International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 3, Issue 9, September 2014

Privacy-Aware Monitoring Framework based Time Monitoring System for Moving Object

T.Shilpa[1], D.Rama Krishna Rao[2]

[1]Student, [2]Professor

[1]Geethanjali College of Engineering & Technology, Hyderabad, India.[2]Geethanjali College of Engineering & Technology, Hyderabad, India.

Abstract: In this paper we propose a privacy-aware monitoring (PAM) framework that addresses both privacy and security issues. The framework distinguishes itself from the existing work by being the first to holistically address the issues of location updating in terms of monitoring accuracy, efficiency, and privacy, particularly, when and how mobile clients should send location updates to the server. Based on the notions of safe region and most probable result, PAM performslocation updates only when they would likely alter the query results. Location tracking and updating are the fundamental issues in the moving objects.Object's moving efficiencyand location detection accuracy are taken into account in the project. The client sends the queries to the server based on the proper result. The server sends the reply to the client maintaining its accuracy as well as efficiency. Main strategies are to update the client request with the exact result. The advantage of this is the flexibility to optimize the efficiency and also to update location of the moving object.

Keywords: Moving Object, Location update, Safe Regions, Object index.location-dependent and sensitive, mobile applications.

I. INTRODUCTION

In the literature, very few studies on continuous querymonitoring are focused on location updates. Two commonlyused updating approaches are periodic update (every clientreports its new location at a fixed interval) and deviationupdate (a client performs an update when its location orvelocity changes significantly) [1], [2], [3], [4]. However,these approaches have several deficiencies. First, themonitoring accuracy is low: query results are correct onlyat the time instances of periodic updates, but not in betweenthem or at any time of deviation updates. Second, locationupdates are performed regardless of the existence of queries a high update frequency may improve the monitoring accuracy, but is at the cost of unnecessary updates and query re-evaluation. Third, the server workload using periodic update is not balanced over time: it reaches the peak when updates arrive (they must arrive simultaneouslyfor correct results) and trigger query re-evaluation, but is idle for the rest of the time. Last, the privacy issue is simply ignored by assuming that the clients are always willing to provide their exact positions to the server.Fig. 1 shows a typical monitoring system, which consists of a base station, a database server, application servers, and a large number of moving objects (i.e., mobile clients).

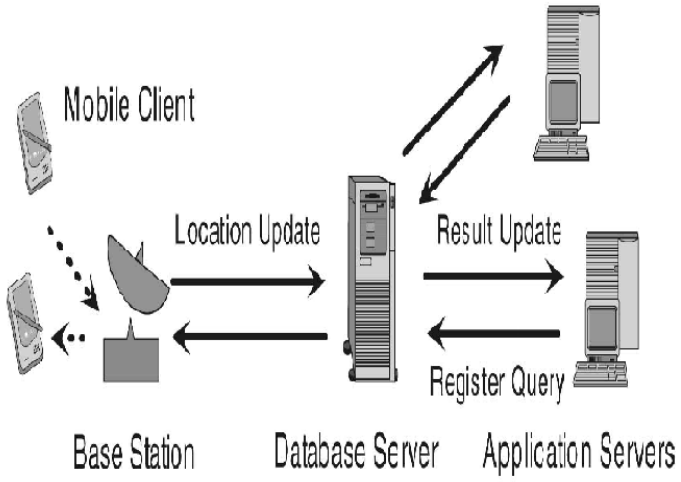


Fig. 1.The system architecture.

