부분순위 연속 k 최근접 객체 탐색 기법

김진덕*

A Method for Continuous k Nearest Neighbor Search With Partial Order

Jindeog Kim*

이 논문은 2010년도 동의대학교 교내연구비를 지원받았음(과제번호 2010AA179)

요 약

위치기반서비스와 지능형교통시스템 등의 응용분야에서는 이동 중인 경로 상에 존재하는 모든 지점에 대해 k개의 최근접 객체를 탐색하는 연속 k 최근접 객체 탐색 질의가 폭넓게 사용되고 있다. 최근접 질의는 위와 같은 응용에 빠른 응답을 요구하고, 공간 네트워크 데이터베이스에 적용가능해야 한다. 또한 잦은 POI(Point of Interest)객체의 변경에 유연하게 대처해야 한다.

이 논문에서는 도로 네트워크에서 이동 중인 질의 객체를 위한 최근접 객체를 효율적으로 탐색하는 새로운 기법을 제안하고자 한다. 제안하는 기법은 다수의 분할점과 그에 상응하는 k개의 최근접 객체 집합들을 결과로 추출하며, POI들 간에는 순서가 없다. 실제 데이터를 이용한 실험은 제안한 기법에 기존 기법에 비해 우수함을 보인다. 최적의 조건에서 제안한 기법이 기존 기법에 비해 짧은 연산 시간(15%)을 보인다.

ABSTRACT

In the application areas of LBS(Location Based Service) and ITS(Intelligent Transportation System), continuous k-nearest neighbor query(CkNN) which is defined as a query to find the nearest points of interest to all the points on a given path is widely used. It is necessary to acquire results quickly in the above applications and be applicable to spatial network databases. It is also able to cope successfully with frequent updates of POI objects.

This paper proposes a new method to search nearest POIs for moving query objects on the spatial networks. The method produces a set of split points and their corresponding k-POIs as results with partial order among k-POIs. The results obtained from experiments with real dataset show that the proposed method outperforms the existing methods. The proposed method achieves very short processing time(15%) compared with the existing method.

키워드

연속최근접객체, 부분순위, 분할점, 공간네트워크

Key word

CkNN, Partial Order, Split Point, Spatial Network

* 정회원 : 동의대학교 (교신저자, jdk@deu.ac.kr)

접수일자 : 2010. 07. 09
심사완료일자 : 2010. 07. 22

Ⅰ. 서 론

데이터베이스 시스템에서 사용자에게 가장 가까운 관심객체(POI)를 검색하는 질의는 가장 많이 사용되는 질의 중의 하나이다. 특히 최근 이동체 데이터베이스의 발달과 함께 이동 중인 사용자에 대한 질의 처리는 반드시 필요하다.

그래서 다양한 유형의 최근접 객체 검색에 대한 연구가 진행되었고, 도로상의 경로를 이동 중인 질의점과 고정객체에 대한 연속 kNN질의는 많은 응용분야에서 사용되고 있다. 이러한 질의를 처리하기 위한 연산들은 많은 수의 질의점과 POI 객체가 관련되어 있으며, 빠른 질의처리를 요구한다. 그리고 공간 네트워크 데이터베이스에서도 적용가능해야 하며, 또한 POI의 잦은 변화에도 유연하게 대처해야 한다.

그러나 지금까지의 다양한 연구는 연속kNN 질의처리를 위해 각 단계마다 거리 및 방향 계산을 요구하여 실시간 질의 응답이 어렵거나, 계산 시간을 줄이기 위해 매우 많은 양의 사전 데이터를 관리하는 비용이 요구된다.

따라서 이 논문에서는 효율적인 부분순위 연속 k 최근접 객체 검색 기법을 제안하고자 한다. 제안하는 기법은 공간 네트워크의 각 노드별로 최근접 객체들만을 사전 데이터로 저장하며, 이동 경로에 대해 kNN이 변경되는 지점인 분할점을 간단한 계산에 의해 도출한다. 제안한 기법은 적은 수의 분할점을 생성하며, 거리와 방향계산을 하지 않아 수행 시간을 단축할 수 있다.

실제 도로 데이터와 POI 정보를 활용하여 분할점의 생성 횟수 및 연산 회수를 비교하는 실험으로부터 얻어진 결과를 통해 이 논문에서 제안하는 기법의 효율성을 입증하고자 한다.

