

# 3차 스플라인 보간법을 이용한 이동 객체의 위치 추정

## (Location Prediction of Mobile Objects using the Cubic Spline Interpolation)

안윤애<sup>†</sup>   박정석<sup>\*\*</sup>   류근호<sup>\*\*\*</sup>  
(Yoon Ae Ahn) (Jeong Seok Park) (Keun Ho Ryu)

**요약** 이동 객체의 위치 정보는 차량 추적, 디지털 전장, 위치 기반 서비스, 텔레매틱스 등에 적용되며, 일정한 시간의 주기마다 측정된 위치 좌표가 시스템에 저장된다. 이 때 시스템에 저장되지 않은 질의 시점에서의 위치 정보를 추정하기 위해 선형 함수가 주로 사용된다. 그러나 선형 함수를 사용한 위치 추정에는 오차가 발생되므로, 위치 표현의 부정확성을 보완하기 위한 방법이 필요하다. 이 논문에서는 선형 위치 추정 함수의 오차를 감소시키기 위해 3차 스플라인 보간법의 적용을 제안한다. 먼저 2차원 공간에서 이동하는 객체의 위치 정보를 정의한다. 다음으로 3차 스플라인 보간법을 제안한 데이터 모델의 위치 추정에 적용하고, 위치 추정 연산 알고리즘을 기술한다. 마지막으로 제안한 위치 추정 연산 모델의 정확성을 실험한다. 실험 결과, 이 논문에서 제안한 위치 추정 연산은 적은 량의 위치 정보를 사용함에도 불구하고, 선형 함수를 사용한 경우보다 더 정확한 결과를 나타내었다. 제안 방법은 이동 객체의 위치 정보 관리를 위한 데이터 저장공간 및 통신비용을 감소시키는 장점을 가진다.

**키워드** : 이동 객체, 위치 추정, 3차 스플라인 보간법

**Abstract** Location information of mobile objects is applied to vehicle tracking, digital battlefields, location based services, and telematics. Their location coordinates are periodically measured and stored in the application systems. The linear function is mainly used to estimate the location information that is not in the system at the query time point. However, a new method is needed to improve uncertainties of the location representation, because the location estimation by linear function induces the estimation error. This paper proposes an application method of the cubic spline interpolation in order to reduce deviation of the location estimation by linear function. First, we define location information of the mobile object moving on the two-dimensional space. Next, we apply the cubic spline interpolation to location estimation of the proposed data model and describe algorithm of the estimation operation. Finally, the precision of this estimation operation model is experimented. The experimentation comes out more accurate results than the method by linear function, although the proposed location estimation function uses the small amount of information. The proposed method has an advantage that drops the cost of data storage space and communication for the management of location information of the mobile objects.

**Key words** : mobile objects, location prediction, cubic spline interpolation

### 1. 서론

이동 객체는 시간에 따라 객체의 위치 및 모양이 연속적으로 변경되는 시공간 데이터이다. 이동 객체의 위치 정보를 데이터베이스를 이용하여 관리하기 위해서는 매 초마다 측정된 위치 정보를 모두 저장해야 한다. 그러나 매 초마다의 변경된 위치 정보를 모두 저장할 경우 통신비용 및 저장비용이 매우 많이 든다. 이로 인해 데이터베이스를 이용한 이동 객체의 위치 정보 관리에

· 이 연구는 대학 IT연구센터 육성지원 및 한국과학재단 RRC 연구지원으로 수행되었음

† 정 회 원 : 청주과학기술대학교 컴퓨터학과 교수  
yeahn@cjnc.ac.kr

\*\* 종신회원 : 청주과학기술대학교 컴퓨터학과 교수  
jspark@cjnc.ac.kr

\*\*\* 종신회원 : 충북대학교 전기전자 및 컴퓨터공학부 교수  
khryu@dblab.chungbuk.ac.kr

논문접수 : 2003년 10월 7일

심사완료 : 2004년 5월 13일

서는 일정한 시간의 주기마다 측정된 위치 정보를 데이터베이스에 저장하게 된다[1-5]. 이 때 데이터베이스에 저장되지 않은 특정 질의 시점(query time point)에서의 위치 정보를 요구하는 질의에 대해 적절한 응답을 하지 못하는 문제점이 발생된다. 이것을 샘플링 오차(sampling error)로 인한 이동 객체의 불확실한 위치 정보라고 한다. 이동 객체의 불확실한 위치 정보로 인해 위치 기반 서비스, 모바일 상거래, 차량 위치 추적, 텔레매틱스 등의 응용 시스템 개발 시 데이터 모델링, 질의 처리, 인덱싱, 질의 결과의 부정확성 등과 같은 다양한 문제점들이 발생된다.

이와 같은 이유로 이동 객체의 불확실한 위치 정보 처리에 대한 연구들이 수행되었다. 지금까지 연구된 관련 연구들을 살펴보면, 실시간 차량 위치 추적을 위한 동적 속성의 불확실한 미래 위치 정보 처리 방법[6-9], 선형 회귀 함수를 이용한 이동 객체의 불확실한 위치 추정 모델[2,10], 선형 함수의 위치 표현의 부정확성을 처리하기 위한 확률 함수 기반의 불확실성 오차 모델 [1], 이동 객체의 부정확한 변화를 처리하기 위한 퍼지 집합 및 확률 이론 모델[5] 등이 있다. 그런데, 기존의 연구들은 대부분 이동 객체의 위치 정보를 이용하여 과거 및 미래의 위치 정보를 추정할 때 선형 함수를 사용하거나, 선형 회귀 함수를 적용하고 있다. 특히 선형 함수에 의한 위치 추정 결과의 부정확성을 보완하기 위해 불확실성 값의 오차 영역을 함께 제공할 것을 제안하고 있다. 그러나, 기존의 연구에서 제안된 불확실성 처리 방법들은 실험을 통해 정량화된 형태의 결과로 제시되지 못하고 있으며 방법론적인 제안에만 그치고 있다.

따라서, 이 논문에서는 불확실성 처리의 이론적인 방법론 대신, 그 동안 사용된 선형 함수에 의한 위치 추정 결과의 오차를 줄일 수 있는 방법으로 3차 스플라인 보간법을 적용한 연산 및 알고리즘을 제안한다. 이를 위해 먼저, 2차원 공간에서 이동하는 이동 객체의 위치 정보를 정의하고, 제안한 이동 객체의 위치 정보 모델을 이용하여 3차 스플라인 보간법을 적용한 위치 추정 함수 및 처리 알고리즘을 기술한다. 제안한 위치 추정 연산 알고리즘을 이용한 실험을 통해 위치 추정 결과의 오차 및 정확성을 분석한다. 실험 결과를 통해 데이터베이스에 저장된 동일한 위치 정보를 활용할 경우, 기존의 선형 함수를 사용한 경우보다 3차 스플라인 보간법을 적용한 경우 위치 추정의 정확성을 높일 수 있음을 보인다. 아울러 제안 방법이 위치 추정 결과의 정확성을 향상시킴으로서, 이동 객체의 위치 정보 관리를 위한 데이터 저장비용 및 위치 정보 송수신 비용을 감소시킬 수 있는 장점을 가짐을 알 수 있다.

이 논문의 구성은 다음과 같다. 먼저, 2절에서는 지금

까지 연구된 이동 객체의 불확실한 위치 정보 처리 방법에 대해 검토한다. 3절에서는 이동 객체의 위치 정보를 정의하고, 3차 스플라인 보간법을 이용한 이동 객체의 위치 추정 함수 및 알고리즘을 제시한다. 4절에서는 실험을 통해 이 논문에서 제안한 이동 객체의 위치 추정 연산의 오차 및 정확성을 분석한다. 마지막으로 5절에서는 결론을 맺는다.

## 2. 관련연구

이동 객체와 같은 시공간 데이터의 불확실성 발생 요인은 측정 오차와 샘플링 오차로 구분된다. 측정 오차(measurement error)는 아무리 정확한 측정 장비를 사용한다 할지라도 오차가 발생함을 말하며, 샘플링 오차는 객체의 위치 샘플을 얼마나 자주 획득하는가에 따라서 샘플 되지 않은 위치의 삽입 시 발생하는 오차의 비율이 달라짐을 의미한다. 이동 객체의 불확실성은 이동 객체의 데이터 모델, 질의 처리, 인덱싱, 질의 결과의 부정확성 등과 같은 문제점들을 발생시킨다[1]. 이와 같은 이유로 이동 객체의 불확실성을 줄이고, 불확실한 위치 정보에 대한 질의에 적절히 응답할 수 있는 방법이 연구되었다. 이 논문에서는 샘플링 오차로 인한 이동 객체의 불확실한 위치 정보 처리 방법에 관한 내용만을 다룬다. 이동 객체의 불확실한 위치 추정을 위해 그 동안 연구된 내용들을 살펴보고, 이들의 특징을 검토한다.

Sistla와 Wolfson[6-9,11]은 실시간 차량 위치 추적을 위한 동적 속성(dynamic attribute)의 불확실성 처리 방법을 제안하였다. 동적 속성은 시간의 변화에 따라 객체의 속성 값이 일정한 함수 형태로 표현되는 것을 의미한다. 특히, Sistla와 Wolfson이 제안한 위치 변화 함수는 부분 구간을 직선의 형태로 표현하는 선형 함수의 특성을 보이고 있다. 이들은 불확실성 값을 이동 객체가 가지는 하나의 속성 값으로 사용하는 방법을 제안하였다. 임의의 이동 객체  $L$ 이 가지는 불확실성 값을  $L$ .uncertainty로 표현하고, " $L$ .uncertainty = 1mile"과 같은 형태로 이동 객체의 속성 정보로 객체가 가질 수 있는 위치 편차의 범위를 제공하고 있다. 예를 들어 "현재 객체  $L$ 의 위치는 어디인가?" 라는 질의에 대해 "현재  $L$ 의 위치는  $(x, y)$ 이고 위치 편차는 1 mile 이다."라는 응답을 제시한다. 그러나, 동적 속성의 불확실성 처리 모델에서는 객체의 현재 위치 정보만을 데이터베이스에 저장하므로, 과거 질의에 대한 이력 정보의 제공이 어렵다. 아울러 데이터 모델링 단계에서 객체의 불확실한 위치 편차 값을 미리 지정해 주어야만 하는 제약사항을 가진다.