The database server manages the location information of the objects. The application servers gather monitoring requests and register spatial queries at the database server, which then continuously updates the query results until the queries are de-registered.There is a large body of research work on spatial temporal query processing. Early work assumed a static data set and focused on efficient access methods (e.g., R-tree [5]) and query evaluation algorithms (e.g., [6], [7]). Recently, a lotof attention has been paid to moving-object databases,where data objects or queries or both of them move.Assuming that object

movement trajectories are known a priori, Saltenis et al. [8] proposed the Time-ParameterizedR-tree (TPR-tree) for indexing moving objects, where thelocation of a moving object is represented by a linearfunction of time. Benetis et al. [9] developed query evaluation algorithms for NN and reverse NN search basedon the TPR-tree. Tao et al. [10] optimized the performanceof the TPR-tree and extended it to the TPR_-tree. Chon et al.[10] studied range and kNN queries based on a grid model.Patel et al. [11] proposed a novel index structure calledSTRIPES using a dual transformation technique.The work on monitoring continuous spatial queries canbe classified into two categories. The first category assumesthat the movement trajectories are known. Continuous Knn monitoring has been investigated for moving queries overstationary objects [12] and linearly moving objects [13], [14].Iworks et al. [13] extended to monitor distance semi joins for two linearly moving data sets [15]. However, as pointed outin [16], the known-trajectory assumption does not hold formany application scenarios (e.g., the velocity of a carchanges frequently on road).The moving objects have connection with thedatabase server. Furthermore, the communicationcost for any location update is a constant. With thelatter assumption, minimizing the cost of locationupdates is equivalent to minimizing the totalnumber of updates.

II. SYSTEM DESIGN MODEL

A. fundamentals of pam framework

In this paper, we assume that the clients are privacyconscious. That is, the clients do not want to expose their genuine point locations to the database server to avoidspatiotemporal correlation inference attack, by which an adversary may infer users' private information such aspolitical affiliations, alternative lifestyles, or medical problems.For example, knowing that a user is inside a heartspecialty clinic during business hours, the adversary caninfer that the user might have a heart problem. This has beencited as a major privacy threat in location-based services andmobile computing. To protect against it, most existing worksuggests replacing accurate point locations by boundingboxes to reduce location resolutions. With a large enough location box covering the sensitiveplace (e.g., the clinic) as well as a good number of otherinsensitive places, the success rate or confidence of suchspatiotemporal correlation inference can be reduced significantly.In our monitoring framework, we take the sameprivacy-aware approach. Specifically, each time a clientdetects his/her genuine point location, it is encapsulatedinto a bounding box. Then, the client-side location updaterdecides whether or not to update that box to the server.1Without any other knowledge about the client locations ormoving patterns, upon receiving such a box, the server canonly presume that the genuine point location is distributeduniformly in this box. Our problem is therefore tomonitor result changes of spatial queries as objects move,and monitor them as accurately as possible and at the lowestcost of location updates. As shown in Fig. 2, the PAM framework consists ofcomponents located at both the database server and themoving objects. At the database server side, we have themoving object index, the query index, the query processor,and the location manager. At moving objects' side, we havelocation updaters.

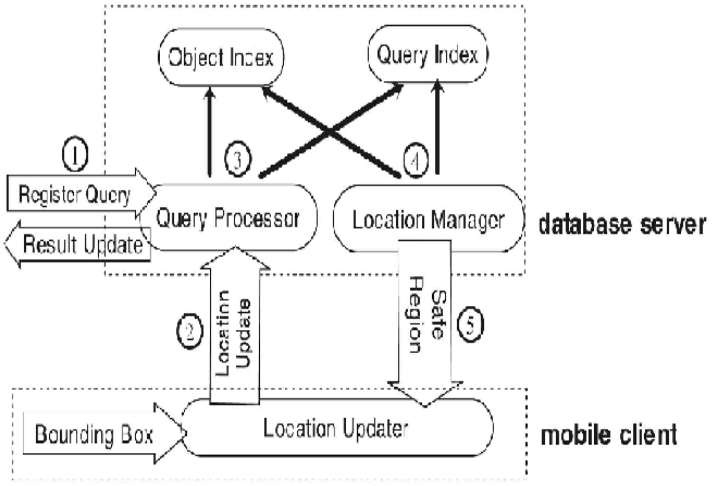


Fig. 2.PAM framework overview.