이 논문의 구성은 다음과 같다. 제 2 장에서는 정규 최근접 객체 검색, 연속 최근접 객체 검색, 이동체 연속 최근접 객체검색 기법에 대한 다양한 관련연구를 살펴보고, 제 3장에서는 이 논문에서 제안하는 부분 순위 연속 k 최근접 객체 기법에 대해 자세히 설명한다. 제 4장에서는 기존 연구와의 성능 비교 실험 결과를 제시하며, 제 5장에 결론을 맺는다.

Ⅱ. 관련 연구

전통적인 데이터베이스에서는 하나의 고정 질의점과 다수의 고정 객체 사이의 kNN에 대한 연구가 진행되었으며, 일반적인 k 최근접 객체 검색 기법은 인덱스[5,7]를 사용하는 기법이 많이 연구되어 왔다. 관련 연구[5]에서는 R-tree를 사용하여 최근접 객체를 검색하는 기법을 제안하였다. 그러나 이 방법은 많은 디스크 탐색이 요구되고 공간 네트워크에 적용하기 어렵다.

네트워크 거리를 기반으로 하는 k 최근접 탐색 알고리즘은 네트워크 확장 방법[10]과 사전 거리 계산법[1, 2]으로 분류할 수 있다. 관련 연구 [10]은 INE(Incremental Network Expansion) 기법을 제시하여, 질의가 속한 에지를 기초로 에지에 존재하는 최근접점을 찾기 위해 네트워크를 확장해 간다. 새롭게 추가되는 에지의 거리가 k 번째 최근접 거리 보다 클 때 까지 수행한다. 이 방법은 Dijkstra 알고리즘을 사용하여 많은 에지가 있을 경우 효율이 떨어진다. 관련 연구 [2]는 보로노이 다이어그램을 네트워크에 활용한 VN3를 제안하였다. 그러나 이 방법은 과도한 보로노이 다이어그램을 생성할 가능성이 높아 확장 비용과 확장시 거리 재계산 비용이 크다. 관련 연구[1]은 ISLAND 기법을 제시하였으며, 이는 각 노드에서 POI까지의 거리를 사전에 계산하여 성능을 높이고자 하였다. 그렇지만 POI 위치 갱신 비용이 매우 증가하는 문제점이 있다.

최근에는 질의점이 이동 중인 연속 kNN에 대한 연구 및 이동 중인 POI 객체에 대한 최근접 객체의 모니터링에 관한 연구[4]도 있으며, 유클리디언(Euclidean) 거리보다 공간 네트워크 거리[3]기반 최근접 객체에 대한 연구가 진행 중이다. 그리고 이동 객체의 연구와 더불어 연속 k 최근접 객체 검색에 관한 연구도 활발하다. 관련연구 [8]에서는 연속 k 최근접 객체 검색 모델을 처음으로 제안하였으며, 관련연구 [6]에서는 시간기반 질의 개념을 도입하였다. 그러나 유클리디언 거리를 기반으로 수행된다. 관련 연구 [9]는 도로 네트워크 기반 cNN 기법을 제시하였으며, 관련 연구 [3]은 도로 네트워크 기반 CkNN 기법인 IE를 제시하였다. 이 연구에서는 질의 경로 상에서 kNN객체가 변하는 분할점을 추출하는 기법과 성능개선 방안으로 UBA 기법도 제시하였다. 그러나 도로 네트워크 상에 존재하며, 두 링크에서

접근 가능한 경로가 다를 경우 효율적인 증감 리스트 관리 방법이 제시되지 않아 거리 및 방향을 조사하는 횟수가 과도하게 많으며, 각 단계마다 증감리스트를 재계산해야 한다.

Ⅲ. 부분순위 연속 kNN 탐색 기법

이 논문에서는 도로 네트워크 기반으로 각 k개 객체간의 부분 순위 CkNN 탐색 기법을 제시하고자 하며, 증가 및 감소 노드 분할법을 통해 분할점을 효과적으로 추출하는 기법을 제안하고자 한다. 부분 순위 연속 k 최근접 객체 탐색 질의는 사용자가 주어진 경로로 이동 중일 때 연속적인 k개의 최근접 객체를 탐색하는 것이며, k개 객체간의 거리의 변동에 따른 순위 변화는 무시하는 것으로 정의한다. 이 때 POI는 고정객체인 것으로 가정한다. 향후, 부분순위 연속 kNN 탐색 기법을 CkNN_PO (Continuous k Nearest Neighbor with Partial Order)로 명명한다.

