# 효과적인 차량 위치 검색을 위한 차량 관리 시스템의 설계 및 구현

## (Design and Implementation of a Vehicle Management System for Effective Retrieval of Vehicle Locations)

이 응 재 † 오 준 석 †† 정 영 진 ††
(Eung Jae Lee) (Jun Seok Oh) (Young Jin Jung)

남 광 우 ††† 이 봉 규 †††† 류 근 호 †††††
(Kwang Woo Nam) (Bong Gyou Lee) (Keun Ho Ryu)

**요 약** 기존의 차량 위치 추적 시스템은 기존의 상용 DBMS에 단순히 모든 차량 정보를 저장하여 관리하기 때문에 시간에 따라 연속적으로 변화하는 차량의 위치 데이타를 효과적으로 검색하지 못한다. 따라서 이 논문에서는 모바일 환경에서 차량의 위치 정보를 효과적으로 관리하고, 사용자에게 차량 관련 정보를 제공할 수 있는 차량 위치 관리 시스템을 설계하고 구현한다. 제안 시스템은 차량 위치 관리 서버와 모바일 클라이언트로 구성되며 이동 차량의 위치와 관련된 시공간 질의를 처리한다. 아울러 시스템에 저장되지 않은 차량의 위치 정보까지도 추정하여 사용자에게 제공한다. 이 시스템은 차량 위치 관리 서버의 이동 객체 색인 관리 모듈을 사용하여 차량의 위치 정보를 효과적으로 관리할 수 있다.

**키워드** : 실시간 위치 추적, 차량 관리 시스템, 이동객체 색인, 모바일 위치 검색

***Abstract*** Various researches on moving object modeling, uncertainty processing, and moving object indexing have been carried out in the field of moving object databases. However, previous location tracking systems cannot efficiently retrieve location data of vehicles, because they manage all location information of vehicles using the conventional database. In this paper, we design the vehicle location management system that is able to manage and retrieve vehicle locations efficiently in mobile environment. The proposed system consists of a server for managing vehicle locations and mobile clients. The system is able to not only process spatiotemporal queries related to locations of moving vehicles but also provide moving vehicles' locations which are not stored in the system. The system is also able to manage vehicle location data effectively using a moving object index.

**Key words** : Real-time Location Tracking, Vehicle Management System, Moving Object Index, Retrieving Location in Mobile Environment

· 이 논문은 과학기술부 · 한국과학재단 지정 청주대학교 정보통신연구센터의 지원에 의한 것입니다.

† 학생회원 : 충북대학교 전자계산학과
eungjae@dblab.chungbuk.ac.kr
†† 비 회 원 : 충북대학교 전자계산학과
seokoh@dblab.chungbuk.ac.kr
yjjeong@dblab.chungbuk.ac.kr
††† 비 회 원 : 군산대학교 컴퓨터정보과학과 교수
kwnam@kunsan.ac.kr
†††† 정 회 원 : 한성대학교 정보전산학부 교수
bong97@hansung.ac.kr
††††† 종신회원 : 충북대학교 전기전자컴퓨터공학부 교수
khryu@dblab.chungbuk.ac.kr

논문접수 : 2003년 11월 27일
심사완료 : 2004년 11월 4일

## 1. 서 론

무선 통신 기술의 발달은 기존의 유선 통신 환경에서만 제공되었던 다양한 정보 시스템들을 무선 환경에서 사용할 수 있도록 하였으며 이로 인하여 모바일 사용자들은 어느 장소에서나 원하는 정보들을 얻을 수 있게 되었다. 또한, GPS와 같은 위치 측위 장비의 발달은 사용자의 위치 정보를 실시간으로 보다 정확하게 획득할 수 있도록 하였고, 이러한 위치 정보를 이용한 다양한 위치 정보 시스템의 개발이 활발히 진행되고 있다. 실시간 위치 정보를 사용하는 대표적인 예로 물류 차량 관리, 항공 교통 통제, 위치 기반 서비스 등을 들 수 있으

며 특히, 관제 센터에서 차량의 위치를 실시간으로 모니터링하는 차량 위치 추적 시스템[1-5]을 대표적인 응용 시스템 사례로 들 수 있다. 그러나 기존의 시스템들은 주로 유선 통신망을 사용하여 중앙 관제 센터에서 차량 위치를 추적하고, 기존의 상용 DBMS를 그대로 사용하기 때문에 시간의 흐름에 따라 위치가 연속적으로 변화하는 이동 객체[6,7]와 관련된 질의를 효과적으로 처리하지 못하고 있다. 또한 대량의 차량 궤적 데이타를 모두 데이타베이스에 저장하기 때문에 특정 시간 구간에서 차량의 위치와 연관된 데이타를 검색할 경우 데이타베이스 내의 모든 데이타를 검사하게 된다. 이로 인하여 사용자가 차량 이동 경로를 검색할 경우 이에 해당하는 차량의 위치 정보를 빠르게 제공하기 어렵다.

이 논문에서는 지금까지의 이동 객체 관련 연구들을 기반으로 무선 네트워크 상에서 모바일 클라이언트를 사용하여 차량 위치를 실시간으로 관리하기 위한 차량 관리 시스템을 구현한다. 제안된 시스템은 클라이언트에서 전송된 차량의 위치를 수집하는 차량 데이타 수집기, 차량의 위치 정보를 저장 및 관리하고 모바일 클라이언트에 차량 위치 검색 결과를 제공하는 서버와 차량 위치 관련 정보를 서버에 요청하고 서버로부터 제공된 위치 검색 결과를 출력하는 모바일 클라이언트로 구성된다. 우선, 차량 관리 시스템의 각 모듈과 이동 객체 색인 관리 모듈의 구조를 설명한다. 그리고 시스템의 각 모듈별 기능 및 처리 알고리즘을 제시한다. 특히, 이동 객체 색인에 저장된 이력 위치 정보를 효과적으로 관리 및 검색을 수행하며 동시에 다수의 사용자가 접근하여 사용하는 모바일 환경에 적합하도록 동시성이 고려된 이동 객체 색인을 제안한다. 또한 모바일 클라이언트를 위한 서버 연결 모듈과 데이타 변환 및 송수신 모듈을 설계, 구현하여 무선 환경에서 실시간으로 차량 위치 검색이 가능함을 보이고 기존의 시스템과 제안 시스템에서 지원되는 질의 종류를 비교, 평가한다.

이 논문의 구성은 다음과 같다. 2장에서는 차량 위치 추적 시스템과 이동 객체 색인 관련 연구를 검토한다. 3장에서는 차량 관리 시스템의 구조, 이동 차량 데이타의 효과적인 관리방법과 주요 모듈의 처리과정을 제시한다. 4장에서는 시스템 구현 환경과 구현 결과를 살펴보고, 5장에서는 결론과 향후 연구 방향을 제시한다.

## 2. 관련 연구

차량 위치 추적 시스템은 이동 중인 차량의 위치 및 상태를 추적하여 실시간으로 전자지도에 표시함으로써 차량의 운행상황을 파악하는 시스템이다. 이 시스템은 차량에 장착된 GPS로부터 수신된 좌표들을 토대로 각 차량의 주행경로, 속도 등을 중앙 컴퓨터의 전자지도에 표시하여 차량업무를 관리하고 통제한다. 기존의 대표적인 차량 위치 추적 시스템으로 첨단 대중교통 시스템(APTS : Advanced Public Transportation Systems)[1-3]의 차량 관리 시스템과 첨단 화물 운송 시스템(CVO : Commercial Vehicle Operation system)[4,5]을 들 수 있다. APTS의 차량 관리 시스템은 미국의 첨단 대중교통 시스템에서 차량의 위치 및 승객과 관련된 정보를 제공해 주는 시스템이다. 이 시스템은 GIS(Geographic Information Systems), AVL(Automatic Vehicle Location) 시스템, APC(Automated Passenger Counters), TOS(Transit Operations Software)로 구성되어 있으며 이 중 AVL 시스템에서 GPS가 장착된 차량의 위치를 파악하고 이 정보를 일정한 간격을 두고 차량 배치 센터에 전송하는 역할을 한다. 첨단 화물 운송 시스템은 화물 및 화물차량을 효율적으로 관리하여 물류비 절감, 안전사고 방지 및 돌발사고에 대한 응급처리 기능을 향상시키기 위해 개발된 시스템이다. 이 시스템은 화물 및 화물 차량의 위치를 추적하여 효율적으로 차량 및 배차를 관리하는 화물 및 화물 차량 관리 시스템(FFMS : Freight and Fleet Management System)과 위험물 적재차량의 위치를 추적하여 차량을 특별 관리하고 돌발상황 시 신속하게 사고를 처리하는 위험물 차량관리 시스템(HMMS : Hazardous Material Management System)로 구성된다.

