이동체의 과거, 현재 및 미래 위치 질의 처리를 위한 통합 색인의 설계 및 구현

Design and Implementation of Unified Index for Query Processing Past, Current and Future Positions of Moving Objects

반재훈*, 전희철**, 안성우***, 김진덕****, 홍봉희*
Chae-Hoon Ban, Hee-chul Jeon, Sung-Woo Ahn, Jin-Deog Kim, Bong-Hee Hong

요 약 최근 이동 통신과 GPS 기술의 발달로 위치 기반 서비스에 대한 요구 및 관련된 연구가 활발히 진행되고 있다. 이동체 색인에 관한 기존 연구는 시간 도메인에 따라 과거 궤적 색인과 현재 및 미래 위치 색인으로 분류된다. 그러나 실세계 응용에서는 과거 궤적뿐만 아니라 현재 및 미래 위치 검색을 모두 요구하므로 모든 시간 도메인에 대한 질의를 지원하는 통합 색인을 개발해야 한다.

이 논문에서는 이동체의 과거 궤적을 표현하는 3차원 공간상의 선분과 이동체의 현재 및 미래 위치를 표현하는 시간에 대한 선형 함수를 하나의 색인에 구성함으로써 이동체의 과거, 현재, 미래의 위치 데이터가 통합된 새로운 색인인 PCR-tree(Past, Current R-tree)을 제안한다. PCR-tree 는 노드 내에 포함된 과거, 현재, 미래 위치 데이터에 대한 새로운 경계 영역을 가지며 색인의 모든 엔트리에 대한 단일 인터페이스를 제공한다. 그리고, 제안된 색인과 색인 실험 도구를 구현하여 모든 시간 도메인에 대한 질의 처리가 가능함을 보인다.

Abstract Recently, application area on the Location Based System(LBS) is increasing because of development of mobile-communication and GPS technique. Previous studies on the index of moving objects are classified as either index for past trajectories or current/future positions. It is necessary to develop a unified index because many applications need to process queries about both past trajectories and current/future positions at the same time.

In this paper, the past trajectories of moving objects are represented as line segments and the current and future positions are represented as the function of time. We propose a new index called PCR-tree(Past, Current R-tree) for unification of past, current and future positions. Nodes of the index have bounding boxes that enclose all position data and entries in the nodes are accessed with only one interface. We implement the proposed index and show a feasibility of processing the queries about temporal-spatial domain with the query tool which we develop.

주요어 : 이동체, 이동체 데이터베이스, 이동체 색인, 과거 질의, 현재 및 미래 질의.
KeyWords : moving object, moving object database, moving object index, past query, current/future query

* 경남정보대학 인터넷응용계열 chban@kit.ac.kr
** 삼성전자 정보통신연구소 차세대 단말팀 S/W Lab. hcjeon@pusan.ac.kr
*** 부산대학교 컴퓨터공학과 {swan,bhhong}@pusan.ac.kr
**** 동의대학교 컴퓨터공학과 jdk@deu.ac.kr

1. 서 론

무선 통신 서비스의 핵심 응용인 위치 기반 서비스는 이용자의 현재 위치를 기반으로 주변 정보를 입수하거나 타인의 위치를 파악해 제공하는 등의 다양한 부가 서비스를 일컫는다. 위치 기반 서비스의 핵심 기술은 연속적으로 이동하는 이동체를 고려한 위치 기반 질의의 효율적인 처리로서, 이동체의 위치 정보를 관리하고 빠른 검색을 제공하기 위해 이동체 데이터베이스를 이용한다.

위치 기반 서비스를 위한 이동체 데이터베이스의 질의는 영역 질의가 대표적이며 질의 시간 범위에 따라 과거 영역질의 또는 현재/미래 영역질의로 분류된다. 이러한 질의 처리를 위한 기존 연구는 첫째, 과거 궤적(trajectory) 검색을 위해 이동체의 위치를 선분(line segment)으로 모델링 한 3DR-tree[1]와 TB-tree[2], STR-tree[2]등이 있으며 둘째, 위치 데이터를 점(Point)로 모델링한 R-tree[7]와 해쉬 기반 색인이 있다. 마지막으로 현재 및 가까운 미래위치 검색을 위해 시간에 대한 선형 함수를 사용하는 TPR-tree[3]와 PMRQuad-tree[8] 등이 있다.

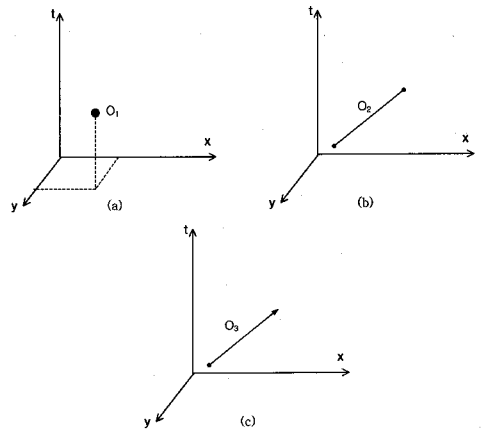
실 세계 응용에서는 과거 궤적 뿐만 아니라 현재 및 미래 위치 검색을 모두 요구하므로 시간 도메인에 대한 질의를 제공하기 위해서는 두 개 이상의 색인을 별도로 유지해야 한다. 예를 들어 "5분 전부터 5분 후까지 서울역 광장 앞을 통과하는 차량을 검색하라"와 같은 질의를 처리하기 위해서는 3DR-tree와 같은 과거 궤적을 검색하는 색인과 TPR-tree와 같은 현재/미래 위치를 검색하는 색인을 구축해야 한다. 그러나 다수개의 색인을 유지하면 부수적인 비용을 필요로 할 뿐만 아니라 과거와 현재/미래 데이터에 대한 공간 근접성을 지원하지 못한다. 따라서 모든 시간 도메인에 대한 질의를 지원하는 통합 색인을 개발해야 한다.

이 논문에서는 이동체의 과거 궤적을 선분으로 저장하고 이동체의 현재 및 가까운 미래 위치를 선형 함수로 저장하여 모든 시간 도메인에 대한 질의를 지원하는 색인인 PCR-tree(Past Current R-tree)을 제시한다. 색인의 노드는 포함된 과거, 현재, 미래 위치 데이터에 대한 새로운 경계 영역을 가지며 색인의 모든 엔트리에 대한 단일 인터페이스를 제공한다. 그리고, 제안된 색인과 색인 실험 도구를 구현하여 모든 시간 도메인에 대한 질의 처리가 가능함을 보인다

이 논문의 구성은 다음과 같다. 2장에서 관련 연구를 살펴보고 3장에서는 연구 동기를 설명한다. 4장에서는 과거 및 현재/미래 질의 수행이 가능하도록 통합된 색인의 구조를 제시한다. 5장에서는 통합된 색인의 삽입 및 탐색 알고리즘을 기술하며 6장에서는 제안된 색인을 구현하고 질의 수행 과정을 설명한다. 마지막으로 7장에서 결론 및 향후 연구를 기술한다.

