

비디오 스트리밍 서비스를 위한 의미 기반 비디오 표현 및 저장 기법¹

Semantic-Based Video Representation and Storing Techniques for Video Streaming Services

이석룡

한국외국어대학교 산업정보시스템공학부

Abstract

본 논문에서는 비디오 스트리밍 서버에서 의미 기반 검색을 가능하게 하기 위하여 대용량 스트림 데이터를 효과적으로 표현하고 저장하는 기법을 제시한다. 비디오 스트림 내의 각 프레임은 다차원 공간상의 점으로 사상함으로써 비디오 스트림은 다차원 시퀀스(multidimensional sequence)로 표현되고, 이 시퀀스는 다시 비디오 세그먼트로 분할된다. 분할된 세그먼트로부터 정적인 특성과 연속된 프레임의 움직임을 나타내는 트렌드 벡터(trend vector) 등의 의미 정보를 추출하여 모델링 함으로서 스트림 데이터를 효과적으로 표현한다. 또한 제안된 기법은 효율적인 검색을 위하여 비디오 세그먼트를 인덱싱하고 저장하는 방법을 제공함으로써 공간 사용의 효율성을 높이고 신속한 검색을 가능하게 한다.

1. 서론

비디오 스트리밍 서버는 비디오 스트림들을 특정 형태로 저장하고 있으며, 클라이언트의 요청이 있을 때 실시간 제약사항(real-time constraints)을 만족시키면서 사용자가 원하는 비디오 스트림을 검색하여 전송해주는 시스템이다. 이러한 관점에서 비디오 서버가 갖추어야 할 조건은 첫째, 다수의 사용자에게 동시에 서비스를 제공해야 하고, 둘째, 사용자의 요구에 대하여 일정 기준에 맞는 신속한 서비스를 제공해야 한다는 점이다. 이러한 요건을 충족시키기 위해서는 대용량, 비정형의 특성을 갖는 비디오 스트림을 효과적으로 표현하고 저장하는 기법이 필수적이다. 본 논문에서는 대용량 스트림 데이터에 대한 신속한 검색과 의미 기반 검색(semantic-based retrieval)을 위하여 스트림 데이터를 효과적으로 표현하고 저장하는 기법을 제시한다. 의미 기반 검색이라 함은 기존의 대부분의 내용 기반 검색(content based retrieval) 방식이 주로 비디오 프레임의 색상과 질감, 모양 등의 정적인 특징(static feature)을 추출하여 검색에 이용한 데 비하여, 본 논문에서는 정적인 특징 뿐 아니라 연속된 프레임의 움직임의 방향성 등의 의미 정보를 검색에 이용함을 의미한다. 비디오 스트림의 효과적 표현을 위하여 먼저 비디오 스트림 내의 각 프레임

및 프레임 내의 특정 영역을 다차원 공간상의 점으로 사상(mapping)하고, 이 점들을 시간과 의미정보를 사용하여 비디오 세그먼트로 분할하며, 이 세그먼트로부터 의미정보를 추출하여 보다 간결한 형태로 표현함으로써 비교적 적은 공간에 저장할 수 있도록 한다.

일반적으로 비디오 데이터는 각 샷(shot)을 대표하는 키 프레임(key frame)들의 집합으로 표현되고 이 샷들을 사용하여 인덱스를 구성하여 비디오 검색에 사용한다. 대표적으로는 QBIC(Query by image and video content) 시스템[4], VisualSEEK[8], Virage[5] 등이 있으며, 여기에서는 각각의 샷 당 소수로 대표되는 키 프레임을 정하여 각각의 키 프레임에 대하여 질의가 처리된다. 그러나, 일반적으로 한 샷을 한 두 개의 대표 프레임으로 나타내는 것은 유용한 정보를 상실하는 위험이 초래될 수 있다. 같은 샷에 속한 프레임일지라도 시간적으로 상당히 떨어져 있을 경우 많이 다를 수 있으며, 따라서 키 프레임에 의한 검색은 질의 프레임과 관련이 있는 프레임들을 검색에서 배제하게 되며, 이는 정적인 특징만을 추출하여 검색에 이용하는 기법의 한계점이다. 그러므로, 정적인 특징 뿐 아니라 연속된 프레임의 움직임의 방향성, 비디오 내 객체의 움직임 등의 의미 정보를 검색에 이용하는 것이 바람직하다.

