내용 기반 멀티미디어 정보 검색을 위한 근사 k-최근접 데이타 탐색 알고리즘

(An Approximate k-Nearest Neighbor Search Algorithm for Content-Based Multimedia Information Retrieval)

송 광 택 † 장 재 우 ††
(Kwang-Taek Song) (Jae-Woo Chang)

요 약 내용 기반 멀티미디어 정보 검색에서 유사성에 기반한 k-최근접 데이타 탐색 질의는 매우 중요한 질의이다. 일반적으로 멀티미디어 데이타는 고차원 특징 벡터로 표현되기 때문에 기존의 k-최근접 탐색 알고리즘은 멀티미디어 정보 검색에 효율적이지 못하다. 따라서 이러한 응용을 위해서는 다소 근사적 검색 결과를 가져오더라도 빠른 검색 성능을 제공하는 근사 k-최근접 탐색 알고리즘이 요구된다. 이를 위해 본 논문에서는 고차원 데이타를 위한 새로운 근사 k-최근접 탐색 알고리즘을 제안한다. 아울러, 제안하는 근사 k-최근접 탐색 알고리즘을 기존의 알고리즘과 검색 성능면에서 성능 평가를 수행한다. 성능 평가 결과, 기존 알고리즘의 검색 성능을 크게 개선할 수 있었다.

Abstract The k-nearest neighbor search query based on similarity is very important for content-based multimedia information retrieval(MIR). The conventional exact k-nearest neighbor search algorithm is not efficient for the MIR application because multimedia data should be represented as high dimensional feature vectors. Thus, an approximate k-nearest neighbor search algorithm is required for the MIR applications because the performance increase may outweigh the drawback of receiving approximate results. For this, we propose a new approximate k-nearest neighbor search algorithm for high dimensional data. In addition, the comparison of the conventional algorithm with our approximate k-nearest neighbor search algorithm is performed in terms of retrieval performance. Results show that our algorithm is more efficient than the conventional ones.

1. 서 론

기존의 데이타베이스 시스템이 숫자나 문자열과 같은 단순한 형태의 데이타를 처리하는데 주력해 온 반면, 최근 들어 컴퓨터 기술의 발달과 인터넷의 확산으로 누구나 쉽게 멀티미디어 자료를 이용할 수 있게 됨으로서, 멀티미디어 데이타베이스를 기반으로 하는 정보 서비스가 대두되고 있다. 한편, 멀티미디어 데이타베이스에 대한 사용자의 다양한 요구를 보다 효율적으로 만족시키기 위하여 효과적인 내용 기반 멀티미디어 검색 기법이 요구된다. 실질적으로, 멀티미디어 데이타는 단순히 기존의 텍스트 정보 뿐만 아니라 이미지, 비디오 등과 같은 다양한 형태의 자료가 복합적으로 구성되어 있다. 또한 텍스트 속성에서는 나타나지 않는 시간적 또는 공간적 속성 정보를 지니며, 자료의 크기가 가변적이고 방대하기 때문에 기존의 데이타 모델링 기법으로는 효율적으로 표현할 수 없다. 따라서 멀티미디어 정보를 효과적으로 검색하기 위해서는 기존의 텍스트를 위한 정보 검색 시스템과는 다른 내용 기반 멀티미디어 정보 검색이 요구된다.

내용 기반 멀티미디어 정보 검색을 위한 중요한 패러다임의 하나는 유사성 검색이다. 유사성 검색은 주어진 질의 객체와 유사한 객체들의 집합을 찾는 질의로써, 실제 객체들간의 유사성을 측정하는 것이 아니라 객체로부터 추출된 n(≥ 1)개의 특성값(feature value)들간의

· 본 연구는 한국과학재단의 97목적기초과제(97-0100-0101-3)의 연구비에 의해 수행되었음
† 학생회원 : 전북대학교 컴퓨터공학과
ktsong@dblab.chonbuk.ac.kr
†† 종신회원 : 전북대학교 컴퓨터공학과 교수
jwchang@dblab.chonbuk.ac.kr
논문접수 : 1999년 4월 21일
심사완료 : 2000년 2월 9일

유사성에 기반한다. 유사성에 기반한 검색 질의 중, k-최근접 객체 탐색 질의는 주어진 질의 객체와 가장 유사한 k(≥1)개의 최근접 객체를 찾는 질의로써 대부분의 멀티미디어 검색 시스템에서 지원하는 중요한 질의이다. 기존의 GIS와 같은 응용 분야에서는 저장되는 객체의 특징의 수가 상대적으로 많지 않기 때문에 정확(exact) k-최근접 데이타를 탐색하는 데 큰 문제가 되지 않는다. 반면, 내용 기반 멀티미디어 정보 검색의 응용에서는 데이타베이스는 매우 대량화되어지고 멀티미디어 객체로부터 추출되는 특징 벡터도 고차원화(대개 10차원 이상) 되는 경향이 있기 때문에, 이에 대한 적중 에러 없는 정확 k-최근접 데이타 탐색은 상당히 많은 디스크 접근 횟수를 요구한다.

본 논문에서는 동적인 데이타베이스 환경에서 대용량의 멀티미디어 데이타베이스에 적합한 새로운 근사 k-최근접 탐색 알고리즘을 제안한다. 제안한 근사 k-최근접 객체 탐색 알고리즘은 다소의 적중 에러는 허용하더라도 디스크 접근 횟수를 줄임으로써 사용자에게 보다 빠른 검색 성능을 제공할 수 있다. 대용량의 멀티미디어 데이타베이스에서 보다 빠른 k-최근접 탐색 질의를 지원하는 것은 멀티미디어 정보 검색과 같은 응용분야에서는 매우 중요한 시스템 요구 사항이다.

본 논문의 구성은 다음과 같다. 2장에서는 유사성에 기반하여 효율적인 검색을 위한 R-트리 유형 기반의 정확 k-최근접 탐색 알고리즘과 기존에 제시된 근사 k-최근접 탐색 알고리즘을 고찰하고, 3장에서는 고차원 특징 공간에서 효율적인 k-최근접 탐색 질의를 지원하기 위한 새로운 근사 k-최근접 탐색 알고리즘을 제안한다. 4장에서는 기존의 정확 및 근사 k-최근접 알고리즘과 본 논문에서 제안한 방법을 페이지 접근 횟수, 검색 시간 측면에 서 성능평가를 수행한다. 마지막으로 5장에서는 본 논문의 결론과 향후 연구방향을 제시한다.

