이동 객체의 과거 및 미래 위치 연산을 위한 데이터 모델

장 승 연[†] · 안 윤 애[†] · 류 근 호[††]

요 약

무선 환경에서 시공간 이동 객체들의 위치 정보를 획득할 수 있는 기술들의 발달로 인해, 차량 추적 시스템, 산불 관리 시스템, 디지털 전장 시스템 등과 같은 다양한 응용 시스템들이 개발되고 있다. 이와 같은 응용에서는 이동 객체의 연속적인 변화 과정을 표현하고, 처리할 수 있는 데이터 모델이 필요하다. 그러나 관계형 모델을 이용하여 이동 객체를 표현할 경우 매 시간 마다 변경된 모든 정보를 저장할 수 없는 문제점이 발생된다. 또한 기존의 이동 객체 데이터 모델에서는 질의 시점을 데이터베이스로 관리되는 과거 시간이나 현재로부터 가까운 미래 시간으로 제한하는 단점이 있다 . 따라서 이 논문에서는 모양 변화 처리 기법과 과거와 미래의 위치 추정 함수를 사용함으로써 이동 점 및 이동 영역 객체의 연속적인 움직임을 표현하고 모든 질의 시점에서의 연산을 처리 할 수 있는 이동 객체 데이터 모델을 제안한다. 아울러 제안 모델을 산불 관리 시스템에 적용 및 구현한 결과를 통해 타당성을 평가한다.

A Data Model for Past and Future Location Process of Moving Objects

Seung Youn Jang[†] · Yoon Ae Ahn[†] · Keun Ho Ryu[††]

ABSTRACT

In the wireless environment, according to the development of technology, which is able to obtain location information of spatiotemporal moving object, the various application systems are developed such as vehicle tracking system, forest fire management system and digital battle field system. These application systems need the data model, which is able to represent and process the continuous change of moving object. However, if moving objects are expressed by a relational model, there is a problem which is not able to store all location information that changed per every time. Also, existing data models of moving object have a week point, which constrain the query time to the time that is managed in the database such as past or current and near future. Therefore, in this paper, we propose a data model, which is able to not only express the continuous movement of moving point and moving region but also process the operation at all query time by using shape-change process and location determination functions for past and future. In addition, we apply the proposed model to forest fire management system and evaluate the validity through the implementation result.

키워드 : 이동 객체(Moving object), 시공간 데이터 모델(Spatiotemporal data model), 위치 추정 함수(Location determination function), 시공간 데이터베이스(Spatiotemporal database)

1. 서 론

현실 세계에서 움직이는 모든 객체들은 시간과 공간 속성을 동시에 가지는 시공간 데이터이다. 이러한 시공간 데이터 중 시간에 따라 연속적으로 위치나 모양이 변하는 데이터를 이동 객체(moving objects)[1, 2, 5, 6]라 한다. 무선 환경에서 객체의 시공간 정보를 획득할 수 있는 기술들이 발전함에 따라 차량 추적 시스템, 응급 구조 시스템, 디지털 전장 시스템 등과 같은 다양한 이동 객체 응용 시스템들이 개발되고 있다.

이러한 응용 시스템들은 "사고지역 A의 응급환자를 이송한 응급차의 전체 이동 경로를 검색하라", 또는 "현재 화재 지역 A의 피해면적 상태와 이 지역에 5분 이내에 도착할 수 있는 소방차를 검색하라" 등과 같은 시간과 공간 제약을 가지는 사용자 질의를 처리할 수 있어야 한다. 이를 위해서는 시간에 따라 위치 및 모양이 변화하는 이동 객체 데이터를 데이터베이스에 저장 관리 할 수 있어야 하며, 질의 처리에 필요한 연산을 수행할 수 있어야 한다.

그러나 기존의 관계형 모델을 이용하여 이동 객체 데이터를 관리할 경우 연속적으로 변화하는 이동 객체의 시공간 정보를 모두 저장할 수 없기 때문에, 저장 되지 않은 정보를 요구하는 사용자 질의를 처리하기 어려운 문제가 있다. 이를 해결하기 위해 이동 점 객체의 위치 추적에 초점을 둔 연구[12]와 이동 객체를 위한 새로운 데이터 타입과 연산자에 관한 연구[2, 5, 6]가 수행되었다. 그리고 GIS를 이용한 이동

※ 이 연구는 한국학술진흥재단(KRF-2002-002-D00152)의 연구비 지원으로 수행되었음.
† 준 회 원 : 충북대학교 대학원 전자계산학과
†† 종신회원 : 충북대학교 전기전자및컴퓨터공학부 교수
논문접수 : 2002년 8월 22일, 심사완료 : 2002년 11월 14일

객체 관리 시스템 프로토타입[16, 20]이 개발되었다. 그러나 기존에 연구된 이동 객체 모델[2-6, 12]은 시간에 따라 위치만 변경되는 이동 점 객체만을 다루거나, 과거, 현재, 미래의 모든 질의 시점에 대한 연산처리를 동시에 지원하지 못하는 문제점들을 가지고 있다.

따라서 이 논문에서는 이동 점과 이동 영역의 연속적인 변화 과정의 표현 및 모든 질의 시점에서의 연산 처리가 가능한 이동 객체 데이터 모델을 제시한다. 먼저 이동 점과 이동 영역의 연속적인 변화 과정을 표현하기 위해 시간 종속 위치 추정 함수와 모양 변화 처리 기법을 제시한다. 그리고 과거, 현재, 미래 시점에서의 모든 연산 처리를 동시에 지원하기 위하여 과거와 미래의 객체 위치를 추정하기 위한 각각의 시간 종속 위치 추정 함수를 사용한다. 아울러 제안 모델의 데이터 저장 방법 및 연산 처리 알고리즘을 소개한 후 제안 모델을 산불 관리 응용에 적용한다. 그리고 적용 결과를 통해 제안 모델을 평가한다.

이 논문의 구성은 다음과 같다. 먼저 제2장에서는 기존에 연구된 이동 객체 모델에 대해 검토한다. 제3장에서는 모든 질의 시점에서의 연산 처리를 위한 이동 객체 표현 방법 및 연산자를 제시한다. 제4장에서는 제안한 이동 객체 모델의 주요 설계 알고리즘을 기술한다. 제5장에서는 제안 모델을 산불 관리 시스템에 적용한 결과를 예시한다. 마지막으로 제6장에서는 결론 및 향후 연구 방향을 기술한다.

2. 관련 연구

이동 객체란 시간에 따라 객체의 공간 정보가 연속적으로 변경되는 객체로 이동 점과 이동 영역으로 나눈다. 이동 점(moving points)은 시간에 따라 객체의 위치가 연속적으로 변화하는 객체를 말하며, 이동 영역(moving regions)은 시간에 따라 객체의 위치뿐만 아니라 모양까지 변화하는 객체이다[1, 2, 5, 6]. 이러한 이동 객체의 연속적인 시공간 정보를 데이터베이스에 저장 관리하기 위해서는 기존의 시공간 데이터베이스 시스템에서 제공하는 데이터 타입과 연산자 이외에도 이동 객체의 연속적인 움직임을 표현하기 위한 데이터 모델이 필요하다.

지금까지 연구된 이동 객체 모델에서 Sistla[12, 13]와 Wolfson[14, 15]은 동적 속성(dynamic attribute)을 통해 이동 점 객체의 연속적인 위치 정보를 제공하는 MOST(moving object spatio-temporal) 모델을 제시하였다. 이 모델에서는 eventually, within, until, nexttime, always등과 같은 시간 연산자를 사용하는 FTL(future temporal logic)언어를 통해 이동 점 객체에 대한 향후 위치 질의를 표현하였다. 그러나 이 모델은 이동 점 객체만을 대상으로 하고 있으며 객체의 이력 정보를 저장하지 않으므로 객체의 과거 위치 및 궤적 정보와 관련된 질의 처리가 어렵다.

