이동 객체 궤적에 대한 최근접 질의

최보윤°, 지정희, 김상호, 류근호
충북대학교 데이터베이스 연구실
{bychoi°, jhchi, shkim, khryu}@dblab.cbu.ac.kr

The Nearest Neighbor Query for Trajectory of Moving Objects

Bo Yoon Choi°, Jeong Hee Chi, Sang Ho Kim, Keun Ho Ryu
Database Laboratory of Chungbuk National University

요 약

이동 객체에 대한 기존 최근접(nearest neighbor, NN) 질의 처리 기법들은 질의 궤적에 대해 연속적으로 정확하게, 질의와 가장 가까운 위치를 유지하면서 움직이는 최근접 객체를 선택할 수 있는 충분한 기준을 가지고 있지 못하다. 이 논문은 질의 객체와 데이터 객체가 모두 이동 객체인 경우에 가장 적합하게 사용되는, 객체 궤적에 대한 연속적인 질의 처리를 통해 정확한 결과를 얻을 수 있는 새로운 최근접 질의 처리 기법, 연속 궤적 최근접 질의(CTNN, continuous trajectory nearest neighbor query)를 제안한다. 우리는 두 가지 Approximate, Exact CTNN 기법을 제안하며, 이들은 모두 항해 시스템, 교통 통제 시스템, 물류정보 시스템 등 각종 위치 기반 서비스(LBS: location based services) 상에서 다양하게 사용될 수 있다. 이들은 이동 객체 궤적이 미리 알려져 있는 경우, 그리고 질의와 데이터 객체가 모두 이동 객체인 경우에 가장 적합하다.

1. 서 론

많은 위치기반서비스들은 실세계 모든 객체에 대해 다양한 질의 처리 기법들을 사용하여 사용자에게 원하는 정보를 제공하고자 한다. 이 때 자주 사용되는 기법 중 하나인 최근접 질의는 사용자의 선택 위치와 가장 가까운 곳에 존재하는 객체를 결과로 반환하며, 위치 기반 서비스 내에서 다음의 예제와 같이 적용될 수 있다: A씨는 집에서 핸드폰으로 가장 가까운 곳에 위치한 피자집을 검색하여 피자를 주문한다.(정적-정적) 운전 중인 B씨는 자신의 차에 장착된 네비게이터(navigator)를 통해서 가장 가까운 곳에 위치한 주유소를 검색한다.(동적-정적) 항해 중인 선박 C는 근처에 항해하고 있는 주유선을 검색하여 급유를 요청하고자 한다.(동적-동적) 고속버스 운전기사 D씨는 고속도로 위에서 사고를 겪었다. D씨는 가장 먼저 자신의 위치에 도착 가능한 버스를 검색하여 도움을 요청하였고, 승객들은 모두 다른 버스를 이용하여 무사히 목적지에 도달할 수 있었다.(정적-동적) 이 예제들에서 괄호 안의 요소는 질의-데이터 객체 타입을 나타내며, 정적 객체는 2차원 공간상의 고정된 공간 객체를, 동적 객체는 3차원 공간상의 이동 객체를 가리킨다.

이와 같이 위치기반서비스들은 사용자가 원하는 정보를 제공하기 위해서 여러 타입의 데이터 객체들에 대해 질의를 처리하게 되며, 이와 관련하여 여러 최근접 지질의 처리 기법들이 제안되었다. 이동 객체들은 시간의 흐름에 따라 연속적으로 자신의 위치를 변경하며 움직이기 때문에 어떤 한 시점에서 계산된 결과는 이후 시간에서 다른 결과로 변경될 수 있다. 하지만 대부분의 시스템들은 이동 객체 궤적의 시작 시점에서만 객체간 거리를 비교하여 결과를 선택하거나, 궤적 상의 일정 간격 시간점 위에서의 거리 계산을 통해서 결과를 반환하기 때문에 전체 질의 시간에 대해 항상 유효한 정보를 제공할 수 없다. 이러한 문제를 해결하기 위해서 연속 또는 영속 질의 처리 개념[1]을 도입한 기법이 제안되기도 하였지만, 이들 역시 객체의 궤적 정보 즉, 객체가 서로 가까워지는지 멀어지는지, 어느 정도의 기울기로 움직이고 있는지 등을 고려하지 않기 때문에 정확한 최근접 객체 탐색이 어려울 뿐만 아니라 선택된 결과가 빠른 시간 안에 다른 객체로 변경될 수 있다는 단점을 가진다. 특히, 하나의 최근접 객체를 선택해야 하는 질의에서 여러 후보 객체를 가지는 경우, 이 중 하나를 결과로 선택할 수 있는 기준이 없기 때문에 k-최근접 질의만이 가능하다.