예를 들어 이동 객체(예, 자동차)가 주어진 경로를 주행 중이며, 3개의 최근거리 식당을 검색하는 질의이다. 이 질의의 결과는 {구간, kNN}의 집합으로 구성된다. 이 때 하는 분할점(Split Point)을 추출함으로써 구간을 설정할 수 있다.

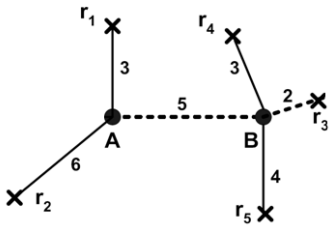

그림 1. 링크와 POI 객체
Fig. 1 Link and POI Objects

따라서 연속 kNN 탐색 질의는 주행 경로 상의 효율적인 분할점 탐색이 매우 중요하며, 이는 그림 1과 같이 경로상의 여러 개의 링크 중 각 링크에서 분할점을 결정하는 문제로 귀결된다.

그리고 기존 IE 기법[3]의 점진적 구간 추출 작업에 비해 제안한 기법은 일괄작업으로 진행된다. 일괄작업 시 최단 경로 검색 과정이 배제되는 기법이다. 제안하는 기법은 사전 검색된 각 노드의 최근접 객체 정보만을 이용한다.

3.1 CkNN_PO를 위한 자료 구조

공간 네트워크는 아래와 같이 노드테이블, 링크 테이블, POI 테이블로 구성된다.

- 노드테이블 :

ID	최근접 k POI ID	거리	연결링크 ID

- 링크 테이블 :

ID	Start Node	End Node	노드길이

- POI 테이블 :

ID	링크 ID	시작노드에서의 거리

이상의 정보에서 각 테이블의 필드는 기존 도로 지도의 위상(Topology)정보를 포함하는 일반적인 모델이며, 다만 노드테이블의 '최근접 k POI ID'와 '거리' 필드는 이 논문에서 사전에 조사하여 저장하는 정보이다. k개의 최근접 POI ID와 거리 정보만으로 경로상의 최근접 객체를 추출할 수 있는 것은 아래 특성[3] 때문이다.

하나의 링크 $P(N_i, N_j)$가 있을 때
- N_i는 출발 노드, N_j는 종료 노드
- N_i의 kNN은 $O_i = \{O_1, \cdots, O_k\}$
- N_j의 kNN은 $O_j = \{O'_1, \cdots, O'_k\}$

단, P의 kNN는 subset of $\{O_i \cup O_j\}$

k 개의 최근접 POI ID와 거리만을 저장하므로 POI의 정보 변경일 생길 경우 대부분 해당 노드에 국한될 경우가 많고, 복잡한 자료구조가 아니므로 갱신 비용이 매우 작다는 장점이 있다.

3.2 CkNN_PO 처리 단계

그림 1에서 노드 A로부터 출발하며, 3NN을 구한다고 가정할 때 첫 번째 분할점 구하기 위해서는 IE&UBA[3]는 아래의 단계를 거친다.

① 각 노드의 3NN 및 거리를 구한다.
- A의 3NN 및 거리 = $\{(r_1, 3), (r_2, 6), (r_3, 7)\}$
- B의 3NN 및 거리 = $\{(r_3, 2), (r_4, 3), (r_5, 4)\}$
- 따라서 링크의 3NN은 $\{r_1, r_2, r_3, r_4, r_5\}$의 subset

② 링크의 출발점에서 증감리스트를 구한다.

- 증가 리스트 {(r_1,3), (r_2, 6)}
- 감소 리스트 {(r_3, 7), (r_4, 8), (r_5, 9)}

③ 분할점 도출
- 두개의 연속 멤버 O_i, O_i+1에 대해 O_i는 증가, O_i+1는 감소리스트일 때 분할점은

$$\frac{d(A,o_{i+1})-d(A,o_i)}{2}$$

- 따라서 SP = [7 – 6] / 2 = 0.5이며, 이는 A로 부터 0.5 지점이 분할점이며, 0.5 지점 이후로는 3NN이 바뀌게 됨을 의미한다.