선형 회귀 모형의 위치 추정 모델[2,10]에서는 데이터베이스에 저장되지 않은 이동 객체의 위치 추정을 위해

선형 회귀 함수(linear regression function)를 이용한 위치 추정 연산 모델을 제시하였다. 이를 위해 이동 지식베이스(movement knowledge base)라는 이름의 위치 정보 저장 구조를 제시하였으며, 불확실성 값을 제공하기 위해 타원 방정식을 이용한 오차 범위 제공 방안을 소개하였다. 질의 처리를 위한 상위 집합(superset), 하위 집합(subset) 개념도 추가로 제시하였지만, 적용 알고리즘이나 구현 사례를 제시하지 않고 있다. 특히 선형 회귀 함수를 이용한 모델에서는 센서 시스템에서 관측된 이동 객체의 위치 정보를 그대로 저장하지 않고, 근사 값을 저장하므로 정확한 이동 궤적의 정보를 검색할 수 없는 문제점이 발생된다.

확률 함수 기반의 불확실성 오차의 정량화 모델[1]에서는 관계 데이터베이스 기반의 이동 객체의 위치 표현 구조를 제시하였다. Pfoser는 데이터베이스에 저장된 두 점의 위치 정보를 직선 형태로 연결한 라인 세그먼트(line segment)를 이용하여 이동 궤적을 표현하였으며, 이 때 직선의 함수로 표현되는 부분 구간의 오차 값의 범위를 제공하기 위해 확률 함수를 사용할 것을 제안하였다. 따라서 실제 과거 및 미래의 위치 정보의 추정은 부분 구간의 선형 함수로 나타낸 것과 동일한 결과를 가진다. 그러나 제안한 확률 함수를 이용한 적용 사례를 제시하지 못하므로 방법론적인 제안에만 그치고 있다.

퍼지 집합 및 확률 이론을 적용한 모델[5]에서는 시공간 시나리오를 나타내는 이동 객체의 위치 변화에 대한 몇 가지 예를 들어 확률과 퍼지 집합 이론을 이용하여 부정확한 변화(indeterminate change) 과정을 모델링하였다. 여기에서 Pfoser는 시공간 불확실성의 요소를 공간 객체, 시간 점, 시간 구간으로 구분하였으며, 서로 다른 네 가지 종류의 시공간 시나리오와 변화에 대해 퍼지 집합 또는 확률을 이용한 변화 함수의 구조를 제시하였다. 그러나 불확실성을 포함하는 시공간 데이터의 표현 방법이 제시되지 않았으며, 실제 적용 가능한 퍼지 및 확률 함수를 제시하지 못하고 있다.

### 3. 3차 스플라인 보간법을 적용한 위치 정보 추정

이 논문에서 대상으로 하는 이동 객체의 위치 정보를 정의하고, 3차 스플라인 보간법을 적용한 위치 추정 방법을 기술한다.

#### 3.1 이동 객체의 위치 정보

이동 객체(mobile objects)는 실세계에 존재하는 임의의 시공간 객체로서 시간이 흐름에 따라 스스로 그 위치를 변화하면서 이동 가능한 객체를 말한다. 이동 객체에는 시간의 변화에 따라 객체의 위치 값만 변화되는 점(point) 객체와 위치 및 모양이 함께 변화되는 영역

(region) 객체가 있다. 점 객체에는 자동차, 동물, 사람, 비행기, 배 등의 예가 있으며, 영역 객체에는 태풍 영역, 오염 지역, 화재 지역, 암 세포 등의 예가 있다[12-15]. 그림 1은 이동 객체의 이동 과정을 점 객체와 영역 객체로 구분하여 표현한 것이다.

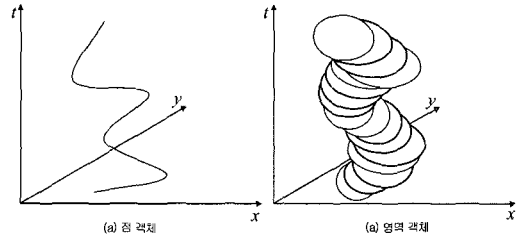


그림 1 이동 객체의 유형 및 위치 변화

그림 1과 같이 이동하는 이동 객체의 위치 정보 관리 는 위치 정보의 획득 방법에 따라, 사건 지향(event driven) 시스템과 관측 지향(observation driven) 시스템의 두 가지 형태로 분류된다. 사건 지향 시스템은 이동 객체의 속도나 방향의 변화를 자동으로 검출하는 것이 가능한 경우의 시스템을 말한다. 관측 지향 시스템은 GPS와 같은 센서 시스템을 이용하여 정기적인 시간의 간격에서 정렬된 순서대로 객체의 위치를 획득하는 시스템을 말한다[2].

이 논문에서 취급하는 이동 객체의 위치 정보는 관측 지향 시스템을 기반으로 하며 다음과 같은 특징을 가진다. 먼저, 이동 객체는 2차원 공간에서 이동하는 점 객체를 대상으로 한다. 이동 객체의 이동 위치 정보는 GPS와 같은 센서 시스템으로부터 획득한다. 이 때, 이동 객체의 위치 변화 관측은 모든 객체에 대해 독립적으로 발생시킬 수 있다. 그러나 하나의 객체에 대한 관측은 정규적인 시간 간격에 따라 획득되며 정렬 순서대로 입력된다. 센서 시스템으로부터 관측된 이동 객체의 위치 관련 정보는  $(Oid, t, v)$ 와 같이 세 쌍의 집합으로 표현되는 상태들의 정렬된 순서로 전송 받는다.  $Oid$ 는 점으로 표현되는 객체의 식별자이고,  $t$ 는 시간을 나타내는 타임 스탬프이며,  $v$ 는 객체의 위치에 해당하는 좌표 값을 나타낸다. 여기에서,  $t$ 를 나타내는 시간 속성은 유효 시간(valid time)만을 고려하며, 처리 시간(transaction time)은 고려하지 않는다. 처리 시간은 유효 시간 즉, 관측 시간과 동일하다고 가정한다.

#### 3.2 위치 추정을 위한 3차 스플라인 보간법의 적용

센서 시스템으로부터 획득된 이동 객체의 위치 변화 정보는  $(mo_{id}, t_i, x_i, y_i)$ 의 연속적인 구조로 데이터베이스에 저장되며, 이를 moving\_history라고 지칭한다. mov-

ing\_history에서  $mo_{id}$ 는 객체의 식별자이고,  $t_i$ 는 관측된 시점을 나타내며,  $(x_i, y_i)$ 는 시점  $t_i$ 에서의 위치 좌표이다. 이 때 moving\_history에 저장되지 않은 특정한 시점에서의 위치 정보를 직접 검색하지 못하는 문제점이 발생된다. 이를 해결하기 위해 이 논문에서는 3차 스플라인 보간법을 적용한 이동 객체의 위치 추정 방법을 제안한다.

3차 스플라인 보간법(cubic spline interpolation)은 데이터 점들의 집합을 다항식 곡선으로 나타낸 것으로, CAD/CAM, 컴퓨터 그래픽스 시스템 등에서 주로 사용한다. 이것의 목적은 주어진 점들을 지나면서 오차에 영향을 받지 않는 유연한 곡선을 그리는 것이다[16,17].  $N+1$ 개의 점으로 구성된 집합  $\{(x_k, y_k)\}_{k=0}^N$ 이 있을 때,  $S_k(x) = s_{k,0} + s_{k,1}(x-x_k) + s_{k,2}(x-x_k)^2 + s_{k,3}(x-x_k)^3$  [16,18]과 같은 3차 다항식의 계수를 구하여 식을 완성한다.

그림 2는 3차 스플라인 보간법을 이동 객체의 위치 변화 과정에 적용한 것이다. 이 논문에서는 3차 보간식을 이동 객체의 위치 추정에 적용하기 위해 집합  $\{(x_k, y_k)\}_{k=0}^N$ 을  $\{(t_i, v_i)\}_{i=0}^n$ 으로 대체하여 사용한다. 이 때  $v_i = (x_i, y_i)$ 를 의미한다.

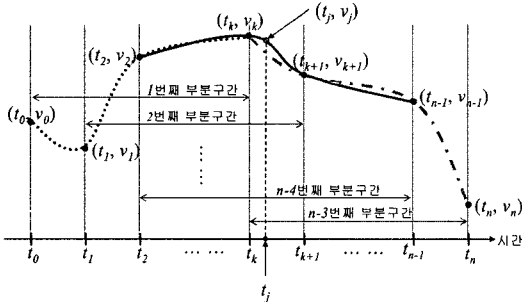


그림 2 3차 스플라인 보간법의 적용

그림 2와 같이  $n+1$ 개의 위치 정보로 이루어진 집합  $\{(t_i, v_i)\}_{i=0}^n$ 이 존재할 때  $n$ 개의 부분구간 3차 다항식을 구함으로써, 연속적으로 존재하는 모든 구간에서의 위치 좌표 값을 구할 수 있다. 임의의 시점  $t_j$ 에 대해  $t_j \in I_k$ ,  $I_k = [t_k, t_{k+1}]$ ,  $t_k < t_j < t_{k+1}$ 의 조건을 만족할 때  $t_j$ 에서의 위치 좌표  $v_j$ 를 구하는 부분구간 3차 스플라인 보간 다항식은  $\{(t_{k-1}, v_{k-1}), (t_k, v_k), (t_{k+1}, v_{k+1}), (t_{k+2}, v_{k+2})\}$ 의 네 점을 이용하여 구한다. 실제 위치 좌표를 구하기 위해서는  $x$ 와  $y$ 의 값을 구하기 위한 개별적인 두개의 3차 다항식이 필요하다. 따라서,  $x$ 좌표는  $\{(t_{k-1}, x_{k-1}), (t_k, x_k), (t_{k+1}, x_{k+1}), (t_{k+2}, x_{k+2})\}$ ,  $y$ 좌표는  $\{(t_{k-1}, y_{k-1}),$

$(t_k, y_k), (t_{k+1}, y_{k+1}), (t_{k+2}, y_{k+2})\}$ 을 이용하여 구한다.