따라서 이 논문에서는 기존 연구들이 풀 수 없었던 문제점들을 해결함과 동시에 객체 간 궤적 정보를 비교하면서 연속적으로 질의와 가장 가까운 거리를 유지하며 움직이는 객체를 찾는 것이 가능한 새로운 최근접 질의 처리 기법을 제안하며, 우리는 이를 연속 궤적 최근접 (CTNN) 질의라 부른다. CTNN 질의는 질의 객체와 데이터 객체들의 이동 방향 정보, 기울기 정보 등 객체 궤적 정보 비교를 통해서 질의와 가장 가까운 위치를 유지하는 객체를 정확히 선택하며, 여러 후보 객체가 있을 때 k값(1또는 n)에 따라 정확히 결과를 선택할 수 있는 기준을 제공하고, 질의 객체의 궤적 상에서 최근접 객체 정보가 변경되는 시점을 연속적으로 지정한 후, 그 시간점 위에서 결과

† 이 논문은 University IT Research Center 프로젝트와 KOSEF RRC 프로젝트(청주대학교 ICRC)의 지원으로 연구되었음

가 변경되는지 여부를 체크함으로써 질의 전체에 대해 항상 유효한 결과를 얻을 수 있도록 한다. 최근접 객체 정보가 변경되는 시점을 우리는 평가시간이라 부르며, 평가시간은 각 세그먼트의 양 끝점, 즉 객체 변곡점과 세그먼트 간의 교차점으로 구성된다.

이 논문에서는 결과 정확도에 어느 정도의 오차를 허용하면서 빠른 응답시간을 보이는 Approximate CTNN 기법과 어느 정도의 응답 시간을 보장하면서 정확히 최근접 객체를 탐색해 내는 Exact CTNN 기법을 제안한다. 논문의 나머지는 다음과 같이 구성되어 있다. 2절에서는 기존 연구들을 소개하면서 이들의 문제점을 살펴보고, 3절에서는 이 논문에 기본 모델과 가정들을 소개하며, 4절에서는 제안된 CTNN 기법들에 대한 질의 처리 과정과 알고리즘을 소개하고, 5절에서는 성능 평가를 보인다. 그리고 6절에서 결론 및 향후 연구를 제시한다.

2. 관련 연구

최근접 질의 처리 기법들은 멀티미디어 데이터베이스, 데이터 마이닝, 공간·시공간·이동체 데이터베이스 등 광범위한 분야에서 연구되고 있다. 이 절에서는 질의 객체와 데이터 객체가 모두 이동 객체인 경우에 적용 가능한 최근접 질의 처리 기법들을 중점적으로 살펴본다.

Benetis[2]는 TPR-tree[3]를 사용하여 데이터를 인덱스하고, DF 탐색 방법[4]에 따라 미분 함수를 사용하여 각 노드 사각형과 질의점 사이의 거리를 계산한 후, 가장 가까운 거리를 가지는 객체를 인터벌 [n, n+Δ] 단위로 찾아 결과로 반환하는(이 때, n은 now 시간이다), 영속 최근접 질의 처리 기법을 제안하였다. 이 기법은 단순 탐색을 행하기 때문에 많은 시간점 위에서 한 데이터에 대해 계속 중복되는 거리 계산을 행하게 되고, 이전에 탐색된 최근접 객체가 현재에도 결과로서 유지되는지 아닌지를 비교하는 작업들을 지속적으로 요구한다. 또한 거리 계산 함수는 객체의 위치나 속도에 의존적이기 때문에 이 값들이 변경될 때마다 함수가 재 계산되므로 큰 오버헤드를 초래할 수 있다. 또한 하위 인터벌의 Δ값 범위에 따라 결과 정확도와 계산 오버헤드 사이의 trade-off를 가지게 된다.

Tao[5]는 연속 최근접 질의 처리 기법인 TPNN(time-parameterized NN) 질의와 CNN(continuous NN) 질의를 제안하였다. 이들은 최근접 객체를 선택하면서 전체 질의 시간 선분 위에서 최근접 객체 정보가 변경될지도 모르는(TPNN) 또는 변경되는(CNN) 시간점을 계산하고, 각각의 최근접 객체 정보와 이 정보의 유효시간 인터벌을 시간 리스트 TL에 저장하였다가 질의 마지막 시간에 결과로 반환한다. 즉, 결과는 TL={<a, [t_1, t_2]>, <b, [t_2, t_3]>} 과 같이 여러 개의 연속적이고 분리된 하위 시간인터벌과 각 인터벌 내에서 유효한 최근접 객체 정보로 구성된다. TPNN 질의는 모든 객체에 대해 질의와 가장 가까워지는 가까운 미래 시간을 계산하고, 계산된 시간 값에 따라 순서대로 실제 최근접 객체가 되는지 여부를 판단한 후, 유효한 객체를 결과로 선택한다. TPNN 질의의 계산적 오버헤드를 줄이기 위하여 제안된 것이 CNN 질의이며, 이는 질의 시작 시간에서 객체들과 질의 q 사이의 거리를 계산한 후, 거리를 오름차순으로 정렬하고, 첫 번째 최근접 객체 p를 선택한다. 그리고 다른 객체가 p와 q 사이에 생성된 근접원(vicinity) 안에 포함되는지 여부를 거리 순서대로 판단하고, 포함된다면 그 객체를 최근접 객체 p'로 선택하면서 유효시간 인터벌을 설정한다. 그리고 p'와 q 사이에 또 근접원을 그리고, 동일 과정을 질의 마지막 시간까지 반복한 후 TL내의 결과를 반환한다. 질의-데이터가 모두 이동 객체인 경우 CNN 질의는 근접구와 수직이등분 평면을 사용하여 최근접 객체와 그 유효시간 인터벌을 결정하게 되는데, 이는 객체 수가 많아질수록 또한 객체 이동 방향 변경이 잦을수록 (궤적 선분 세그먼트 수가 많아진다) 큰 오버헤드를 초래할 수 있다. 또한 이 경우 CNN질의는 k-최근접 질의처리만이 가능하다. 동일한 거리를 가지는 k개의 후보 객체가 모두 근접구 안에 포함될 때, CNN 질의는 하나의 객체를 선택할 수 있는 기준이 없기 때문에 k개의 객체들을 모두 최근접 객체로 선택한다. 하나의 최근접 객체 정보를 원하는 경우 CNN 질의는 임의의 객체를 결과로 선택하게 되며, 이는 부정확한 결과를 보일 수 있다. 또한 선택된 객체가 빠른 시간 안에 다른 객체로 변경된다면 잦은 리스트 갱신을 발생시키게 되는데, 이를 방지할 수 있는 대안을 가지지 못한다.