The Object Index

The object index is the server-side view on all objects. More specifically, to evaluate queries, the server must store the spatial range, in the form of a bounding box, within which each object can possibly locate. Note that this bounding box is different from a -square because its shape also depends on the client-side location updater. That is, it must be a function (denoted by δ) of the last updated δ-square and the safe region. As such, this box is called abbox as a mark of distinction. In particular, for the standard update strategy, the bbox is the safe region enlarged by δ/2 on each side, of the safe region and a δ/2

square. With the same rationale for which we assume the genuine point location of an updating object to distribute uniformly in the δ-square, we assume that the genuine point locations are distributed uniformly in their respective bboxes when queries are evaluated or reevaluated. The object index is built on the bboxes to speed up the evaluation. While many spatial index structures can serve this purpose, this paper employs the R*-tree index which is most widely adopted in the literature. Since the bbox changes each time the object updates, the index is optimized to handle frequent updates

The Query Index

For each registered query, the database server stores: 1) the query parameters (e.g., the rectangle of a range query, the query point, and the kvalue of a kNN query); 2) the current query results; and 3) the quarantine area of the query. The quarantine area is used to identify the queries whose results might be affected by an incoming location update. It originates from the quarantine line, which is a line that splits the entire space into two regions: the inner region and the outer region. An object becomes a result object if it enters the inner region; likewise, it becomes a nonresult object once it enters the outer region.

Query Processor and Location Manager

In the PAM framework, based on the object index, the query processor evaluates the most probable result when a new query is registered, or reevaluates the most probable result when a query is affected by location updates. Obviously, the re-evaluation is more efficient as it can be based on previous results.

The location manager computes the safe region of an Object. A safe region is a rectangle within which the change of the centroid of object's δ-square does not change the most probable result of any registered query. As queries are independent of each other, we can further define the safe region for a single query Q as a change of the centroid of object's δ-square does not change the most probable result of query Q.

Query Evaluation

The best-known algorithm to evaluate a kNN query q in Conventional euclidean space is the best-first search (BFS). It uses a priority queue H to store the to-be-explored index entries which may contain kNN's. The entries in H are sorted by their minimum distances to the query point q. BFS works by always popping up the top entry from H, pushing its child entries into H, and then, repeating the process all over. When a leaf entry, i.e., an entry of a leaf node, is popped, the corresponding object is returned as a nearest neighbor. The algorithm terminates if k objects have been returned.

B. System framework

ITS (Intelligent Transport System)Framework

The system framework comprises of an application server which is connected to a MobileClient, database and transport server that is used instead of GPRS. Mobile clients and transport server interact through application server.

Mobile clients have to register themselves with the application server. When mobile clients send request queries, all the queries are forwarded toapplication server and then it is given to transport server for processing. Transport server sends the result to application server which in turn sends to Mobile clients. There is a map that shows the moving objects which are nothing but buses along with the requested information. A monitoring framework was proposed where the clients were aware of the spatial queries being monitored, so they could send location updates only when the results for some queries might change. Basic idea there was to maintain a rectangular area, called safe region, for each object. The safe region is computed based on the queries in such a way that the current results of all queries remain valid as long as all objects reside inside their respective safe regions.A client updates its location on the server only when the client moves out of its safe region. This significantly improves the monitoring efficiency and accuracy compared to the periodic or deviation update methods. However, this framework fails to address the privacy issue, thatis, it only addresses when but not how the location updates are sent and also earlier work assumed a static data set and focused on efficient access methods and query evaluation algorithms.Recently, a lot of attention has been paid to moving-object databases, where data objects or queries or both of them move.

