사물인터넷융복합논문지 Vol. 12, No. 4, pp. 131-136, 2026 https://doi.org/10.20465/KIOTS.2026.12.4.131

이동 환경에서 Nearest Surrounder 기반 연속 최대 각노출 질의 처리 기법

정재화*
한국방송통신대학교 컴퓨터과학과 교수

Efficient Processing of Continuous Maximum Angular Exposure Queries Using Nearest Surrounders in Mobile Environments

Jaehwa Chung*
Professor, Department of Computer Science, Korea National Open University

요 약 모바일 위치 기반 서비스 등의 시스템에서는 사용자가 이동하는 동안 시야에서 가장 크게 노출되는 객체를 지속적으로 파악해야 한다. 그러나 이동 지점마다 Nearest Surrounder(NS)를 재수행하면 계산 비용이 급격히 증가한다. 본 논문은 공간에서 이동 질의점, 이동 구간, 객체 집합이 주어졌을 때, 객체별 가시각구간 길이를 시간에 따라 누적한 각노출 점수를 이용하여 CMAE(Continuous Maximum Angular Exposure) 질의를 정의한다. Snapshot-MAE는 R*-tree의 각구간과 madist 기반 가지치기를 이용하여 NS(q)를 구성하고, 객체별 각구간 길이를 집계하여 순간 MAE를 계산한다. Continuous-MAE는 현재 NS 결과와 NPP를 유지하며, 질의점이 NPP 내부에 있으면 CART/ART와 Range Ledger를 통해 각구간과 누적 점수만 갱신한다. NPP를 벗어난 경우에는 전체 각공간이 아닌 위반 각구간에 대해서만 PartialSweep을 수행한다. 성능 평가를 통해 제안 기법이 매 시점 Snapshot-MAE를 반복하는 Periodic MAE보다 낮은 처리 비용을 보임을 확인하였다.

주제어 : 이동 질의, Nearest Surrounder, 최대 각노출, CMAE, NPP

Abstract In the system such as mobile location-based services, it is necessary to continuously identify the object that occupies the largest angular portion of a moving user's view. Recomputing Nearest Surrounder(NS) results at every sampled location is costly. This paper proposes the Continuous Maximum Angular Exposure(CMAE) query in a two-dimensional Euclidean space. Given a moving query point , a time interval , and an object set, CMAE accumulates the length of each object's visible angular range, and returns the object with the highest exposure score. Snapshot-MAE constructs NS(q) using angular bounds and madist-based pruning, and then aggregates object-angular-range tuples. Continuous-MAE maintains the current NS result and a Non-Provoked Polygon(NPP). Inside the NPP, it updates angular ranges and scores using CART/ART and a Range Ledger; outside the NPP, it performs PartialSweep only over invalid angular ranges. Experiments show that the proposed method reduces the processing cost compared with a Periodic MAE baseline.

Key Words : Mobile Query, Nearest Surrounder, Maximum Angular Exposure, CMAE, Non-Provoked Polygon

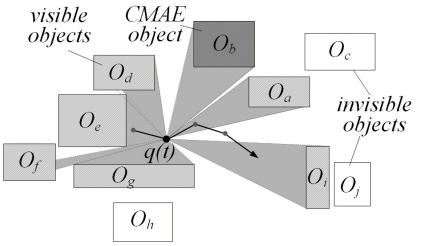
이 논문은 2022년도 3분기 한국방송통신대학교 학술연구비 지원을 받아 작성된 것임
*교신저자 : 정재화(jaehwachung@knou.ac.kr)
접수일 2026년 07월 21일 수정일 2026년 08월 10일 심사완료일 2026년 08월 19일

1. 서론

디지털 트윈 및 위치 기반 서비스에서는 사용자의 현재 위치에서 가장 가까운 객체 또는 가시적인 객체 탐색뿐만 아니라 이동 과정에서 가장 많이 노출되는 객체를 판단해야 하는 경우도 있다. 예를 들어 도시 안내 시스템은 사용자가 이동하는 동안 시야에서 큰 비중을 차지하는 표지판의 위치선정, 보행자용 안내 시스템은 보행자의 이동 과정에서 지속적으로 크게 보이는 랜드마크나 안내 시설물을 우선적으로 제공할 수 있다.