그러나 이러한 기존의 시스템들은 상용 DBMS를 사용하기 때문에 연속적으로 변하는 차량의 이동 정보를 효과적으로 처리하지 못하며 데이타베이스에 저장되지 않는 위치 정보를 사용자가 요구할 경우 이와 관련된 연산을 처리하지 못하는 문제점이 있다. 기존 차량 위치 추적 시스템의 문제점을 해결하기 위하여 이동 객체 관리를 위한 많은 연구들이 진행되었으며 대표적인 응용 시스템 연구에는 DOMINO(Databases fOr MovINg Objects)[6-9], CHOROCHRONOS[10,11], Battlefield Analysis[12]가 있다. 먼저 DOMINO 프로젝트에서 개발된 프로토타입은 이동 차량의 현재 위치, 속도, 방향 정보를 이용하여 차량의 미래 위치를 예측하는 방법에 초점을 맞춘 시스템이다. CHOROCHRONOS에서는 이동 차량의 데이타 모델링 및 색인에 관한 연구와 위치 및 궤적을 관리하는 차량 관리 시스템에 대한 연구가 이루어졌다. 하지만 이 연구는 GPS 기반의 수송 관리 시스템과 멀티미디어 시스템에 적용한 응용 시나리오는 제시되었으나 프로토타입으로 개발된 사례가 없다. Battlefield Analysis는 부대와 탱크를 이동 점 객체의 특성을 갖는 이동 객체로 정의하고 모의 전장에서 이들의 움직임을 예측하여 의사결정에 활용할 수 있도록 개발된 전장분석 프로토타입이다. 그러나 이 프로토타입은

실시간으로 제공되는 데이타를 효과적으로 처리하지 못하며 이로 인하여 모바일 환경에는 적용할 수 없는 단점이 있다.

최근에는 차량을 이동 점 객체로 정의하고 이동 객체 데이타베이스를 이용한 차량 위치 추적 시스템의 연구[8]가 수행되었다. 이 시스템들은 이동 객체를 시간에 따라 객체의 공간 정보가 연속적으로 변경되는 시공간 데이타로 이동 점과 이동 영역으로 구분한다. 그러나 이러한 시스템은 데이타베이스에 많은 양의 차량 데이타를 모두 저장하기 때문에 사용자가 요청한 차량의 위치 정보를 빠르게 제공하기 어렵다. 이와 같은 문제점 해결을 위하여 이동 차량의 위치 정보를 효과적으로 관리하기 위한 이동 객체 색인[10,11]에 대한 많은 연구가 수행되어 왔다.

이 논문에서는 이동 객체 관련 연구들을 기반으로 한 차량 관리 시스템을 제안한다. 제안된 시스템은 모바일 환경에서 시간에 따라 이동하는 이동 차량의 궤적과 관련된 질의를 빠르게 처리하기 위하여, TB-tree[11]를 기반으로 하여 차량의 궤적 데이타를 관리한다. 또한 다수의 사용자가 동시에 접속하는 모바일 환경에 적합하도록 동시성 제어 기법을 적용한 색인 알고리즘을 제안한다. 따라서 제안된 시스템은 이동 궤적 색인을 사용하여 이동 차량 데이타를 효과적으로 관리하고 모바일 클라이언트를 사용하여 차량의 위치 관련 질의를 처리할 수 있으며 다양한 차량 위치 추적 시스템에 적용이 가능하다.

## 3. 차량 관리 시스템의 설계

이 장에서는 효과적인 차량 위치 검색을 위한 차량 관리 시스템의 구조와 시스템을 구성하고 있는 각 모듈의 기능을 제시한다. 그리고 이동 차량의 위치 데이타 관리를 위한 이동 궤적 색인 관리 모듈 구조와 이동 궤적 색인 파일을 구성하고 있는 이동 차량 정보의 형태를 제시한다. 마지막으로 시스템의 각 모듈별 알고리즘을 제시한다.

### 3.1 차량 관리 시스템 구조 및 모듈 별 기능

제안된 시스템은 그림 1과 같이 차량 데이타 수집기, 차량 위치 관리 서버, 모바일 클라이언트로 구성된다. 차량 데이타 수집기는 GPS 수신기에서 수신된 차량의 위치를 TM 좌표로 변환하여 차량 위치 관리 서버로 전송한다. 차량 데이타 수집기는 데이타 수신 모듈, TM 좌표 및 패킷 데이타 변환 모듈, 데이타 전송 및 저장 모듈로 구성되어 있다. 차량 위치 관리 서버는 차량 위치 데이타를 저장, 관리하고 모바일 클라이언트에서 요청된 차량 위치 검색 질의에 응답한다. 차량 위치 관리 서버의 구성은 패킷 데이타 송신 및 수신 모듈, 패킷 데이타 변환 모듈, 차량 데이타 저장 모듈, 질의 처리 모듈, 위치 추정 모듈, 이동 객체 색인 관리 모듈로 되어 있다. 모바일 클라이언트는 서버에 실시간으로 이동 차량의 위치 검색에 관한 질의를 요청하고 그 결과를 확인할 수 있다. 모바일 클라이언트의 구성은 사용자 인터페이스, 질의 입력 모듈, 패킷 데이타 변환 모듈, 결과 출력 모듈, 패킷 데이타 송수신 모듈로 되어있다.

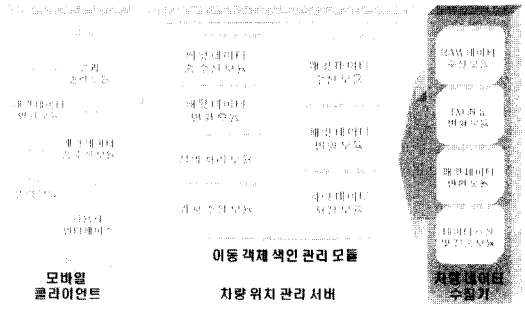



그림 1 차량 관리 시스템 구조도

차량 위치 관리 서버의 각 모듈별 기능을 살펴보면 패킷 데이타 수신 모듈은 차량 데이타 수집기로부터 주기적으로 전송된 패킷 형태의 차량 데이타를 수신한다. 수신된 패킷 데이타는 패킷 데이타 변환 모듈에서 이동 객체 색인 파일에 저장 될 수 있는 형태로 변환되며 차량 데이타 저장 모듈을 통하여 색인 페이지 파일에 저장된다. 실시간으로 전달되는 차량의 데이타를 효과적으로 관리하기 위하여 이동 객체 색인 관리 모듈에서는 이동 차량 데이타를 날짜별로 관리한다. 패킷 데이타 송수신 모듈을 통하여 모바일 클라이언트에서 전달된 패킷 형태의 질의 데이타는 패킷 데이타 변환 모듈을 거쳐 질의 처리가 가능한 형태로 변환된다. 변환된 정보는 질의 처리 모듈에서 해당 연산 처리기로 전송된다. 연산 처리기에서는 이동 객체 색인 관리 모듈을 통하여 해당 날짜의 차량 위치 정보를 검색한다. 모바일 클라이언트로부터 이동 객체 색인 파일에 저장되지 않은 과거 및 미래 위치정보와 관련된 질의가 요청되면 위치 추정 모듈이 저장되지 않은 위치 정보를 계산한다.

### 3.2 이동 차량의 위치 데이타 관리

이 논문에서 제안하는 시스템은 차량 데이타 수집기로부터 전달된 대용량의 차량 정보를 효과적으로 관리 및 검색하기 위하여 그림 2와 같은 구조를 갖는 이동 객체 색인 관리 모듈을 사용한다. 차량 데이타 수집기로부터 전달된 차량의 위치 정보는 차량 위치 관리 서버의 이동객체 색인 관리 모듈을 통하여 이동 객체 색인 파일에 실시간으로 저장된다.

이동객체 색인 관리 모듈은 현재 위치 색인과 과거

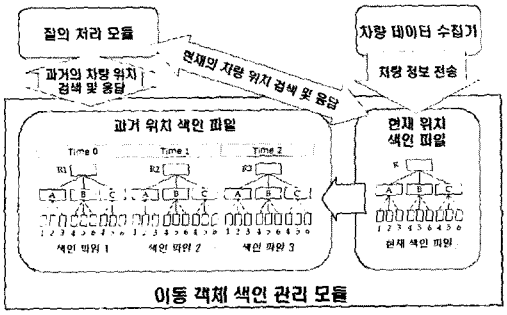



그림 2 이동 객체 색인 관리 모듈 구조도

위치 색인으로 구성된다. 차량 데이타 수집기에서 전달된 위치 정보는 일단 현재 위치 색인 파일에 저장된다. 현재 위치 색인은 R*-Tree[13]를 이용하여 구현하였다. 차량의 위치가 변경되었을 경우, 차량의 변경된 이전 위치 정보는 차량의 궤적을 구성하는 Line Segment로 변환되어 과거 위치 색인에 저장된다.

제안된 시스템은 이동 객체의 궤적을 효과적으로 관리하고, 궤적 관련 질의 요청 시 빠르게 궤적 데이타를 검색하기 위하여 TB-tree를 기반으로 하는 이동 객체 색인을 사용하였다. TB-tree는 특정 이동 객체의 이동 궤적을 보호하기 위하여 동일한 궤적을 구성하는 Line Segment 정보를 동일한 리프 노드에 저장하고, 같은 궤적을 구성하는 Line Segment를 저장하는 리프 노드들 간에는 링크드리스트를 사용하여 연결하였기 때문에 특정 객체의 궤적 정보를 빠르게 검색할 수 있는 장점을 갖는다. 과거 위치 색인 관리 모듈에서 생성되는 색인 페이지 파일은 다음과 같은 구조를 갖는다. 색인 파일은 페이지 단위의 데이타 집합으로 이루어져 있으며 색인 파일에 저장된 데이타들을 트리 형태로 표현하면 그림 3과 같다.