### 3.3 과거 시점의 위치 정보 추정 연산

이동 객체의 과거 시점의 모든 위치 정보를 모델링하는 함수를 과거 위치 추정 함수(past location change function)라 정의하고, PCF로 표기한다. PCF는  $t_p \mapsto (x_t, y_t)$ 의 구조를 가지며, 입력 값은  $t_p$ , 출력 값은 시점  $t_p$ 에서의 위치 좌표 값  $(x_{t_p}, y_{t_p})$ 이다. 이 때  $t_p$ 는  $t_0 \leq t_p \leq t_n$ 의 조건을 만족하며,  $t_p$ 는 moving\_history의 원소가 될 수도 있고 그렇지 않을 수도 있다. 만약,  $t_p$ 가 moving\_history의 원소이면 저장된 위치 정보를 검색하여 결과를 출력하고, 그렇지 않으면 과거 위치 추정 함수 PCF에 의해 계산된 위치 좌표 값이 출력된다.

moving\_history에는 하나의 이동 객체에 대해  $n+1$ 개의 이력 위치 좌표가 저장되며,  $\{(t_0, x_0, y_0), (t_1, x_1, y_1), \dots, (t_{n-1}, x_{n-1}, y_{n-1}), (t_n, x_n, y_n)\}$ 의 형태로 표현할 수 있다. 이동 객체의 이력 위치 정보가 저장된 moving\_history의 전체 유효 시간 간격이  $T_a = [t_0, t_n]$ 이라 하자. 여기에서  $t_0$ 는 저장된 유효 시간의 시작 시점을 의미하고,  $t_n$ 는 마지막 시점을 나타낸다.  $T_a$ 는  $I_0 = [t_0, t_1]$ ,  $I_1 = [t_1, t_2], \dots, I_{n-2} = [t_{n-2}, t_{n-1}], I_{n-1} = [t_{n-1}, t_n]$ 과 같이  $n$ 개의 부분 시간 구간  $I_k$ 로 분할할 수 있으며, 이 때  $k=0, 1, \dots, n-1$ 이 된다. 과거 위치 추정 함수 PCF는 질의 시점  $t_p$ 가  $t_p \in I_k$ ,  $I_k = [t_k, t_{k+1}]$ 의 조건을 만족할 때의 위치 좌표  $(x_{t_p}, y_{t_p})$ 를 연산한다.

PCF는 3.2절에서 언급한 3차 다항식  $S_k(x) = s_{k,0} + s_{k,1}(x-x_k) + s_{k,2}(x-x_k)^2 + s_{k,3}(x-x_k)^3$  [16,18]을  $x_{t_p}$ 를 구하는 함수  $F_{PS_x}(t_p)$ 와  $y_{t_p}$ 를 구하는 함수  $G_{PS_y}(t_p)$ 로 구분하여 연산한다. 먼저, 질의 시점  $t_p$ 에서  $x$  좌표 값  $x_{t_p}$ 를 구하는 함수  $F_{PS_x}(t_p)$ 는 식 (1)과 같이 구한다.

$$F_{PS_x}(t_p) = s_{k,0} + s_{k,1}(t_p - t_k) + s_{k,2}(t_p - t_k)^2 + s_{k,3}(t_p - t_k)^3 \quad (1)$$

식 (1)의  $F_{PS_x}(t_p)$ 는  $t_p \in [t_k, t_{k+1}]$ ,  $t_k \leq t_p \leq t_{k+1}$ 의 조건을 만족하는  $t_p$  시점에서의  $x$ 축 과거 위치 추정 함수이다. 식 (1)에는  $s_{k,0}, s_{k,1}, s_{k,2}, s_{k,3}$ 의 네 개의 미지수가 존재하며, 식 (2)와 같이 구한다. 식 (2)는 [16]에서 제시된 방법을 식 (1)에 적용한 것이다.

$$s_{k,0} = x_b, \quad s_{k,1} = d_k - \frac{h_k(2m_k + m_{k+1})}{6},$$

$$s_{k,2} = \frac{m_k}{2}, \quad s_{k,3} = \frac{m_{k+1} - m_k}{6h_k},$$

$$h_k = t_{k+1} - t_k, \quad d_k = \frac{x_{k+1} - x_k}{h_k}, \quad k=0, 1, \dots, n-1 \quad (2)$$

그런데 식 (2)에서는 아직  $m_k$  값이 결정되지 않았다.  $m_k$  값을 구하기 위해서는 다음 선형 연립방정식의 해를 구한다. 이 방법은 [16]에서 제시된 방법을 적용한 것이다.

$$\begin{aligned} (3h_0 + 2h_1)m_1 + h_1m_2 &= u_1 \\ h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} &= u_k \\ k &= 2, 3, \dots, n-2, \\ h_{n-2}m_{n-2} + (2h_{n-1} + 3h_{n-1})m_{n-1} &= u_{n-1} \end{aligned}$$

이 선형 연립방정식의 해를 구하기 위한  $u_k$  값은 다음과 같이 계산한다. 식의 풀이과정은 [16]을 참고하기 바란다.

$$u_k = 6(d_k - d_{k-1}), \quad k = 1, 2, \dots, n-1$$

이제 위의 선형 연립방정식을 풀면  $m_1, m_2, \dots, m_{n-1}$ 을 구할 수 있다. 마지막으로  $m_0$ 와  $m_n$ 을 구하면 된다.  $m_0$ 와  $m_n$ 을 구하는 전략을 3차 스플라인 함수의 끝점 제약조건이라 한다. 이 끝점 제약조건을 구하는 전략에 따라 다양한 형태의 3차 스플라인 함수가 존재한다 [16-18]. 이 논문에서는 포물선으로 끝나는 3차 스플라인 보간법을 이용하여 과거 위치를 추정한다. 포물선으로 끝나는 3차 스플라인의 끝점 제약 조건[16]은 다음과 같다.

$$m_0 = m_1, \quad m_n = m_{n-1}$$

이제  $m_0, m_1, \dots, m_n$  값을 모두 구할 수 있으므로,  $m_k$  값을 식 (2)에 대입하여  $s_{k,0}, s_{k,1}, s_{k,2}, s_{k,3}$  값을 구한 후 마지막으로 식 (1)에 대입하면,  $t_p$  시점에서의 위치 좌표  $x_{t_p}$ 를 구할 수 있다. 다음으로  $t_p$ 에서  $y$  좌표 값  $y_{t_p}$ 를 구하는 함수  $G_{FS_s}(t_p)$ 는 다음 식 (3)과 같이 구한다.

$$G_{FS_s}(t_p) = s_{k,0} + s_{k,1}(t_p - t_k) + s_{k,2}(t_p - t_k)^2 + s_{k,3}(t_p - t_k)^3 \quad (3)$$

식 (3)의  $G_{FS_s}(t_p)$ 는  $t_p \in [t_k, t_{k+1}]$ ,  $t_k \leq t_p \leq t_{k+1}$ 의 조건을 만족하는  $t_p$  시점에서의  $y$ 축 과거 위치 추정 함수이다. 식 (3)에서  $s_{k,0}, s_{k,1}, s_{k,2}, s_{k,3}$ 을 구하는 방법은  $F_{FS_s}(t_p)$ 에서의 미지수를 구하는 방법과 동일하며 앞의 식 (2)를 사용하여 구한다. 단지 식 (2)에서  $s_{k,0} = x_k$  값을  $s_{k,0} = y_k$ 로 바꾸어 계산하면 된다.  $G_{FS_s}(t_p)$ 를 구하는 방법은  $F_{FS_s}(t_p)$ 를 구하는 방법과 동일하다.

### 3.4 미래 시점의 위치 정보 추정 연산

미래 위치 추정 함수  $FCF$ (future location change function)는  $t_f \mapsto (x_{t_f}, y_{t_f})$ 의 구조를 가진다. 입력 값은  $t_f$ , 출력 값은  $t_f$  시점에서의 위치 좌표 값  $(x_{t_f}, y_{t_f})$ 이다. 이 때  $t_f$ 는  $t_n < t_f$ 의 조건을 만족하며,  $t_f$ 는 *moving\_*

*history*의 원소가 아니다.

$FCF$ 는 3차 다항식  $S_k(x) = s_{k,0} + s_{k,1}(x - x_k) + s_{k,2}(x - x_k)^2 + s_{k,3}(x - x_k)^3$  [16,18]을  $x_{t_f}$ 를 구하는 함수  $F_{FS_s}(t_p)$ 와  $y_{t_p}$ 를 구하는 함수  $G_{FS_s}(t_p)$ 로 구분하여 연산한다.

$FCF$ 는  $t_f$  시점에서의  $x$  좌표를 구하는  $F_{FS_s}(t_f)$ 와  $y$  좌표를 구하는  $G_{FS_s}(t_f)$ 로 나누어 연산한다.

