論文
4-12-12

위치기반서비스를 위한 이동 객체 관리 시스템
A Moving Object Management System for Location Based Service

안윤애(Yoon-Ae Ahn)[1)]

요 약

이동 객체 관리 시스템은 사람, 동물, 자동차, 휴대용 단말기 등과 같이 시간에 따라 연속적으로 위치를 변경하는 시공간 데이터를 관리하며, 차량 추적 시스템, 디지털 전장, 동물 서식지 관리 등과 같은 위치기반서비스에 적용된다. 기존의 이동 객체 관리 시스템은 이동 객체의 불확실한 위치 추정 기능을 제공하지 못하며, 실시간 환경에서 발생되는 위치 정보의 손실을 처리하지 못한다. 이로 인해 사용자가 요청하는 질의에 응답하지 못하거나 부정확한 질의 처리 결과를 제공하는 문제점이 발생된다. 이 논문에서는 이와 같은 문제점을 해결할 수 있는 이동 객체 관리 시스템을 설계한다. 제안 시스템은 시간에 종속적인 위치 변화 함수를 이용하여 이동 객체의 과거 및 미래의 위치 정보를 함께 처리할 수 있다. 또한, 실시간 환경에서 발생되는 이동 객체의 위치 정보 손실을 보완하기 위한 위치 정보 트리거링 기법을 제안한다. 마지막으로, 제안 시스템을 PDA 기반의 차량 검색 시스템에 적용 및 구현한다. 이로 인해 제안 시스템이 위치기반서비스에 적용 가능함을 보인다.

Abstract

A moving object management system manages spatiotemporal data of moving objects which change their location continuously over time such as people, animals, cars, cellular phones, and so on. This system can be applied to location based services such as vehicle tracking systems, digital battlefields, and animal habitat management. The existing systems neither suggest location estimation of the moving objects nor handle the loss data of the moving objects in real-time environment. Thus the existing systems have problems that they give the uncertain results of the query processing to the user query. In this paper, we design a new moving object management system. The proposed system processes the past and future location information of the moving objects by the location change function. Also we propose a location triggering method, which supplements loss of the location data of the mobile objects in real-time environment. Finally, we implement and apply the proposed system to a vehicle tracking system based on PDA. Thus we ascertain that the proposed system can be applied to the location based system.

1) 정회원 : 청주과학대학 컴퓨터과학과

논문접수 : 2003. 12. 12.
심사완료 : 2003. 12. 18.

1. 서론

자동차, 비행기, 사람, PDA, 휴대폰, 노트북 등과 같은 이동 객체의 연속적인 위치 변화 과정을 관리하고, 저장된 위치 정보를 이용하여 사용자에게 다양한 질의 처리 기능을 제공하는 시스템을 이동 객체 관리 시스템이라 한다. 최근 텔레메틱스, 물류 및 수송 관리, 모바일 상거래 등과 같은 위치기반서비스 분야에서 이동 객체(moving objects)의 위치 정보를 이용한 응용의 연구가 활발하게 추진되고 있다[1].

지금까지 연구된 이동 객체 관리 시스템 모형에는 추측 항법 기반의 차량 추적 프로토타입인 DOMINO[2,3], 시공간 이동 객체 데이터베이스 프로토타입인 CHOROS[4,5,6,7,8], 선형 제약사항 기반의 시공간 객체 관리 프로토타입인 DEDALE[9,10], 시공간 추론 기법을 적용한 전장분석 프로토타입인 STRBA[11,12,13], 물류 차량 관리 시스템인 MOMS[14] 등이 있다. 그런데, 기존의 연구에서는 데이터베이스에 저장된 이동 객체의 불확실한 위치 정보 값을 제공하지 못하고, 실시간 환경의 불확실성을 고려하지 않고 있다.

특히, 기존의 상용화된 데이터베이스 관리 시스템을 이용하여 이동 객체를 관리할 경우 이동 객체의 연속적인 위치 변화 정보를 정확하게 저장 및 검색할 수 있는 함수를 제공받지 못한다. 이로 인해 이동 객체의 이동 궤적(trajectory)을 완전하게 표현하지 못하는 문제점이 발생된다. 따라서 기존의 상용 데이터베이스 시스템을 이용하여 이동 객체의 위치 정보 및 이동 궤적을 효과적으로 관리하기 위한 시스템 모델이 필요하게 되었다.

이 논문에서 설계하는 이동 객체 관리 시스템은 기존의 상용 데이터베이스에서 처리할 수 없는 이동 객체의 연속적인 위치 정보를 처리하는 연산 및 위치 추정 기능을 지원한다. 또한, 이력 정보의 불확실성 값의 범위를 제공하며, 실시간 환경에서 발생되는 위치 정보의 결손을 보완하는 위치 정보 트리거링을 토대로 기존의 이동 객체 관리 시스템이 갖는 문제점을 보완한다. 아울러, 제안 시스템을 차량 위치 검색 시스템에 적용하여 구현한 결과를 통해 이 논문에서 설계한 이동 객체 관리 시스템이 위치기반서비스에 적용 가능함을 보인다.

이 논문의 전체 구성은 다음과 같다. 2절에서는 이동 객체 관리 시스템의 관련연구를 기술한다. 3절에서는 위치 기반 응용을 위한 이동 객체 관리 시스템을 설계하고, 각 구성요소별 기능 및 처리 방법을 기술한다. 4절에서는 제안 시스템을 차량 위치 검색 시스템에 적용 및 구현한 결과를 보인다. 마지막으로, 5장에서는 결론을 맺는다.

2. 관련연구

이동 객체 관리 시스템[1]은 자동차, 비행기, 배, PDA, 노트북 등과 같은 객체들의 연속적인 위치 변화 과정을 저장 및 관리하며, 저장된 위치 정보를 이용하여 사용자에게 다양한 질의 처리 기능을 제공하기 위한 연산 기능을 수행한다. 지금까지 연구된 대표적인 관련 프로토타입에는 DOMINO, CHOROS, DEDALE, STRBA, MOMS 등이 있다.