CHOROCHRONOS 컨소시엄에서 Guting[5, 6]과 Erwig[1]은 이동 객체를 표현하기 위한 시공간 데이터 모델을 제안하였다. 이들은 이동 점과 이동 영역 이라는 추상 데이터 타입을 정의하였으며 이동 객체들에 대한 연산 처리를 위해 trajectory, atinstant, duration, length, intersection, velocity, overlaps, area, traversed등과 같은 다양한 이동 객체 연산자를 제안하였다. 이러한 추상 데이터 모델을 이산적으로 표현하기 위해 Forlizzi[2]는 슬라이스드 표현(sliced representation) 방법을 제시하였다. 이 방법은 시간에 대한 단순 함수와 이 단순함수가 적용되는 시간 구간으로 이루어진 유닛(unit)들의 집합을 통해 이동 객체를 나타낸다. 이러한 방법들을 통해 현재 이 모델은 SECONDO 시스템을 기반으로 구현 중에 있다[8]. 그러나 시간에 대한 함수로서 표현된 유닛들의 집합을 통해 이동 객체에 대한 연산을 처리하기 때문에 연산 처리 복잡도가 증가하는 단점이 있다. 이들은 연산 속도를 향상시키기 위해 연산 처리시 필요한 정보들을 유닛 단위별로 저장해두는 요약 필드(summary field)[8]를 두고 있다. 이때 요약 필드는 추가적인 저장 공간을 필요로 하며, 요약 필드의 내용을 유닛 생성 과정에서 계산하고 연산 처리시 이들 정보들을 다시 조합해야 하기 때문에 오버헤드가 발생한다. 아울러 이 모델은 [14]에서 지적한 바와 같이 이동 객체의 과거 정보를 다루는데 적합한 모델이다.

Grumbach[3, 4]는 무한한 점들의 집합으로 구성된 다차원 데이터를 제약 조건을 통해 유한하게 표현하는 제약 데이터베이스(constraint databases)를 기반으로 DEDALE이라는 시공간 응용 시스템의 프로토타입을 개발하였다. 이 프로토타입에서 이동 객체는 시간과 공간 제약 조건이 분리된 중첩 릴레이션(nested relation)을 통해 표현된다. 이 접근은 다차원의 데이터를 대량으로 다룰 경우 발생되는 질의 처리 복잡도를 고려하여 시간 차원과 공간 차원을 분리시켰지만 연산 처리시 제약된 시간 구간에서 정확한 객체의 위치를 알 수 없다는 단점이 있다. 또한 객체의 미래 위치 표현을 위해 필요한 최근 속도 정보를 알 수 없다. 따라서 데이터베이스로 관리 되지 않는 현재 또는 가까운 미래 시점에서의 연산 처리를 지원하지 못한다.

Shumilov[11]는 시공간 객체를 표현하기 위해 객체를 구성하는 정점의 위치를 시간에 대한 함수처럼 표현하는 정점-이동(vertex-movement)모델을 제시하였다. 이 모델에서는 변화하는 객체의 기하정보를 추정하기 위해 매 시점마다 객체의 공간 정보를 이전 시점에서의 공간 정보와 근사시킨 상태(pre-discretization)와 이후 시점에서의 공간 정보와 근사시킨(post-discretization) 상태로 분리하여 표현한 후 두 상태와 대응 되는것 사이에 보간법을 적용하는 방법을 사용하였다. 그러나 이 모델은 강유역의 기하정보 변화와 같은 오랜 시간에 걸쳐 변화하는 객체를 대상으로 하기 때문에 이 방법을 연속적인 이동 영역 객체에 그대로 적용하는

경우 많은 저장 공간을 요구하는 단점이 있다.

이처럼 기존의 이동 객체 연구들은 이동 점 객체의 현재 및 미래의 위치 추적에 초점을 맞추거나 이동 객체의 과거 정보만을 다루기 때문에, 질의 객체나 질의 시간을 제한하는 문제점이 있다. 따라서 이 논문에서는 모양 변화 처리 과정과 시간 종속 위치 추정 함수를 사용하여 연속적인 이동 객체의 움직임을 표현하고, 과거, 현재, 미래의 모든 시점에서 사용자 질의를 처리할 수 있는 시공간 이동 객체 데이터 모델을 제안한다.

3. 이동 객체 데이터 모델

이 장에서는 시간 종속 위치 추정 함수와 모양 변화 처리 기법을 통하여 이동 객체의 연속적인 움직임을 표현하는 방법 및 모든 질의 시점에서의 연산을 처리할 수 있는 방법을 제시한다.

연속적인 표현 대상이 되는 객체들은 <표 1>과 같은 시간과 공간 속성을 가지며 다음과 같은 제약 사항들을 가진다.

〈표 1〉 이동 객체의 시공간 속성

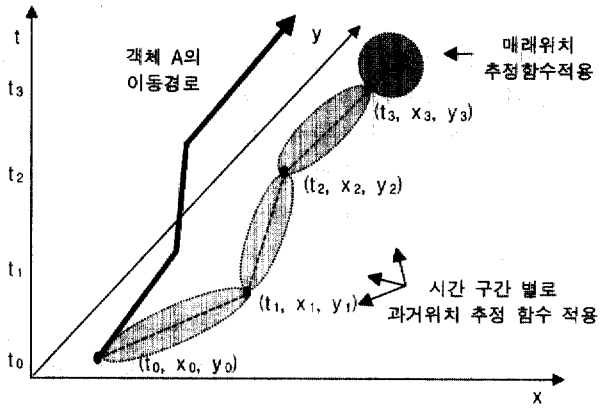
속 성	타 입	의 미
시 간	유효시간	객체 속성 값이 참값을 가질 때의 시간 [Validfrom, Validto)
공 간	포 인 트	2차원 평면상에서 x, y 좌표값으로 표현 되는 점
	폴 리 곤	n개의 점들로 구성된 다각형

이동 점은 일정 경로를 따라 움직이는 2차원 평면상의 점으로 일정 시간 간격별로 샘플링 하여 표현한다. 따라서 시간 속성으로 Validfrom과 Validto에 의해 표현되는 유효 시간값을 갖는다. 이동 영역은 어떠한 방향으로도 이동이 가능한 객체로서 2차원 평면상의 폴리곤으로 표현한다. 이동 영역 또한 일정 시간 간격별로 샘플링하여 표현하며 시간 속성으로 유효시간을 가진다. 또한 객체의 모양 변화(커짐과 작아짐)와 각 샘플링 시점마다 폴리곤 정보를 구성하는 정점수 변화를 허용한다.

3.1 이동 점

이동 점 객체의 연속적인 움직임은 일정 시간 간격별로 샘플링하여 얻은 객체 정보와 시간 종속 위치 추정 함수를 이용함으로 해서 표현할 수 있다. 예를 들어 $[t_0, t_3]$시간 구간 동안 이동 점 객체 A가 (그림 1)처럼 샘플링 되었다고 가정하자.

시간상에서 연속적으로 이동한 객체 A의 시공간 정보는 (그림 1)처럼 일정 시간 간격별로 얻어진 객체 정보 (t_0, x_0, y_0), (t_1, x_1, y_1), (t_2, x_2, y_2), (t_3, x_3, y_3)와 각 시간 구간에 적용하는 시간 종속 위치 추정 함수를 통해 표현된다. 즉 샘플링 된 각각의 시간 구간에 시간 종속 위치 추정 함



(그림 1) 이동 점의 이산적 표현

수를 적용함으로써 샘플링 되지 않은 객체의 위치 정보를 나타내며 이를 통해 이동 점 객체의 연속적인 움직임을 표현한다. 이때 질의 시점에 따라 객체의 과거 위치를 추정하기 위한 함수나 객체의 가까운 미래 위치를 표현하기 위한 함수를 별도로 적용함으로써 모든 질의 시점에서 객체의 불확실한 위치를 추정할 수 있다. 이동 객체의 불확실한 위치를 추정하기 위해 사용하는 시간 종속 위치 추정 함수는 여러 가지[7, 9, 10, 17, 18]가 있지만, 이 논문에서는 객체 위치 추정 시 연산 처리 복잡도를 고려하여 시간에 대한 선형 함수를 위치 추정 함수로 사용한다. 다음 식 (1)과 식 (2)에서 표현된 시간 종속 위치 추정 함수는 객체의 과거 위치와 미래 위치 추정을 위해 사용된다.

즉, 과거 시점 t에서 저장 되지 않은 객체의 위치 정보를 추정하기 위한 함수는 식 (1)로 나타낸다.

$$f(t) = \left(\frac{x_i - x_{i-1}}{t_i - t_{i-1}}(t - t_{i-1}) + x_{i-1}, \frac{y_i - y_{i-1}}{t_i - t_{i-1}}(t - t_{i-1}) + y_{i-1} \right) \text{ if } (t_{i-1} < t < t_i) \quad (1)$$

식 (1)은 샘플링 되지 않은 이동 점 객체의 과거 위치를 추정하기 위한 시간 종속 위치 추정 함수를 나타낸다. 이 함수는 데이터베이스에 저장 되지 않은 t시점에 대한 질의가 들어온 경우 t시점이 포함되는 샘플링 구간에서의 객체 정보 $(t_{i-1}, x_{i-1}, y_{i-1})$, (t_i, x_i, y_i)를 통해 생성 되고 질의 시점 t에서의 객체 위치 (x, y)를 반환한다.