한 페이지는 트리의 한 노드에 해당하는 정보들이 저장되어 있다. 모든 노드에는 노드의 깊이를 나타내는 레벨과 노드 내의 엔트리 사용량을 나타내는 Used, 그리고 각 엔트리 정보들이 저장되어있다. 각 엔트리에는 차량의 이동 시점과 그 때의 위치 좌표를 MBB(Minimum Bounding Box) 형태로 저장하고 있으며 하위 노드의 페이지 번호를 저장하고 있다. 또한 리프 노드에는 차량 ID가 저장되며, 각각의 엔트리에는 MBB뿐만 아니라 차량의 방향 정보(Dir)도 저장된다. 또한 각각의 리프 노드들은 같은 차량 ID를 갖는 노드들끼리 리스트로 연결되어 있다.

모바일 환경에서는 동시에 많은 사용자들이 데이타에 접근하기 때문에 동시성 제어를 통한 성능 향상이 필요하다. 이 논문에서는 동시성 제어를 효과적으로 제공하기 위하여 기존의 TB-Tree에 2PL(Two PhaseLocking) protocol 기법을 적용하여 확장한 방법을 제안한다. 다차원 색인 구조를 위한 대부분의 동시성 제어 알고리즘은 분할이 수행되는 노드에 배타 잠금(Exclusive Lock)을 획득한다. 따라서 다차원 색인 구조에서 2PL Protocol을 그대로 적용하였을 경우, 동시성을 저하시키는 가장 큰 요인은 새로운 데이타 삽입으로 인하여 발생되는 노드 분할과 변경된 MBB 정보의 상위 노드 전파이다. 분할 연산은 상위 노드에 반영되고, 다시 상위 노드는 분할될 수 있다. 이때 배타 잠금은 탐색 연산을 지연시키며, 색인의 동시성을 저하시키는 원인이 된다. MBB 변경 역시 배타 잠금을 통하여 탐색 연산의 성능을 저하시킨다. MBB 변경은 분할 연산 보다 덜 복잡하고 짧은 시간을 필요로 하지만, 훨씬 빈번하게 발생하여 동시성을 저하시킨다. 이 논문에서는 새로운 데이타가 삽입될 때 색인의 배타 잠금을 최소화하여 탐색 연산의 성능을 저하시키지 않도록 Ravi Kanth 등이 제안한

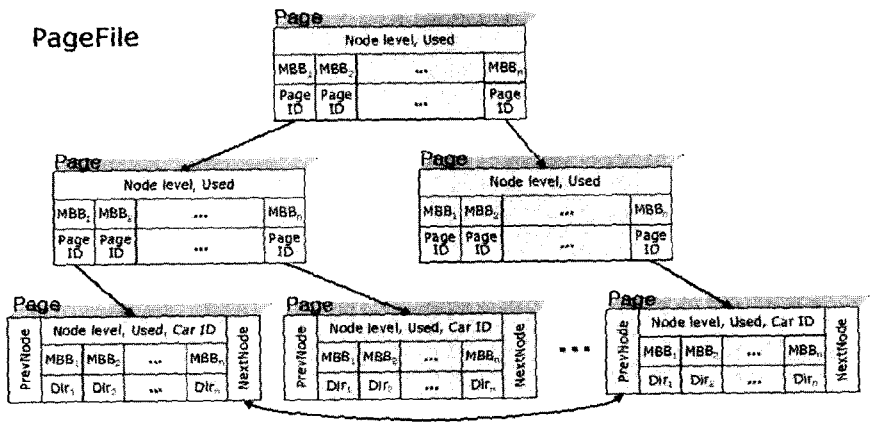



그림 3 색인 페이지 파일의 트리구조

TDIM (Top Down Index Modification) 동시성 제어 방법[14]을 사용하여 TB-Tree의 동시성 제어 기능을 향상시켰다.

TDIM 기법은 새로운 데이타가 삽입될 리프노드를 검색할 때, 루트노드부터 리프노드까지 경로를 따라 가면서 단 하나의 노드에 대하여 배타잠금을 획득하고 MBB를 변경시킨다. 이 방법은 노드가 분할 될 경우를 제외하고 리프노드까지의 경로를 따라 Top Down으로 한 단계씩 MBB 변경을 수행한다. 또한 MBB 변경을 위해 획득한 잠금은 노드를 write하는 순간을 제외하고는 탐색자가 획득하는 잠금과 호환이 가능하기 때문에 탐색자는 정상적으로 탐색을 수행할 수 있다. TB-Tree의 경우 Append only 데이타를 다루기 때문에 데이타의 삭제가 발생하지 않으며, 노드의 병합도 발생하지 않는다. 또한 리프 노드에 저장되는 궤적 정보의 유지를 위하여 노드의 분할을 수행하지 않고, 대신 새로운 노드를 생성하여 삽입한 데이타를 저장한다. 따라서 TDIM 기법에서의 노드 분할 시에 요구되는 복잡한 연산이 불필요하다. 제안된 시스템의 과거 위치 색인은 먼저 데이타가 삽입될 리프 노드까지의 경로를 구하고, 구해진 경로를 따라 루트 노드부터 한 단계씩 노드의 MBB를 변경하며 최종적으로 리프 노드에 데이타를 저장한다.

과거 위치 색인에서 차량의 모든 궤적 정보를 하나의 색인 파일로 관리할 경우, 시간의 흐름에 따라 발생하는 방대한 양의 데이타를 저장하여야 하기 때문에 색인 파일의 크기가 커지고 검색성능이 저하되는 문제가 발생한다. 이러한 문제를 해결하기 위하여 제안된 시스템에서는 색인 파일의 다중 버전 기법을 사용하였다. 이동 차량 데이타는 하루 단위로 새로운 페이지 파일에 저장되며, 이동 객체 색인 관리 모듈에서는 하루 단위로 만들어진 이동 객체의 과거 위치 색인 파일을 날짜별로 관리한다. 이때 새롭게 입력되는 데이타가 이틀에 걸쳐서 보고되는 경우 버전 분할에 의하여 두개의 색인 파일에 저장된다. 즉, 2004년 2월 10일~11일에 걸쳐서 보고되는 객체에 대하여 시간 축에 대한 정보를 2004년 2월 10일, 2004년 2월 11일로 분할하여 중복 저장한다. 모바일 클라이언트에서 전송된 질의 요청은 질의 처리 모듈에서 날짜별로 분리되고 이동 객체 색인 관리 모듈에서 해당 날짜의 색인 파일을 검사하여 질의 결과를 반환한다. 이러한 색인 파일의 다중 버전 및 데이타의 버전 분할은 데이타의 중복 저장 및 이틀 이상의 날짜에 걸치는 질의를 처리할 때 색인의 성능을 저하시킨다. 하지만 시간의 흐름에 따라 저장되는 데이타의 크기가 급속히 증가하는 모바일 응용 환경에서 색인 파일의 관리 및 오래된 색인 파일의 이주 정책 설계가 용이해지며, 각각의 날짜에 대한 질의 요청 시 검색하기 위한 색인의 깊이가 낮아지기 때문에 특정 시간 구간의 차량 궤적을 검색할 경우 오히려 질의 처리 성능이 좋아지는 장점을 갖는다.

### 3.3 차량 데이타 수집기

차량 데이타 수집기는 GPS 수신기를 이용하여 차량의 위치 정보를 수집한다. GPS로부터 수신된 위치 정보는 네트워크를 통하여 차량 위치 관리 서버에 전송되거나 데이타 저장기에 저장된다.

GPS 수신기로부터 수신된 차량 정보는 그림 4와 같은 데이타 수집 및 좌표 변환 알고리즘을 통하여 수집되고 TM좌표 형태로 변환된다.

```
Algorithm RecieveNMEA(strNMEA)
입력 => strNMEA : 수신된 차량 정보 String
Begin
  If (수신된 String이 GPGGA를 포함) Then
    time  ←정보 수신 시간 저장
    longd ←Longitude의 Dgree 값 저장
    longm ←Longitude의 Minitue 값 저장
    longs ←Longitude의 Second 값 저장
    latd  ←Latitude의 Dgree 값 저장
    latm  ←Latitude의 Minute 값 저장
    lats  ←Latitude의 Second 값 저장
    transTM (longd, longm, longs, latd, latm, lats)모듈을 호출하여 실행
  End
End
```

그림 4 GPS 데이타 수집 및 좌표 변환 알고리즘

GPS 수신기는 NMEA(National Marine Electronics Association)-0813 프로토콜을 사용하여 위치 정보를 수집한다. NMEA-0813 프로토콜에서는 차량의 위치, 속도 및 기타 정보들을 구분하기 위한 GPGGA, GPGSA, GPGSV, GPRMC등의 헤더를 포함한 데이타들을 수집한다. GPGGA 헤더가 포함된 패킷은 데이타 수신 시간 및 차량의 위치 정보를 가지고 있다. 데이타 수집 모듈에서는 수신된 정보 중 GPGGA 헤더를 포함할 경우 패킷에서 해당하는 정보를 가져와 해당 변수들에 저장하고 이를 좌표 변환 함수로 보낸다. 데이타 수집 알고리즘에서 호출되는 좌표 변환함수는 좌표 변환 공식들을 사용하여 경, 위도 좌표를 TM 좌표로 변환한다.

### 3.4 차량 위치 관리 서버

시스템의 통신을 위하여 우선 차량 위치 관리 서버가 실행되어 접속 대기 상태에 있어야 한다. 차량 데이타 수집기로부터 차량 위치 관련 데이타가 전송될 경우 차량 데이타 수신 및 저장 모듈을 통하여 이동 객체 색인에 차량 정보를 저장한다. 모바일 클라이언트로부터 질의가 입력 된 경우에는 적절한 질의 처리 모듈을 통하여 질의를 처리하고 결과를 다시 반환해준다.