DOMINO[2,3,15]는 추측 항법(dead-reckoning)의 위치 추정 기법을 적용한 실시간 이동 객체의 위치 추적 프로토타입이다. 이동 객체의 동적 속성을 위한 MOST(moving objects spatio-temporal) 모델을 제안하였으며, FTL을 이용한 미래 시간에 대한 질의 표현 방법을 제시하였다. 이 시스템은 기존의 상용 DBMS 기술과 GIS를 이용한 인터페이스를 사용하였다. 그러나, 이 프로토타입은 이동 객체의 현재 위치, 속도, 방향 정보를 이용하여 미래의 위치를 예측하는 데 주로 초점을 맞추고 있으며, 이동 객체의 과거 위치 정보를 데이터베이스에 전혀 저장하지 않는다. 따라서, 과거 시점부터 현재 시점까지를 모두 포함하는 이동 객체의 이동 궤적과 관련된 질의 처리 기능을 지원하지 못하는 문제점을 가진다.

CHOROS[4,8,16]는 시공간 데이터베이스(spatiotemporal database) 시스템의 설계 및 구현에서 포함된 문제들을 연구하여 시공간 데이터베이스 시스템의 구조를 제안하고 부분적으로 구현하고 있다. 또한, GPS 기반의 수송 관리 시스템과 멀티미디어 시스템에 적용한 응용 시나

리오를 제시하였다. 그러나, 아직 이동 객체 데이터베이스를 활용한 응용 시스템의 모델 및 개발 사례는 제시되지 않고 있으며, 개발 중인 질의 처리 시스템에서는 이동 객체의 불확실한 과거 및 미래의 위치 정보 추정에 관한 구체적인 방법이 제시되지 않고 있다.

DEDALE[9,10]은 제약사항 데이터베이스 모델을 이용하여 시공간 데이터를 모델링하고 질의 처리를 하기 위해 개발된 프로토타입이다. DEDALE은 기존의 시공간 데이터의 모델뿐만 아니라 이동 객체의 궤적 등과 같은 데이터 모델 및 질의 표현도 제공한다. 그러나, DEDALE 프로토타입은 데이터베이스에 시간의 변화에 따른 이동 객체의 위치 정보가 직접 저장되지 않고, 특정 구간의 궤적을 표현하는 선형 제약사항의 공식이 저장되므로 실시간 위치 모니터링을 위한 응용 시스템의 질의 처리 기능을 지원하지 못한다.

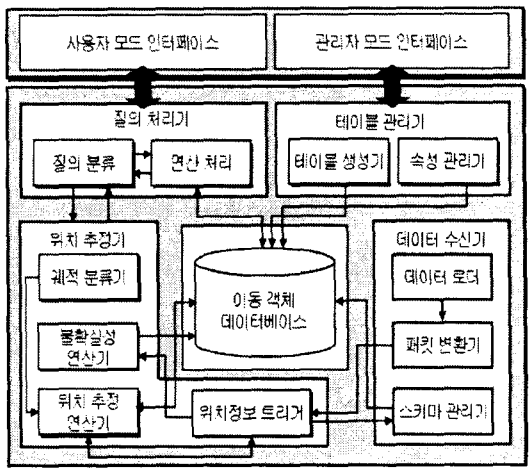
STRBA[11,12,13]는 전장분석을 위한 시공간 추론 시스템 프로토타입으로 모의 전장에서 이동하는 부대 및 탱크들의 움직임을 예측하여 이를 의사결정에 활용할 수 있도록 개발되었다. 전장분석 프로토타입은 이동 객체 관리기와 추론 엔진을 접목시키고자 하는 데 초점이 맞추어졌다. 특히, 시공간 이동 객체의 연산 결과를 추론 엔진에서 활용하는 새로운 이동 객체 추론 모델을 제시하였다. MOMS[14]는 물류 차량 관리를 위한 이동 객체 관리 엔진이다. 이 프로토타입은 기존의 차량 추적 시스템의 기능을 제공함은 물론 이동 차량의 과거 및 현재의 위치 정보를 제공한다. 아울러, 기존의 GPS, Beacon, ITS 등의 서로 다른 차량 추적 시스템의 정보를 통합하여 하나의 시스템에서 관리하려는 시도를 하였다.

3. 이동 객체 관리 시스템의 설계

이동 객체의 관리는 위치 정보의 획득 방법에 따라, 사건 지향(event driven) 시스템과 관측 지향(observation driven) 시스템의 두 가지 형태로 분류된다[17]. 사건 지향 시스템은 이동 객체의 속도나 방향의 변화를 자동으로 검출하는 것이 가능한 경우의 시스템을 말한다. 관측 지향 시스템은 GPS와 같은 센서 시스템을 이용하여 정규적인 시간의 간격에서 정렬된 순서대로 객체의 위치를 획득하는 시스템을 말한다. 이 논문에서는 관측 지향 위치 기반 응용을 기반으로 하는 이동 객체 관리 시스템을 설계한다.

3.1 시스템 구조

위치 기반 응용을 위한 이동 객체 정보 관리 시스템은 상용 DBMS 기술을 활용하여 이동 객체의 위치 변화 정보를 효율적으로 관리하고, 이와 관련된 유용한 정보를 응용 시스템 사용자에게 제공한다. 제안 시스템은 인터페이스, 질의 처리기, 위치 추정기, 테이블 관리기, 데이터 수신기, 이동 객체 데이터베이스로 구성된다.



(그림 1) 이동 객체 관리 시스템의 구조

(그림 1)에서 사용자 모드는 질의를 입력하고 실행 결과를 제공받는다. 관리자 모드는 데이터베이스의 릴레이션 관리 및 위치 정보 수신을 위한 환경 설정을 한다. 질의 처리기는 질의 수행 계획을 수립하여 연산을 수행한 후 결과를 반환한다. 위치 추정기는 데이터베이스에 직접 저장되지 않은 이동 객체의 위치 정보를 추정한다. 테이블 관리기는 관리자가 이동 객체 데이터베이스에 새로운 객체 릴레이션을 생성하고, 일반 속성 정보를 관리할 수 있도록 한다. 데이

터 수신기는 이동 객체의 실시간 위치 정보를 수신하고 이를 데이터베이스에 저장한다. 이동 객체 데이터베이스는 객체의 속성 정보 및 실시간 위치 정보가 저장된다.

3.2 데이블 관리기

이동 객체의 위치 관련 정보는 세 개의 원소 (Oid, t, v)로 구성되며, Oid는 객체의 식별자, t는 유효 시간, v는 객체의 위치에 해당되는 (x, y) 좌표 값을 나타낸다[17]. 이 논문에서는 관계형 모델을 이용하여 이동 객체의 위치 정보를 표현하며, 모두 세 개의 릴레이션으로 구성한다.

먼저, Moving_Objects 릴레이션은 (mo_id, name, manager, type, tag)로 구성되며, 이동 객체의 일반 속성 정보가 저장된다.

