Abstract

Spatial Alarms are personalized Location Based service(LBS), that are triggered by a specific location of a moving user, instead of time. In this paper, we introduce an efficient algorithm to evaluate spatial alarm queries in obstructed space. Existing work in this area has focused mainly on Euclidean distance and road network models. The key idea of our approach is to compute a specific region within which the answer set of our query remains unchanged. Our aim is to reduce redundant computations in client side, while preserving the accuracy of the alarms triggered.

Chapter 1

Introduction

The high availability of smart phones has led to the proliferation of location based services. Starting from static LBS(Location Based Service) such as finding the nearest pharmacy for a user's location, now-a-days LBSs are tailored for moving users. According to many, the next step in location based services is Spatial Alarms. Many believe this particular feature is going to dominate the future mobile-phone computing systems. Spatial alarms are an extension of time-based alarms. It is, however, triggered by a specific location irrespective of time."Remind me if I'm within 100 meters of a pharmacy" is a possible example of a spatial alarm. It is a personalized location based service which can vary from user to user.
Existing research has categorized spatial alarms into three types: public, shared and private. Public alarms are alarms which are active for every user within the system, such as an alarm must be sent to everyone within 100 meters of a building on fire. Private alarms are user defined alarms which are only viewable by the user herself, such as a user might set an alarm to alert her if she is within 100 meters of her favourite coffee shop. Shared alarms are shared between specific groups of people. In the previous example if a user chooses to share the alarm for the coffee shop with some of her friends it becomes a shared spatial alarm.

It is noteworthy that spatial alarms are quite dissimilar to spatial range query. Spatial alarms are based on a fixed location thus applying the techniques that are used in answering spatial range queries is both inefficient and

wasteful for the two dominating reasons, Firstly, in spatial range query as the user is the main point of interest, continuous re-evaluation of her location is needed in case of a mobile user. In contrast, spatial alarms need only be re-evaluated when the user in approaching a specific location. Secondly, in spatial alarm, the main point of interest is a static location. So the user's location is not relevant at all times. It is quite clear that applying spatial range query techniques in evaluating spatial alarms is going to result in wastage of resources. If we start to evaluate spatial alarms as soon as the user is on the move even if the user is far away from her desired location our efforts will be futile. Existing work in this area has focused mainly on Euclidean distance and Road Network models [**?**],[**?**],[**?**]. However, Spatial alarm evaluation in the obstructed space is different than road network or Euclidean space as it considers the obstacles in the path to the location of alarm. It is better approximated by a pedestrian scenario while road networks are approximated by vehicle scenarios. A pedestrian's path is not limited by roads. However, a pedestrian is obstructed by various obstacles such as buildings or trees. Thus while calculating the distance from alarms, we have to consider the obstructed distance[**?**]. In this paper we propose a unique approach to evaluate spatial alarms in obstructed space. To the best of our knowledge this query has not been addressed in any existing research work.

Spatial alarms are location-based, user-defined triggers which will possibly shape the future mobile application computations. They are distinct from spatial range query and do not need immediate evaluation after the user has activated them.The spatial alarm evaluation strategies are judged based on two features, High accuracy and High system scalability. High accuracy refers to the quality that guaranties no alarms are missed. And High scalability is the feature that ensures that the system can adapt to a large number of spatial alarms. In this paper, We propose a novel approach to evaluate spatial alarms in obstructed space which ensures both high accuracy and high scalability.

1.1 Problem Setup

Existing research has categorized spatial alarms into three types: public, shared and private. Public alarms are alarms which are active for every user within the system, such as an alarm must be sent to everyone within 100 meters of a building on fire. Private alarms are user defined alarms which can be viewed by the user, such as a user might set an alarm to alert her if she is within 100 meters of her favorite coffee shop. Shared alarms are shared between specific groups of people. In the previous example if a user chooses to share the alarm for the coffee shop with some of her friends it becomes a shared spatial alarm. In [**?**] spatial alarms has been categorized into three different types: 1)moving subscriber with static target, 2) static subscriber with moving target 3) moving subscriber with moving target. In this paper we are only considering the first type, that is moving subscriber with static target. In [**?**] spatial alarms have been approximated by rectangular bounding box, in our approach we are considering the spatial alarm region as a circle with radius r.