2. 관련 연구

내용 기반 멀티미디어 정보검색 시스템은 멀티미디어 객체로부터 n(≥ 1)개의 특징 벡터를 추출하여 데이타베이스에 저장한다. 멀티미디어 객체로부터 추출된 특징 벡터는 기존의 GIS 시스템과는 달리 매우 고차원의 특성을 지닌다. 기존 R*-트리[1]를 비롯한 여러 색인기법들은 차원의 수(특징의 수)가 증가함에 따라 시간 또는 공간 요구량이 지수적으로 증가되어 고차원의 색인 구조로서 효율성을 상실하게 된다. 이러한 문제를 해결하기 위해 최근에 고차원 특징 벡터를 수용할 수 있는 다수의 색인기법들이 연구되었다. 대표적인 예로, TV-트리[2], SS-트리[3], VAM KD-트리, VAM Split R-트리[4], X-트리[5]등이 제안되었다. 내용 기반 멀티미디어 정보 검색에서 주된 이슈는 특징벡터 공간에서 객체간의 유사성에 기반한 검색을 효율적으로 지원하는 것이다. 유사성 검색 질의 중 k-최근접 탐색 질의는 내용 기반 멀티미디어 정보 검색에서 매우 중요한 질의이며, 따라서 이를 효율적으로 처리하기 위한 많은 연구들이 수행되었다. 아울러 대용량의 멀티미디어 데이타베이스가 등장함에 따라 보다 빠른 k-최근접 탐색 질의를 지원하기 위한 근사 k-최근접 탐색 알고리즘에 관한 연구가 이루어졌다. 본 연구에서는 R-트리 유형의 색인 구조에 기반한 연구 방향에 따라, R-트리 유형 기반의 정확 k-최근접 탐색 알고리즘과 기존의 근사 k-최근접 탐색 알고리즘을 살펴본다.

2.1 정확 k-최근접 객체 탐색을 위한 알고리즘

[6]에서는 R-트리를 이용하여 다차원 특징 벡터 공간에서 주어진 질의 객체와 가장 가까운 객체의 점을 찾는 방법을 제시하였다. 이 방법은 트리에서 검색되는 노드의 수를 줄이고자 다차원 검색 공간에서 주어진 질의 점으로부터 객체 또는 검색 공간을 나타내는 최소경계 사각형까지의 최소거리(minimum distance-MINDIST)와 최대탐색거리(minmax distance: MINMAXDIST)를 이용한 가지치기 정책을 이용하고 있다. 탐색 알고리즘에서 가지치기(pruning)란 원하는 객체를 포함하지 않는 노드들을 접근 후보 노드 리스트에서 제거하는 연산이고, 가지치기 전략은 후보 노드 리스트에서 제거할 노드들을 찾기 위한 방법이다.

일반적으로, n($n \geq 1$)-차원 공간상에서 임의의 점 P는 n개의 특징 값들로 $P=(p_1, p_2, \ldots, p_n)$와 같이 표현된다. n차원의 유클리디언 공간상에서 MBR(Minimum Bounding Rectangle) R은 두 꼭지점 S와 T로 정의된다.

$$R=(S, T)$$
$$S=[s_1, s_2, \ldots, s_n], \quad T=[t_1, t_2, \ldots, t_n]$$
단, $s_i \leq t_i, 1 \leq i \leq n$

가지치기 전략에서 MINDIST와 MINMAXDIST를 이용하여 불필요한 탐색 영역을 제거함으로서 탐색 영역을 줄이고자 하였다. MINDIST는 질의 점 P에서 최소경계사각형 R의 가장 가까운 변 또는 점까지 최소거리로서 임의의 점 P와 MBR R간의 거리, MINDIST(P, R)는 다음과 같이 정의된다.

$$MINDIST(P, R) = \sum_{i=0}^{n} |p_i - r_i|^2$$

$$r_i=\begin{cases} s_i \text{ if } p_i < s_i; \\ t_i \text{ if } p_i > t_i; \\ p_i \text{ otherwise.} \end{cases}$$

MINMAXDIST는 질의 점P와 객체를 포함하고 있는 최소경계사각형 R과의 가능한 최대거리들 중에서 최소거리이다. 즉, P로부터 R을 구성하는 모든 축에서 가장 먼 점까지의 거리들 중에서 최소가 되는 거리를 의미한다. 임의의 점 P와 MBR R간의 거리 MINMAXDIST(P, R)은 다음과 같이 정의된다. 그림 1은 질의 점과 최소경계사각형과의 MINDIST와 MINMAXDIST를 나타낸다.

$$MINMAXDIST(P,R)=\min_{1\le k\le n}\left(|p_k-rm_k|^2+\sum_{\substack{1\le i\le n \\ i\ne k}}|p_i-rM_i|^2\right)$$

$$rm_k=\begin{cases} s_k & \text{if } p_k \le \frac{(s_k+t_k)}{2}; \\ t_k & \text{otherwise.} \end{cases}$$

$$rM_i=\begin{cases} s_i & \text{if } p_i \ge \frac{(s_i+t_i)}{2}; \\ t_i & \text{otherwise.} \end{cases}$$

[6]에서 제안한 최근접 탐색 알고리즘을 k-최근접질의 알고리즘에서는 R-트리를 검색하는 동안 방문할 필요가 없는 노드들을 방문대상에서 제외하기 위해 MINDIST와 MINMAXDIST를 조합하여 다음과 같은 전략을 사용하였다.

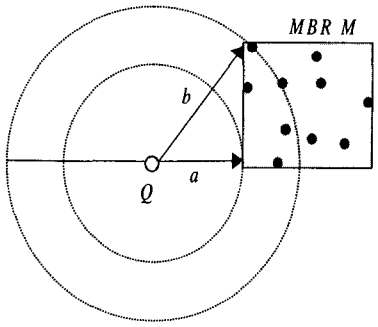
S1. 질의 점 P로부터 k번째 최소경계사각형 R'까지의 MINMAXDIST(P,R')보다 MINDIST(P,R)가 더 큰 값을 갖는 최소경계사각형 R이 존재하면, R은 k-최근 이웃 객체를 포함하지 않기 때문에 R에 해당하는 노드는 검색대상에서 제외한다.

S2. 질의 점 P로부터 객체 O까지의 거리가 k번째 최소경계사각형 R에 대한 MINMAXDIST(P, R)보다 크다면, 객체 O를 k-최근접 객체 대상에서 제외한다.

S3. 질의 점 P로부터 k번째 객체 O까지의 거리보다 큰 MINDIST(P,R)를 갖는 모든 최소경계사각형 R은 검색대상에서 제외한다.