2. 관련 연구

이동체란 시간이 지남에 따라 연속적으로 그 위치가 갱신되는 객체를 의미하며 이동체 데이터란 이동체로부터 유/무선통신을 통해 수신된 위치 데이터를 의미한다. 수신된 데이터는 <그림 1> 과 같이 응용에 따라 점, 선분, 시간에 대한 함수 등으로 모델링 된다.



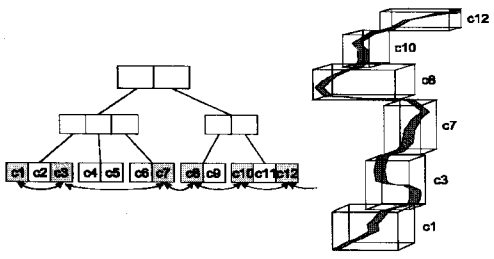
<그림 1> 이동체 데이터 모델링

<그림 1>의 (a)는 이동체의 위치 데이터가 2차원 공간과 시간 축을 합한 3차원 공간상의 한 점으로 모델링 된 것을 보여준다. (b)는 특정 시간간격 동안의 이동체의 위치정보를 3차원 공간상의 두 점을 연결한 선분으로 표현한 것이며, (c)는 특정시간에서의 X-Y 공간상의 위치와 방향 및 속도를 나타내는 벡터로 표

현하여 가까운 미래 위치에 대한 예측이 가능한 이동체의 데이터 모델이다.

이동체의 위치는 시간이 지남에 따라 연속적으로 변화하기 때문에 이러한 변화를 모두 데이터베이스에 유지하는 것은 불가능하다. 따라서 일반적인 과거궤적 검색을 위한 색인에는 특정한 시간 간격에 따른 이동체의 위치 보고를 <그림 1>의 (b)와 같이 3차원 공간상의 두 점을 연결한 선분의 형태로 저장하고 그 시간 간격 사이의 위치는 선분에 대해 보간법(interpolation)을 적용하여 대략적인 위치를 추출한다. 이러한 색인의 대표적인 예로서 STR-Tree[2], TB-Tree[2]와 3DR-Tree[1]가 있다.

STR-Tree, TB-Tree는 이동체의 궤적을 선분으로 표현한 색인으로 R-Tree의 변형이다. TB-Tree는 <그림 2>와 같이 하나의 단말 노드에 동일한 이동체의 궤적을 시간 순서대로 저장하고 동일한 이동체의 궤적을 저장한 노드를 별도의 링크로 연결함으로써 궤적 추출 성능이 뛰어난 구조이다. 그러나 이동체 데이터의 공간 근접성(spatial closeness)을 전혀 고려하지 않았기 때문에 단말 노드의 영역이 커지고 중첩(overlap)이 심해져 영역질의 성능이 크게 떨어지는 단점이 있다. STR-Tree는 부분 궤적 보호(partial trajectory preservation)를 통해 TB-Tree와 R-Tree의 특성을 혼합한 색인으로서 궤적 보존에 대한 매개인자(preservation parameter)를 이용해 공간 근접성을 어느 정도 고려한 구조이다. STR-Tree는 R-Tree에 비해 영역질의 성능이 떨어지지만 궤적 추출 연산 효율이 좋고 TB-Tree에 비해 영역질의 성능은 우수하지만 궤적 추출 성능이 나쁘다.



<그림 2> TB-Tree의 구조

3DR-Tree[1]는 R-Tree에 시간 축의 개념을 추가하여 3차원으로 확장한 구조이다. R-Tree와 같이 공간 근접성을 최대한 고려하여 높은 영역질의 성능을 나타낸다. 그러나 이동체 데이터의 시간 도메인에 대한 고려가 없기 때문에 공간활용도(space utilization)가 크게 떨어지는 단점이 있다.

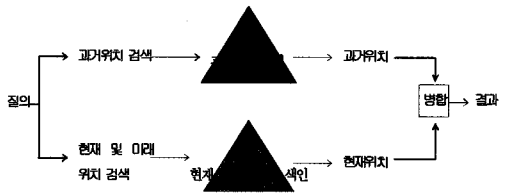
TPR-tree[3]는 <그림 1>(C)의 이동체 모델을 사용함으로써 이동체가 위치를 보고한 시점 이후의 대략적인 위치(특정 임계 값 범위를 초과하지 않는)를 검색할 수 있도록 하였다. 이 경우 이동체의 위치보고는 현재 시간과 공간 좌표, 그리고 이동체의 이동속성을 포함하게 되고 이동속성을 사용하여 예측된 위치와 실제 위치간에 오차가 임계 값을 초과하는 순간에 보고가 이루어진다. 따라서 충분히 넓은 임계 값을 설정할 경우 색인에 발생하는 삽입 연산의 횟수가 크게 감소한다는 특징이 있다.

3. 연구동기

이동체의 과거, 현재, 미래위치는 실 세계에서 다양한 목적과 응용에 사용되고 있다. 이동체의 과거 위치 데이터는 이동체의 이동경로를 감시하여 운송업체로 하여금 보다 효율적으로 차량을 활용할 수 있게 한다. 또한 과거데이터를 분석하여 시가지 지역의 교통흐름 패턴을 추출하여 도로건설 계획 작성에 도움이 되기도 한다. 이동체의 현재위치는 차량이나 사람의 위치 조회나 도로상태 등에 사용된다. 여기서 말하는 현재 위치는 이동체가 가장 최근에 보고한 위치를 의미하고 실제 이동체의 위치와 오차가 발생한다. 이동체의 예측된 미래 위치는 이동체의 위치보고 시점 이후의 실제 현재 시간의 위치를 예측함으로써 오차율을 줄이는데 사용될 수 있고 항공기나 선박 등의 가까운 미래 위치로써 충돌 방지시스템 등에 사용될 수도 있다.

이러한 다양한 응용은 시간 도메인의 관점에 따라 적합한 색인을 사용하게 된다. 그러나 두 가지 이상의 응용을 동시에 서비스하는 경우 모든 시간도메인의 데이터를 다루는 기존 연구가 없기 때문에 응용 별로 다수개의 색인을 유지할 수 밖에 없다. 그러나 다수개의 색인을 유지하는 방법은 부수적인 비용을 필요로

할 뿐만 아니라 한번의 위치보고가 발생하면 다수개의 색인에 대해 모두 삽입연산을 수행해야 한다는 단점이 있으며 질의시 그림 3과 같이 복잡한 과정을 거쳐야만 한다. 또한 이동체가 최근 보고한 위치와 새로 보고하는 위치간에는 공간 근접성이 매우 높은데 다수 개의 색인을 사용하게 되면 물리적, 논리적으로 완전히 별개인 각각의 색인에 저장되므로 이러한 공간 근접성이 완전히 무시된다는 단점이 있다.



