 이용한 검색 기법을 기술한다. 5절에서는 비디오 스트림 및 가상 시퀀스 데이터에 대한 실험 결과를 기술하고, 6절에서 결론을 맺는다.

2. 관련 연구

시계열 데이터에 대한 유사성 검색은 최근 데이터베이스 응용 분야에서 중요한 연구 주제 중 하나로 관심을 받아 왔다 [1, 5, 16, 15, 18, 10]. Agrawal 등[1]은 최초로 DFT(discrete Fourier transform)를 사용하여 시간 도메인의 시퀀스 데이터를 저 차원의 주파수 도메인으로 사상(mapping)하여 고차원 문제(dimensionality curse problem)를 해결한 차원 축소 기법을 소개했다. 이 방법에서는 하나의 시간 시퀀스는 주파수 도메인에서 저 차원 상의 한 포인트로 표현된다. Faloutsos 등 [5]는 슬라이딩 윈도우(사이즈 w)를 사용하는 빠른 서브 시퀀스 매칭 방법(fast subsequence matching method)을 제안했다. 일 차원상의 시퀀스 위에 윈도우를 이동시킴으로써 윈도우 내의 서브 시퀀스가 w-차원의 포인트로 변환된다. 따라서, 원래의 시간 시퀀스는 w-차원의 시퀀스로 변환된다. 이 시퀀스는 서브 시퀀스로 분할되고 각 서브 시퀀스는 최소 경계 사각형(minimum bounding rectangle: MBR)으로 표현되며, 이 MBR이 데이터베이스에 인덱싱되고 저장된다. 그러나 비디오 스트림과 같은 다차원 시퀀스에서 한 포인트의 개념은 일차원 시계열 데이터의 포인트와는 다른 의미를 가진다. 다차원 시퀀스에서는 포인트 자체가 다양한 특징을 각각 차원으로 하는 다차원 공간에서의 벡터가 되며, 이 벡터의 각 차원의 요소들은 해당 특징의 추출 값들이 된다. 따라서, 다차원 시퀀스나 서브 시퀀스를 하나의 대표적인 포인트로 사상할 수 없기 때문에 위의 기존 방법들은 다차원 시퀀스에는 적용될 수 없다.

Rafiei 등 [16]은 시계열 데이터에서 유사성 질의를 위한 기초로 사용될 수 있는 주어진 시퀀스의 안전한 선형 변환(safe linear transformation)의 집합을 제안했다. 이 기법에서는 이동 평균(moving average), 역 변환(reversing)과 타임 워핑(time warping)과 같은 변환 방법들을 정형화하여 표현하였다. 이러한 변환은 [15]에서 다양한 변환으로 확장되는 데, 여기에서는 인덱스를 여러 번 검색하고 매 검색마다 매번 변환하는 대신에, 인덱스를 단지 한번 검색하고 다수의 변환이 동시에 이 인덱스에 적용된다.

최근에 Yi 등 [18]은 시간 시퀀스를 같은 길이의 세그먼트로 나누고 세그먼트내의 포인트들의 평균을 저장하는 근사 기법(approximating technique)을 제안하였다. 이 기법에서는 한 시퀀스가 s개의 세그먼트로 나누어지며 그 시퀀스는 s개의 세그먼트 평균들을 요소로 가지는 벡터로 표현된다. [18]을 확장하여 Keogh 등 [10]은 APCA(adaptive piecewise constant approximation)로 불리는 새로운 차원

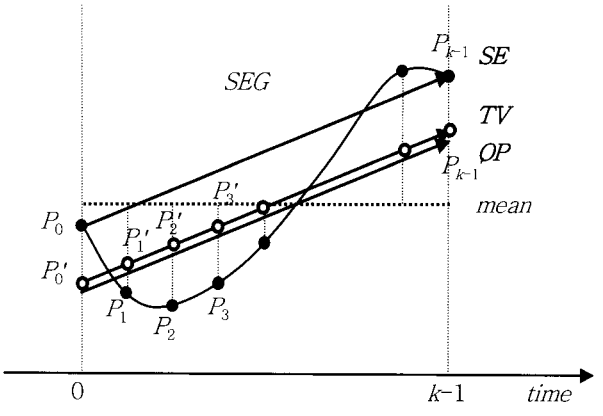
축소 기법을 소개하였다. 이 기법에서는 하나의 시간 시퀀스를 복원 오차가 최소가 되도록 하는 길이가 다른 복수 개의 세그먼트의 집합으로 근사하여 표현하였다. 또한, APCA에서는 각각의 길이가 다른 세그먼트가 다차원 인덱스 구조로 인덱싱될 수 있음을 보였고, D_{LB}와 D_{AE}, 즉 하한 유클리드 거리 근사(lower bounding Euclidean distance approximation)와 비하한 유클리드 거리 근사(non-lower bounding tight approximation)를 제공하는 두 개의 거리 측정 기준을 제안하였다.

그러나 기본적으로 위에 언급한 기법들은 모두 일차원 시계열 데이터를 위한 유사성 검색 기법들이다. 따라서 이들은 다차원 시퀀스에는 근본적으로 적용하기가 어렵고, 또한 위 기법들의 근사 방법에서는 세그먼트를 단순히 포인트나 최소 경계 사각형으로 표현함으로써 세그먼트 내의 포인트들의 움직임에 관한 중요한 정보를 상실하게 된다는 점이다. 또 다른 문제점으로써, 평균에 기초한 접근 방법들[18, 10]은 원래의 시퀀스에 대하여 본질적으로 높은 복원 오차를 초래하게 된다. 이 방법에서 세그먼트 내의 포인트들의 움직임에 관한 정보를 보존하고 복원 오차를 줄이기 위해 세그먼트의 크기를 매우 작게 유지하는 대안을 고려할 수 있지만, 이렇게 하면 평균 값들의 수 (즉, 세그먼트의 수)가 늘어나게 되고 결국 심각한 처리 오버헤드를 초래하게 된다.

한편, 비디오 검색에서는 각 비디오 샷(video shot)을 세그먼트로 간주할 수 있으며, 각 세그먼트에 대하여 키 프레임(key frame)이 선택되고 선택된 프레임들에 기초하여 질의를 처리하는 것이 일반적이다 [6]. 그러나 키 프레임에 의한 검색은 키 프레임이 비디오 샷 내의 모든 프레임들을 항상 적절하게 요약할 수 없고, 또한 세그먼트 내에서 포인트들의 움직임을 나타내는 트랜드를 검색 단계에서 사용할 수 없기 때문에 질의와 유사한 많은 시퀀스들을 놓치게 된다. Lee등은 [12]에서 두 개의 하한 (low bounding) 거리 측정 기준인 D_{mbr} 과 D_{norm}에 기초한 다차원 시퀀스를 위한 유사성 검색 방법을 제안했다. 이 검색 방법에서는 D_{mbr} 과 D_{norm}을 사용하여 질의에 무관한 시퀀스들을 데이터베이스에서 여과한다. 또한, [20]에서는 의미정보를 이용한 다차원 데이터 시퀀스의 유사성 척도를 제안하였고, 이를 기반으로 비디오 데이터 스트림에 대한 유사성 검색 알고리즘 [19]가 연구되었다. 그러나 이러한 접근법 역시 다차원 시퀀스를 위한 검색은 지원하지만 포인트들의 움직임에 관한 트랜드의 정보를 검색에 적용하지는 못하고 있다.