가장 최근에 샘플링된 두 시점의 객체 정보를 각각 $(t_{i-1}, x_{i-1}, y_{i-1})$, (t_i, x_i, y_i)라 하자. 이때, 가까운 미래 시점 t에 대한 질의가 들어온 경우, 객체의 미래 위치 추정은 식 (2)를 통해 표현 할 수 있다.

$$f(t) = \left(\frac{x_i - x_{i-1}}{t_i - t_{i-1}}(t - t_i) + x_i, \frac{y_i - y_{i-1}}{t_i - t_{i-1}}(t - t_i) + y_i \right) \text{ if } (t_i < t) \quad (2)$$

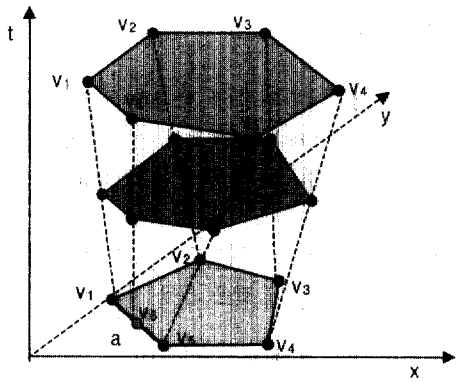
식 (2)는 샘플링 되지 않은 이동 점 객체의 미래 위치를 추정하기 위한 시간 종속 위치 추정 함수를 나타낸다. 이 함수는 가까운 미래 t시점에서 객체의 위치를 요구하는 질의가 들어온 경우 가장 최근에 샘플링 된 두 시점의 객체 정보를 통해 최근 시점에서의 벡터 정보를 구한다. 그 후, 해당 벡터 정보를 이용하여 가까운 미래 위치를 추정한다. 이 함수 결과 반환 되는 값은 질의 시점 t에서의 객체 위치 (x, y)이다.

일정 시간 간격별로 샘플링 된 객체의 시공간 정보를 통하여 연속적인 이동 점 객체의 움직임을 표현하는 방법을 요약하면 다음과 같다. 즉 특정 시점 t에서의 객체 위치 정보를 요구하는 질의가 들어온 경우 질의 시점에 대한 정보가 데이터베이스에 저장 되어 있는지 확인한다. 저장된 정보는 그대로 처리되며 저장 되지 않은 경우에는 질의 시점을 확인하게 된다. 즉 과거 또는 현재 정보를 요구하는 것인지 또는 가까운 미래 정보를 요구하는 것인지 분석한다. 그런 다음 각각의 경우에 맞게 시간 종속 위치 추정 함수를 적용함으로써 모든 질의 시점에서 불확실한 객체의 위치를 표현할 수 있다. 이때 객체의 불확실한 위치를 추정하는 시간 종속 위치 추정 함수는 응용에 따라 다른 함수로 대체가 가능하다. 이 연구에서는 연산 처리의 단순화를 위하여 선형 함수를 사용하였다.

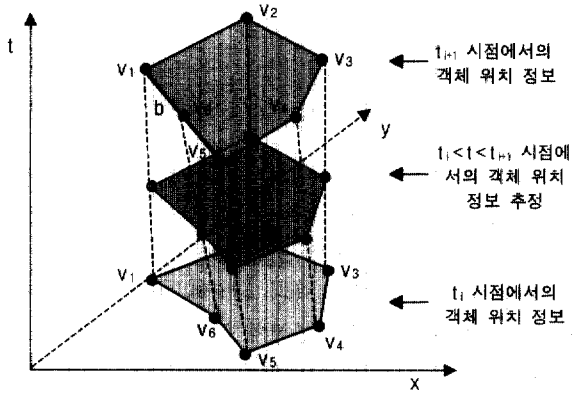
3.2 이동 영역

이동 영역 또한 이동 점과 마찬가지로 일정 시간 간격별로 샘플링하여 얻은 객체의 시공간 정보와 시간 종속 위치 추정 함수를 이용하여 연속적인 움직임을 표현할 수 있다. 그러나 이동 점과 달리 이동 영역은 폴리곤으로 표현되며 변화속성으로 위치 정보 뿐만 아니라 모양 정보도 가진다. 따라서 이동 영역의 연속적인 움직임을 표현하기 위해서는 특별한 방법론이 필요하다. 이 논문에서는 각 시점의 정점 정보를 통하여 이동 영역의 모양 변화 처리를 가능하게 하고 연속적인 이동 영역의 위치 정보를 얻을 수 있는 방법을 아래와 같이 제안한다.

예를 들어 t_i, t_{i+1} 각 시점에서 (그림 2)와 같이 이동 영역 정보가 샘플링 되었다고 가정하자. $[t_i, t_{i+1}]$ 시간 구간 내에서 이동 영역 객체의 위치 정보는 t_i와 t_{i+1}각 시점에서의 이동 영역 객체의 정점 정보들에 질의 시점에 따른 위치 추정 함수를 적용함으로써 추정할 수 있다. 즉 두 시점에서 폴리곤을 구성하는 각 정점들에 시간 종속 위치 추정 함수를 적용함으로써 객체의 모양 정보 및 위치 정보를 추정할 수 있다. 그러나 이 방법을 사용하기 위해서는 시간 종속 위치 추정 함수가 적용될 t_i와 t_{i+1}시점의 폴리곤을 구성하는 정점수가 같아야 하는 제약이 있다. 객체의 기하 정보 추정을 위해 새로운 정점을 생성시켜 두 시점의 정점수를 같게 해준 다음 이에 보간법을 적용하는 방법은 Shumilov의 key-frame 보간법[11]에서 소개된 방법이다. Shumilov는 샘플링이 이루어진 각 시점에 이전 시점에서의 공간 정보와 근사시킨 상태(pre-discretization)와 이후 시점에서의 공간 정보와 근사시킨 상태정보(post-discretization)를 모두 저장함으로써 보간법이 적용될 시간 구간의 정점수를 같게 만들어 준다. 그러나 이 방법은 오랜 시간에 걸쳐 기하정보가 변화하는 객체를 대상으로 하며 한 시점에 두 개의 상태정보를 저장하기 때문에 이를 이동 영역 표현에 그대로 적용하면 대량의 저장 공간을 요구하는 단점이 있다.



(a) 정점수가 증가된 경우



(b) 정점수가 감소된 경우

(그림 2) t 시점에서의 이동 영역 위치 정보 추정

따라서 이 논문에서는 샘플링이 이루어진 각 시점에는 객체의 샘플링 정보만을 저장하고 정점 수가 이전 시점과 다르게 변화한 경우 변화 정보를 추정할 수 있는 정점을 생성한다. 그런 다음 생성된 정점을 포함하는 폴리곤 정보를 객체 위치 추정을 위해 별도로 저장하는 방법을 사용한다. 즉 (그림 2)(a)처럼 t_{i+1}시점에서의 정점 수가 t_i시점보다 증가하는 경우 t_i시점의 폴리곤 정보 중에서 폴리곤을 구성하는 마지막 라인 세그먼트 a의 중점에 새로운 정점 v_6을 생성 시킨후 이를 포함하는 폴리곤 정보를 객체 위치 추정을 위해 별도로 저장한다. 정점수가 감소한 (그림 2)(b)의 경우에는 t_{i+1}시점의 폴리곤 정보 중에서 마지막 라인 세그먼트 b의 중점에 새로운 정점 v_6을 생성 시킨후 이들

정보를 객체 위치 추정을 위해 별도로 저장한다. 그런 다음 예를 들어 (그림 2)(b)처럼 t시점 ($t_i < t < t_{i+1}$)에 대한 사용자 질의가 들어온 경우 t_i시점에서의 객체 정보와 모양 정보 추정을 위해 별도로 저장한 객체 정보에 시간 종속 위치 추정 함수를 적용함으로 해서 질의 시점 t에서 객체의 불확실한 위치를 추정할 수 있다. 이러한 방법을 통해 이동 영역의 저장 되지 않은 정보를 추정하고 연속적인 객체의 움직임을 표현한다.

3.3 이동 객체 연산자

이 장에서는 제시한 이동 객체 표현 방법을 바탕으로 모든 질의 시점에서 객체의 정보를 요구하는 사용자 질의를 처리하기 위한 이동 객체 연산자를 제시한다. 그리고 질의 예를 통해 연산자의 사용을 보인다. 제안 연산자들은 [19]에서 제안한 시공간 이동 객체 연산자들을 확장한 것으로 연산자들이 적용되는 대상 객체에 따라 세 부류로 나누었으며 기타 연산을 위해 기타 연산자를 두었다. 연산자 표기를 위해 사용한 기호 중에서 Ti는 질의 시간 구간을 의미하며 Tp는 질의 시점을 의미한다. 그리고 N은 객체의 이름을 나타낸다.

3.3.1 이동 점 연산

제안된 시공간 이동 객체 연산자들 중 이동 점 객체를 대상으로 수행되는 연산자들은 <표 2>와 같다.