3. 질의 모델

이 절에서는 CTNN 질의의 기본 데이터 구조와 질의 타입을 설명하며, 이 논문이 포함하고 있는 몇 가지 가정들과 표기들을 소개한다.

3.1 데이터 모델

이 논문은 시간의 흐름에 따라 자신의 위치를 변경하며 연속적으로 움직이는 이동 점 객체에 대한 최근접 질의를 대상으로 한다. 객체의 위치 정보는 GPS 등의 장치를 통하여 주기적으로 추출되고, 객체의 삽입, 삭제, 속도나 이동 방향의 변경 등과 같은 갱신이 발생할 때 데이터베이스에 저장된다. 즉, 이동 객체는 갱신이 발생되는 각 시간 t에서 <id_i, (x, y), t> 형태로 저장된다. 일반적으로 이동 객체의 궤적은 3차원 공간상의 폴리라인(polyline) 즉, 연속적이고 개별적인 분리된 선분 세그먼트들의 집합으로 표현된다. 각각의 선분 세그먼트는 데이터베이스 내에서 연속되어 나타나는 두 개의 데이터 점을 연결한 선분이며, 따라서 한 세그먼트 위에서 객체는 항상 일정한 속도와 동일한 방향을 가진다. 이동 객체 궤적은 크게 자유궤적과 제약궤적으로 구분된다[6]. 우리는 미리 이동 경로를 예측할 수 있는 객체를 '제약궤적을 가진 객체'로 간주하며, 이 논문에서 제안된 CTNN 기법들은 버스, 비행기, 기차 등의 제약 궤적 이동 점 객체에 적용 가능하다.

데이터베이스에 저장되지 않은 시점에서의 공간 좌표는 다음의 시간 종속적 선형 위치 추정 함수를 사용하여 추정될 수 있다. 즉, $t_i < t < t_{i+1}$ 이라면,

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

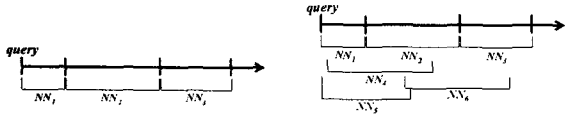
이다. 이 함수는 데이터베이스에 저장된 시간 t_i와 t_{i+1}에서의 좌표 값을 이용하여 그 사이에 있는 어떤 시점 t에서의 객체 위치를 계산하기 때문에, 한 선분 세그먼트의 양 끝점 정보만으로 어떤 시간에서의 객체 공간 점 위치를 추정할 수 있다.

3.2 질의 타입

Porkaew[7]에 따르면 시공간 최근접 질의는 크게 두 가지, 시간 범위 질의와 공간 최근접 질의의 합, 공간 범위 질의와 시간 최근접 질의의 합으로 구분된다. CTNN은 기본적으로 전자의 경우를 따르며, 좀 더 엄밀히 말하자면 최근접 객체가 선택된 한 시점으로부터 결과가 변경되는 가장 가까운 미래 시간을 빠르게 찾아내고 그 시간점에서 공간적으로 가장 가까운 위치를 가지는 객체를 선택하기 때문에 공간 최근접 탐색과 시간 최근접 탐색이 동시에 가능한 시공간 최근접 질의 타입이다.

또한 CTNN 질의는 먼저 전체 질의 시간상에서 최근접 객체가 변경될 시점을 선택하고, 그 시간점에서 실제 변경 여부를 판단하여 결과를 저장하기 때문에 항상 유효하고 연속적인 결과를 얻을 수 있다. 즉, 연속 최근접 질의 타입을 가진다. 이는 TPNN 질의보다 더 적은 계산을 통해 결과 정보 변경 시간점을 선택하게 되고, 먼저 시간점을 선택한 후 변경여부만 가리기 때문에 최근접 객체 선택 과정에서 계속 시간점을 계산해내는 CNN 질의보다 더 빠른 계산 시간을 가진다.

그리고 질의 객체와 데이터 객체들이 서로 멀어지는지, 가까워지는지, 마주보는 방향으로 움직이는지 같은 방향으로 이동하고 있는지 등 객체 궤적 정보를 비교하여 최근접 객체를 선택하기 때문에 질의 궤적과 가까운 위치를 유지하면서 움직이는 객체를 선택할 수 있다. 우리는 이를 궤적 최근접 질의라 부른다. 궤적 최근접 질의를 통해서 질의 궤적과 가까운 위치를 유지하는 객체를 찾는다면, 우리는 탐색된 최근접 객체 정보가 자주 변경되는 것을 방지할 수 있다.