질의 시점  $t_f$ 가  $t_n < t_f$ 의 조건을 만족할 때,  $x_{t_f}$ 를 구하는 함수  $F_{FS_s}(t_f)$ 는 식 (4)와 같다.

$$F_{FS_s}(t_f) = s_{k,0} + s_{k,1}(t_f - t_k) + s_{k,2}(t_f - t_k)^2 + s_{k,3}(t_f - t_k)^3 \quad (4)$$

식 (4)는  $t_f \notin [t_0, t_n]$ ,  $t_n < t_f$ 의 조건을 만족하는  $t_f$  시점에서의  $x$ 축 미래 위치 추정 함수이다. 이 때  $t_k$ 는  $t_k \in [t_{n-3}, t_n]$ ,  $t_k < t_f$ 의 조건을 만족한다. 식 (4)를 구하는 방법은 3.3절의 과거 위치 추정 함수를 구하는 식 (2)와 동일하게 적용된다. 단지  $m_k$ 를 구하기 위한 선형 연립방정식과  $m_0$ 와  $m_n$ 을 구하기 위한 끝점 제약조건만 다르게 적용된다. 먼저  $m_k$ 를 구하기 위한 선형 연립방정식은 다음과 같다. 이 방법은 [16]에서 제시된 방법을 적용한 것이다.

$$(3h_0 + 2h_1 + \frac{h_0^2}{h_1})m_1 + (h_1 - \frac{h_0^2}{h_1})m_2 = u_1$$

$$h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k, \quad k = 2, 3, \dots, n-2$$

$$(h_{n-2} - \frac{h_{n-1}^2}{h_{n-2}})m_{n-2} + (2h_{n-2} + 3h_{n-1} + \frac{h_{n-1}^2}{h_{n-2}})m_{n-1} = u_{n-1}$$

이 선형 연립방정식의 해를 구하면  $m_k$ 를 구할 수 있다. 마지막으로  $m_0$ 와  $m_n$ 을 구하기 위한 끝점 제약 조건은 다음과 같이 외삽 스플라인[16]을 적용한다.

$$m_0 = m_1 - \frac{h_0(m_2 - m_1)}{h_1}$$

$$m_n = m_{n-1} + \frac{h_{n-1}(m_{n-1} - m_{n-2})}{h_{n-2}}$$

이제  $m_k$  값을 3.3절의 식 (2)에 대입하여  $s_{k,0}, s_{k,1}, s_{k,2}, s_{k,3}$  값을 구한 후 식 (4)에 대입하여  $x_{t_f}$ 를 구한다. 다음으로  $t_f$ 에서  $y$  좌표 값  $y_{t_f}$ 를 구하는 함수  $G_{FS_s}(t_f)$ 는 식 (5)와 같다.

$$G_{FS_s}(t_f) = s_{k,0} + s_{k,1}(t_f - t_k) + s_{k,2}(t_f - t_k)^2 + s_{k,3}(t_f - t_k)^3 \quad (5)$$

$G_{FS_s}(t_f)$ 는  $t_f \notin [t_0, t_n]$ ,  $t_n < t_f$ 의 조건을 만족하는  $t_f$  시점에서의  $y$ 축 미래 위치 추정 함수이다. 이 때  $t_k$ 는  $t_k \in [t_{n-3}, t_n]$ ,  $t_k < t_f$ 의 조건을 만족한다.  $G_{FS_s}(t_f)$ 를 구하는 방법은  $F_{FS_s}(t_f)$ 를 구하는 방법과 동일하다.

3.5 알고리즘

과거 위치 추정 함수  $F_{PS_s}(t_p)$  와  $G_{PS_s}(t_p)$  를 구하는 처리 과정은 알고리즘 1과 같다.

알고리즘 1 이동 객체의 과거 위치 추정 함수

```

Algorithm PastSpline(mo_id, t_p)
Input: mo_id(객체 식별자); t_p(과거의 임의의 시점);
Output: past_location(시점 t_p에서의 (x, y) 좌표 값);
Begin
    moving_history에서 mo_id에 대해  $t_i \leq t_p \leq t_{i+1}$ 의 조건을 만족하는 위치 좌표
    ( $t_{i-1}, x_{i-1}, y_{i-1}$ ), ( $t_i, x_i, y_i$ ), ( $t_{i+1}, x_{i+1}, y_{i+1}$ ),
    ( $t_{i+2}, x_{i+2}, y_{i+2}$ ) 검색;
     $t[0] \leftarrow t_{i-1}$ ;  $t[1] \leftarrow t_i$ ;  $t[2] \leftarrow t_{i+1}$ ;  $t[3] \leftarrow t_{i+2}$ ;
     $x[0] \leftarrow x_{i-1}$ ;  $x[1] \leftarrow x_i$ ;  $x[2] \leftarrow x_{i+1}$ ;  $x[3] \leftarrow x_{i+2}$ ;
     $y[0] \leftarrow y_{i-1}$ ;  $y[1] \leftarrow y_i$ ;  $y[2] \leftarrow y_{i+1}$ ;  $y[3] \leftarrow y_{i+2}$ ;
     $x_{t_p} \leftarrow SplineInterpolation(t, x, t_p)$ ; //  $F_{PS_s}(t_p)$  함수 호출
     $y_{t_p} \leftarrow SplineInterpolation(t, y, t_p)$ ; //  $G_{PS_s}(t_p)$  함수 호출
    past_location[0]  $\leftarrow x_{t_p}$ ;
    past_location[1]  $\leftarrow y_{t_p}$ ;
    Return past_location;
End
    
```

알고리즘 1의 PastSpline은 임의의 이동 객체의 식별자 *mo\_id*와 과거의 임의의 시점 *t\_p*를 입력받은 후, *t\_p* 시점에서의 위치 좌표  $x_{t_p}$ 와  $y_{t_p}$ 를 추정하여 결과로 반환하는 함수이다. 먼저, 위치 추정을 위한 3차 스플라인 보간법을 적용하기 위해 *moving\_history*에 저장된 위치 정보 중, *mo\_id* 객체에 대해  $t_i \leq t_p \leq t_{i+1}$ 의 조건을 만족하는 4점의 위치 좌표 ( $t_{i-1}, x_{i-1}, y_{i-1}$ ), ( $t_i, x_i, y_i$ ), ( $t_{i+1}, x_{i+1}, y_{i+1}$ ), ( $t_{i+2}, x_{i+2}, y_{i+2}$ )를 검색한다. 그리고, ( $x_i, y_i$ )를 구하기 위해 위치 추정 함수  $F_{PS_s}(t_p)$ ,  $G_{PS_s}(t_p)$ 를 처리하는 SplineInterpolation 함수를 호출한다. 함수 호출에 사용되는 입력 값 *t*는 스플라인 함수를 구하기 위한 네 점의 시간 값이고, *x*와 *y*는 스플라인 함수를 구하기 위한 네 점의 *x* 좌표와 *y* 좌표 값이다.

위치 추정을 위한 스플라인 함수를 처리하는 과정은 다음 알고리즘 2와 같다.

알고리즘 2 위치 추정을 위한 스플라인 함수

```

Algorithm SplineInterpolation(time, loc, t_k)
Input: ① 3차 스플라인 다항식을 구하는데 사용되는 좌표 쌍의 집합
    (time[0], loc[0]), (time[1], loc[1]), (time[2], loc[2]),
    (time[3], loc[3])
    ② 임의의 질의 시점 t_k
Output: result(질의 시점 t_k에서의 x 또는 y 좌표 값)
    
```

```

N: 입력 좌표 쌍의 수(N의 값은 4이상, 이 논문에서는 N=4로 실험함)
Begin
    For i  $\leftarrow$  0, N-1
        h[i]  $\leftarrow$  time[i+1] - time[i];
    For i  $\leftarrow$  0, N-3
        m[i]  $\leftarrow$   $6 * ((loc[i+2] - loc[i+1]) / h[i+1] - (loc[i+1] - loc[i]) / h[i])$ ;
        m[i]  $\leftarrow$  h[i];
        m2[0]  $\leftarrow$   $3 * (h[0] + 2 * h[1])$ ;
        m2[1]  $\leftarrow$   $3 * (h[2] + 2 * h[1])$ ;
        m3[0]  $\leftarrow$  h[1];
    For i  $\leftarrow$  1, N-3
        m2[i]  $\leftarrow$  m2[i] -  $((m[i] / m2[i-1]) * m3[i-1])$ ;
        m4[i]  $\leftarrow$  m4[i] -  $((m[i] / m2[i-1]) * m4[i-1])$ ;
        s[2]  $\leftarrow$  m4[1] / m2[1];
        s[1]  $\leftarrow$   $(m4[0] - m3[0] * s[2]) / m2[0]$ ;
    If (과거의 위치 추정) Then //포물선 스플라인의 끝점 제약조건 적용
        s[0]  $\leftarrow$  s[1];
        s[3]  $\leftarrow$  s[2];
    Else //미래의 위치 추정, 외삽 스플라인의 끝점 제약조건 적용
        s[0]  $\leftarrow$   $s[1] - h[0] * (s[2] - s[1]) / h[1]$ ;
        s[3]  $\leftarrow$   $s[2] + h[2] * (s[2] - s[1]) / h[1]$ ;
    For i  $\leftarrow$  0, N-2
        a[i]  $\leftarrow$   $(s[i+1] - s[i]) / 6 * h[i]$ ;
        b[i]  $\leftarrow$  s[i] / 2;
        c[i]  $\leftarrow$   $(loc[i+1] - loc[i]) / h[i] - (2 * h[i] * s[i] + h[i] * s[i+1]) / 6$ ;
        d[i]  $\leftarrow$  loc[i];
    If (time[i] < t_k < time[i+1]) Then
        result  $\leftarrow$   $d[i] + (t_k - time[i]) * (c[i] + (t_k - time[i]) * (b[i] + a[i] * (t_k - time[i])))$ ;
    Return result;
End
    
```

알고리즘 2의 SplineInterpolation 함수에는 3차 스플라인 다항식을 구하는데 필요한 네 점의 좌표 쌍의 집합과 임의의 질의 시점 *t\_k*가 입력된다. 네 점의 좌표 쌍은 (*time*<sub>0</sub>, *loc*<sub>0</sub>), (*time*<sub>1</sub>, *loc*<sub>1</sub>), (*time*<sub>2</sub>, *loc*<sub>2</sub>), (*time*<sub>3</sub>, *loc*<sub>3</sub>)의 형태로 구성된다. 여기에서 *time*은 *moving\_history* 테이블에 저장된 시간 값이고, *loc*는 *time* 시점에 이동 객체가 위치하는 *x* 또는 *y*의 좌표 값을 의미한다. 이 함수의 결과 값은 *result* 변수에 저장된 값으로, 입력 좌표 쌍으로부터 생성된 스플라인 함수를 이용한 질의 시점 *t\_k*의 값이다. SplineInterpolation 함수의 알고리즘은 먼저, 스플라인 다항식의 네 개의 계수 *a*<sub>*i*</sub>, *b*<sub>*i*</sub>, *c*<sub>*i*</sub>, *d*<sub>*i*</sub>를 계산하고, 구해진 계수들과 입력된 네 점의 좌표 쌍을 이용하여 질의 시점 *t\_k*의 *x* 또는 *y* 값을 구한 후 그 결과 값을 반환한다.