이러한 응용에서 중요한 것은 객체의 물리적 크기 자체가 아니라 사용자의 위치에서 해당 객체가 차지하는 방향 범위, 즉 각구간(angular range)의 크기이다[1].

기존 Nearest Neighbor(NN) 질의는 질의점에서 가장 가까운 객체를 반환하지만, 한 방향에 한정된 결과만 제공할 수 있다. 이에 비해 Nearest Surrounder(NS) 질의는 질의 q를 중심으로 전체 각공간을 여러 각구간으로 분할하고, 각 각구간에서 가장 가까운 객체를 반환한다[2,3]. 즉 $NS(q)$의 결과는 일반적으로 $\langle o_i,[\alpha,\beta)\rangle$와 같은 객체-각구간 튜플의 집합으로 표현된다. 그러나 NS 질의는 방향별 최근접 객체의 집합을 반환하는 데 초점을 두며, 특정 객체가 전체 각도 공간에서 얼마만큼 노출되는지는 판단하지 않는다. Continuous Nearest Surrounder(CNS) 질의는 이동 질의점에서 NS 결과를 지속적으로 유지하기 위해 안전 영역(safe region), 부분 재평가(partial query reevaluation), 증분 결과 갱신(incremental query result update)을 활용한다[4,5]. 이동 질의점의 연속 kNN 질의에서는 유효 구간, 결과 변화 시점 및 임계값 등을 이용하여 반복 탐색을 줄이는 방법이 연구되어 왔다[6-9]. 질의점 이동에 따라 기존 각구간 사이에는 중첩, 분할, 배제, 포함이 발생할 수 있다[5]. 이러한 특성은 이동 환경에서 각구간 기반 결과를 효율적으로 유지하는 데 중요하지만, CNS만으로는 객체별 가시각률 점수의 누적 및 최대 각노출 객체 판정을 직접 제공하지 않는다. 본 논문은 2차원 유클리드 공간에서 이동 질의점 $q(t)$가 주어졌을 때, 객체별 가시적 각구간의 길이를 노출 각도 점수로 정의하고, 이동 구간 $T=[t_s,t_e]$에서 누적 가시각률 점수가 가장 큰 객체를 찾는 CMAE(Continuous Maximum Angular Exposure) 질의를 제안한다. 객체 $o\in O$는 결과 후보이면서 동시에 다른 객체의 가시성을 차단할 수 있는 차폐 객체로 동작한다. 따라서 질의 처리는 단일 객체 집합 O와 단일 공간 인덱스를 대상으로 수행된다.



[Fig. 1] An example of CMAE query

Fig. 1은 제안한 CMAE 질의의 예이다. $q(t)$에서 각 객체의 보이는 크기 즉, 가시각구간 또는 가시각률을 비교하여 $MAE(q)$를 선택하고, 객체에 의해 가려진 객체는 낮은 가시성 상한을 갖는다. 현재 최대 객체가 경쟁 객체의 상한보다 크다고 인증되는 영역 R에서는 $q(t)$가 이동하더라도 동일한 객체를 대상으로 각구간만을 계산하여 재사용한다.

본 논문의 기여는 다음과 같다.

첫째, 2D 환경에서 객체별 가시각구간을 정의하고, 이동 구간 $q(t)$, $t\in[t_s,t_e]$에서 CMAE 질의를 정의한다.

둘째, CMAE 질의를 위해 객체별 가시각구간 집합 $\theta(q,o)$, 순간 가시각구간 $VAng(q,o)$, 정규 가시각률 $NAng(q,o)$, 이동 구간 누적 가시각률점수 $Score_T(o)$를 정의한다.

셋째, 모든 객체를 후보 객체이자 차폐 객체로 통합하여, 별도의 공간 모델 없이 단일 객체 집합만으로 가시성과 비가시성을 처리하는 질의 환경을 제시한다.