3. 트랜드 벡터 표현

세그먼트 내 데이터 포인트들의 움직임의 트랜드를 나타내기 위해서는 보통 그 데이터 포인트들을 근사(approximation)하는 직선의 기울기를 사용한다. (그림 1)에서와 같이 k개의 포인트 $<P_0, P_1, \cdots, P_{k-1}>$을 포함하고 있는 세그먼트 SEG를 고려해 보자. 세그먼트 SEG의 트랜드를 나타내기 위해서 세그먼트의 시작 포인트 P_0 과 끝 포인트 P_{k-1}을 잇는 선분의 기울기를 채택한다. 이 선분에 '방향' 특성을 부여하기 위하여 이것을 start-end 벡터 ***SE*** 로 표현한다. 벡터 ***SE*** 는 세그먼트 내의 데이터 포인트들을 잇는 모든 선분들을 연결하여 생성되는 벡터 합(vector sum)과 동일하게 되므로 합리적인 선택이 된다. 그러나 세그먼트 자체를 대표하기 위해 벡터 ***SE***를 채택하는 것은 원래의 시퀀스에 대하여 많은 경우에 매우 큰 복원 오차를 야기하므로 바람직한 선택이 아니다. (그림 1)과 같이 벡터 ***SE*** 에 평행하면서 복원 오차를 최소화 하는 최적 벡터 (optimal vector: ***OP***)를 세그먼트를 대표하기 위해 채택하는 것도 계산 오버헤드를 야기하므로 바람직한 선택이 아니다. 따라서 벡터 ***SE***에 평행하면서 세그먼트의 평균(mean)선의 중간을 지나는 벡터를 세그먼트를 대표하기 위해 선택하고, 이를 트랜드 벡터 ***TV***라 한다.



(그림 1) 세그먼트의 시작-끝 벡터(***SE***), 최적 벡터(***OP***) 및 트랜드 벡터(***TV***)

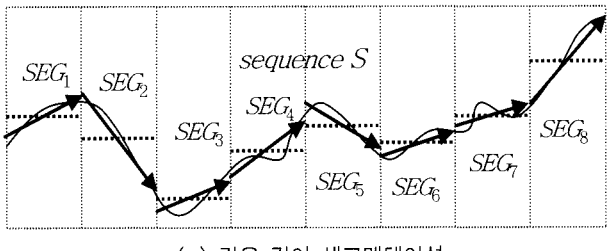
(그림 1)은 일차원 시간 공간(time space)에서의 트랜드 벡터를 나타낸 것이다. 편의상 가로축을 시간 축으로 표현했기 때문에 평균(mean)이 선으로 나타나 있지만, 시간축이 배제된 다차원 공간에서 평균은 포인트로 표현되며, 따라서 트랜드 벡터 ***TV***는 벡터 ***SE***에 평행하면서 세그먼트의 평균점을 지나게 된다. 평균에 기초한 접근 방법들의 경우에는 모든 데이터 포인트 $<P_0, P_1, \cdots, P_{k-1}>$ 가 단 하나의 값 - 평균에 의하여 근사되는 반면, 트랜드 벡터 표현 방법에서는 각 데이터 포인트 $<P_0, P_1, \cdots, P_{k-1}>$를 벡터 ***TV*** 상에 각각 사상한 점 $<P_0', P_1', \cdots, P_{k-1}'>$ 으로 근사시킴으로써 복원 오차를 줄이게 된다.

트랜드 벡터는 세그먼트 내의 포인트의 수, 세그먼트의 평균 벡터, 그리고 각 요소가 다차원 공간에서 벡터 ***SE***의 각 차원에 대한 기울기로 구성된 기울기 벡터 α로 정의될 수 있다. 트랜드 벡터의 정형적 정의는 다음과 같다:

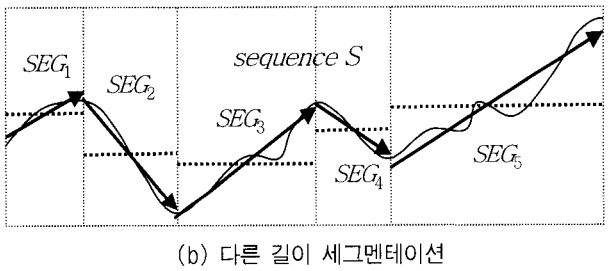
[정의 1] d-차원 공간에서 트랜드 벡터 ***TV***는 ***TV*** = <k, ***m***, α>로 정의되며, 여기에서 k 는 세그먼트 안에 포함되어 있는 포인트의 수이며, ***m*** = $(m_1, m_2, \cdots, m_d)$은 각 차원에서의 데이터 포인트들의 평균으로 구성되어 있는 세그먼트의 평균 벡터이고, $\alpha=(\alpha_1, \alpha_2, \cdots, \alpha_d)$는 세그먼트의 시작 포인트로부터 끝 포인트를 잇는 선분의 각 차원에서의 기울기를 나타내는 기울기 벡터이다.

α=NULL으로 하면 위의 표현은 'APCA 표현[10]'으로 축소될 수 있으며, k=NULL 과 α=NULL 로 함으로써 위의 표현은 '세그먼트-평균 표현[18]'으로 축소될 수 있다.

트랜드 벡터 TV 는 다음과 같은 여러 가지 장점을 가지고 있다: (1) 세그먼트 내의 포인트들의 움직임의 경향에 대한 효과적인 근사를 제공한다. (2) 원래의 시퀀스에 대하여 복원 오차율이 낮다. (3) 트랜드 벡터의 계산은 단순하고 매우 빠르다. (3) 유사성 검색에 사용함으로써 주어진 질의에 무관한 대다수의 시퀀스를 검색 대상에서 제외시킬 수 있으며, 따라서 매우 높은 정밀도를 달성할 수 있다. 다음 (그림 2)(a) 는 일차원 시계열 데이터에 대한 원래 시퀀스의 세그먼트-평균 표현[18]과 제안한 방법에서 같은 길이로 세그멘테이션할 경우의 트랜드 벡터 근사를 보여주고 있으며, (그림 2)(b) 는 APCA 표현[10]]과 제안한 방법에서 서로 다른 길이로 세그멘테이션할 경우의 트랜드 벡터 근사를 보여주고 있다. 그림에서 보는 바와 같이 트랜드 벡터 근사가 평균 값 근사에 비하여 상당히 효과적임을 관찰할 수 있다.



