Tracking Network-Constrained Moving Objects with Group Updates

Jidong Chen, Xiaofeng Meng, Benzhao Li, and Caifeng Lai

School of Information, Renmin University of China,
Beijing, 100872, China,
{chenjd, xfmeng, bzli, laicf}@ruc.edu.cn

Abstract. Advances in wireless sensors and position technologies such as GPS enable location-based services that rely on the tracking of continuously changing positions of moving objects. The key issue in tracking techniques is how to minimize the number of updates, while providing accurate locations for query results. In this paper, for tracking network-constrained moving objects, we first propose a simulation-based prediction model with more accurate location prediction for objects movements in a traffic road network, which lowers the update frequency and assures the location precision. Then, according to their predicted future functions, objects are grouped and only the central object in each group reports its location to the server. The group update strategy further reduces the total number of objects reporting their locations. A simulation study has been conducted and proved that the group update policy based on the simulation prediction is superior to traditional update policies with fewer updates and higher location precision.

1 Introduction

The continued advances in wireless sensors and position technologies such as GPS enable new data management applications such as traffic management and location-based services that monitor continuously changing positions of moving objects [2, 7]. In these applications, large amounts locations can be sampled by sensors or GPS periodically, then sent from moving clients to the server and stored in a database. Therefore, continuously maintaining in a database current locations of moving objects namely tracking technique becomes a fundamental component of these applications [1, 2, 9, 10]. The key issue is how to minimize the number of updates, while providing precise locations for query results.

The number of updates from moving objects to the server database depends on both the update frequency and the number of objects to be updated. To reduce the location updates, most existing works are proposed to lower the update frequency by a prediction method [1, 9, 10]. They usually use the linear prediction which represents objects locations as linear functions of time. The objects do not report their locations to the server unless their actual positions exceed the predicted positions to a certain threshold. This provides a general principle for the location update policies in a moving object database system.

However, few research works focus on improving the update performance from the aspect of reducing the number of objects to be updated. We observe that in many applications, objects naturally move in clusters, including vehicles in a congested road network, packed goods transmitted in a batch, animal and bird migrations. It is possible that the nearby objects are grouped and only one object in the group reports its location to the server to represent all objects within it. Considering real life applications, we focus on objects moving on a road network. Figure 1 gives an example of grouping vehicles on a part of road network. Due to the grouping of vehicles in each road segment, the total location updates sent to the server are reduced from 9 to 5.

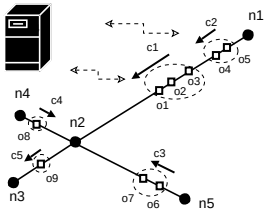

Fig. 1. Group location updates

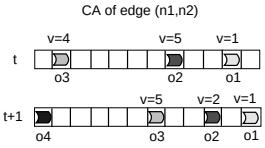

Fig. 2. A transition of the CA on an edge

The idea of grouping objects for location updates is similar to the GBL proposed in [6], but the GBL groups objects by their current locations and predicted locations after a time parameter τ. In fact, it obtains the predicted locations also by the linear prediction model assuming the linear movement with current velocity. However, in the urban road network, due to complex traffic conditions, cars may update their velocities frequently even for each timestamp. In this case, the linear prediction used in the GBL and other location update methods is inapplicable because the inaccurate predicted locations result in frequent location updates and lots of group management. In this paper, for the purpose of improving the performance of tracking for network-constrained moving objects, we focus on the both two factors affecting location updates and propose our solutions. One is a better prediction model to lower update frequency, and the other is a group update strategy to reduce the total number of objects reporting their locations. The accurate prediction model also reduces the maintenance of the groups and assures the location precision for querying.

Therefore, we first propose a simulation-based prediction (SP) model which captures traffic features in constrained networks. Specifically, we model road networks by graphs of cellular automata, which are also used to simulate vehicles future trajectories in discrete points in accordance with the surrounding traffic conditions. To refine the accuracy, we simulate two future trajectories to obtain the predicted movement function, which correspond to the fastest and slowest possible movements. We then propose a group location update strategy based on the SP model (GSP) to minimize location updates. In the GSP, for each edge in

the road network, the objects with their predicted movement functions similar are grouped or clustered and only the object nearest to its group center needs to report the location of the whole group. Within a certain precision, the locations of other objects can be approximated to their group location. Finally, through the experimental evaluations, we show that the GSP strategy has more efficient update performance as well as higher location precision.