넷째, R*-tree 기반 휴리스틱을 결합하여 이동 중 반복 계산과 서버 갱신 요청을 줄이는 CMAE 처리 기법을 제안한다.

2. 관련 연구

NS 질의는 질의 q를 중심으로 전체 방향을 각도 구간으로 나누고 각 구간에서 가장 가까운 객체를 반환한다. Lee 등은 R-tree의 angle-based bound와 distance-bound property를 이용하여 Sweep 및 Ripple 알고리즘을 제안하였다[2,10]. 반면 Round-Eye는 이동 객체 환경에서 NS 결과를 지속적으로 추적하기 위한 개념을 제안하였다[4]. 또한 Chung은 CNS 질의를 위해 NPP, CART/ART, PartialSweep을 포함한 연속 NS 유지 기법을 제시하였다[5]. Visible Nearest Neighbor 및 Continuous Visible Nearest Neighbor 질의는 장애

물 환경에서 가시적인 최근접 객체를 탐색하거나 연속적으로 유지하는 문제를 다룬다[11,12]. Viewshed 분석은 특정 위치에서 어떤 영역 또는 객체 표면이 보이는지를 정밀한 계산 방법을 다루었다[13]. 거리 기반 best-first 탐색 및 distance browsing은 NN 계열 질의의 대표적인 탐색 방식이다[14]. 그러나 기존 NS/CNS 연구는 방향별 최근접 객체 또는 NS 결과의 연속 유지에 초점을 두고, 가시 최근접 질의는 가시적인 최근접 객체 탐색에 초점을 둔다. 반면 CMAE는 객체별 가시각률을 이동 구간 동안 누적하여 최대 각노출 객체를 반환하므로, 기존 질의와 목적이 다르다.

3. 문제 정의

전체 공간은 2차원 유클리드 평면 $\mathbb{R}^2$이다. 질의 q는 사용자의 현재 위치를 의미하며, 연속 질의에서는 시간에 따라 이동하는 위치 $q(t)$로 표현된다.

$$q(t)\in R^2,\quad t\in T=[t_s,t_e]$$

객체 집합 $O=\{o_1,o_2,\cdots,o_n\}$의 $o\in O$는 2차원 공간상의 폐곡선 또는 다각형 객체로 정의되고 각 객체 o는 MBR(minimum bounding rectangle)로 추상화된다. 또한 질의 q는 객체 내부에 위치하지 않는다고 가정한다. 즉, $q(t)$는 이동 가능한 자유공간에 존재한다.

질의 q를 원점으로 하는 극좌표계를 사용한다. θ는 $[0,2\Pi)$ 범위의 각도이다. 각구간은 반개구간 $[\alpha,\beta)$에서 $\beta-\alpha\geq 2\pi$이면 $[\alpha,2\pi)$와 $[0,\beta)$의 두 개의 각구간으로 분할한다.

질의 q와 각도 θ에 대한 반직선 r은 극좌표계에서 다음과 같이 정의될 수 있다.

$$r(q,\theta)=\{q+\lambda(\cos\theta,\sin\theta)|\lambda\geq 0\}$$

각 r에서 가장 먼저 만나는 객체를 가시 객체(visible object)라고 한다.

$$VO(q,\theta)=arg\min_{o\in O} dist(q,r(q,\theta)\cap o_{MBR})$$

단, $r(q,\theta)$가 어떤 객체와도 교차하지 않으면 $VO(q,\theta)=\perp$이다.(여기서 $\perp$는 빈 각구간을 의미한다.) $NS(q)$를 처리하는 과정에서 객체 o의 가시각구간 집합은 다음과 같이 정의한다.

$$\theta(q,o)=\{[\alpha_k,\beta_k)|k=1,\ldots,m\}$$

여기서 각 $[\alpha_k,\beta_k)$는 모든 $\theta\in[\alpha_k,\beta_k)$에 대해 $NS(q,\theta)=o$가 성립하는 최대 연속각구간이다. 즉, $\theta(q,o)$는 객체 o가 질의 q에서 가시 객체로 관찰되는 모든 분리된 각구간의 집합이다. 따라서 NS 결과와의 관계는 다음과 같이 정의된다.

$$NS(q)=\{\langle o,I\rangle|o\in O\cup\{\perp\}\}$$

여기서 각구간 $I=[\alpha,\beta)$이다. $NS(q)$의 결과는 서로 겹치지 않으며, 전체 합집합은 $[0,2\pi)$이다.

