Nearest Neighbor Query Processing in the Mobile Environment

Hyun Mi Choi*
Proton Engineering Frontier Project, Korea Atomic Energy Research Institute
Dukjin-dong 150, Yusong-gu, Daejeon, 305-353, Korea
*{hmchoi}@kaeri.re.kr

Young Jin Jung**, Eung Jae Lee**, Keun Ho Ryu**
Database Laboratory, ChungBuk National University
Cheongju Chungbuk, 361-763, Korea
**{yjjeong, eungjae, khryu}@dblab.chungbuk.ac.kr

Abstract: In the mobile environment, according to the movement of the object, the query finds the nearest special object or place from object position. However, because query object moves continuously in the mobile environment, query demand changes according to the direction attribute of query object. Also, in the case of moving of query object and simply the minimum distance value of query result, sometimes we find the result against the query object direction. Especially, in most road condition, as user has to return after reaching U-turn area, user rather spends time and cost.
Therefore, in order to solve those problems, in this paper we propose the nearest neighbor method considering moving object position and direction for mobile recommendation system.
Keywords: Nearest Neighbor Query, Location-Based Service

1. Introduction

Recent development in a wireless computing and Global Positioning System(GPS) technology causes the active development in the application using location management system in real-time environment such as transportation vehicle management, air traffic control and location based system. With this, we got more the present location information exactly by real time and developed various location information system using the location information.

The Location based Service(LBS) provides an various information related the position to grasp user's position: transportation vehicle management, aviation traffic control, location based service, etc, especially, vehicle location tracking system. To support the LBS effectively, it is essential to manage user's position provided in real time and to provide the information.

As nearest neighbor query retrieves the nearest located target objects to the requester, it is one of the queries used in the mobile environment very frequently. Because of in the mobile environment, the location of the query requester or data objects are continuously moving over time, it is necessary to require a new type of query processing such as the continuous query processing and the variation of the query processing towards moving direction. However, in existing spatial database, because the NN query calculates simply the distance of query/data objects, which returns the nearest located target objects, it is difficult to process such requirements in the mobile environment.

In the mobile environment, because the query/data objects are moving continuously, the query occurs over time. But the existing methods have some limitations for continuous nearest neighbor search. For example, the loading problem makes the query process system hard to process in a real time. Also, this query returns the ineffective result, because user position in the query request time does not correspond with that of the query result acquisition time. Therefore, the query result from previous work spends a lot of money and time.

In this paper, we propose a novel nearest neighbor query processing technique that is able to retrieve nearest located target object from the user who is continuously moving with a direction. The proposed method retrieves objects using the direction property of moving object as well as Euclidean distance to target object.

The rest of this paper is structured as follows: Section 2 outlines existing methods for processing NN, and Section 3 describes a characteristic and data model. Section 4 describes our proposed techniques. Section 5 experimentally evaluates our techniques and then concludes this paper and future work presents in Section 6.

2. Related Work

Roussopoulos et al[1] proposed the method using R-tree index for nearest neighbor search quickly. The method presents an efficient branch-and-bound R-tree traversal algorithm to find the nearest object to a point, and then generalize it to finding the k nearest neighbors

The previous nearest neighbor query processing required much money using previous query processing in the spatial database, because query objects change their position continuously over time. Recently Kollios[2], Benetis[3] and Tao at al[4] proposed the query processing method of new type using moving object property in the mobile environment.

This paper proposed the Direction-based Nearest Neighbor(DNN) query processing using moving property of query object and proposed the Quantized direction-based Nearest Neighbor(QDNN) using the direction property of moving object to correct a detailed direction error.

3. Mobile Object Modeling

1) Characteristic of Mobile Object

Mobile objects are spatio-temporal objects which change their position continuously over time. Moving type of the mobile object can divide into two types; the free trajectory and the constrained trajectory. Based on the following assumptions, we are modeling the position information of mobile object in this paper.

1. The mobile object is a target of moving point object from two dimension spaces.

2. The query object is continuously moving at present and we regard the position change process as the free trajectory.

3. The position information management method of mobile object is based on the observation driven system using GPS, Beacon etc.