한편, 일 차원 시계열 데이터에 대한 유사성 검색이 데이터베이스 응용 분야에서 중요한 연구 주제 중 하나로 관심을 받아 왔다. Agrawal 등[1]의 DFT(discrete Fourier transform)를 사용한 검색, Faloutsos 등[3]의 슬라이딩 윈도우를 사용한 빠른 서브 시퀀스 매칭 방법 등이 그것이다. 최근에 Yi 등 [9]은 시간 시퀀스를 같은 길이의 세그먼트로 나누고 세그먼트내의 포인트들의 평균을 저장하는 근사 기법(approximating technique)을 제안하였다. 이 기법에서는 한 시퀀스가 s 개의 세그먼트로 나누어지며 그 시퀀스는 s 개의 세그먼트 평균(segmented mean)들을 요소로 가지는 벡터로 표현된다. [9]를 확장하여 Keogh 등 [6]은 APCA(adaptive piecewise constant approximation)로 불리는 새로운 차원 축소 기법을 소개하였다. 이 기법에서는 하나의 시간 시퀀스를 복원 오차가 최소가

¹ 이 논문은 2003년도 한국학술진흥재단의 지원에 의하여 연구되었음. (KRF-2003-041-D00628)

되도록 하는 길이가 다른 복수 개의 세그먼트의 집합으로 근사하여 표현하였다. 또한, APCA 에서는 각각의 길이가 다른 세그먼트가 다차원 인덱스 구조로 인덱싱될 수 있음을 보였고, D_{LB} 와 D_{AB} , 즉 각각 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 방법의 문제점은 세그먼트를 단순히 점이나 최소 경계 사각형으로 표현함으로써 세그먼트 내의 점들의 움직임에 관한 중요한 정보를 상실하게 된다는 점이다. 또 다른 문제점으로써, 평균에 기초한 접근 방법들[9, 6]은 원래의 시퀀스에 대하여 본질적으로 높은 복원 오차를 초래하게 된다. 이 방법에서 세그먼트 내의 점들의 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.

2. 비디오 스트림의 표현 및 저장 기법

2.1 제안한 기법의 개요

제안한 기법은 먼저, 비디오 스트림의 각 프레임으로부터 적절한 특징 추출(feature extraction) 과정을 거쳐 각 프레임을 다차원 공간상의 점으로 나타냄으로써 비디오 스트림을 다차원 시퀀스로 표현한다. 추출하는 특징 정보에는 색상, 질감, 모양 등의 기본적인 정보와 윤곽 정보, 그리고 영상 내 개체 간의 위치 정보들이 포함된다. 색상의 경우, 색상 히스토그램 (color histogram), 평균 색상, 대표 색상 등을 추출한다.

다음으로, 시퀀스를 같거나 다른 길이의 세그먼트로 분할한다. 같은 길이의 세그멘테이션은 시퀀스를 세그먼트들로 분할할 때 분할된 모든 세그먼트들이 같은 길이를 갖도록 하는 것이다. 후자의 세그멘테이션 방법은 분할된 세그먼트들이 임의의 길이를 갖도록 하는 것이다. 본 논문에서는 [7]에서 제안된 기법, 즉 시퀀스의 기하학적이고 의미적인 특성에 따라서 세그멘테이션을 수행하는 기법을 활용한다. 임의의 길이의 세그멘테이션 방법에는 다양한 최적화 표현 기법들(optimal piecewise polynomial representations)이 있으나 이러한 방법들은 상당한 오버헤드가 야기된다. 반면에 비록 최적의 기법은 아니지만 그리디 접근법(greedy approaches)이 더욱 효율적이고 또한 현실적인 선택이 될 수 있다. 지면상의 제약 때문에 본 논문에서는 세그멘테이션 알고리즘을 상세하게 기술하지는 않는다. 세그멘테이션의 조건, 알고리즘, 그리고 실험 결과와 같은 자세한 사항들에 대해서는 [7]을 참고하기 바란다.