(표 1) Moving_Objects 릴레이션

mo_id	name	manager	type	tag
char(10)	char(10)	char(10)	char(10)	int

Moving_History 릴레이션은 (mo_id, t_start, t_end, x_start, y_start, x_end, y_end, u_id)의 속성을 가지며, 일정한 유효 시간의 간격에 따라 GPS로부터 측정된 이동 객체의 위치 좌표 값이 저장된다.

(표 2) Moving_History 릴레이션

mo_id	t_start	t_end	x_start	y_start	x_end	y_end	u_id
char(10)	char(20)	char(20)	float	float	float	float	char(10)

Uncertainty_History 릴레이션은 (u_id, center_x, center_y, radius)로 구성되며, Moving_History 릴레이션에 저장된 각 이력 튜플에 대한 불확실성 영역 값이 저장된다.

(표 3) Uncertainty_History 릴레이션

u_id	center_x	center_y	radius
char(10)	float	float	float

(표 1, 2, 3)에서 mo_id는 객체의 식별자, name은 객체의 이름, manager는 관리자 이름, type은 객체의 위치 변화 유형을 나타낸다. t_start와 t_end는 유효 시간의 시작 시점과 종료 시점을 나타낸다. x_start와 y_start는 t_start 시점의 (x, y) 좌표 값이고, x_end와 y_end는 t_end 시점의 (x, y) 좌표 값이다. u_id는 하나의 이력 위치 정보 튜플이 가지는 불확실성 영역 값을 나타내는 식별자이다. center_x와 center_y는 불확실성 영역 값의 중심 좌표 값이고, radius는 (center_x, center_y)로부터의 반경을 나타내는 값이다.

3.3 데이터 수신기

이동 객체의 위치 정보는 외부 위치 제공 서버로부터 데이터 로더를 통해 수신된다. 데이터 로더에서는 실시간으로 전송되는 이동 객체의 위치 정보를 TCP/IP를 이용하여 패킷 형태로 읽어들인다. 데이터 로더를 통해 수신된 위치 정보 패킷은 16진수 형태로 구성되어 있다. (그림 2)는 이동 객체의 위치 정보를 패킷 형태로 표현한 예이다.

```
7e 00 1d 11 15 41 08 2a 01 3d 11 06 02 b1 cb d5
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
7e 00 1d 11 15 41 04 11 01 26 de d5 02 af 25 f5
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
7e 00 1d 11 15 41 03 39 01 2d b5 d2 02 af 48 a1
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
7e 00 1d 11 15 41 07 8d 01 21 ac 5e 02 ab fb 73
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
7e 00 1d 11 15 41 05 0c 01 2d d5 27 02 92 f2 7e
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
7e 00 1d 11 15 41 02 5b 01 3b 68 63 02 c0 00 2e
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
7e 00 1d 11 15 41 02 5b 01 3b 68 63 02 c0 00 2e
11 20 38 00 00 00 41 00 00 00 00 00 00 00 00 00
```

(그림 2) 위치 정보 패킷

수신된 값은 시스템에서 처리할 수 있는 형태로 변환된다. 클라이언트 소켓을 이용한 데이터 로더의 패킷 수신 및 변환 과정은 (알고리즘 1)과 같다.

(알고리즘 1) 위치 정보 수신 및 변환

Algorithm DataLoader(ServerIp)

Input: *ServerIp*(실시간 위치 제공 서버의

주소),
Output: *LoadPacket*(수신된 위치 정보 패킷 배열)
Begin
  socket ← new Socket(*ServerIp*, 포트번호);
  byte[] *LoadPacket* ← new byte[32];
  in ← new BufferedInputStream(socket.getInputStream());
  For i ← 0, 31
    ReadOneByte ← (byte) in.read();
    LoadPacket[i] ← *ReadOneByte*;
  Return *LoadPacket*;
End

Algorithm PacketTranslator(LoadPacket)

Input: *LoadPacket*(바이트 형태로 구성된 패킷 데이터 배열)
Output: *ParsedPacket*(DB 저장 형태로 변환된 패킷 정보)
Begin
  mo_id ← ((*LoadPacket*[4]*255+*LoadPacket*[5])*255+*LoadPacket*[6])*255+*LoadPacket*[7];
  x ← ((*LoadPacket*[8]*255+*LoadPacket*[9])*255+*LoadPacket*[10])*255+*LoadPacket*[11]/100;
  y ← ((*LoadPacket*[12]*255+*LoadPacket*[13])*255+*LoadPacket*[14])*255+*LoadPacket*[15]/100;
  time ← ""+*LoadPacket*[16]+"-"+*LoadPacket*[17]+"-"+*LoadPacket*[18];
  ParsedPacket[0] ← *mo_id*; *ParsedPacket*[1] ← *x*;
  ParsedPacket[2] ← *y*; *ParsedPacket*[3] ← *time*;
Return *ParsedPacket*;
End

(알고리즘 1)의 *DataLoader*는 실시간 위치 제공 서버의 URL을 입력 값으로 받은 후 수신된 위치 정보의 패킷 배열을 출력 값으로 제공한다. *PacketTranslator*는 데이터 로더를 통해 수신된 위치 정보 패킷 중에서 이동 객체 데이터베이스에 실제 저장될 *mo_id*, *x*, *y*, *time* 부분의 값만을 추출하여, 16진수 형태로 입력된 각각의 값들을 10진수 형태로 변환한 후 해당되는 데이터 타입으로 변경하여 *ParsedPacket*에 저장한 후 반환한다.

3.4 질의 처리기

이동 객체의 위치 정보에 대한 질의 처리를 위한 기본 함수들은 (표 4)와 같다. 이 함수는 [7,18]에서 제시된 이동 객체 연산들을 참조하여 정의한 것이다. (표 4)의 입/출력에서 *mid*, mid_A, mid_B는 이동 객체의 식별자이고, $[t_s, t_e]$는 유효 시간의 간격이다. 이 때 $t_s \leq t_e$의 관계가 항상 성립한다. *real*은 실수 값, *line*은 선으로 표현되는 공간 속성, *location*은 이동 객체의 (x, y) 좌표 값이다.