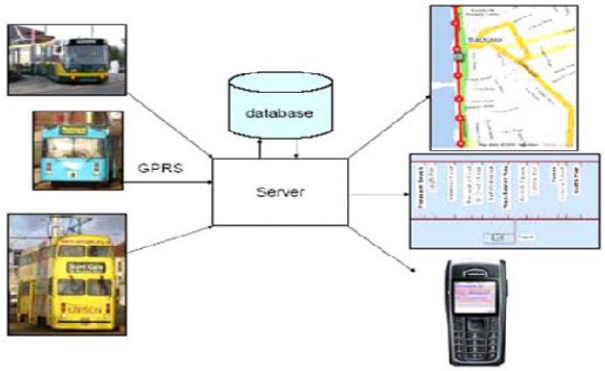


Fig 3.ITS Framework

C. System Architecture

Large systems are always decomposed into sub systems that provide some related set of services.The initial design process of identifying these sub systems and establishing a framework for sub-system control and communication is called architecture design and the output of this design process is a description of the software architecture.The architectural design process is concerned with establishing a basic structural framework for asystem. It involves identifying the major componentsof the system and communications between these components. The system architecture shows the blocks required for the project. Fig.2.2 shows the existing system architecture.

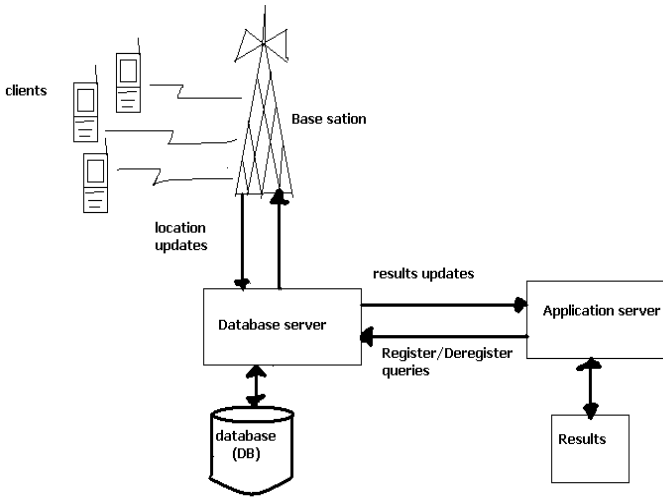


Fig 4.System architecture.

Some recent work attempted to remedy the privacy issue. Location cloaking was proposed to blur the exact client positions into bounding boxes. By assuming a centralized and trustworthy third-party server that store all exact client positions, various location cloaking algorithms were proposed to build the bounding boxes while achieving the privacy measure such as k-anonymity. However, the use of bounding boxes makes the query results no longer unique.As such, query evaluation in such uncertain space is more complicated. A common approach is to assume that the probability distribution of the exact client location in the bounding box is known and well formed. Therefore, the results are defined as the set of all possible results together with their probabilities. However, all these approaches focused on one time cloaking or query evaluation;they cannot be applied to monitoring applications where continuous location update is required and efficiency is a critical concern.

Query evaluation

In conventional euclidean space, a new range query is evaluated as follows: We start from the index root and recursively traverse down the index entries that overlap with the query window until the leaf entries storing the objects are reached. Then, we test each object using the containment relation in the new space.The best-known algorithm to evaluate a kNN query q in conventional euclidean space is the best-first search (BFS). It uses a priority queue H to store the to-be-explored index entries which may contain kNNs. The entries in H are sorted by their minimum distances to the query point q. BFS works by always popping up the top entry from H, pushing its child entries into H, and then, repeating the process all over. When a leaf entry, i.e., an entry of a leaf node, is popped, the corresponding object is returned as a nearest neighbor. The algorithm terminates if k objects have been returned.

