위치기반 서비스(LBS)를 위한 이동체 위치획득 기법

민 경 욱† · 조 대 수††

요 약

무선인터넷 기술의 발달과 응용의 확산으로 인하여 위치정보(Location Information)를 이용한 서비스 형태는 아주 다양하여 지고 있으며, LBS(Location-Based Services)라는 형태로 국내외 시장에서 급속히 확장하고 있다. 이러한 서비스의 기본정보인 위치정보는 고부가가치의 정보로 인정되고 있으며 또한 이러한 위치를 기반으로 한 서비스는 기존의 GIS(Geographic Information System)를 일반사람들에게 생활 속의 정보로 밀접하게 활용할 수 있게 되었다. 즉, LBS로 인하여 GIS는 새로운 패러다임의 전환을 맞이하게 되었으며 대용량 이동체 데이터베이스(MODB : Moving Object Database)는 LBS를 위한 핵심 플랫폼 기능으로 인식되고 있다. 이러한 대용량 이동체 데이터베이스와 이동통신망과의 위치획득 인터페이스를 통한 위치정보 획득 기능은 아주 중요한 요소가 되며, 향후 위치를 기반으로 한 다양한 서비스가 제공될 경우, 이동성을 가지는 수많은 고객의 위치정보를 획득함으로써 심각한 시스템의 부하는 미리 예측할 수가 있다. 즉, 대용량의 이동체의 위치정보를 획득함에 있어서, 심각한 통신부하로 인한 LBS 플랫폼의 성능 저하가 발생 할 수가 있다. 따라서, MODB에서의 위치획득 기능은 안정적인 LBS 플랫폼의 구동을 위하여 통신부하의 증가를 방지함과 동시에 안정적인 위치정보 획득 기능을 보장하여야 한다. 이에 본 논문에서는 LBS 플랫폼이 이동통신망으로부터 대용량의 위치정보를 획득함에 있어서, 이동체의 과거 이동 패턴을 이용하여 불필요한 위치정보 획득 회수를 줄임으로써 시스템의 부하를 줄이고자 한다. 본 논문에서는 이러한 이동체의 이동 패턴을 이용하여 다양한 위치획득 모델을 설계하였으며, 시뮬레이션 환경에서 위치획득 모델을 구현함으로써 성능을 평가하였다.

Techniques for Acquisition of Moving Object Location in LBS

Kyoung-Wook Min† · Dae-Soo Cho††

ABSTRACT

The types of service using location Information are being various and extending their domain as wireless internet technology is developing and its application part is widespread, so it is prospected that LBS (Location-Based Services) will be killer application in wireless internet services. This location information is basic and high value-added information, and this information services make prior GIS (Geographic Information System) to be useful to anybody. The acquisition of this location information from moving object is very important part in LBS. Also, the interfacing of acquisition of moving object between MODB and telecommunication network is being very important function in LBS. After this, when LBS are familiar to everybody, we can predict that LBS system load is so heavy for the acquisition of so many subscribers and vehicles. That is to say, LBS platform performance is fallen off because of overhead increment of acquiring moving object between MODB and wireless telecommunication network. So, to make stable state of LBS platform, in this MODB system, acquisition of moving object location part must provide guarantee location information as well as reduce telecommunication overhead. In this paper, we intend that system load be lessening as reducing the number of acquisition of unnecessary moving object location. We study problems in acquiring a huge number of moving objects location and design some acquisition model using past moving pattern of each object to reduce telecommunication overhead. And after implementation these models, we estimate performance of each model.

키워드 : 이동체(Moving Object), 이동체 데이터베이스(MODB), 위치기반 서비스(LBS), 위치정보 획득(Acquisition of Location)

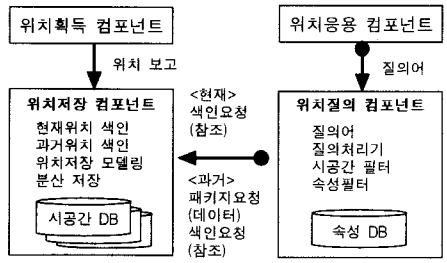
1. 서 론

최근 무선인터넷의 급속한 발전으로 인해서, 개인의 mobility의 특성에 의한 서비스는 점차적으로 확대 되고 있다. 위치기반 서비스(Location-Based Service : LBS)는 이동 통신 단말기 소지자의 위치를 실시간으로 추적해 이를 활용한 다양한 응용 서비스를 제공하는 무선인터넷 서비스를 말하며 현재 국내에서의 위치기반 서비스는 주변 시설 정보 서비스, 추적 서비스, 교통, 항법 서비스 등 현재의 위치정보를 이용한 서비스가 시행되고 있으며, 향후에는 최적 경로 선택 기능, 이동 차량의 동적 트래킹(tracking), 이동체 주변의 시설물 컨텐츠(contents) 검색, 부가 가치 시공간 분석, GPS 위치 데이터와 지도 데이터간의 맵매칭(map matching) 그리고 과거의 위치 이력정보를 이용한 시공간 데이터마이닝 및 CRM에서 마케팅을 위한 고부가가치 정보 서비스 등으로 확대 될 전망이다[12]. 또한 LBS는 기존 GIS의 방향을 전환시키는 새로운 분야이며, 빠른 속도로 그 영역이 확대되고 있다. 이러한 LBS를 가능하게 하는 LBS 플랫폼에서는 이동체의 위치정보를 저장하고 관리하는 기능

† 정 회 원 : 한국전자통신연구원 LBS 연구팀 연구원
†† 정 회 원 : 한국전자통신연구원 LBS 연구팀 선임 연구원
논문접수 : 2003년 2월 11일, 심사완료 : 2003년 6월 23일

을 필수적으로 포함하고 있어야 하며, LBS의 서비스의 질(quality)을 결정짓는 중요한 요소이다. 현재의 LBS는 이동체의 현재 위치정보를 이용한 실시간 서비스에 국한되어 있으며, 향후에는 과거의 위치정보를 이용한 고가의 정보 서비스가 됨이 틀림없다. 또한 앞으로 이동 장비 사용자의 이동 위치 및 개인 각각의 차별화된 위치 서비스에 대한 요구가 증가 되고 있는 추세이다.

시간의 흐름에 따라 객체가 이동하면서 그 위치 및 모양을 연속적으로 변경하는 특징을 가지는 데이터를 이동체(Moving Object)[10]라 하며, LBS 플랫폼의 가장 핵심은 이러한 이동체의 정보를 효과적으로 관리하는 방법과 관련된 부분이다. 즉, 이동체의 정보를 다양한 디바이스를 통해서 획득하는 부분부터 시작하여 대용량 정보를 데이터베이스에서 모델링하고, 저장하고 질의하는 부분까지의 연구가 중요한 역할을 차지하고 있다. 이러한 부분의 기술을 이동체 데이터베이스(Moving Object Database)라고 하며 이동체의 현재의 위치정보뿐만 아니라, 과거의 위치 이력정보까지도 관리하는 것을 기본으로 하고 있으며 그 구조는 (그림 1)과 같다. 구조에 대해서 간단히 살펴보면, 위치획득 컴포넌트는 이동체의 위치정보를 이동통신 망에 요구하고 그 결과로 위치정보를 획득하는 역할을 한다. 특히, 이 부분에서는 대용량의 이동체의 위치정보를 획득함에 있어서 발생하는 통신부하 및 이로 인한 시스템 부하를 감소시키기 위한 역할까지 수행하게 된다.



(그림 1) 이동체 데이터베이스 구조

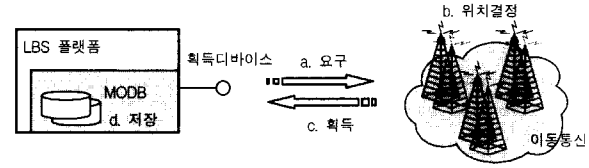
위치 저장 컴포넌트는 획득한 이동체의 위치정보를 저장하는 역할을 한다. 특히, 이동체의 과거 위치정보는 시간이 지날수록 무한정 늘어나기 때문에 그 방대한 양을 어떻게 저장하고 관리하는 지가 중요한 문제이다. 또한 방대한 과거 이력 데이터를 얼마나 빨리 검색할 수 있는지도 아주 중요한 문제이다. 현재 위치데이터의 빠른 검색을 위한 현재 색인, 과거의 이동 궤적에 대한 검색을 위한 과거 색인, 이동체 데이터 모델링, 분산 저장 기능 등을 위치 저장 컴포넌트에서 제공하고 있다. 위치 질의 컴포넌트는 상위 응용 컴포넌트로부터의 요청 질의를 처리하는 컴포넌트이다. 속성필터 및 위치저장 컴포넌트의 현재 및 과거 위치색인을 참조하여 질의를 수행하고 그 결과 값을 리턴하는 기능을 포함하고 있다.

향후 LBS 대상 이동체가 증가하면 증가할수록 시스템의 통신부하에 의한 성능은 심각하게 저하 될 수 있다. 즉, LBS 플랫폼의 MODB에서 위치정보를 저장하기 위해서 이동통신 망에 위치를 요구하고 획득한 후 저장하기까지의 일련의 과정에 있어서, 이동체의 개수가 수백만, 수천만이 될수록, 획득하여 저장하기까지의 시간이 계속 증가할 수 있다. 여기서 어느 정도의 획득 시간 간격(acquisition interval)을 가지고 획득하느냐에 따라서도 시스템의 성능을 좌우하게 된다. 예를 들어 대상 이동체가 100만개이고 획득 시간 간격이 1분이라고 하였을 경우 하루 동안의 획득 회수와 저장 용량은 아래와 같다.