(표 4) 질의 처리 함수

함수	입력	출력	연산식
mdistance	$mid_A \times mid_B \times [t_s, t_e]$	real	$mdistance = \bigcup_{i=1}^{n} distance_{t_i}$ $distance_{t_i} = \sqrt{(By_{t_i} - Ay_{t_i})^2 + (Bx_{t_i} - Ax_{t_i})^2}$
trajectory	$mid \times [t_s, t_e]$	line	$trajectory = \bigcup_{i=1}^{n} Traj_{I_i}$ $Traj_{I_i} = \frac{y_{t_{i+1}} - y_{t_i}}{x_{t_{i+1}} - x_{t_i}}(x - x_{t_{i+1}}) + y_{t_{i+1}}$
length	$mid \times [t_s, t_e]$	real	$length = \sum_{i=1}^{n} distance_{I_i}$
velocity	$mid \times [t_s, t_e]$	real	$velocity = \frac{length}{t_e - t_s}$
attime	$mid \times [t_s, t_e]$	location	$attime = \bigcup_{i=1}^{n} (x_{t_i}, y_{t_i}), t_i \in [t_s, t_e]$
mnearest	$mid \times [t_s, t_e]$	location	$mnearest = loc(\min\{mdistance(mid, candidate_i)\}_{i=1}^{n})$
mfarthest	$mid \times [t_s, t_e]$	location	$mfarthest = loc(\max\{mdistance(mid, candidate_i)\}_{i=1}^{n})$
minvalue	$mid \times [t_s, t_e]$	location	$minvalue = loc(\min\{(x_i, y_i)\}_{i=1}^{n})$
maxvalue	$mid \times [t_s, t_e]$	location	$maxvalue = loc(\max\{(x_i, y_i)\}_{i=1}^{n})$
uncertainty	$mid \times [t_s, t_e]$	real	$uncertainty = \bigcup_{i=1}^{n} area_{I_i}$ $area_{I_i} = \langle center_{x_i}, center_{y_i}, radius_i \rangle$

(표 4)에서 *mdistance*는 임의의 두 이동 객체 mid_A와 mid_B 간의 유효 시간 간격 $T = [t_s, t_e]$ 동안의 거리를 계산한다. $distance_{t_i}$는 t_i 시점에서 mid_A와 mid_B의 거리이다. *trajectory*는 T 동안 *mid*가 이동한 위치 좌표를 모두 검색하여 궤적을 표현한다. *length*는 *trajectory* 연산 결과로 생성된 이동 경로의 총 거리를 계산한다. *velocity*는 T에서 *mid*의 평균 이동 속도를 구한다. *attime*은 T 동안 각 시점 t_i에서 *mid*의 위치 좌표 값을 모두 검색한다. *mnearest*는 T 동안 *mid*와 가장 가까운 곳에 위치한 객체의 위치 좌표를 구한다. *mfarthest*는 *mnearest*와 반대이다. *minvalue*와 *maxvalue*는 T 동안 존재하는 이동 객체의 모든 위치 좌표 중 최소 또는 최대가 되는 값을 추출한다. *uncertainty*는 T 동안 이동 객체 *mid*의 이동 궤적에 관한 불확실성 영역 값을 제공한다.

3.5 위치 추정기

관측 지향 이동 객체 관리 시스템에서 가장 많이 사용되는 위치 결정 도구는 위성 신호를 이용하는 GPS 단말기이다. 그러나, 위성 신호는 지하에는 도달하지 못할 뿐만 아니라, 높은 빌딩 또는 밀집된 나무 숲에 의해 방해를 받는다 [19]. 이와 같은 지역에서는 GPS에 의해 관측된 위치 정보를 제공받지 못하게 되고, 데이터베이스에 위치 정보를 저장할 수 없는 문제점이 발생된다. 이 논문에서는 수신되지 않은 위치 정보를 추정하고, 그 결과를 데이터베이스에 저장하는 위치 정보 트리거링 방법을 제시한다. 위치 정보 트리거는 위치 추정 연산기를 통해 손실된 위치 값의 추정 결과를 반환받으며, 이 값을 다시 스키마 관리기로 넘겨준다.

(알고리즘 2) 실시간 위치 정보 트리거링

Algorithm

LocationTrigger(ParsedPacket, Tag)

Input: *ParsedPacket*(파싱된 패킷 정보); *Tag*(이동 궤적 유형)
Output: *InputData*(DB에 저장될 최종 위치 정보 저장 배열)
Begin
InputData ← *ParsedPacket*; *mo_id* ← *ParsedPacket*[0];
temp_x ← *ParsedPacket*[1]; *temp_y* ← *ParsedPacket*[2];
t_now ← *ParsedPacket*[3];
If (*temp_x* == 'null' or *temp_y* == 'null') Then
If (Tag == 1) Then
future_location ← *FutureLinear*(*mo_id, t_now*);
InputData[1] ← *future_location*[0];
InputData[2] ← *future_location*[1];
If (Tag == 2) Then
future_location ← *FutureSpline*(*mo_id, t_now*);
InputData[1] ← *future_location*[0];
InputData[2] ← *future_location*[1];
Else *InputData*[1] ← *temp_x*; *InputData*[2] ← *temp_y*;
Return *InputData*;
End

Algorithm SchemaManager(InputData)

Input: *InputData*(Moving_History에 저장될 객체의 이동 정보)
Begin
mo_id ← *InputData*[0];
table ← Moving_Objects에서 mo_id가 저장된 테이블 검색;
target ← "Moving_History_"+table;
time ← *InputData*[3];
x ← *InputData*[1];
y ← *InputData*[2];
target 릴레이션에 새로운 위치 정보를 추가;
End

(알고리즘 2)의 *LocationTrigger* 모듈은 패킷 변환기로부터 생성된 위치 정보인 *ParsedPacket*과 Moving_Objects 릴레이션에 저장된 tag 정보를 입력받아 데이터베이스에 저장될 최종 위치 정보가 저장된 *InputData*를 반환한다. *SchemaManager*는 *InputData*를 입력받은 후 이 정보를 해당되는 이력 릴레이션에 추가시킨다. 스키마 관리기가 실행될 때 위치 추정기의 불확실성 연산 모듈이 함께 수행된다. 불확실성 연산 모듈을 통해 이력 위치 정보의 불확실성 영역 값이 생성된다. 이동 객체의 불확실성은 데이터베이스의 모델링, 질의 처리, 인덱싱 등에 대해 다양한 영향을 미치게 된다. 특히, 측정 에러나 샘플링 에러로 인해 발생되는 위치 정보의 불확실성으로 인해 발생되는 질의 결과의 부정확성은 사용자에게 잘못된 의사 결정 요인을 제공할 수 있게 된다. 따라서, 이 논문에서는 다음과 같은 방법으로 위치 추정 결과에 대한 불확실성 값의 범위를 정량화하여 사용자의 질의 결과로서 제공한다.