(a) 같은 길이 세그멘테이션



(b) 다른 길이 세그멘테이션

(그림 2) 시퀀스 및 각 세그먼트의 트랜드 벡터 표현

4. 트랜드 벡터를 이용한 시퀀스 패턴 검색

4.1 세그먼트 사이의 유사성

이 절에서는 세그먼트 간의 거리를 측정하기 위한 두 가지의 거리 함수 d_hr과 d_seg를 소개한다. 함수 d_hr은 데이터베이스로부터 질의와 무관한 세그먼트를 배제하기 위한 첫 번째 여과(filtering) 함수로 사용되며, 벡터 TV를 둘러싸고 있는 하이퍼사각형 간의 최소 거리로 표현된다. 거리 함수 d_seg는 세그먼트 내의 각 포인트들을 (그림 1)과 같이 트랜드 벡터 상에 사상(mapping)한 후, 두 트랜드 벡터 상의 각 포인트들의 쌍 사이의 거리를 평균한 거리로 나타내며 두 번째 여과(filtering) 함수로 사용된다. 다음의 정의와 정리는 거리 함수 d_hr과 d_seg의 특성 및 이들 사이의 연관 관계를 나타낸다.

[정의 2] 두 세그먼트 SEG_1 과 SEG_2 사이의 거리 d_hr (SEG_1, SEG_2) 은 SEG_1 과 SEG_2 의 트랜드 벡터 TV_1과 TV_2를 각각 최소한으로 둘러싸고 있는 두 하이퍼사각형 간의 최소 거리로 정의된다.

$$d\_hr(SEG_1, SEG_2) = \sqrt{\sum_{j=1}^{d} u_j^2},$$

$$\text{where } u_j = \begin{cases} |SEG_1.H_j - SEG_2.L_j| & \text{if } SEG_1.H_j < SEG_2.L_j \\ |SEG_1.L_j - SEG_2.H_j| & \text{if } SEG_2.H_j < SEG_1.L_j \\ 0 & \text{otherwise .} \end{cases}$$

여기에서, $SEG.L$과 $SEG.H$는 각각 SEG의 트랜드 벡터 TV를 최소한으로 둘러싸고 있는 하이퍼사각형의 저점과 고점이다.

[정의 3] 포함하고 있는 데이터 포인트의 수가 같은 두 세그먼트 SEG_1 과 SEG_2 사이의 거리 d_seg(SEG_1, SEG_2)는 두 트랜드 벡터 TV_1과 TV_2 상의 각 데이터 포인트들의 쌍 사이의 거리를 평균한 거리로 정의된다.

$$d\_seg(SEG_1, SEG_2) = \frac{1}{k} \cdot \sum_{i=0}^{k-1} d(P_{1,i}', P_{2,i}')$$

데이터 포인트의 수가 서로 다른 세그먼트 간의 거리는 세그먼트 내의 포인트들이 일 대 일로 대응되지 않으므로 위의 식으로는 계산할 수 없다. 이 경우에는 짧은 길이의 세그먼트가 상대 세그먼트의 시작부터 끝까지 한 포인트 씩 슬라이딩하면서 비교하여 평균 거리들을 산출하고, 이 중 가장 짧은 거리를 두 세그먼트 간의 거리로 채택한다.

[관찰 4] 거리 d_hr 은 SEG_1 의 트랜드 벡터 TV_1와 SEG_2 의 트랜드 벡터 TV_2 상에 각각 속해 있는 임의의 포인트들 사이의 거리보다 항상 작다. 즉, 다음 식이 성립한다.

$$d\_hr(SEG_1, SEG_2) \le \min_{P_1' \in TR_1, P_2' \in TR_2} d(P_1', P_2')$$

여기에서 TR_1과 TR_2는 각각 트랜드 벡터 TV_1와 TV_2 상의 포인트들의 집합이다.

[정리 5] (하한 거리: $d\_hr(SEG_q, SEG_s) \le d\_seg(SEG_q, SEG_s)$): 질의 세그먼트 SEG_q 와 데이터베이스 세그먼트 SEG_s 사이의 거리 $d\_hr(SEG_q, SEG_s)$ 은 두 세그먼트 사이의 거리 $d\_seg(SEG_q, SEG_s)$ 의 하한이다.

$$d\_hr(SEG_q, SEG_s) \le d\_seg(SEG_q, SEG_s)$$

[증명] 세그먼트 SEG_q, SEG_s 가 각각 k 개의 포인트들을 가지고 있고, P_q' , P_s' 가 SEG_q, SEG_s 의 트랜드 벡터 상에 있는 임의의 포인트라 하면 다음 식이 성립한다.

$$d\_seg(SEG_q, SEG_s) = \frac{1}{k} \cdot \sum_{i=0}^{k-1} d(P_{q,i}', P_{s,i}') \geq \min_{P_q' \in TR_q, P_s' \in TR_s} d(P_q', P_s')$$

관찰 4에 의하여 다음 식이 성립한다.

$$d\_hr(SEG_q, SEG_s) \leq \min_{P_q' \in TR_q, P_s' \in TR_s} d(P_q', P_s')$$

따라서, $d_hr(SEG_q, SEG_s) \leq d_seg(SEG_q, SEG_s)$이 성립한다.

세그먼트의 길이가 다른 경우에는 작은 세그먼트를 큰 세그먼트의 처음부터 끝까지 슬라이딩하면서 생성된 거리들 중 최소 거리로 선택되기 때문에 정리 5를 다른 길이의 세그먼트로 쉽게 확장할 수 있다. 정리 5는 인덱스 검색을 위하여 거리 함수 d_hr을 과오 누락 없이 안전하게 사용할 수 있음을 의미한다. 즉, d_hr에 의해 얻어진 후보 집합은 d_seg에 의하여 얻어진 후보 집합의 슈퍼 집합이 된다. 본 논문에서 데이터 공간은 $[0,1]^d$ 하이퍼 큐브로 정규화되어 있다고 가정하므로, 임의의 두 포인트의 최대 허용 거리는 이 큐브의 대각선의 길이인 $\sqrt{d}$가 된다. 따라서 두 세그먼트 SEG_1과 SEG_2 사이의 유사성 함수 $sim_seg(SEG_1, SEG_2)$는 세그먼트 사이의 거리를 0과 1 사이의 값을 갖도록 변환함으로써 쉽게 계산할 수 있다.

$$sim\_seg(SEG_1, SEG_2) = 1 - \frac{1}{\sqrt{d}} \cdot d\_seg(SEG_1, SEG_2)$$

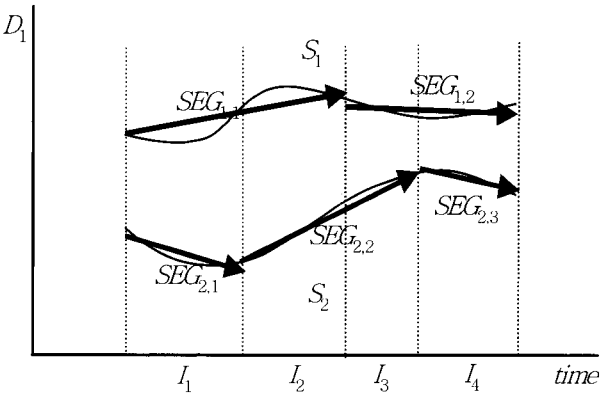
마찬가지로 사용자가 제공한 유사성 임계 값(ζ)은 인덱스 검색을 위하여 거리의 임계 값(ε)으로 쉽게 변환할 수 있고, 역의 경우 역시 성립된다. 즉, $\zeta = 1 - \varepsilon/\sqrt{d}$ 와 $\varepsilon = \sqrt{d}(1-\zeta)$이 된다.