- *acquisition count per 1 day = 1,000,000×1440*
  = 1,440,000,000 times
- *space per 1 day = 1,000,000×s×1440*
  = 1.44×s Gbytes(s = sizeof moving object)

만약 통신부하로 인하여, 1분이라는 획득 시간 간격 동안 100만개의 이동체 위치를 획득하지 못하였을 경우, 이미 1분이라는 시간이 지나서 획득한 이동체의 정보는 이미 과거의 정보가 되며, 정보의 신뢰성을 보장할 수 없으며, 이로 인하여 MODB에서 획득 컴포넌트의 부하가 증가되어 전체 시스템의 성능 저하를 초래하게 된다. 따라서 시스템의 안정적인 구동을 위해서는 이동체의 개수가 증가하면 증가할수록 획득 시간 간격을 늘릴 필요가 있다.

(그림 2)는 LBS 플랫폼에서 이동통신망에 위치를 요구하여 획득 한 후 저장하기까지의 일련의 흐름을 나타내고 있다. 여기서 이동체의 개수가 증가할수록 시스템의 성능을 저하시키는 부분은 다음과 같다.



- a, c : 위치획득을 위한 통신부하
- b : 통신망에서 위치를 결정하기 위한 내부적인 신호 트래픽(traffic) 부하
- d : 위치정보의 중복 저장을 피하기 위한 필터링(filtering) 및 색인 구축 부하

(그림 2) 이동체 위치획득 과정

본 논문에서는 이러한 LBS의 안정적인 구동을 위하여 위치획득 컴포넌트에서 대용량의 이동체의 위치정보를 획득함에 있어서 통신부하를 최소화하는 부분에 대한 기법에 관하여 기술하였다. 일반적으로 이동체는 목적지를 향하여 이동하며, 최적의 경로를 선택하는 경향이 있고[8], 이동체는 동일한 이동 패턴을 반복할 확률이 높다[9]. 이에 본 논문에서는 이동체의 이동 패턴을 파악하여 획득 회수를 줄임으로써 통신부하를 줄이기 위한 획득 모델과 전체 시스템의 안정적인 구동을 위하여 동적으로 이동체의 획득 시간 간격을 조

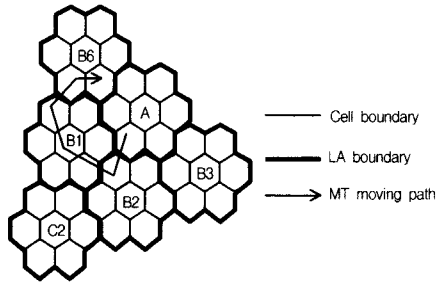
절하는 획득 모델에 대하여 연구하였다. 즉, 이동체 데이터베이스 관리 시스템에서의 위치정보를 획득함에 있어서, 통신부하를 줄이기 위한 획득 모델을 제시하고, 성능 평가를 한다. 먼저, 2장에서는 이동체 데이터베이스 및 위치획득의 관련연구에 대해서 살펴 볼 것이며, 3장에서는 이동통신망에서의 위치정보를 획득함에 있어서, 통신부하를 최소화하기 위한 몇 가지 위치획득 모델에 대해서 살펴 볼 것이다. 그리고 4장에서는 이러한 위치획득 컴포넌트를 이동체 데이터베이스 관리 시스템의 서브 시스템으로 구현하기 위한 시스템 구조에 대해서 살펴보고, 이를 바탕으로 하여 각각의 획득 모델의 성능을 평가할 것이다. 마지막으로 5장에서는 결론 및 향후 연구방향에 대해서 제시를 할 것이다.

2. 관련 연구

무선이동통신 환경이 발달함에 따라, 먼저 이동통신분야에서 무선 단말기의 위치를 관리하는 연구가 활발히 진행되어 왔으며, 데이터베이스 분야에서도 이동체 데이터베이스 분야에서 많은 연구가 진행되어 왔었다. 이동통신분야에서는 이동통신망에서 위치를 계산하는 방법과 시스템의 오버헤드를 최소화하면서 효율적으로 위치를 관리하기 위한 페이징(paging) 기법과 위치보고(location update)와 관련된 연구가 주를 이루었으며, 데이터베이스 분야에서는 LBS를 가능하게 하는 대용량의 이동객체를 저장 및 관리하기 위한 데이터 모델, 색인, 불확실성관리, 질의처리 등에 관한 연구가 주를 이루고 있다.

2.1 이동통신망에서 위치정보 관리

이동통신망에서의 위치 관리 작업은 페이징과 위치보고로 이루어진다. 이동통신망에서 착신호가 들어왔을 때 이를 해당 이동단말기에 연결시키기 위해 통신망이 단말기의 위치를 찾는 과정을 터미널 페이징이라고 하고, 페이징을 용이하게 하기 위해 단말기가 자신의 위치를 통신망에게 보고하는 과정을 위치보고라 한다[25]. 단말기가 위치보고를 자주 할수록 단말기의 위치에 대한 정보가 정확하기 때문에 단말기의 위치를 찾는데 드는 페이징 비용은 줄어들지만 위치보고 비용은 늘어난다. 따라서 위치보고와 페이징 비용 사이에는



(그림 3) Location Area(LA)의 예

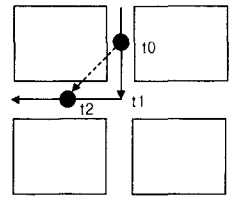
trade-off가 존재하며, 효율적인 위치관리 기법을 위해서는 사용자의 이동패턴(mobility pattern)을 고려하여 설계해야 한다. 또한 전체 페이징 비용을 감소시키기 위하여, (그림 3)과 같이 전체 통신망을 하나 이상의 셀로 구성된 location area(LA)로 재구성하는 개념을 사용하고 있다.

위치를 보고하는 방법 중에서 정해진 기준에 따라 보고하는 방법을 많이 사용하고 있지만, 단말기가 다른 LA로 이동할 때마다 위치보고를 하는 것은 비효율적이며, 위치보고 시행 조건을 정하고 이 조건이 만족되었을 때 위치보고를 하는 많은 방법들이 제안되었다. 일정 보고 시간 간격을 정하여, 그 시간이 지나면 다시 위치보고를 하는 time-based location update[1], 이동거리 threshold를 정해 놓고, 위치보고를 한 마지막 LA에서 정해진 거리만큼 떨어진 LA로 들어갈 때 위치보고를 하는 distance-based location update[2-4], LA로의 이동회수 threshold를 정해놓고, 위치보고를 한 후 정해진 회수만큼의 다른 LA로의 이동이 있었을 때 위치보고를 하는 movement-based location update[5]등이 있다. 즉, 현재까지의 연구를 정리하면, 이동통신망에서는 이동 단말기의 현재 위치를 유지하기 위하여 페이징과 위치보고의 trade-off를 적절히 조절하고 있으며, LBS를 위해서 위치 요구가 발생할 경우에는 단말기의 위치를 찾기 위해 터미널 페이징 과정을 거쳐서 위치정보를 제공해주고 있으며, 일반적으로 통신망의 게이트웨이 개수에 따라 다르지만 수초가 소요된다.

2.2 이동체 데이터베이스에서 위치정보 관리

이동객체를 관리하는 측면에서, 이동체 데이터베이스에서는 데이터의 모델링(Modeling), 질의 표현(Query, Operator), 인덱싱(Indexing) 기법, 그리고 불확실성(Uncertainty) 처리 등의 분야에서 주로 연구가 진행되어 왔다[6]. 데이터 모델링, 질의 처리, 인덱싱에 관한 연구는 기존의 공간 데이터베이스(Spatial database), 또는 시공간 데이터베이스(Spatial-temporal database)의 확장된 부분으로 많은 연구가 진행되어 왔다. 데이터 모델링은 데이터의 종류 및 연산자에 초점을 맞춘 모델링[25, 26], 이동체의 위치 변화에 초점을 둔 모델링[6, 16] 등이 있다. 이동체의 질의 표현에 관한 연구는 관계형 SQL 형태의 질의어를 포함시키는 방법[26]과 이동체의 동적 속성을 다루기 위한 FTL(Future Temporal Logic)[6, 16, 17, 19] 질의 표현에 대해서 연구해 왔다. 인덱싱에 관한 연구는 기존의 공간 데이터베이스의 공간색인을 확장한 형태의 색인에 대하여 많은 연구가 진행되고 있다[28-30]. 불확실성 처리에 대한 연구는 이동통신망에서 위치를 측정하는데 발생하는 에러와 실제 위치를 샘플링하는 에러로 인해서 발생되는 부분이다[16]. 여기서 샘플링 에러는 얼마나 자주 위치를 획득 해야만 하는가에 대한 부분으로, 위치를 자주 획득할수록 정보에 대한 신뢰가 높아지는 대신 획득 부하와 저장용량의 증가 및 저장할지 말지를 결

정하는 필터링의 부하가 발생하고, 반대로 아주 드물게 획득할수록 정보에 대한 신뢰가 떨어지는 대신 시스템의 성능은 안정적으로 유지될 수 있다[19]. (그림 4)와 같이 t0, t2 시간에 이동체의 위치를 획득하였을 경우보다, t0, t1, t2 시간에 획득한 이동체의 이동 정보가 더욱 더 신뢰가 있다.