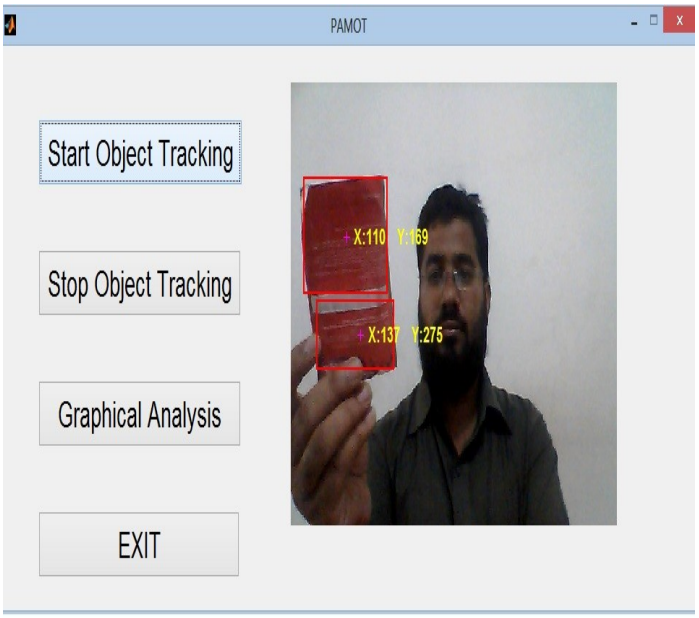
Algorithm 1:.**Evaluating a new kNN Query**
Input:*root*: root node of object index
q: the query point
Output:*C*: the set of kNNs
Procedure:
1: initialize queue H and *H;*
2: enqueue (*root, (q,root*)) into H;
3: while |C|< k and *H* is not empty **do**
4: *u*=*H*.pop();
5: if u is a leaf entry then
6: while d(*q,u*)>D(*q,v*) **do**
7: *v*=*H*.pop();
8: insert *v* to C;
9: enqueue *u* into *H*;

10: **else if** *u* is an index entry then
11: **for** each child entry *v* of *u* **do**
12: enqueue (*v*,*d*(*v, q*)) into *H*
To reevaluate an existing kNN query that is affected by the updating object p, the first step is to decide whether p is a result object by comparing p with the kth NN using the "closer" relation: if p is closer, then it is a result object; otherwise, it is a nonresult object. This then leads to three cases: 1) case 1:pwas a result object but is no longer so; 2) case 2:pwas not a result object but becomes one; and 3) case 3:pis and was a result object.

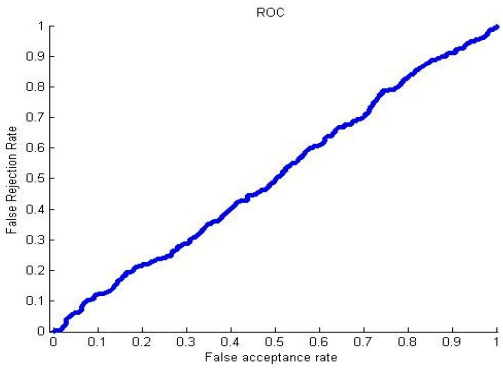
Algorithm 2.Re-evaluating a kNN Query
Input:*C*: existing set of kNNs
p: the updating object
Output:*C*: the new set of kNNs
Procedure:
1: if *p* is closer to the k-th NN then
2: if *p* ε *C* then
3: *p**=the rank of p in *C*;
4: **else**
5: *p**=k
6: enqueue p into *C*;
7: **else**
8: if *p* ε *C* then
9: evaluate 1NN query to find *u*;
10: *p**=k;
11: remove p and enqueue *u* into *C*;
12: relocate *p* or *u* in *C*, starting from *p**.

III. SIMULATION RESULTS

To evaluate the monitoring performance, we implement a simulation test bed, where N moving objects move within a unit-square space [0..1, 0..1]. Each object detects its point location at frequency f, encapsulates it into a _-square, and forwards the square to the location updater. Each object has an individual δ and it follows a normal distribution with mean value. We compare our PAM framework with two other frameworks, namely, the optimal monitoring (denotedas OPT) and the periodic monitoring (denoted as PRD).