따라서 객체 o가 q에서 NS로 차지하는 각구간의 집합은 다음과 같이 정의할 수 있다.

$$\theta(q,o)=\{I|\langle o,I\rangle\in NS(q)\}$$

즉, $\theta(q,o)$는 객체 o가 q에서 가시적인 모든 서로 다른 각구간의 집합이다. 여기서 특정 시점의 q에서 객체 o에 대한 순간 가시각구간은 다음과 같이 정의한다.

$$VAng(q,o)=\sum_{I\in\theta(q,o)}|I|$$

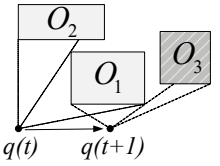
$VAng(q,o)$의 정규 가시각률 $NAng(q,o)$는 다음과 같다.

$$NAng(q,o)=\frac{VAng(q,o)}{2\pi}$$

따라서 $MAE(q)$는 위치 q에서 정규 가시각률 점수가 가장 큰 객체이므로 다음과 같이 정의된다.

$$MAE(q)=arg\max_{o\in O} NAng(q,o)$$

예를 들어 [Fig. 2]와 같이 어떤 위치 $q(t)$에서 $NS(q)$의 결과가 $\{\langle\perp,[0,\pi/8)\rangle,\langle o_1,[\pi/8,\pi/3)\rangle,\langle o_2,[\pi/3,\pi/2)\rangle,\langle\perp,[\pi/2,2\pi)\rangle\}$ 일 때, o_1은 $[\pi/8,\pi/3)$의 각구간에서 보이므로 $VAng(q,o_1)=5\pi/24$이다. o_2는 $[\pi/3,\pi/2)$를 차지하므로 $VAng(q,o_2)=\pi/6$이다. $\perp$ 구간은 빈 공간이므로 $MAE(q)$ 결과 후보에 포함하지 않는다. 이동 후 다음 위치 $q(t+1)$에서 o_1의 각구간이 줄고 o_3의 각구간이 새로 생긴다면, 전체 객체 집합을 다시 집계하는 대신 변경된 튜플에 포함된 o_1, o_3의 누적각구간률 점수(score)만 갱신할 수 있다.



[Fig. 2] $VAng(q,o)$ and $NAng(q,o)$

이동 구간 T에서 각 이벤트 시점마다 $NAng(q_i,o)$를 객체별로 누적한다. 따라서 이동 구간 누적 가시각점수 $Score$는 다음과 같이 정의할 수 있다.

$$Score_T(o) = \sum_i \Delta t_i \bullet NAng(q_i, o)$$

$Score_T(o)$를 이용하면 $CMAE(T)$는 다음과 같이 이동 구간 T에서 누적 가시각률 점수가 가장 큰 객체를 구하는 문제로 정의할 수 있다.

$$CMAE(T) = arg \max_{o \in O} Score_T(o)$$

동일한 최대 점수를 갖는 객체가 둘 이상이면 객체 식별자가 가장 작은 객체를 반환한다.

4. 제안 기법

본 논문에서 제안하는 기법은 스냅숏 처리(snapshot processing)와 연속 처리(continuous processing)를 분리한다. 스냅숏 처리는 현재 위치 q에서 정확한 $NS(q)$ 각구간 집합을 만들고, 그 각구간의 크기를 객체별로 집계하여 $MAE(q)$를 찾는다. 반면 연속 처리는 이동 과정 T에서 매번 전체 $NS(q)$를 NPP를 이용하여 재계산하지 않고, 현재 가시 객체의 가시각률의 변화를 추적하고 누적 가시각률 점수를 갱신한다.