4. The query object performs the query process to search fixed target object from moving object in a particular time in the mobile environment.

2) Data Model

The proposed method is about two kinds of objects; one is point objects that change only their position continuously over time (e.g., car, people, ship etc.). And the other is fixed objects that have specific position (e.g., gas station, hospital, banks etc.). The data structure to save into the database and the operation process method changes according to modeling method of moving position information. Therefore, the mobile object is defined as follows:

Definition 1. The mobile object is the point object and defines as $MV =< T_A, S_A, G_A >$.

Definition 2. The time attribute of mobile object is composed of valid time interval and defined by $T_A =< vt_s, vt_e >$.

Definition 3. The spatial attribute presents a position coordinate value in specific time of moving vehicle and defines as $S_A =< x, y >$.

Definition 4. The general attribute, G_A is defined as an attribute by user.

4. Nearest Neighbor Query using Direction

1) Direction-based Nearest Neighbor Query

The DNN Query is the method to process the nearest neighbor query processing using the direction of mobile object. This query processing converts the direction of mobile object into the weight, and then the query decides the ranking of final target object using multiplying Euclidean distance by weight.

θ_{diff}, the angle difference between the mobile object and the search target object is calculated by following a formula 1, case ($\theta_{diff} > 180°$) is adjusted to get value of ($0 \le \theta_{diff} \le 180°$) by formula 2.

$$\theta_{diff} = |\theta_{target} - \theta_{mo}| \quad (1)$$

$$\theta_{diff} = 360° - \theta_{diff} \quad (2)$$

Following Table 1 shows notations defined for weight calculation.

Table 1. Notations for Weight calculation

Notation	Descriptions
θmo	the angle value for moving direction of mobile object
θtarget	the angle of search target object as the center the mobile object
θdiff	the angle difference between the mobile object and the target object
Wmin	the minimum value of weight
Wmax	the maximum of weight
Wdiff	The difference between maximum weight(Wmax)and minimum weight(Wmin)

The weight for direction information is calculated by θ_{diff}, and θ_{diff} between (0°~180°) is calculated by following formula (3).

$$Weight = 1 - \frac{(180 - \theta_{diff})}{180} \quad (3)$$

The range of weight is ($0 \le Weight \le 1$), and weight is generalized formula (3) to get between W_{min} and W_{max}. It is calculated by the following formula (4).

$$Weight = W_{max} - \frac{(180 - \theta_{diff})}{180} \times W_{diff} \quad (4)$$

Therefore, the nearest neighbor query using direction information is processed to multiply previous calculated weight by *Euclidean distance.*

The nearest neighbor query processing using direction information algorithm is below:

Table 2. Algorithm of DNN Query Processing. (continue)

```
Algorithm findNearest(moID, ts)
input  moID   : any mobile object,  ts : qeury time
output List  NL : nearest neighbor query list
Begin
Step 1. Search the position of moID position and target from query
        time ts
        1.1  moID position : (moID, ts, moX, moY, moDirection,
                              moSpeed)
        1.2  target position : (targetID, targetX, targetY)
Step 2. Calculate the distance between moID position and target
        position
```

Step 3. Search the direction information of moID information and target information
3.1 Search moving direction angle about moID position
3.2 Calculate an angle about target position using trigonometric function
Step 4. Calculate the weight with direction information of step 2
4.1 if (θ_{mo} > θ_{target}) then
// if mobile object' processing direction angle is bigger than target object angle
$\theta_{diff} = \theta_{mo} - \theta_{target}$
else then // Otherwise, the target is bigger than mobile object's moving direction angle
$\theta_{diff} = \theta_{target} - \theta_{mo}$
if (θ_{diff} > 180) then // The angle is bigger than 180°
$\theta_{diff} = 360 - \theta_{diff}$ // angle difference(θ_{diff})
4.2 $Weight = W_{max} - ((180 - \theta_{diff}) / 180) * W_{diff}$
Step 5. Make nearest neighbor query list based distance and weight
5.1 *Estimation = Distance * Weight*
5.2 Add *Estimation* to NL
5.3 Sort NL with *Estimation* value
Step 6. Return NL
End

2) Quantized Direction-based Nearest Neighbor Query

QDNN query processing method is the extended DNN query processing to overcome an error according to process direction of mobile object. For example, in case of quantization as 30° interval, an angle difference between (0°~30°) have the same weight.