4.2 시퀀스 사이의 유사성

시퀀스는 같은 길이 혹은 다른 길이[1]의 세그먼트로 분할될 수 있다. 각각 다른 길이의 세그먼트로 분할된 두 시퀀스 S_1 과 S_2를 고려해보자. 두 시퀀스 자체의 길이는 같다고 할 때, 두 시퀀스 사이의 $sim_seq(S_1, S_2)$를 구하기 위해서, 전체 시간 구간을 각 시퀀스의 세그먼트 경계를 기준으로 복수 개의 단위 시간 구간으로 나눈다. 다음에 각 단위 구간에서 각각 sim_seg를 구한 후 전체를 병합하는 방식으로 $sim_seq(S_1, S_2)$를 계산한다. 이 과정은 다음 예에 자세히 기술되어 있다.

[예 6] 두 시퀀스 S_1과 S_2이 각각 (그림 3)과 같이 2개와 3개의 세그먼트로 구성되어 있다: $S_1 = (SEG_{1,1}, SEG_{1,2})$, $S_2 = (SEG_{2,1}, SEG_{2,2}, SEG_{2,3})$. 여기에서 $SEG_{1,1} = \langle k_{1,1}, m_{1,1}, a_{1,1} \rangle$, $SEG_{1,2} = \langle k_{1,2}, m_{1,2}, a_{1,2} \rangle$, $SEG_{2,1} = \langle k_{2,1}, m_{2,1}, a_{2,1} \rangle$, $SEG_{2,2} = \langle k_{2,2}, m_{2,2}, a_{2,2} \rangle$, $SEG_{2,3} = \langle k_{2,3}, m_{2,3}, a_{2,3} \rangle$ 이다. $k_{1,1} = 18$, $k_{1,2} = 14$, $k_{2,1} = 10$, $k_{2,2} = 14$, 그리고 $k_{2,3} = 8$ 이라 가정하면, 이러한 세그멘테이션은 비교를 위한 4개의 구간, I_1, I_2, I_3, I_4를 생성한다. 세그먼트의 비교를 위하여 구간 I_1와 I_2에서 세그먼트 $SEG_{1,1}$를 $SEG_{1,1,I1}$과 $SEG_{1,1,I2}$로 분할하고, 구간 I_3와 I_4에서 세그먼트 $SEG_{1,2}$를 $SEG_{1,2,I3}$과 $SEG_{1,2,I4}$로, 그리고 구간 I_2와 I_3에서 세그먼트 $SEG_{2,2}$를 $SEG_{2,2,I2}$와 $SEG_{2,2,I3}$로 각각 분할한다. 그러면 $k_{1,1,I1} = k_{2,1} = 10$, $k_{1,1,I2} = k_{2,2,I2} = 8$, $k_{1,2,I3} = k_{2,2,I3} = 6$, $k_{1,2,I4} = k_{2,3} = 8$을 얻을 수 있다. 세그먼트를 분할함으로써 각 구간에서의 유사성 값을 구할수 있고, 따라서 이 값들을 통합함으로써 두 시퀀스 S_1과 S_2 사이의 유사성을 다음 식으로 계산할 수 있다.

$$\begin{aligned} sim\_seq(S_1, S_2) = \{ & 10 \cdot sim\_seg(SEG_{1,1,I1}, SEG_{2,1}) + \\ & 8 \cdot sim\_seg(SEG_{1,1,I2}, SEG_{2,2,I2}) + \\ & 6 \cdot sim\_seg(SEG_{1,2,I3}, SEG_{2,2,I3}) + \\ & 8 \cdot sim\_seg(SEG_{1,2,I4}, SEG_{2,3}) \} \; / \; 32 \end{aligned}$$



(그림 3) 각각 다른 길이의 세그먼트로 분할된 두 시퀀스 사이의 유사성

예 6에서 볼 수 있는 바와 같이 다른 길이의 세그먼트를 비교하기 위해서는 둘을 비교하기 전에 긴 세그먼트를 짧은 세그먼트의 길이에 맞도록 분할함으로써 비교 대상이 되는 세그먼트의 길이가 같도록 조정할 필요가 있다. 세그먼트 $SEG = \langle k, \boldsymbol{m}, a \rangle$가 r개의 서브 세그먼트, $SEG_1 = \langle k_1, \boldsymbol{m}_1, a_1 \rangle$, …, $SEG_r = \langle k_r, \boldsymbol{m}_r, a_r \rangle$ 로 분할되는 경우를 고려해 보자. 세그먼트 SEG_l $(1 \leq l \leq r)$ 은 이 세그먼트가 속하는 구간에서의 벡터 $\boldsymbol{TV}$의 부분으로 근사될 수 있다. 그러면 $k = \sum_{1 \leq l \leq r} k_l$, $a = a_1 = a_2 = \cdots = a_r$, 그리고 간단한 계산에 의하여 d-차원 공간에서 평균점은 $\boldsymbol{m}_l = (m_{l,1}, m_{l,2}, \cdots, m_{l,d})$ 이 됨을

1) 임의의 길이의 세그멘테이션 방법에는 다양한 최적화 표현 기법들(optimal piecewise polynomial representations) [14, 4]이 있으나 이러한 방법들은 상당한 오버헤드가 야기된다. 반면에 비록 최적의 기법은 아니지만 그리디 접근법(greedy approaches) [11, 17, 13]이 더욱 효율적이고 또한 현실적인 선택이 될 수 있다. 본 논문에서는 [13]에서 제안된 기법, 즉 시퀀스의 기하학적이고 의미적인 특성에 따라서 세그멘테이션하는 기법을 활용한다. 지면상의 제약 때문에 본 논문에서는 세그멘테이션 알고리즘을 상세하게 기술하지는 않는다. 자세한 사항들은 [13]을 참고할 수 있다.