미래 위치 추정 함수  $F_{FS_s}(t_f)$  와  $G_{FS_s}(t_f)$  를 구하는 알고리즘은 다음과 같다.

알고리즘 3의 FutureSpline은 임의의 이동 객체의 식별자 *mo\_id*와 미래의 임의의 시점 *t\_f*를 입력받은 후,

알고리즘 3 이동 객체의 미래 위치 추정 함수

```

Algorithm FutureSpline(mo_id, t_f)
Input: mo_id(객체 식별자); t_f(미래의 임의의 시점)
Output: future_location(시점 t_f에서의 (x, y) 좌표 값)
Begin
    moving_history에서 mo_id에 대해  $t_n < t_f$ 의 조건을 만족하는
    위치 좌표
    ( $t_{n-3}, x_{n-3}, y_{n-3}$ ), ( $t_{n-2}, x_{n-2}, y_{n-2}$ ), ( $t_{n-1}, x_{n-1}, y_{n-1}$ ),
    ( $t_n, x_n, y_n$ ) 검색;
     $t[0] \leftarrow t_{n-3}$ ;  $t[1] \leftarrow t_{n-2}$ ;  $t[2] \leftarrow t_{n-1}$ ;  $t[3] \leftarrow t_n$ ;
     $x[0] \leftarrow x_{n-3}$ ;  $x[1] \leftarrow x_{n-2}$ ;  $x[2] \leftarrow x_{n-1}$ ;  $x[3] \leftarrow x_n$ ;
     $y[0] \leftarrow y_{n-3}$ ;  $y[1] \leftarrow y_{n-2}$ ;  $y[2] \leftarrow y_{n-1}$ ;  $y[3] \leftarrow y_n$ ;
     $x_{t_f} \leftarrow \text{SplineInterpolation}(t, x, t_f)$ ; //  $F_{FS}(t_f)$  함수 호출
     $y_{t_f} \leftarrow \text{SplineInterpolation}(t, y, t_f)$ ; //  $G_{FS}(t_f)$  함수 호출
    future_location[0]  $\leftarrow x_{t_f}$ ;
    future_location[1]  $\leftarrow y_{t_f}$ ;
    Return future_location;
End
    
```

*t\_f* 시점의 (*x*, *y*) 좌표인  $x_{t_f}$ 와  $y_{t_f}$ 를 결과로 반환하는 함수이다.

스플라인 함수를 적용하기 위해 *moving\_history*에 저장된 위치 정보 중, *mo\_id* 객체에 대해  $t_n < t_f$ 의 조건을 만족하는 위치 좌표 ( $t_{n-3}, x_{n-3}, y_{n-3}$ ), ( $t_{n-2}, x_{n-2}, y_{n-2}$ ), ( $t_{n-1}, x_{n-1}, y_{n-1}$ ), ( $t_n, x_n, y_n$ )를 검색한다. 다음으로 ( $x_{t_f}, y_{t_f}$ )를 구하는 함수  $F_{FS}(t_f)$ 와  $G_{FS}(t_f)$ 를 호출한다.

4. 실험

제안한 위치 추정 연산 모델의 특성을 분석하기 위해 동일한 실험 데이터를 이용하여 선형 함수와 3차 스플라인 함수를 이용한 위치 추정 실험을 수행하고, 각 실험에서 산출된 위치 추정 결과를 비교 분석하였다. 확률을 이용한 방법은 관련연구에서 구체적인 적용 방법이 언급되지 않았기 때문에 실험을 통한 비교가 어렵다. 따라서 선형 함수와 비교 실험한 결과를 제시한다. 먼저, 실험 대상이 되는 이동 객체는 도로에서 이동하는 차량으로 한정한다. 위치 추정에 사용된 차량의 이동 궤적은 서울시 데이터베이스의 주요 간선도로 중 세 곳을 채택하였으며 그림 3과 같다. 하나의 이동 구간에서 이동 객체의 움직임은 연속적으로 발생된다.

위치 추정 연산에 의해 추정된 위치 좌표 값과 실제 위치 좌표와의 편차(deviation)를 구하기 위해서는 실험 대상이 되는 이동 궤적에서 이동 객체의 완전한 이력 위치 좌표 값이 필요하다. 따라서, Traj\_1, Traj\_2, Traj\_3에서 이동하는 객체의 완전한 이동 경로를 나타낼 수 있는 이력 위치 좌표 값을 미리 선정하여 각각의

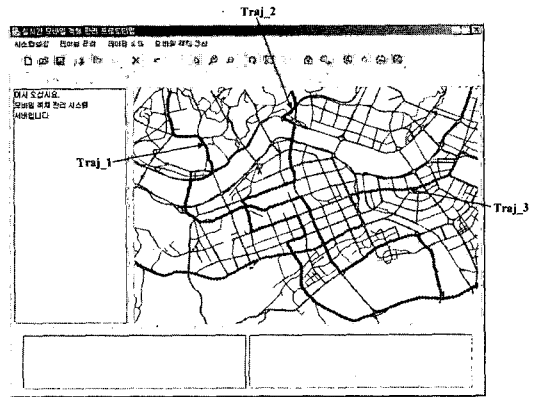


그림 3 위치 추정 실험에 사용된 이동 궤적

이동시간과 위치 좌표 값을 세 개의 서로 다른 임시 테이블에 미리 저장하였다. 완전한 이동 경로 집합은  $Traj_k = \{(t_i, x_i, y_i)\}_{i=0}^n$  형태로 표기하며, *k*는 이동 구간의 식별자로서 여기에서는 1, 2, 3의 값을 가진다. *i*는 유효 시간의 순서를 나타내며 모두 *n*+1개의 위치 정보로 구성됨을 나타낸다. 즉  $t_i = t_0$ 는 이력 테이블에 저장된 가장 최초의 시간 값을 나타내고,  $t_i = t_n$ 은 가장 마지막에 저장된 시간 값을 의미한다. ( $t_i, x_i, y_i$ )는  $t_i$  시점의 위치 좌표를 나타낸다. 실험을 위해 이동 구간 Traj\_1, Traj\_2, Traj\_3 각각에 대해 생성한 완전한 이동 경로의 집합은  $Traj_1 = \{(t_i, x_i, y_i)\}_{i=0}^{205}$ ,  $Traj_2 = \{(t_i, x_i, y_i)\}_{i=0}^{192}$ ,  $Traj_3 = \{(t_i, x_i, y_i)\}_{i=0}^{138}$ 이다. 이력 위치 정보 테이블에서 사용된 좌표 값은 Java 프로그래밍 언어의 좌표 체계를 사용하였다. 좌표의 기본 단위는 픽셀이고, 저장된 위치 샘플은 1 픽셀당 실제 거리로 약 30m 정도에 해당된다. 이동 객체의 이동 경로를 화면에 보이기 위한 지도는 ZEUS 공간 객체 관리 시스템[19]에 저장된 서울시 데이터베이스의 맵 이미지를 활용하였다.

4.1 위치 추정 실험

4.1.1 과거의 위치 추정 연산 결과

과거 위치 추정 실험을 위해 세 이동 구간의 완전한 이동 경로 집합 Traj<sub>1</sub>, Traj<sub>2</sub>, Traj<sub>3</sub>에 대해 위치 샘플링 간격을 2에서 16까지 2씩 증가시키면서 새로운 위치 샘플을 발생시켰다. 각각의 이동 구간에 대해 서로 다른 8개의 위치 샘플을 생성하여 개별적인 테이블에 저장하였다. 생성된 위치 정보 샘플 집합은 표 1과 같다.