Uncertainty_History에 저장되는 불확실성 영역에서 center_x와 center_y는 불확실성 범위의 중심 좌표이고, radius는 중심 점 (center_x, center_y)로부터 하나의 원을 이루기 위한 반지름이 된다. 스키마 관리기를 통해 생성되는 릴레이션은 Moving_History 릴레이션이다. 이 때 이동 객체 데이터베이스에는 Uncertainty_History 릴레이션도 함께 생성되어 저장된다. Uncertainty_History 릴레이션은 위치 추정기의 불확실성 연산기에 의해 생성된다. 불확실성 연산 모듈은 (알고리즘 3)과 같다.

(알고리즘 3) 이력 정보의 불확실성 영역

Algorithm UncertaintyCreator(HistoryTuple,u_id,table)

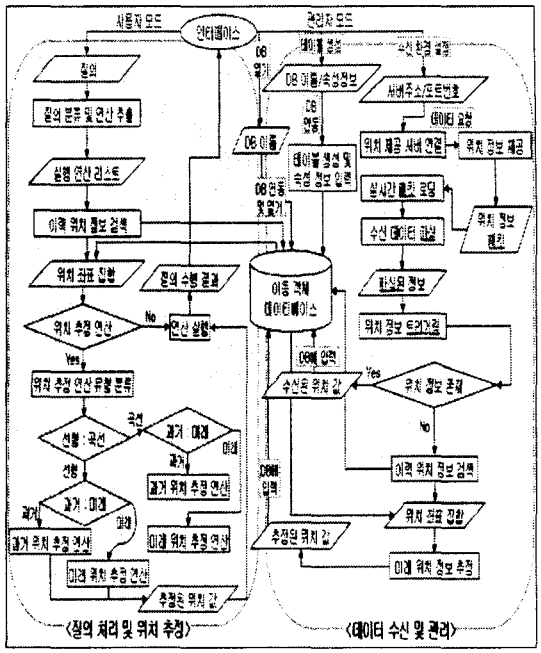
Input: *HistoryTuple*(가장 최근의 이력 튜플);

u_id(불확실성 값의 식별자); *table*(이력 릴레이션명)
Begin
x_start ← *HistoryTuple*[0]; y_start ← *HistoryTuple*[1];
x_end ← *HistoryTuple*[2]; y_end ← *HistoryTuple*[3];
table ← Moving_Objects에서 mo_id가 저장된 테이블 검색;
center_x ← x_start + (x_end - x_start)/2;
center_y ← y_start + (y_end - y_start)/2;
radius ← SQRT((x_end-x_start)^2+(y_end-y_start)^2)/2;
target ← "Uncertainty_History_"+table;
target 릴레이션에 새로운 불확실성 값을 추가;
End

Uncertainty_History는 Moving_History에 저장된 각 튜플의 불확실성 영역 값을 저장하며, Uncertainty_History에 저장되는 정보는 실시간 위치 정보가 Moving_History에 저장되는 이력 위치 정보의 수만큼 생성된다.

3.6 시스템의 동작 과정

이동 객체 관리 시스템의 전체적인 동작 시나리오를 그림으로 표현하면 (그림 3)과 같다. 시스템의 동작 흐름에 따라 전체 구성을 질의 처리 및 위치 추정 부분과 데이터 수신 및 관리 부분으로 나누어 구성하였다.



(그림 3) 이동 객체 관리 시스템의 동작

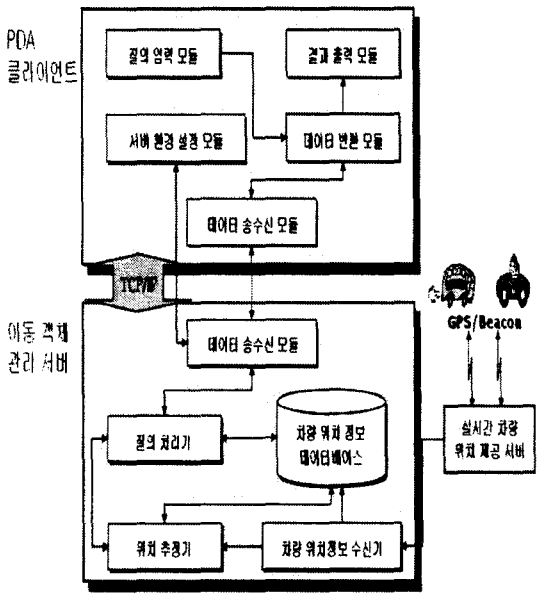
(그림 3)의 질의 처리 및 위치 추정 부분에서는 사용자 질의를 입력받아 이동 객체 연산 처리를 한 후 그 결과를 다시 사용자에게 반환한다. 위치 추정 연산은 데이터 모델의 유형에 따라 직선 궤적 또는 곡선 궤적으로 구분되고, 질의 시간에 따라 과거 및 미래로 구분된다. 데이터 수신 및 관리 부분에서는 외부에서 송신되는 실시간 위치 정보 패킷을 수신 받아, 데이터베이스에 저장 가능한 형태로 변환한다. 위치 정보 트리거는 변환된 패킷 데이터 중에서 위치 정보를 추출한다. 이 때 추출된 위치 정보가 정상적으로 존재하면, 이 데이터를 스키마 관리기로 넘겨주고 데이터베이스의 이력 위치 정보 릴레이션에 저장한다. 그러나 변환된 패킷 데이터의 위치 정보가 존재하지 않으면, 미래 위치 추정 연산을 통해 손실된 위치 정보를 추정한다. 그리고 추정된 위치 정보를 스키마 관리기를 통해 데이터베이스에 저장한다.

4. 적용

4.1 시스템 구성 및 구현 환경

이 논문에서는 제안한 이동 객체 관리 시스

템을 PDA를 이용한 차량 위치 검색 시스템에 적용하였다. 적용 시스템은 이동 객체 관리 시스템을 서버로 하고, PDA를 클라이언트로 하여 구현하였다. PDA 클라이언트를 이용한 차량 위치 검색 시스템은 차량 위치 정보를 저장 및 관리하고 이동 인터페이스의 차량 위치 검색 질의를 수행한다. 적용 시스템은 이동 객체 관리 서버와 PDA 클라이언트로 구성되며, 시스템의 구조는 (그림 4)와 같다.