알 수 있다. 두 시퀀스 S_1, S_2 가 T 개의 구간에서 각각 k, l 개의세그먼트를 가질 때 $T \le k+l-1$ 이 되며, 여기에서 부등호는 두 시퀀스에 속한 세그먼트 경계가 일치하는 경우에 발생하게 된다. 각각 다른 길이의 세그먼트를 가지는 두 시퀀스 사이의 유사성은 일반화된 형식으로 표현하면 다음과 같다:

$$sim\_seq(S_1, S_2) = \frac{1}{\sum_{t=1}^{T} k_{I_t}} \cdot \sum_{t=1}^{T} (k_{I_t} \cdot sim\_seg(SEG_{1,I_t}, SEG_{2,I_t}))$$

여기에서, $k_{It} = k_{1,It} = k_{2,It}$이고, $SEG_{1,It}$ 와 $SEG_{2,It}$ 는 각각 구간 I_t에서 S_1과 S_2의 세그먼트이다. 다음의 정리에 근거하여 질의에 무관한 시퀀스를 데이터베이스에서 여과하기 위하여 유사성 함수 *sim_seq*를 안전하게 사용할 수 있다.

[정리 7] $sim\_seq(Q, S) \ge \zeta$ 이면, $sim\_seg(SEG_q, SEG_s) \ge \zeta$ (단,$SEG_q \in Q$, $SEG_s \in S$)를 만족시키는 세그먼트의 쌍 (SEG_q, SEG_s) 이 적어도 하나는 존재한다.

[증명] (*By contradiction*) 다른 길이의 세그멘테이션의 경우가 같은 길이의 세그멘테이션의 경우를 포함한다고 볼 수 있으므로, 다른 길이의 세그멘테이션의 경우만 증명한다. 시퀀스 Q, S 가 각각 k, l 개의세그먼트를 갖고, $Q = (SEG_1, \cdots, SEG_k)$, $S = (SEG_1, \cdots, SEG_l)$라 하자. $k > l$ 인 경우에는 Q, S를 교환하면 되므로 일반성을 상실하지 않고 $k \le l$ 이라 가정할 수 있다. 시퀀스 Q와 S 로부터 생성된 구간의 수 T 는 $T \le k+l-1$ 이 된다. $sim\_seq(Q, S) \ge \zeta$ 일 때 $sim\_seg(SEG_q, SEG_s) \ge \zeta$ $(1 \le q \le k,\ 1 \le s \le l)$을 만족시키는 세그먼트의 쌍 ($SEG_q$, SEG_s) 이 전혀 존재하지 않는다고 가정하면, 다음 식을 유도할 수 있다:

$$\begin{aligned} sim\_seq(Q,S) &= \frac{1}{\sum_{t=1}^{T} k_{I_t}} \cdot \sum_{t=1}^{T} (k_{I_t} \cdot sim\_seg(SEG_{q,I_t}, SEG_{s,I_t})) \\ &< \frac{1}{\sum_{t=1}^{T} k_{I_t}} \cdot (k_{I_1} \cdot \zeta + k_{I_2} \cdot \zeta + \ldots + k_{I_t} \cdot \zeta) \\ &= \frac{1}{\sum_{t=1}^{T} k_{I_t}} \cdot (k_{I_1} + k_{I_2} + \ldots + k_{I_t}) \cdot \zeta \\ &= \zeta \end{aligned}$$

이는 결국 $sim\_seq(Q, S) < \zeta$ 이 되어 처음 세운 가정에 모순이므로 정리 7은 성립한다.

정리 7이 의미하는 바는 세그먼트 사이의 유사성 함수 *sim_seg*가 시퀀스 사이의 유사성 함수 *sim_seq*에 대하여 안전한 여과를 제공한다는 점이다. 즉, *sim_seg*에 의해서 얻은 후보 시퀀스의 집합은 *sim_seq*에 의해 얻은 집합의 슈퍼 집합이 된다.

본 논문에서는 같은 길이의 시퀀스 사이의 유사성 척도만을 제시한다. 다른 길이의 시퀀스 사이의 유사성은 세그먼트의 경우처럼 짧은 시퀀스를 긴 시퀀스의 시작부터 끝까지 슬라이딩하는 방식으로 계산될 수 있다. 한 번 비교할 때마다 슬라이딩하는 양은 또 다른 연구 주제이며, 한 포인트, 여러 개의 포인트, 또는 한 세그먼트 단위 등으로 결정할 수 있다. 슬라이딩하면서 생성된 각각의 시퀀스 쌍의 유사성 값들 중 가장 큰 값을 두 시퀀스 간의 유사성으로 간주할 수 있다.

다음은 패턴 유사성 검색 알고리즘을 기술한 것이다. 전처리 단계로써, 데이터 시퀀스들을 세그먼트 단위로 분할하고 각 세그먼트로부터 트랜드 벡터를 구한 후, 인덱싱하여 데이터베이스에 저장한다. 첫 번째 단계로써, 주어진 질의 시퀀스를 데이터 시퀀스와 마찬가지로 세그먼트로 분할하고 각 세그먼트로부터 트랜드 벡터를 구한다. 다음으로, 질의 시퀀스의 각 세그먼트에 대하여 거리의 척도인 *d_hr*을 사용하여 후보 세그먼트를 추출하기 위하여 인덱스를 검색한다. 이 단계에서 사용되는 거리 임계 값 ε은 사용자가 제공한 유사성 임계값 ζ으로부터 변환하여 구한다. 발견된 후보 세그먼트들의 집합에서 질의와 무관한 세그먼트를 한 번 더 여과하기 위하여 세그먼트 간의 유사성 척도인 *sim_seg*를 사용하여 후보 세그먼트들을 검증하는 단계를 거친다.

선정된 후보 세그먼트를 포함하는 시퀀스들이 후보 시퀀스 집합을 형성하게 된다. 최종 단계로써, 후보 시퀀스 집합에 있는 각 후보 시퀀스에 대하여 시퀀스 간의 유사성 척도인 *sim_seq*를 사용하여 결과 시퀀스를 구한다.

```
Algorithm Video_Similarity_Search
/* 집합의 초기화 */
SET_cand ← ∅   /* 후보 시퀀스의 집합          */
SET_ans ← ∅    /* 결과 시퀀스의 집합          */
      /* i 는 질의 시퀀스 Q 의 세그먼트 번호를 위한 첨자
      */
      /* j 는 데이터 시퀀스 S 의 세그먼트 번호를 위한 첨자
      */
      /* k 는 데이터베이스에 있는 데이터 시퀀스 번호를 위한 첨자*/
Step 0: /* 전처리 과정      */
      각 데이터 시퀀스 S를 하나나 그 이상의 세그먼트로 분할
      각 세그먼트로부터 트랜드 벡터를 추출
      트랜드 벡터를 인덱싱하고 데이터베이스에 저장
Step 1: /* 질의 시퀀스 의 세그멘테이션   */
      질의 시퀀스 Q를 하나나 그 이상의 세그먼트 SEG_Q,i 로 분할
      각 질의 세그먼트 SEG_Q,i 로부터 트랜드 벡터를 추출
Step 2: /* d_hr (인덱스 검색) 에 의한 첫 번째 Pruning      */
      for each SEG_Q,i of a query sequence Q
        거리척도 d_hr을 사용하여 인덱스를 검색하여 후보 세그먼
        트 SEG_k,j를 추출
      /* 세그먼트 간의 유사성 척도 sim_seg를 사용한 두 번째
Pruning */
        for each candidate segment SEG_k,j
          if (sim_seg(SEG_Q,i, SEG_k,j) ≥ ζ) then
            SET_cand ← SET_cand ∪ {S_k}
Step 3: /* 시퀀스 간의 유사성 척도 sim_seq를 사용한 세 번째
Pruning */
      for each selected sequence S_k in the set SET_cand
        if (sim_seq(Q, S_k) ≥ ζ ) then
          SET_ans SET_ans ∪ {S_k}
Step 4: /* 결과 Report */
      return SET_ans
```