(a) 최근접 질의 (b) k-최근접 질의

그림1. CTNN 질의에서의 결과 유효시간 인터벌

또한 CTNN 질의는 사용자가 몇 개의 최근접 객체를 결과로 원하는지에 상관없이 조건을 만족시킬 수 있다는 장점을 가진다. 즉, 최근접 질의, k-최근접 질의가 모두 가능하며 이 때 k값은 사용자의 지정에 따른 k이거나 결과로 나타날 수 있는 모든 후보 객체 개수가 될 수 있다. 그림 1은 최근접 질의와 k-최근접 질의의 결과 객체 정보와 해당 객체의 유효시간 인터벌 간 차이를 보인다. 최근접 질의의 경우 탐색된 각 최근접 객체는 전체 질의 시간에 대해 개별적이고 분리된, 연속적인 하위 유효시간 인터벌을 가지며, k-최근접 질의는 서로 겹치는 하위 시간 인터벌을 가진다.

3.3 가정 및 표기

CTNN 질의는 다음 가정을 바탕으로 처리된다.

1. 어떤 객체 p가 최근접 객체로 선택된 후 1분 이내에 변경된다면, p는 무시되고 유효시간 인터벌은 이전 또는 이후 인터벌에 포함된다. 이 시간 인터벌은 하나의 결과를 표현하기에는 너무 짧은 시간이기 때문에, 실세계 에서는 무의미하게 사용될 수 있다.
2. 두 객체 사이의 거리가 1m 이내인 경우, 이 객체들은 서로 만나거나 교차한다고 간주한다. 실세계에서 객체들이 같은 시간에 같은 좌표를 가지는 경우는 충돌을 의미하며, 이런 상황은 거의 발생하지 않는다.

또한 이 논문 전반에 걸쳐 다음의 표기들을 사용한다.

표1. 표기

$\square a_i$	객체 a의 i번째 세그먼트 ($i \geq 1$인 정수)
(x_a^t, y_a^t)	시간 t에서 객체 a의 공간 점 좌표
Δx (Δy)	$x(y)$ 축에서 객체가 단위 시간당 움직인 거리
x_{ab} (y_{ab})	객체 a, b의 $x(y)$ 좌표 차이 값
$\lvert a - b \rvert^t$	시간 t에서 객체 점 a와 b사이의 공간 거리 $= \sqrt{(x_{ab}^t)^2 + (y_{ab}^t)^2}$
$/p$	세그먼트 p의 기울기 = Δx / Δy
∂p	변위 값. 객체가 단위 시간당 움직인 거리

4. 연속 궤적 최근접(CTNN) 질의

이 절에서는 CTNN 질의에 사용되는 기본 알고리즘과 계산 과정들을 소개한 후, 이 논문이 제안하는 두 가지 질의처리 기법 Approximate, Exact CTNN의 알고리즘들을 소개한다.

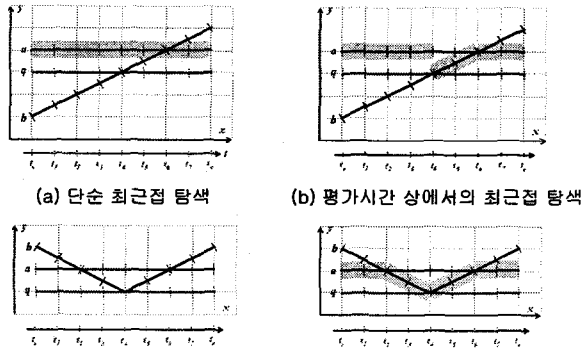
4.1 CTNN 질의 기본 계산 알고리즘

CTNN 질의는 맨 처음 단계로, 주어진 질의 시간 범위 내에 존재하는 모든 살아 있는 데이터 객체들을 검색하고, 질의 시간 인터벌로 이 세그먼트들의 시간 인터벌을 Clipping한 후 최근접 질의를 통해 결과를 반환한다. 최근접 객체를 선택하는 과정은 크게 두 가지 단계, 연속 질의를 위한 평가시간 설정 단계와 궤적 질의를 위한 객체 궤적 정보 비교 단계로 이루어진다. 평가시간이란, 질의 객체의 전체 궤적 선분 상에서 최근접 객체가 획득되는 시간, 또는 최근접 객체 정보가 변경되는 시간으로서, 결과로 반환되는 정보들이 추출되는 시간점이다. 두 개의 연속된 평가시간 점은 하나의 평

가시간 인터벌을 구성하게 되고, 각각의 최근접 객체들은 탐색될 때마다 자신이 유효하게 존재하는 평가시간 인터벌과 함께 시간 리스트 TL에 {<a, [t_1, t_2]>, <b, [t_2, t_3]> 형태로 저장되었다가, 질의 마지막 시간에 결과로 반환된다.

4.1.1 평가시간 설정

평가시간은 질의 객체를 포함한 모든 데이터 세그먼트의 양 끝 점 시간, 즉 객체 갱신이 발생하는 각 변곡점들과, 세그먼트 간의 교차점으로 구성된다. 이 때, CTNN 질의는 다음의 그림 2와 같이, 데이터 세그먼트의 선분 대칭 이동을 활용하여 평가시간을 설정한다.