④ 0.5 지점을 새로운 출발점으로 하여 다시 단계 1부터 시작하며, 분할점이 종료점 B보다 같거나 클 때까지 반복한다.

그러나 이 방법[3]은 매번 3NN 및 거리를 구해야 하며, 각 지점에서 증감리스트를 다시 구해야 한다. 또한 k개의 객체 집합 자체는 변하지 않지만 k개 객체내의 순서가 바뀌어도 분할점을 생성하므로 연산 비용이 크다는 단점이 있다.

따라서 본 논문에서는 위와 같은 문제점을 개선하며, 보다 연산 효율이 좋은 기법을 제시하고자한다. 이 기법은 3단계로 구성된다.

1단계는 각 노드에 인접해 있는 n개의 POI 객체와 거리를 구하는 것으로서 사전계산에 의해 테이블에 저장된다. 이 때 Dijkstra 알고리즘에 의해 노드와 POI간의 거리를 구하여 가장 가까운 n개의 POI를 선택한다. 2단계는 두개의 노드에 연결된 2k개의 POI에 대해 증가객체와 감소객체를 구별한다. 3단계는 일괄 처리 방법을 이용하여 분할점을 추출한다.

여기서는 1단계는 기존 방법과 동일하므로 설명을 생략하기로 하며, 2단계와 3단계를 처리 방법을 자세히 설명한다.

(1) 증가객체와 감소객체의 구분

증가 객체라 함은 질의점이 이동함에 따라 거리가 증가하는 객체이며, 감소객체는 거리가 감소하는 객체이다. S와 E노드는 각각 출발점과 종료점이고 링크의 길이를 L이라 할 때, 아래의 분류법에 따라 증가 및 감소 여부를 판단한다. 그리고 구해진 증가 및 감소 리스트를 정렬한다.

① S에 있고 E에는 없다면 증가객체
② S에 없고 E에만 존재하면 감소객체
③ 두 노드에 모두 있고 Dist(S,O) = L + Dist(E,O)이면 감소객체
④ 두 노드에 모두 있고 Dist(S,O) + L = Dist(E,O)이면 증가객체
⑤ 그 이외의 경우에는 [Dist(E,O) + L – Dist(S,O)] / 2 가 증가객체에서 감소객체로의 전환점이 됨.

(2) 분할점 및 kNN의 추출

우선 분할점이 존재하는 최소점과 최대점은 아래와 같이 정의할 수 있다.

- 최소점 = [min(dec list) – max(inc list)] / 2
- 최대점 = [max(dec list) – min(inc list)] / 2

만약, 최소점이 L보다 크면 그 링크에는 분할점이 존재하지 않으며, 경로상의 다음 링크로 전진할 수 있다.

최대값이 음수이면 분할점 없이 다음 링크로 전진할 수 있다. 따라서 UBA기법[3]과 같은 효과가 있으면서도 간단한 수식으로 계산된다.

그 외의 경우에 분할점의 추출은 아래와 같은 6단계를 거친다.

① 초기에 증가 리스트와 감소 리스트에서 2원합병으로 kNN구함. 이 때 같은 거리 값을 갖는 증가 객체와 감소객체가 있을 경우 감소 객체를 kNN에 포함시킴(방향성고려).
② kNN에 포함되는 감소 리스트의 객체는 리스트에서 제외(규칙1).
③ kNN에서 빠진 증가 리스트의 객체는 kNN 후보 집합과 리스트에서 제외(규칙2).
④ 분할점{SPi = [min(dec list) – max(inc list)] / 2}를 구함. 이 때 분할점 계산에 참여한 증가객체는 kNN 후보집합에서 제외, 감소객체는 kNN 후보집합에 포함.
⑤ kNN 후보집합으로부터 kNN을 구함
⑥ 감소 리스트가 존재하지 않거나 분할점이 최대점보다 크면 종료, 아니면 단계 2로 분할점 및 5NN 추출 과정을 그림 2의 예를 들어 설명하고자 한다.