각 세그먼트로부터 세그먼트의 다양한 특성들을 고려하여 트렌드 벡터가 계산된다. 이것은 한 세그먼트내의 점들이 가지는 값과 점들의 움직임의 경향을 요약하여

표현하며 유사성 검색을 위해 사용된다. 그 다음으로, 각 세그먼트의 트렌드 벡터를 내포(minimum bounding)하는 하이퍼 사각형을 나중의 검색 과정을 위해 데이터베이스에 인덱싱하고 저장한다. 이를 위해 제안한 방법은 R*-트리[2]를 사용한다. 이것은 R*-트리가 다양한 응용 도메인에서 널리 사용되고 있는 검증된 인덱싱 구조이기 때문이다.

2.2 트렌드 벡터

세그먼트 내의 데이터 점들의 트렌드를 나타내기 위하여 어떤 근사 선(approximation line)을 사용하는 것이 좋은가? 그림 1(a)의 세그먼트 SEG를 고려해 보자. 그림에 도시한 것과 같이 한 세그먼트는 그 세그먼트 내의 점들의 궤적이 이루는 곡선의 기울기가 0인 점(즉, 고점과 저점)을 기준으로 하여 복수 개의 선분으로 분할할 수 있다. 이 선분들 중에서 다른 선분 보다 많은 점을 포함하는 지배 선분(dominant line)을 선택하고 다른 선분들은 무시함으로써, 그 선분을 해당 세그먼트의 점들의 트렌드로 간주할 수 있다. 그러나 이러한 접근법은 지배 선분이 실제로는 다른 선분에 비하여 '지배적'이지 않는 경우 - 세그먼트 내의 각 선분이 비슷한 크기일 때 - 가 흔히 발생할 수 있으므로 바람직하지 못하다. 이러한 경우에는 지배 선분의 대표성이 약해지게 된다. 보다 바람직한 선택으로 세그먼트의 시작 점과 끝 점을 잇는 선분을 지배 선분으로 선택하는 안이 있을 수 있다. 이 경우에는 이 선분이 세그먼트 내의 모든 선분들을 연결하여 생성되는 벡터 합(vector sum)과 동일하게 되므로 보다 설득력이 있는 선택이 된다. 따라서, 세그먼트의 시작 점과 끝 점을 연결하여 생성되는 벡터 SE의 기울기를 그 세그먼트 내의 점들의 움직임의 경향을 나타내는 것으로 간주한다.

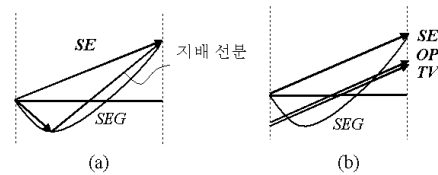


그림 1. (a) 세그먼트 내의 점들의 움직임을 대표하는 선분의 결정 (b) 세그먼트의 시작-끝 벡터(SE), 최적 벡터(OP) 및 트렌드 벡터(TV)

그러나 벡터 SE를 세그먼트를 대표하는 특징으로 채택하는 것은 비록 그것이 점들의 이동 경향을 잘 표현하고 있다고는 하지만 적절하지 못하다. 왜냐하면 이 방식은 복원 오차가 매우 높을 것이기 때문이다. 그림 1(b)를 고려해 보자. 벡터 SE에 평행하면서 복원 오차를 최소로 하는 최적 벡터(optimal vector) OP를 계산할 수 있다. 그러나 벡터 OP를 계산하는 것은 심각한 계산 오버헤드를 야기한다. 따라서, 세그먼트를 대표하기 위한 최적에 근사한 벡터로써 벡터 SE에 평행하면서 세그먼트의 평균 선(mean line)의 중간을 지나는 벡터를 채택하고, 이를 트렌드 벡터(trend vector: TV)라 한다. 평균 선의 중간을

지나는 벡터를 채택하는 것은 복원 오차율을 획기적으로 감소시키며, TV 를 계산하는 것은 단순하고 매우 빠르다. d -차원 공간에서 트렌드 벡터 TV 는 $TV=(k, m, \alpha)$ 로 정의되며, 여기에서 k 는 세그먼트 안에 포함되어 있는 점의 수이며, $m=(m_1, m_2, \dots, m_d)$ 은 각 차원에서의 데이터 점들의 평균으로 구성되어 있는 세그먼트의 평균 벡터이고, $\alpha=(\alpha_1, \alpha_2, \dots, \alpha_d)$ 는 세그먼트의 시작 점으로부터 끝 점을 잇는 선분의 각 차원에서의 기울기를 나타내는 기울기 벡터이다. 트렌드 벡터를 사용하여 세그먼트 SEG 는 다음 형식으로 표현될 수 있다.

$$SEG = \langle TV \rangle = (k, m, \alpha)$$

$\alpha = \text{NULL}$ 으로 하면 위의 표현은 'APCA 표현[6]'으로 축소될 수 있으며, $k = \text{NULL}$ 과 $\alpha = \text{NULL}$ 로 함으로써 위의 표현은 '세그먼트-평균 표현[9]'으로 축소될 수 있다.