In optimal monitoring, every object has the perfect knowledge of the registered queries and the δ-squares of other moving objects at any time. OPT serves as the lower bound for all monitoring frameworks.
In periodic monitoring, all objects periodically send out location updates simultaneously and the server reevaluates all registered queries based on these updates.



In the simulation test bed, each object moves according to the random way point mobility model: the client chooses a random point in the space as its destination and moves to it at a speed randomly selected from the range (0,2v) upon arrival or expiration of a constant movement period (randomly picked from the range (0, 2t*v*), it chooses a new destination and repeats the same process. To eliminate the effect from hardware configuration, the simulation uses logical time units instead of clock time

units. Each simulation run lasts for 5,000 time units or until the measured value stabilizes.

IV. CONCLUSION

This project focuses on providing an economical solution for public transport which is made available to people at different levels. The integrated architecture follows the trend of Intelligent Transportation System (ITS). Since the third party service provider is provided by usingthe mobile phone network, the operational cost can be reduced, giving a successful economical real-time solution. The framework is the first to holistically address the issue of location updating with regard to monitoring accuracy, efficiency, and privacy.We provide detailed algorithms for query evaluation/reevaluation and safe region computation in this framework.We also devise three-client update strategies that optimize accuracy, privacy, and efficiency, respectively. The performance of our framework is evaluated through a series of experiments. The results show that it substantially outperforms periodic monitoring in terms of accuracy and CPU cost while achieving a close-to-optimal communication cost. As the future work we can plan to further optimize the performance of the framework. In particular, the minimum cost update strategy shows that the safe region is a crude approximation of the ideal safe area, mainly because we separately optimize the safe region for each query, but not globally..

REFERENCES

[1] C.S. Jensen, D. Lin, and B.C. Ooi, "Query and Update Efficient B+-Tree Based Indexing of Moving Objects," Proc. Int'l Conf. Very Large Data Bases (VLDB), 2004.
[2] M.F. Mokbel, X. Xiong, and W.G. Aref, "SINA: Scalable Incremental Processing of Continuous Queries in Spatio-Temporal Databases," Proc. ACM SIGMOD, 2004.
[3] S. Prabhakar, Y. Xia, D.V. Kalashnikov, W.G. Aref, and S.E. Hambrusch, "Query Indexing and Velocity Constrained Indexing: Scalable Techniques for Continuous Queries on Moving Objects," IEEE Trans. Computers, vol. 51, no. 10, pp. 1124-1140, Oct. 2002.
[4] X. Yu, K.Q. Pu, and N. Koudas, "Monitoring k-Nearest Neighbor Queries over Moving Objects," Proc. IEEE Int'l Conf. Data Eng. (ICDE), 2005.
[5] A. Guttman, "R-Trees: A Dynamic Index Structure for Spatial Searching," Proc. ACM SIGMOD, 1984.

Authors Profile:

T.SHILPA is Pursuing M. Tech From Geethanjali College Of Engineering & Technology, JNTUH with specialization in Electronics & Communications Engineering (ECE). She completed her B.Tech from Anurag college of engineering, JNTUH with Specialization in ECE in the year 2012.

Duggirala Ramakrishna Rao graduated in Engineering (B.E. in ECE) from Government College of Engineering (Presently College of Engineering, JNTUK), Kakinada, Completed Post Graduation (M.Tech in Electronic Instrumentation) from Regional Engineering College (Presently NIT Warangal), Warangal. He has vast experience in the fields of Image Processing, Remote Sensing Radars and LIDARs. He has published and presented about 40 technical papers at the national and international level journals and conferences. He worked for the prestigious Indian Space Research Organization (ISRO) about 35 years and was the Deputy Project Director of Space Borne Lidar Project in ISRO. Presently he is the Professor in ECE department and Dean R & D of Geethanjali College of engineering and Technology. He is the Fellow of IETE and senior member of ISTE.