2.2 근사 k-최근접 탐색을 위한 알고리즘

이미지와 비디오 검색과 같은 내용 기반 멀티미디어 정보 검색 시스템에서는 정확한 질의 결과를 얻는 것보다는 빠른 검색이 요구되므로 정확 탐색보다는 근사 탐색이 보다 적절하다. 실제로 데이타베이스가 점점 대용량화되고 멀티미디어 객체로부터 추출된 특징의 수가 증가함에 따라 k-최근접 객체 탐색 질의에 대한 검색 성능이 현저히 저하되는 문제점을 안고 있다. 근사 k-최근접 탐색은 비록 다소의 적중에러는 허용하지만, 보다 빠른 검색을 위한 질의이다[4][7]. [7]은 근사 탐색(approximate search)을 절대적인 해답이 요구되지 않는 환경에서 주어진 근사율(approximate rate) ε 값만큼의 오류를 허용하는 탐색으로 정의하고 있다.



a : MINDIST(Q, M)
b : MINMAXDIST(Q, M)

그림 1 MINDIST와 MINMAXDIST

[8]에서는 근사율 ε에 기반한 근사 k-최근접 탐색 알고리즘을 제안하였다. [8]에서는 질의 점으로부터 최소경계사각형과의 최대 거리 MAXDIST를 이용하여 기존의 정확 탐색을 보완하고 근사율 ε에 기반하여 근사 탐색으로 확장하였다. k-최근접 객체 탐색에서의 근사율 ε 의 개념적 의미는, 정확 탐색으로 찾을 수 있는 k번째 가까운 객체까지의 거리에 $(1+\varepsilon)$을 곱한 거리 이내의 객체들 중에서 탐색 과정 중에 일찍 발견된 객체들을 반환한다는 것이다. k-최근접 객체 탐색에서의 근사율 $\varepsilon(\varepsilon \ge 0.0)$은 다음을 만족한다.

$$\frac{DISTANCE(P,o)}{DISTANCE(P,o_k)} \le (1+\varepsilon)$$

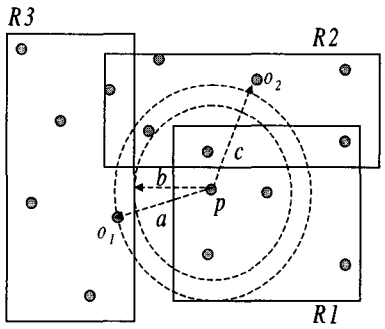


그림 2 정확 탐색과 근사 탐색 알고리즘의 비교 (k=5)

o는 근사 탐색의 결과로 반환된 객체이고, o_k는 정확 탐색에 의한 k번째 최근접 객체이다. 근사 k-최근접 탐색 질의는 ε값에 의해 탐색 영역을 줄임으로서 보다 빠른 k-최근접 탐색 질의를 수행할 수 있다.

그림 2는 k=5인 경우에 대하여 정확 탐색과 근사 탐색의 차이점을 보여주는 예이다. 정확 탐색의 경우, 객체 o_1이 k번째 가까운 객체이므로 질의점 p로부터 o_1까지의 거리인 a를 반지름으로 하고, 질의점 p를 중심으로 하는 원과 겹치는 모든 MBR, 즉, R1, R2, R3를 접근하게 된다. 이에 반하여, 근사 탐색의 경우, MINDIST 거리가 작은 순서로 MBR R1과 R2를 먼저 접근하여 현재까지 발견된 k번째 객체인 o_2까지의 거리 c와 MINDIST(p, R3)인 b를 비교하여 $c/b \leq (1+\varepsilon)$를 만족하면 MBR R3를 접근하지 않고 객체 o_1 대신 o_2를 k 최근접 객체로서 반환하게 된다.

3. 근사 k-최근접 탐색 알고리즘

기존의 k-최근접 탐색 알고리즘은 고차원 데이타에 적용하였을 경우, 검색 성능이 현저히 감소하는 문제점을 갖는다. 본 장에서는 고차원 특징 데이타를 다루는 대용량 멀티미디어 내용 기반 검색 시스템과 같은 응용에서 보다 빠른 k-최근접 질의 처리을 지원하기 위한 효율적인 근사 k-최근접 탐색 알고리즘을 제안한다.

3.1 고차원 특징 벡터의 특성

멀티미디어 객체로부터 추출된 특징 벡터는 매우 고차원의 특성을 지니고 있다. 이러한 고차원 특징 벡터에 대하여 기존의 정확 k-최근접 탐색 알고리즘은 많은 탐색 비용을 요구하게 된다. 이는 차원이 증가함에 따라 정확 탐색 과정에서 가지치기 되는 양이 급격히 줄어들면서 정확 k-최근접 탐색 알고리즘의 검색 성능은 현저히 저하하게 된다. 본 절에서는 기존의 정확 k-최근접 탐색 알고리즘의 가지치기 전략의 문제점을 야기하는 고차원 특징 벡터의 특성을 제시한다. R-트리 유형의 고차원 색인 구조에 기반한 정확 k-최근접 탐색 알고리즘에서의 가지치기 전략으로서 사용되는 *MINDIST*와 *MINMAXDIST* 거리의 분포는 차원이 증가함에 따라 다음과 같은 특성을 갖는다.

- 차원이 증가함에 따라 MINMAXDIST와 OBJECTDIST는 크게 증가한다.
- 균일한 분포(uniformed)의 데이타에 대해 MINMAXDIST와 MINDIST의 거리 차이와 아울러 OBJECTDIST와 MINDIST의 거리 차이는 거의 평행한 특성을 지닌다.

3.1.1 고차원에서의 MINMAXDIST, OBJECTDIST 거리

기존의 R-트리 유형의 색인 구조에 기반한 k-최근접 탐색 알고리즘에서 제시하는 가지치기 전략은 다음과 같이 크게 두 단계로 구분할 수 있다.

i) MINMAXDIST(P, M_k)에 의한 가지치기 단계

ii) OBJECTDIST(P, o_k)에 의한 가지치기 단계

이러한 두 단계의 가지치기를 이용하여 탐색 알고리즘은 실제로 고차원 데이타에 적용하였을 경우 검색 성능이 크게 개선되지 않는 문제점이 발생한다. 가지치기 전략 첫 번째 단계에서 가지치기 전략으로서 사용되는 *MINDIST*와 *MINMAXDIST* 거리는 차원이 증가함에 따라 *MINDIST*와 *MINMAXDIST*의 차이가 크게 증가하는 성질을 보인다. *MINDIST*는 차원이 증가함에 따라 급격히 감소하며, 반면 *MINMAXDIST*는 증가한다. 따라서 고차원 데이타에 대해서 가지치기 전략 첫 번째 단계를 적용하였을 경우, k 번째 작은 *MINMAXDIST*보다 큰 *MINDIST*를 갖는 *MBR*이 존재하지 않게 되어 가지치기가 거의 이루어지지 않는 문제가 발생한다. 기존 근사 최근접 탐색 알고리즘에서 근사율 ε을 이용한 가지치기에서, 근사율 ε(단, $0 \leq \varepsilon < 1$)은 정확 탐색에서 k번째 최근접 객체의 거리에 대한 근사 탐색에서의 k번째 최근접 객체 거리 차의 비율을 의미한다. 따라서 근사율 ε에 따라 비록 정확한 k개의 최근접 객체를 검색할 수는 없지만 매우 유사한 객체들을 검색할 수 있다. 하지만 차원이 증가함에 따라 객체의 거리 또한 증가하고, 그에 따라 가지치기 거리에 해당하는 k번째 객체의 거리를 근사율 ε에 의해 확장하는 비율로서는 효율적인 가지치기를 수행할 수 없게 된다.