<그림 3> 다수 개의 색인 사용시의 질의처리 과정

이 논문에서는 이러한 단점을 극복하기 위해 이동체의 과거와 현재 및 가까운 미래위치 데이터에 대한 검색이 가능하도록 이동체의 과거와 현재 및 미래위치 정보를 하나의 색인으로 구성한다. 또한 제안된 색인을 구현하여 모든 시간 도메인에 대한 질의처리가 가능한 것을 보인다.

4. 통합 색인 구조 PCR-tree

이 장에서는 제안하는 통합 색인인 PCR-tree의 구조에 대하여 용어정의, 과거 데이터와 현재 및 미래를 나타내는 선형 함수에 대한 논리적인 결합방법과 하나의 노드에 저장하는 방법, 그리고 단일화된 인터페이스에 대해 소개한다.

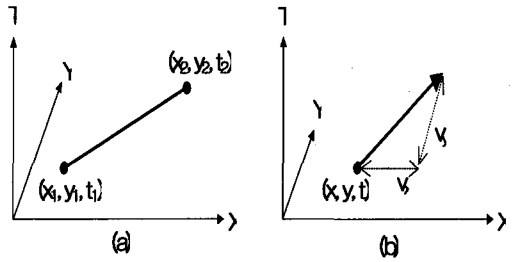
4.1 용어정의

4.1.1 이동체의 최근 보고시간(T_{uc})

T_{uc}는 특정 이동체가 가장 최근에 위치를 보고한 시간을 말한다. T_{uc}는 질의 타입을 정의하는데 사용되며 실제 구현된 색인 내에서 노드 탐색 경로를 결정하는 중요한 변수가 된다.

4.1.2 OL과 CL

<그림 4>의 (a)와 같이 CL(Close Line)은 3차원 공간상의 두 점을 연결한 선분으로서 (x_1, y_1, t_1, x_2, y_2, t_2)로 표현된다. 이것은 이동체가 t_1 시간에 공간 좌표 x_1, y_1에서부터 t_2 시간에 공간 좌표 x_2, y_2로 이동한 것을 의미하며 시간 축과 공간 축에 대해 닫혀있다. OL(Open Line)은 <그림 4>의 (b)와 같이 공간 좌표 x, y와 위치 보고 시간인 t, 그리고 이동체의 이동속성을 나타내는 V_x, V_y로써 표현되는 벡터이다. 이 벡터가 가리키는 방향은 이동체의 이동 방향이며 벡터가 나타내는 길이는 속도이다. 즉, 이동체가 X축 방향으로 V_x의 속도로, Y축 방향으로 V_y의 속도로 이동하는 것을 의미한다.



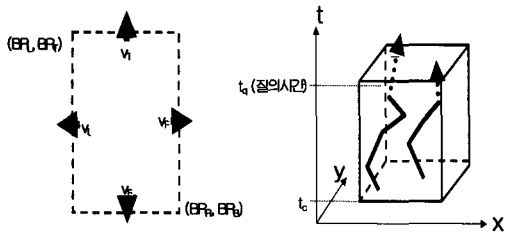
<그림 4> 3차원 공간 상에 표현된 CL과 OL

4.1.3 BR(Bounding Rectangle)

BR(bounding rectangle)은 [3]에서 정의한 TPBR의 변형이다. BR은 선분으로 이루어진 이동체의 과거궤적 정보(CL)와 시간에 대한 선형함수로 표현된 이동체의 현재 및 미래 위치정보(OL)에 대한 경계 영역을 의미한다.

이 논문에서 제시하는 새로운 경계설정 방법인 BR은 <그림 5>와 같이 (BR_T, BR_B, BR_L, BR_R, V_t, V_b, V_l, V_r, t_1, t_2)로 표현된다. BR_T, BR_B, BR_L, BR_R는 BR생성시에 노드에 포함되어 있던 CL과 OL을 최소의 영역으로 감싸는 직사각형의 각 변의 공간 좌표를 나타내고 V_t, V_b, V_l, V_r 는 각 변의 속도를 나타내며 t_1, t_2는 각각 BR의 시간 축 값이다. BR은 열려있는데 <그림 5>의 오른쪽과 같이 BR의 시간 간격의 끝을 나타내는 시간 값은 미리 설정되어 있지 않고 삽입이나 질의

등이 발생했을 때의 해당 시간으로 설정된다.



<그림 5> BR의 구조

색인의 노드에 포함된 CL과 OL이 BR의 각 값을 설정하는데 CL과 OL은 BR의 BR_T, BR_B, BR_L, BR_R 값을 설정한다. 즉, BR_L은 노드에 포함된 CL 및 OL들의 x 값 중에서 최소값으로 설정되고 BR_R은 반대로 최대값으로 설정되며 BR_T, BT_B는 각각 y 값 중에서 최대, 최소값으로 설정된다. V_R의 값은 노드 내에 포함된 OL들의 V_x 값 중에서 최대인 값으로 설정되며 나머지 변의 속도 값들도 동일한 방법으로 설정된다.

4.2 통합색인의 엔트리에 대한 단일 인터페이스

이 논문이 제안하는 새로운 색인에서는 CL과 OL에 대한 통합 인터페이스를 정의하여 두 가지 이동체 데이터 타입에 대해 동일한 방법으로 접근 가능하게 한다.

t_{ac} : 엔트리에 대한 접근 시간

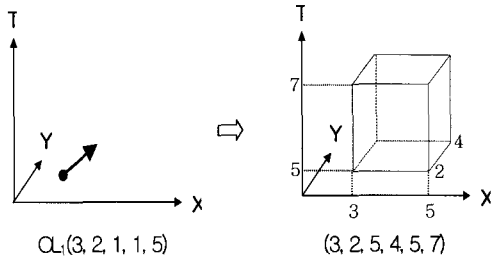
$E(CL, t_{ac}) = (x, y, x + (v_x * t_{ac} - t), (v_y * t_{ac} - t), t, t_{ac})$

$E(CL, t_{ac}) = (x_1, y_1, x_2, y_2, t_1, t_2)$

$E(BR, t_{ac}) = (BR_T + V_t(t_{ac} - t_c), BR_B + V_b(t_{ac} - t_c), BR_L + V_l(t_{ac} - t_c), BR_R + V_r(t_{ac} - t_c), t_c, t)$

<그림 6> 노드의 엔트리에 대한 통합 인터페이스 E()

t_{ac}는 색인의 엔트리에 접근한 시간을 의미한다. 이것은 OL이나 BR의 각 변이 확장되어야 할 만큼의 시간 값을 의미한다. E()는 함수로서 색인의 모든 엔트리에 대한 인터페이스의 역할을 하며 이동체 데이터 타입과 관계없이 항상 정방형(cube)의 엔트리를 유도한다.