(그림 4) PDA 기반 차량 위치 검색 시스템

(그림 4)에서 차량 위치 정보 관리를 위한 이동 객체 관리 서버는 차량 위치 정보 수신기, 데이터 송수신 모듈, 질의 처리기, 위치 추정기, 차량 위치 정보 데이터베이스로 구성된다. 차량 위치 검색을 위한 PDA 클라이언트는 질의 입력 모듈, 서버 환경 설정 모듈, 데이터 변환 모듈, 결과 출력 모듈, 데이터 송수신 모듈로 구성된다.

시스템의 구현 환경은 다음과 같다. 이동 객체 관리 서버는 Pentium-III 933MHZ, Windows 2000 Server 운영체제에서 구현되었다. 구현 프로그래밍 언어는 JAVA이고, 개발 도구는 JDK 1.3, DBMS는 관계형 기반의 SQL server 2000을 사용하였다. PDA 클라이언트는 Microsoft Windows CE 2002(Pocket PC) 운영체제를 사용하였으며, Microsoft Embedded Visual C++(EVC) 4.0으로 구현하였다. 클라이언트의 테스트는 Pocket PC 운영체제를 내장한 PDA 및 Desktop Pocket PC Emulator에서 실행하였다.

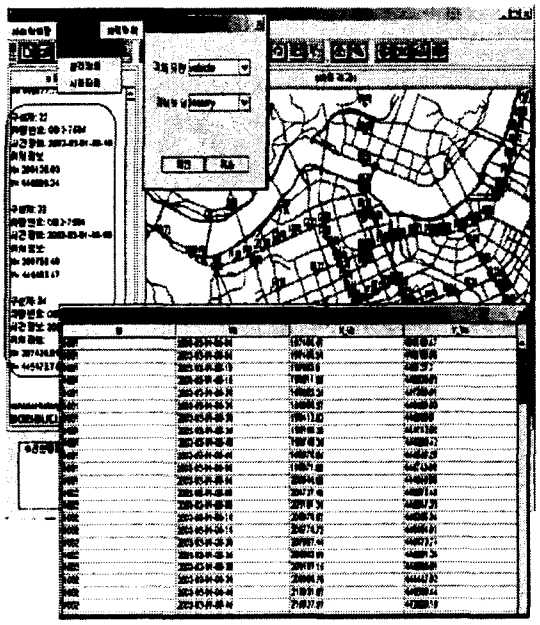
4.2 응용 시나리오 및 질의 실행

구현한 이동 객체 관리 시스템 서버와 PDA 클라이언트의 질의 실행을 보이기 위해 다음과 같은 응용 시나리오를 구성하였다. 첫째, 이동 객체의 적용 대상은 이동 차량으로 간주하였다. 둘째, 이동 차량은 서울시의 주요 간선 도로에서 이동하고 있다. 셋째, GPS를 이용한 이동 차량의 위치 정보는 5분 간격으로 외부의 실시간 위치 제공 서버로부터 전송받는다. 넷째, 이동 차량의 위치 정보 생성을 위해 ZEUS 공간 객체 관리 시스템에 구축된 서울시 데이터베이스의 맵 정보와 좌표를 활용하였다. 다섯째, 사용된 위치 좌표는 TM 좌표 체계를 따른 것이다.

4.2.1 이동 객체 관리 서버의 질의 실행

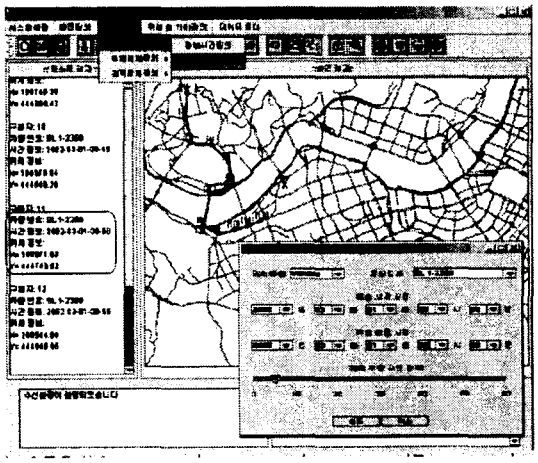
이동 객체 관리 서버의 인터페이스는 전체 네 부분의 메뉴로 구성되었으며, 전체 메뉴는 다음과 같다. 첫째, 시스템 설정에는 이동 객체 관리 데이터베이스와의 연결, 해제, 시스템 종료 메뉴가 있다. 둘째, 테이블 관리에는 테이블 검색, 테이블 생성, 테이블 삭제 메뉴가 있다. 셋째, 데이터 로더에는 위치 제공 서버와의 수신 환경 설정 메뉴가 있다. 넷째, 이동 객체 연산에는 mdistance, trajectory, length, velocity, attime, mnearest, mfarthest, minvalue, maxvalue, uncertainty 메뉴가 있다.

이동 객체 관리 서버의 질의 결과를 보이면 다음과 같다. 먼저, 이력 위치 정보 테이블 검색 기능은 현재 이동 객체 관리 데이터베이스에 저장된 모든 이력 위치 정보를 검색한다. 인터페이스를 통한 질의 입력 화면 및 실행 결과는 (그림 5)와 같다.



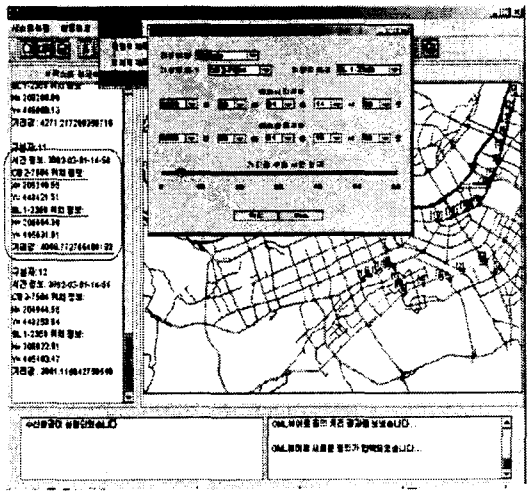
(그림 5) 이동 객체의 이력 위치 정보 검색

(그림 5)는 이력 릴레이션에 저장된 위치 정보를 검색한 결과를 사용자 인터페이스를 통해서 출력한 화면이다. 이력 테이블에 저장된 검색뿐만 아니라 동일한 질의 메뉴를 통해서 객체의 일반 속성 정보 및 현재 정보 테이블도 검색이 가능하다.