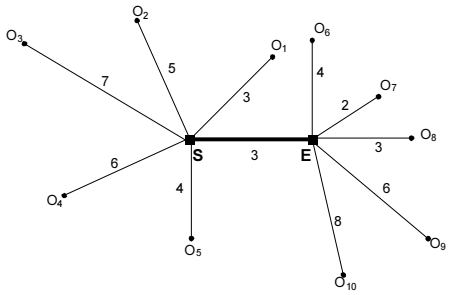

그림 2. 5NN위한 링크와 POI
Fig. 2. A Links and POIs for 5NN

단계1의 결과로서 S노드의 5NN{(O_1,3), (O_5,4), (O_2,5), (O_7,5), (O_4,6)}과 E노드의 5NN{(O_2,2), (O_8,3), (O_6,4), (O_9,6), (O_1,6)}이 사전에 저장된다. 이 때 각 쌍은 표1과 같이 POI ID와 거리이다.

표 1. 초기 증감 리스트
Table 1. Initial Increment and Decrement List

Inc List		Dec List	
POI ID	Dist	POI ID	Dist
1	3	7	5
5	4	8	6
2	5	6	7
4	6	9	9

초기 5NN은 {(O_1,3), (O_5,4), (O_2,5), (O_7,5), (O_8,6)}이다. 그리고 (규칙1)에 의해 O7, O8이 감소 리스트에서 제외된다. 그리고 (규칙2)에 의해 증가 리스트에 있던 O_4는 리스트에서 제외 된다.

표 2에서 아래첨자로 된 것은 제거된 POI이며, 이탤릭 문자는 max(inc_list)와 min(dec_list)로서 분할점 계산에 적용되는 객체들이다.

표 2. 새로운 증감 리스트
Table 2. New Increment and Decrement List

Inc List		Dec List	
POI ID	Dist	POI ID	Dist
1	3	7	5
5	4	8	6
2	5	6	7
4	6	9	9

초기 kNN 추출 후 새로운 증감리스트는 다음과 같은 성질을 만족하게 된다.

∀[Dist(inc list) < Dist(dec list)]

그리고 단계 4의 {SPi = [min(dec list) – max(inc list)] / 2}에 의해 SP1이 1(= [7-5]/2)이 된다. 따라서 구간은 [0, 1]이며, 5NN은 {O_1, O_5, O_2, O_7, O_8}이다. 단계 6에 의해 감소리스트가 존재하므로 단계 2가 수행되며, 두 번째 분할점을 위한 증감리스트는 표 3과 같다.

표 3. 첫 번째 분할점 후 증감 리스트
Table 3. Inc & Dec List after first SP

Inc List		Dec List	
POI ID	Dist	POI ID	Dist
1	3	7	5
5	4	8	6
2	5	6	7
4	6	9	9

표 3을 근거로 한 분할점 SP2는 2.5(= [9-4]/2)가 된다. 따라서 구간은 [1, 2.5]이며, 5NN은 {O_1, O_5, O_7, O_8, O_6}이다. 두 번째 분할점 이후의 증감리스트는 표 4와 같다. 감소리스트가 존재하지 않으므로 링크상의 분할점은 더 이상 존재하지 않으며, 링크의 도착점이 마지막 분할점이 된다. 따라서 구간은 [2.5, 3]이며, 5NN은 {O_1, O_7, O_8, O_6, O_9}이다.

표 4. 두 번째 분할점 후 증감 리스트
Table 4. Inc & Dec List after second SP

Inc List		Dec List	
POI ID	Dist	POI ID	Dist
1	3	7	5
5	4	8	6
2	5	6	7
4	6	9	9

그러므로 그림 2의 링크에서는 두 개의 분할점이 존재하며, 그로 인해 표 5와 같이 3개의 구간과 각 5NN들이 질의 결과로 반환된다.

표 5. 최종 질의 결과
Table 5. Final Query Result

구간	5NN
0 - 1	{O_1, O_5, O_2, O_7, O_8}
1 - 2.5	{O_1, O_5, O_7, O_8, O_6}
2.5 - 3	{O_1, O_7, O_8, O_6, O_9}

Ⅳ. 성능평가

여기서는 기존의 CkNN의 연구인 IE&UBA 기법[3]과 CkNN_PO의 성능을 평가하고자 한다.