<그림 7> 통합 엔트리 타입 E의 동작

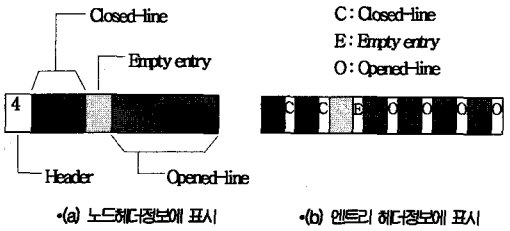
<그림 7>은 t_{ac}가 7일 때 E(OL,t_{ac})가 계산된 결과이다. OL1의 t가 5이고 t_{ac}가 7이라 하면 시간의 차이와 OL1이 가진 속도를 거리로 환산하여 그림 7의 오른쪽과 같은 정방형의 엔트리가 유도된다. E()는 CL에 대해서는 어떠한 연산도 하지 않고 CL 그대로를 유도한다. 왜냐하면 CL은 3차원 상의 선분이고 선분이 갖는 MBB는 곧 3차원 상의 정방형 엔트리로 사용될 수 있기 때문이다. BR에 대한 E()의 동작은 OL의 예와 유사하다. BR의 네 변에 대해 각 변에 설정된 속도를 BR의 t_c와 t_{ac}의 차이만큼 계산하여 위치를 보정한 후 그 결과를 유도한다.

이 논문에서 제안하는 새로운 색인의 모든 엔트리는 통합 엔트리 타입 E()로 접근 가능하다. 또한 E()의 결과는 정방형의 데이터로써 기존 R-tree의 방법과 유사한 알고리즘을 사용할 수 있다. 이 논문은 E()와 함께 R-tree의 수정된 삽입과 분할정책을 사용하여 통합 색인을 구성한다.

4.3 노드구조

공간 근접성이 높은 데이터를 색인에서도 근접한 노드에 저장하는 것이 색인의 성능에 좋다. 통합된 색인에서 이동체의 과거 위치 데이터는 CL로 현재 및 미래 위치 데이터는 OL로 표현되기 때문에 이동체의 직전위치와 새롭게 보고되는 위치간의 공간 근접성을 최대한 활용하기 위해서는 CL과 OL이 하나의 노드에 삽입될 수 있어야 한다. 이 논문에서는 CL과 OL을 함

께 저장할 수 있는 노드 구조로서 그림 8과 같은 두 가지 구조를 제시한다.



<그림 8> 노드 구조

<그림 8>의 (a)는 노드 헤더 정보를 이용하는 방법으로서, CL들은 노드의 왼쪽에 저장하고 새롭게 삽입되는 OL들은 오른쪽에 저장하며 노드의 헤더에 OL의 개수를 저장하는 방법이다. 노드의 전체 엔트리 사용 개수는 별도로 저장되어 있으며 CL의 개수는 전체 엔트리 사용 개수에서 OL의 개수를 제외한 나머지가 된다. (b)는 각 엔트리의 헤더 정보를 이용하는 방법으로서 각 엔트리마다 OL인지 CL인지 비어있는 엔트리인지의 여부를 알리는 별도의 표시(flag)를 저장한다. (a)의 구조는 노드의 엔트리가 변경될 때 마다 또는 노드가 재구성 될 때 마다 종류별로 정렬하는 단점이 있지만 질의 처리시에 OL들 혹은 CL들만을 따로 참조하기에 적합한 구조이다. (b)는 (a)와 달리 정렬에 드는 비용이 필요 없지만 특정 엔트리에 접근하기 위해서 모든 엔트리를 탐색하는 단점이 있다.

기본적으로 R-tree에 기반을 둔 이동체 색인은 아래에서 위로 성장한다[7]. 그리고 TB-tree와 같이 이동체 데이터의 시간 축 특성을 반영한 이동체 색인은 좌측에서 우측으로 성장한다[2]. 이때 이동체의 최근 위치 데이터는 색인의 우측에 집중되는 현상이 있기 때문에 만약 TB-tree에서 각 이동체의 최근 보고한 위치를 검색하기 위해서는 색인의 오른쪽부터 검색하면 된다. 따라서 제안하는 색인에서는 타입별 엔트리에 대한 참조가 유리한 <그림 8>(a)의 구조를 사용한다.

앞에서 언급한 것처럼 CL의 경우 (x_1, y_1, t_1, x_2, y_2, t_2)의 형태를 가지며 OL의 경우 (x, y, v_x, v_y, t)의 형태를 가진다. 각 변수를 4byte, node size를 2048 byte라고 가정하면 단말노드의 경우 사이즈가 큰 CL의 형태로 entry가 구성되며 oid를 포함하여 한 entry 당 28 byte의 크기를 가지며 노드에는 총 73개의 entry가 저장된다. 반면에 비단말 노드의 경우 (BR_T, BR_B, BR_L, BR_R, V_t, V_b, V_l, V_r, t_1, t_2)의 형태를 가지므로 oid를 포함한 44 byte의 크기를 가지는 entry로 구성되며 총 46개의 entry가 저장된다.

5. 삽입 및 검색 알고리즘

5.1 삽입

이 논문에서는 이동체의 위치 보고가 OL의 형태라고 가정한다. 이동체의 이전 OL을 현재 시간으로 확장한 결과와 현재 위치 값의 차이가 특정 임계 값 이상이면 위치 보고를 수행한다. 즉, 임계 값을 SendTH라고 할 때 다음 조건을 만족하면 새로운 위치를 보고한다.