그림 3은 균일한 분포를 이루고 있는 20차원 특징 벡터 데이타의 200,000건에 대한 질의 점과 최소경계 사각형과의 ***MINDIST, MINMAXDIST*** 그리고 최소경계사각형 내에 객체 점 중에서 질의 점과 가장 가까운 객체 점과의 거리 분포, 즉 ***OBJECTDIST***를 보이고 있다. 그림에서처럼 MINMIMDIST보다 큰 MINDIST는 존재하지 않으며, OBJECTDIST보다 큰 MINDIST를 갖는 MBR은 거의 없기 때문에 효율적인 k-최근접 질의 탐색을 위한 가지치기를 수행할 수 없게 된다.

3.1.2 MINMAXDIST와 MINDIST, OBJECTDIST와 MINDIST의 거리 차

차원이 증가함에 따라 MINMAXDIST와 MINDIST, 그리고 OBJECTDIST와 MINDIST간의 거리 차이가 증가하지만, 균일한 분포(Uniformed distribution)를 이

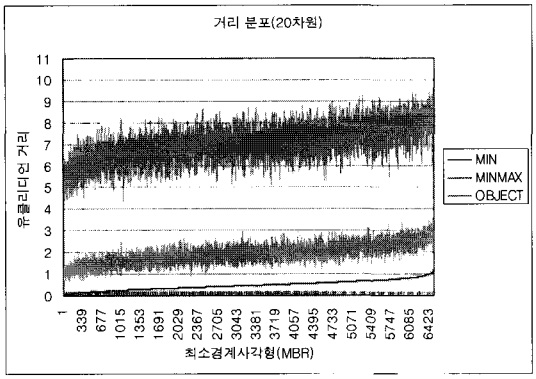
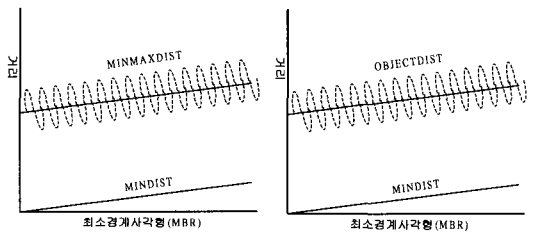


그림 3 거리분포(20차원)

루고 있는 데이타에서는 거리 차에서 규칙성을 얻을 수 있다. 그림 4의 (a)는 MINMAXDIST와 MINDIST의 거리 분포를 도식화한 형태를 보이고 있다. 탐색 후보 영역인 최소경계사각형을 MINDIST에 의해 정렬한 상태이며 MINMAXDIST는 그 특성에 따라 약간의 진폭을 가진 형태로 증가하고 있다. 그림 4의 (a)처럼 진폭을 가진 형태의 MINMAXDIST와 MINDIST가 평행한 형태로 증가한다면, 절하된 MINMAXDIST에 의해 MINDIST를 가지치기할 수 있게 된다. 그림 4의 (b)는 OBJECTDIST와 MINDIST의 거리 분포를 도식화한 형태를 보이고 있다. 탐색 후보 영역인 최소경계사각형을 MINDIST에 의해 정렬한 상태이며 OBJECTDIST도 MINMAXDIST와 마찬가지로 그 특성에 따라 약간의 진폭을 가진 형태로 증가하고 있다. 균일한 분포의 데이타에 대해 OBJECTDIST와 MINDIST가 평행한 형태를 취하고 있으며, 이에 따라 절하된 OBJECTDIST를 이용하여 MINDIST를 가지치기할 수 있게 된다.

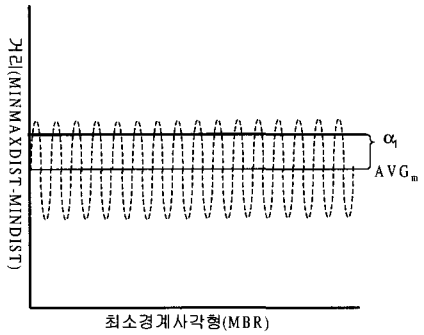


(a) (b)

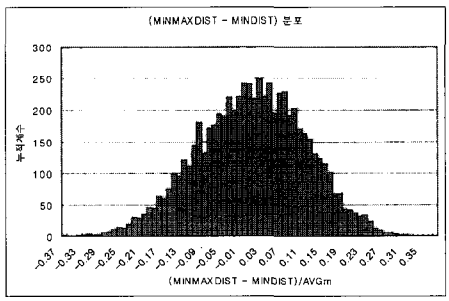
그림 4 MINMAXDIST, OBJECTDIST의 MINDIST와의 거리 차

3.2 근사 k-최근접 탐색의 가지치기 전략

제안하는 근사 k-최근접 데이타 탐색 알고리즘에서는 기존의 정확 k-최근접 질의 탐색 알고리즘에서의 가지치기 전략을 보완하기 위해 고차원 데이타에 대해 각 단계에서 보다 효율적인 가지치기가 이루어지도록 고차원에서의 거리 특성을 이용한 새로운 가지치기 거리(metric)를 정의한다.



(a) MINMAXDIST-MINDIST



(b) α_1/AVG_m 에 따른 (MINMAXDIST-MINDIST)의 분포

그림 5 MINMAXDIST에 의한 근사 가지치기 거리 추정

그림 4의 (a)는 MINMAXDIST와 MINDIST의 거리 차이 분포를 도식화한 형태를 보이고 있다. AVG_m은 MINMAXDIST와 MINDIST의 거리 차이값들의 평균값을 의미한다. 따라서 α_1(단, -최대진폭값$\leq \alpha_1 \leq$최대진폭값)가 최대 진폭값을 갖을 경우 $AVG_m + \alpha_1$은 모든 MINMAXDIST를 포함할 수 있는 거리가 된다. 그림 5의 (b)는 α_1/AVG_m에 따른 (MINMAXDIST-MINDIST)의 분포를 보이고 있다. α_1에 따라 적절한 비율의 MINMAXDIST를 확보할 수 있게 된다. 이 적절한 비율의 MINMAXDIST는 검색 결과의 정확율을 보장할 수 있으며 보다 많은 노드를 제거할 수 있게 된다. 정의 1은 기존 가지치기 거리인 k번째 MIN-

MAXDIST의 거리를 절하하며 그 절하 비율 β_1는 α_1에 의해 결정된다.