우선 두 가지 기법의 특징을 분석해 보고자 한다. 그림 1에 대해 분할점 생성을 하고자 할 경우 IE&UBA 기법은 6개의 분할점이 생성되며, 두 링크에서 접근 가능한 경로가 다를 경우의 효율적인 증감 리스트 관리 방법이 제시되지 않아 각 분할점마다 분할점으로부터 각 객체간의 거리 및 방향을 계산해야 한다. 또한 후보 집합에 대한 새로운 정렬이 필요하다.

그렇지만 이 논문에서 제안한 CkNN_PO 기법은 2개의 분할점(1, 2.5)이 생성되며, 분할점으로부터 각 객체간의 거리 및 방향 계산이 불필요한 장점이 있다. 또한 2원 합병기법으로 한번 정렬된 후보 집합에 대한 부분정렬만이 필요하다[12].

이러한 분석 결과는 실험 결과로도 입증된다. 그림 3은 실험 평가를 위한 부산시 실제 도로 네트워크를 나타낸 것이다. 실험데이터는 전체 7623개의 링크로 구성되어 있다. 그리고 POI 데이터는 14068개로 구성되어 있다. 각 POI는 x 및 y 좌표와 POI 이름으로 구성되어 있다.

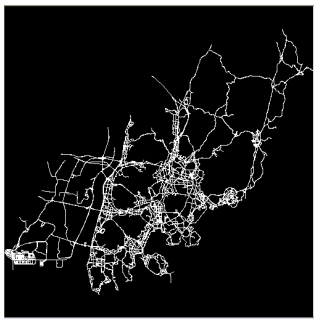

그림 3. 실험 데이터(도로망)
Fig. 3. Test Data(Road Network)

실험은 임의의 링크에 대해 각 기법을 500회 수행한 뒤 평균 수행 시간과 평균 분할점의 개수를 측정하였으며, 또한 각 기법의 모듈별 수행시간을 측정하였다.

그림 4는 IE&UBA 기법과 CkNN_PO 기법의 전체 연산 시간을 3NN과 5NN에 대해 각각 나타낸 그래프이다. 실험 결과 이 논문에서 제안한 CkNN_PO가 보다 좋은 성능을 보여 주었다. 특히 구하고자 하는 최근접 객체의 수가 많을수록 연산시간은 증가했지만, CkNN_PO의 연산 시간은 완만히 증가하여 기존 기법에 비해 15%에 불과해 매우 좋은 성능을 보였다.

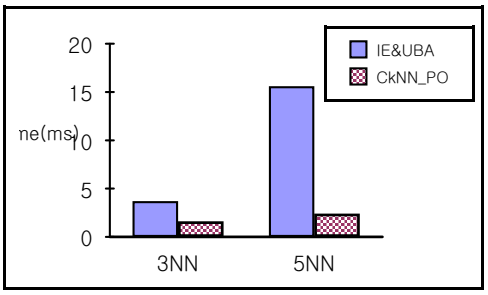

그림 4. 전체 연산 시간
Fig. 4. Total Processing Time

표 6은 각 기법별 분할점의 개수를 나타낸 것으로서 CkNN_PO가 보다 적게 도출되어 좋은 성능을 보였다.

표 6. 분할점의 개수
Table 6. Number of Split Point

	IE&UBA	CkNN_PO
분할점 개수(3NN)	1.9	1.1
분할점 개수(5NN)	7.6	2.7

표 7은 5NN을 수행했을 경우의 각 기법별 세부 모듈의 연산 시간을 나타낸 것이다. 기존기법은 '거리 계산' 및 '증감 리스트 관리'모듈이 많은 부분을 차지하고 있음을 알 수 있었다. 그리고 표 6에 나타난 결과와 같이 분할점의 개수가 많은 기존 기법이 전체적으로 모듈별 수행시간이 길게 측정되었다.