2.3 세그먼트 내의 점들의 근사

그림 2에서와 같이 k 개의 점 P_0, P_1, \dots, P_{k-1} 을 포함하고 있는 세그먼트 SEG 를 고려해 보자. 세그먼트-평균 표현 방식에 따르면 세그먼트 내의 모든 점들은 하나의 값, 즉 평균 값으로 근사하게 된다. 반면에 트렌드 벡터를 사용하게 되는 경우에는 세그먼트의 각 점 P_0, P_1, \dots, P_{k-1} 은 트렌드 벡터 선상의 각 대응점인 $P'_0, P'_1, \dots, P'_{k-1}$ 으로 근사되게 된다. 차원 D 에 대하여 점 P_i 와 P'_i 의 값을 각각 x_i 와 x'_i 라 하자.

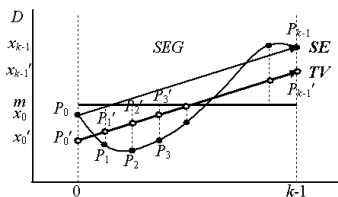


그림 2. 세그먼트-평균과 트렌드 벡터에 의한 세그먼트의 근사

d -차원의 공간에서 세그먼트-평균 표현 방식에 의한 복원 오차율은 다음 식으로 나타낼 수 있다.

$$RE_{mean} = \frac{1}{k} \cdot \sum_{i=0}^{k-1} \sqrt{\sum_{j=1}^d (x_{i,j} - m_j)^2}$$

한편, 벡터 TV 의 기울기 α 는 식 $\alpha = (x_{k-1} - x_0)/(k-1) = (x_{k-1} - x_0)/(k-1)$ 로 주어진다. 벡터 TV 에 의해 표현되는 직선의 방정식을 $f(i) = \alpha i + \beta$ 라 하면, 이 직선이 점 $((k-1)/2, m)$ 을 지나게 되므로 $\beta = m - \alpha(k-1)/2$ 을 얻을 수 있다. 따라서 차원 j 에 관한 식은 다음과 같이 된다.

$$x_{i,j}' = \alpha_j \cdot i + \left(m_j - \alpha_j \cdot \frac{(k-1)}{2} \right)$$

그러므로 d -차원의 공간에서 TV 벡터 표현 방식에 의한 복원 오차율은 다음 식으로 정의될 수 있다.

$$RE_{TV} = \frac{1}{k} \cdot \sum_{i=0}^{k-1} \sqrt{\sum_{j=1}^d (x_{i,j} - x_{i,j}')^2}$$

2.4 세그먼트의 인덱싱

효율적인 검색을 위하여 각 시퀀스의 세그먼트들을 공간 인덱스 구조를 사용하여 인덱싱하고 데이터베이스에 저장한다. 세그먼트의 인덱싱 방식에 관해서는 여러 가지 선택이 있을 수 있다. 그림 3은 2차원 공간에서 이러한 선택들을 보여주고 있다. 먼저, 세그먼트 내의 모든 점들을 내포하는 MBR을 선택하는 것이다. 이 경우, 세그먼트는 두 점 (min_1, max_1) 과 (min_2, max_2) 로 표현할 수 있다. 이 선택은 널리 알려져 있는 방식이며 'no-false dismissal'의 특성을 유지한다. 그러나 이 선택은 수많은 'false hits'가 발생할 가능성이 있으며, 또한 $\langle TV \rangle$ 이외에도 두 점 (min_1, max_1) 과 (min_2, max_2) 을 저장해야 되는 저장 오버헤드가 있으므로 바람직한 선택은 아니다. 다음으로, 그림 7에서 보인 바와 같이 벡터 TV 자체를 최소한으로 포함하는 사각형을 선택하는 방안이다. 이 선택은 비록 'no-false dismissal'을 보장하지는 못하지만 MBR을 선택하는 것에 비하여 검색 공간을 줄일 수 있고, 따라서 검색 속도를 향상시킨다는 점에서 바람직한 방안이라 할 수 있다. 이것은 효율성(efficiency)과 정확성(correctness) 사이의 타협(trade-off)이라 할 수 있다. 실제로 많은 응용 분야에서 정확성을 엄격하게 요구하지는 않는다. 또한, TV 자체를 포함하는 사각형을 선택하는 방안은 이 사각형의 저점과 고점을 2.3절의 식으로 간단히 계산할 수 있으므로 MBR 방안과 같이 추가의 저장 공간을 필요로 하지 않는다.