〈표 2〉 이동 점 연산자

대상 객체	연산자	표 기(입력값 → 출력값)
이동 점	MPTraj	Ti, N → trajctory
	MPLength	Ti, N → length
	MPDistance	Tp, N1, N2 → distance, Tp
	MPNearest	Tp, N → nearestN, distance, Tp
	MPFarthest	Tp, N → farthestN, distance, Tp
	MPatVel	Tp, N → average velocity, Tp

<표 2>는 이동 점 객체를 대상으로 수행되는 연산자들을 나타낸다. MPTraj는 이동 점 객체의 궤적 정보를 반환하며 MPLength 연산은 질의 시간 구간 동안에 객체가 이동한 길이값을 나타낸다. MPDistance 연산은 두 이동 점 객체간의 직선 거리값을 나타내며 MPNearest 연산자는 질의 시점에서 질의 객체와 가장 가까이 존재하는 객체 정보를 반환한다. 또한 MPatVel 연산자는 질의 시점에서 객체의 평균 속도 정보를 반환한다.

<표 2>의 이동 점 연산자들을 사용자 질의에 적용시킨 결과는 (질의 1)의 표현과 같다. 이때 (질의 1)에서 사용되는 시나리오는 화재지역과 소방차, 응급차를 대상으로 하였으며 이용되는 추상 스키마는 <표 3>과 같다.

〈표 3〉 예제 질의를 위한 릴레이션

FireTruck { *name* : *string*, *geometry* : *moving point* }
EMTruck { *name* : *string*, *geometry* : *moving point* }
FireArea { *name* : *string*, *geometry* : *moving region* }

FireTruck과 EMTruck은 이동 점 객체를 표현한 릴레이션이며 FireArea는 이동 영역 객체를 나타낸다.

(질의 1) 이동 점 연산 질의 : "현재 소방차 N1Ftruck의 위치에서 1km보다 가까이 존재하는 응급차를 검색하라"와 같은 질의는 다음과 같은 구문을 통해 표현할 수 있다.

```
Select e.name
From FireTruck, EMTruck e
Where ProjValue(MPDistance(current, 'N1Ftruck', e.name))
      < 1000 ;
```

즉 현재 시간값과 N1Ftruck 그리고 EMTruck 개체들을 입력값으로 받아서 현재 시간에서 N1Ftruck 소방차와 응급차들간의 거리값을 구한 후 그 값이 1km보다 작을 경우의 응급차 이름을 추출함으로 해서 (질의 1)의 결과를 도출할 수 있다.

3.3.2 이동 영역 및 기타 연산

제안하는 연산자들 중에서 이동 영역에 적용되는 연산자와 이동 객체 모두에 적용되는 연산자, 그리고 기타 연산을 위한 연산자는 <표 4>와 같이 정리할 수 있다.

〈표 4〉 이동 영역 및 기타 연산자

대상 객체	연산자	표 기(입력값 → 출력값)
이동 영역	MRTrav	Ti, N → traverse
	MRArea	Tp, N → area, Tp
	MRPerimeter	Tp, N → perimeter, Tp
	MRatCenter	Tp, N → location, Tp
	MRatDistance	Tp, N1, N2 → distance, Tp
이동 점 - 이동 영역	MOatTime	Tp, N → location, Tp
	MOatDistance	Tp, mpN1, mrN2 → distance, Tp
기 타	ProjTime	Value, Tp → Tp
	ProjValue	Value, Tp → Value
	CalMin	{(Value, Tp)} → MinValue, Tp
	CalMax	{(Value, Tp)} → MaxValue, Tp

<표 4>는 이동 영역 및 기타 연산자들을 나타낸 것으로 이동 영역 연산자 중에서 MRTrav는 이동 영역 객체가 이동한 이동 모습을 반환한다. MRArea는 질의 시점에서 객체의 면적 정보를 제공하는 역할을 하며 MRPerimeter는 질의 시점에서 질의 객체의 둘레길이 정보를 나타낸다. MRat Center 연산자는 질의 시점에서 질의 객체의 중심점 정보를

반환하며 MRatDistance 연산자는 질의 시점에서 두 이동 영역 객체간의 거리값을 제공한다. 이동 점 객체와 이동 영역 객체 모두에 적용되는 연산자들 중에서 MOatTime 연산자는 제안하는 연산자들의 가장 근본이 되는 연산자로써 모든 질의 시점에서 객체의 위치 정보를 반환하는 역할을 한다. 객체간의 거리 연산이나 둘레길이, 속도, 최근접 연산 등은 이 연산자를 통해 질의 시점에서의 객체 정보를 얻고 이에 기존 공간 연산을 그대로 적용함으로 해서 결과값을 얻을 수 있다. MOatDistance 연산자는 질의 시점에서 질의된 이동 점 객체와 이동 영역 객체 사이의 거리값을 계산한다. 기타 ProjTime과 ProjValue는 시간과 값의 쌍 중에서 각각 시간 값과 특정 값을 추출하는 역할을 한다. CalMin과 CalMax 연산자는 각각 최소값과 최대값을 반환하는 연산자이다.

<표 3>의 추상 스키마를 이용하여 <표 4>의 연산자들을 사용자 질의에 적용 시킨 결과는 (질의 2)와 (질의 3)의 표현과 같다.

(질의 2) 이동 영역 연산 질의 : "현재 산불 A의 위치와 피해 면적은 얼마인가 ?"는 아래와 같은 질의 구문을 통해 나타낼 수 있다.

```
Select ProjValue(MRArea(current, '산불A')),
       ProjValue(MOatTime(current, '산불A'))
From FireArea ;
```

(질의 2)에서는 현재 시점과 산불 A라는 입력값을 MRArea와 MOatTime연산에 적용함으로써 현재 시간에서의 산불 A의 피해 면적과 위치 정보를 얻을 수 있다.

(질의 3) 이동 점-이동 영역 연산 질의 : "현재 산불 B지역에 N1Ftruck가 도달하기 위해 남은 거리는 얼마인가?"는 다음과 같은 질의 구문을 통해 표현 할 수 있다.

```
Select ProjValue(MOatDistance(current, 'N1Ftruck', '산불B'))
From FireArea, FireTruck ;
```

(질의 3)의 표현에선 N1Ftruck과 산불 B사이의 거리값을 얻기 위해 MOatDistance 연산자를 사용하였다. 이 연산자는 내부적으로 현재시간에서의 두 객체의 위치 정보를 얻기 위해 MOatTime 연산자와 MRatCenter 연산자를 사용한다.

4. 이동 객체 저장 및 연산 알고리즘

이 장에서는 제시한 이동 객체 데이터 모델의 주요 설계 알고리즘을 설명한다. 제시 알고리즘은 이동 영역 표현 시 정점 정보가 변화한 경우 연속적인 이동 정보 표현을 위해 필요한 객체 정보를 생성하는 알고리즘이다. 그리고 특정 질의 시점에서 이동 객체의 위치 정보를 검색하는 MOatTime 연산자의 알고리즘을 소개한다. 다음으로 제시할 알고리즘은 MOatTime 연산 알고리즘 내에서 불확실한 이동 영역의 위치 정보를 추정하기 위해 모양 변화 처리 기법과 시간 종속 위치 추정 함수를 사용하는 이동 영역의 불확실한 위치 정보 추정 알고리즘이다.

4.1 이동 영역의 정점 정보 생성

(알고리즘 1)은 각 샘플링 시점에서 영역을 구성하는 정점수를 통해 이동 영역의 모양 변화를 확인한다. 그리고 모양 변화가 발생한 경우, 객체의 연속적인 기하 정보를 추정하기 위해 필요한 객체 정보를 생성한다.

```
function make_polyInfo(Tp, PolyInfo)
input : 샘플링 시점(Tp), 샘플링 시점에서의 이동 영역 정보
        (PolyInfo)
output : 이동 영역 기하 정보 추정을 위한 다각형 정보(InferVinfo)
method :
  if (preVinfo == null) then
    preVinfo ← PolyInfo ;
  endif
  if (preVinfo ≠ null) then
    compare PolyInfo's vertex number with preVinfo's vertex
    number
    if (PolyInfo's vertex number > preVinfo's vertex number)
       then  /* 증가한 경우 */
      search the last segment of preVinfo ;
      dVinfo ← divideSegment(line segment) ;
      InferVinfo ← preVinfo ∪ dVinfo ;
    endif
    if (PolyInfo's vertex number < preVinfo's vertex number)
       then /* 감소한 경우 */
      search the last segment of PolyInfo ;
      dVinfo ← divideSegment(line segment) ;
      InferVinfo ← PolyInfo ∪ dVinfo ;
    endif
  endif
  return InferVinfo ;
end make_polyInfo
```