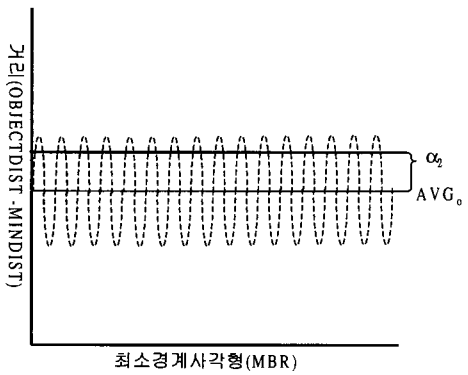
정의 1. 임의의 점 P와 *MBR* R간의 거리(metric) $MINMAXDIST_\beta(P, R)$은 다음과 같이 정의한다.

$$MINMAXDIST_\beta(P,R) = MINDIST(P,R_k) + (1-\beta_1)(MINMAXDIST(P,R) - MINDIST(P,R_k))$$

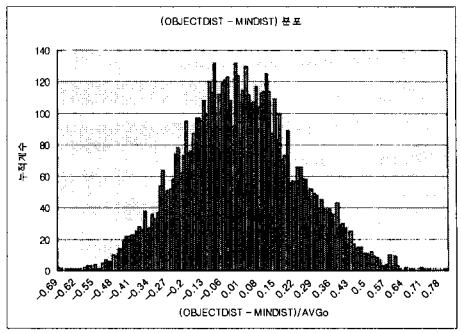
$$\beta_1 = 1 - \frac{\alpha_1}{AVG_m} \quad (\text{단, } 0 \le \beta_1 < 1)$$

$MINMAXDIST_\beta$는 고차원에서 *MINMAXDIST*와 *MINDIST*의 차이가 커서 첫 번째 단계에서 가지치기가 거의 이루어지지 않는 문제점을 보완하기 위해 k번째 *MINMAXDIST*를 절하비율 β_1에 따라 절하한 거리이다. α_1/AVG_m는 검색 결과의 질을 보장할 수 있는 값이며 이는 절하비율 β_1을 결정한다. $MINDIST(P, R_k)$는 *MBR*들을 *MINDIST*에 의해 정렬한 후의 k번째 *MBR* R의 *MINDIST*를 나타낸다.

또한, 고차원에서의 OBJECTDIST와 MINDIST의 특성을 이용하여 가지치기 거리인 질의점과 k번째 객체와의 거리 OBJECTDIST를 가지치기가 보다 많이 이루어지도록 절하할 수 있다. 정의 2는 기존 가지치기 거리인 k번째 OBJECTDIST를 절하하며 그 절하 비율 β_2는 α_2에 의해 결정된다. 그림 6의 (a)는 OBJECTDIST와 MINDIST의 거리 차이를 도식화한 형태를 보이고 있으며 그림 6의 (b)는 데이타 분포에 의하여 적절한 검색 결과를 얻을 수 있는 α_2를 구할 수 있다. 정의 2는 기존 가지치기 거리인 k번째 OBJECTDIST의 거리를 절하하며 그 절하 비율 β_2는 α_2에 의해 결정된다.



(a) OBJECTDIST-MINDIST의 차



(b) α_1/AVG_m에 따른(OBJECTDIST-MINDIST)의 분포

그림 6 OBJECTDIST에 의한 근사 가지치기 거리 추정

정의 2. 임의의 점 P와 현재까지 발견된 객체 중에서 k번째로 가까운 객체와의 거리(metric) $OBJECTDIST_\beta(P, o_k)$은 다음과 같이 정의한다.

$$OBJECTDIST_\beta(P,o_k) = MINDIST(P,R_k) + (1-\beta_2)(OBJECTDIST(P,o_k) - MINDIST(P,R_k))$$

$$\beta_2 = 1 - \frac{\alpha_2}{AVG_o} \quad (\text{단, } 0 \le \beta_2 < 1)$$

고차원의 데이타에서 객체간의 OBJECTDIST를 β_2 값에 의해 절하함으로써 탐색 영역을 줄일 수 있다. α_2/AVG_o는 검색 결과의 질을 보장할 수 있는 값이며, 이는 절하비율 β_2를 결정한다. 정의 1, 2에서 정의한 새로운 거리(Metrics)에 기반하는 본 논문에서 제안하는 근사 k-최근접 데이타 탐색 알고리즘의 새로운 가지치기 전략(PS ; pruning strategy)은 다음과 같다.

PS1. 질의 점 P로부터 k번째로 작은 *MINMAXDIST*를 갖는 *MBR* R'에 대한 $MINMAXDIST_\beta(P, R')$거리보다 큰 $MINDIST(P, R)$거리를 갖는 *MBR* R은 제거한다.

PS2. 질의 점 P로부터 현재까지 발견된 k 번째로 가까운 객체 o_k에 대한 $DISTANCE_\beta(P, o_k)$보다 큰 $MINDIST(P, R)$거리를 갖는 *MBR* R은 제거한다.

그림 7은 k=5인 경우에 대하여 PS1의 $MINMAXDIST_\beta$에 의한 가지치기 예이다. D1은 질의 점으로부터 k번째 가까운 *MINMAXDIST*이고 D2는 질의 점으로부터 k번째 가까운 *MINDIST*이다. P는 정의 1에 의해, D1을 가지치기 거리 절하율 β에 의해 절하된 거리로 실제 가지치기를 위한 거리이다. P에 의한 가지치기는 정확 k-최근접 탐색에서 D1에 의해 가지치기를 했을 경우 가지치기가

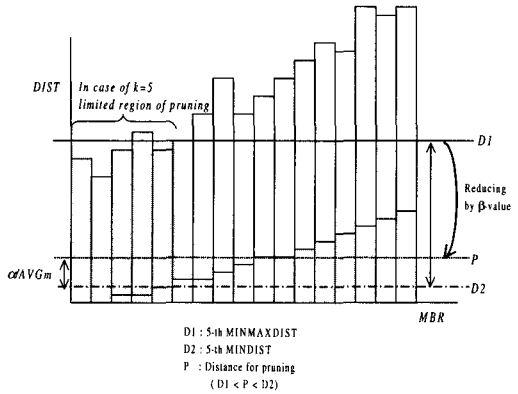


그림 7 $MINMAXDIST_\beta$에 의한 가지치기의 예

전혀 이루어지지 않는 문제를 해결할 수 있으며, 절하 비율에 따른 적절한 검색의 질을 유지할 수 있다.