The rest of the paper is organized as follows. Section 2 surveys related work by classifying the existing tracking techniques. In Section 3, a road network modeled as a graph of cellular automata is represented and our simulation-based prediction model is proposed. Section 4 describes our group update strategy. Section 5 contains an experimental analysis, and finally Section 6 concludes.

2 Related Work

Research on tracking of moving objects has mainly focused on location update policies. Existing methods can be classified according to the threshold, the route, the update mode or the representation and prediction of objects future positions.

Updates differ in threshold and route

Wolfson et al.[9] first proposed the dead-reckoning update policies to reduce the update cost. According to the threshold, they are divided into three policies, namely the Speed Dead Reckoning (SDR) having a fixed threshold for all location updates, the Adaptive Dead Reckoning (ADR) having different thresholds to different location updates and the Disconnection Detection Reckoning (DTDR) having the continuously decreasing threshold since last location update. The policies also assume that the destination and motion plan of the moving objects is known a priori. In other words, the route is fixed and known. In [4], Gowrisankar and Nittel propose a dead-reckoning policy that uses angular and linear deviations. They also assume that moving objects travel on predefined routes. Lam et al. propose two location update mechanisms for further considering the effect of the continuous query results on the threshold [7]. The idea is that the moving objects covered by the answers of the queries have a lower threshold, leading to a higher location accuracy. Zhou et al. [11] also take the precision of query results as a result of a negotiated threshold by the Aqua location updating scheme proposed.

Updates differ in representation and prediction of future positions

Wolfson and Yin [10] consider tracking with accuracy guarantees. They introduce the deviation update policy for this purpose and compare it with the distance policy. The difference between the two polices lies in the representation of future positions respectively with the linear function in the former and constant function in the latter. Based on experiments with artificial data generated to resemble real movement data, they conclude that the distance policy is outperformed by the deviation policy. Similarly, Civilis et al. [1, 2] propose three update policies: a point policy, a vector policy, and a segment-based policy, which differ in how they predict the future positions of a moving object. In fact, the first and third policy are the good representatives of the policies in

[10]. They further improve the update policies in [2], by exploiting the better road-network representation and acceleration profiles with routes. It should also be noted that Ding and Guting [3] have recently discussed the use of what is essentially segment-based tracking based on their proposed data model for the management of road-network constrained moving objects. In paper [8], the non-linear models such as the acceleration are used to represent the trajectory which is affected by the abnormal traffic such as traffic incident.

Updates based on individual object and their group

Most existing update techniques are developed to process individual updates efficiently [1, 2, 9, 10]. To reduce the expensive uplink updates from the objects to the location server, Lam et al. [6] propose a group-based scheme in which moving objects are grouped so that the group leader will send location update on behalf of the whole group. A group-based location update scheme for personal communication network is also proposed in [5]. The aim is to reduce location registrations by grouping a set of mobile objects at their serving VLRs.

Our work improves the tracking technique from the aspect of prediction model and update mode, and focuses on the accuracy of the predicted positions of the objects in urban road networks. Based on their predicted movement functions, we groups objects to further reduce their location updates. To the best of our knowledge, there exists no proposal for tracking of moving objects that combines the simulation based prediction and grouping of objects by exploiting the movement features of objects in traffic systems.

3 Data Model and Trajectory Prediction

We model a road network with a graph of cellular automata (GCA), where the nodes of the graph represent road intersections and the edges represent road segments with no intersections. Each edge consists of a cellular automaton (CA), which is represented, in a discrete mode, as a finite sequence of cells. The CA model was used in this context by [12].