(a) 단순 최근접 탐색 (b) 평가시간 상에서의 최근접 탐색

(c) 데이터 객체의 선분 대칭이동 (d) 대칭 이동 이후, 최근접 탐색

그림2. CTNN 질의 기본 알고리즘 예제

교차점 검출은 다음의 알고리즘 1과 같이 행해진다.

알고리즘 1. Find_Intersection

Input: 질의 객체와 데이터 객체 세그먼트 집합 *seg*
Output: 교차점 리스트 *IL*
1. 교차점 탐색
   1.1 *seg*를 2차원 공간 상의 선분 세그먼트로 간주
   1.2 **Sweep Line Algorithm** → 교차점 *ip* 추출
2. *ip*의 교차 시간 *t* 설정
   // 객체 A와 B가 각각 *t*A, *t*B 시간에 *ip*에 도달한다고 가정했을 때,
   1.1 *t*A > *t*B 라면 *t* = *t*B
   1.2 이외의 경우 *t* = *t*A
3. *ip* , *t* → *IL*
4. *IL* 반환

4.1.2 변위 계산

궤적 최근접 질의의 기본 목적은 가능한 한 질의와 가장 가까운 거리를 유지하면서 움직이는 객체를 최근접 객체로 선택하는 것이다. 따라서 각 평가시간 점에서 최근접 객체를 선택할 때, k값에 따라 궤적 질의 조건을 만족하는지 판단할 필요가 있다. 이 때, CTNN 질의는 다음의 알고리즘 2와 같이, 변위 계산을 통하여 각각의 후보 객체들이 질의 세그먼트와 만나는 시간을 구한다. 그리고 가장 작은 양수 값을 반환하는 객체, 즉 가장 먼저 질의와 가까워지는 객체 순서대로 k값에 알맞게 선택한다. 이 계산에서 반환되는 결과는 두 객체가 현재를 기준으로 몇 단위 시간 이전 또는 이후에 교차하는지를 나타내는 시간 정수 값이다.

알고리즘 2. Calculate_Displacement

Input: □*A* , □*B*
Output: 정수 T
1. */A* == */B* 이고 ∂*A* == ∂*B*인 경우,
   1.1 $x_{ab} == y_{ab} == 0$이면 T=1 (일치)
   1.2 이외의 경우 T=0 (평행)
2. |*/A* | == 1/|*B*|인 경우,
   2.1 */A*의 부호 == */B*의 부호일 때,
      2.1.1 $x_{ab} == y_{ab}$이면 $T = x_{ab} / ||\Delta y| - |\Delta x||$
      2.1.2 이외의 경우 T=0
   2.2 */A*의 부호 != */B*의 부호일 때,
      2.2.1 $(|\Delta y| + |\Delta x|) / ||\Delta y| - |\Delta x|| == x_{ab} / y_{ab}$이면
         $T = x_{ab} / (|\Delta y| + |\Delta x|)$
      2.2.2 이외의 경우 T=0
3. */A* == - */B*인 경우,
   3.1 $x_{ab} == 0$이고 y_{ab}!=0이면 $T = y_{ab} / (|\Delta y| + |\Delta x|)$
   3.2 $y_{ab} == 0$이고 x_{ab}!=0이면 $T = x_{ab} / (|\Delta y| + |\Delta x|)$
   3.3 이외의 경우 T=0
4. 이외의 모든 경우, $X = x_{ab} / |\partial A - \partial B|$이고 $Y = y_{ab} / |\partial A - \partial B|$
   4.1 */A*>1이고 */B*>1이라면,
      4.1.1 X>Y이면 T=X
      4.1.2 이외의 경우 T=Y
   4.2 (*/A*와 */B*) >0 이고 (*/A* 또는 */B*)<1 이라면
      $T = |x_{ab} - y_{ab}| / |\partial A - \partial B|$
   4.3 이외의 경우 T = X+Y

4.1.3 최근접 객체 탐색

CTNN 질의는 시작 시간에서 전체 데이터 객체와 질의점 사이의 거리 계산이후, 최소한의 추가 계산 작업만으로 다음 하위 시간 인터벌의 최근접 객체들을 선택한다. 단지, 현재 p의 유효시간 인터벌 내에서 어떤 객체가 삽입 또는 삭제되었을 때 이 갱신으로 인해 인터벌이 변경되는지, 다른 최근접 객체가 선택되는지만 체크하면 된다. 나머지 평가시간 위에서는, 질의와 교차하는 어떤 세그먼트 p', 또는 현재의 p와 교차하는 세그먼트 p'가 있을 때, 교차 시간 이후의 최근접 객체를 p에서 p'로 변경시켜주기만 하면 된다.

4.2 Approximate CTNN 질의

실세계 응용에서 어떤 사용자들은 결과의 정확도보다는 빠른 응답 시간을 더욱 중요하게 생각할 수도 있다. 이러한 조건을 만족시키고자 할 때 우리는 근사(approximate) 질의 처리 기법을 사용한다. CTNN 질의는 기존 연구들보다 적은 계산을 행하기는 하지만 세그먼트 개수가 많아질수록 전체 교차점 검색이나 선분 세그먼트의 대칭 이동 횟수가 크게 증가하기 때문에 느린 응답시간을 보일 수도 있다. 따라

서 우리는 빠른 시간 안에 근사 결과 정보를 제공해 줄 수 있는 Approximate CTNN 기법(알고리즘 3)을 제안한다.