3.3 근사 k-최근접 탐색 알고리즘

R*-트리에서의 [6] 알고리즘은 트리를 깊이우선(depth-firth fashion)으로 탐색한다. 따라서 자식노드로 확장하기 전에 노드에 있는 엔트리는 정렬이 되며, 형제 노드에 있는 다른 엔트리가 k-최근접 데이타를 포함하고 있더라도 가지치기는 현재 노드에 있는 엔트리에 대해서만 이루어진다. 따라서 실제적으로 필요한 탐색 영역보다 넓은 영역을 탐색하는 문제점을 갖는다[9]. 본 논문에서 제안하는 근사 k-최근접 탐색 알고리즘은 큐(Queue) 구조로 표현되는 전역 분기 리스트(BranchList)를 이용하여 탐색해야 할 노드를 관리한다. 따라서 탐색 영역을 최소로 줄임으로서 최적의 근사 k-최근접 탐색을 수행할 수 있다. 근사 k-최근접 데이타 탐색 알고리즘은 다음과 같다.

Input:
Q : 질의 점, $\{q_i \mid q_i \in Q, 0 \le i < n,$ n은 자연수$\}$
K : 검색할 객체의 개수
Output:
R : 검색 결과 k-최근접 이웃 데이타를 저장할 리스트, $\{o_i \mid o_i \in R, 0 \le i < K\}$
Variable:
BranchList : 방문해야 할 후보 노드 리스트, $\{node[i] \mid node[i] \in BranchList, 0 \le i < BranchSize\}$
node[i] {
mindist; /* Q와의 MINDIST 거리 */
minmaxdist; /* Q와의 MINMAXDIST 거리 */
pointer; /* 자식 노드를 가리키는 포인터 */
}
BranchSize : 현재 BranchList에 있는 후보 노드들의 수
BetaMinmaxdist : $MINMAXDIST_\beta(P, R)$
BetaMaxdist : $MAXDIST_\beta(P, R)$
BetaDist : $DISTANCE_\beta(P, o_k)$
NearestDist : 질의 점 Q와 지금까지 발견된 객체 중 k번째 객체와의 거리, 또는 k번째의 MINMAXDIST
Node : 현재 방문된 노드
NodeSize : 현재 방문된 Node에 있는 MBR 또는 Object의 수

```
Procedure: k_nearest_neighbor_search(Q, k, BranchList) {
    Expand root node and add its all mbr in BranchList;
    Sort all nodes in BranchList by using its mindist;
    while( BranchSize > 0 ) {
        Make Node as child node of node[0] from
        BranchList;
        if( Node.type == LEAF ) { /* 리프 노드인 경우 */
            checkflag = FALSE;
            for( i=0; i<Node; i++ ) {
                dist = ObjectDIST(Q, Node.object_i);
                if( dist < NearestDist ) {
                    Insert Node.object_i to R;
                    Calculate NearestDist from R;
                    PruningBranchList(BranchList);
                }
            }
        }
        else { /* 중간 노드인 경우 */
            Add alll Node's mbr in BranchList;
            PruningBranchList(BranchList);
        }
        Get BranshSize from BranchList;
    }
} /* k_nearest_neighbor_search */
```

k-최근접 탐색은 색인 트리의 root 노드에서 시작한다. 중간 노드에서는 자식 노드의 MBR과의 거리 정보를 이용하여 BranchList에 삽입하게 된다. 중간 노드에서는 k번째 가까운 MINMAXDIST를 구하고 PruningBranList에서 불필요한 후보 노드들을 제거하게 된다. 리프 노드인 경우 리프 노드에 있는 객체와의 거리를 계산하여 k개의 객체를 결과 후보로 결과 리스트 R에 저장한다. 결과 리스트 R은 OBJECTDIST에 의해 정렬이 되어 저장된다. 결과 리스트 R에서의 k번째 객체의 OBJECTDIST는 NearestDist의 후보가 된다. 결과 리스트로부터 얻은 NearestDIST에 의해 BranchList에서 불필요한 노드들을 제거하게 된다. 다음은 제안하는 근사 k-최근접 데이타 탐색 알고리즘의 PruningBranchList 서브 함수를 나타낸다.

Procedure PruningBranchList(BranchList) { /* PS1, PS2에 의한 가지치기 */
/* BetaMinmaxdist에 의한 가지치기(PS1) */

```
    Calculate BetaMinmaxdist from BranchList;
    for( i=0; i<BranchSize; i++) {
        Calculate mindist of node[i] from BranchList;
        if( BetaMinmaxdist < mindist; )
            Delete candidate node[i] from BranchList;
    }

    /* BetaDist에 의한 가지치기(PS2) */
    Calculate BetaDist from BranchList and NearestDist;
    for( i=0; i<BranchSize; i++ ) {
        Calculate mindist of node[i] from BranchList;
        if( BetaDist < mindist ) Delete candidate node[i]
from BranchList;
    }
} /* PruningBranchList */
```

PruningBranchList는 가지치기 전략에 의해 탐색영역에서 불필요한 노드를 최대한 제거한다. PruningBranchList는 근사 탐색을 위해 기존의 k번째 MINMAXDIST를 가지치기 절하 비율 β_1에 의해 절하한 BetaMinmaxdist를 이용한다. BetaMinmaxdist보다 큰 MINDIST를 갖는 노드는 탐색영역에서 제거된다. 또한 기존의 k번째 OBJECTDIST를 가지치기 절하 비율 β_2에 의해 절하한 BetaDist를 이용한다. 두 번째 가지치기 전략에 의해 BetaDist보다 큰 MINDIST를 갖는 노드는 탐색영역에서 제거된다.

4. 실험 평가

4.1 구현 및 실험 환경

본 논문에서 제안하는 근사 k-최근접 데이타 탐색 알고리즘은 SUN Enterprise 150상에서 Solaris 2.6 운영체제를 사용하여 X-트리에 기반하여 구현하였다. 성능 평가를 위해 본 논문에서 제안하는 근사 k-최근접 데이타 탐색 알고리즘은 X-트리에서의 정확 k-최근접 질의 처리 알고리즘을 근사 k-최근접 질의 처리 알고리즘으로 확장하여 구현하였다. 성능 평가 실험은 표 1의 실험 인자에 기반하여 이루어진다. 본 성능평가 실험에서는 균일한 분포를 이루는 데이타(uniform data)를 사용한다. 데이타의 각 차원 값은 [0, 1)의 범위를 갖도록 랜덤하게 생성하였다. 실험 데이타는 10차원, 20차원 각각 200,000건을 이루고 있다. 색인 트리의 모든 노드는 4K의 디스크 블록 크기를 갖는다.