In the GCA, a moving object is represented as a symbol attached to the cell and it can move several cells ahead at each time unit. Intuitively, the velocity is the number of cells an object can traverse during a time unit. Let i be an object moving along an edge. Let $v(i)$ be its velocity, $x(i)$ its position, $gap(i)$ the number of empty cells ahead (forward gap), and $P_d(i)$ a randomized slowdown rate which specifies the probability it slows down. We assume that V_{max} is the maximum velocity of moving objects. The position and velocity of each object might change at each transition of the GCA according to the rules below (adapted from [12]):

1. if $v(i) < V_{max}$ and $v(i) < gap(i)$ then $v(i) \leftarrow v(i) + 1$
2. if $v(i) > gap(i)$ then $v(i) \leftarrow gap(i)$
3. if $v(i) > 0$ and $random() < P_d(i)$ then $v(i) \leftarrow v(i) - 1$
4. if $(x(i) + v(i)) \leq l$ then $x(i) \leftarrow x(i) + v(i)$

The first rule represents linear acceleration until the object reaches the maximum speed V_{max}. The second rule ensures that if there is another object in front

of the current object, it will slow down in order to avoid collision. In the third rule, the $P_d(i)$ models erratic movement behavior. Finally, the new position of object i is given by the fourth rule as the sum of the previous position with the new velocity if it is in the CA. Figure 2 shows a transition of the cellular automaton of edge (n_1, n_2) in Figure 1 in two consecutive timestamps. We can see that at time t, the speed of the object o_1 is smaller than the gap (i.e. the number of cells between the object o_1 and o_2). On the other hand, the object o_2 will reduce its speed to the size of the gap. According to the fourth rule, the objects move to the corresponding positions based on their speeds at time $t+1$.

We use GCAs not only to model road networks, but also to simulate the movements of moving objects by the transitions of the GCA. Based on the GCA, a *Simulation-based Prediction (SP)* model to anticipate future trajectories of moving objects is proposed. The SP model treats the objects simulated results as their predicted positions. Then, by the linear regression, a compact and simple linear function that reflects future movement of a moving object can be obtained. To refine the accuracy, based on different assumptions on the traffic conditions we simulate two future trajectories to obtain its predicted movement function. Figure 3 and Figure 4 show the comparison of the SP model and the linear prediction (LP) model. We can see from Figure 3 that the LP model cannot predict accurately the future trajectories of objects due to the frequent changes of the object velocity in traffic road networks.

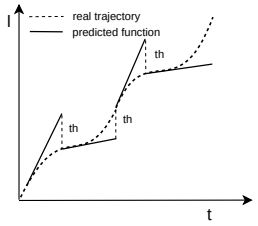

Fig. 3. The *Linear* Prediction

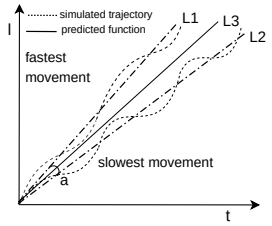

Fig. 4. The *Simulation Based* Prediction

Most existing work uses the CA model for traffic flow simulation in which the parameter $P_d(i)$ is treated as a random variable to reflect the stochastic, dynamic nature of traffic system. However, we extend this model for predicting the future trajectories of objects by setting $P_d(i)$ to values that model different traffic conditions. For example, laminar traffic can be simulated with $P_d(i)$ set to 0 or a small value, and the congestion can be simulated with a larger $P_d(i)$. By giving $P_d(i)$ two values, we can derive two future trajectories, which describe, respectively, the fastest and slowest movements of objects. In other words, the object future locations are most probably bounded by these two trajectories. The value of $P_d(i)$ can be obtained by the experiences or by sampling from the given dataset. Our experiments show one of methods to choose the value of $P_d(i)$. It is proved that 0 and 0.1 are realistic values of $P_d(i)$ in our cases.