(알고리즘 1) 이동 영역의 정점 정보 생성 알고리즘

(알고리즘 1)에서 preVinfo는 이전 샘플링 시점에서 이동 영역의 다각형 정보를 나타내며 InferVinfo는 이동 영역 객체 위치 추정을 위해 생성한 객체 정보를 나타낸다. divide Segment 함수는 입력 받은 라인 세그먼트를 분할하여 분할된 지점의 정보를 포인트 형태로 반환한다. 이때 분할할 라인 세그먼트는 preVinfo와 PolyInfo를 구성하는 각 정점들의 순서를 기반으로 매핑한 후 정점수가 증가한 경우에는 preVinfo의 마지막 라인 세그먼트를, 정점수가 감소한 경우에는 PolyInfo의 마지막 라인 세그먼트로 결정한다. divide Segment 함수를 통해 반환된 정보는 dVinfo에 기록된다. (알고리즘 1)의 수행 과정은 다음과 같다. 즉 이동 영역 객체 정보가 일정 시간 간격별로 샘플링 되어 들어오는 경우

먼저 이전 시점의 정점 정보와 비교하여 정점 수 변화를 확인한다. 확인 결과 정점 수에 변화가 있는 경우 정점수가 증가한 경우와 감소한 경우에 따라 preVinfo와 PolyInfo정보를 기반으로 divideSegment 함수를 수행함으로 해서 새로운 정점을 생성한다. 그 후 생성된 정점 정보를 preVinfo나 PolyInfo 정보에 더해줌으로 해서 이동 영역의 위치 추정을 위해 필요한 객체 정보를 생성한다. 일정 시간 간격별로 샘플링 된 객체의 정점 정보가 시점별로 많이 변화하면 위치 추정시 오차가 크게 발생하는 문제가 있다. 따라서 이 논문에서는 좁은 샘플링 시간 간격을 사용함으로써 샘플링 시 발생되는 정점 변화율이 크지 않다고 가정한다.

4.2 이동 객체의 위치 정보 검색

(알고리즘 2)는 과거, 현재, 가까운 미래의 어느 특정 질의 시점에서 이동 객체의 위치 정보를 검색하여 반환하는 MOatTime 연산자에 대한 것이다. 이 연산자는 질의 시점에 따른 객체 위치 정보 반환시 저장된 정보는 검색하여 그대로 반환하고 저장되지 않은 객체 정보에 대해서는 시간 종속 위치 추정 함수와 모양 변화 처리 방법을 통하여 이동 객체의 위치 정보를 추출해 낸다. 제 3.2절에서 소개한 이동 객체 연산자 대부분은 MOatTime 연산자를 기반으로 질의 시점에서의 객체 정보를 취득하며 취득된 정보를 기존의 공간 연산에 적용함으로써 이동 객체 연산을 수행한다. (알고리즘 2)는 MOatTime 연산자의 연산 수행 과정 중에서 질의 객체 유형 확인 후 수행되는 과정을 나타낸다.

```
Function MOatTime(Tp, N)
input : 질의 시점(Tp), 질의 객체 이름(N)
output : 질의 시점, 질의 객체 위치 정보(result)
method :
  Check the Query time
  if (query time == past time) then                    /* 질의 시점이
                                                          과거인 경우 */
    Search the information of target object which satisfies the
      constraint : (mo_name = N) and (vf <= Tp < vt)
    if (target object information exists) then         /* 객체 정보가
                                                          존재하는 경우 */
      mrresult ← the existent information endif
    else
      mrresult ← HUncertain(id, Tp) endif              /* 과거 위치 추정
                                                          함수 사용 */
  endif
  if (query time == current or future time) then /* 질의 시점이
                                                    현재 또는 미래인 경우 */
    Search the information of target object which satisfies the
      constraint : (mo_name = N) and (Tp = vt)
    if (target object information exists) then         /* 객체 정보가
                                                          존재하는 경우 */
      mrresult ← the existent information endif
    else
      mrresult ← FUncertain(id, Tp) endif              /* 미래 위치 추정
                                                          함수 사용 */
  endif
  return result
end MOatTime
```

(알고리즘 2) 이동 객체의 위치 정보 검색 알고리즘

(알고리즘 2)는 특정 질의 시점에서의 이동 점 객체나 이동 영역 객체의 위치 정보를 반환하는 MOatTime 연산자의 알고리즘으로 질의 객체 유형 확인 후에 수행되는 과정을 나타낸다. 질의 객체가 이동 점이든 이동 영역이든 먼저 질의 시점이 과거인지 또는 현재 및 미래인지를 확인한다. 그리고 질의 시점에서 객체 정보가 저장되어 있는지 확인한다. 객체 정보가 저장되어 있는 경우에는 저장된 정보를 그대로 검색하여 반환하고 그렇지 않은 경우에는 질의 시점이 과거인지 미래인지에 따라 그에 맞는 위치 추정 함수를 사용한다. 이때 이동 점의 경우에는 제 3.1절에서 소개한 방법을 이용하여 과거 및 미래 시점에서의 객체의 불확실한 위치 정보를 추정하고 이동 영역의 경우에는 제 3.2절에서 소개한 방법을 통해 각 시점에서의 이동 영역 객체의 불확실한 위치 정보를 추정한다. 이동 영역의 경우 불확실한 위치 추정시에 모양 변화 처리과정을 거치는데 이는 다음절의 (알고리즘 3)과 같다.

4.3 이동 영역의 불확실한 위치 정보 추정

이 알고리즘은 이동 영역 객체의 연속적인 움직임을 표현하기 위하여 저장 되지 않은 시점에서의 객체 정보를 추정하는 것으로 (알고리즘 3)과 같다. 이 알고리즘 내에서는 객체의 불확실한 기하 정보를 추정하기 위하여 이동 점과 마찬가지로 시간에 대한 선형의 위치 추정 함수를 사용한다. 그러나 이동 점과 달리 객체 정보를 구성하는 모든 정점들에 위치 추정 함수가 적용 되어지며 위치 추정 이전에 모양 변화 처리가 이루어진다.

```
function ShUncertain(id, Tp)
input : 객체 아이디(id), 질의 시점(Tp)
output : 질의 시점에서의 객체 기하 정보(result)
method :
  Search the information of target object which satisfies the
    constraint : (mo_id=id) and (vf <= Tp < vt)
  Check the shape-change of target object
  if (shape-change == 0) then /* 모양 변화가 없는 경우 */
    for each vertex belonging to information of target object at
      vf, vt do    /* vf와 vt에서의 정점들의 각 쌍에 시간 종속
                      위치 추정 함수 적용 */
      result ← result ∪ result of UlocationF function endfor
  endif
  if (shape-change == 1) then   /* 모양의 커진 경우 : 정점 수 증가 */
    for each vertex belonging to information of target object at
      vt, and polych's value do
      /* vt에서의 정점들과 polych의 정점들의 각 쌍에 시간
         종속 위치 추정 함수 적용 */
```

```
        result ← result ∪ result of UlocationF function endfor
    endif
    if (shape-change == -1) then    /* 모양이 작아진 경우 :
                                       정점 수 감소 */
      for each vertex belonging to information of target object at
        vf, and polych's value do
        /* vf에서의 정점들과 polych의 정점들의 각 쌍에 시간
           종속 위치 추정 함수 적용 */
        result ← result ∪ result of UlocationF function endfor
    endif
    return result   /* 추정된 질의 객체의 기하정보(다각형) */
end ShUncertain
```

(알고리즘 3) 이동 영역의 불확실한 위치 정보 추정 알고리즘

불확실한 이동 영역의 위치 정보를 추정하는 (알고리즘 3)의 동작 과정을 보면 다음과 같다. 먼저 객체의 모양 변화를 확인하고 정점수가 증가 또는 감소한 경우에 따라 시간 종속 위치 추정 함수에 적용할 정점 정보들을 다르게 사용한다. 이때 사용되는 객체의 정점 정보는 질의 시점 바로 이전과 이후에서의 샘플링 정보와 이동 영역의 모양 변화 추정을 위해 (알고리즘 1)을 통해 생성한 정보이다. 예를 들어 질의 시점이 포함된 시간 구간$[t_i, t_{i+1}]$에서 객체가 줄어든 경우 (알고리즘 1)을 통해 생성된 정점들과 t_i시점에서 샘플링 된 정점들의 쌍에 시간 종속 위치 추정 함수를 적용함으로 해서 질의 시점에서의 객체의 위치 정보를 추정할 수 있다.

5. 적용 및 평가

이 장에서는 제안한 이동 객체 데이터 모델을 산불 관리 응용에 적용시켜 구현하고 이에 대한 질의 예를 보인다. 그리고 이를 통하여 기존 모델과 제안 모델을 비교 평가한다.