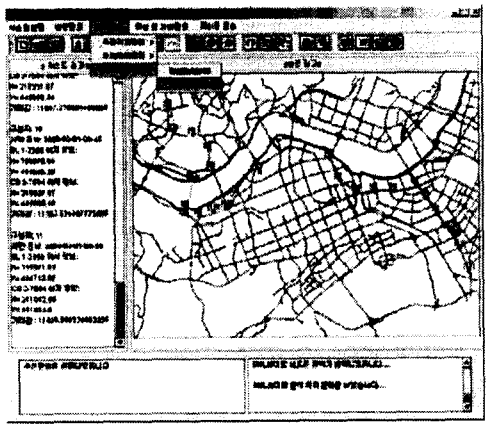
(그림 6) 특정 시간 구간 동안 이동 객체의 궤적 검색

trajectory 질의는 객체의 이동 궤적을 검색한다. (그림 6)은 차량번호가 'SL 1-2358'인 객체가 '2002년 3월 1일 08시 00분 ~ 2002년 3월 1일 09시 00분' 까지 이동한 궤적을 검색한 결과 화면이다. 출력 화면의 왼쪽 텍스트 창에는 데이터베이스 저장된 차량의 위치 좌표를 보이고, 오른쪽의 지도 화면에는 이동 지점을 표시하고 있다.



(그림 7) 이동 차량간의 이동 거리 검색

mdistance 연산은 특정 유효 시간 동안 두 객체간의 거리를 계산한다. (그림 7)은 차량번호가 'CB 2-7584'인 객체와 'SL 1-2358'인 객체가 '2002년 3월 1일 14시 00분 ~ 2002년 3월 1일 15시 00분' 까지 이동한 경로와 두 객체간의 거리를 검색한 결과 화면이다. 화면의 왼쪽 텍스트 창에는 동일한 유효 시점에서 두 이동 객체의 위치 좌표 및 거리 계산 결과가 나타나며, 오른쪽 화면에는 두 객체의 이동 경로를 보이고 있다.

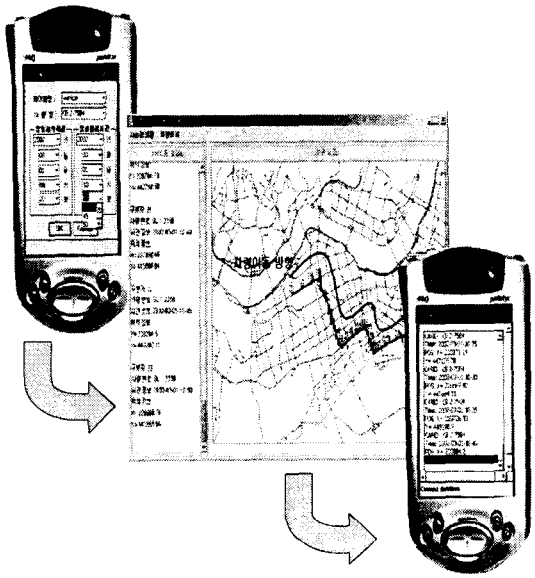


(그림 8) 특정 객체와 가장 가까운 곳의 객체 검색

mnearest 연산은 특정 유효 시간 동안 임의의 이동 차량과 가장 가까운 곳에 위치한 객체의 위치를 검색한다. (그림 8)과 같이 검색된 차량 정보는 텍스트 및 그래픽 형태로 출력된다. 텍스트 형태의 출력 데이터는 시간 정보, SL-2358 차량의 위치 정보, 이 차량과 가장 가까운 차량들의 같은 시간에서의 위치 정보, 두 차량간의 거리값이다. 그래픽 형태의 출력은 SL-2358 차량의 시간별 이동한 궤적과 이 차량과 가장 가까운 거리에 위치한 차량의 궤적을 보여준다.

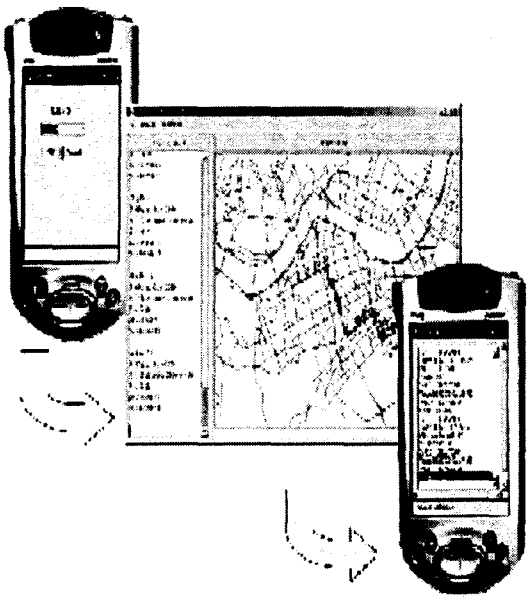
4.2.2 PDA 클라이언트의 질의 실행

PDA 클라이언트에서는 이동 객체 관리 서버의 모든 질의를 수행하지는 못한다. 현재 구현된 기능은 특정 유효 시간 동안 이동 차량의 위치와 관련된 검색만이 가능하다. PDA 클라이언트의 질의 입력은 위치 검색 서버에 접속하는 것으로 시작된다. (그림 9)와 같이 PDA 클라이언트의 사용자 인터페이스를 통해 서버에 질의를 입력한다.



(그림 9) 특정 유효 시간 동안 차량의 위치 검색

(그림 9)와 같이 PDA 인터페이스의 질의 메뉴에서 질의를 선택하고 테이블명, 차량명, 유효시작 및 종료시간을 입력하면 서버로 질의 데이터가 전송된다. 서버에서는 질의를 처리하고 (그림 9)의 서버 인터페이스와 같이 텍스트 및 그래픽 형태로 결과를 보여주고 클라이언트로 결과를 보낸다. 마지막으로 PDA 클라이언트는 결과를 전송 받아 텍스트 형태로 출력한다.



(그림 10)은 모든 차량의 전체 시간구간 위치 검색 질의 수행과정이다. (그림 10)은 이동 객체 관리 서버에 저장된 데이터베이스의 이력 릴레이션을 검색하여 결과를 보인 것이다.