For getting the future predicted function of an object from the simulated discrete points, we regress the discrete positions to a linear function by the Least Square Estimation (LSE) in Statistics. It can be calculated efficiently with low data storage cost. Let the discrete simulated points be $(t_0, l_0), (t_1, l_1), ..., (t_i, l_i), ..., (t_{n-1}, l_{n-1})(i \geq 0, n > 0)$, where t_i is the time at $i+1$ timestamp, l_i is the relative distance of the moving object in an edge at timestamp t_i, n is the total time units for the simulation, a linear function of time variable t can be obtained as follows:

$$l = a_0 + a_1 t \tag{1}$$

where the slope a_1 and the intercept a_0 can be calculated in Statistics

$$a_1 = \frac{n \sum_{i=0}^{n-1} t_i l_i - \sum_{i=0}^{n-1} t_i \sum_{i=0}^{n-1} l_i}{n \sum_{i=0}^{n-1} t_i^2 - (\sum_{i=0}^{n-1} t_i)^2} \tag{2}$$

$$a_0 = \frac{1}{n} \sum_{i=0}^{n-1} l_i - \frac{a_1}{n} \sum_{i=0}^{n-1} t_i \tag{3}$$

After regressing the two simulated future trajectories to two linear function denoting L_1 and L_2 in Figure 4, we can compute the middle straight line L_3, the bisector of the angle a between L_1 and L_2 as the final predicted function $L(t)$.

Through the SP model, we obtain a compact and simple linear prediction function for the moving object. However, this is different from the linear prediction in that the simulation-based prediction method not only considers the speed and direction of each moving object, but also takes correlation of objects as well as the stochastic behavior of the traffic into account. The experimental results also show it is a more accurate and effective prediction approach.

4 Group Location Update Strategy

As the number of updates from moving objects to the server database depends on both the update frequency and the number of objects updated, we propose a group location update strategy based on the SP model (GSP) to minimize location updates. In the GSP, for each edge in a road network, the objects are grouped or clustered by the similarity of their predicted future movement function and their locations are represented and reported by the group (Figure 1). It means that the nearby objects with similar movement during the future period on the same edge are grouped and only the object nearest to its group center needs to report the location of the whole group. Within a certain precision, the locations of other objects can be approximated to their group location.

The idea of grouping objects for location updates is similar to the GBL proposed in [6]. The main differences are that the GSP groups the objects by their future movement function predicted from the SP model instead of their current locations and predicted locations after a time parameter τ obtained by

current velocity. Grouping by objects predicted movement function can insure the validity of the groups. The accurate prediction from the SP model can also reduce the maintenance of the groups. Due to the constraint of the road network, each group in the GSP has its lifetime in accordance to the edge. A group only exists on one edge and will be dissolved when objects within it leave the edge. Furthermore, unlike the GBL in which objects have to send a lots of messages to each other and compute the costly similarities for grouping and leader selection, the GSP executes the grouping on the server after predicting. This alleviates the resource consumption of moving clients and overloads of wireless communication.

The similarity of two objects simulated future trajectories in the SP model has to be computed by comparing a lot of feature points on the trajectories. A straightforward method is to select some of the simulated points to sum their distance difference. However, the computation cost for simulated trajectories is very high. For simplicity and low cost, we group objects by comparing their final predicted linear functions. Therefore, the movement similarity of two objects on the same edge can be determined by their predicted linear functions and the length of the edge. Specifically, if both the distance of their initial locations and their distance when one of the objects arrives the end of the edge are less than the given threshold (corresponding to the update threshold ε), we group the two objects together. These distances can be easily computed by their predicted functions. Figure 5 shows the predicted movement functions (represented as $L1, L2, L3, L4, L5$) of the objects o_1, o_2, o_3, o_4, o_5 on the edge (n_1, n_2) from Figure 1. le is the length of the edge and $t1, t2, t3, t4$ are respectively the time when the objects o_1, o_2, o_3, o_4 arrive the end of the edge. Given the threshold is 7, for objects o_1, o_2, the location difference between them at initiate time and t_1 are not larger than 7, therefore, they are clustered in one group c_1. We then compare the movement similarities of o_3 and o_1 as well as o_3 and o_2. The location differences are all not larger than 7, so o_3 can be inserted to c_1. Although at the initiate time, o_3 and o_4 are very close with the distance less than 7, they move far away each other in the future and their distance exceeds 7 when o_3 arrives the end of the edge. They cannot be grouped in one cluster. In the same way, o_4 and o_5 form the group c_2. Therefore, given a threshold, there are three cases of the objects predicted linear function when they are grouped together on one edge. These cases can be seen in the Figure 5 respectively labeled by a ($L2$ and $L3$ with objects moving close), b ($L1$ and $L3$ with objects moving far away) and c ($L1$ and $L2$ with one object exceeding another one).