4.1 Snapshot-MAE

R*-tree의 엔트리 e에 대해 질의 q에서 차지할 수 있는 각구간을 다음과 같이 정의할 수 있다[2,15].

$$AB(q,e) = [\theta^-_{q,e}, \theta^+_{q,e})$$

각구간은 Sweep 방식에서 R*-tree 접근 순서를 결정하는 데 사용된다. $madist(q, R, \alpha)$는 각도 α에서 질의 q로부터 MBR R의 가장 가까운 경계점까지의 거리를 나타낸다. 만약 $\alpha \notin AB(q, R)$이면, 해당 각도에서 R은 후보가 아니므로 $madist(q, R, \alpha) = \infty$이다. 이제 각구간 I에 대해 다음 각 기반 거리를 정의할 수 있다.

$$\min\_madist(q, R, I) = \min_{\alpha \in I} madist(q, R, \alpha)$$

$$\max\_madist(q, R, I) = \max_{\alpha \in I} madist(q, R, \alpha)$$

여기서 $madist$, min_madist, max_madist는 가시 각구간이 아닌 특정 각구간에서 어떤 객체가 가시 객체가 될 수 있는지 판정하고 e를 가지치기(pruning)하기 위한 거리이다. 객체 o의 $\theta(q, o)$는 $NS(q)$가 완성된 뒤 튜플의 각구간 길이를 합산하여 계산한다

현재 $NS(q)$에서 각구간 I를 차지하는 객체를 o_{near}라 할 때, 어떤 e에 대해 다음 조건이 성립하면 e는 해당 각구간 I에서 o_{near}보다 가까운 가시 객체가 될 수 없다. 즉, 다음 조건이 만족하는 경우, e가 o_{near}보다 가까워질 가능성이 없으므로 제거할 수 있다.

$$min\_madist(q, e, I) > max\_madist(q, o_{near}, I)$$

가지치기되지 않은 R*-tree의 말단 노드에 포함된 객체는 기존 $NS(q)$ 튜플과 비교된다. ObjectCompare 함수는 두 객체의 각구간 거리를 비교하여 둘 중 한 객체가 전체 가시각 구간을 차지하는지, 아니면 분리된 가시 각구간을 나눠 차지하는지 결정한다. 기본 탐색은 MBR 기반 각구간과 $madist$를 사용한다고 가정한다.

Fig. 3의 Snapshot-MAE(q, O, R)은 MAE 질의의 처리 과정에 대한 알고리즘이다. 현재 위치 q에서 $NS(q)$를 먼저 구성한 뒤, 각 튜플의 각구간 길이를 객체별로 합산하여 $NAng(q,o)$를 계산하고 $MAE(q)$를 반환한다. 1-18행은 R*-tree 엔트리의 angular bound와 madist 기반 가지치기를 이용해 $NS(q)$를 갱신한다. 19-22행은 ⊥를 제외한 객체별 각구간 길이를 정규화하여 최대값을 선택하는 집계 단계이다.

```
Input : query point q, object set O, R*-tree R
Output: MAE(q), Υ=NS(q), NAng

1  NS(q) ← {⟨⊥,[0,2π)⟩}
2  PQ ← root entries of R ordered by angular start
3  while PQ is not empty do
4       e ← PQ.pop()
5       B ← AB(q,e)
6       for each ⟨o_old,I_old⟩ in NS(q) overlapping B do
7            I ← I_old ∩ B
8            if o_old ≠ ⊥ and
                 min_madist(q,e,I) > max_madist(q,o_old,I)
9                 skip e over I
10           else if e is an index entry
11                Insert non-pruned child entries into PQ
12           else
13                Comp ← ObjectCompare(q,o_old,e,I)
14                split/replace NS(q) by Comp
15           end if
16      end for
17      merge adjacent tuples with same object
18 end while
19 for each object o appearing in NS(q) do
20      NAng(q,o) ← (Σ_{⟨o,I⟩∈NS(q)} |I|)/(2π)
21 end for
22 return argmax_o NAng(q,o), NS(q), NAng
```

[Fig. 3] Algorithm 1. Snapshot-MAE(q,O,R)