표 1에서 위치 샘플 집합은  $S_{i,j}$ 로 표기한다. 여기에서 *i*는 이동 구간의 식별자로  $i=1,2,3$ 의 값을 가지며, *j*는 위치 샘플의 간격을 나타내며  $j=2,4,6,\dots,16$ 의 값을 가진다. 따라서  $S_{i,j}$ 는  $Traj_i = \{(t_i, x_i, y_i)\}_{i=0}^{205}$ 에서

표 1 위치 추정 실험을 위한 위치 샘플

이동구간 \ 샘플간격	2	4	6	8	10	12	14	16
$S_{1,j}$	$S_{1,2}$	$S_{1,4}$	$S_{1,6}$	$S_{1,8}$	$S_{1,10}$	$S_{1,12}$	$S_{1,14}$	$S_{1,16}$
$S_{2,j}$	$S_{2,2}$	$S_{2,4}$	$S_{2,6}$	$S_{2,8}$	$S_{2,10}$	$S_{2,12}$	$S_{2,14}$	$S_{2,16}$
$S_{3,j}$	$S_{3,2}$	$S_{3,4}$	$S_{3,6}$	$S_{3,8}$	$S_{3,10}$	$S_{3,12}$	$S_{3,14}$	$S_{3,16}$

$i$ 의 간격을  $j$ 만큼 증가시키면서 추출한 위치 샘플의 집합을 나타낸다. 생성된 위치 샘플을 이용하여 완전한 이동 경로 집합  $Traj_1, Traj_2, Traj_3$ 에 저장된 모든 과거의 시점에 대해 위치 추정 연산을 수행하였다. 위치 샘플 집합  $S_{i,j}$ 에 대해 생성된 과거 위치 추정 결과 집합은  $P_{i,j}$ 이다. 여기에서  $i$ 는 이동 구간의 식별자로  $i=1,2,3$ 이고,  $j$ 는 위치 샘플의 간격을 나타내며  $j=2,4,6, \dots, 16$ 이다. 이렇게 생성된 위치 추정 연산 결과의 오차 계산은 완전한 이동 경로 집합  $Traj_1, Traj_2, Traj_3$ 와 위치 추정 결과 집합  $P_{i,j}$ 와의 절대 오차를 식 (6)과 같이 계산하였다.

$$\Delta E = \frac{\sum_{i=0}^n \Delta e_i}{n+1}, \quad \Delta e_i = \frac{(|x_i - \hat{x}_i| + |y_i - \hat{y}_i|)}{2} \quad (6)$$

식 (6)에서  $\Delta E$ 는  $n+1$ 개의 위치 정보로 구성된 하나의 이동 구간 집합에 대해 계산된 위치 추정 결과의 평균 오차 값을 나타낸다.  $\Delta e_i$ 는 하나의 유효 시점에서의 위치 추정 결과의 오차 값이다. 여기에서  $x_i$ 와  $y_i$ 는  $i$  시점에서의 실제 위치 값을 나타내고,  $\hat{x}_i$ 와  $\hat{y}_i$ 는 위치 추정 연산에 의해 얻은 위치 값을 나타낸다. 선형 함수와 스플라인 함수를 이용한 실험을 통해 구한 각 구간 별 과거 위치 추정 오차의 비교 그래프는 그림 4와 같다.

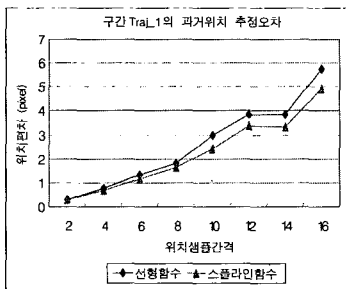
그림 4는 이동 구간  $Traj_1, Traj_2, Traj_3$ 의 과거 위치 추정 오차 결과를 비교한 그래프이다. x축은 시간 주기에 따른 위치 샘플의 간격을 나타내고, y축은 각 구간별 위치 샘플 간격에 따른 위치 추정 오차 값을 나

타낸다. 위치 샘플 간격은 위치 좌표를 획득하는 일정한 시간의 주기를 의미한다. 위치 샘플의 간격이 2일 때 데이터베이스에 저장된 이력 위치 정보의 양이 가장 많고, 16일 때 가장 적은 이력 위치 정보를 가진다. 그래프에서 알 수 있듯이 세 곳의 이동 구간에서 실험한 이동 객체의 과거 위치 추정 연산 결과를 보면 모든 구간에서 공통적으로 선형 함수를 이용한 경우보다 스플라인 함수를 이용한 위치 추정 결과가 더욱 정확함을 확인하였다. 특히, 위치 샘플의 간격이 커질수록 선형 함수를 이용한 결과와 스플라인 함수를 이용한 결과 값의 차이가 더 커지는 특징을 발견하였다.

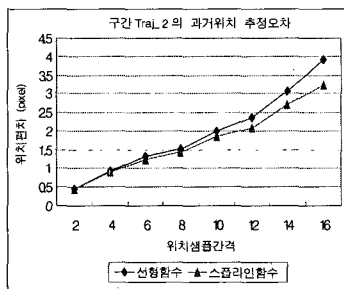
4.1.2 미래의 위치 추정 연산 결과

미래 위치 추정 실험을 위해 사용된 미래의 질의 시점은 이동 경로 집합  $Traj_1, Traj_2, Traj_3$ 에 저장된 시간 값과 관계 적용하였다. 위치 샘플 집합  $S_{i,j}$ 에 대해 생성된 미래 위치 추정 결과 집합은  $F_{i,j}$ 이다.  $i$ 는 이동 구간의 식별자로  $i=1,2,3$ 이고,  $j$ 는 위치 샘플의 간격을 나타내며  $j=2,4,6, \dots, 16$ 이다. 생성된 위치 추정 연산 결과의 오차 계산은 완전한 이동 경로 집합  $Traj_1, Traj_2, Traj_3$ 와 위치 추정 결과 집합  $F_{i,j}$ 와의 절대 오차를 구하여 계산하였다. 각각의 이동 구간에서의 미래 위치 추정 실험 결과는 그림 5와 같다.

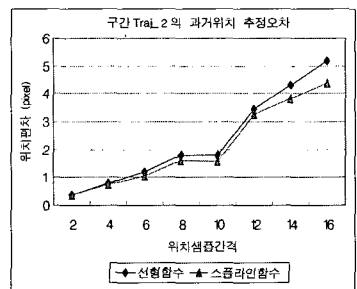
그림 5는 이동 구간  $Traj_1, Traj_2, Traj_3$ 의 미래 위치 추정 오차 결과를 비교한 그래프이다. 실험 결과 스플라인 함수를 이용한 위치 추정 연산 결과가 선형 함수를 이용한 경우보다 더 정확하게 나타났다.



(a) 이동경로 Traj\_1



(b) 이동경로 Traj\_2



(c) 이동경로 Traj\_3

그림 4 과거의 위치 추정 실험 결과



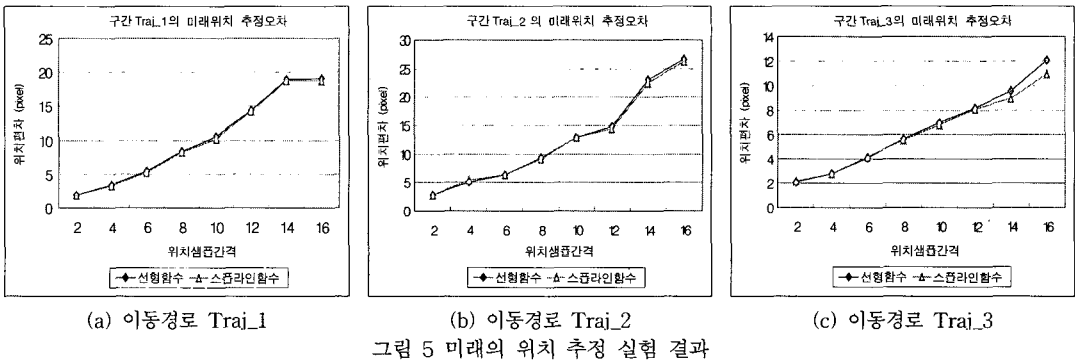


그림 5 미래의 위치 추정 실험 결과

4.1.3 위치 추정 오차의 변화 추이

위치 추정 실험 결과를 토대로 세 곳의 이동 구간에서 발생된 위치 추정 연산 결과의 오차 변화 추이 분석을 위해, 세 이동 구간에서의 위치 추정 결과의 평균값을 계산하였으며, 각 위치 샘플의 간격에 따라 평균 위치 추정 오차 값을 비교하였다. 오차 변화 추이 결과는 그림 6과 같다.

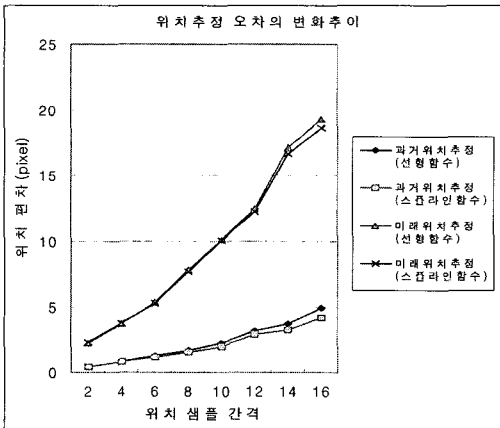


그림 6 위치 추정 결과의 오차 변화 추이

그림 6의 결과에서 알 수 있듯이 과거 위치 추정 연산의 결과가 미래 위치 추정 연산의 결과보다 약 4배정도 정확하게 추정됨을 알 수 있었다. 아울러 이 논문에서 제안한 스플라인 함수를 이용한 위치 추정 연산 결과의 오차가 기존의 선형 함수를 사용한 경우보다 더 적게 나타남을 확인하였다. 전체적인 오차 변화 추이는 위치 샘플의 간격이 커질수록 위치 추정 연산 결과의 오차가 커지는 현상을 보이고 있다.

4.1.4 위치 추정 결과의 고찰

과거 및 미래의 위치 추정 결과를 보면 항상 과거의 위치 추정 결과가 더욱 정확하게 나타난다. 이 결과는 실험 대상인 이동 차량의 특성으로 인해 나타난 것이다.

과거의 위치 추정은 차량이 이동한 실제 이동 경로 내에서 특정 시점에서의 위치 좌표를 추정하므로, 위치 추정 결과의 오차가 그리 크지 않음을 알 수 있다. 반면에 미래의 위치 추정은 지금까지의 이동 경로를 이용하여 아직 이동하지 않은 특정한 시점에서의 위치 좌표를 추정하므로, 과거의 위치 추정 결과에 비해 오차가 다소 크게 나타남을 알 수 있다. 이것은 실험 대상인 이동 차량이 도로와 같은 특정한 이동 경로를 따라서 움직이는 특성을 가지기 때문이다. 아울러 모든 실험 결과에서 스플라인 함수를 이용한 위치 추정 결과가 선형 함수를 이용한 경우보다 더 정확하게 나타나고 있다. 이것은 이동 차량의 움직임이 수 미터의 짧은 거리에서는 직선으로 나타나지만, 수십 미터 이상의 먼 거리에서는 부드러운 곡선의 형태로 이동하는 특성을 띠기 때문이다.