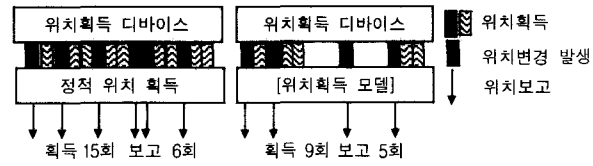
(그림 4) 이동체 샘플링 에러

(그림 4)에서 t0, t2의 시간에 위치를 획득하였을 경우, [t0, t2]사이의 시간에 위치정보를 요구할 경우에 대한 불확실성의 해결책으로는 DOMINO 프로젝트의 연구 결과의 일부분인 [6, 16, 19]에서는 객체의 현재 위치, 속도, 방향을 토대로 하여 객체의 이동을 예측하는 방법을 제안하고 있으며, 또한 데이터 모델의 확장, 질의어의 확장, 확률을 이용한 불확실한 질의에 대한 may, must 응답, 데이터베이스의 갱신비용과 불확실성 비용의 절충안 등을 제시하고 있다. [13]에서는 이러한 과거 이력정보 중에서 불확실한 과거정보뿐만 아니라 불확실한 미래 위치까지 예측하는 모델을 제시하고 있다. 또한 이동체 데이터베이스 분야에서는 이동체의 위치정보가 대용량이기 때문에 획득 후 정보를 저장하기 전에 저장할지에 대한 필터링 단계를 거쳐서 이동체를 선별하기도 한다[11, 20].

본 논문에서는 이동통신망과 LBS 플랫폼의 이동체 데이터베이스간의 위치정보를 요구하고 획득하는 회수를 줄임으로써 비용을 줄이는 것이 목적이다. 이를 통하여 이동통신망에서의 단말기의 불 필요한 위치를 파악하기 위한 오버헤드와 이동체 데이터베이스에서의 불확실성의 최소화, 그리고 저장하기 전 필터링 오버헤드의 최소화를 유지하는 것이다. 본 논문에서 이동체의 이동패턴 또는 시스템의 안정화를 유지할 수 있는 획득 데이터 모델을 통하여 LBS 플랫폼의 이동체 데이터베이스의 안정화를 유지할 수가 있다.

3. 위치획득 모델

이 절에서 제시하는 위치획득 모델의 목적은 이동체데이터베이스에 이동체의 위치 이동정보를 저장, 관리하기 위하여, 위치정보를 획득하려고 할 때, LBS 플랫폼의 이동체 데이터베이스와 이동통신망사이의 통신부하를 최소화하기 위한 것이다. 우선 이동체의 이동패턴을 파악하여 불필요한 위치정보를 획득하지 않음으로 해서 획득 회수를 줄이는 방법, 그리고 시스템의 안정적인 구동을 위하여 동적으로 이동체의 위치획득 간격을 적절히 조절함으로써 이동체 데이터베이스 시스템의 안정을 도모하는 것이다. 또한 획득 회수를 줄임으로써 발생되는 이동체 정보의 신뢰도 하락을 방지할 수 있도록 해야 한다. (그림 5)는 이러한 위치획득 모델을 적용하였을 경우 그 효과를 나타낸 것이다. 즉, 정적 위치획득의 경우에는 위치획득 15회가 발생하였다. 이 중 실제 보고가 필요한 회수는 6회이다. 즉, 위치정보가 변경되었을 경우에만 위치 보고가 필요하다. 반면, 위치획득 모델을 적용한 경우에는 다양한 획득 모델 알고리즘을 적용하여, 위치획득 9회, 위치보고 4회로, 실시간으로 획득 간격을 늘림으로써 통신부하를 줄일 수가 있다.

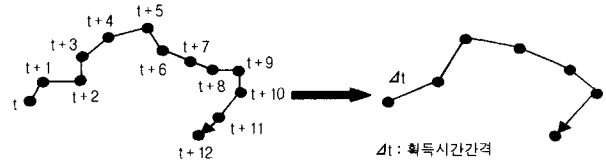


(그림 5) 위치획득 모델을 적용하였을 경우의 효과

본 논문에서는 다양한 상황에 적용할 수 있는 정적, 거리기반, 개별 거리기반, 그룹기반, 예측기반, 동적 위치획득의 6가지 획득 모델을 제시한다. 또한 이후에 획득 모델 각각의 성능 평가를 하기 위한 벤치마킹 모델을 제시한다.

3.1 정적 위치획득 모델

정적 위치획득(Static Acquisition) 모델은 위치획득 시간 간격을 일정하게 설정하여 위치를 획득하는 방법이다.



(그림 6) 정적 위치획득

위치를 획득하려고 하는 동일 집단(016 단말기 사용자 레이어 등)에 속한 모든 이동체의 획득 시간 간격(Δt)은 동일하다. 이 방법은 단순한 방법인 동시에 이동체의 개수에 따라 시스템이 안정적으로 구동 할 수 있는 획득 시간 간격을 설정하는 것이 중요하다. 획득 시간 간격(Δt)이 작으면 작을수록 신뢰도가 높은 이동정보를 획득하여 관리할 수 있는 반면 저장 용량은 증가할 것이다. 즉, 정보의 신뢰도와 시스템의 성능에는 trade-off관계가 성립하며 적절한 조화가 필요하다. 또한 이 방법은 다른 위치획득 모델과의 비교를 위해서 이용된다. 즉, Δt = 1(최소 시간 간격)인 정적 위치획득 모델과 다른 위치획득 모델의 위치획득 회수, 거리의 변화를 산출하여 성능 비교에 이용된다. (그림 7)은 정적 위치획득 모델의 구동 과정을 나타낸 것이다. 다른 위치획득 모델들도 (그림 7)의 구동 과정과 비슷하며, 획득 시간 간격 재설정 알고리즘은 서로 다르게 동작된다.

위치획득 모델 구동 과정

① acquireTimer = new AcquireTimer(⊿t) // timer instance 생성
② for all each moving object mo, acquireTimer.Queue.Enqueue(mo)
③ acquireTimer.Start( ) // 획득 구동 시작
④ acquireTimer.Callback( ) // ⊿t 시간이 되면 호출되는 callback 함수
④-1 for all each moving object mo in acquireTimer.Queue
call moDevice.Acquire (mo)
⑤ moDevice.Acquire (mo)
⑤-1 update mo.x, mo.y, mo.time from gateway server
⑤-2 acquisitionModel.ResetAcquireTime (mo)
⑤-3 about this mo, acquireTimer.Queue.Enqueue (mo)

정적 위치획득 모델 시간 간격 재설정 알고리즘

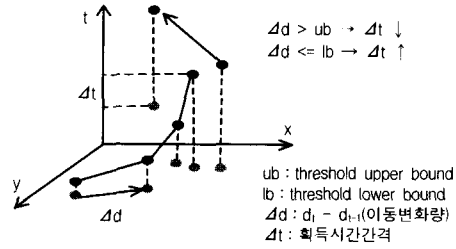
⑥ staticAcquisitionModel.ResetAcquireTime (mo)
⑥-1 mo.acquisitionInterval = ⊿t

(그림 7) 정적 위치획득 모델의 구동 과정 및 획득 시간 간격 재설정 알고리즘

①의 acquireTimer는 모든 위치획득 모델에서 1개 이상의 인스턴스를 가지게 된다. Thread로 동작하며, 획득 시간 간격마다 자동적으로 Callback 함수를 호출하게 된다. 즉, 정적 위치획득 모델은 모든 이동체의 획득 시간 간격이 동일하기 때문에 하나의 acquireTimer 인스턴스를 생성하게 된다. ④의 Callback 함수는 대기 큐에서 대기중인 모든 이동체에 대해서 ⑤의 위치획득 디바이스의 Acquire 함수를 호출하게 되며 실제 이동통신망의 게이트웨이 서버로부터 위치정보를 획득한 후 해당 이동체의 획득 시간 간격을 재설정하게 된다. 여기서 획득 모델별로 획득 시간 간격의 설정이 각각의 알고리즘에 따라서 달라진다. 정적 위치획득 모델에서는 모든 이동체의 획득 시간 간격이 동일함으로, 초기에 설정되었던 시간 간격 ⊿t의 값은 변하지 않는다.

3.2 거리기반 위치획득 모델

거리기반 위치획득(Distance-based Acquisition) 모델은 (그림 8)과 같이 이동체의 이동양의 변화를 이용하여 획득 시간 간격을 적절히 조정하는 것이다. 기본 개념은 이동체의 이동변화 양이 설정된 구간 threshold 값보다 큰 경우에는 획득 시간 간격을 줄이고, 보다 적은 경우에는 획득 시간 간격을 늘여서 통신부하를 줄임과 동시에 신뢰도 높은 정보를 유지하는 방법이다.