표 1 근사 탐색에서의 검색 결과 정확율 보장을 위한 β_1, β_2값

		α 1/AVGm	β 1	α 1/AVGo	β 2
10 차원	80%를 보장하는 경우	0.17	0.83	0.3	0.7
	90%를 보장하는 경우	0.27	0.72	0.47	0.53
20 차원	80%를 보장하는 경우	0.09	0.83	0.18	0.7
	90%를 보장하는 경우	0.14	0.86	0.29	0.71

본 실험에서는 제안한 근사 k-최근접 탐색 알고리즘의 거리 절하 비율 β_1와 β_2 에 따른 정확율을 예측하고 그에 따른 검색 성능을 평가한다. 본 실험에서는 검색 성능 평가 요소로서 검색된 결과의 정확율과 디스크 페이지 I/O 횟수, 검색 시간을 측정한다. 정확율은 정확 k-최근접 데이타 탐색에 의해 검색된 객체를 기준으로 측정한다. X-트리의 노드는 디스크 페이지와 같기 때문에 디스크 페이지 I/O 횟수는 접근된 노드의 수에 기준한다. 본 실험에서 데이타 검색 결과의 정확율을 80%와 90%로 보장하기 위해, 제안한 근사 k-최근접 탐색 알고리즘에 적용할 β_1와 β_2는 표 1과 같다. 10차원의 실험 데이타에서는 80%의 정확율을 보장하기 위해 MINMAXDIST를 절하하는 절하 비율 β_1는 0.83, OBJECTDIST를 절하하는 절하 비율 β_2는 0.7을 갖는다.

4.2 실험 결과 및 분석

본 실험에서는 균일한 분포를 이루고 있는 10차원, 20차원 실험 데이타에 대한 정확 k-최근접 질의 탐색, [8]의 ε에 기반한 근사 k-최근접 질의 탐색, 그리고 본 논문에서 제안한 근사 k-최근접 질의 탐색 알고리즘의 디스크 페이지 I/O 접근 횟수, 검색 시간을 측정한다. 실험 결과는 표 1, 표 2와 같다. [8]의 근사 k-최근접 질의 탐색은 ε=0.1, 0.2일 때를 측정하였다. 각각의 실험 데이타에서 임의의 데이타 100개를 질의 데이타로 사용하며 디스크 페이지 I/O 접근 횟수와 검색 시간의 평균치를 이용한다. 정확 탐색은 차원이 증가함에 따라 디스크 페이지 I/O 횟수와 검색 시간이 급격히 증가함을 보이고 있다. 10차원일 경우 디스크 페이지 I/O 횟수는 219, 검색 시간은 0.33초, 20차원일 경우에는 각각 5373, 3.48초로 정확 탐색은 고차원에서 비효율적임을 알 수 있다. [8]의 근사 탐색은 10차원일 경우 정확 탐색과 거의 유사한 성능을 보인다. 하지만 20차원일 경우 오히려 정확 탐색에 비해 성능이 저하하는 문제점을 갖는다. 이는 ε에 의해 가지치기 되는 양은 적은 데 비해, [8]에서 제안한 가지치기 거리 MAXDIST의 계산량의 증가로 검색 시간이 저하되기 때문이다. 본 논문에서 제안하는 근사 탐색은 정확 탐색에 비해 디스크 페이지 I/O 접근 횟수는 10차원일 경우 약 60~70%, 20차원일 경우 약 90~95%의 감소율을 보인다. 검색 시간은 10차원일 경우 약 40~60%, 20차원일 경우, 약 80%~90%의 감소율을 보인다. 표 3에서처럼 80%를 보

장하는 경우가 90%를 보장하는 근사 탐색보다 빠른 검색을 할 수 있지만, 정확율이 다소 감소함을 보이고 있다. 본 논문에서 제안한 근사 탐색의 정확율은 예상한 정확율보다 더 높은 정확율을 보이고 있다.

표 2 정확 탐색과 근사 탐색의 비교 (D=10, k=10)

10차원	정확 탐색	근사탐색 [Loh97]		제안하는 근사탐색	
				80%를 보장하는 경우	90%를 보장하는 경우
		ε=0.1	ε=0.2	β1=0.17 β2=0.3	β1=0.27 β2=0.47
I/O접근 횟수	218.9	177.7	156.4	56.2	83.7
Search Time(s)	0.33	0.328	0.316	0.117	0.188
정확율(%)	100%	100%	100%	100%	100%

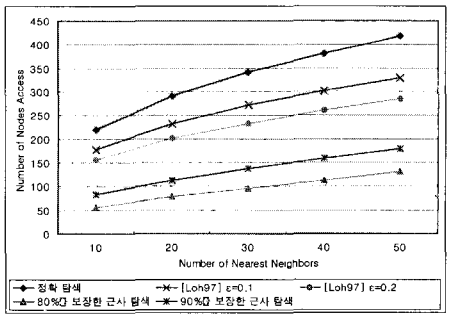
표 3 정확 탐색과 근사 탐색의 비교 (D=20, k=10)

20차원	정확 탐색	근사탐색 [Loh97]		제안하는 근사탐색	
				80%를 보장하는 경우	90%를 보장하는 경우
		ε=0.01	ε=0.02	β1=0.09 β2=0.18	β1=0.14 β2=0.29
I/O접근횟수	5372.7	4756.2	4258.4	359.6	778.6
Search Time(s)	3.48	4.145	3.602	0.399	0.709
정확율(%)	100%	100%	100%	91%	100%

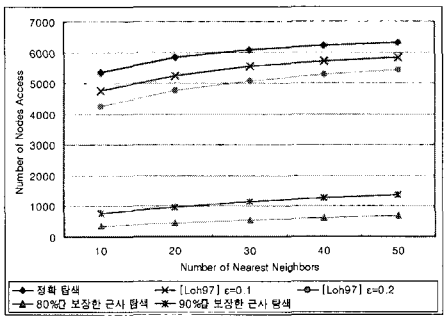
다음은 k값에 따른 정확 탐색, 근사율 ε에 기반한 근사 탐색, 그리고 본 논문에서 제안한 근사 탐색의 성능을 비교한다. 그림 8, 4.2는 실험 결과를 보인다. 근사율 ε에 기반한 근사 탐색은 정확 탐색에 비하여 디스크 페이지 I/O 접근 횟수는 약간 감소하고, 검색 시간은 거의 비슷한 결과를 갖는다. 20차원일 경우, 근사율 ε에 의하여 가지치기되는 양은 적으나, 가지치기 거리 MAXDIST의 계산량으로 검색 시간은 정확 탐색에 비하여 증가함을 보이고 있다. 본 논문에서 제안하는 근사 탐색은 20차원의 데이타에 대해 보다 많은 성능 개선이 이루어지고 있음을 보인다. 디스크 페이지 I/O 접근 횟수는 정확 탐색에 비하여 약 80%의 감소율을 보이며, 검색 시간은 약 70~75%의 감소율을 보인다. 본 논문에서 제안하는 근사 탐색은 절하된 가지치기 거리를 이용하여 보다 많은 페이지 노드를 탐색 영역에서 제거함으로써 보다 빠른 k-최근접 데이타 탐색을 수행한다.