제안 모델을 적용한 산불 관리 응용은 Windows2000 운영체제와 SQLServer2000 데이터베이스, 그리고 Java(IDK1.3)를 통하여 구현 하였다. 구현시 이용한 실험 데이터는 2002/02/28/10/00~2002/02/28/13/20 유효 시간 구간 동안에 2분과 20분을 주기로 소방차 3대와 하나의 화재 지역을 대상으로 샘플링이 이루어졌다고 가정하고 프로그램을 통해 임의로 생성하였다. 이러한 샘플 데이터들은 다음 절에서 제시하는 이동 객체 데이터베이스 구조 형태로 저장 된다.

5.1 산불 관리를 위한 이동 객체 데이터베이스

이 절에서는 일반 속성 정보와 시공간 정보를 동시에 가지는 실험 데이터들을 저장하기 위한 이동 객체 데이터베이스의 스키마를 소개하고 각각에 대해서 설명한다.

5.1.1 데이터베이스 스키마

이산적으로 표현한 이동 점과 이동 영역 객체의 시공간 정보는 다음과 같은 5개의 릴레이션 스키마 형태로 저장 관리한다.

```
MOIdentity {MO_ID, MO_type, MO_name}
MPhistory {MP_ID, Vfrom, Vto, PointF, PointT}
MPcurrent {MP_ID, Vfrom, Vto, PointF, PointT}
MRhistory {MR_ID, Vfrom, Vto, PolyF, PolyT, Ch_check, PolyCh}
MRcurrent {MR_ID, Vfrom, Vto, PolyF, PolyT, Ch_check, PolyCh}
```

MOIdentity 스키마는 이동 객체의 일반 속성 정보를 관리하기 위한 것이며 MPhistory와 MPcurrent 스키마는 이동 점 객체의 시공간 정보를 관리하기 위한 것이다. 그리고 MR history와 MRcurrent는 이동 영역 객체의 시공간 정보를 관리하기 위한 스키마이다. 이 논문에서는 실세계에서 빈번하게 발생하는 현재로부터 가까운 미래 질의를 효율적으로 처리하기 위하여 이동 점과 이동 영역의 이력 정보와 현재 정보를 각각 별도의 릴레이션으로 저장 관리한다. 각각의 릴레이션 스키마에 대한 설명은 다음과 같다.

5.1.2 일반 속성 릴레이션

MOIdentity는 데이터베이스로 관리되는 이동 객체를 식별하기 위한 릴레이션으로 이동 객체를 표현하기 위한 일반 속성값들을 가진다. 객체 정보가 저장된 모습은 <표 5>와 같다.

<표 5> MOIdentity

MO_ID	MO_type	MO_name
135	1	Fire 135
102	0	ChFireT 102
...	...	...

<표 5>는 데이터베이스로 관리하는 이동 객체 식별 릴레이션으로 객체 아이디와 객체 유형 정보, 그리고 객체의 이름과 같은 일반 속성 정보를 제공한다. 객체의 유형을 표현하는 MO_type 속성의 경우 데이터베이스로 관리되는 객체가 이동 영역인 경우 1값을, 이동 점인 경우 0값으로 표현한다.

5.1.3 이동 점 릴레이션

이동 점 객체 정보는 <표 6>과 <표 7>과 같은 스키마 형태로 저장된다. MPhistory와 MPcurrent 릴레이션 각각은 이동 점의 이력 정보와 현재 정보를 저장하는 릴레이션을 나타낸다.

<표 6> MPhistory

MP_ID	Vfrom	Vto	PointF	PointT
102	2002/01/28/10/00	2002/01/28/10/02	(5, 8)	(7, 11)
102	2002/01/28/10/02	2002/01/28/10/04	(7, 11)	(10, 15)
...	...	...	...	...

〈표 7〉 MPcurrent

MP_ID	Vfrom	Vto	PointF	PointT
102	2002/01/28/10/02	2002/01/28/10/04	(7, 11)	(10, 15)
...	...	...	...	...

이동 점에 대한 사용자의 과거시점 질의를 처리하고자 하는 경우 MPhistory 릴레이션을 사용한다. 그리고 각 객체마다 가장 최근에 샘플링된 두 시점의 위치 정보를 저장, 관리하는 MPcurrent 릴레이션은 현재나 가까운 미래 시점에 대한 질의 처리시 이용한다.

5.1.4 이동 영역 릴레이션

이동 영역 객체 정보는 <표 8>과 <표 9>와 같은 스키마 형태로 저장된다. MRhistory와 MRcurrent는 이동 영역의 이력 정보와 현재 정보를 저장하는 릴레이션이다.

〈표 8〉 MRhistory

MR_ID	Vfrom	Vto	PolyF	PolyT	Ch_check	PolyCh
135	2002/01/28/10/00	2002/01/28/10/20	(1, 5), (3, 3), (5, 5)	(1, 4), (2, 2), (5, 3), (4, 6)	1	(1, 5), (3, 3), (5, 5), (3, 6)
135	2002/01/28/10/20	2002/01/28/10/40	(1, 4), (2, 2), (5, 3), (4, 6)	(3, 6), (4, 4), (7, 5), (6, 8)	0	null
...	...	...	...	...	...	...

〈표 9〉 MRcurrent

MR_ID	Vfrom	Vto	PolyF	PolyT	Ch_check	PolyCh
135	2002/01/28/10/20	2002/01/28/10/40	(1, 4), (2, 2), (5, 3), (4, 6)	(3, 6), (4, 4), (7, 5), (6, 8)	0	null
...	...	...	...	...	...	...

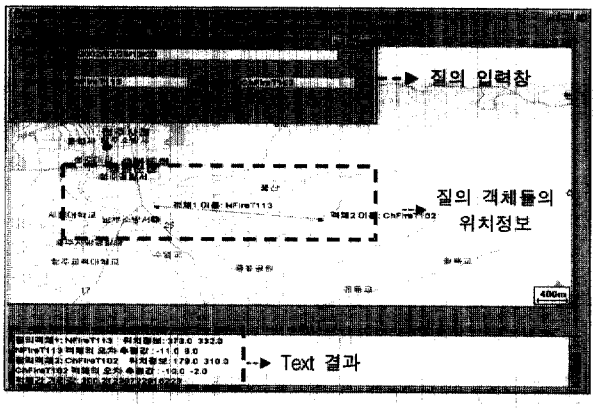
<표 8>은 이동 영역 객체의 이력 정보를 저장하는 릴레이션으로 샘플 데이터가 저장된 모습을 보여준다. 샘플링된 객체 정보 저장시에 객체의 정점수가 증가, 감소한 경우에 따라 Ch_check 속성값을 1 또는 -1로 조정하고 변화된 모양을 표현하기 위해 필요한 추가적 정보를 PolyCh 속성값에 저장한다. 정점수의 변화가 없는 경우 Ch_check 속성값은 0 값을, PolyCh 속성은 null값을 가진다. <표 9>의 MRcurrent는 이동 영역 객체들의 최근 샘플링 정보를 저장하는 릴레이션으로 사용자의 현재나 가까운 미래 질의 처리시 이용한다.

MRhistory와 MRcurrent 릴레이션의 PolyCh 속성은 객체의 연속적인 기하 정보를 추정하기 위해 이용되는 속성으로 4.1절의 (알고리즘 1)과 같은 방법을 통해 속성값을 생성한다.

5.2 질의 수행 결과

이 절에서는 제안 모델이 적용된 산불 관리 응용에서 다음과 같은 4개의 질의 수행 예를 보임으로써 제안 모델이 과거, 현재, 가까운 미래 시점의 연산 처리를 지원함을 보인다.

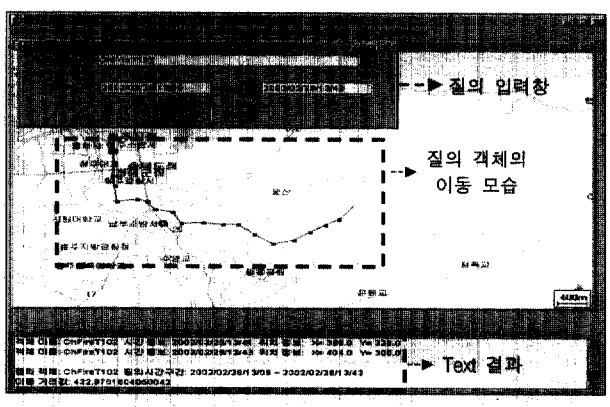
(질의 4) "소방차 NFireT113이 목적지에 도달한 2002/02/28/13/21 시간에서 다른 소방차 ChFireT102와의 거리는 얼마인가?"