알고리즘 3. Approximate_CTNN

Input: 데이터 객체 세그먼트 집합 *seg*, 질의 객체 궤적
Output: 시간 리스트 *TL*
// 후보 객체가 여럿인 경우, 항상 **Calculate_Displacement( )** 적용
1. **Set_EvaluationTime( )** // 두 세그먼트가 실제로 교차하는 교차점만을 평가시간으로 선택
2. $|seg - q|^{ts}$
   2.1 *mindist* 가지는 최근접 객체 p 선택 → $TL = <p, [t_s, t_{el}]>$
3. 탐색원 h그리기 // 지름: $|q\text{-}p|^{ts}$, 중심점: 선분pq의 중점
4. h와 교차하는 □p'가 존재하는지 검색
   4.1 □p'가 존재 한다면,
      4.1.1 교차시간 t_h 설정
         4.1.1.1 h와 □p'가 교차하는 두 점 중, 이후 시간을 가지는 교차점 선택
         4.1.1.2 한점만 교차한다면, 그 점을 교차점으로 선택
      4.1.2 $|q - p'|^{th} < |q - p|^{th}$ 라면 $TL = \{<p, [t_s, t_h]>, <p', [t_h, t_{el}]>\}$
      4.1.3 아니면 $TL = <p, [t_s, t_{el}]>$
   4.2 □p'가 존재하지 않는다면, $TL = <p, [t_s, t_{el}]>$
5. **Until** t_{el}
   5.1 t_i시간에 IL내의 교차점 i_1가 존재할 때,
      // 단계 순서대로 조건이 만족하면 TL 갱신
      5.1.1 □q ⊥ □p'라면 $TL = \{<p, [t_s, t_i]>, <p', [t_i, t_{el}]>\}$
      5.1.2 □p ⊥ □p''라면 p''에 대해 3~4단계 반복
      5.1.3 이외의 경우 $TL = <p, [t_s, t_{el}]>$
   5.2 t_c시간에 EL내의 생성점 p'가 존재할 때,
      5.2.1 $|p' - q|^{tc} < |p' - q|^{tc}$ 이면 $TL = \{<p, [t_s, t_c]>, <p', [t_c, t_{el}]>\}$
      5.2.2 이외의 경우 $TL = <p, [t_s, t_{el}]>$
   5.3 t_d시간에 EL내의 소멸점 p'가 존재할 때,
      5.3.1 p'==p이면 2단계 반복하고 최근접 객체 p'' 선택,
         $TL = \{<p, [t_s, t_d]>, <p'', [t_d, t_{el}]>\}$
      5.3.2 이외의 경우 $TL = <p, [t_s, t_{el}]>$
6. **Until** t_e
   6.1 각 질의 세그먼트 q_i에 대해, t_{si}에서 3~4단계, t_{ei}까지 5단계 반복 수행
7. *TL* 반환

Approximate 기법은 선분 대칭이동 작업을 필요로 하지 않기 때문에 좀 더 빠른 응답시간을 가질 수 있다. 그리고 교차점을 탐색하는 과정에서, 알고리즘2와는 다르게 실제로 두 세그먼트가 같은 시간에 도달하는 교차점만을 평가시간으로 선택하기 때문에, 최근접 객체 정보 변경 여부를 탐색하는 평가시간 점 개수가 줄어든다. 이를 통해 정확도가 줄어들 수 있으므로 이를 해결하고자 우리는 탐색원과 어떤 세그먼트의 교차점을 평가시간에 포함한다.

4.3 Exact CTNN 질의

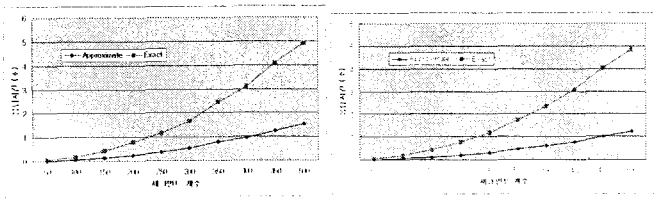
CTNN 기법에서 선분 세그먼트의 대칭 이동은 많은 계산 비용을 필요로 할 수 있기 때문에, 우리는 어떤 세그먼트들을 어느 위치로 언제 이동 시킬 것이지를 결정함으로써 이 비용을 최소한으로 줄여야 한다. 이 때 선택된 세그먼트들이 대칭 이동 후 집합되는 공간을 '기준 공간'이라 부른다. Exact CTNN 기법은 다음의 알고리즘 4와 같이 처리된다.