4.2 위치 추정 결과의 정확성

위치 추정 연산 결과의 오차 실험 값을 이용하여, 선형 함수와 스플라인 함수에 의한 위치 추정 결과의 정확성을 비교하였다. 정확성 값을 계산하기 위해 그림 7과 같이 불확실성 영역(uncertainty area)의 반경을 변경시켜가며 정확성 값을 계산하였다.

그림 7은 불확실성 영역의 범위에 따라 동일한 위치 추정 결과의 정확성 판단 결과가 다르게 나타남을 보이고 있다. 만약, 시스템이 제시한 위치 추정 결과의 위치

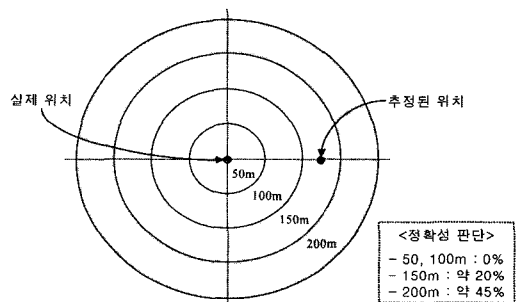


그림 7 불확실성 영역에 따른 정확성 판단

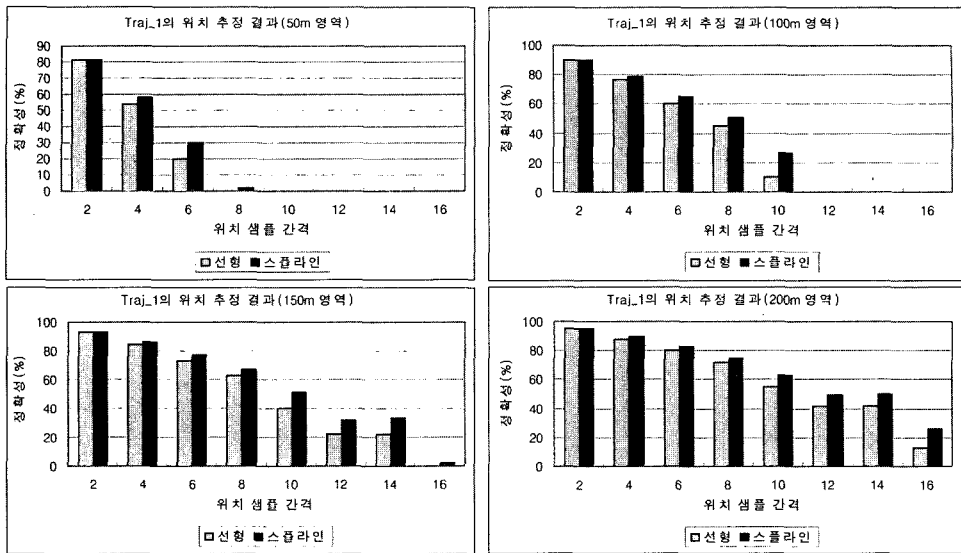


그림 8 이동 구간 Traj\_1에서의 위치 추정 결과의 정확성

편차가 50m 라고 가정하자. 이 때 A라는 사용자는 50m 정도의 편차는 무시할 수 있다고 생각하여 위치 추정 결과의 정확성을 100%로 생각할 수 있다. 반면 B라는 사용자는 위치 편차가 30m 내에 들어오길 원한다면, 이 경우의 정확성은 0%라 할수 있다. 정확성을 퍼센트(%)로 나타내기 위한 계산은 식 (7)과 같다.

$$\text{정확성} = 100 - \frac{|\text{원값} - \text{추정값}|}{\text{원값}} * 100 \quad (7)$$

이 논문에서는 원값 대신 불확실성 영역의 반경을 나타내는 값을 사용하였다. 그리고 |원값 - 추정값|은 위치 추정 결과로 발생하는 추정 값과 실제 값과의 위치 편차로 대신하였다. 따라서 정확성은 식 (8)과 같이 계산된다.

$$\text{정확성} = 100 - \frac{\text{추정된 위치 값의 편차}}{\text{불확실성 영역의 반경}} * 100 \quad (8)$$

4.1절에서 위치 추정 연산의 실험 결과로 발생하는 오차 즉, 위치 편차의 단위는 픽셀이었다. 그리고 실험에서 사용된 1 픽셀은 약 30m라고 가정하였으므로, 위치 추정 실험 결과로 얻은 오차 값에 30을 곱하면 다음과 같이 추정된 위치 값의 편차를 구할 수 있다.

$$\text{추정된 위치 값의 편차} = \text{위치 추정 오차} * 30 \quad (9)$$

만약 추정된 위치와 실제 위치와의 편차가 불확실성 영역 값보다 크면 식 (8)에 의해 정확성 값은 음수가 된다. 이 경우에는 정확성 값을 모두 '0'으로 할당하였다.

그림 8은 이동 구간 Traj\_1의 위치 추정 실험 결과에 대해, 불확실성 영역의 반경을 각각 50, 100, 150, 200m 로 변경시켜가며 위치 샘플 간격별로 정확성을 계산한

결과이다. 이동 구간 Traj\_1에서는 불확실성 반경이 100m, 위치 샘플 간격이 10인 경우 선형 함수와 스플라인 함수에 의한 위치 추정 결과의 정확성에 가장 큰 차이가 발생함을 알 수 있다.

그림 9는 Traj\_2의 위치 추정 실험 결과의 정확성 결과이다. Traj\_2에서는 전체적으로 위치 편차가 120m 보다 적게 발생되었기 때문에 불확실성 영역을 30, 60, 90, 120m로 설정한 후 위치 추정 결과의 정확성을 계산하였다.

그림 9의 그래프에서 알 수 있듯이, Traj\_2에서는 불확실성 반경이 120m, 위치 샘플 간격이 16일 때 선형 함수와 스플라인 함수에 의한 정확성에 가장 큰 차이가 발생되었다.

그림 10은 이동 구간 Traj\_3의 위치 추정 실험 결과의 정확성 값을 그래프로 나타낸 것이다. Traj\_3에서는 불확실성 반경을 50, 100, 150, 200m로 설정하였다.

### 4.3 실험 결과의 분석

지금까지의 위치 추정 실험 결과를 다음과 같이 두 가지 측면에서 분석하였다.

첫째, **위치 추정의 정확성** : 데이터베이스에 저장된 동일한 위치 정보를 활용할 경우, 기존의 선형 함수만을 사용한 경우보다 이 논문에서 제안한 3차 스플라인 보간법을 적용한 경우 위치 추정 결과의 오차를 전체적으로 줄일 수 있음을 확인하였다. 특히 위치 샘플 간격이 커질수록 스플라인 보간법에 의한 위치 추정 결과가 더 정확함을 알 수 있었다.