(그림 8) 거리기반 위치획득

이 방법의 중요한 요소는 획득 시간 간격을 조절하기 위한 기준 구간을 설정하는 것이다. 즉, (그림 8)과 같이 threshold upper bound, threshold lower bound의 구간을 어떻게 설정하는 지에 따라서 획득 시간 간격의 조절이 달라지게 된다. 이러한 구간 [lb, ub]의 값은 전체 집단(layer)의 몇회 위치정보 획득 후 평균 이동거리를 바탕으로 하여 설정하게 된다. 즉, 집단의 이동 특성(사람, 자동차 등)에 따라 다양하게 threshold 구간을 설정하여야 한다. 또한 획득 시간 간격의 증감 단계의 범위를 어느 정도 허용하느냐에 따라서 성능이 달라지게 된다. 만약 속도의 변화가 없는 이동체의 경우에는 획득 시간 간격이 무한정 증가할 수도 있다. 그래서 획득 시간 간격의 증감 단계를 고정적으로 정할 수가 있다. 예를 들어, 아래와 같이 6 단계로 고정하여 이동변화 양에 따라 획득 시간 간격을 범위 내에서 자유롭게 조절할 수가 있다.

- acquisitionIntervalSet = $\{\Delta t\times 2^{-2}, \Delta t\times 2^{-1}, \Delta t\times 2^{0}, \Delta t\times 2^{1}, \Delta t\times 2^{2}, \Delta t\times 2^{3}\}$

(그림 9)는 거리기반 위치획득 모델의 구동 과정과 획득 시간 간격 재설정 알고리즘을 나타낸다.

위치획득 모델 구동 과정

acquireTimers : a set of acquireTimer
acquisitionIntervalSet : {⊿t×2^-2, ⊿t×2^-1, ⊿t×2^0, ⊿t×2^1, ⊿t×2^2, ⊿t×2^3}
lb, ub : distance threshold interval
n : lb, ub를 계산하기 위한 초기 획득 회수
sd : sum of moving distance, ad : average moving distance
∂ : ad×ratio, 0 < ratio <= 0.1

① for all each time interval ⊿t in acquisitionIntervalSet,
acquireTimers [index] = new AcquireTimer (⊿t)
② for all each moving object mo,
acquireTimers [0].Queue.Enqueue (mo)
③ for all each acquireTimer in acquireTimers,
acquireTimer.Start ( )
④ acquireTimers[index].Callback ( )
④-1 for all moving object mo in
acquireTimers[index].Queue
call moDevice.Acquire (mo)
⑤ moDevice.Acquire (mo)
⑤-1 update mo.x, mo.y, mo.time from gateway server
⑤-2 acquisitionModel.ResetAcquireTime (mo)
⑤-3 index = GetTimerIndex (mo.⊿t)
⑤-4 about this mo,
acquireTimers[index].Queue.Enqueue(mo)

거리기반 위치획득 모델 시간 간격 재설정 알고리즘

⑥ distanceBasedAcquisitionModel.ResetAcquireTime(mo)
⑥-1 if # of acquisition < n×# of moving object then
sd = sd + mo.movingDistance and return this function
⑥-2 if not set lb, ub then
ad = sd / n×# of moving object
lb = ad ∂, ub = ad + ∂
⑥-3 if mo.movingDistnce <= lb then
increase mo.⊿t, increased
⊿t ∈ acquisitionIntervalSet
else if mo.movingDistance > ub then
decrease mo.⊿t, decreased
⊿t ∈ acquisitionIntervalSet

(그림 9) 거리기반 위치획득 모델의 구동 과정 및 획득 시간 간격 재설정 알고리즘

거리기반 위치획득 모델은 획득 시간 간격의 집합인 acquisitionIntervalSet의 크기만큼의 acquireTimer를 가지게 된다. 즉, 여러 개의 acquireTimer는 각각 서로 다른 획득 시간 간격 Δt을 가지게 되며, 설정된 시간에 획득 Callback 함수를 호출하게 된다. ①~④의 과정은 정적 위치획득 모델과 동일하며, ⑤-2에서 획득 시간 간격 재설정 한 후, ⑤-3에서는 바뀐 시간 간격 Δt에 해당하는 acquireTimer를 찾은 후 Queue에 삽입하게 된다. ⑥-1은 구간 [lb, ub]의 값을 설정하기 위하여, 초기 n 번까지는 모든 이동체의 위치를 획득 시간 간격 변화 없이 초기에 설정한 시간 간격 Δt로 획득 하게 되며, n 번 이상이 되었을 경우, ⑥-2에서 구간 [lb, ub]의 값을 설정하게 된다. ⑥-3에서는 실제 이동체의 이동변화 양(movingDistance)의 값을 구간 [lb, ub]의 값과 비교하여, 시간 간격을 증가 또는 감소시키게 된다.

3.3 개별 거리기반 위치획득 모델

개별 거리기반 위치획득(Respective Distance-based Acquisition) 모델은 앞서 설명한 거리기반 위치획득 모델을 약간 변형한 형태이다. 3.2절에서는 기준 거리 구간 [lb, ub]의 값을 전체 집단의 평균 이동거리를 바탕으로 하여 설정한 반면에, 이 모델에서는 전체 집단이 아닌 이동체 각각의 평균 이동거리를 바탕으로 하여 설정하게 된다. 즉, 각각의 이동체 인스턴스에 구간 [lb, ub]의 값을 멤버 변수로 가지게 된다. 또 다른 차이점은 각각의 이동체의 지속적인 상태를 파악하여 좀 더 안정적인 상태를 유지하기 위하여, 동적으로 구간을 변화시키게 된다. 각각의 이동체의 이동거리 변화 양의 변동 상황에 따른 상태를 아래와 같이 구분할 수 있다.

- Low State : 이동체의 획득 시간 간격이 최고의 값에 연속하여 머물러 있는 경우
- Stable State : 이동체의 획득 시간 간격이 lb, ub 사이에 연속하여 머물러 있는 경우
- High State : 이동체의 획득 시간 간격이 최소의 값에 연속하여 머물러 있는 경우

계속하여 Low State에 머물러 있는 이동체에 대해서는 구간 [lb, ub]를 감소시키고, High State에 계속하여 머물러 있는 경우에는 구간 [lb, ub]를 증가시킴으로써 이동체의 안정적인 상태를 유지할 수 있는 적절한 구간을 찾을 수 있다. 이렇게 함으로써 이동체 별로 안정적으로 동작할 수 있는 획득 시간 간격을 설정함으로써 전체 시스템의 안정화를 유지할 수 있다. (그림 10)은 개별 거리기반 위치획득 모델의 획득 시간 간격 재설정 알고리즘을 나타낸다.

①에서는 이동체의 threshold 구간 [lb, ub]를 설정하여 획득 시간 간격을 조절하는 부분이며, ②의 mo.Reset 함수에서 이동체의 threshold 구간 [lb, ub]의 값을 이동체의 상태를 파악하여 적절한 값으로 조절하는 부분이다.

개별 거리기반 위치획득 모델 시간 간격 재설정 알고리즘

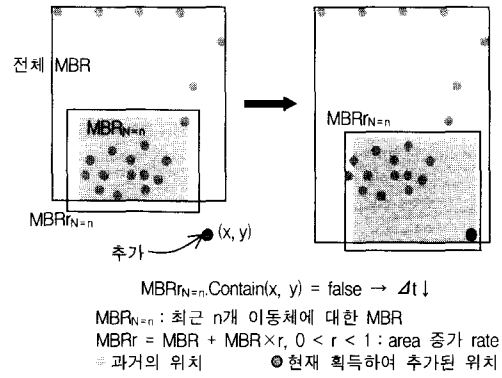
mo.lb, mo.up : distance threshold interval of mo
mo.sd : sum of moving distance of mo,
mo.ad : average moving distance of mo
unit : unit of increase or decrease value of threshold interval
LowerState : mo. Δt = max value in acquisitionIntervalSet
UpperState : mo. Δt = min value in acquisitionIntervalSet
continuousCount : continuous count of staying the same state,
th : threshold count of changing state

```
① respectiveDistanceBasedAcquisitionModel.ResetAcquireTime (mo)
      ①-1 if # of acquisition of this mo < n
            mo.sd = mo.sd + mo.movingDistance and return
                  this function
      ①-2 if not set mo.lb, mo.ub then
            mo.ad = mo.sd/n, mo.lb = mo.ad ∂,
            mo.ub = mo.ad + ∂
      ①-3 if mo.movingDistnce <= mo.lb then
            increase mo.Δt, increased
                  Δt ∈ acquisitionIntervalSet
          else if mo.movingDistance > mo.ub then
            decrease mo.Δt, decreased
                  Δt ∈ acquisitionIntervalSet
      ①-4 mo.Reset()
② mo.Reset()
      ②-1 if mo.State = LowerState and continuousCount
                        > th then
            lb = lb - unit, ub = ub - unit
      ②-2 else if mo.State = UpperState and continuousCount
                              > th then
            lb = lb + unit, ub = ub + unit
```

(그림 10) 개별 거리기반 위치획득 모델의 획득 시간 간격 재설정 알고리즘

3.4 그룹기반 위치획득 모델

그룹기반 위치획득(Group-based Acquisition) 모델은 이동체가 특정 시간대에 특정 지역에 군집하였다가 그 지역을 벗어날 때까지의 획득 시간 간격을 늘여서 통신부하를 줄이는 방법이다. 이 방법은 이동체의 최근에 위치한 곳들에 대한 최소경계사각형(MBR : Minimum Bounding Rectangle)을 중심으로 하여, 추가로 획득한 이동체의 위치가 MBR에 포함되는지 또는 벗어나는지를 판단하여 위치획득 시간 간격을 조절한다. 이동체 데이터베이스는 대용량의 이동체의 위치정보를 관리하고 빠른 접근을 지원해야 하기 때문에 main memory 데이터베이스의 사용이 필수적이다. 따라서 해당 이동체의 MBR을 계산하기 위해서 main memory 버퍼에 저장되어 있는 이동체의 이동정보를 바탕으로 하여 산출해 낼 수 있다. 여기서 MBR에 포함되지 않지만, 근접한 곳으로의 이동에 대해서도 포함되는 것으로 간주하기 위하여 MBR 증가 rate($0 < r < 1$)을 정의하여 MBR의 크기를 확대한다. 이 방법에서 시스템 성능에 영향을 미치는 주요 파라미터는 MBR을 계산하기 위하여 최근 몇 번까지의 획득 정보를 바탕으로 할 것 인지와, MBR의 증가 rate r 이다. 마찬가지로 집단의 특성을 파악하여 주요 파라미터를 적절히 설정할 수 있다.