In this case, the quantized angle is QD, and divided number of quantized angle is defined as QN. They are calculated by formula (5) and (6) respectively.

$$QD = \frac{180}{QN} \tag{5}$$

$$QN = \frac{180}{QD} \tag{6}$$

The weight calculation by the quantized angle using formula (6) is calculated by following formula.

$$Weight = 1 - \frac{180 - (\theta_{diff} - QD)}{QN} \tag{7}$$

Here, the range of the weight value is ($0 \le Weight \le 1$), and if the weight is generalized to get between W_{min} and W_{max}, we can lead a formula (8).

$$Weight = W_{max} - \frac{QN - \lfloor \theta_{diff} / QD \rfloor}{QN} \times W_{diff} \tag{8}$$

The nearest neighbor query processing using quantized direction information algorithm is following:

Table 3. Algorithm of QDNN Query Processing.

Algorithm *findQNearest*(moID, ts)
input moID : any mobile object, ts : query time
output List NL : nearest neighbor query list
Begin
Step 1. Search the position of moID position and target from query time ts
1.1 moID position : (moID, ts, moX, moY, moDirection, moSpeed)
1.2 target position : (targetID, targetX, targetY)
Step 2. Calculate the distance between moID position and target position
Step 3. Search the direction information of moID information and target information
3.1 Search moving direction angle about moID position
3.2 Calculate an angle about target position using trigonometric function
Step 4. Calculate the weight with direction information of step 2
4.1 if (θ_{mo} > θ_{target}) then
// if mobile object' processing direction angle is bigger than target object angle
$\theta_{diff} = \theta_{mo} - \theta_{target}$
else then // Otherwise, the target is bigger than mobile object's moving direction angle
$\theta_{diff} = \theta_{target} - \theta_{mo}$
if (θ_{diff} > 180) then // The angle is bigger than 180°
$\theta_{diff} = 360 - \theta_{diff}$ // angle difference(θ_{diff})
4.2 $Weight = W_{max} - ((QN - (\theta diff / QD)) / QN) * W_{diff}$
Step 5. Make nearest neighbor query list based distance and weight
5.1 *Estimation = Distance * Weight*
5.2 Add *Estimation* to NL
5.3 Sort NL with *Estimation* value
Step 6. Return NL
End

5. Experiments

In this Section, we implemented the proposed nearest neighbor query processing for the mobile system. We use the Java JDK 1.3 and Microsoft SQL Server 2000 for vehicle's general attribute data management.

In the experiment, we used the following table structure of mobile object database. The mobile object and the target object information by sampling according to specific time interval that follows to save and to manage by two tables.

The mobile object information table, moHistory get an attribute value like table 4.

Table 4. Table of Mobile Object

attribute	moID	ts	moX	moY	moDiretion	moSpeed
data type	int	string	double	double	int	int
description	key	query valid time	x coordinate	y coordinate	direction (°)	Speed (km/h)

Table 5. Table of Target Object

Notation	tartgetID	targetX	targetY
data type	int	double	double
description	key	x coordinate	y coordinate

Table 5 is maintained to perform moving object and related query for target object. The direction for target object can be calculated an angle information using trigonometric function about target object's position information.

We showed the query result by proposed DNN query processing method in the fig. 1.

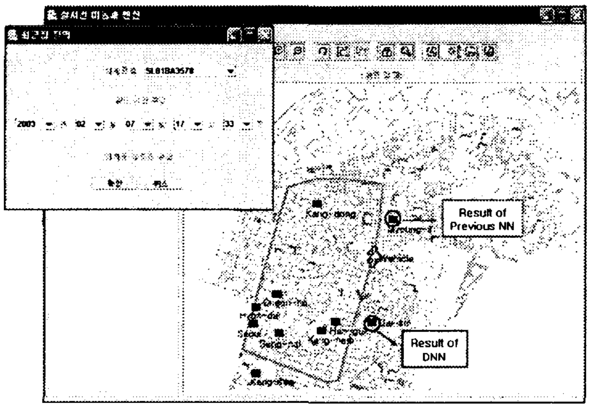

Fig. 1. Result of DNN Query Processing

The nearest oil station from vehicle is "Myoung-il" at query time in the fig. 1. However, we can select "Gai-sin" from DNN query processing.