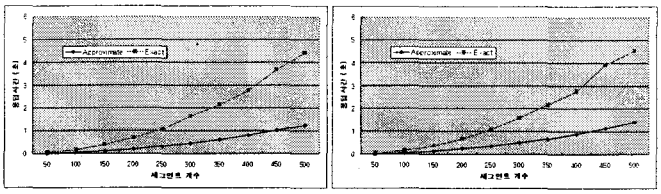
알고리즘 4. Exact_CTNN

Input: 데이터 객체 세그먼트 집합 *seg*, 질의 객체 궤적
Output: 시간 리스트 *TL*
// 후보 객체가 여럿인 경우, 항상 **Calculate_Displacement( )** 적용
// 기준 공간: 첫 번째 교차점 i_1이 존재하는 공간
1. **Set_EvaluationTime( )**
2. $|seg - q|^{ts}$ //기준 공간에 존재하는 seg에 대해서만 거리 계산
   2.1 *mindist* 가지는 최근접 객체 p 선택 → $TL = <p, [t_s, t_{el}]>$
3. 탐색원 h그리기 // 지름: $|q\text{-}p|^{ts}$, 중심점: 선분pq의 중점
4. h와 교차하는 □p'가 존재하는지 검색
   // 알고리즘 3의 4단계와 동일하게 수행
5. **Until** t_{el}
   5.1 t_i시간에 IL내의 교차점 i_1가 존재할 때,
      // 단계 순서대로 조건이 만족하면 TL 갱신
      5.1.1 □q ⊥ □p'라면 $TL = \{<p, [t_s, t_i]>, <p', [t_i, t_{el}]>\}$
         5.1.1.1 t_i시간 이후의 □p'를 기준 공간으로 대칭이동
         5.1.1.2 □p'와 기준 공간 seg사이의 교차점 추출 → IL
      5.1.2 □p ⊥ □p''라면 i_1과 q에 대해 3~4단계 반복
      5.1.3 이외의 경우 $TL = <p, [t_s, t_{el}]>$
   5.2 t_c시간에 EL내의 생성점 p'가 존재할 때,
      5.2.1 $|p' - q|^{tc} < |p' - q|^{tc}$ 이면 $TL = \{<p, [t_s, t_c]>, <p', [t_c, t_{el}]>\}$
      5.2.2 이외의 경우 $TL = <p, [t_s, t_{el}]>$
   5.3 t_d시간에 EL내의 소멸점 p'가 존재할 때,
      5.3.1 p'==p이면 2단계 반복하고 최근접 객체 p'' 선택.
         $TL = \{<p, [t_s, t_d]>, <p'', [t_d, t_{el}]>\}$
      5.3.2 이외의 경우 $TL = <p, [t_s, t_{el}]>$
6. **Until** t_e
   6.1 각 질의 세그먼트 q_i에 대해, t_{si}에서 3~4단계, t_{ei}까지 5단계 반복 수행
7. *TL* 반환

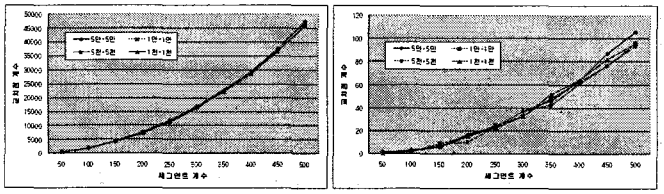
5. 실험 및 성능 평가

우리는 GSTD[8]를 사용하여 공간 범위 5만m^2, 1만m^2, 5천m^2, 1천m^2 내에서 세그먼트의 양 끝점 시간차를 10, 5, 3시간차로 정하고 데이터 50~500개를 50단위 차로 10번씩 생성하였다. 그리고 세그먼트가 분포하고 있는 공간 범위에 따라, 세그먼트의 끝점 시간차에 따라, 이동 객체 세그먼트 수당 각 기법이 얼마만큼의 질의 응답시간을 보이는지 살펴보기 위하여, 10번 생성된 임의 데이터 집합에 실험을 행하고 평균값을 결과로 제출하였다.

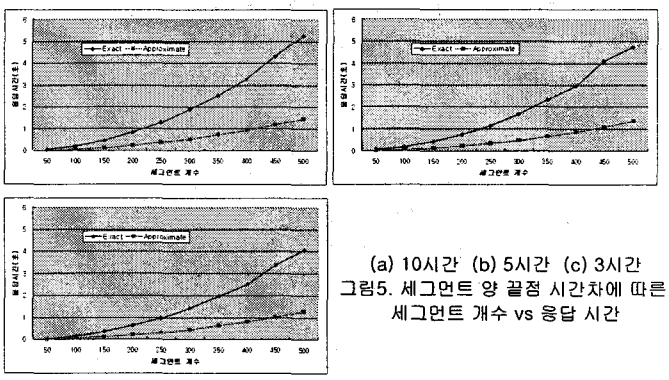




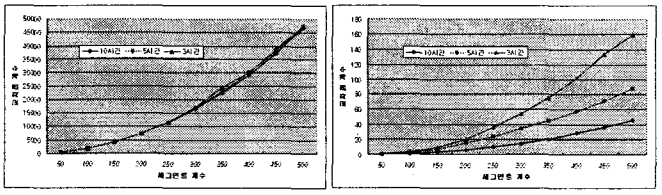
(a) 5만^2, (b) 1만^2, (c) 5천^2, (d) 1천^2
그림3. 공간 범위 분포에 따른, 세그먼트 개수 vs 응답 시간