OL_{last} : 동일 이동체가 직전에 보고한 위치데이터
T_{NOW} : 현재시간
X, y : 이동체의 현재 위치
tGab : $T_{NOW} - OL_{last}.t$
bSend : $OL_{last}.x + (OL_{last}.vx * tGab) - x < Send_{TH}$
AND
$OL_{last}.y + (OL_{last}.vy * tGab) - y < Send_{TH}$
If(bSend) 새로운 위치보고

<그림 9> 이동체의 새로운 위치보고 조건

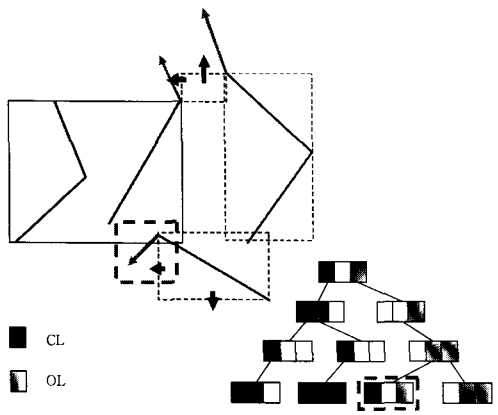
5.2 성능 비교

<그림 9>의 조건을 만족해 이동체가 새로운 위치를 보고하는 경우에 삽입 연산이 발생한다. 제안하는 색인의 삽입 알고리즘은 <그림 10>과 같다. 객체 식별자 oid를 가진 이동체의 새로운 위치 데이터 ol은 FindPrevOL의 인자로 넘겨져 동일 이동체의 이전 위치를 검색하게 된다. 새로운 ol의 이전 위치는 ol.x, ol.y로 나타나는 공간 좌표와 SendTH의 오차로 계산되어진 영역 질의를 통해 검색될 수 있다.

```
Algorithm Insert(OID oid, OL ol)
{
  // oid에 해당하는 이동체의 직전위치와 이를 포함하는 노드를 검색
  e <- FindPrevOL(oid, ol);

  // 검색된 엔트리 e를 새로운 ol과 함께 CL로 논리적 갱신 수행
  if (e exist)
       LogicalUpdate(e, ol);

  // 새로운 OL을 루트노드로부터 삽입
  RtreeInsert(ol);
}

Algorithm FindPrevOL(OID oid, OL ol)
{
  // ol에 오차범위를 적용
  minX <- ol.x - Send_TH;
  maxX <- ol.x + Send_TH;
  minY <- ol.y - Send_TH;
  maxY <- ol.y + Send_TH;

  // ol을 검색
  E_SET = RangeQueryToOL(minX, maxX, minY, maxY, ol.t);

 // 검색된 엔트리중 동일한 oid를 가진 것을 리턴
 for(e <- each entry in E_SET){
   if(e.oid == oid) ret e;
 }
 ret NULL;
}
```

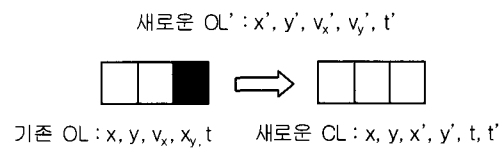
<그림 10> 삽입 알고리즘

검색된 이동체의 이전 위치 e는 LogicalUpdate를 통해 동일 노드 내의 새로운 CL로 변경된다. 즉 이전 위치를 삭제하지 않고 CL로 저장하는 것으로서 과거 데이터에 대한 질의가 가능하게 된다.



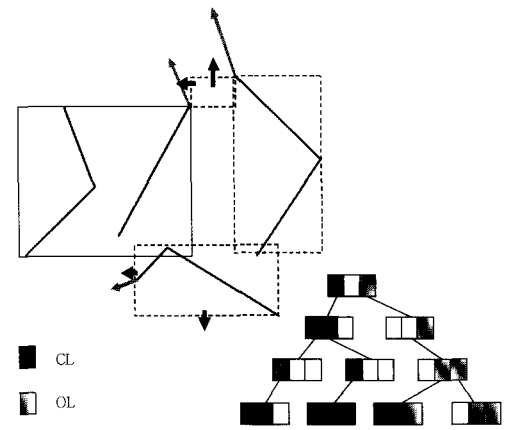
<그림 11> 색인의 구조 및 검색된 이동체의 직전위치

<그림 11>과 같이 굵은 점선으로 표시된 두 개의 사각형은 FindPrevOL을 통해 검색된 동일 이동체의 이전 위치와 이를 포함하고 있는 노드를 나타낸다. 검색된 노드는 <그림 12>와 같은 과정을 통해 CL로 변환되어 동일 노드 내의 왼쪽에 삽입된다.



<그림 12> LogicalUpdate로 OL을 CL로 변경

이후에 새로운 OL을 공간적으로 가장 근접한 BR을 가지는 노드에 삽입되게 된다. 이 논문에서 언급하지 않은 알고리즘들은 모두 R-tree의 것을 사용하며 통합 엔트리 타입 E()를 통해 새로운 위치 데이터의 보고 시간으로 계산된 값을 사용한다. <그림 11>의 예에서 OL의 삽입을 완료하게 되면 <그림 13>과 같은 색인이 구성된다. 새로운 OL은 제시한 예에서와 같이 이전 위치가 존재하는 노드에 삽입되는 경우가 많은데 이는 동일 이동체의 이전 위치와 이후 위치 간에는 공간 근접성이 매우 높으며 연속되어 있기 때문이다.



<그림 13> 삽입이 완료된 후의 색인 구성

5.2 질의 처리

5.2.1 질의 타입

이동체 색인의 응용에는 응용에 부합하는 다양한 질의 타입이 사용된다. 대표적인 예로 "어제 오후 2시부터 3시까지 서울역 광장 앞을 통과한 이동체를 검색하라" 와 같이 특정 영역과 시간 간격에 포함되는 이동체를 검색하는 영역 질의가 있으며 이러한 질의 처리를 위한 색인으로서 3DR-tree가 있다. 또한 "차량번호가 0000인 이동체의 지난 5시간 동안의 이동 궤적을 검색하라" 와 같이 특정 oid를 가진 이동체의 특정 시간 간격 동안의 이동 궤적을 검색하는 궤적 질의가 있으며 이러한 질의 처리를 위한 색인으로서 TB-tree가 있다. 이 외에도 영역 질의와 궤적 질의의 부분 혹은 조합으로 구성된 여러 가지 질의가 있다.

이 논문에서는 모든 시간 도메인에 대한 영역질의를 지원하는 색인을 제안한다. 영역 질의의 시간 축의 값을 T_{q1}, T_{q2}, 이동체의 최근 보고시간을 T_{uc}, 현재시간을 T_{now}라 하면 다음과 같이 6가지의 타입으로 분류된다.