5. 결론

이동 객체 정보 관리 시스템은 자동차, 비행기, 배, PDA, 노트북 등과 같은 객체들의 연속적인 위치 변화 과정을 저장 및 관리하며, 저장된 위치 정보를 이용하여 사용자에게 다양한 질의 처리 기능을 제공하기 위한 연산 기능을 수행한다. 대표적인 이동 객체 정보 관리 시스템에는 DOMINO, CHOROS, DEDALE, STRBA, MOMS 등이 있다. 그런데, 기존의 연구에서는 데이터베이스에 저장된 이동 객체의 불확실한 위치 정보 값을 제공하지 못하고, 실시간 환경의 불확실성을 고려하지 않고 있다.

이 논문에서는 데이터베이스에 저장된 이력 위치 정보의 불확실성 값을 제시하기 위해 두 점의 위치 좌표를 이용한 불확실성 연산 방법을 제시하였다. 또한, 실시간 환경에서 발생되는 위치 정보의 손실을 보완하기 위해 실시간 위치 정보 트리거링 방법을 제시하였다. 제안하는 이동 객체 관리 시스템 모형은 기존의 상용 데이터베이스에서 처리할 수 없는 연속적인 위치 정보를 관리할 수 있는 연산 기능 및 위치 추정 기능을 가진다. 이로 인해 사용자가 원하는 모든 시점에서의 위치 정보를 제공할 수 있다. 아울러, 실시간 이동 객체의 이력 위치 정보 관리 기능은 물론 저장된 이력 정보의 불확실성 값의 범위를 제공하며, 실시간 환경에서 발생되는 위치 정보의 결손을 보완하는 위치 정보 트리거링을 토대로 기존의 이동 객체 관리 시스템이 갖는 문제점을 보완할 수 있도록 하였다. 특히, 제안 시스템을 차량 위치 검색 시스템에 적용 및 구현하여 이와 유사한 위치기반서비스에 적용 가능함을 확인하였다.

앞으로는 제안한 시스템 모형이 다양한 위치 기반 응용 시스템에 적합한 구조가 되도록 확장 및 보완하는 연구를 추가로 진행할 것이다. 아울러, 좀더 다양한 위치기반 서비스에 직접 적용하는 연구를 진행할 것이다.

참고 문헌

[1] O. Wolfson, "Moving Objects Databases: Issues and Possible Solutions", Keynote Address, Mobile Data Management (MDM'01), January 2001.

[2] P. Sistla, O. Wolfson, S. Chamberlain, S. Dao, "Modeling and Querying Moving Objects", Proceedings of the 13th International Conference on Data Engineering (ICDE13), April 1997.

[3] O. Wolfson, B. Xu, S. Chamberlain, L. Jiang, "Moving Objects Databases: Issues and Solutions", Proceedings of the 10th International Conference on Scientific and Statistical Database Management, SSDBM'98, pp. 111-122, July 1998.

[4] S. Dieker and R. H. Guting, "Plug and Play with Query Algebras: SECONDO A Generic DBMS Development Environment", Proceedings of the

International Databases Engineering and Applications Symposium(IDEAS 2000), pp. 380-390, September 2000.

[5] M. Erwig, R.H. Güting, M. Schneider, and M. Vazirgiannis, "Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases," GeoInformatica Vol. 3, No. 3, pp. 269-296, 1999.

[6] L. Forlizzi, R.H. Güting, E. Nardelli, and M. Schneider, "A Data Model and Data Structures for Moving Objects Databases," Proceedings of the ACM SIGMOD Conference, pp. 319-330, 2000.

[7] R. H. Guting, and et. al, "A Foundation for Representing and Querying Moving Objects", ACM Transactions on Database Systems, Vol. 25, No. 1, pp. 1-42, 2000.

[8] J. A. C. Lema, L. Forlizzi, R. H. Guting, E. Nardelli and M. Schneider, "Algorithms for Moving Objects Databases", Fern University, Hagen, Informatik-Report 289, October 2001.

[9] S. Grumbach, P. Rigaux, and L. Segoufin, "Spatio-Temporal Data Handling with Constraints," ACM GIS, 1998.

[10] S. Grumbach, P. Rigaux, M. Scholl, and L. Segoufin, "The Design and Implementation of DEDALE", 1999.

[11] S. S. Park, Y. A. Ahn, and K. H. Ryu, "Moving Objects Spatiotemporal Reasoning Model for Battlefield Analysis," Proceedings of Military, Government and Aerospace Simulation part of ASTC2001, pp. 108-113, April 2001.

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

[13] 안윤애, 조동래, 류근호, "전장분석을 위한 이동 객체의 위치 예측 시스템", 정보과학회논문지, 2002년 12월.

[14] Dong Ho Kim, Jin Suk Kim, Yoon Ae Ahn, and Keun Ho Ryu, "Moving Objects Relational Model and Design for e-Logistics Applications", Proceedings of International Conference, ICITA2002, November 2002.

[15] O. Wolfson, P. Sistla, B. Xu, J. Zhou, S. Chamberlain, N. Rishe,Y. Yesha, "Tracking Moving Objects Using Database Technology in DOMINO", Proceedings of NGITS'99, The 4th Workshop on Next Generation Information Technologies and Systems, pp. 112-119, July 1999.

[16] R. H. Guting, S. Dieker, C. Freundorfer, L. Becker, and H. Schenk, "Secondo/QP: Implementation of a Generic Query Processor", 10th Internatioal Conference on Database and Expert System Applications (DEXA'99), LNCS 1677, Springer Verlag, pp. 66-87, 1999.

[17] J. Moreira, C. Ribeiro, and J.M. Saglio, "Representation and Manipulation of Moving Points: An Extended Data Model for Location Estimation", Cartography and Geographic Information Systems(CaGIS), ACSM, Vol. 26, No. 2, April 1999.

[18] M. Erwig, R.H. Guting, M. Schneider, and M. Vazirgiannis, "Abstract and Discrete Modeling of Spatio-Temporal Data Types," Chorochronos Technical Report, CH-98-14, 1998.

[19] R. Casey, "Automatic Vehicle Location Successful Transit Applications", A Cross-Cutting Study, in Intelligent Transportation Systems Electronic Document Library, Document No. 11487, U.S DOT Publication No. FHWA-JPO-99-022, August 2000.

안윤애

1993년 한남대학교 전자계산공학과(공학사)
1996년 충북대학교 대학원 전자계산학과(이학석사)
2003년 충북대학교 대학원 전자계산학과(이학박사)
2003년 - 현재 청주과학대학 컴퓨터과학과 전임강사
관심분야 : 시공간 데이터베이스, 모바일 데이터베이스, 모바일 GIS, 지식기반 시스템 등