3.4.1 질의 처리 모듈

질의 처리 모듈은 질의를 날짜 별로 분리하고 해당되는 실행 모듈을 호출한다. 알고리즘에 사용되는 색인 파일은 3.2절의 색인 파일 구조를 토대로 한다.

그림 5는 특정 시점에서 특정 공간 위치로부터 사용자가 입력한 범위 안에 위치한 이동 차량의 위치를 검색하는 *TrajDistQuery* 연산자의 실행 알고리즘이다. 우선 입력된 날짜 정보를 이용하여 해당하는 색인 페이지 파일로부터 해당 시점의 모든 차량의 데이타를 검색한다. 이를 위하여 *retrievePageFile*함수에서 차량번호를 널 값으로, 시작 시간과 종료 시간을 동일하게 입력한다. 또한, 특정 시점의 모든 차량의 위치를 검색하여야 하므로 공간 범위는 변수의 최소, 최대값을 입력한다. 차량의 위치가 검색되면 원 방정식을 사용하여 변수로 입력된 중심 표로부터 반경 안에 속하는 차량의 데이타만 검출한다.

그림 6은 특정 시공간 구간에 대한 이동 차량의 위치를 검색하는 *TrajQuery* 연산자의 실행 알고리즘이다. 입력 된 시간 정보 중 날짜 정보만 추출하여 하루이내의 차량 정보를 검색하는지 여부를 비교한다. 하루이내의 차량 정보를 검색하는 경우 해당하는 색인 페이지 파일의 차량 데이타를 검색한다. 여러 날의 차량 정보를 검색하는 경우 질의에 포함된 모든 날짜의 페이지 파일을 통하여 데이타를 검색한다. 또한 색인에 저장하지 않는 시간 정보에 대한 질의는 불확실한 위치 처리 연산 함수를 호출하여 결과를 구한다.

이동 객체의 불확실한 위치 추정은 선형함수에 의한 위치 추정 모델과 곡선을 표현하는 3차 스플라인 함수를 이용하는 방법[7]이 있다. 선형함수를 사용한 방법은 간편하게 구현 가능하며 최소한의 연산으로 위치 추정을 하기 때문에 빠른 연산 처리가 가능한 반면, 스플라인 함수를 이용한 방법은 일반적으로 선형 함수에 의한 위치 추정 결과보다 다소 정확한 값을 얻을 수 있으나 3차함수를 구하기 위한 연산 처리 과정이 다소 복잡하고 처리 속도가 지연되는 단점을 갖는다. 제안된 시스템에서는 과거 위치 추정을 위하여 선형 보간법을 사용하며, 미래 위치 추정은 보다 정확한 추정을 위하여 3차 스플라인 보외법을 사용하였다. 또한 사용자가 불확실

```
Algorithm TrajDistQuery( t, CPX, CPY, Dist)
입력 => t : 검색 시간, CPX : 반경 중심 X 좌표, CPY : 반경 중심 Y 좌표,
        Dist : 원의 반경
출력 => Result : 검색 결과
Begin
   QueryDate ← t 에서 날짜 부분만 분리하여 스트링에 저장
   Filename  ← "tbtree"+ QueryDate
   Result    ← retirevePageFile(Filename, "", t, minDouble,minDouble, t,
               maxDouble, maxDouble) 함수를 호출하여 페이지 파일에서 차량
               정보를 검색하여 저장
   Result ← IsinCircle(Result, CPX, CPY, Dist) 함수를 호출하여 사용자가
            입력한 거리를 반경으로 하는 원안에 포함되는 정보만 검출하여 저장
   return Result
End
```

그림 5 시공간반경 연산 처리 알고리즘

```
Algorithm TrajQuery(CarID, St, SPX, SPY, Et, EPX, EPY)
입력 => CarID : 차량 번호, St : 유효 시작 시간, Et : 유효 종료 시간,
 SPX : 유효 시작 X 좌표, SPY : 유효 시작 Y 좌표,
 EPX : 유효 종료 X 좌표, EPY : 유효 종료 Y 좌표
출력 => Result : 검색 결과
Begin
   SQueryDate ← St에서 날짜 부분만 분리하여 스트링에 저장
   EQueryDate ← Et에서 날짜 부분만 분리하여 스트링에 저장
   Filename ← "tbtree"+ QueryDate
   If (SQueryDate == EQueryDate)
      Result = retirevePageFile(Filename, CarID, St, SPX, SPY, Et,
      EPX, EPY) 함수를 호출하여 페이지 파일에서 차량 정보 검색한 후 저장
   Else
      While (SQueryDate < EQueryDate)
        Result = retirevePageFile(Filename, CarID, St, SPX, SPY, Et,
        EPX, EPY) 함수를 호출하여 페이지 파일에서 차량 정보 검색한 후 저장
        SQueryDate를 하루씩 증가
      End
   End
   If (Result==null) UncertainFuture(CarID, Et)를 사용한 미래 위치 추정
   return Result
End
```

그림 6 연산 처리 알고리즘

영역의 크기를 제한하도록 함으로써 추정된 불확실한 위치의 오차가 너무 커지지 않도록 하였다. 즉, 불확실한 위치를 추정하기 위한 계산된 위치가 사용자가 정의한 불확실 영역의 범위를 벗어날 경우 검색이 실패하도록 한다.

3.4.2 이동 객체 색인 관리 모듈

차량 데이타 수신 및 변환 모듈에서 전송된 차량 정보는 이동 객체 색인 관리 모듈의 그림 7과 같은 페이지 파일 생성 알고리즘을 통하여 색인 페이지 파일에 저장된다. 또한 차량 정보 검색 시에는 그림 8과 같은 페이지 파일 검색 알고리즘을 통하여 해당되는 색인 파일에서 차량 정보를 검색한다.

차량 데이타 변환 모듈에서 변환된 차량 정보를 입력받아 이 데이타중 시간 정보를 사용하여 오늘 날짜 정보를 추출한다. 추출된 날짜에 해당하는 색인 페이지 파일이 존재하지 않을 경우에는 새로운 페이지 파일과 MBB를 생성하고 데이타 삽입 함수로 전달되어 트리를 구성한다. 기존에 페이지 파일이 존재할 경우에는 존재하는 페이지 파일에 차량 정보를 삽입한다.

또한 질의 처리 모듈에서 생성된 검색 페이지 파일명과 질의 정보는 그림 8의 페이지 파일 검색 함수로 전달된다. 페이지 파일 검색 함수에서는 검색이 필요한 페이지 파일을 호출하고 전달된 질의 정보 중 좌표 질의 정보들을 사용하여 MBB를 구성한다. 그리고 호출된 페이지 파일에서 조건에 해당하는 차량 정보를 검색하기 위하여 색인 클래스의 영역 검색 함수에 차량 번호와 질의 MBB를 전달하고 검색된 결과를 질의 처리 모듈로 넘겨준다.

그림 9는 동시성 제어 기법을 적용하여 수정한 TB-Tree의 삽입 알고리즘이다.

### 3.5 모바일 클라이언트

구현된 차량 위치 추적 시스템의 작동을 위하여 우선

```
Algorithm TrajQuery(CarID, St, SPX, SPY, Et, EPX, EPY)
입력 => CarID : 차량 번호, St : 유효 시작 시간, Et : 유효 종료 시간,
 SPX : 유효 시작 X 좌표, SPY : 유효 시작 Y 좌표,
 EPX : 유효 종료 X 좌표, EPY : 유효 종료 Y 좌표
출력 => Result : 검색 결과
Begin
   SQueryDate ← St에서 날짜 부분만 분리하여 스트링에 저장
   EQueryDate ← Et에서 날짜 부분만 분리하여 스트링에 저장
   Filename ← "tbtree"+ QueryDate
   If (SQueryDate == EQueryDate)
      Result = retirevePageFile(Filename, CarID, St, SPX, SPY, Et,
      EPX, EPY) 함수를 호출하여 페이지 파일에서 차량 정보 검색한 후 저장
   Else
      While (SQueryDate < EQueryDate)
        Result = retirevePageFile(Filename, CarID, St, SPX, SPY, Et,
        EPX, EPY) 함수를 호출하여 페이지 파일에서 차량 정보 검색한 후 저장
        SQueryDate를 하루씩 증가
      End
   End
   If (Result==null) UncertainFuture(CarID, Et)를 사용한 미래 위치 추정
   return Result
End
```

그림 7 페이지 파일 생성 알고리즘

```
Algorithm retrievePageFile(FileName, CarID, Stime, Etime, SPosX, SPosY,
EPosX, EPosY)
입력 => FileName : 페이지 파일 명, CarID : 차량 번호,
Stime : 유효 시작 시간, Etime : 유효 종료 시간,
SPosX : 유효 시작 X 좌표, SPosY : 유효 시작 Y 좌표,
EPosX : 유효 종료 X 좌표, EPosY : 유효 종료 Y 좌표
출력 => resultVector : 페이지 파일에서 검색된 차량 정보들의 집합
Begin
   min MBB[0] = SPosX;  min MBB[1] = SPosY;  min MBB[2] = Stime;
   max MBB[0] = EPosX;  max MBB[1] = EPosY;  max MBB[2] = Etime;
   색인 클래스의 TBtree(FileName)생성자를 사용하여 검색이 필요한 날짜의
   페이지파일 호출
   resultVector = RangeOverapQuery(CarID, minP, maxP)함수에서 검색된
    차량 좌표들을 결과 벡터에 저장
   return resultVector
End
```