<표 1> 질의의 시간 간격에 따른 질의 타입

질의타입	시간 축 질의 영역	시간조건
RQ_P	과거	$T_{q1} < T_{q2} < T_{uc}$
RQ_C	현재	$T_{q1} = T_{q2} = T_{now}$
RQ_F	미래	$T_{uc} < T_{q1} < T_{q2}$
RQ_{PC}	과거-현재	$T_{q1} < T_{uc}$, $T_{q2} = T_{now}$
RQ_{CF}	현재-미래	$T_{q1} = T_{now}$, $T_{uc} < T_{q2}$
RQ_{PCF}	과거-현재-미래	$T_{q1} < T_{uc} < T_{q2}$

5.4.2 질의 타입에 따른 질의처리

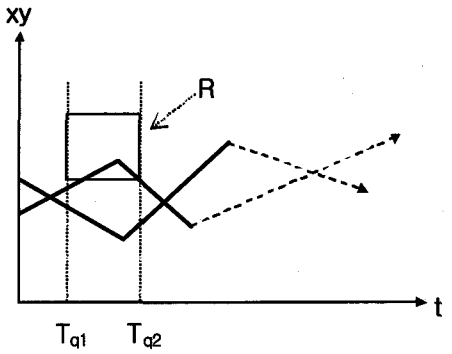
제안하는 색인은 질의 타입에 따라 참조하는 노드가 각각 다르다. 정의된 질의 타입 중 T_{q1}과 T_{q2}중 어느 것이라도 T_{uc}보다 작을 경우 CL이 저장되어 있는 노드를 반드시 참조해야 하고 T_{q1}, T_{q2}중 T_{now}이후의 시간이 포함된 질의는 기본적으로 OL이 포함되어 있는 모든 노드를 참조해야 한다. <표 2>에서 볼 수 있듯이 RQ_C, RQ_F, RQ_{CF}와 RQ_{PC}, RQ_{PCF}는 각각 참조가 되는 대상이 같으므로 동일한 동작으로 질의처리가 가능하다. 따라서 질의는 다음의 3가지 방법으로 모두 처리될 수 있다.

<표 2> 질의 타입에 따른 참조 대상 노드

질의타입	시간 조건	OL	CL	OL&CL
RQ_P	$T_{q1} < T_{q2} < T_{uc}$		O	O
RQ_C	$T_{q1} = T_{q2} = T_{now}$	O		O
RQ_F	$T_{uc} < T_{q1} < T_{q2}$	O		O
RQ_{PC}	$T_{q1} < T_{uc}$, $T_{q2} = T_{now}$	O	O	O
RQ_{CF}	$T_{q1} = T_{now}$, T_{uc}, $< T_{q2}$	O		O
RQ_{PCF}	$T_{q1} < T_{uc}$, $< T_{q2}$	O	O	O

- **RQ_P 의 질의처리**

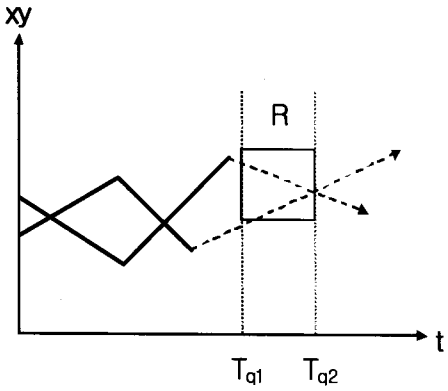
과거 시간에 대한 질의 처리는 색인 내의 CL들에 대한 참조로써 가능하다. <그림 14>와 같이 T_{q1}, T_{q2}이 T_{uc}보다 과거이므로 질의 영역은 OL과 교차하지 않는다. 따라서 T_{uc} 시간에 재구성된 BR의 각 변의 공간 좌표 값을 그대로 사용하여 질의 처리를 수행한다.



<그림 14> RQP 질의 처리

- **RQ_C, RQ_F, RQ_{CF} 의 질의 처리**

T_{now}는 항상 T_{uc}의 이후 시간이다. 따라서 세 가지 질의의 시간은 모두 T_{uc}이후가 된다. 따라서 질의는 OL을 참조하며 OL에 대해 T_{now}와 T_{uc}의 차이만큼의 시간 차를 적용해야 한다. 즉, T_{now}에서의 현재 위치는 T_{uc}이후의 예측된 미래 위치를 의미한다.



<그림 15> 미래영역 데이터에 대한 질의 처리

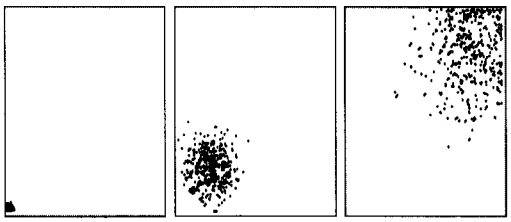
- **RQ_{PC}, RQ_{PCF}의 질의 처리**

이 두 개의 질의 타입은 T_{q1}가 T_{uc} 이전의 시간이며 T_{q2}는 T_{uc} 이후의 미래 시간인 경우이다. 주어진 질의 영역 R에 해당하는 모든 엔트리가 탐색되어야 하며 이 경우 T_{uc}는 T_{q2}의 값으로 설정된다.

6. 구현

6.1 구현 환경

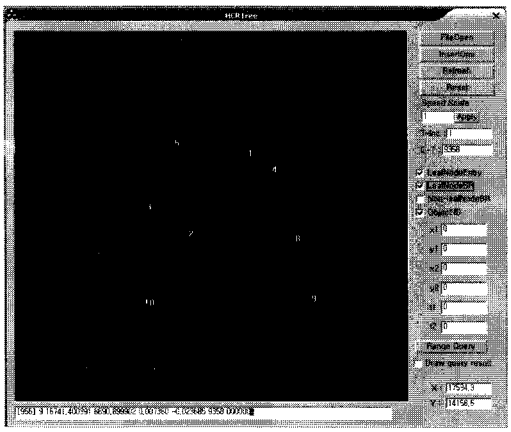
제안한 색인 구조는 Microsoft Visual C++ 6.0을 사용하여 구현하였으며 실험은 256MB의 메인메모리를 가진 펜티엄 1.7에서 수행하였다. 데이터는 GSTD 알고리즘을 사용하여 생성하였으며 그림 16과 같이 이동체들이 북동쪽으로 점차 확산되면서 이동되는 형태이다. 이러한 데이터뿐만 아니라 다양한 방향으로 확산되는 데이터에 대하여 실험을 수행하였으며 방향에 따른 추가적인 특징이 발견되지 않아 이 논문에서는 이 데이터에 대한 실험만을 언급한다. 데이터는 점 데이터로서 이 논문을 위해 별도의 데이터 타입 변환 툴을 작성하여 OL의 형태로 변환 하였다. 이 툴은 이전 위치의 데이터와 이후 위치의 데이터간의 시간과 공간의 차이를 이용하여 각 축에 대한 Vx, Vy를 추출하며 임계 값(SendTH)을 100으로 설정 하였다. 생성된 데이터의 범위는 X, Y축 모두 0부터 14000이다.