5. 실험 결과 및 고찰

이 장에서는 제안한 기법의 효과와 성능을 실험적으로 검증한다. 다차원 데이터 시퀀스로써 비디오 데이터 스트림 뿐만 아니라 프랙탈 함수를 사용해서 생성된 가상 데이터 시퀀스를 이용하여 실험을 행하였다. 가상 데이터 시퀀스를 이용하여 실험한 이유는 실제 데이터는 일반적으로 그 크기와 여러 가지 실험 파라미터의 설정에 제한이 있는 반면, 가상으로 생성된 데이터는 보다 나은 유연성을 제공하기 때문이다. 비디오 데이터 스트림은 다수의 TV 뉴스, 드라마 및 다큐멘터리 필름들을 포함한다.

실험은 질의와 무관한 시퀀스들을 여과하는 데 있어서 제안한 기법의 효과를 보여주는 데 초점을 맞추었다. 실험을 위한 구현 환경으로써, 하드웨어는 Pentium IV 프로세서, 1G메모리, 120GB HDD가 사용되었고, 소프트웨어는 Window 2000 Server 하의 Java 2 SDK1.4 가 사용되었다. 제안한 기법을 평가하기 위하여 기존의 방법을 다차원 시퀀스를 수용하기 위하여 약간 변형하여 사용하였다. <표 1>은 비교 대상이 되는 기법들을 기술하고 있다.

〈표 1〉 비교 대상이 되는 기법들

기 법	설 명
SS	순차 스캐닝에 의한 방법 (시퀀스 내의 모든 포인트 비교)
MB-Same	평균에 기초한 기법 - 같은 길이 세그멘테이션 [18]
MB-Diff	평균에 기초한 기법 - 다른 길이 세그멘테이션 [10]
TV-Same	트랜드 벡터를 사용한 제안한 기법 - 같은 길이 세그멘테이션
TV-Diff	트랜드 벡터를 사용한 제안한 기법 - 다른 길이 세그멘테이션

5.1 실험을 위한 준비

실험을 위해 가상 다차원 데이터 시퀀스(SYNTH) 및 비디오 스트림(VIDEO)을 사용했고, 편의상 SYNTH와 VIDEO는 3 차원 데이터 시퀀스이지만 제안한 기법은 차원에 제한을 두지 않으며 다른 차원에서도 사용 가능하다. 실험을 위하여 유형 별로 다음과 같이 데이터 세트를 준비하였다. SYNTH는 앞에서 기술한 바와 같이 23,500 개의 가상 시퀀스들을 프렉탈 함수을 사용하여 생성하였고, 비디오 스트림에 대한 데이터 시퀀스는 다양한 비디오 데이터 소스의 각 프레임의 픽셀로부터 색상 특징(RGB) 값을 추출하고 그들을 평균함으로 생성되었다. RGB 특징들의 값의 범위는 [0, 255]이며 이를 255로 나누면 $[0, 1]^3$ 단위 큐브 내에 포함되게 된다. 따라서, 각 프레임은 단위 큐브내의 한 포인트로 사상된다. 질의 시퀀스는 데이터 시퀀스로부터 임의로 추출하였으며, SYNTH는 500 개, 그리고 VIDEO는 100 개를 사용하였다. 다음의 표는 실험에 사용된 파라미터들을 요약한 것이다.

〈표 2〉 실험을 위한 파라미터

데이터 세트	SYNTH	VIDEO
시퀀스 개수	23,500	7,500
시퀀스 길이	64-1024 points	64-1024 points
유사성 임계 값	0.70-0.95	0.70-0.95

5.2 실험 결과 및 고찰

유사성 검색은 주어진 질의 시퀀스에 대해서 지정된 임계값 안에 포함되는 유사한 시퀀스들을 데이터베이스에서 검색하는 것이다. 각 테스트에서는 복수 번의 질의를 실행하고 질의 결과의 평균을 취하였다. 다음과 같은 여러 가지 측면에서 제안한 기법과 기존 방법들을 비교 평가하였다.

정밀도(precision)와 리콜(recall) : 정보 검색 분야에서 널리 알려진 정밀도와 리콜에 대하여 평가하였다. $Set(ret)$를 질의에서 검색된(retrieved) 시퀀스의 집합이라 하고, $Set(rel)$를 질의와 유사한(relevant) 시퀀스의 집합이라 하자. 집합 S의 요소의 수를 $|S|$로 표기할 때 정밀도와 리콜은 다음과 같이 정의된다.

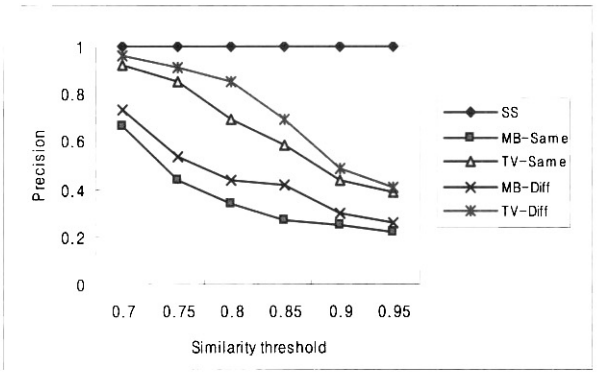
$$precision=\frac{|Set(ret)\cap Set(rel)|}{|Set(ret)|},\ recall=\frac{|Set(ret)\cap Set(rel)|}{|Set(rel)|}$$

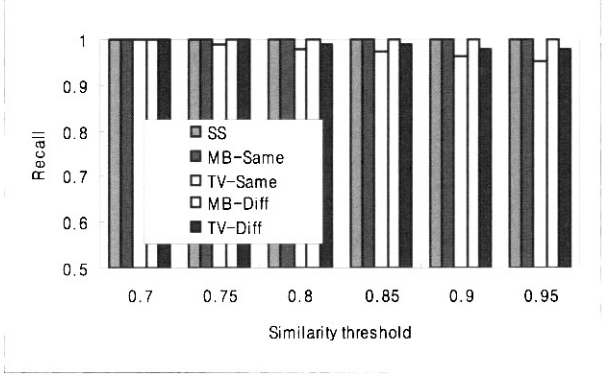
제안한 기법의 효과를 측정하기 위해 검색의 품질을 평가하는 기준(ground truth)을 설정하는 것이 필요하다. 일반적으로 비디오 스트림과 같은 시퀀스 데이터는 대용량의 정보이므로, 사람이 일일이 확인하여 두 시퀀스가 유사한 지를 결정하는 것은 쉽지 않고, 또한 이 경우에는 관찰자의 주관이 개입될 여지가 매우 높다. 본 실험에서는 평가 기준을 SS 방법에 의해 검색된 시퀀스의 집합으로 정하였다. 즉, SS 방법에 의해 검색된 시퀀스들을 질의와 유사한 시퀀스로 간주한다는 의미이다.