(그림 3) MPDistance 연산자 수행 결과

(질의 4)를 수행하기 위해서는 두 이동점 객체간의 거리값을 반환하는 MPDistance 연산자가 필요하다. MPDistance 연산자는 질의 처리 내용이 질의 입력창을 통해 입력되면 먼저 MOatTime 연산자를 통해 질의 시점에서 두 질의 객체의 공간 정보를 획득한다. 그 후, 획득된 공간 정보에 거리 연산을 적용함으로써 두 질의 객체간의 거리값을 구한다. (그림 3)은 질의 시점 값으로 2002/02/28/13/21을 그리고 질의 객체로 NFireT113과 ChFireT102값을 입력하여 MPDistance 연산을 수행한 결과 화면이다.

(질의 5) "소방차 ChFireT102의 전체 이동 거리는 얼마인가?"

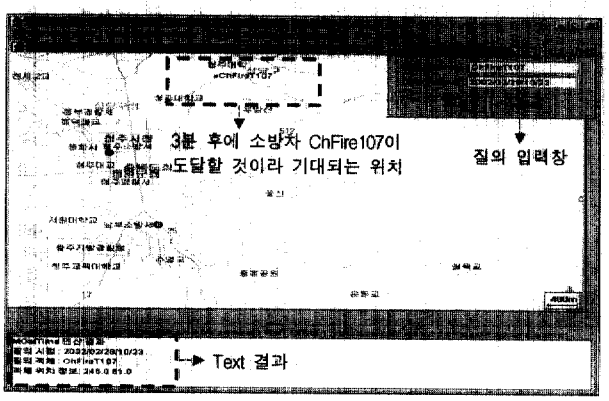


(그림 4) MPLength 연산자 수행 결과

(질의 5)를 수행하기 위해서는 질의 시간 구간 동안 이동점 객체의 이동 거리값을 반환하는 MPLength 연산자가 필요하다. (그림 4)는 질의 객체 ChFireT102와 전체 이동 시간 구간값 2002/02/28/13/05-2002/02/28/13/43를 입력하여

MPLength 연산을 수행한 결과 화면이다. 이때 화면 결과는 질의 객체가 질의 시간 구간 동안 이동한 모습을 보여주며 연산 수행 결과 반환된 이동 거리값은 텍스트 결과 창에 보여 진다.

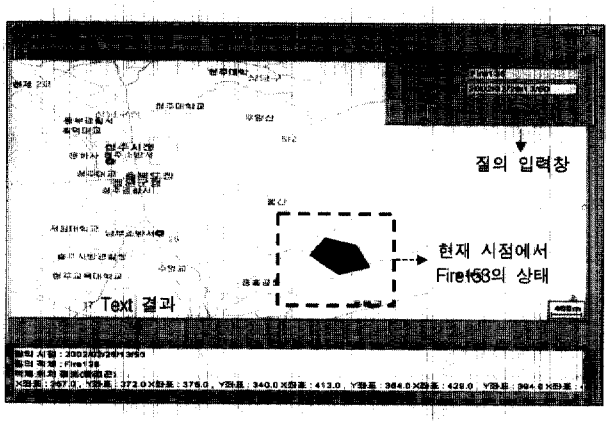
(질의 6) "3분후에 소방차 ChFire107이 도달할 것이라 기대되는 위치는 어디인가?"
(가정 : 현재 시간 - 2002/02/28/10/20)



(그림 5) 이동 점 객체에 대한 MOatTime연산자 수행 결과

(질의 6)을 수행하기 위해서는 과거, 현재, 가까운 미래 시점에서 객체의 위치 정보를 반환해 주는 MOatTime 연산자가 필요하다. (그림 5)는 3분 후의 질의 시점 2002/02/28/10/23값과 질의 객체 이름 ChFire107를 입력값으로 받아서 MOatTime 연산자를 수행한 결과 화면이다. 화면 결과는 3분후에 ChFire107이 도달할 것이라 기대되는 위치를 보여주며 좌표 정보는 텍스트 결과 창에 보여진다. 이때 도출된 결과 값은 미래 위치 추정 함수를 통해 계산 된다.

(질의 7) "Fire135의 현재 화재 상태를 보여라."(가정 : 현재 시간 - 2002/02/28/13/50)



(그림 6) 이동 영역 객체에 대한 MOatTime 연산자 수행 결과

(질의 7)을 수행하기 위해선 질의 시점에서의 객체 위치 정보를 반환하는 MOatTime 연산자가 필요하다. (그림 6)은 질의 시점 2002/02/28/13/50값과 이동 영역 객체 Fire135 값을 입력값으로 받아서 MOatTime 연산자를 수행한 결과 화면이다. 이때 도출된 결과는 이동 영역의 모양 변화 처리 및 위치 추정을 거쳐 계산 된다. 위의 질의에 대한 연산 결과, 해당 시점에서의 폴리곤 정보 즉, 객체의 위치 정보가 화면과 텍스트 결과 창에 나타난다.

5.3 평 가

최근 시공간 데이터베이스 관련 연구에서는 이동 객체 데이터를 관리하기 위한 여러 가지 모델들을 제시하였다. 기존 이동 객체 관리 모델들을 데이터 표현이나 연산 처리 등의 부분에서 요약하면 <표 10>과 같다.

〈표 10〉 이동 객체 관리 모델 비교

모 델	데이터 표현		연산 처리 시점		
	이동점	이동영역	과거	현재	미래
DOMINO의 MOST모델	○	×	×	○	○
CHOROCHRONOS의 시공간 데이터 모델	○	○	○	△	×
제약 데이터베이스 기반 접근	○	○	○	△	×
제안 모델	○	○	○	○	○

○ : 처리 가능 △ : 조건적으로 가능 × : 처리하기 어려움

<표 10>에서처럼 MOST 모델은 이동 점 객체만을 대상으로 하며 동적 속성을 통해 객체에 대한 현재로부터 가까운 미래 시점까지의 질의만을 다룬다. 따라서 [1,5]에서 지적된 바와 같이 이동 영역 및 객체의 전체 과거 정보를 다루지 않는 단점이 있다. 또한 CHOROCHRONOS 컨소시엄에서 제시한 시공간 데이터 모델은 이동 점과 이동 영역을 모두 다루고 있지만 [14]에서 지적된 바와 같이 이동 객체의 과거 이력 정보를 다루는데 적합하며 연산 처리 시 유닛 형태로 데이터베이스에 관리되는 객체 정보만을 다룬다. 이때 이 모델은 현재 또는 가까운 미래 시점에 대한 객체의 유닛 정보를 관리하고 있지 않기 때문에 데이터베이스로 관리되지 않는 시점 즉 현재 또는 가까운 미래 시점에서의 연산 처리를 지원하지 않는다. 만약 현재 시점에서의 객체 정보가 유닛 형태로 데이터베이스에 저장되어 있다면 현재 시점의 연산 처리는 가능하다. 제약 데이터베이스를 기반으로 한 접근의 경우 시간 차원과 공간 차원이 분리된 제약 조건들로 이동 객체를 표현함으로써 객체의 속도 정보를 알 수 없는 단점이 있다[3]. 따라서 미래 위치 표현을 위해 필요한 객체의 최근 속도 정보를 알 수 없기 때문에 데이터베이스로 관리 되지 않는 현재 또는 가까운 미래 시점에서의 연산 처리를 지원하지 못한다.

이 논문에서 제시하고 있는 이동 객체 데이터 모델은 기존 모델들의 문제점을 해결하기 위하여 시간 종속 위치 추정 함수와 모양 변화 처리 기법을 통해 이동 점과 이동 영역의 연속적인 움직임을 모두 표현할 수 있도록 하였다. 또

한 질의 시점 별로 저장 되지 않은 객체의 불확실한 위치를 추정하기 위한 과거와 미래 각각의 위치 추정 함수를 사용함으로써 연산 처리시 과거 시점에서 가까운 미래 시점까지를 모두 처리할 수 있도록 하였다. 따라서 제시 모델은 사용자 질의 시점을 제한하였던 기존 모델들과 달리 과거, 현재, 가까운 미래 시점에 대한 사용자 질의를 모두 처리할 수 있는 장점을 가진다. 그러나 현재로부터 가까운 미래 질의를 효율적으로 처리하기 위하여 이동 객체의 샘플링 정보를 이력 정보와 현재 정보로 나누어 저장함으로 해서 추가적인 저장 공간이 필요하다는 단점이 있다. 이를 해결하기 위해 이동 객체의 가까운 미래 질의 처리를 효율적으로 처리하기 위한 저장 기법이나 인덱싱 기법에 대한 연구가 필요하다.