Obstructed Space Path Problem [**?**] denotes the problem of finding the shortest route between two query-points in Obstructed Space where non-intersecting 2D polygons represent *obstacles* and where the route does not traverse through any obstacles. The length of the Obstructed route between points a and b is called the *Obstructed Distance* between a and b, denoted by $dist_o(p_i, q)$.

A **Spatial Alarm Query in Obstructed Space** is formally defined as follows: Given a moving query point q and a range r for an alarm, a Spatial Alarm Query returns $\forall p_i \in P = \{p_1, p_2, p_3 ... p_n\}$ which have $dist_o(p_i, q) < r$

1.2 Preliminaries

Spatial alarms are location-based, user-defined triggers which will possibly shape the future mobile application computations. They are distinct from spatial range query and do not need immediate evaluation after the user has activated them. The spatial alarm evaluation strategies are judged based on two features, correctness and scalability. Correctness refers to the quality that guaranties no alarms are missed. And scalability is the feature that measures the number of

POI's the system can adapt to. In this paper, We propose a novel approach to evaluate spatial alarms in obstructed space which ensures both high accuracy and high scalability.

We define three different type of regions: *Known Region,Reliable Region* and *Safe Region*

Known Region: We define two different known regions for the POIs and the obstacles. The region containing at least 1 POI is the known region for POI.
The region circulating the POIs known region containing none or single colliding obstacles is the known region for the obstacles. The set of obstacles and POIs within this region is known to the client. Both of the known regions are represented by a parabola whose focus point is the users location q,with the equation $y^2 = 4ax$ where $a = mr$ which are bounded by a straight line.

Reliable Region: Within which region, no further query to the server has to be done to compute a consistent set of answers, that is termed as a reliable region. The reliable region is also a parabolic region bounded by a straight line,where each bounding point of the parabolic curve is at a distance r from the known regions parabolic curve. $\forall p_i = (x, y)$ in the known region, $\forall p_j = (x_r, y_r)$ in the reliable region, $dist_E(p_i, p_j) \geq r$ By this definition no further queries to the server has to be done to compute a consistent set of answers. Because, for any $q = p_j dist_E(q, p_j) \geq r$ where p_j is a point on the boundary of the reliable region. And for all other points p_i inside the reliable region $dist_E(q, p_i) > r$

Safe Region: A safe region is the region located inside reliable region within which the answer set of POIs remains unchanged for a moving client. We will denote the radius of the safe region as r_{safe}. Given the user's previous location P_1 and the current location P_2, if $(P_1 - P_2) < r_{safe}$, then no recalculation is needed to compute the answer. If the safe region surpasses the reliable region at some points

Table 1.1: Symbol Table

Symbols	Meaning
P	Set of POIs
O	Set of Obstacles
q	Location of user
r	Alarming range
V_G	Visibility Graph
$dist_E$(p,q)	Euclidean distance between points p and q
$dist_O$(p,q)	Obstructed distance between points p and q
r_{safe}	Radius of safe region
m	Real number in the range $[2, \infty)$
n	Real number in the range $[1, \infty)$
θ_i	Angle between consecutive path segments
S	Users path history as a set of straight lines
(m_i, c_i, l_i)	A line with slope m_i, intercept c_i and length l_i

The key idea of our approach is to calculate a dynamic *safe region*, within which no computation has to be done to provide an accurate alarm trigger. We will use an R-tree structure to index both obstacles and POI's in our approach. Our spatial alarm processing system has two different modes for efficient and effective processing of spatial alarms,namely, Bandwidth saving mode and Computational Cost Saving mode.

1.3 Contributions

Our approach accounts for both accuracy and efficiency by focusing on (1) No alarms being missed in user's proximity (2) Avoiding wasteful computation in client side (3) minimizing data transfer between server and client. In summary the our main contributions are:

- We introduce an efficient algorithm for evaluation of spatial alarm queries in obstructed space.

- We provide a spatial alarm evaluation system for mobile user.
- We provide an algorithm to calculate a dynamically changing region to accurately evaluate spatial alarm queries without any computation.
- We provide an extensive experimental analysis to compare the accuracy of our approach with other naive approaches based on different parameters.

1.4 Thesis Organization

The next chapters are organized as follows. In Chapter 2, related works are discussed. In Chapter 3 and 4, we present a naive approach and our approach, respectively, to evaluate Spatial Alarm in Obstructed Space queries. Chapter 5 shows the experimental results for our proposed algorithm. Finally, in Chapter 6, we conclude with future research directions.