그림 8 페이지 파일 검색 알고리즘

```
Algorithm Insert(CarID, St, SPX, SPY, Et, EPX, EPY)
입력 => CarID : 차량 번호, St : 유효 시작 시간, Et : 유효 종료 시간,
 SPX : 유효 시작 X 좌표, SPY : 유효 시작 Y 좌표,
 EPX : 유효 종료 X 좌표, EPY : 유효 종료 Y 좌표
Begin
   MakePath();   ← 새로운 데이터가 삽입될 leaf node까지의 경로 생성
                   리프노드에 데이터 삽입을 위한 공간이 없을 경우, 새로운
                   리프 노드를 생성하여 그 경로를 반환
   Node = Root;
   While (Node is not Leaf)
      Obtain a shared lock.
      If MBB need to be modified then
         Release a shared lock
         Reobtain the exclusive lock
         Update MBB;
      End
      Release the lock on the node;
      Node = Node.Child;
   End
   Obtain a exclusive lock ← 리프 노드에 새로운 데이터 삽입
   Add new data into leaf node;
   Release the lock on the leaf node;
End
```

그림 9 확장된 TB-Tree 삽입 알고리즘

```
Algorithm TransSendData(Tname, CarID, Ts, Te)
입력 => Tname : 테이블 명, CarID : 차량 번호
        Ts : 유효 시작 시간, Te : 유효 종료 시간
Begin
    (String)Packet 변수에 Tname값과 구분자 할당
    (String)Packet 변수에 CarID값과 구분자 추가 할당
    (String)Packet 변수에  Ts값과 구분자 및 Te값과 구분자 추가 할당
    (String)Packet 변수에  질의 타입인 Type값을 할당
    TransferPacket(Packet) 함수를 호출하여 변환된 데이터 전송
end
Algorithm TransRecieveData(Packet)
입력 => Packet : 서버에서 전송된 질의 결과 패킷 데이터
Begin
  for (Packet의 길이)
    if (Packet이 구분자인 경우)
      ListBox에 출력하지 않고 통과
    Else
      ListBox에 식별자를 붙여 출력.
End
```

그림 10 데이타 변환 알고리즘

모바일 클라이언트와 서버가 소켓 통신이 가능해져야 한다. 서버가 작동중일 경우 모바일 클라이언트의 서버 연결 모듈을 실행하면 접속 대기 중인 서버에 연결되어 TCP/IP를 사용한 소켓 통신이 가능해진다. 그 후 사용자가 질의 입력 모듈을 사용하여 질의를 입력하면 입력된 질의 데이타는 데이타 변환 모듈에서 패킷 데이타로 변환되고 서버로 전달된다. 서버에서 처리된 질의 결과는 모바일 클라이언트의 데이타 수신 모듈에 패킷 형태로 수신된다. 데이타 변환 모듈은 수신된 결과 데이타를 결과 출력 모듈 화면에 출력 가능한 형태로 변환하고 사용자는 변환된 질의 결과를 텍스트 형태로 확인할 수 있다.

데이타 변환 모듈은 모바일 클라이언트의 사용자 인터페이스에서 입력된 질의 데이타를 차량 위치 추적 시스템 서버로 전송하기 위한 패킷 형태의 데이타로 변환한다. 그리고 서버에서 전송 받은 결과 패킷 데이타를 사용자 화면에 출력하기 위한 데이타 형태로 변환하는 기능을 수행한다. 그림 10은 데이타 입력 모듈에서 전송된 데이타들을 서버의 데이타 송수신 모듈에서 수신하기 알맞은 형태로 변환하는 알고리즘과 서버에서 전달된 결과 패킷 데이타를 사용자 인터페이스화면에 출력하기 위한 형태로 변환하는 알고리즘을 보여준다.

*TransSendData()* 함수는 우선 데이타 입력 모듈로부터 각 *Tname*, *CarID*, *Ts*, *Te* 데이타를 전달받고

*Type* 데이타와 결합하여 패킷 형태로 변환한다. *Type* 데이타는 서버에서 처리될 각 질의 종류를 구분하기 위한 질의 타입이다. 또한 이러한 데이타들은 서버에서 다시 분리되어야 하기 때문에 각 데이타 사이에 구분자를 추가한다. *TransRecieveData*() 함수는 서버에서 전송된 패킷 형태의 질의 결과 데이타를 모바일 클라이언트 화면에 출력하기 위한 형태로 변환한다. 서버에서 전달된 데이타는 구분자를 중심으로 분리되고 각 데이타별 식별자를 붙여 화면에 출력된다.

## 4. 구현 및 평가

이 장에서는 제안된 시스템의 구현 결과, 색인의 성능 및 시스템에서 지원하는 기능을 중심으로 기존의 시스템과의 비교 분석을 수행한다.

### 4.1 구현 환경

차량 관리 시스템은 Pocket PC(EVC) 및 JDK 1.3(Java)을 사용하여 구현하였으며 DBMS로는 Microsoft SQL Server 2000를 사용하였다. 차량 데이타 수집기 및 모바일 인터페이스는 Microsoft Windows CE 2002(Pocket PC) 운영체제를 기반으로 하는 PDA를 사용하였다. 또한 차량 데이타 수집기 및 모바일 클라이언트는 Microsoft Embedded Visual C++(EVC) 4.0을 사용하여 구현하였다. 차량 위치 검색 서버는 플랫폼에 독립적으로 운영될 수 있는 JDK에서 사용될 수 있도록 JAVA를 사용하여 구현하였고, 테스트는 Windows 2000 Server 운영체제, JDK 1.3 환경에서 실시하였다.

### 4.2 차량 정보 검색

차량 정보 검색은 차량 위치 관리 서버와 모바일 클라이언트에서 모두 가능하다. 차량 위치 관리 서버는 모바일 클라이언트와 별도로 질의처리 관련 모든 기능을 갖는다. 그림 11은 시공간구간질의 결과를 보여준다. 질의 처리의 예로 '2003년 2월 6일 10시 12분에서 2003년 2월 7일 17시 45분 사이에 천호시장에서 명일여자고등학교 사이를 이동한 차량 서울 81바 3578 차량의 위치를 검색하라.'와 같은 질의를 요청할 경우, 먼저 서버 프로그램의 사용자 인터페이스를 통하여 차량번호, 유효 시작 시간, 유효 종료 시간, 그리고 공간 범위를 입력한다. 입력된 질의 데이타는 서버 프로그램의 질의 처리 모듈에 전달되어 해당되는 날짜의 색인 페이지 파일에서 차량 정보를 검색한다. 검색된 차량 정보는 그림 11과 같이 텍스트 및 그래픽 형태로 출력된다. 화면 왼쪽은 차량 번호, 시간 정보, 차량의 위치 정보 등을 텍스트 형태로 출력하고, 오른쪽 부분은 지정된 차량의 시간별 이동 궤적을 그래픽 형태로 출력한다.

다음은 시공간반경질의 결과이다. 질의는 '2003년 2월 6일 16시 55분에 강동 성모병원의 700미터 반경 안에

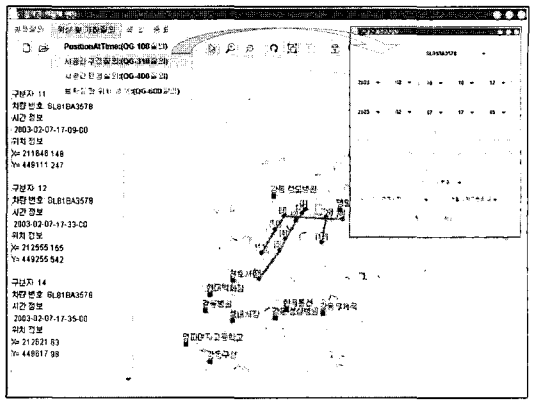


그림 11 시공간구간질의 결과

위치한 차량을 검색하라.'이고, 그림 12에서 처리 결과를 보여준다. 시공간반경질의 또한 서버 프로그램의 질의 입력 인터페이스를 사용하여 질의시점과 질의 중심 점 그리고 반경 값을 입력한다. 입력된 데이타는 프로그램의 질의처리모듈과 색인 관리 모듈을 통하여 사용자 인터페이스에 출력된다.

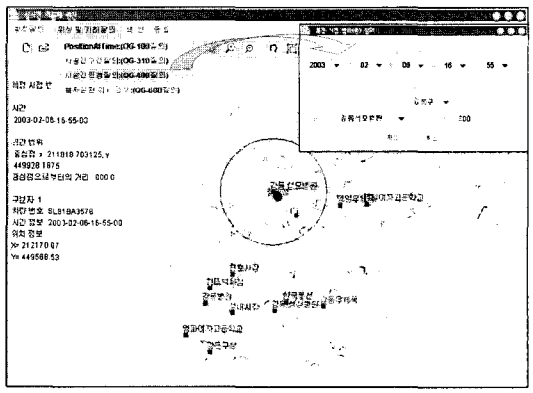


그림 12 시공간반경질의 결과 화면

다음 질의는 불확실한 위치 검색 질의이다. 사용된 질의는 '2003년 2월 7일 16시 33분에 서울 81바 3578 차량의 위치를 검색하라.'이고 그림 13에서 연산 결과를 보여준다. 질의하고자 하는 차량 및 미래 시간 구간을 지정하고 불확실 영역의 크기를 입력하면 그림과 같은 결과 화면을 얻을 수 있다.