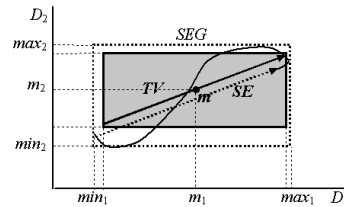


그림 3. 세그먼트를 인덱싱하기 위한 사각형의 선택

3. 실험 및 고찰

이 장에서는 제안한 방법의 효과와 성능을 실험적으로 검증한다. 프랙탈 함수를 사용해서 생성된 가상 데이터 시퀀스와 실제 비디오 데이터 스트림 등을 이용하여 실험을 행하였다. 비디오 데이터 스트림은 다수의 TV 뉴스, 드라마 및 다큐멘터리 필름들을 포함한다.

실험을 위해 사용한 모든 데이터 세트는 편의상 3차원으로 고정하였으나 제안한 방법은 차원에 제한을 두지 않으며 다른 차원도 실험에서 사용 가능하다. 다음의 표는 실험에 사용된 파라미터들을 요약한 것이다.

표 1. 실험을 위한 파라미터

데이터 세트	SYNTH	VIDEO
시퀀스 개수	4,500	2,300
시퀀스 길이	64-1024 points	64-1024 points
차원	3	3

데이터 시퀀스의 각 유형 (SYNTH, VIDEO)는 각각의 길이 (64, 128, 256, 512, 1024)에 따라서 5개 그룹으로 분류된다.

본 실험에서는 MB-Diff(다른 길이 세그멘테이션을 이용한 평균에 기초한 방법) [6]과 TV-Diff(제한한 방법: 다른 길이 세그멘테이션을 이용한 트렌드 벡터에 기초한 방법)의 복원 오차율에 관하여 비교하였다.

그림 4와 5에서 가장 데이터의 접두어는 'S', 비디오는 'V'로 각각 나타내었다. 각 데이터 세트는 5개 그룹(1~5)으로 시퀀스의 길이(64~1024)에 따라 분류하였다. 예를 들어, V3은 길이 256의 비디오 데이터 세트를 나타낸다. 복원 오차는 2.3절에서 제시한 식을 이용해서 계산된다. 그림 4은 MB-Diff와 TV-Diff사이의 복원 오차를 비교한 것이다. 실험 결과, TV-Diff의 복원 오차가 원래 시퀀스에 대해 MB-Diff와 비교해서 평균 63%, 그리고 최고 70%까지 감소하였다.

MB-Diff와 TV-Diff에서 세그먼트를 저장하기 위한 저장공간을 고려하자. MB-Diff에서는 세그먼트의 길이와 평균을 저장하기 위해 2개의 요소가 필요하다. 그러나, TV-Diff에서는 이에 더하여 기울기를 더 저장해야 한다. 3차원 공간의 경우를 예로 보면, MB-Diff에서는 4개의 요소를 저장해야 한다. k를 위해 1개와 m을 위해 3개의 숫자를 저장하기 위함이다. 반면 TV-Diff에 대해서는 7개 요소를 저장해야 한다. 같은 저장 공간 요구 조건 하에서의 대등한 비교를 위해, TV-Diff의 세그먼트 평균 길이를 MB-Diff보다 거의 2배로 할 필요가 있다. 그림 5는 이와 같은 적절한 조정을 한 후의 MB-Diff와 TV-Diff사이의 복원 오차를 비교한 그래프이다. 이 경우에서의 실험 결과는 TV-Diff의 복원 오차가 원래의 시퀀스에 대해서 MB-Diff와 비교해서 평균 52%, 그리고 최고 65%까지 감소하였음을 관찰할 수 있다.