Chapter 2

Experimental Study

2.1 Experiment Setup

To test and support our theoretical approach to compute spatial alarms in the obstructed space efficiently, we have done extensive experiments using both real and synthetic data-sets. In this chapter, we are going to present the validation and comparison of the proposed approach against the naive approach regarding various effective parameters

In this section, the detailed setup of the experiments is described as per the following sub-sections

2.1.1 Data Set Used

We have used both real and synthetic data-sets to evaluate our solution. In case of real data-sets, we have used obstacles and point of interests (POIs) of Germany. The obstacle set has 30674 minimum bounded rectangles (MBRs) of railway lines (rrlines). In our experiment, we assume obstacles to be presented by MBRs, but our algorithm can handle any type of obstacles. The POI set has 76999 MBRs of hypsography data (hypsogr). We assume data-points to be endpoints of the hypsography data. In this way, the POIs and obstacles are in the same plane which allows us to simulate a real-life scenario. We do not allow intersections between POIs and obstacles, neither do we allow duplicate POIs or obstacles. In case of synthetic data-sets, we have generated the obstacles and POIs from the real datasets following a uniform distribution. Before using in the real experiment, we have always normalized both real and synthetic dataset

in a $10,000 \times 10,000 grid$.
In our experiments, we have assumed only one type of POIs. However, our approach also can handle multiple types of POI.
Prior to running the main algorithms, two separate R-trees are built to store the POIs and the obstacles respectively from the used data-sets.

2.1.2 Sample Query Generation

We have simulated the movement of the client randomly in the runtime. During the experiments, we have assumed that any movement-path of the client can be synthesized as piecewise-linear, i.e., a set of directed straight lines. In the explicit form of any straight line,$y = mx + c$, for any client position (x, y), we have randomized the slope m giving some value of c each time. The direction of the client along this new straight line is also randomized to be either in forward or backward along the path, with a bias towards the forward direction, as the real world user-movement with a definite source and destination usually proposes.
After determining the clients new position along this path, in the naive approach, the algorithm 1 directly queries the server for POIs and obstacles within the clients alarming range. Alternately, in the main approach, the algorithm 6 checks the region-crosses and queries the server if necessary.
The clients velocity is also randomized within a certain range to give a new position of the client along the current direction in the next iteration, which again generates a new server-query accordingly.
Meanwhile, the direction of the client is predicted in the main approach from the latest set of piecewise linear paths in the clients movement history. This prediction procedure has already been described in the Algorithm section.

2.1.3 Measurement of the performance parameters

In our experiments, we vary the following query parameters:
(i) Clients Velocity Range
(ii) the Alarm Range r

(iii) the size of data sets (synthetic data set)
. Table 2.1 summarizes the parameter values used in our experiments. In all experiments, we estimate I/O accesses and the query processing time to measure the efficiency of our algorithms. In each set of experiments, we run the experiment for 100 queries and present the average result.

Parameter	Values	Default
Clients Movement lenght(v) Range (km)	40,100,300,400	40
Alarm Range r	40, 60, 100, 150, 200	150
Synthetic data set size	5K, 10K, 15K, 20K	-

Table 2.1: Values of different query parameters used in our experiments

2.1.4 Implementation Language and Tools

The project of our experiment is implemented using C++ language and have been compiled, debugged and tested using Microsoft Visual Studio 2015 Enterprise edition with a full version student-license from DreamSpark.

2.1.5 PC configuration

We have run our experiments in three PCs of the following configurations:
1. Intel Core i5 2.9 GHz (Quad Core), 12 GB RAM
2. Intel Core i5 2.3 GHz (Quad Core), 4 GB RAM
3. AMD FX 6100 3.3 GHz (Hexa Core), 8 GB RAM
The average of these multiple runs is taken to measure and compare the performance of both the nave approach and our approach.

In Section 2.1.6, we compare the results of two approaches.