모바일 클라이언트에서 차량 위치 검색을 실시하기 위해서는 우선 차량 위치 관리 서버를 실행시킨다. 그리고 모바일 클라이언트를 실행시키고 서버에 접속한다. 모바일 클라이언트는 질의 메뉴를 통하여 질의를 입력하고 차량 위치 관리 서버에 전송한다. 서버에서 수행된 질의 결과는 클라이언트의 리스트 박스에서 텍스트 형

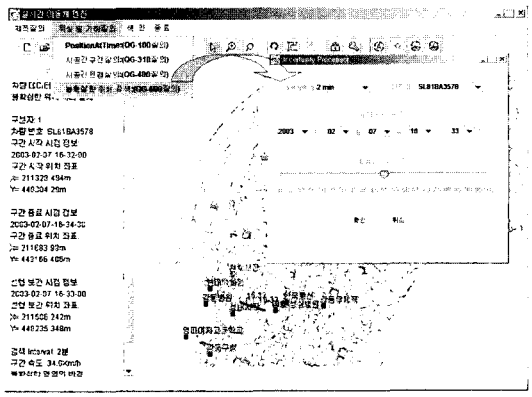


그림 13 불확실한 위치 검색 결과 화면

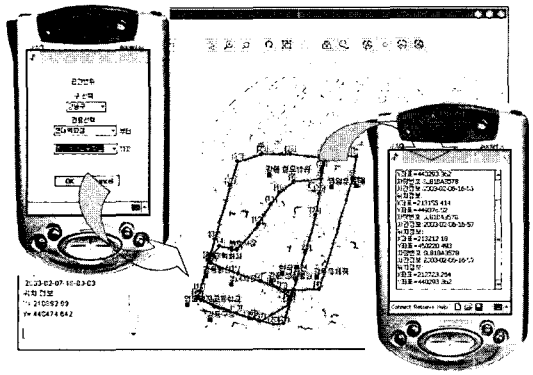


그림 14 특정공간구간질의 결과 화면

태로 볼 수 있다. 그림 14는 '강동구 현대 백화점과 명일 여자고등학교 사이를 이동한 모든 차량을 검색하라'는 특정공간구간의 질의 수행 결과를 보여준다. 모바일 클라이언트의 질의 메뉴에서 질의를 선택하고 차량 번호와 질의 시점을 입력하면 서버로 질의 데이타가 전송된다. 질의 결과는 차량 위치 관리 서버에서는 텍스트 형태 및 궤적 형태로 확인 할 수 있고 모바일 클라이언트에서는 텍스트 형태로 확인할 수 있다.

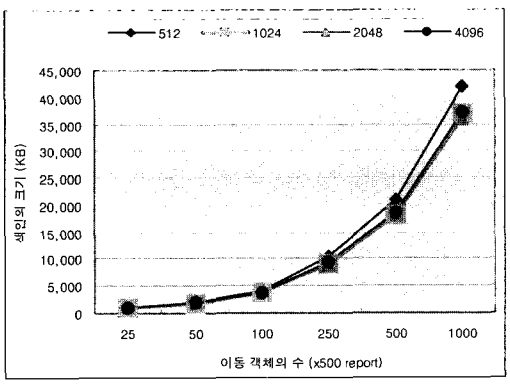
### 4.3 색인의 성능 평가

4.3.1 실험 데이타

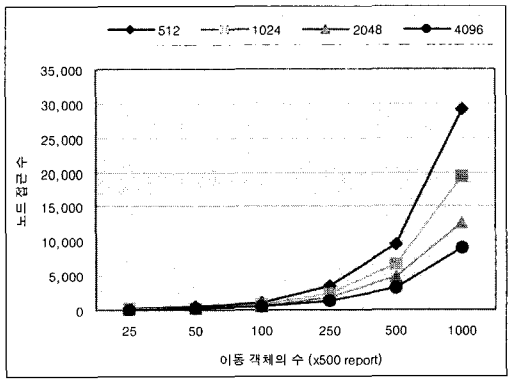
제안된 색인의 성능을 평가하기 위하여 CitySimulator[15]에 의해 생성된 데이타를 이용하였다. CitySimulator는 (x, y, z)의 3차원 위치를 모델링하여 최대 100만 명까지의 모바일 사용자들의 행동 패턴을 모의 실험할 수 있도록 가상의 동적인 데이타를 생성해주는 프로그램이다. 이 프로그램은 3차원 공간상에서 사람들이 도로 상에서의 이동, 빌딩간의 이동, 빌딩의 각 층간의 이동을 표현하는 등 실세계의 모바일 환경을 Simulation할 수 있도록 설계되었다. 실험에 사용된 데이타는 25, 50, 100, 250, 500, 1000개의 객체가 이동하면서 각각의 객체가 500번씩 객체의 위치를 Report하도록 설정하여 실험 데이타를 생성하였다.

4.3.2 색인 생성 시간 및 크기

그림 15는 제안된 시스템에서 사용된 색인의 성능을 나타낸다. 그림 15(a)는 데이타 크기에 따른 색인의 크기 변화를 나타낸다. 색인 파일의 각 노드의 크기에 따른 색인의 크기 변화를 보여준다. 또한 그림 15(b)는 색인의 데이타 삽입 연산 성능을 보여준다. 실험 결과 노드의 크기를 크게 함에 따라 데이타 삽입에 따른 노드의 접근 횟수가 줄어드는 것을 알 수 있다. 이것은 노드의 크기가 커짐에 따라 생성되는 노드의 수가 줄어들고, 상대적으로 색인을 구성하는 트리의 높이가 낮아지기 때문이다. 트리의 높이가 낮아짐에 따라 새로운 데이타를 삽입할 때의 삽입할 데이타가 저장될 리프 노드를 검색하기 위한 비용이 적어지기 때문에 노드의 접근 수가 줄어든다.




(a) 색인 파일의 크기




(b) 데이타 삽입

그림 15 색인 파일의 크기 및 데이타 삽입 성능

4.3.3 질의 처리

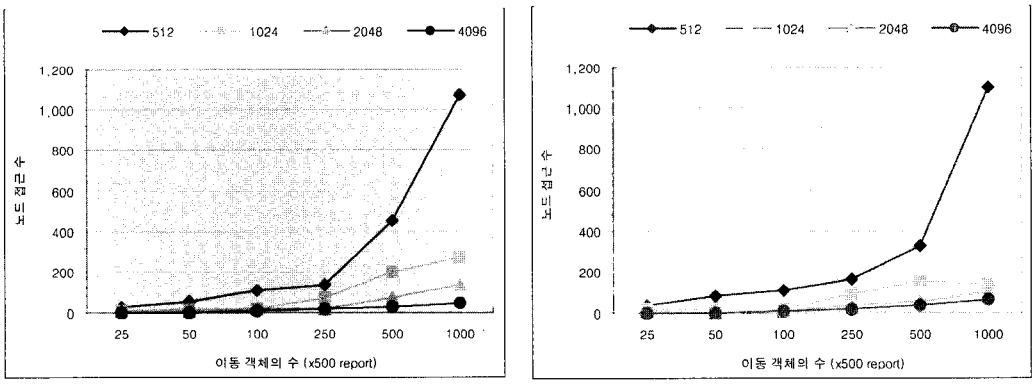
제안된 시스템에서 사용된 색인의 질의 처리 성능이 그림 16에 보여준다. 질의에 사용된 데이타는 전체 실험 데이타 도메인의 10% 범위를 질의 요청 영역으로 설정하여 실험하였다. 또한 색인의 각 노드를 위한 페이지 크기를 각각 512Byte, 1KB, 2KB, 4KB로 설정하여 비교 실험하였다.

시간 구간 질의는 특정 시간 구간에서의 이동 객체 검색을 위한 공간 질의를 나타내는 질의로서, 그림 16(a)는 특정 시간 구간 질의 처리 결과를 보여준다. 또한 그림 16(b)는 시공간 질의 성능을 보여준다. 그림에서와 보는바와 같이 페이지 크기가 커짐에 따라 질의 성능은 좋아지는데, 이는 페이지 크기가 커지면서 색인을 구성하는 트리의 깊이가 낮아지기 때문에 질의 처리 시에 방문할 노드의 수가 줄어들기 때문이다.

4.3.4 동시성 제어

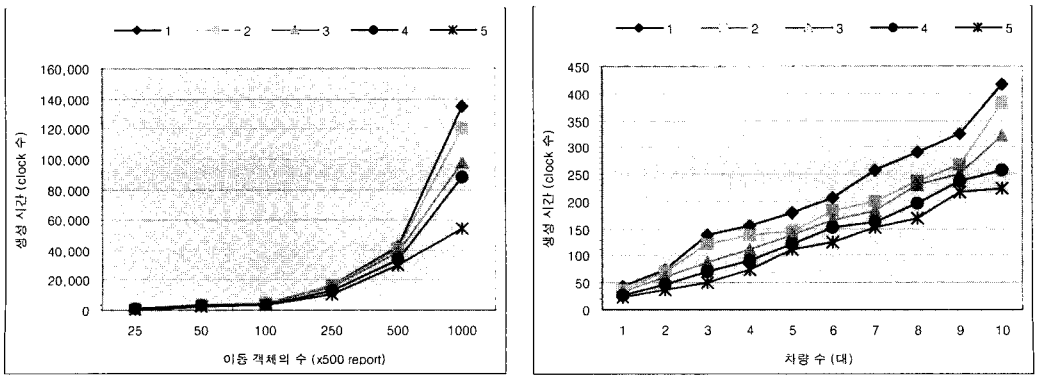
이 논문에서 사용된 색인은 모바일 환경의 다수 사용자 접근 환경에 적응하도록 기존의 TB-Tree를 동시성 제어가 가능하도록 확장하였다. 색인의 동시성 제어 성능을 평가하기 위하여 1~5개의 프로세서에 의해 데이타를 색인에 삽입하면서 색인의 성능을 비교하였다. TB-Tree의 경우 새로운 데이타의 삽입 과정은 먼저 새로운 데이타가 저장된 리프 노드를 검색하여 검색된 노드에 새로운 데이타를 삽입한다. 따라서 데이타 삽입 과정에 리프 노드 검색을 위한 질의 처리 과정이 내포되어 있기 때문에 데이타의 삽입 성능으로 색인의 동시성 제어 능력을 실험하였다.