4.2 Continuous-MAE

연속처리에서는 현재 유지되는 $NS(q)$ 결과를 Υ로 둔다. 각 엔트리 $\eta \in \Upsilon$는 다음 정보를 가진다.

$$\eta = \langle o, I, optISP^{\nabla}, optISP^{\Delta} \rangle$$

여기서 o는 가시 객체이고, $optISP^{\nabla}$와 $optISP^{\Delta}$는 객체가 가시적인 것을 막는 역그림자객체이다. 이를 통해 NPP는 다음과 같이 정의한다.

$$\Sigma = NPP(q) = \bigcap_{\eta \in Y} \eta.optISP$$

질의 q'가 Σ 내부에 있으면 현재 Y 밖의 객체가 새롭게 가시 객체로 추가되지 않는다.

$$q' \in \Sigma \Rightarrow ActiveSet(q') \subseteq ActiveSet(q)$$

이 성질은 $MAE(q)$가 변하지 않음을 의미하지 않는다. $q' \in \Sigma$인 경우에도 기존 가시 객체의 각구간은 계속 변한다. 따라서 NPP 내부에서는 서버의 재질의 없이 CART/ART를 수행하여 Y의 각구간만 갱신하고, 그 결과를 누적가시각점수테이블(range ledger)에 누적한다. 즉 객체별 누적 점수 Score[o]와 현재 가시각률 CurrNAng[o]를 저장하는 테이블이다.

반대로 $q' \notin \Sigma$이면 현재 Y만으로 객체의 비가시성을 보장할 수 없다. 이때 전체 $[0, 2\Pi)$를 다시 탐색하지 않고, 위반된 역그림자객체를 찾은 뒤 재탐색이 필요한 각구간에 대해서만 부분 $NS(q)$를 수행한다.

$$q' \notin \Sigma \Rightarrow \exists\, \eta \in Y : q' \notin \eta.optISP^{\nabla} \vee q' \notin \eta.optISP^{\Delta}$$

부분 $NS(q)$는 Algorithm 1의 *madist* 기반 가지치기를 사용하되, 탐색 범위를 전체 각도 공간이 아니라 위반된 각구간으로 제한한다. 이 방식은 매 $MAE(q)$마다 전체 NS를 재수행하는 방식보다 R*-tree 노드의 접근과 ObjectCompare 횟수를 줄인다.

```
Algorithm 2. Continuous-MAE(q(t),T,O,R)

Input : moving query q(t), interval T=[ts,te], object set O, R*-tree R
Output: CMAE(T)

1  q ← q(ts), t ← ts
2  ⟨MAE(q), Υ,NAng⟩ ← Snapshot-MAE(q,O,R)
3  compute optISP for each η∈Υ
4  Σ ← ∩{η∈Υ} η.optISP
5  Ledger.Init(Υ,NAng)
6  while t < te do
7       ⟨q',t'⟩ ← next query point
8       Δt ← t' - t
9       if q' ∈ Σ then
10           Υ' ← CART/ART(Υ,q,q')
11           CurrNAng ← aggregate tuple lengths of Υ'
12           Ledger.Commit(Δt, Υ, Υ',CurrNAng)
13      else
14           Violated ← {η∈Υ | q'∉η.optISP}
15           Iinv ← compute invalid/covering arcs(Violated)
16           ChangeLog ← PartialSweep(q',Iinv,Υ,R)
17           Υ' ← apply ChangeLog to Υ
18           CurrNAng ← aggregate tuple lengths of Υ'
19           Ledger.Commit(Δt, Υ, Υ',CurrNAng)
20           recompute optISP and Σ for Υ'
21      end if
22      q ← q', t ← t', Υ ← Υ'
23 end while
24 return argmaxo Ledger.Score[o]
```

[Fig. 4] Algorithm 2. Continuous-MAE(q(t),T,O,R)

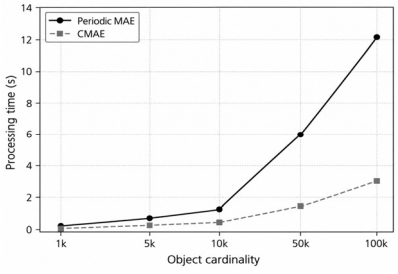
Fig. 4는 CMAE 질의의 처리 방식이다. 시작부에서만 완전 $MAE(q)$를 수행한 후 질의점이 NPP 내부에 있으면 기존 Y의 각구간만 갱신하고, NPP를 벗어나면 위반된 각구간에 대해서만 부분 탐색을 수행한다.