표 7. 모듈 별 연산 시간
Table 7. Processing Time of Modules

(5NN)	IE&UBA	CkNN_PO
Find kNN Candidate	1.4(9%)	0.65(27%)
Sort Candidate	2.18(14%)	0.36(15%)
Calc. Distance	6.71(43%)	-
Manage Inc & Dec list	2.65(17%)	0.77(32%)
Calc. Split Point	1.09(7%)	0.26(11%)
Generate kNN	0.76(5%)	0.14(6%)
ETC.	0.8(5%)	0.22(9%)
Total Time(ms)	15.6(100%)	2.4(100%)

Ⅴ. 결 론

이 논문에서는 도로 네트워크에서 이동 중인 질의 객체와 정적인 POI 객체들 간의 네트워크 거리를 기반으로 부분순위 k 최근접 객체를 효율적으로 검출하기 위한 기법인 CkNN_PO를 제시하고 실험으로 그 성능을 평가하였다. CkNN_PO는 각 노드에서 k개의 최근접 객체를 사전에 계산하여 저장하며, 주어진 경로에 대해 kNN의 구성요소가 변경되는 지점인 분할점을 간단한 계산에 의해 추출하는 효율적인 방법이다.

CkNN_PO는 부분순위 객체를 검출하므로 보다 적은 분할점을 생성하며, 알고리즘 수행 도중에는 거리와 방향계산을 하지 않아 수행 시간을 단축할 수 있는 장점이 있으며, 하나의 링크에 대해 분할점이 링크의 길이보다 크다면, 언제든지 추가 분할점 계산이 생략될 수 있어 보다 효율적인 기법이다.

실험 결과 이 논문에서 제안한 CkNN_PO 기법이 기존의 기법에 비해 최대 1/6로 연산 시간이 단축됨으로 알 수 있었다. 이 논문의 결과는 최근 활성화 되고 있는 모바일 단말기 기반 LBS 응용 프로그램에 기반 기술로 유용하게 사용될 수 있을 것이다.

참고문헌

[1] X. Huang, C.S. Jensen, S. Saltenis, “The Islands Approach to Nearest Neighbor Querying in Spatial Networks”, Proc. of Int’l Symp. On Spatial and Temporal Databases, SSTD, pp. 73-90, 2005

[2] M. Kolahdouzan, C. Shahabi, “Voronoi-Based K-Nearest Neighbor Search for Spatial Network Databases”, Proc. Of Int’l Conf. on Very Large Databases, VLDB, pp. 840-851, 2004

[3] M. Kolahdouzan, C. Shahabi, “Continuous K Nearest Neighbor Queries in Spatial Network Databases”, Proc. of STDBM, 2004

[4] K Mouratidis, M.L. Yiu, D. Papadias, N. Mamoulis, “Continuous Nearest Neighbor Monitoring in Road Networks”, VLDB, pp. 43-54, 2006

[5] N. Roussopoulos, S. Kelley, F. Vincent, “Nearest Neighbor Queries”, SIGMOD, 1995

[6] Y. Tao, D. Papadias, “Time Parameterized Queries in Spatio-Temporal Databases”, SIGMOD, 2002

[7] 이상철, 김상욱, “도로 네트워크 데이터베이스에서 근사 색인을 이용한 k-최근접 질의 처리”, 한국정보과학회 논문지 : 데이터베이스, 제 35권, 제 5호, pp. 447-458, 2008

[8] P. Sistla, O. Wolfson, S. Chamberlain, D. S, “Modeling and Querying Moving Objects”, IEEE ICDE 1997

[9] Y. Tao, D. Papadias, Q. Shen, “Continuous Nearest Neighbor Search”, VLDB, 2002

[10] D. Papadias, J. Zhang, M. Mamoulis, Y. Tao, ″Query Processing in Spatial Network Databases″, Proc. of VLDB, pp. 802-813, 2003

[12] 김진덕, “무순위 연속 k 최근접 객체 탐색을 위한 효율적인 분할점 추출기법”, 한국해양정보통신학회 춘계종합학술대회 논문집, 14권, 1호, 2010

저자소개

김진덕(Jindeog Kim)

1993년 부산대 컴퓨터공학과 (공학사)
1995년 부산대 대학원 컴퓨터공학과(공학석사)
2000년 부산대 대학원 컴퓨터공학과(공학박사)
1998.3~2001.2 부산정보대학 정보통신계열 전임강사
2001.3~ 현재 동의대학교 컴퓨터공학과 부교수
※관심분야 : 객체 지향 DB, 지리정보시스템, 공간 질의, 공간 색인, 모바일 데이터베이스, 텔레매틱스, GIS 스마트 동기화, 스트림 데이터베이스, 자동차 네트워크, 측위 시스템