그림 17은 색인의 동시성 제어 성능 실험 결과를 보여준다. 그림 17(a)는 City Simulator에 의해 생성된 데이타에 의한 실험 결과이고, 그림 17(b)는 실세계에서 얻은 데이타를 이용하여 실험하였다. 실세계 데이타는 10대의 차량에 대하여 1분 간격으로 그 위치를 샘플링하였고, 각각의 차량에 대하여 5시간 동안의 차량 위치를 축적하여 실험 데이타를 구축하였다. 그림 17에서 보는 바와 같이 제안된 색인은 다중 사용자 환경에 좋은 성능을 보임을 알 수 있다.




(a) 특정 시간 구간 질의 성능 (b) 특정 시공간 구간 질의 성능

그림 16 색인의 범위 질의 성능




(a) City Simulator 데이타 (b) 실세계 데이타

그림 17 동시성을 고려한 색인 생성 시간

4.3.5 날짜별 색인의 분할 생성

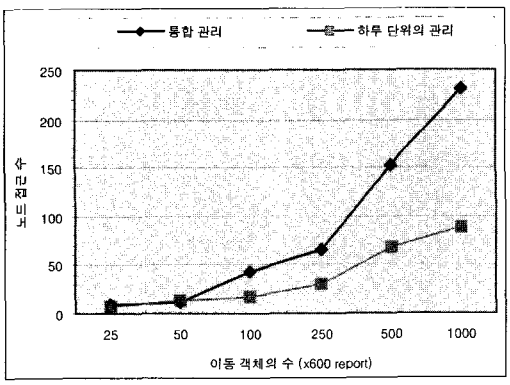
3.2절에서 언급한 바와 같이 모든 차량 데이타를 하나의 파일로 관리할 경우, 시간이 지남에 따라 색인 파일의 크기가 너무 커져 시스템 내에서 관리하기 어려운 문제가 발생한다. 따라서 제안된 시스템은 하루단위로 색인을 생성하여 관리하는 다중 버전 기법을 사용하였다. 이 방법의 성능을 측정하기 위하여 차량이 하루에 200회씩, 3일에 걸쳐 위치 정보를 갱신하는 것을 가정하여 실험 데이타를 생성하였다.

그림 18와 그림 19는 차량 데이타를 하나의 색인에 통합 관리하였을 경우와 하루 단위로 분할하여 관리하였을 경우의 성능을 보여준다. 그림 18처럼 하루 단위의 색인 관리 방법은 색인할 데이타 량이 적고 통합 관리 방법보다 tree의 높이가 낮기 때문에 하루 이내의 질의에서 성능이 향상된다. 반면 이틀에 걸친 질의에서는 하루 단위 색인 관리 방법이 tree의 깊이는 낮지만 두 개의 파일을 검색하여야 하기 때문에 질의 성능이 낮아진다. 또한 그림 19(b)와 같이 궤적 질의에서도 하루 단위의 궤적 관리 방법은 날짜가 바뀔 때마다 궤적 선택을 위한 범위 질의를 수행하여야 하기 때문에 성능이 저하된다. 하지만 그림 19(a)처럼 하루 단위 색인 관리 방법은 하나의 파일에 색인되는 데이타의 량이 적기 때문에 색인의 삽입 성능은 향상된다.

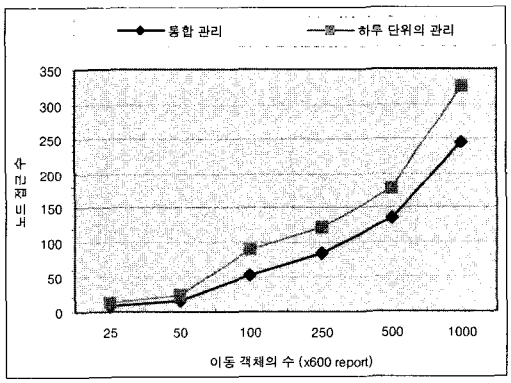
### 4.4 시스템 처리 능력 비교

기존의 차량 위치 추적 시스템과 이동 객체 관련 연구의 프로토타입들에서는 이동 차량의 위치를 추적하고 관리하기 위한 여러 가지 방법들이 제시되었다. 표 1은 각 시스템에서 지원하는 기능 및 데이타 처리 방법을 기준으로 기존의 시스템들과 제안된 시스템을 비교하였다.

기존 시스템과의 데이타 처리 및 세부적인 성능 비교가 어렵기 때문에 이 논문에서는 각 시스템이 지원하는 기능 및 데이타 관리 방법을 중심으로 각 시스템을 비교 평가한다. 표 1에서처럼 기존의 차량 위치 추적 시스템은 차량의 현재 위치를 데이타베이스에 저장, 관리하고 기존의 DBMS에서 지원되는 공간 색인을 사용한다. 따라서 연속적으로 변화하는 차량의 위치 정보를 효과

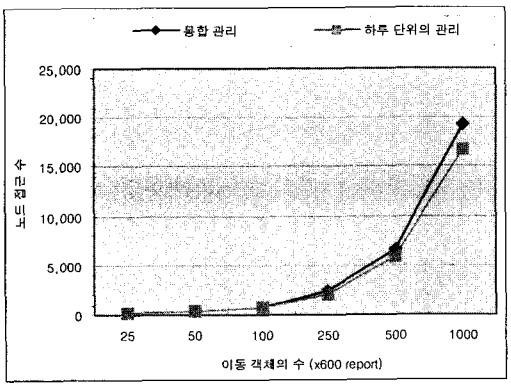



(a) 하루 이내에서의 시공간 질의

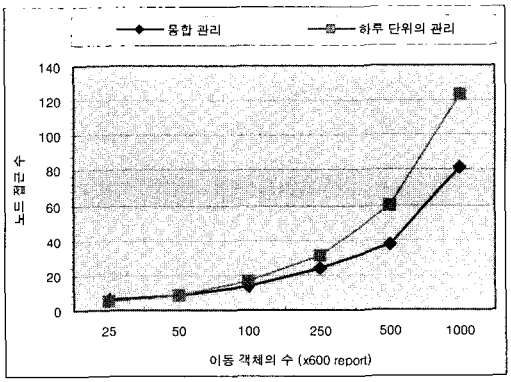



(b) 이틀에 걸친 시공간 질의

그림 18 색인의 범위 질의 성능




(a) 데이타 삽입 성능




(b) 이틀에 걸친 궤적 질의

그림 19 데이타 삽입 및 궤적 질의 성능

표 1 각 시스템의 이동 차량 데이타 관리 방법 비교

| 기능 및 관리방법 / 시스템 | 데이타 종류 | | 질의 처리 종류 | | | 질의 처리 시점 | | | 색인 지원 |
|---|---|---|---|---|---|---|---|---|---|
| | 이동 점 | 이동 영역 | 공간 질의 | 시간 질의 | 시공간 질의 | 과거 | 현재 | 미래 | |
| APTS[5] | ○ | X | △ | △ | X | △ | ○ | X | 공간 색인 |
| DOMINO | ○ | X | ○ | ○ | ○ | X | ○ | ○ | 사용 안함 |
| CHOROCHRO NOS | ○ | ○ | ○ | ○ | ○ | ○ | △ | X | 이동객체 색인 |
| 제안된 시스템 | ○ | X | ○ | ○ | ○ | ○ | ○ | ○ | 이동객체 색인 |

○: 처리가능 △: 조건적으로 처리 가능 X: 처리하기 어려움

적으로 관리하지 못하며, 모바일 환경에서 사용자가 빈번하게 요청하는 시공간 연산을 효과적으로 처리하지 못하는 문제가 있다.

DOMINO 프로젝트에서 개발된 프로토타입은 MOST (Moving Object Spatio-Temporal) 모델을 사용하여 이동 점 객체만을 대상으로 현재로부터 가까운 미래 시점까지의 질의만을 다루기 때문에 이동 영역 및 이동 객체의 모든 과거 정보를 다루지 못하는 단점이 있다. 한편 CHOROCHRONOS에서 제시한 시공간 데이타 모델은 데이타베이스에 저장된 이동 차량의 과거 이력 정보만 다루기 때문에 현재 또는 가까운 미래 시점에서의 질의 처리를 제공하지 못한다. 또한 CHOROCHRONOS 프로젝트의 경우 시공간 데이타 모델과 이동 객체에 관한 기초 연구로서만 연구가 진행되어 왔기 때문에 프로토타입으로 개발된 사례가 없다.