여과 효과는 유사성 임계 값을 변경시킴으로써 측정하였다. 실험에서 유사성 임계 값을 0.70 - 0.95 사이의 값으로 정하였는 데, 이는 실험에서 0.70 이하의 값을 갖는 두 시퀀스는 매우 다른 시퀀스인 것으로 판명되어 유사성 검색에서 별 의미가 없기 때문이다.

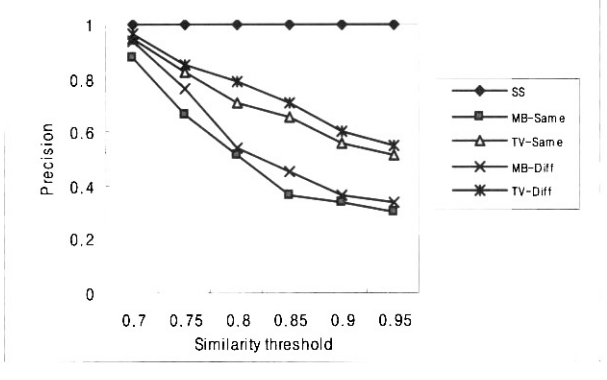
(그림 4), (그림 5)는 가상 데이터 및 비디오 데이터 세트에 대한 정밀도와 리콜 결과를 보여 준다. SS 방법은 기준으로 사용되는 방법으로써 정밀도와 리콜은 당연히 1이 된다. SS방법을 제외하고는 TV-Diff가 정밀도 면에서 가장 좋은 결과를 보여 주며, 이는 다른 기법에 비하여 많은 수의 무관한 시퀀스를 여과했기 때문으로 분석된다. 또한, 비디오 데이터 세트의 정밀도는 다른 데이터 세트에 비하여 더 좋은 결과를 보여 주고 있는데, 이는 비디오 데이터가 일반적으로 샷 경계에서 자연스럽게 분할되기 때문이다.

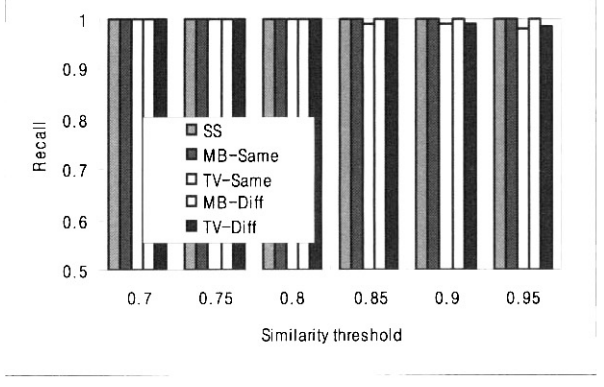
제안한 기법은 평균에 기초한 방법에 비하여 향상된 정밀도를 보여 주고 있다. 같은 길이의 세그멘테이션의 경우, 정밀도는 비디오 스트림에 대해서는 기존 기법에 비하여 평균 1.5 배, 최대 1.8 배까지 개선되었고, 가상 데이터에 대해서는 평균 1.8 배, 최대 2.1 배까지 개선되었음을 알 수 있다.





(그림 4) 가상 데이터 세트에 대한 정밀도 및 리콜 비교

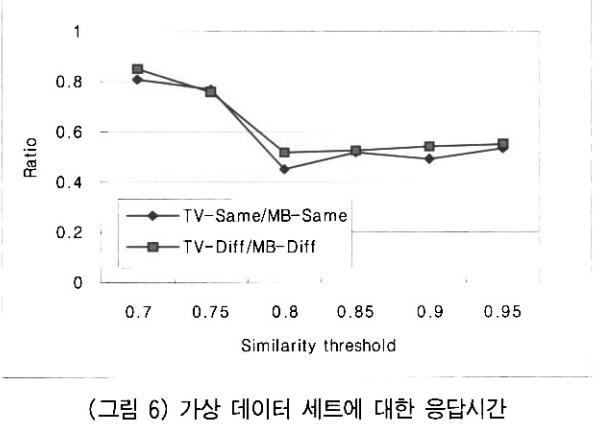




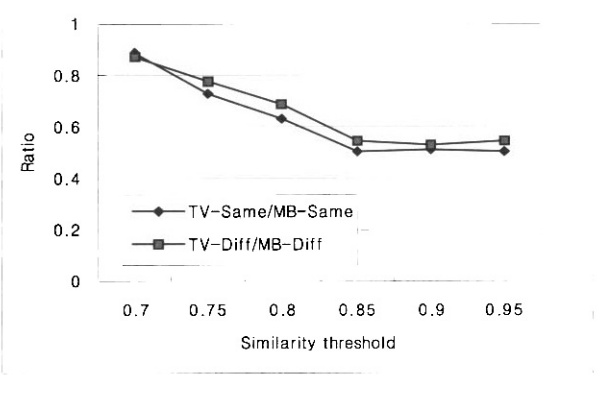
(그림 5) 비디오 데이터 세트에 대한 정밀도 및 리콜 비교

반면에 다른 길이의 세그멘테이션의 경우, 정밀도는 비디오 스트림에 대해서는 평균 1.4 배, 최대 1.7 배까지, 그리고 가상데이터에 대해서는 평균 1.6 배, 최대 1.9 배까지 개선되었음을 보이고 있다. 또 다른 관찰은 정밀도는 모든 데이터세트에 대하여 유사성 임계 값이 증가할수록 감소한다는 점이다. 이것은 작은 임계 값에 대해서는, 검색된 시퀀스의 수에 비하여 검색된 시퀀스 중 질의에 무관한 시퀀스의 수가 더 커지기 때문이다. 리콜에 관해서는 비록 제안한 기법이 정확성(correctness)을 보장하지는 못하지만 95% 이상으로써 상당히 좋은 결과를 보여주고 있다.

[응답 시간] (그림 6), (그림 7)은 각각 가상 데이터와 비디오 데이터 세트에서의 4가지 방법(MB-Same, MB-Diff, TV-Same 및 TV-Diff)의 평균 질의 응답 시간을 보여준다. SS 방법에서의 응답 시간은 너무 큰 값이므로 그림에서 생략하였다. 실험 결과는 TV-Diff 대 MB-Diff의 응답 시간 비(TV-Same/MB-Same)는 비디오 스트림에서는 각각 평균 0.63, 최대 0.50, 가상 데이터 세트에서는 각각 0.60, 0.45가 됨을 알 수 있다. 한편, 응답 시간 비 TV-Diff/MB-Diff 도 앞의 경우와 유사한 경향을 보이고 있으며, 비디오 스트림에서는 각각 0.67, 0.52, 가상 데이터 세트에서는 각각 0.63, 0.51 이 되어 상당히 응답 시간이 개선되었음을 알 수 있다.