2.1.6 Comparison of Our Approach with Naive aproach

Effect of Clients movement length: Figure 2.1(a) 2.1(b)2.1(c) and 2.1(d) show processing time,server query, i/o (obstacle) and i/o(POI) respectively, for processing spatial alarm queries using naive approach and our approach. We observe that for both algorithms processing time and server query increase with the increase in length of user's movement. We also observe that in terms

of server query our approach performs almost 14times better than the naive approach. and incase of runtime our approach runs almost 3 times better than the naive approach. Incase of I/o our approach initially is higher than the naive approach but, in the average case our approach is better than the naive approach by approximately 10 times.

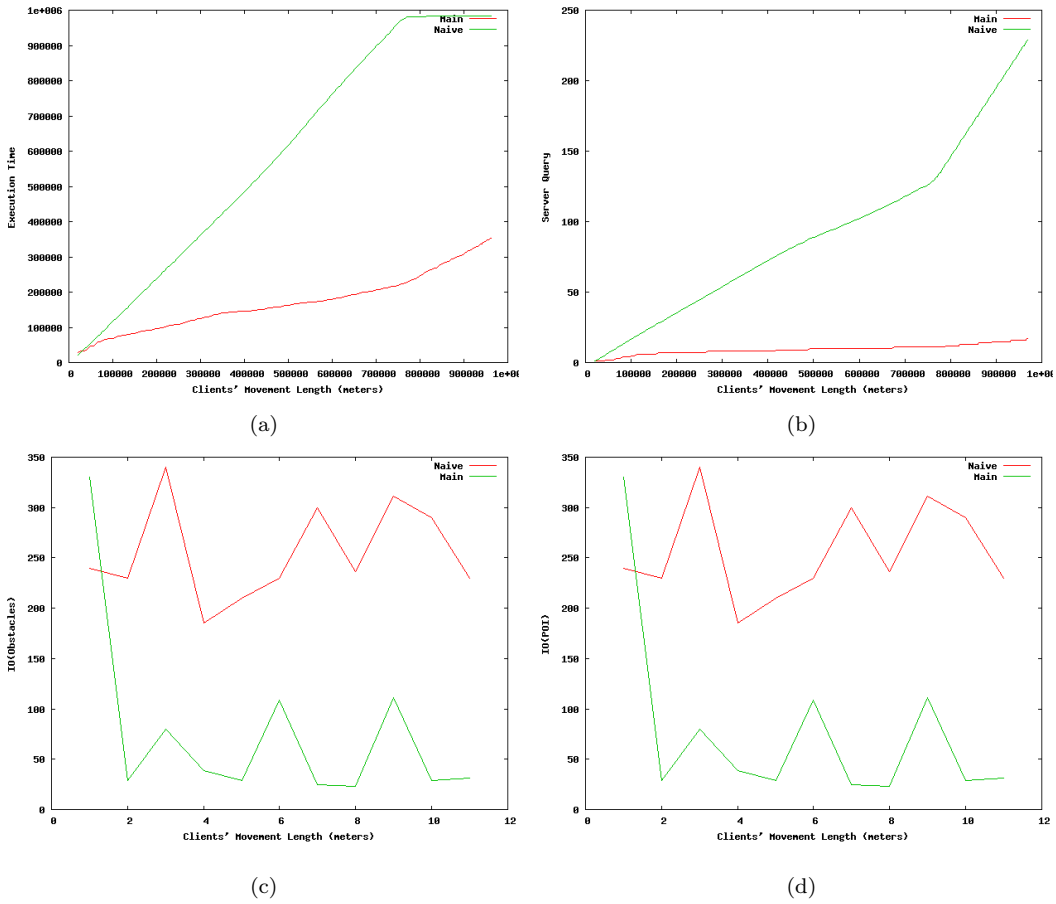

Figure 2.1: Effect of Client's movement length for Germany data (a)query processing time and (b)Server Query (c)IO-obstacles (d)IO-POI

Chapter 3

Conclusion

In this paper we have addressed evaluation of spatial alarm in obstructed space for the first time. We have proposed two variations of our approach to incorporate both accuracy and efficient communication bandwidth.Our experimental setup provides a comparative analysis between our approach and the naive approach on different parameters. The results of the experiment conducted shows that our proposed approach is better than the naive approach in execution time, IO access and number of server queries.

3.1 Future Directions

In the future we wish to extend our thesis work in several directions.

- In future, we aim to manipulate the geometrical properties of the regions to find a larger safe region.
- We wish to find a better prediction function for approximating the clients' next direction.
- In the future we wish to provide authentication and privacy while accessing the user's location.