5. 결 론

멀티미디어 내용 기반 검색에서 k-최근접 탐색 질의는 유사성에 기반한 검색 질의로서 매우 중요하며, 내용 기반 멀티미디어 정보 검색 시스템은 멀티미디어 객체로부터 추출된 특징 벡터가 매우 고차원이기 때문에 사용자에게 보다 효율적인 k-최근접 탐색 질의를 제공해

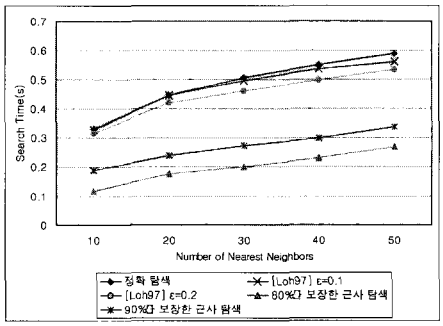


(a) D=10

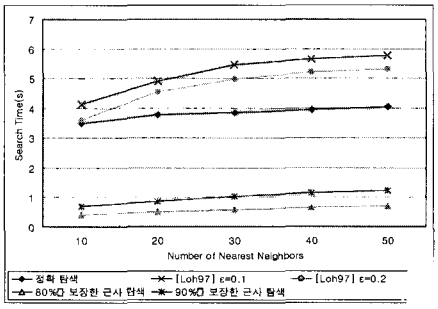


(b) D=20

그림 8 k값에 따른 디스크 페이지 I/O 접근 횟수



(a) D=10



(b) D=20

그림 9 k값에 따른 검색 시간

야 한다. [6]에 기반한 정확 k-최근접 탐색 질의는 차원이 증가함에 따라 검색 성능이 현저히 저하한다. 실험 결과 40차원 이상의 데이타에 대해서는 전체 탐색 영역을 검색하는 비효율성을 보였다. [8]에서 제안한 근사 k-최근접 탐색은 근사율 ε을 사용하여 검색 결과의 질을 보장할 수 있는 반면, 차원이 증가함에 따라 고차원 데이타에 대해서는 정확 k-최근접 탐색과 마찬가지로 매우 비효율적이었다.

본 논문에서는 고차원 데이타에 대해서 가지치기 양을 늘리기 위한 새로운 가지치기 거리 측도를 제안하였다. 제안한 거리 측도는 검색 질을 보장하는 거리 절하 비율 β에 의해 기존의 가지치기 거리 측도를 절하한다. 이는 고차원 데이타에 대해 가지치기를 할 수 없었던 기존의 가지치기 단점을 해결하였다. 또한, 이를 이용하여 고차원의 데이타에 대해 보다 효율적인 근사 k-최근접 탐색 질의를 수행하는 가지치기 전략을 제안하고 그 성능을 평가하였다. 제안한 근사 k-최근접 탐색 알고리즘은 실험을 통하여 기존의 정확 k-최근접 탐색 알고리즘과 비교하여 80% 이상의 정확율을 유지하면서 평균적으로 70~80%의 페이지 I/O 접근 횟수 감소율을 얻을 수 있었다. 제안한 근사 k-최근접 데이타 알고리즘은 약간의 적중에러가 발생하지만 많은 페이지 I/O 접근 횟수를 줄임으로써 사용자에게 보다 빠른 k-최근접 탐색 질의를 수행한다.

향후 연구 과제로는 제안하는 근사 탐색의 실용성을 검증하기 위해 실제 멀티미디어 데이타로부터 추출된 수 십만개의 특징 벡터 데이타를 통한 성능평가를 수행하는 것이다.

참 고 문 헌

[1] Beckmann N., Kriegel H.-P, Schneider R., Seeger B., 'The R*-tree: An Efficient and Robust Access Method for Points and Rectangles,' Proc. ACM SIGMOD Int. Conf. on Management of Data, Atlantic City, NJ, 1990, pp. 322-331.

[2] K.I. Lin, H. Jagadish, C. Faloutsos, 'The TV-tree: An Index Structure for High Dimensional Data,' VLDB Journal, Vol. 3, pp. 517-542, 1994.

[3] D. A. White and R. Jain, 'Similarity Indexing with the SS-tree,' In Proc. Intl. Conf. on Data, Engineering, pp. 516-523, 1996.

[4] D.A. White and R. Jain, 'Similarity Indexing : Algorithms and Performance,' In Proc. of the SPIE : Storage and Retrieval for Image and Video Databases IV, Vol. 2670, pp. 62-75, 1996.

[5] Berchtold S., Keim D., Kriegel H.-P., 'The X-tree: An Index Structure for High-Dimensional Data,' 22nd Conf. on Very Large Databases, 1996, Bombay, India.

[6] Roussopoulos N., Kelley S., Vincent F., 'Nearest Neighbor Queries,' Proc. ACM SIGMOD Int. Conf. on Management of Data, 1995, pp. 71-79.

[7] Arya, S. et al., 'An Optimal Algorithm for Approximate Nearest Neighbor Searching," In Proc. ACM-SIAM Symposium on Discrete Algorithms, pp. 573-582, 1994.

[8] 노웅기, 황규영, '멀티미디어 내용기반 검색을 위한 빠른 k-최근접 객체 탐색 알고리즘', '97 한국정보과학회 가을 학술발표논문집, Vol. 24, No. 2, pp. 167-170, 1997.

[9] S. Berchtold, C. Bohm, D. Keim, and H. -P. Kriegel, 'A Cost Model for Nearest Neighbor Search in High-Dimensional Data Space,' In Proceesings of ACM PODS Symposium on Principles of Database System, 1997.

[9] Myron Flickner and et. al., 'Query by Image and Video Content: The QBIC System,' IEEE Computer, 28(9), 1995.

송 광 택

1997년 원광대학교 컴퓨터공학과(공학사). 1999년 전북대학교 컴퓨터공학과(공학석사). 1997년 ~ 현재 전북대학교 컴퓨터공학과 박사과정. 관심분야는 멀티미디어 데이타베이스, 멀티미디어 정보 검색, 고차원 색인 기법 등

장 재 우

1984년 서울대학교 전자계산기공학과(공학사). 1986년 한국과학기술원 전산학과(공학석사). 1991년 한국과학기술원 전산학과(공학박사). 1996년 ~ 1997년 Univ. of Minnesota, Visiting Scholar. 1991년 ~ 현재 전북대학교 컴퓨터공학과 부교수. 관심분야는 멀티미디어 데이타베이스, 멀티미디어 정보검색, 하부저장구조