(그림 11) 그룹기반 위치획득

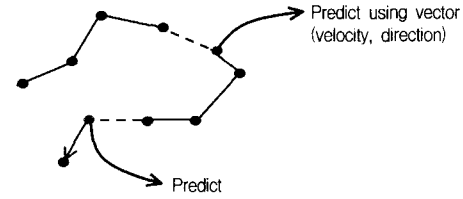
(그림 11)은 그룹 기반 위치획득 모델을 나타낸다. 이동체의 과거 전체를 포함하는 전체 MBR 및 최근에 위치하였던 지역을 포함하는 MBRr$_{N=n}$을 구성하고 있으며, 현재 새로운 위치가 획득되어 보고되었을 경우, 새로운 위치가 MBRr$_{N=n}$에 포함되는지 여부를 판단하여 획득 시간 간격을 조정하게 되고 전체 MBR 및 MBRr$_{N=n}$을 재구성하게 된다. (그림 12)는 그룹기반 위치획득 모델의 획득 시간 간격 재설정 알고리즘을 나타낸다.

그룹기반 위치획득 모델 시간 간격 재설정 알고리즘
n : MBR을 구성하기 위한 초기 획득 회수, *mo.Buffer : memory buffer, size = n* *r : MBR increase rate, 0 < r < 1*
① groupBasedAcquisitionModel.ResetAcquireTime(mo) ①-1 if # of acquisition of this mo < n insert mo.x, mo.y to mo.Buffer and return this function ①-2 mo.MBRr = MBR.Reset(mo.Buffer)×r ①-3 if mo.MBRr.Contain(mo.x, mo.y) incrcase mo.Δt, increased Δt ∈ acquisitionIntervalSet else decrease mo.Δt, decreased Δt ∈ acquisitionIntervalSet ①-4 insert mo.x, mo.y to mo.Buffer

(그림 12) 그룹기반 위치획득 모델의 획득 시간 간격 재설정 알고리즘

3.5 예측기반 위치획득 모델

예측기반 위치획득(Predict-based Acquisition) 모델은 과거 이동체의 이동정보인 방향, 속도를 이용하여 다음 이동 위치를 예측하는 방법이다. 즉, 통신부하가 심하여, 시스템의 성능이 급격히 떨어진 경우 이 방법을 적용할 수가 있다. 하지만, 이 방법은 많은 위험이 따른다. 왜냐하면 과거 이동 정보를 이용하는 데에는 한계가 있고, 이동체 별로 복잡한 예측 모델을 적용하기에는 그 오버헤드가 크기 때문이다. 따라서, 이 모델은 가장 기본적인 벡터 정보인 (방향, 속도, 시작점)을 이용하여 위치를 예측하며, 이동정보의 정확도 및 신뢰도를 보장하기 위하여 (그림 13)과 같이 일정 회수까지는 실제 위치를 획득한 후 통신부하를 고려하여 위치를 예측한다.



(그림 13) 예측기반 위치획득

또한 이 모델에서 이동체의 위치정보 신뢰도를 더욱 더 보장하기 위해서, 이동체의 위치예측을 모든 이동체를 대상으로 하지는 않는다. 각각의 이동체 별로 매번 위치를 획득할 때마다 예측 정확도를 계산하여 정확도가 일정 threshold를 넘어서는 이동체에 대해서만 위치를 예측할 대상이 되는 것이다. 예측 정확도는 과거에 위치를 획득하였을 당시, 예측 값과 실제의 값에 대한 거리, 방향 등을 고려하여 오차의 정도를 파악하여 계산하게 된다. (그림 14)는 예측기반 위치획득 모델의 구동 과정과 획득 시간 간격 재설정 알고리즘을 나타낸 그림이다.

위치획득 모델 구동 과정
th1 : 획득 효율 threshold, th1이하일 경우 예측 수행 *th2 : 예측정확도 threshold, th2 상일 경우 예측 수행* *predictDistance : 이전 획득 정보를 이용한 예측 이동 거리,* *movingDistance : 실제 이동거리* *predictDirection : 이전 획득 정보를 이용한 예측 방향,* *movingDirection : 실제 이동방향*
① acquireTimers[0] = new AcquireTimer (Δt), acquireTimers [1] = new AcquireTimer (Δt×2) ② for all each moving object mo, acquireTimers[0].Queue.Enqueue(mo) ③ acquireTimers [0].Start ( ), acquireTimers [1].Start ( ) ④ moDevice.Acquire (mo) ④-1 update mo.x, mo.y, mo.time from gateway server ④-2 mo.CalculatePredictAccuracy ( ) ④-3 calculate acquisitionEfficiency using mo.time, mo.Δt ④-4 acquisitionModel.ResetAcquireTime (mo) ⑤ mo.CalculatePredictAccuracy ( ) ⑤-1 distDiff = |predictDistance movingDistance| ⑤-2 angleDiff= |predictDirection movingDirection| ⑤-3 set predictAccuracy using distDiff, angleDiff
예측기반 위치획득 모델 시간 간격 재설정 알고리즘
⑥ predictBasedAcquisitionModel.ResetAcquireTime (mo) ⑥-1 if acquisitionEfficiency < th1 and mo.predictAccuracy > th2 then mo.LocationPredict ( ) mo.Δt = mo.Δt×2 acquireTimers [1].Enqueue (mo) ⑥-2 else acquireTimers [0].Enqueue (mo)

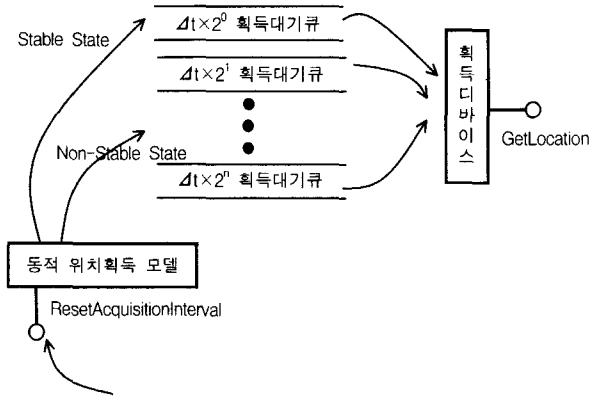
(그림 14) 예측기반 위치획득 모델의 구동 과정 및 획득 시간 간격 재설정 알고리즘

①에서 예측기반 위치획득 모델에서는 2개의 acquiresTi

mer 인스턴스를 이용한다. $\Delta t \times 2$ 시간 간격을 가지는 acquireTimers[1]는 예측을 수행 한 이동체를 포함하게 된다. 즉, 예측을 수행 이동체에 대해서는 다음 시간 Δt에는 실제 획득을 수행하지 않고, $\Delta t \times 2$ 시간에 실제 위치를 획득하게 된다. ④-3에서 획득 효율 정도의 계산은 모든 이동체의 실제 획득한 시간(time)과 미리 설정된 획득 시간 간격(Δt)를 비교하여 측정하게 된다. 즉, 설정된 시간 간격(Δt)에 획득하여야만 하는 모든 이동체의 위치를 획득 한 경우에는 100%이며, 그렇지 못한 경우는 시스템 리소스 부족 또는 통신의 부하의 증가를 원인으로 판단할 수 있다. ⑤에서 이동체의 예측 정확도는 최근에 획득한 이동체의 벡터정보를 이용하여 예측 이동거리, 예측 이동방향과 실제 이동거리, 실제 이동방향의 오차정도를 계산하여 설정한다. ⑥에서 획득 효율이 떨어지거나(th1 이하), 예측 정확도가 보장(th2 이상)되는 상황에서는 이동체의 다음 위치를 예측한 후, acquireTimers[1].Queue에 삽입하게 된다.

3.6 동적 위치획득 모델

동적 위치획득 모델은 초기획득 단계에서는 3.1의 정적 위치획득 모델과 동일하게 동작한다. 시간이 지날수록, 시스템의 부하가 심각해질 경우, 예를 들어 다른 프로세스가 시스템 리소스를 점유한다든지, 갑작스런 통신부하가 발생하는 경우에 이동체 집단을 여러 군(group)으로 나누어서 여러 개의 획득 시간 간격(Δt)을 설정하여 시스템이 안정적으로 유지될 때까지 위치를 획득하는 모델이다. (그림 15)에서와 같이 시스템이 안정적인 상태인 경우는 초기 획득 시간 간격인 Δt 획득 대기 큐를 통해서 획득을 하게 되고, 그렇지 않은 경우에는 여러 개의 획득 시간 간격을 가지는 대기 큐를 두어서 이동체를 분산시켜서 획득을 수행하게 된다.