In a road network, we group objects on the same edge. When objects move out of the edge, they may change direction independently. So we dissolve this group and regroup the objects in adjacent edges. Each group has its lifetime from the group formation to all objects within it leaving the edge. For each edge, with the objects predicted functions, groups are formed by clustering together sets of objects not only close to each other at a current time, but also likely to move together for a while on one edge. We select the object closest to the center of its group both the current time and some period in future on the edge to represent the group. The central object represents its group and is responsible for reporting

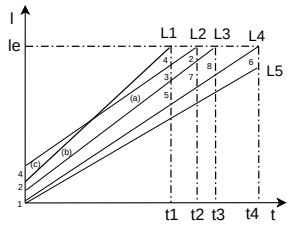

Fig. 5. Grouping objects by their predicted functions

the group location to the server. For reselecting the central object, according to objects predicted future functions, we can choose the objects close to the center of the group during its lifetime as the candidates of the central object. We can also identify when the central object will move away from the group center and choose another candidate as a new central object. A joining from a moving object to a group must be executed as follows. The system first finds the nearby groups according to the edge the object lies and then compares the movement similarity of the object and the group by their predicted functions. If the object cannot join to the nearby groups, a new group will be created with only one member. When a moving object leaves a group, the central object of the group needs to be reselected. However, for the object leaving an edge, to reduce the central object reselection of its group, we just delete it from its group and do not change the central object until the central object leaves the edge.

In the GSP, the grouping method assures the compactness and movement similarity of the objects within a group. Given the precision threshold ε, the objects locations in a group may be approximated by the location of the group (i.e. location of its central object). Only the location update from the central object of the group to the location server is necessary. After the server makes predictions for objects in a road network and initiates their groups, the client of the central object measures and monitors the deviation between its current location and predicted location and reports its location to the server. Other objects do not report their locations unless they enter the new edge. The prediction and grouping of objects are executed in the server and the group information (including the edge id, the central object id, its predicted function and a set of objects within the group) is also stored in the database of the server. The update algorithm in the server is described in Algorithm 1.

5 Performance Evaluation

In this section, we experimentally measure the performance of the *point-based*, *segment-based* [1], and our *GSP* update policies. We also evaluate the simulation based prediction (SP) method used in the *GSP* update policy with the simulation parameter P_d and prediction accuracy compared to the linear prediction (LP) method. We implemented the three update policies in Java and carried out experiments on a Pentium 4, 2.4G PC with 256MB RAM running Windows XP.

Algorithm 1: GroupUpdate($objID, pos, vel, edgeID, grpID$)

input : $objID, edgeID$ and $grpID$ are respectively the identifier of the object to be updated, its edge and group, pos, vel are its position and velocity
Simulate two future trajectories of $objID$ with different P_d by the CA;
Compute the future predicted function $l(t)$ of $objID$;
if *$objID$ does not enter the new edge* **then**
  if *$objID$ is the central object of $grpID$* **then**
    Update the current position pos and predicted function $l(t)$ of $grpID$;
    Send the predicted function $l(t)$ of $grpID$ to the client of $objID$;
  end
else
  if $GetObjNum(grpID) > 1$ **then**
    Deletes $objID$ from its original group $grpID$;
    if *$objID$ is the central object of $grpID$* **then**
      Reselect the central object of $grpID$, update and send its group info;
    end
  else Dissolve the group $grpID$;
  Find the nearest group grp_1 for $objID$ on $edgeID$;
  Compute the time t_e when $objID$ leaves $edgeID$ by $l(t)$ and $edgeID$ length;
  if *Both distances between $objID$ and grp_1 at initiate time and $t_e \leq \varepsilon$* **then**
    Insert $objID$ into grp_1 and send grp_1 identifier to the client of $objID$;
    Reselect the central object of grp_1, update and send its group info;
  else Create a new group grp_2 only having $objID$ and send its group info;
end