<그림 16> GSTD 알고리즘을 통해 생성된 실험데이터

6.2 색인 실험 툴

새로운 색인을 개발하는 경우 색인이 올바로 동작하는지의 여부를 판단하기가 용의하지 않으므로 제안하는 색인과 별도로 색인 실험 툴을 개발 하였다.

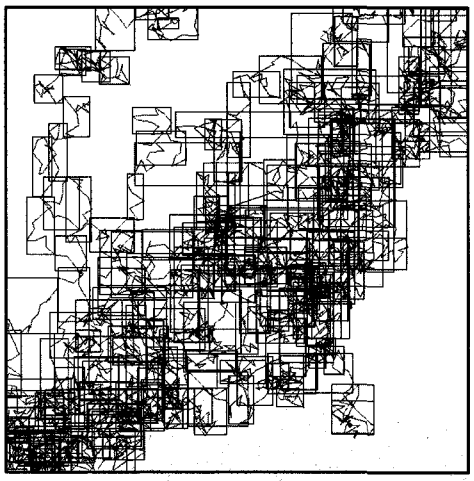


<그림 17> 색인 실험 툴

<그림 17>은 실험 데이터 중 10개 이동체의 데이터만 추출하여 삽입한 후 색인 실험 툴을 통해 화면에 출력한 것이다. 색인 테스트 툴의 오른쪽은 다양한 기능을 가진 버튼과 상태를 나타내는 편집 박스로 구성되어 있으며 이를 통해 구현된 색인으로 삽입이나 질의를 수행 할 수 있다. 이 색인 실험 툴은 R-tree에 기반한 대부분의 색인에 대해 거의 수정 없이 사용할 수 있도록 하였으므로 새로운 색인을 개발하거나 기존 색인의 성능 향상 정책 등의 효과를 직접 확인하는 경우 유용하게 사용될 수 있다.

6.3 구현 결과

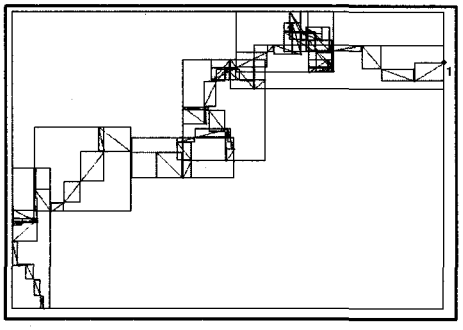
6.3.1 실험 데이터 전체 삽입



<그림 18> 실험 데이터 전체를 삽입한 모습

<그림 18>과 같이 삽입된 전체 데이터는 남서쪽에서 북동쪽을 향해 이동하는 형태이다. 단말노드의 구성을 보다 쉽게 관찰하기 위해 팬아웃(fan-out)은 10으로 설정했다.

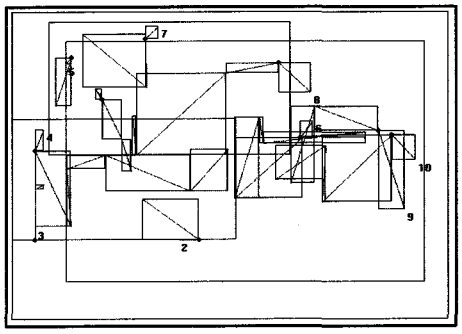
6.3.2 LogicalUpdate의 동작 테스트



<그림 19> LogicalUpdate를 통한 색인 구성

<그림 19>는 실험 데이터 중 oid가 1인 것만을 추출하여 연속적으로 삽입한 후 비단말 노드의 BR, 단말노드의 BR, 단말노드 엔트리, 이동체 ID등을 출력한 모습이다. 이 화면은 동일 이동체의 이전 위치 검색과 LogicalUpdate가 올바르게 수행되고 있다는 것을 증명한다. 이 예에서도 마찬가지로 *oid* 1의 이동체는 북동쪽을 향해 이동하고 있음을 확인할 수 있다. 역시 팬아웃은 10으로 설정하였다.

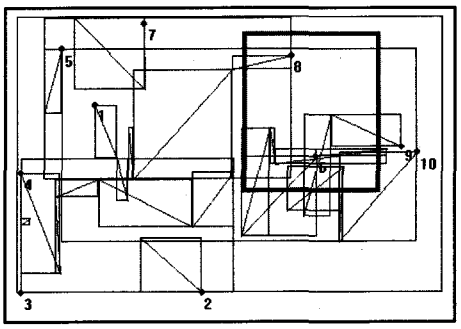
6.3.3 OL의 이동 속성으로 인한 시간에 따른 BR의 확장



<그림 20> 시간의 흐름에 따른 BR의 확장

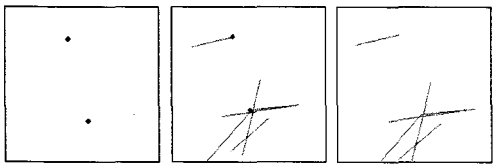
<그림 20>은 BR의 확장이 올바르게 동작하는지를 확인하기 위해 특정 시간 이후 BR이 확장된 모습을 나타낸다. 즉, 실험 데이터의 OL(점으로 표현)들은 3000의 시간이 흐르면 그림과 같은 위치로 이동(점과 연결된 선분)하게 된다.

6.3.4 시간 도메인 질의



<그림 21> 영역질의

<그림 21>은 영역 질의 테스트에 사용될 질의 영역 R을 색인 실험 툴을 통해 출력 한 것이다. R 내에는 다수의 CL과 두 개의 OL이 포함되어 있다. 이 실험에서는 T_{q1}과 T_{q2}의 값을 5.4.2절에서 설명한 세 가지 경우에 각각 해당하도록 설정한 후 색인 실험 툴을 이용하여 출력하였다.



<그림 22> 질의 시간에 따른 영역질의의 결과

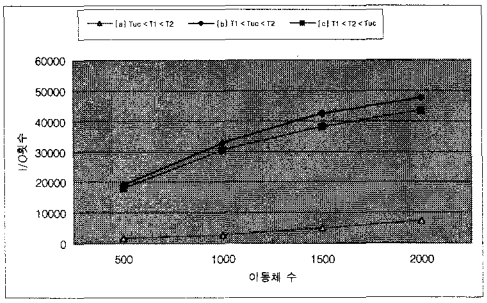
<그림 22>는 <표 3>과 같이 설정된 질의 영역 R에서 T_{q1}과 T_{q2}의 시간 값에 따른 질의 결과를 화면에 출력한 것이다. <그림 22>의 (a)는 현재 및 미래위치 데이터 검색의 결과이고 (b)는 이동체의 과거 궤적과 현재 및 미래위치 데이터를 포함하는 질의의 결과이며 (c)는 과거궤적 검색의 결과이다.