5. 성능 평가

제안 기법의 효율성을 검증하기 위한 시뮬레이션을 수행한다. 모든 시뮬레이션은 CPU Intel Core i9, RAM 32GB, NVIDIA Geforce RTX 4090, 소프트웨어는 Python 3.14를 사용하였다. 실험 공간은 1000×1000의 2차원 공간으로 설정하고, 객체는 MBR로 R*-tree에 인덱싱된다. 이동 샘플 수는 100개이다.

객체는 공간 내에서 균일 분포에 따라 생성하였고, MBR의 폭과 너비는 [0.2,1.0] 범위에서 무작위로 설정하였다. 이동 경로는 무작위 위치에서 100개로 구성하였으며, Incremental(Inc), Roam-Narrow(RN), Roam- Wide(RW) 패턴은 x 또는 y 좌표에 각각 [0,0.025], [-0.025,0.025], [-0.05,0.05] 범위의 변위를 적용하여 생성하였다.

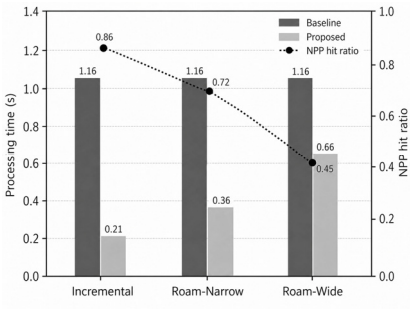
비교 대상 이동하는 질의에 대해 Periodic MAE는 모든 질의 이벤트마다 Snapshot-MAE를 수행하고, 각 $NS(q_i)$ 결과를 객체별로 집계한다. 반면 CMAE는 초기 위치에서만 $NS(q_0)$을 수행하고, 이후 $q' \in \Sigma$이면 CART/ART와 누적가시각점수테이블만 수행하며, $q' \notin \Sigma$이면 무효 각구간(invalid arc)에 대해서만 부분 탐색을 수행한다.



[Fig. 5] Processing Time by Object cardinality

Fig. 5는 객체 수가 증가할수록 Periodic MAE의 처리 시간이 급격히 증가하는 반면, 제안 CMAE는 NPP 내부 이동에서는 ART와 누적가시각점수테이블 갱신만 수행하고 NPP 이탈 시에도 무효 각구간에 대해서만 부분 탐색을 수행하므로 증가폭이 완만함을 보인다.

Fig. 6은 이동 패턴별 처리 시간과 NPP 적중률을 나타낸다. Inc 및 RN 패턴은 인접 샘플 간 이동 거리가 작아 NPP 적중률이 높으므로, 대부분의 이동 이벤트가 서버 측 PartialSweep 없이 ART와 Range Ledger 갱신으로 처리된다. 반면 RW는 NPP 이탈 빈도가 증가하여 제안 기법의 처리 시간이 증가하지만, 이탈 시에도 전체 $[0, 2\pi)$가 아니라 무효 각구간만 탐색하므로 Periodic MAE보다 낮은 처리 시간을 유지한다.



[Fig. 6] Processing Time and NPP Hit Ratio by Different Query Movement Patterns

6. 결론

본 논문은 이동 질의점 $q(t)$에서 객체별 가시각구간 길이를 시간에 따라 누적하고, 누적 가시각률 점수가 가장 큰 객체를 반환하는 CMAE 질의를 제안하였다. 스냅숏 처리로 $MAE(q)$를 계산하고, 연속 처리에서는 NPP 이탈 시 위반 각구간에 대해서만 부분 $NS(q)$를 수행하여 반복적인 전체 $NS(q)$ 재수행 비용을 줄였다. 시뮬레이션 결과, 제안 기법은 객체 수와 이동 패턴 변화에도 Periodic MAE보다 낮은 처리 비용을 보였다. 향후 실제 3차원 지도와 실내외 객체 데이터셋을 이용하여 NPP 적중률, 무효 각구간 크기, 누적 가시각 점수표 갱신 비용이 성능에 미치는 영향을 정량적으로 검증할 계획이다.