5.1 Datasets

The datasets of our experiments are generated by Thomas Brinkhoff Network-based Generator of Moving Objects [13], which is used as a popular benchmark in many related work. The generator takes a map of a real road network as input and may simulate the moving behaviors of various kinds of moving objects in real world. Our experiment is based on the real map of Oldenburg city with 7035 segments. For modeling the road network, we associate those adjacent but not crossed segments together to form edges of the graph. After that, the total number of edges is 2980 and their average length is 184. We set the generator the parameter "maximum time" to be 20, "maximum speed" 50 and the number of initial moving objects 100000. The generator places these objects at random positions on the road network, and updates their locations at each time-stamp. The positions of the objects are given in two dimensional X-Y coordinates. We transform them to the form of (`edgeid`, `pos`), where `edgeid` denotes the edge identifier and `pos` denotes the object relative position.

5.2 Update Performance

For evaluating update performance and accuracy, we consider two metrics, namely, the number of updates (for 100000 moving objects during 20 time-stamps) and average error of the location of each object at each times-tamp as following.

$$average\_error = \frac{1}{mn} \sum_{j=0}^{n-1} \sum_{i=0}^{m-1} |l_{ij} - l_{rij}| \tag{4}$$

where l_{ij} is the predicted location of mo_j or approximated location by its group at the timestamp t_i, l_{rij} is the real location of mo_j at timestamp t_i, m is total update time-stamps and n is the number of moving objects.

Figure 6 and 7 show the update number and average error of three update policies respectively with different update thresholds. We observe that with increase of the threshold, the update number will decrease and the average error will increase in any one of these three policies. This is because the larger the threshold is, the larger the allowable deviation between the predicted location and its real location, and the less updates it causes. However, the GSP update policy outperforms the other two policies for fewer number of update and average error. Specifically, the GSP only causes 30%-40% updates of segment-based policy and 15%-25% of point-based policy, while improves the location accuracy with lower average error. This owns to the accurate prediction of the SP method and the technique of grouping moving objects. For the GSP policy, larger threshold results in more objects in one group and therefore fewer group updates and higher location average error. In addition, notice that the largest performance improvement of the GSP policy over other policies is for smaller thresholds. For thresholds below 10, the GSP policy is nearly three times better than the segment-based policy and four times than the point-based policy.

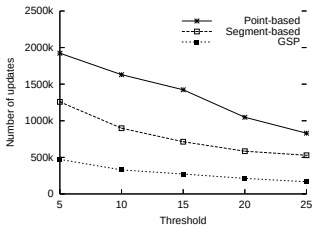

Fig. 6. Number of Updates

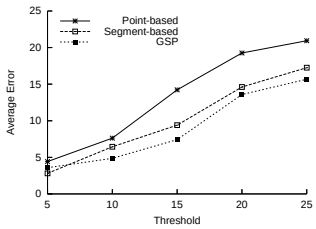

Fig. 7. Average Error of Updates

5.3 Prediction Performance

The Slowdown Rate P_d We study the effect of the choices of different P_d, which determines two predicted trajectories corresponding to the fastest and slowest movements. We use P_d from 0 to 0.5 and measure the prediction accuracy by the average error and overflow rate. The overflow rate represents the probability of the predicted positions exceeding the actual positions. The purpose of this metric is to find the closest two trajectories binding the actual one as future trajectories. In this way, we choose the P_d with both the lower average error and overflow

rate, which can also be treated as one of methods to set the proper values of P_d in a given dataset. Figure 8 and Figure 9 show the prediction accuracy of the SP method with different P_d. We can see that when P_d is set to 0 and 0.1, both the average error and overflow rate are lower than others. Therefore, we use them in the experiments to obtain better prediction results.