(그림 15) 동적 위치획득

시스템의 상태가 안정적인지 그렇지 않은지의 판단은 획득 효율을 측정하여 판단한다. 획득 효율은 이동체 별로 획득 시간 간격(Δt)을 파악하여, 획득 지연 없이 정확한 시간에 획득을 수행하는지의 여부를 기준으로 하여 판단한다.

위치획득 모델 구동 과정
allowanceCount : 여유시간을 계산하여 해당 acquireTimers가 추가로 더 수용할 수 있는 이동체의 개수 *Unstable : 이동체가 획득 시간 Δt가 지난 후 획득을 수행 한 상태*
① acquireTimers [index].Callback ( ) ①-1 call moDevice.Acquire (mo) for all mo in acquireTimers [index].Queue ①-2 calculate allowanceCount using total time of acquisition of all mo in acquireTimers [index].Queue and acquireTimers [index].Δt ② moDevice.Acquire(mo) ②-1 update mo.x, mo.y, mo.time from gateway server ②-2 set mo.State using mo.Δt - mo.time, if < 0, Unstalbe else Stable ②-3 acquisitionModel.ResetAcquireTime (mo)
동적 위치획득 모델 시간 간격 재설정 알고리즘

③ dynamicAcquisitionModel.ResetAcquireTime (mo)
③-1 if mo.State = Unstable then
increase mo.Δt, increased
$\Delta t \in$ acquisitionIntervalSet
index = GetTimerIndex (mo.Δt)
acquireTimers [index].Queue.Enqueue (mo)
③-2 else
index = GetTimerIndex (mo.Δt)
if acquireTimers [index-1].allowanceCount > 0 then
mo.Δt = acquireTimers [index-1].Δt
acquireTimers [index-1].Queue.Enqueue (mo)
acquireTimers [index-1].allowanceCount--
else
acquireTimers [index].Queue.Enqueue (mo)

(그림 16) 동적 위치획득 모델의 구동 과정 및 획득 시간 간격 재설정 알고리즘

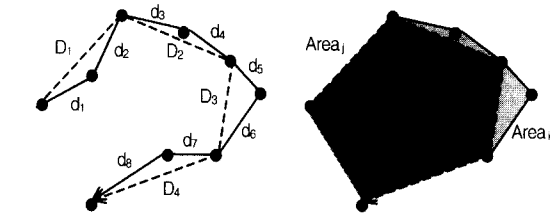
①-2에서 기존의 다른 모델과는 다르게, 해당 acquireTimer가 추가로 수용할 수 있는 allowanceCount를 계산한다. 계산 방법은 해당 acquireTimer.Queue의 모든 이동체의 위치정보를 획득하는데 소요되는 시간과 acquireTimer.Δt를 비교하여, 여유 시간이 있을 경우에는 allowanceCount > 0이게 된다. ②-2에서는 실제 위치를 획득한 시간과 이동체의 획득 시간을 비교하여 제 시간에 획득 하지 못한 경우는 시스템 리소스의 부족 또는 통신부하가 증가하여 획득 효율이 떨어짐을 판단하여 이동체의 상태를 Unstable로 설정하게 된다. 따라서 ③-1에서 이동체의 상태를 파악하여 Unstable인 경우에는 획득 시간 간격을 늘이고 다음 acquireTimer의 Queue에 삽입하게 된다. 반대로 Stable한 상태이면서 상위(이동체의 Δt 보다 적은 Δt를 가진) acquireTimer의 alloanceCount를 비교하여, > 0인 경우 해당 acquireTimer의 Queue에 삽입하게 된다.

3.7 위치획득 벤치마킹 모델

이동체 데이터베이스 시스템의 위치획득 컴포넌트에서 제공되는 위치획득 모델의 성능을 평가하기 위하여 평가 기준을 크게 두 가지로 나눈다.

- 통신전송 감소율, 획득 효율
- 획득된 위치정보의 신뢰도

통신전송 감소율의 측정은 단위시간에 위치획득 회수를 비교하여 측정할 수 있다. 즉, Δt = 1(최소 값)인 정적 위치획득의 통신전송율을 100%로 볼 경우, 다른 위치획득 모델의 상대적인 위치획득 회수를 측정함으로써 가능하다. 또한 획득 효율은 획득 시간 내에 실제로 획득을 수행할 수 있는지 없는지를 측정하여 판단할 수 있다. 획득 시간을 벗어난다면, 어느 정도의 시간 오차에 획득하는지를 판단하여 획득 효율을 측정할 수 있다. 획득한 위치정보의 신뢰도의 측정을 위해서 본 논문에서는 이동거리의 변화율, 이동면적의 변화율을 측정하여 비교한다. 즉, Δt = 1인 정적 위치획득 모델의 이동거리 및 이동면적의 변화율과 다른 위치획득 모델의 상대적인 값을 비교함으로써 측정을 할 수가 있다. 다른 획득 모델들의 이동거리 및 이동면적 변화율이 Δt = 1인 정적 위치획득 모델의 이동거리 및 이동면적 변화율에 가까울수록 신뢰도는 높다고 판단할 수가 있다. (그림 17)과 같이 두 모델간의 신뢰도를 측정하는 함수를 만든 후, 성능을 측정할 수가 있다.



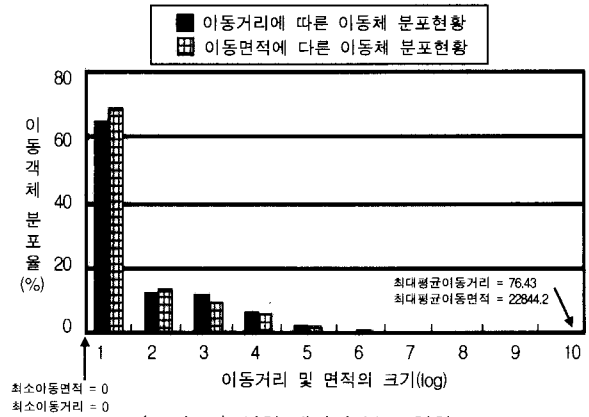
(그림 17) 획득 정보의 신뢰도 측정

4. 성능 평가

이번 장에서는 위치정보 제공 시뮬레이션 환경에서 이동체 위치획득 모델을 구현하여 성능을 평가한다. 본 논문에서는 이동통신망과 LBS 플랫폼의 이동체데이터베이스 사이에 위치정보를 획득하는 과정에서 발생하는 통신부하를 최소화하고, 이동체 데이터베이스의 안정적인 구동을 보장하는 획득 모델에 대하여 연구하였다. 본 논문에서 제시하고 있는 획득 모델은 성격상 크게 두 분류로 나눌 수가 있다. 정적, 거리기반, 개별 거리기반, 그룹기반 위치획득 모델은 시스템의 부하(통신부하, 리소스의 부족 등)가 심할 경우에는 자체적으로 시스템을 안정화시키지 못하고, 각각의 모델의 주요 파라미터들을 조정하여야 한다. 반면에 예측기반, 동적 위치획득 모델은 이러한 상황에서 자체적인 대처능력을 가지고 시스템을 안정화시킨다. 따라서 실험은 이러한 점을 고려하여 수행한다. 구현한 모델의 성능을 실험하기 위하여 이동통신망에서의 실제 데이터를 이용하는 것은 거의 불가능한 현실이다. 따라서 본 논문에서는 위치정보제공 시뮬레이터인 CitySimulator[23]로부터 데이터를 생성하여 실험을 수행한다. 시뮬레이터의 데이터는 100,000개의 이동객체를 대상으로 하였으며, 위치보고는 무한정이다. 테스트 환경은 펜티엄IV-2GHz, 1Gbytes 메모리에 Windows XP에서 Visual.NET에서 구현하여 실험하였다. 실험 항목은 다음과 같다.

- 정적, 거리기반, 개별 거리기반, 그룹기반 통신전송 감소율 비교
- 정적, 거리기반, 개별 거리기반, 그룹기반 통신전송 신뢰도(이동거리, 이동면적 오차율) 비교
- 예측기반 위치획득 모델의 획득 효율 비교
- 동적 위치획득 모델의 획득 효율 비교 및 이동체의 시간 별 대기 큐 분포율

(그림 18)은 실험 데이터의 분포 현황이다. 그림에서 보는 바와 같이 10만개 이동체의 1000번의 위치획득 후 평균 이동거리 및 평균 이동면적을 측정하였다. 약 60~70%의 이동체가 전체적으로 움직임이 적은 경우의 데이터이다. 즉, 사람과 같이 이동양이 적은 집단에 속하는 데이터이다.