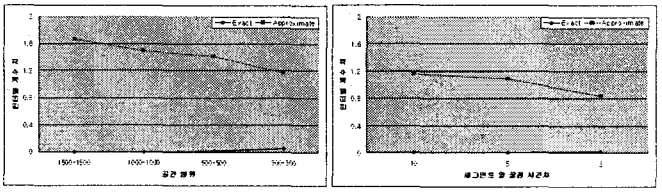
(a) Exact CTNN (b) Approximate CTNN
그림4. 공간 범위 분포에 따른, 교차점 개수



(a) 10시간 (b) 5시간 (c) 3시간
그림5. 세그먼트 양 끝점 시간차에 따른, 세그먼트 개수 vs 응답 시간



(a) Exact CTNN (b) Approximate CTNN
그림6. 세그먼트 양 끝점 시간차에 따른, 교차점 개수



(a) 공간 범위에 따른 정확도 (b) 시간 범위에 따른 정확도
그림7. Exact, Approximate CTNN의 정확도

이 때, 두 기법은 모두 세그먼트의 공간 분포 범위에 상관없이 일정한 교차점 개수를 가지며 따라서 응답 시간도 일정하게 나타나는 것을 발견하였다(그림 3). 이는 그림 4와 같이, 세그먼트 간 교차점 개수가 공간 범위당 크게 달라지지 않기 때문이다. 그리고 세그먼트 양 끝 시간차가 줄어들수록 Exact는 응답시간이 줄어들고, Approximate는 응답시간에 큰 영향을 받지 않는 것을 알았다(그림 5). 이는 Exact의 경우, 시간차가 줄어들수록 세그먼트가 밀집되어 분포하므로 한 곳에서 교차하는 세그먼트의 개수가 많아지기 때문에 실제 평가시간으로 사용되는 교차점 개수가 줄어들지만, Approximate는 이에 크게 영향 받지 않기 때문이다(그림 6). 그림 7은 단순 탐색을 통해 얻어진 최근접 질의 결과와 Approximate, Exact 기법의 하위 인터벌 개수 차이를 나타낸다. 이는 각 기법의 정확도를 알아보고자 한 것이며, 실험 결과의 하위 인터벌의 시작, 끝 시간이 단순 탐색 기법이 반환한 결과 인터벌의 시작, 끝 시간과 1초 이내의 차이를 보이는 경우만 두 인터벌이 동일한 결과를 보였다고 간주한 결과이다.

6. 결 론

지금까지 우리는 질의와 데이터 객체가 모두 이동 객체인 경우에 가장 유용하게 사용될 수 있는 새로운 최근접 질의 처리 기법들을 살펴보았다. CTNN 기법은 궤적 최근접 질의 개념과 변위 계산을 사용하여 질의와 가장 가까운 위치를 유지하며 움직이는 객체를 결과로 선택하였으며, 최근접 객체가 변경되는 연속적인 시간을 탐색하기 위하여 객체 세그먼트들의 변곡점과 교차점 위에서 결과 변경 여부를 비교하였다. 그리고 각 평가시간점 위에서 최소한의 추가 거리 계산 작업만을 수행함으로써 질의 처리 시간을 줄이고자 하였다. 또한 기존 기법들은 모두 질의 시작 시간에서 모든 데이터와 질의 사이의 거리를 계산하지만 Exact 기법의 경우 질의 시작 시간에서 질의 궤적을 기준으로 한 쪽 공간에 존재하는 세그먼트와 질의 사이의 거리만을 계산하므로 더 적은 거리 계산 횟수를 가진다.

최근접 질의 처리는 인덱스 구조를 이용하여 데이터를 저장, 관리하고 노드 단위로 질의 처리하는 경우 훨씬 효율적인 성능을 보일 수 있다. 우리는 현재 인덱스 구조 기반 CTNN 기법을 실험중이며, 현재까지 3DR-tree가 가장 적합한 인덱스 구조임을 발견하였다. 이에 대한 자세한 성능 평가와 알고리즘 제시는 이후에 보일 것이며, 그 이후로는 이동 영역 객체에 대한 CTNN 기법도 연구하고자 한다.

참고 문헌

[1] A. Prasad Sistla, Ouri Wolfson, Sam Chamberlain, Son Dao, "Modeling and Querying Moving Objects", ICDE 1997, pp.422~432

[2] Rimantas Benetis, Christian S. Jensen, Gytis Karciauskas, Simonas Saltenis, "Nearest Neighbor and Reverse Nearest Neighbor Queries for Moving Objects", IDEAS 2002, pp.44~53

[3] Simonas Saltenis, Christian S. Jensen, "Indexing the Position of Continuously Moving Object", SIGMOD Conference 2000, pp.331~342

[4] Nick Roussopoulos, Stephen Kelley, Fredeic Vincent, "Nearest Neighbor Queries", SIGMOD Conference 1995, pp.71~79

[5] Yufei Tao, Dimitris Papadias, "Spatial Queries in Dynamic Environments", *TODS 2003*

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

[7] Kriengkrai Porkaew, Iosif Lazaridis, Sharad Mehrotra, "Querying Mobile Objects in Spatio-Temporal Databases", SSTD 2001, pp.59~78

[8] Yannis Theodoridis, Mario A. Nascimento, "Generating Spatiotemporal Datasets on the WWW", SIGMOD 2000, pp.39~43