REFERENCES

[1] S.Masud, F.M.Choudhury, M.E.Ali, and S.Nutanong, "Maximum Visibility Queries in Spatial Databases," Proceedings of the 29th IEEE International Conference on Data Engineering, pp.637-648, 2013.

[2] K.C.K.Lee, W.-C.Lee, and H.V.Leong, "Nearest Surrounder Queries," Proceedings of the 22nd IEEE International Conference on Data Engineering, p.85, 2006.

[3] X.Guo, B.Zheng, Y.Ishikawa, and Y.Gao, "Direction-Based Surrounder Queries for Mobile Recommendations," The VLDB Journal, Vol.20, No.5, pp.743-766, 2011.

[4] K.C.K.Lee, J.Schiffman, B.Zheng, W.-C.Lee, and H.V.Leong, "Round-Eye: A System for Tracking Nearest Surrounders in Moving Object Environments," Journal of Systems and Software, Vol.80, No.12, pp.2063-2076, 2007.

[5] J.Chung, "Spatial Query Processing for Mobile Context- and Location-Based Services," Ph.D. Dissertation, Korea University, 2011.

[6] Z.Song and N.Roussopoulos, "K-Nearest Neighbor Search for Moving Query Point," Advances in Spatial and Temporal Databases, SSTD 2001, LNCS Vol.2121, pp.79-96, 2001.

[7] Y.Tao, D.Papadias, and Q.Shen, "Continuous Nearest Neighbor Search," Proceedings of the 28th International Conference on Very Large Data Bases, pp.287-298, 2002.

[8] X.Yu, K.Q.Pu, and N.Koudas, "Monitoring k-Nearest Neighbor Queries over Moving Objects," Proceedings of the 21st IEEE International Conference on Data Engineering, pp.631-642, 2005.

[9] K.Mouratidis, D.Papadias, S.Bakiras, and Y.Tao, "A Threshold-Based Algorithm for Continuous Monitoring of k Nearest Neighbors," IEEE Transactions on Knowledge and Data Engineering, Vol.17, No.11, pp.1451-1464, 2005.

[10] A.Guttman, "R-Trees: A Dynamic Index Structure for Spatial Searching," Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, pp.47-57, 1984.

[11] S.Nutanong, E.Tanin, and R.Zhang, "Visible Nearest Neighbor Queries," Proceedings of the 12th International Conference on Database Systems for Advanced Applications, LNCS Vol.4443, pp.876-883, 2007.

[12] Y.Gao, B.Zheng, W.-C.Lee, and G.Chen, "Continuous Visible Nearest Neighbor Queries," Proceedings of the 12th International Conference on Extending Database Technology, pp.144-155, 2009.

[13] P.F.Fisher, "Algorithm and Implementation Uncertainty in Viewshed Analysis," International Journal of Geographical Information Systems, Vol.7, No.4, pp.331-347, 1993.

[14] G.R.Hjaltason and H.Samet, "Distance Browsing in Spatial Databases," ACM Transactions on Database Systems, Vol.24, No.2, pp.265-318, 1999.

[15] N.Beckmann, H.-P.Kriegel, R.Schneider, and B.Seeger, "The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles," Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, pp.322-331, 1990.

정 재 화(Jaehwa Chung) [종신회원]

- 2006년 8월 : 고려대학교 컴퓨터교육과 (이학석사)
- 2011년 8월 : 고려대학교 컴퓨터교육과 (이학박사)
- 2012년 3월 ~ 현재 : 한국방송통신대학교 컴퓨터과학과 교수

〈관심분야〉

시공간 데이터베이스, 위치기반 서비스, ANN 기반 음원분리 및 공간기반 음향생성