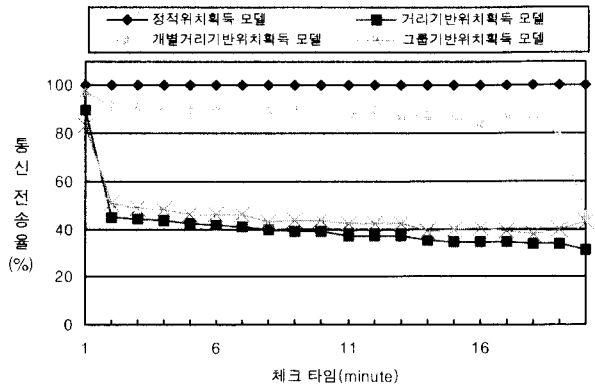


(그림 18) 실험 데이터 분포 현황

실험환경에서 10만개의 데이터를 획득함에 있어서, 획득 효율 100%인 경우는 대략적으로 획득 시간 간격을 5초로 설정하였을 경우이다. 즉, 10만개의 위치정보를 획득디바이스를 통해서 가져오는데 소요되는 시간이 5초이다. 정적 획득 모델인 경우 Δt = 5로 설정하여야만 모든 위치 데이터를 획득할 수가 있다.

(그림 19)는 4가지 획득 모델의 통신전송율을 측정한 그래프이다. 위치획득 모델의 주요한 파라미터의 설정 값들은 아래와 같다.

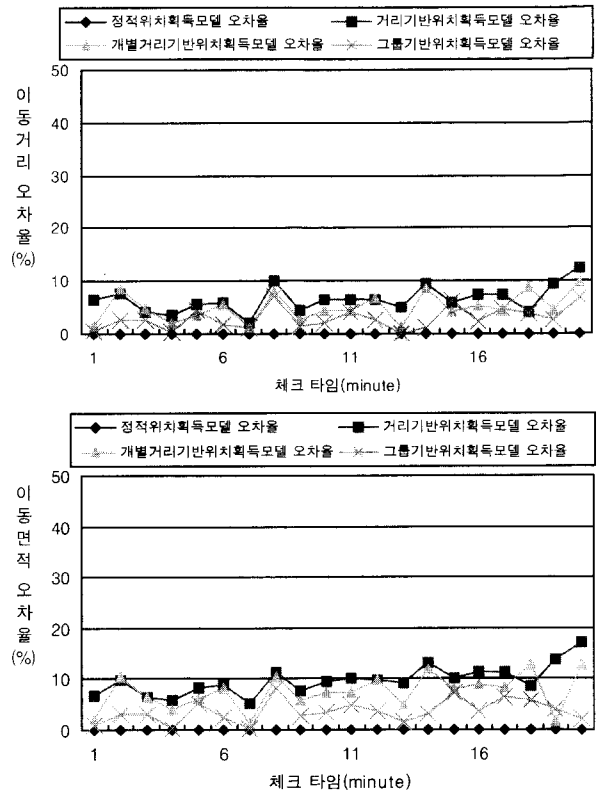
- 초기 획득 시간 간격 Δt = 5
- 평균 이동거리 및 MBR의 구성을 위한 초기 획득 회수 n = 5
- acquisitionInterrvalSet = {$\Delta t \times 1$, $\Delta t \times 2$, $\Delta t \times 3$, $\Delta t \times 4$, $\Delta t \times 5$}
- MBR 증가 rate r = 0.2



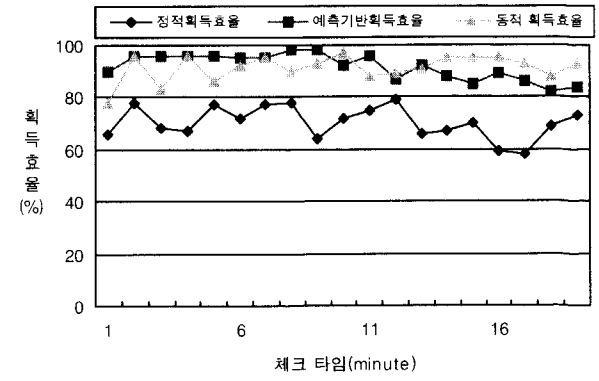
(그림 19) 4가지 위치획득 모델 통신전송율 측정 그래프

정적 위치획득 모델의 경우는 100%의 통신전송율을 나타내고 있다. 처음 1분 동안은 거리기반, 개별 거리기반, 그룹기반 위치획득 모델의 평균 이동거리, MBR을 구성하기 위하여 통신전송율 90%를 유지하고 있으며, 이 시간 이후에는 각각의 모델의 알고리즘이 적용되어 통신전송율이 감소하게 된다. 개별 거리기반 위치획득 모델의 경우, 각각의 이동체가 기준 거리 구간 [lb, ub]를 가지고 있으며, 대부분의 이동체가 초기에 설정한 기준 거리 구간에 대부분 머물러 있는 상태 또는 Low State, High State에 머물어 있기 때문에 통신전송율의 감소 효과를 보지 못하고 있다. 하지만 마지막 15분 이후에서는 개별 거리기반 획득 모델의 알고리즘 중에서 이동체의 기준 거리 구간 [lb, ub]의 값이 Low State, High State에 머물러 있는 경우, 구간 [lb, ub]의 값을 적절히 조절하는 부분이 적용되어 통신전송 감소의 효과를 보이는 부분이다. 거리기반 위치획득 모델의 경우에는 전체 이동체의 평균 이동거리를 이용하여 구간 [lb, ub]를 설정하기 때문에, 각각의 이동체의 획득 시간 간격 조절의 빈번한 발생으로 인하여 통신전송 감소의 효과를 보이고 있다. 즉, 실험 환경 데이터의 60~70%가 이동양이 적은 데이터이기 때문에, 상대적으로 평균 이동거리가 적으며, 30~40%의 이동양이 큰 이동체의 경우에는 구간 [lb, ub]보다 큰 영역에 존재하기 때문에 획득 시간 간격이 줄어드는 대신, 반대로 나머지 이동체의 획득 시간 간격은 늘어나기 때문에 전체적인 통신전송율의 감소를 보이는 것이다. 그룹기반 위치획득 모델의 경우에도, 대부분의 이동체가 움직임이 작기 때문에, 초기에 MBR에 포함되어 머물러 있는 이동체가 많으므로 획득 시간 간격이 늘어나서 통신전송율의 감소를 보이는 것이다. 실제 데이터 군의 종류에 따라 결과가 다를 수 있지만, 상황에 맞게끔, 획득 모델의 주요 파라미터들을 적절히 조절하여 통신전송율을 감소시킬 수 있다.

(그림 20)은 획득 효율 100%인 정적 획득 모델을 기준으로 하여 거리기반, 개별 거리기반, 그룹기반 획득 모델의 이동거리 오차율 및 이동 면적 오차율을 측정한 그래프이다. 그림에서 보는 바와 같이 그룹기반 위치획득 모델의 오차율이 가장 낮으며, 모든 획득 모델의 오차율이 10% 이하이다. 즉, 신뢰도의 오차율이 10% 이하인 것이다. (그림 19)에서 통신전송 감소의 효과가 큰 거리기반, 그룹기반 위치획득 모델의 경우에 그만큼의 오차율이 클 것이라고 예상할 수가 있다. 하지만, (그림 20)의 그래프에서 보는 바와 같이 오차율은 3개의 모델에 대해서 비슷하게 나타나고 있다. 즉, 위치정보의 획득 회수를 줄였음에도 불구하고 오차율이 적다는 것은, 그만큼 위치획득 모델의 성능, 효율적인 면에서 효과를 보이는 것이다.



(그림 20) 이동거리 및 이동면적 오차율 측정 그래프



(그림 21) 예측기반, 정적, 동적 획득 효율 측정

(그림 21)는 예측기반, 동적 위치획득 모델의 획득 효율을 나타낸 것이다. 획득 효율은, 획득 시간 간격 Δt = 1이고, 이

동체의 개수가 100인 경우, Δt 동안 100개의 이동체의 위치를 획득 하였을 경우 100%이다. 실험 데이터의 경우 획득 시간 간격을 5초로 설정한 경우 10만개 이동체의 위치를 모두 획득할 수 있다. 이 경우 획득 효율은 100%이다. 하지만, 획득 시간 간격을 5초 이내로 설정하거나, 또는 시스템의 리소스가 부족할 경우에는 효율이 낮아진다. 이 경우에, 예측기반, 동적 위치획득 모델은 획득 효율을 높여서 안정적인 시스템의 상태를 유지하도록 동작한다.

예측기반, 동적 위치획득 모델의 주요 파라미터 설정 값은 아래와 같다.

- 초기 획득 시간 간격 $\Delta t = 3$
- 예측 조건 : 획득 효율 < 85%, 예측 정확도 > 30%
- 동적 획득 모델에서의 대기 큐 개수 = 3,
  획득 시간 간격 = {Δt, $\Delta t \times 2$, $\Delta t \times 3$}

(그림 21)에서, 정적 획득 모델의 경우 효율이 60~80%임을 나타내고 있으며, 예측기반, 동적 획득 효율은 80~100%를 나타내고 있다. 예측기반 획득 모델은 3.5에서 설명한 바와 같이, 획득 정보의 신뢰도를 보장하기 위하여, 한번 예측한 이동체에 대해서는 반드시 다음 번에 위치를 획득하여야 한다. 즉, 최대 예측 가능율은 50%가 된다. 그리고, 모든 이동체에 대해서 예측을 수행하는 것이 아니고, 예측 정확도를 각각의 이동체 별로 측정하여 일정 threshold 이상인 이동체에 대해서만 예측을 수행한다. 그래프에서 획득 효율이 높으면 높을수록 시스템은 안정적으로 동작을 하며, 예측기반, 동적 획득 효율의 그래프가 낮은 시점은 획득 효율이 떨어지는 시점이고, 다시 높이 올라가는 시점은, 예측을 수행하였거나, 동적으로 획득 시간 간격을 조절하는 시점이다. 그리고, 획득 효율이 안정적이라고 판단되었을 경우에 예측 수행율이 떨어지며, 이동체가 낮은 시간 간격의 획득 대기 큐로 이동하기 때문에 위의 그래프는 계속해서 아래위로 높낮이가 계속 반복을 하게 된다.