따라서 이 논문에서는 이동 객체 색인을 사용하여 실시간으로 제공되는 이동 차량의 위치 정보를 관리하고 이동 객체 관련 연산자들을 사용하여 이동 차량의 위치와 관련된 다양한 질의를 처리할 수 있는 시스템을 제안하였다. 지금까지 프로토타입 형태로 개발된 시스템의 경우 이동 객체의 위치 정보 관리를 위한 색인 기법을 적용한 사례는 없으며, 기존의 DBMS에서 제공하는 테이블 형태의 데이타 관리 방법을 이용하거나 공간 색인을 이용하여 처리하고 있다. 제안된 시스템은 또한 불확실한 위치 처리 연산 함수를 사용하여 이동 객체 색인 파일에 저장되지 않은 과거, 현재 및 가까운 미래 시점에서의 차량 위치 검색이 가능하다.

## 5. 결 론

차량 위치 수신 기술의 발달로 인하여 이동 차량의 위치를 제공해주는 차량 위치 검색 시스템 개발이 활발하게 진행되어 왔다. 그러나 기존의 차량 위치 검색 시스템은 차량의 이동 궤적을 효과적으로 관리 및 검색하는 방법을 제시하지 못하고 있다. 또한 주로 유선 네트워크 상에서 차량의 위치 검색 서비스를 제공하기 때문에 무선 통신을 이용하는 모바일 클라이언트에서의 실시간 위치 검색 기능을 수행할 수 없다. 따라서 이 논문에서는 이동 객체 색인을 이용하여 차량의 위치를 효과적으로 관리하고 무선 네트워크 상에서 모바일 클라이언트를 이용하여 실시간으로 차량의 위치를 검색할 수 있는 차량 관리 시스템을 설계 및 구현하였다. 제안 시스템은 차량 데이타 수집기, 차량 위치 관리 서버와 모바일 클라이언트로 구성된다. 차량 데이타 수집기는 차량에 탑재된 GPS 수신기로부터 차량의 위치 정보를 수신 받아 TM좌표 형태로 변환한 후 무선 네트워크를 통하여 차량 관리 서버에 전송한다. 차량 위치 관리 서버는 차량의 위치 정보를 색인 페이지 파일에 저장, 관리하고 모바일 클라이언트에 차량 위치 검색 결과를 제공하는 기능을 한다. 모바일 클라이언트는 사용자가 실시간으로 차량 위치 관련 정보를 서버에 요청하고 서버로부터 제공된 위치 검색 결과를 사용자에게 제공한다.

이러한 차량 관리 시스템 구현을 위하여 이동 차량 데이타의 효과적인 관리 방법과 시스템 구성 및 처리 알고리즘을 제시하였다. 구현 시스템은 대량의 차량 데이타를 관리하기 위하여 이동 객체 색인 관리 모듈을 사용하며 차량 위치 정보는 날짜 별로 관리 된 이동 객체 색인을 통하여 검색한다. 구현 시스템에서 제공하는 차량 위치 검색 기능은 특정 시간 구간뿐만 아니라 주변 공간 객체들과의 공간 위상 질의까지도 포함하며 이 논문에서는 이 질의들을 만족하는 차량 위치 검색 기능이 잘 수행됨을 확인하였다. 아울러 제안된 시스템의 실행 모델 및 가상 시나리오를 적용한 구현 결과를 보임으로써 모바일 환경에서 이동 클라이언트들의 차량 위치 검색 질의를 실시간으로 처리할 수 있음을 보였다.

향후 연구로는 제안된 시스템의 이동 객체 색인의 갱신 및 검색 성능의 개선을 위하여 궤적 정보를 효과적으로 처리함과 동시에 공간 구별력을 높여 범위 질의와 같은 공간 질의도 빠르게 처리할 수 있도록 하는 연구가 현재 진행 중이며, 모바일 환경의 빈번한 위치 갱신 요청을 빠르게 처리하기 위한 방법도 개발 중에 있다. 또한, 모바일 클라이언트의 차량 위치 추적 결과 출력 모듈에 전자 지도를 추가적으로 구성할 수 있도록 시스템을 확장하는 연구를 진행할 것이다.

## 참 고 문 헌


[1] 김종혁, "첨단 교통관리 시스템", 정보과학회지, 제16권 6호, 1998년, pp. 5-13.

[2] 이승룡, 홍영래, 김형일, 배수강, 최대순, "첨단 대중교통 시스템", 정보과학회지, 제16권 6호, 1998년, pp. 23-29.

[3] Federal Transit Administration, "Advanced Public Transportation Systems : The State of the Art Update '96," U.S. Department of Transportation FTA-MA-26-7007-96-1, January 1996.

[4] 교통개발연구원, 한국통신, "첨단 화물운송시스템(CVO) 기본설계", 1997.

[5] 안승범, "안전도향상을 위한 첨단 화물운송 시스템(CVO)의 서비스와 기술", 정보과학회지, 제16권 6호, 1998년, pp. 30-35.

[6] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang, "Moving Objects Databases : Issues and Solutions," Proc. of the 10th International Conference on Scientific and Statistical Database Management (SSDBM'98), Capri, Italy, 1998.

[7] 안윤애, 류근호, 김동호, "차량 위치 추적을 위한 이동객체 관리 시스템의 설계", 한국정보처리학회 논문지, 2002년 10월.

[8] P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, "Modeling and Querying Moving Objects," Proc. of the 13th Int'l Conf. on Data Engineering(ICDE'97), Birmingham, UK, Apr. 1997.

[9] O. Wolfson, P. Sistla, B. Xu, J. Zhou, S. Chamberlain, N. Rishe, and Y. Yesha, "Tracking Moving Objects Using Database Technology in DOMINO," Proc. of the 4th Workshop on Next Generation Information Technologies and Systems, Zikhron-Yaakov, Israel, 1999.

[10] S. Saltenis, C. S. Jensen, S. Leutenegger, and M. Lopez, "Indexing the Positions of Continuously Moving Objects," Proc. of the ACM SIGMOD Conf., 2000.

[11] D. Pfoser, C. S. Jensen, and Y. Theodoridis, "Novel Approaches in Query Processing for Moving Objects," Proc. of the Int'l Conf. on VLDB, 2000.

[12] K. H. Ryu, and Y. A. Ahn, "Application of Moving Objects and Spatiotemporal Reasoning," TimeCenter TR-58, 2001.

[13] N. Beckmann, H. P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree : An Efficient and Robust Access Method for Points and Rectangles," Proc. of the ACM SIGMOD Conf., Atlantic City, New Jersey, USA, May 1990.

[14] K. V. Ravi Kanth, D. Serena, and A.K. Singh, "Improved Concurrency Control Techniques for Multi-dimensional Index Structures," Proc. of the 1st Merged Int'l and Symposium on Parallel and Distributed Processing (IPPS/SPDP), 1998.

[15] http://www.alphaworks.ibm.com/tech/citysimulator, 2001.



이 응 재

1994년 충북대학교 컴퓨터과학과(이학사) 1996년 충북대학교 대학원 전자계산학과(이학석사). 2001년~현재 충북대학교 대학원 전자계산학과 박사과정. 관심분야는 시공간 데이타베이스, 이동객체 데이타베이스, LBS/텔레매틱스, 지리정보 시스템



오 준 석

2002년 한성대학교 정보공학과 졸업(공학사). 2004년 충북대학교 대학원 전자계산학과 졸업(이학석사). 관심분야는Mobile database, Mobile GIS, LBS, ITS



정 영 진

2000년 충북대학교 전자계산학과 졸업(이학사). 2002년 충북대학교 대학원 전자계산학과 졸업(이학석사). 2003년~현재 충북대학교 대학원 전자계산학과 박사과정. 관심분야는 이동객체 데이타베이스, 이동객체 색인, 지리정보 시스템, 바이오인포매틱스, 바이오 색인



남 광 우

1995년 충북대학교 전자계산학과 졸업 1997년 충북대학교 대학원 전자계산학과(이학석사). 2001년 충북대학교 대학원 전자계산학과(이학박사). 2001년~2004년 한국전자통신연구원 LBS연구팀 선임연구원. 2004년~현재 군산대학교 컴퓨터정보과학과 전임강사. 관심분야는 LBS/텔레매틱스, 시공간 및 메인메모리 데이타베이스, 데이타마이닝

이 봉 규

1988년 연세대학교 졸업(학사). 1992년 Cornell University Dept. of CRP 졸업(석사). 1994년 Cornell University Dept. of CRP 졸업(박사). 1993년~1997년 Cornell University Dept. of CRP 조교수. 1997년~현재 한성대학교 정보전산학부 부교수. 관심분야는 Mobile Computing, XML, GML, GIS, ITS 등

류 근 호

1976년 숭실대학교 전산학과(이학사). 1980년 연세대학교 대학원 전산전공(공학석사). 1988년 연세대학교 대학원 전산전공(공학박사). 1976년~1986년 육군군수 지원사 전산실(ROTC 장교), 한국전자통신연구원(연구원), 한국 방송대 전산학과(조교수) 근무. 1989년~1991년 Univ. of Arizona Research Staff(TempIS 연구원, Temporal DB). 1986년~현재 충북대학교 전기전자컴퓨터공학부 교수. 관심분야는시간 데이타베이스, 시공간 데이타베이스, Temporal GIS, 객체 및 지식베이스 시스템, 지식기반 정보검색 시스템, 데이타마이닝, 데이타베이스 보안 및 바이오인포매틱스