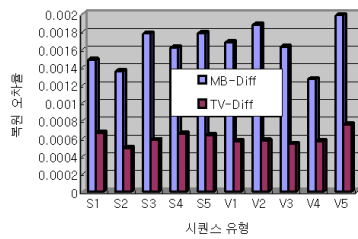


그림 4. MB-Diff와 TV-Diff사이의 복원 오차 비교

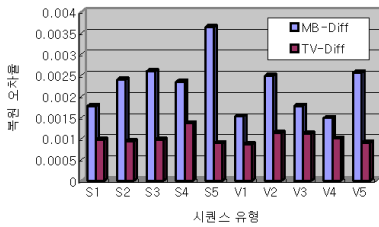


그림 5. 대등한 저장 공간 요구 조건을 고려한 MB-Diff와 TV-Diff사이의 복원 오차 비교

4. DISCUSSION - 비디오 스트림 질의 처리

비디오 스트림 질의 처리는 데이터베이스에 저장된 비디오 의미정보와 질의 비디오로부터 추출된 의미정보를 비교함으로써 수행된다. 두 비디오 스트림 간의 유사성 함수에 의거하여 스트림 간의 유사성이 결정되는 데, 이 유사성은 두 객체 간의 유클리디안 거리(Euclidean distance)에 의하여 정의된다. 색상의 경우, RGB, YCbCr 색상 공간에서 유클리디안 거리가 유사성 함수로 사용된다. HSV 공간에서는 특징들이 극좌표에서 콘의 형태로 표현되기 때문에 색상의 유사도를 측정하기 위하여 각도 거리(angular distance)가 사용된다. 비디오 프레임의 색상, 질감, 모양 등의 기본적인 정보와 연속된 프레임의 움직임의 방향성, 비디오 내 객체의 움직임 등의 의미 정보 등에 기초한 비디오 스트림 유사성 함수를 정의하고, 이러한 유사성 함수를 이용한 비디오 스트림 비교/검색 알고리즘을 개발할 계획이다.

5. 결론 및 향후 연구 계획

본 논문에서는 비디오 스트림 서버에서 효과적인 비디오 검색을 지원하기 위하여 스트림 데이터를 표현하고 저장하는 문제를 연구하였다. 이 문제를 해결하기 위하여 각 시퀀스를 세그먼트로 분할하고, 각 세그먼트를 세그먼트 내의 점들의 움직임의 경향을 나타내는 트렌드 벡터로 표현하였다. 제한한 방법의 트렌드 벡터 표현은 세그먼트의 점들의 원래 값에 대한 매우 뛰어난 근사 방법을 제공하므로 원래의 시퀀스에 대해 매우 낮은 복원 오차를 보여주었다. 향후 연구 과제로써, 트렌드 벡터를 사용하여 스트림 서버에서 비디오 데이터를 검색하는 문제에 대하여 연구할 계획이다.

참고 문헌

- [1] R. Agrawal, C. Faloutsos, A. Swami. Efficient similarity search in sequence databases. *Proc. of Foundations of Data Organizations and algorithms (FODO)*, pages 69-84, Evanstone, Illinois, 1993.
- [2] 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*, pages 322-331, Atlantic City, New Jersey, 1990.
- [3] C. Faloutsos, M. Ranganathan, Y. Manolopoulos. Fast subsequence matching in time-series databases. *Proc. of ACM SIGMOD*, pages 419-429, Minnesota, 1994.
- [4] 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, pages 23-32, September 1995.
- [5] A. Hampapur, A. Gupta, B. Horowitz, C. F. Shu, C. Fuller, J. Bach, M. Gorkani, R. Jain, "Virage Video Engine, Proceedings of SPIE Storage and Retrieval for Image and Video Databases," pages 188-197, San Jose, USA, 1997.
- [6] E. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *Proc. of ACM SIGMOD*, pages 151-162, 2001.

- [7] 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, pages 139-168, 2002.
- [8] J. R. Smith and S.-F. Chang. "VisualSEEK: A Fully Automated Content-based Image Query System," ACM Multimedia 96, Boston, MA, 1996.
- [9] B. K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary Lp norms. *Proc. of Int'l Conference on VLDB*, pages 385-394, 2000.