둘째, **데이터 저장 및 송수신 비용** : 위치 정보의 샘플

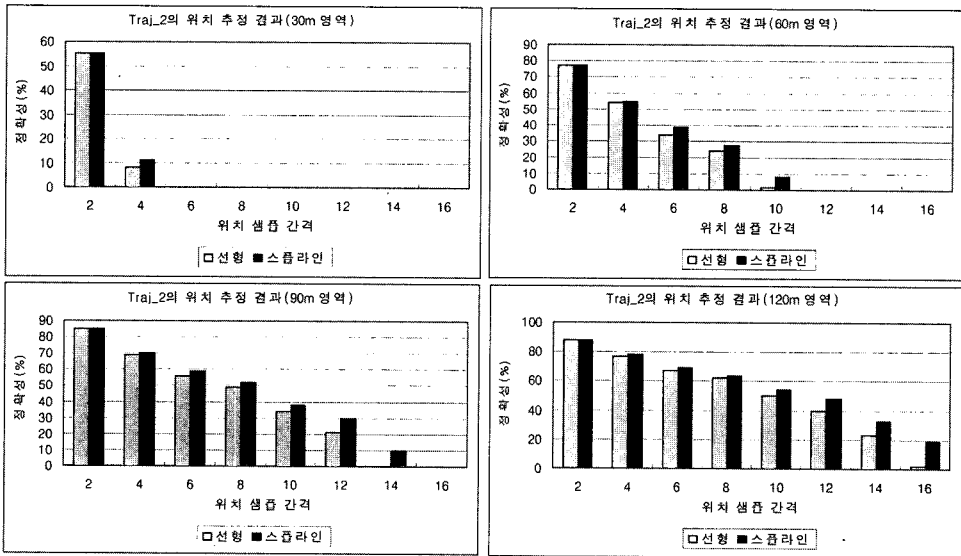


그림 9 이동 구간 Traj\_2에서의 위치 추정 결과의 정확성

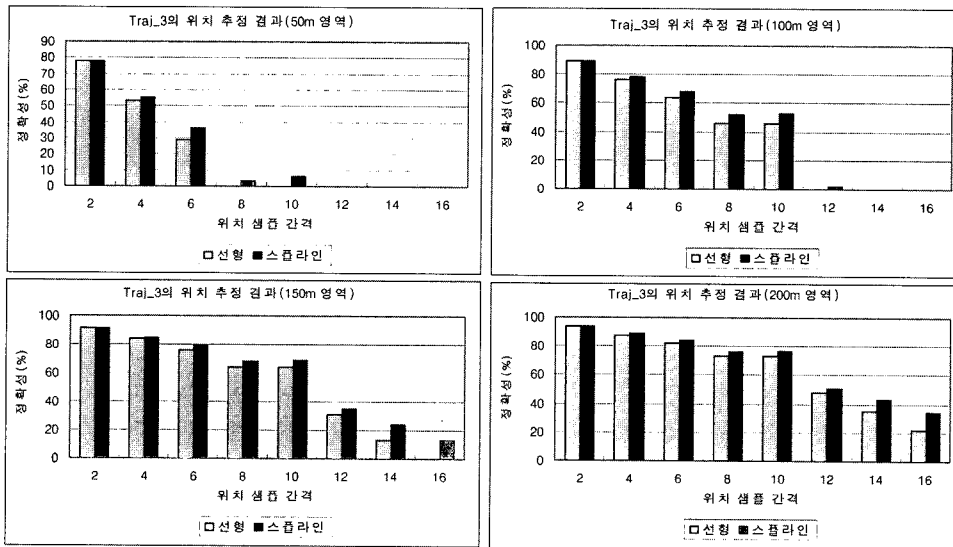


그림 10 이동 구간 Traj\_3에서의 위치 추정 결과의 정확성

플 간격은 데이터베이스에 저장된 데이터 량과 반비례한다. 즉 샘플 간격이 커질수록 데이터베이스에 저장되는 데이터 량이 적어진다. 그런데 실험을 통해 위치 샘플의 간격이 커질수록 이 논문에서 제안한 위치 추정 연산이 더 정확한 결과를 발생시킴을 확인하였다. 이 결과는 적은 량의 위치 정보만을 가지고도 기존에 제안된 방법과 비슷한 수준의 정확성을 얻을 수 있음을 의미한다. 결국 이 논문에서 제안한 위치 추정 연산을 사용하

면 데이터의 저장 공간을 절약할 수 있을 뿐만 아니라, 실시간 환경에서 이동 객체의 위치 정보 획득을 위한 데이터 송수신 비용도 감소시킬 수 있음을 확인하였다.

이 실험에서는 위치 추정 연산의 계산속도에 대해서는 고려하지 않았다. 위치 추정 연산은 어느 한 시점에서 하나의 위치 좌표만을 추정한다. 이 때, 하나의 위치 좌표를 추정하는 데 걸리는 계산속도는 선형 보간법과 3차 스플라인 보간법 사이에 거의 차이가 없기 때문이

다. 다량의 위치 좌표 추정 연산이 동시에 수행된다면 계산속도에 차이를 보일 수 있을 것이다. 그러나, 이 논문에서는 한번에 하나의 위치 좌표만을 추정하도록 연산하였으며, 실험에서는 다수의 위치 좌표를 개별적으로 연산한 후 전체의 정확성을 평균값으로 제시하였다.

## 5. 결론

이동 객체의 위치 정보는 차량 위치 추적, 모바일 상거래, 위치 기반 서비스, 텔레메틱스 등의 응용 시스템 개발에 사용된다. 이 때 시스템에 저장되지 않은 이동 객체의 불확실한 위치 정보로 인해 데이터 모델링, 질의 처리, 인덱싱, 질의 결과의 부정확성 등이 발생된다. 이와 같은 문제점을 해결하기 위해 지금까지 연구된 이동 객체의 위치 처리 방법에서는 주로 선형 함수를 사용하거나, 선형 회귀 함수를 적용하고 있다. 그러나 일반적으로 선형 함수를 이용한 위치 추정은 가장 많은 오차를 발생시키므로, 좀 더 정확한 위치 추정 방법이 필요하다.

따라서 이 논문에서는 선형 함수에 의한 위치 추정 결과의 오차를 줄일 수 있는 방법으로 3차 스플라인 보간법의 사용을 제안하였다. 먼저 2차원 공간에서 이동 객체의 위치 정보 및 위치 추정 함수를 정의하였다. 제안한 이동 객체의 위치 정보 모델을 이용하여 3차 스플라인 보간법을 적용한 위치 추정 연산 및 처리 알고리즘을 기술하였다. 아울러 위치 추정 연산 알고리즘을 이용한 실험을 통해 위치 추정 결과의 오차 및 정확성을 분석하였다.

실험 결과를 통해 데이터베이스에 저장된 동일한 위치 정보를 활용할 경우, 기존의 선형 함수를 사용한 경우보다 제안한 3차 스플라인 보간법을 적용한 경우 위치 추정의 정확성을 높일 수 있음을 확인하였다. 결국 이 논문에서 제안한 위치 추정 연산을 사용하면 데이터의 저장공간을 절약할 수 있을 뿐만 아니라, 실시간 환경에서 이동 객체의 위치 정보 획득을 위한 데이터 송수신 비용도 감소시킬 수 있음을 확인하였다.

이 논문에서는 이동 객체의 위치 추정 방법에 대해서만 언급하였다. 향후에는 위치 추정 연산뿐만 아니라, 불확실성 영역 및 위치 추정 결과의 오차 값을 정량화하여 제시할 수 있는 방법의 연구를 진행할 것이다. 아울러 실시간 환경에서 발생하는 위치 정보의 불확실성 처리 방법에 대한 연구도 수행될 것이다.

## 참 고 문 헌

- [1] D. Pfoser and C. S. Jensen, "Capturing the Uncertainty of Moving Object Representations," CHOROCRONOS, Technical Report CC-99-2, Apr. 1999.
- [2] 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, Apr. 1999.
- [3] 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 ASTC'01, Apr. 2001, pp. 108-113.
- [4] 안윤애, 류근호, 조동래, "전장분석을 위한 이동 객체의 위치 예측 시스템," 정보과학회논문지, 제 8 권 제 6 호, 2002년 12월, pp. 765-777.
- [5] D. Pfoser and N. Tryfona, "Fuzziness and Uncertainty in Spatiotemporal Applications," CHOROCRONOS, Technical Report CH-00-04, Feb. 2000.
- [6] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, "Querying the Uncertain Position of Moving Objects," Springer Verlag Lecture Notes in Computer Science number 1399, 1998, pp. 310-337.
- [7] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez, "Cost and Imprecision in Modeling the Position of Moving Objects," Proceedings of the 14th International Conference on Data Engineering, Orlando, FL, Feb. 1998.
- [8] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang, "Moving Objects Databases: Issues and Solutions," Proceedings of the 10th International Conference on Scientific and Statistical Database Management, SSDBM'98, Capri, Italy, Jul. 1998, pp. 111-122.
- [9] O. Wolfson, P. Sistla, B. Xu, J. Zhou, S. Chamberlain, N. Rishe, and Y. Yesha, "Tracking Moving Objects Using Database Technology in DOMINO," Proceedings of NGITS'99, The 4th Workshop on Next Generation Information Technologies and Systems, Zikhron-Yaakov, Israel, Jul. 1999, pp. 112-119.
- [10] T. S. Yeh and B. D. Cambray, "How to Model Highly Variable Data in a Complex Object Data Model," Proceedings of the COMAD'94, 1994, pp. 169-186.
- [11] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, "Modeling and Querying Moving Objects," Proceedings of the International Conference on Data Engineering, 1997, pp. 422-432.
- [12] M. Erwig, R. H. Gutting, M. Schneider, and M. Vazirgiannis, "Spatio-Temporal Data Types : An Approach to Modeling and Querying Moving Object in Databases," GeoInformatica Vol. 3, No. 3, 1999, pp. 269-296.
- [13] R. H. Gutting, M. H. Bohlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, and M.

- Varzirgiannis, "A Foundation for Representing and Querying Moving Objects," ACM Transactions on Database Systems, Vol. 25, No. 1, Mar. 2000, pp. 1-42.
- [14] L. Forlizzi, R. H. Guting, E. Nardelli, and M. Schneider, "A Data Model and Data Structures for Moving Objects Databases," Proceedings of the ACM SIGMOD'00, 2000, pp. 319-330.
- [15] S. Shumilov and J. Seibeck, "Database Support for Temporal 3D Data : Extending the Geo-ToolKit", Proceedings of the 7th EC-GI & GIS Workshop, ECGIS'01, Potsdam, Germany, Jun. 2001.
- [16] 정상권, 이승래, 권옥현, "CEMTool을 활용한 수치해석", 인터넷 공개강좌, <http://www.cemworld.com>, 2004.
- [17] R. Bartels, J. Beatty, and B. Barsky, "An Introduction to Splines for Use in Computer Graphics & Geometric Modeling," Morgan Kaufmann Publishers, Inc., 1987.
- [18] 유해영, 이준신, 고영주, 심홍태, "수치해석 제6판", 사이텍미디어, 1999.
- [19] 한국통신데이터, "GIS를 위한 객체관계형 공간 데이터베이스 관리시스템", ZEUS 2000, ZEUS/X 사용자 매뉴얼, 한국통신데이터(주), 2002년.
- [20] 안윤애, "위치 추정을 지원하는 모바일 객체 관리 시스템", 이학박사학위논문, 2003년 2월.



안 윤 애

1993년 한남대학교 전자계산공학과(공학사). 1996년 충북대학교 대학원 전자계산학과(이학석사). 2003년 충북대학교 대학원 전자계산학과(이학박사). 2003년~현재 청주과학대학 컴퓨터공학과 전임강사

관심분야는 시공간 데이터베이스, 이동 객체 관리, 지리정보 시스템, 지식기반 시스템



박 정 석

1981년 숭실대학교 전자계산학과(공학사). 1983년 숭실대학교 대학원 전자계산학과(공학석사). 2000년 충북대학교 대학원 전자계산학과(이학박사). 1983년~1996년 한국원자력연구소(선임연구원). 1996년~현재 국립청주과학대학 컴퓨터공학과

부교수. 관심분야는 이동 객체 데이터 관리, continuous query 처리, context database

류 근 호

정보과학회논문지 : 데이터베이스  
제 31 권 제 1 호 참조