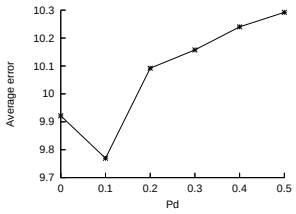

Fig. 8. Average Error with Different P_d

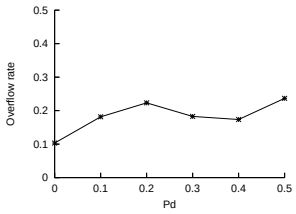

Fig. 9. Overflow Rate with Different P_d

Prediction Accuracy and Cost Finally, we compare the prediction accuracy of the SP method with the LP method. We measure the average error for predicted locations (without grouping) with different thresholds. From Figure 10, we observe that the average error will increase when the threshold increases. This is tenable in both the LP and SP method. However, the SP method predicts more accurately than the LP method with any threshold. For the costs of SP method, as its time complexity depends on many factors, we compute average CPU time when simulating and predicting the movements of one object along the edge with length 1000. The results show that the average cost of one prediction is about 0.25ms. This is acceptable even for large number of moving objects.

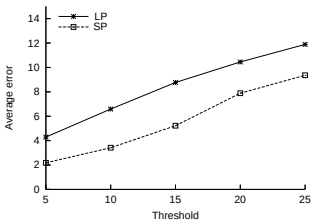

Fig. 10. Comparison of Prediction Accuracy

6 Conclusion

Motivated by the features of vehicles movements in traffic networks, this paper presents new techniques to track network-constrained moving objects. Our contribution is twofold. First we propose a prediction model, based on simulation,

which predicts with a great accuracy the future trajectories of moving objects. This lowers location update frequency in tracking. Then, based on the prediction, we propose a group update strategy which further reduces location updates and minimizes the cost of wireless communication. The experiments show that the update strategy has much higher performance and location accuracy.

Acknowledgments

This work was partially supported by the grants from the Natural Science Foundation of China with grant number 60573091, 60273018; China National Basic Research and Development Program's Semantic Grid Project (No.2003CB317000); the Key Project of Ministry of Education of China under Grant No.03044; Program for New Century Excellent Talents in University(NCET); Program for Creative PhD Thesis in University.

References

1. A. Civilis, C. S. Jensen, J. Nenortaite, S. Pakalnis. Efficient Tracking of Moving Objects with Precision Guarantees. In MobiQuitous 2004: 164-173.
2. A. Civilis, C. S. Jensen, S. Pakalnis. Techniques for Efficient Road-Network-Based Tracking of Moving Objects. In IEEE Trans. Knowl. Data Eng. 17(5): 698-712 (2005).
3. Z. Ding, R. H. Guting. Managing Moving Objects on Dynamic Transportation Networks. In SSDBM 2004: 287-296.
4. H. Gowrisankar, S. Nittel. Reducing Uncertainty In Location Prediction Of Moving Objects In Road Networks. In GIScience 2002: 228-242.
5. Y. Huh, C. Kim. Group-Based Location Management Scheme in Personal Communication Networks. In ICOIN 2002: 81-90.
6. G. H. K. Lam, H. V. Leong, S. C. Chan. GBL: Group-Based Location Updating in Mobile Environment. In DASFAA 2004: 762-774.
7. K. Y. Lam, O. Ulusoy, T. S. H. Lee, E. Chan, and G. Li, An Efficient Method for Generating Location Updates for Processing of Location-Dependent Continuous Queries. In DASFAA 2001: 218-225.
8. G. Trajcevski, O. Wolfson, B. Xu, Peter Nelson: Real-Time Traffic Updates in Moving Objects Databases. In DEXA 2002: 698-704.
9. O. Wolfson, A. P. Sistla, S. Camberlain, Y. Yesha. Updating and Querying Databases that Track Mobile Units. In Distributed and Parallel Databases 7(3): 257-387 (1999).
10. O. Wolfson and H. Yin. Accuracy and Resource Consumption in Tracking and Location Prediction. In SSTD 2003: 325-343.
11. J. Zhou, H. V. Leong, Q. Lu, K. C. Lee. Aqua: An Adaptive QUery-Aware Location Updating Scheme for Mobile Objects. In DASFAA 2005: 612-624.
12. K. Nagel and M. Schreckenberg, A Cellular Automaton Model for Free Traffic, In physique I, 1992, 2: 2221-2229.
13. T. Brinkhoff. A Framework for Generating Network-based Moving Objects, In GeoInformatica 6(2): 153-180 (2002).