(그림 6) 가상 데이터 세트에 대한 응답시간



(그림 7) 비디오 데이터 세트에 대한 응답 시간

이 비율은 유사성 임계 값이 0.8 (가상 데이터 세트) 혹은 0.85 (비디오 데이터 세트)이하에서 감소하고 있고, 그 외 구간에서는 비교적 일정함을 알 수 있다. 이 결과는 제안한 기법이 전 범위의 임계 값에서 기존의 기법 보다 빠르고, 특히 높은 임계 값 범위에서는 더 좋은 성능을 보여주어 이전의 정밀도에 대한 실험 결과와 일관성이 있음을 보여준다. 즉, 높은 정밀도는 질의와 무관한 시퀀스를 보다 많이 여과하며, 이는 성능 향상으로 연결되기 때문이다.

6. 결 론

본 논문에서는 비디오 스트림과 같은 다차원 시퀀스로 표현될 수 있는 데이터 세트에서 유사한 패턴을 데이터베이스에서 검색하는 문제를 연구하였다. 이 문제를 해결하기 위하여 각 시퀀스를 세그먼트로 분할하고, 각 세그먼트를 세그먼트 내의 포인트들의 움직임의 경향을 나타내는 트랜드 벡터로 표현하였다. 트랜드 벡터의 표현 방식은 보다 향상된 근사 방법을 제시한다는 점에서 기존에 제시된 평균에 기초한 방법들에 비하여 장점을 가지고 있다. 트랜드 벡터를 기초로 세그먼트 간, 그리고 시퀀스 간의 유사성의 척도를 정의하였으며, 이 척도를 사용하여 데이터베이스로부터 질의에 무관한 시퀀스들을 여과하는 알고리즘을 제시하였다.

또한, 본 논문에서는 비디오 스트림 등의 실세계에 존재하는 데이터 세트뿐만 아니라 가상으로 생성된 데이터에 관해서 폭 넓은 실험을 수행하였다. 실험 결과, 제안한 기법은 기존의 평균에 기초한 방법에 비하여 정밀도는 2.1 배까지 향상되었고 응답 시간은 기존 방법에 비해 45% 까지 감소된 반면, 리콜은 거의 비슷한 수준으로 유지되고 있음을 관찰할 수 있었다. 본 연구에서는 비디오 스트림에 초점을 맞추어 유사성 검색 기법을 제시하였지만, 데이터의 형식이 다차원 시퀀스로 표현될 수 있는 다른 응용 분야들에서도 또한 제안한 방법을 적용할 수 있을 것이다.

참 고 문 헌

[1] R. Agrawal, C. Faloutsos, A. Swami. Efficient similarity search in sequence databases. Proc. of Foundations of Data Organizations and algorithms (FODO), pp.69-84, Evanstone, Illinois, 1993.

[2] S. Berchtold, D. Keim, H. Kriegel. The X-tree: an index structure for high-dimensional data. Proc. of Int'l Conference on VLDB, pp.28-39, Bombay, India, 1996.

[3] N. Beckmann, H. Kriegel, R. Schneider, B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. Proc. of ACM SIGMOD, pp.322-331, Atlantic City, New Jersey, 1990.

[4] C. Faloutsos, H.V. Jagadish, A. Mendelzon, and T. Milo. A signature technique for similarity-based queries. SEQUENCES, Italy, 1997.

[5] C. Faloutsos, M. Ranganathan, Y. Manolopoulos. Fast subsequence matching in time-series databases. Proc. of ACM SIGMOD, pp.419-429, Minnesota, 1994.

[6] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker. Query by image and video content: The QBIC system. IEEE Computer,Vol.28, No.9, pp.23-32, 1995.

[7] A. Guttman. R-trees: a dynamic index structure for spatial searching. Proc. of ACM SIGMOD, pp.47-57, Boston, Massachusetts, 1984.

[8] H. V. Jagadish. Linear clustering of objects with multiple attributes. Proc. of ACM SIGMOD, pp.332-342, Atlantic City, New Jersey, 1990.

[9] N. Katayama, S. Satoh. The SR-tree: an index structure for high-dimensional nearest neighbour queries. Proc. of ACM SIGMOD, pp.369-380, Tucson, Arizona, 1997.

[10] E. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. Proc. of ACM SIGMOD, pp.151-162, 2001.

[11] E. Keogh and P. Smyth. A probabilistic approach to fast pattern matching in time series databases. Proc. of Int'l Conference of Knowledge Discovery and Data Mining, pp.20-24, 1997.

[12] S. L. Lee, S. J. Chun, D. H. Kim, J. H. Lee, and C. W. Chung. Similarity search for multidimensional data sequences. Proc. of IEEE ICDE, pp.599-608, 2000.

[13] S. L. Lee and C. W. Chung. Hyper-rectangle based segmentation and clustering of large video data sets. Information Science, Vol.141, No.1-2, pp.139-168, 2002.

[14] T. Pavlidis. Waveform segmentation through functional approximation. IEEE Transactions on Computers, Vol.C-22, No.7, 1976.

[15] D. Rafiei. On similarity queries for time series data. Proc. of IEEE ICDE, pp.410-417, Sydney, Australia, 1999.

[16] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. Proc. of ACM SIGMOD, pp.13-25, Tucson, Arizona, 1997.

[17] C. Wang and S. Wang. Supporting content-based searches on time series via approximation. Int'l Conference on

Scientific and Statistical Database Management, 2000.

[18] B. K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary Lp norms. Proc. of Int'l Conference on VLDB, pp.385-394, 2000.

[19] 이석룡, "비디오 데이터 세트의 하이퍼 사각형 표현에 기초한 비디오 유사성 검색 알고리즘", 정보처리학회논문지D, 제11권 제4호, 2004년 8월, pp.823-834.

[20] 이석룡, 전석주, 이주홍, "의미정보를 이용한 다차원 데이터 시퀀스의 유사성 척도 연구", 정보처리학회논문지D, 제10권 제2호, 2003년 4월, pp.283-292.

이 석 룡

e-mail : sllee@hufs.ac.kr

1984년 연세대학교 기계공학과(학사)

1993년 연세대학교 산업공학과 전자계산 전공(석사)

2001년 한국과학기술원 정보및통신공학과 컴퓨터공학전공(박사)

1984년~1995년 한국IBM 소프트웨어 연구소 선임연구원

1995년~2001년 안산1대학 전자계산학과 조교수

2002년~현재 한국외국어대학교 산업정보시스템공학부 부교수

관심분야 : 멀티미디어 데이터베이스, 데이터마이닝, 정보검색