6. 결 론

최근 무선 환경을 기반으로 차량 추적 시스템, 산불 관리 시스템, 디지털 전장 시스템 등과 같은 다양한 시공간 응용들이 개발되고 있다. 이러한 시공간 응용들은 시간상에서 연속적으로 변화하는 객체 정보를 저장 관리하고 시간과 공간 제약을 가지는 사용자 질의를 처리할 수 있는 데이터 모델을 필요로 한다.

그러나 기존의 관계형 모델을 이용하여 이동 객체 데이터를 관리하는 경우 매 시간 마다 변경된 모든 정보들을 저장할 수 없기 때문에 일정 시간 간격별로 샘플링 된 객체 정보만을 저장한다. 이러한 경우 저장 되지 않은 정보를 요구하는 사용자 질의를 처리하기 어려운 문제점이 발생된다. 또한 기존 이동 객체 관련 모델들은 이동 점 객체의 현재 및 미래의 위치 추적에 초점을 맞추거나 제약 조건이나 유닛으로 표현된 이동 객체의 과거 정보만을 다루고 있기 때문에 질의 객체나 질의 시간을 제한하는 단점이 있다.

따라서 이 논문에서는 모양 변화 처리 기법과 과거와 미래의 위치 추정 함수를 사용함으로써 이동 점 및 이동 영역 객체의 연속적인 움직임을 표현하고 모든 질의 시점에서의 이동 객체 관련 연산을 처리 할 수 있는 이동 객체 데이터 모델을 제안하였다. 제안 모델에서는 이동 점의 연속적인 움직임을 표현해 주기 위해 시간 종속 위치 추정 함수를 사용하였다. 또한, 이동 영역의 경우, 모양 변화 처리 기법과 시간 종속 위치 추정 함수를 사용하여 객체의 연속적인 이동 모습을 표현하였다. 아울러 연산 처리 시 질의 시점별로 과거와 미래 위치 추정 함수를 사용함으로써 모든 질의 시점에서의 연산 처리가 가능하도록 하였다. 아울러 제안 모델의 주요 설계 알고리즘들을 기술하였으며 제안 모델을 산불 관리 응용에 적용 평가하였다. 그리고 이를 통해 응급 구조 시스템, 물류 관리 시스템, 오염 방제 시스템 등과 같은 다양한 응용들에 제안한 이동 객체 데이터 모델을 적용 할 수 있음을 보였다.

이 연구를 기존 다른 연구들과 비교 평가한 결과 과거, 현재, 가까운 미래 시점에 대한 사용자 질의를 모두 처리할 수 있는 장점을 가진다. 그러나 가까운 미래 질의를 효율적으로 처리하기 위해 현재 정보를 별도의 릴레이션에 저장함으로써 추가적인 저장 공간을 필요로 하는 단점이 있다. 따라서 향후 이를 해결하기 위한 저장 기법이나 인덱싱 기법에 대한 연구가 필요하다.

현재 제안 모델 중 연산자들을 개방형 차량 추적 시스템에 적용하는 연구가 진행 중이며 향후에는 실세계 응용에서 필요로 되어지는 궤적/위상 관련 이동 객체 연산자들을 기존 모델에 추가하는 연구가 수행 될 것이다. 또한 효율적인 질의 처리를 위한 저장 기법이나 인덱싱 기법에 대한 연구가 수행 될 것이며 아울러 제안 모델을 실세계 응용에 적용 평가하는 작업이 수행될 것이다.

참 고 문 헌

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

[2] L. Forlizzi, R. H. Guting, E. Nardelli and M. Schneider, "A Data Model and Data Structures for Moving Objects Databases," In Proc. of the Intl. Conference on ACM SIGMOD, pp.319-330, 2000.

[3] S. Grumbach, P. Rigaux and L. Segoufin, "Spatio-Temporal Data Handling with Constraint," ACM International Workshop on Advances in Geographic Information Systems, 1998.

[4] S. Grumbach, P. Rigaux, M. Scholl and L. Segoufin, "The Design and Implementation of DEDALE," Verso Report number 171, 1999.

[5] R. H. Guting, M. H. Bohlen, M. Erwig, C. S. Jensen and N. A. Lorentzos, M. Schneider and M. Vazirgiannis, "A Foundation for Representing and Querying Moving Objects," ACM Transactions on Database Systems, Vol.25, No.1, pp.1-42, 2000.

[6] R. H. Guting, M. H. Bohlen, M. Erwing, C. S. Jensen, et al., "A Foundation for Representing and Querying Moving Object," Chorochronos TR, CH-98-3, 1998.

[7] M. H. Huh, Y. A. Ahn and K. H. Ryu, "Moving Object Location Change Function using Cubic Spline Interpolation," SCI2002, 2002.

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

[9] 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 Science(CaGIS), Special Issue on Dealing with Time, Vol.26, No.2, pp.109-123, 1999.

[10] D. Pfoser and C. S. Jensen, "Capturing the Uncertainty of Moving Object Representations," In Proc. of the Intl. Conference on SSDBM, pp.111-132, 1999.

[11] S. Shumilov and J. Siebeck, "Database Support for Temporal 3D Data Extending the GeoToolkit," EC-GI & GIS Workshop, 2001.

[12] A. P. Sistla, O. Wolfson, S. Chamberlain and S. Dao, "Modeling and Querying Moving Objects," ICDE, pp.422-432, 1997.

[13] A. P. Sistla, O. Wolfson, S. Chamberlain and S. Dao, "Querying the Uncertain Position of Moving Objects," Dagstuhl, pp.310-337, 1997.

[14] O. Wolfson, B. Xu, S. Chamberlain and L. Jiang, "Moving Object Databases : Issues and Solutions," In Proc. Of the Intl. Confercence on Scientific and Statistical Database Management, 1998.

[15] O. Wolfson, A. P. Sistla, B. Xu, J. hou and S. Chamberlain, "DOMINO : Databases fOr MovINg Objects tracking," In Proc. of the Intl. Conference on SIGMOD, pp.547-549, 1999.

[16] 신기수, 안윤애, 배종철, 정영진, 류근호, "GIS를 이용한 시공간 이동 객체 관리 시스템", 정보처리학회논문지D, 제8-D권 제2호, pp.105-116, 2001.

[17] 안윤애, 류근호, 조동래, "전장분석을 위한 이동 객체의 위치 예측 시스템", 정보과학회논문지, 제8권 제6호, 2002.

[18] 양은주, 정영진, 장승연, 안윤애, 류근호, "이동 객체의 경로 추정을 위한 다항 회귀 함수 적용 및 구현", 추계학술발표논문집, 정보처리학회, 2001.

[19] 장승연, 정영진, 안윤애, 류근호, "시공간 이동 객체 연산자의 설계", 춘계학술발표논문집, 한국정보과학회, 2002.

[20] 정영진, 배종철, 안윤애, 류근호, "Smallworld를 이용한 이동 객체 관리 및 위치 예측기의 구현", 춘계학술발표논문집, 한국정보과학회, 2001.

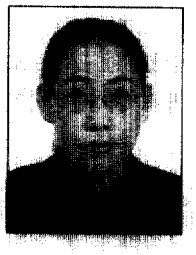
장 승 연

e-mail : syjang@dblab.chungbuk.ac.kr

2001년 충북대학교 경영정보학과(경영학사), 멀티미디어공학(공학사)

2001년~현재 충북대학교 대학원 전자계산학과 석사과정

관심분야 : 시공간 데이터베이스, 이동 객체 데이터베이스, 지리정보 시스템



안 윤 애

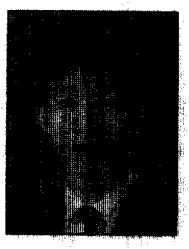
e-mail : yeahn@dblab.chungbuk.ac.kr

1993년 한남대학교 전자계산공학과 (공학사)

1996년 충북대학교 대학원 전자계산학과 (이학석사)

1999년~현재 충북대학교 대학원 전자계산학과 박사수료

관심분야 : 시공간 데이터베이스, 모바일 데이터베이스, 지리정보 시스템, 지식기반 시스템



류 근 호

e-mail : khryu@dblab.chungbuk.ac.kr

1976년 숭실대학교 전산학과(이학사)

1980년 연세대학교 공학대학원 전산전공 (공학석사)

1988년 연세대학교 대학원 전산전공 (공학박사)

1976~1986년 육군군수 지원사 전산실(ROTC 장교), 한국전자통신연구원(연구원), 한국방송통신대 전산학과(조교수) 근무

1989년~1991년 Univ. of Arizona Research Staff(TempIS 연구원, Temporal DB)

1986년~현재 충북대학교 전기전자및컴퓨터공학부 교수

관심분야 : 시간 데이터베이스, 시공간 데이터베이스, Temporal GIS, 객체 및 지식기반 시스템, 지식기반 정보검색 시스템, 데이터 마이닝 및 데이터베이스 보안, 바이오인포메틱스