<표 3> 각 변수의 값

	T_{q1}	T_{q2}
(a)	330	400
(b)	150	450
(c)	150	310

C-T	320
X1	2390
Y1	1100
X2	2700
Y2	2100

<그림 23>은 이동체의 수가 늘어남에 따른 질의 성능에 대한 결과이다. 이동체 수가 각각 증가함에 따라 <그림 22>의 세 가지 경우에 대해 영역 질의를 수행하는 경우 접근한 Disk I/O를 나타낸다. (a)의 경우 전체 색인에서 OL의 비중이 낮고 그 대부분이 색인의 우측으로 편중하기 때문에 다른 경우에 비해 비용이 적게 든다. (b)와 (c)는 색인의 거의 모든 노드를 참조 대상으로 하므로 크게 차이가 나지 않음을 볼 수 있다. 본 논문에서 제안하는 색인은 (b)와 같이 모든 시간 도메인을 포함하는 질의에 대해 좋은 성능을 나타내며 (a)와 (c)와 같이 과거 또는 미래에 관한 질의에 대해서는 각 질의에 대한 기존 색인보다 성능이 다소 낮아진다.



<그림 23> 이동체 수에 따른 영역질의성능

7. 결론 및 향후 연구

이 논문에서는 이동체의 과거 궤적과 현재 및 미래 위치를 하나의 색인에 구성하여 모든 시간 도메인에 대한 질의를 지원하는 새로운 색인을 제안하였다. 통합 색인의 필요성에 대해 언급하였고 이동체의 과거 데이터와 현재 및 미래위치 데이터를 통합하는 과정에서 발생하는 문제점에 대해 논의한 후 새로운 색인에 필요한 개념들을 제시하였다.

이동체의 과거 데이터와 현재 및 미래 데이터를 모두 포함 할 수 있도록 BR을 새롭게 정의하였으며 이동체의 이전 위치에 해당하는 시간에 대한 함수를 선분으로 갱신하는 과정을 설명하였다. 또한 동일 이동체의 최근 보고한 위치와 현재 위치는 서로 공간 근접성이 높기 때문에 공간 근접성을 최대한 활용하기 위한 구조로서 선분과 시간에 대한 함수를 함께 저장하여 관리할 수 있는 새로운 노드 구조를 제시 하였다. 마지막으로 이 논문에서 제시된 새로운 색인이 지원하는 질의 타입을 요청 시간에 따라 과거, 현재, 미래 등으로 구분하였고 각각에 따른 질의 처리 방법을 제시한 후 실제 구현을 통한 실험으로써 질의처리가 가능함을 보였다.

향후 연구로써 이 논문이 제시하는 색인의 응용에

따른 최적화가 필요하다. 최근에 연구 발표된 이동체 색인의 여러 가지 성능 향상 기법들을 적용하여 보다 빠른 검색 성능을 나타낼 수 있도록 하는 연구가 필요하다.

참고문헌

[1] Yannis Theodoridis, "Spatio-Temporal Indexing for Large Multimedia Applications", In Proc. Of the 3rd IEEE Conf. on Multimedia Computing and Systems, pages 441-448, June 1996

[2] S. Saltenis, C. S. Jensen, S.T. Leutenegger, and M. A. Lopez,"Indexing the Positions of Continuously Moving Objects." , In Proc. ACM SIGMOD on Management of data, p331 - 342, 2000.

[3] Pfoser, D., Jensen, C., Theodoridis Y., "Novel Approaches to the Indexing of Moving Object Trajectories", In Proc. Of the 26th Int'l Conference on VLDB, pp. 395-406, 2000.

[4] Norbert Beckmann,Hans-Peter Kriegel,Ralf Schneider, Bernhard Seeger,"The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles." SIGMOD Conference 1990; 322-331

[5] Betty salzberg, vassilis j. tsotras, "Comparison of Access Methods for Time-Evolving Data" ACM Computing Surveys, Vol.31, No.2, pp.158-221, 1999

[6] Mario A. Nascimento, Jefferson R. O. Silva, Yannis Theodoridis "Evaluation of Access Structures for Discretely Moving Points." Spatio-Temporal Database Management,171-188 1999

[7] A. Guttman, "R-trees: A dynamic index structure for spatial searching," Proc. of the ACM SIGMOD Int'l Conf. on Management of Data, pp. 47-54, 1984.

[8] Randal C. Nelson, Hanan Samet, "A Population Analysis for Hierarchical Data Structures.", SIGMOD Conference, pp 270-277, 1987

[9] 이창헌, 임덕성, 홍봉희, "KDB-Tree 를 사용한 이동체 색인의 동적 변경", 한국정보과학회 데이터베이스 연구회 2002학술발표논문집, 제18권 2호, p117-124.

반재훈

1997년 부산대학교 컴퓨터공학과 (공학사)

1999년 부산대학교 컴퓨터공학과 (공학석사)

2001년 부산대학교 컴퓨터공학과 박사수료

2002년 ~ 현재 경남정보대학 인터넷응용계열 조교수

관심분야 : 개방형 GIS, 이동객체, 객체지향데이터베이스

홍봉희

1982년 서울대학교 컴퓨터공학과 졸업 (학사)

1984년 서울대학교 컴퓨터공학과 졸업 (석사)

1988년 서울대학교 컴퓨터공학과 졸업(박사)

1987년 ~ 현재 부산대학교 컴퓨터공학과 교수

관심분야 : 이동객체 데이터베이스, 모바일 데이터베이스, 공간 데이터베이스

전희철

2002년 동의대학교 컴퓨터공학과 졸업 (공학사)

2004년 부산대학교 대학원 GIS협동과정 졸업(공학석사)

2004년 ~ 현재 삼성전자 정보통신 연구소 차세대 단말팀

관심분야 : 지리정보 시스템, RFID 미들웨어, 모바일 GIS, 이동체 색인

안성우

1999년 부산대학교 컴퓨터공학과 졸업 (공학사)

2001년 부산대학교 대학원 컴퓨터공학과 졸업(공학석사)

2003년 부산대학교 대학원 컴퓨터공학과 박사과정 수료

2001 ~ 현재 (주)사이버맵월드 부설연구소 재직

관심분야 : 지리정보 시스템, RFID 미들웨어, 모바일 GIS, 이동체 색인

김진덕

1993년 부산대학교 컴퓨터공학과 졸업 (공학사)

1995년 부산대학교 컴퓨터공학과 졸업 (공학석사)

2000년 부산대학교 컴퓨터공학과 졸업(공학박사)

1998년 ~ 2001년 부산정보대학 정보통신계열 전임강사

2001년 ~ 현재 동의대학교 컴퓨터공학과 조교수

관심분야 : 지리정보시스템, 텔레메틱스