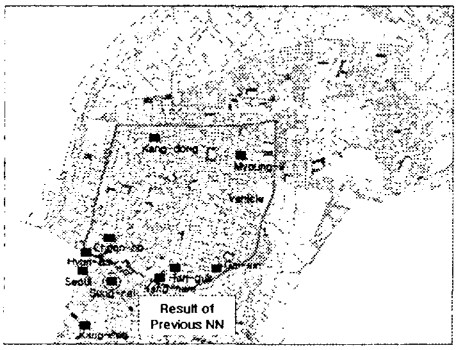

Fig. 2. Moving of Other Type's Mobile Object

The fig. 2 have showed moving trajectory of other type's mobile object.

In the case fig. 2, the result generated by proposed query processing method is like Table 6.

Table 6. Result of Query Processing

Oil station name	Real distance	θ_{diff}	W_{DNN}	$DNN_{distance}$	WQ_{DNN}	$QDN_{ditance}$
Sung-nai	76.8981	53	0.65	49.9837	0.63	48.4458
Kang-nam	112.2309	177	0.99	111.1086	0.96	107.746
Cheon-ho	121.2312	30	0.59	71.5264	0.59	71.5264
Hyun-dai	156.7305	2	0.51	79.9325	0.50	78.3652
Seoul	157.3261	23	0.57	89.6758	0.55	86.5293
Han-guk	160.2743	160	0.95	152.0000	0.92	147.4523
Kang-seo	235.8452	73	0.71	167.4501	0.67	158.0163
Gai-sin	286.7212	161	0.95	272.3851	0.92	263.7835
Kang-dong	440.5295	83	0.73	321.5865	0.71	312.7759
Myoung-il	525.2683	117	0.83	435.9727	0.80	420.2146

By the previous nearest neighbor query result, five nearest query lists are [Sung-nai, Kang-nam, Cheon-ho, Hyun-dai, Seoul] like table 6. However, DNN/QDNN query processing can make the query result lists change into [Sung-nai, Cheon-ho, Hyun-dai, Seoul, Kang-nam]

6. Conclusion

Nearest neighbor query which is one of the query processing methods frequently used in the mobile environment retrieves nearest located target objects. Because query object is continuously moving over time, continuous query requirement occurs in according to the query requester's environment. And the query result can be different according to moving direction of query object.

In this paper, we proposed a novel nearest neighbor query processing technique.

The proposed method calculated the query requester's direction information as weight to reflect the mobile object direction information in the query processing and applied it to the query processing. Also the mobile object has some characteristics that process in the same direction. In this time the direction error can occur. To modify it, during calculating the weight, we used the quantized direction to process the nearest neighbor query.

The proposed method is applicable to the application system such as traffic information system, travel information system, and location-based recommendation system which require retrieving nearest located object. In the future, we will consider the mobile object's direction attribute as well as moving speed, road information.

References

[1] N. Roussopoulos, S. Kelley, F. Vincent, "Nearest Neighbor Queries", SIGMOD Conference 1995, pp 71 ~ 77

[2] G. Kollios, D. Gunopulos, V. J. Tsotras, "Nearest Neighbor Queries in a Mobile Environment", Spatio-Temporal Database Management 1999, pp. 119 ~ 213

[3] R. Benetis, C. S. Jensen, G. Karciauskas, S. Saltenis, "Nearest Neighbor and Reverse Nearest Neighbor Queries for Moving Objects", IDEAS 2002, pp. 44 ~ 45

[4] Y. Tao, D. Papadias, Q. Shen, "Continuous Nearest Neighbor Search", VLDB 2002, pp. 287 ~ 298