5. 결론 및 향후 연구 과제

향후 위치기반 서비스의 핵심 부분인 LBS 플랫폼 중에서 이동체를 관리하는 이동체 데이터베이스 시스템은 이동체의 개수가 증가할수록 시스템 전체 성능의 저하를 초래하게 되며, 또한 과거 위치 이력 데이터를 관리하기 위해서도 시스템의 성능은 중요한 요소가 된다. 대용량의 이동체를 획득함에 있어서, 이동체 데이터베이스와 위치정보를 제공하는 이동통신망 사이의 심각한 통신부하는 쉽게 예측할 수 있으며, 전체 시스템의 성능을 보장하기 위해서 통신부하를 최소화시킬 필요가 있다. 이에 본 논문에서는 이러한 통신부하를 최소화시키기 위한 6가지의 위치획득 모델에 대해서 연구해 보았다. 6가지의 위치획득 모델은 이동체의 이동 패턴에 따라서 획득 회수를 줄임으로써 통신부하를 최소화하거나, 또는 시스템의 안정적인 구동을 위하여 획득 시간 간격을 동적으로 조절 또는 예측하는 기능들을 포함하고 있다. 향후에는 이러한 위치획득 모델을 실제 이동통신망의 GMLC, MPC등의 게이트웨이 서버에서 제공되는 위치 데이터를 이용하여 정확한 성능 분석이 필요하며, 또한 이동체 데이터베이스와 이동통신망과 어느 정도 분리되어서 구동되는 것이 아닌, 실제 이동통신망의 터미널 페이징 오버헤드와 이동체 데이터베이스에서의 저장 오버헤드를 직접적으로 최소화할 수 있는 방향으로 연구가 확장될 필요가 있다.

참 고 문 헌

[1] A. Bar-Noy, I. Kessler and M. Sidi, "Mobile users : To update or not to update?," Wireless Networks, Vol.1, No.2, pp.187-196, 1995.

[2] I. F. Akyildiz and J. S. M. Ho, "Dynamic mobile user location update for wireless PCS networks," Wireless Networks, Vol.1, No.2, pp.187-196, 1995.

[3] J. S. M. Ho and I. F. Akyildiz, "Mobile user location update and paging under delay constraints," Wireless Networks, Vol.1, No.4, pp.413-425, 1995.

[4] A. Abutaleb and V. O. K. Li, "Location update optimization in personal communication systems," Wireless Networks, Vol.3, No.3, pp.205-216, 1997.

[5] I. F. Akyildiz, S. M. Ho and Y. B. Lin, "Movement-based location update and selective paging for PCS networks," IEEE/ACM Trans. Networking, Vol.4, No.4, pp.629-638, Aug., 1996.

[6] Ouri Wolfson, Bo Xu, Sam Chamberlain, Liqin Jiang, "Moving Objects Databases : Issues and Solutions," Proc. of the 10th Int. Conf. on Scientific and Statistical Database Management (SSDBM '98), Capri, Italy, pp.111-122, July, 1998.

[7] Jensen, C. S. Jensen, A. Friis-Christensen, T. B. Pedersen, D. Pfoser, S. Saltenis and N. Tryfona, "Location-Based Services A Database Perspective," Proceedings of the Eighth Scandinavian Research Conference on Geographical Information Science, As, Norway, pp.59-68, June, 2001.

[8] Jean Marc saglio, Jose Moreira, "Oporto : A Realistic Scenario Generator for Moving Objects," Geoinformatica, pp. 71-93, 2001.

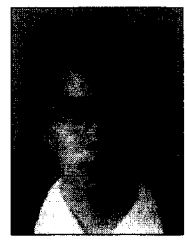
[9] Ming Hour Yang, Lien Wu Chen, Yu Chee Tseng, "A Traveling Salesman Mobility Model and Its Location Tracking in PCS Networks," International Conference on Distributed Computing Systems, pp.517-523, 2001.

[10] Ralf H. Guting, Mike H. Bohlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos, Markus Schneider, Michalis Vazirgiannis, "A Foundation for Representing and Query-

ing Moving Objects," ICDE, pp.422-432, 1997.
[11] Zhexuan song, Nick Roussopoulos, "Hashing Moving Objects," International Conference on Mobile Data Management, pp.161-172, 2001.
[12] 류근호, 안윤애, 이준욱, 이용준, "이동 객체 데이터베이스와 위치기반 서비스의 적용", 데이터베이스연구, 제17권 제3호, 2001.
[13] 안윤애, 김동호, 류근호, "차량 위치 추적을 위한 이동 객체 관리 시스템의 설계", 정보처리논문지, 제9권 제5호, 2002.
[14] Rui Jos, Nigel Davies : Scalable and Flexible Location-Based Services for Ubiquitous Information Access, HUC pp.52-66, 1999(DBLP : conf/huc/JoseD99).
[15] Dieter Pfoser, Christian S. Jensen, "Capturing the Uncertainty of Moving-Object Representations," SSD, pp.111-132, 1999.
[16] Ouri Wolfson, S. Chamberlain, S. Dao, L. Jiang, G. Mendez "Cost and Imprecision in Modeling the Position of Moving Objects," Proceedings of the Fourteenth International Conference on Data Engineering (ICDE14), 1998.
[17] A. Prasad Sistla, Ouri Wolfson, Sam Chamberlain, Son Dao, "Modeling and Querying Moving Objects," ICDE, pp.422-432, 1997.
[18] Ouri Wolfson, Sam Chamberlain, Son Dao, Liqin Jiang, "Location Management in Moving Objects Databases," WOS BIS '97, pp.7-12, 1997.
[19] P. Sistla, O. Wolfson, S. Chamberlain, S. Dao, "Querying the Uncertain Position of Moving Objects," Springer Verlag Lecture Notes in Computer Science number 1399, 1998.
[20] Ouri Wolfson, Liqin Jiang, A. Prasad Sistla, Sam Chamberlain, Naphtali Rishe, and Minglin Deng, "Databases for Tracking Mobile Unitis in Real Time," ICDT, pp.169-186, 1999.
[21] OpenLS Initiative, A Request for Technology In Support of an Open Location Services (OpenLS TM) Testbed, http://www.openls.org, 2000.
[22] ISO TC/211, 19132 Geographic Information-Location based services possible standards, http://www.isotc211.org/scope.htm #19132.
[23] LIF (Location Inter-operability Forum), Statement Version 4, LIF.
[24] http://www.alphaworks.ibm.com/tech/citysimulator.
[25] 전화숙, 정동근, "차세대 이동통신망을 위한 사용자 이동패턴에 근거한 페이징 기법의 설계", 정보과학회논문지 : 정보통신, 제29권 제3호, pp.216-223, 2002.
[26] L. Forlizzi, R. H. Guting, E. Nardelli and M. Schneider, "A Data Model and Data Structures for Moving Objects Databases," In Proc. of the ACM SIGMOD Conf., pp.319-330, 2000.
[27] 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.
[28] D. Pfoser, Y. Theodoridis and C. S. Jensen, "Indexing Trajectories of Moving Point Objects," Chorochronos Technical Report, CH-99-3, October, 1999.
[29] D. Pfoser, C. S. Jensen and Y. Theodoridis, "Novel Approaches in Query Processing for Moving Objects," In Proc. of the VLDB Conference, pp.395-406, 2000.
[30] S. Saltenis, C. S. Jensen, S. Leutenegger and M. Lopez, "Indexing the Positions of Continuously Moving Objects," In Proc. of the ACM SIGMOD International Conference on the Management of Data, pp.331-342, 2000.
[31] Yannis Theodoridis, Jefferson R. O. Silva, Mario A. Nascimento, "On the Generation of Spatiotemporal Datasets," CHOROCHRONOS Technical Report CH-99-01.
[32] Dietoer Pfoser, Yannis Theodoridis, "Generating Semantics-Based Trajectories of Moving Objects," International Workshop on Emerging Technologies for Geo-Based Applications, Ascona, Switzerland, 2000.

민 경 욱

e-mail : kwmin92@etri.re.kr
1996년 부산대학교 전자계산학과(학사)
1998년 부산대학교 전자계산학과(이학석사)
2001년~현재 한국전자통신연구원 LBS 연구팀 연구원
관심분야 : 공간 데이터베이스, 이동체 데이터베이스, LBS, GIS 등



조 대 수

e-mail : junest@etri.re.kr
1995년 부산대학교 컴퓨터공학과(학사)
1997년 부산대학교 컴퓨터공학과(공학석사)
2001년 부산대학교 컴퓨터공학과(공학박사)
2001년~현재 한국전자통신연구원 LBS 연구팀 선임 연구원
관심분야 : 공간 데이터베이스, 이동체 데이터베이스, LBS, GIS 등