This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2008 proceedings.

Mobility-assisted Location Management for Vehicular Ad Hoc Networks

Zhaomin Mo, Hao Zhu, Kia Makki, and Niki Pissinou
Telecommunications and Information Technology Institute
Florida International University, Miami, FL 33174

***Abstract*— Vehicular Ad-hoc Networks (VANETs) are gaining importance for inter-vehicle communication, because they allow for the local communication between vehicles without any infrastructure, configuration effort, and without the expensive cellular networks. As geographic routing can be used to achieve efficient data delivery in VANETs, how to provide location management service in VANETs to facilitate geographic routing remains a fundamental issue. In this paper we will present a novel location management protocol, call MALM, to provide location service to vehicles in VANETs. In MALM, a vehicle calculates the current location of other vehicles by using Kalman filtering based on the historical location information of other nodes. Theoretical analysis is provided to show that MALM is able to achieve high location information availability in the network. We evaluate the performance of MALM via extensive simulations. The simulation results show that MALM works efficiently in VANETs.**

I. Introduction

Vehicular Ad-hoc Networks (VANET) using 802.11-based WLAN technology have recently received considerable attention. By leveraging low cost, high bandwidth interface, VANETs can be deployed quickly and economically. It offers high data rates, and is more cost-effective than cellular, since there are no high monthly fees for each wireless device or user. With global position system (GPS) and digital map (e.g. MapMechanics [4]) equipped with vehicles, a number of geographic routing protocols for VANETs have been proposed. In [9] Zhao *.et. al* studied the carry-and-forward scheme, named VADD, which delivers packets in sparse VANETs where disconnection happens frequently. In [5] an efficient multi-hop routing protocol, named MURU, was proposed for urban VANETs. An advanced greedy forwarding (AFG) [6] based on GPSR is proposed to increase packet delivery ratio based on realistic vehicular traces. All these protocols assume that a location management service is available to provide the source with the destination's location. In addition, many applications in VANETs may require location management service. For example, a real-time fleet management requires vehicles to be tracked and located. Therefore, it is fundamentally important to design a good location management scheme in VANETs.

Some location management systems have been proposed in ad hoc networks. Most of them adopt the client-server architecture, and usually involve two steps: location update and location query, respectively. In the location update process, mobile nodes periodically send their own location information to one or more location servers in the network to update their current locations. In the location query process, a node queries location servers to get other nodes' location information.

One example of such systems can be found in [3], where a decentralized location service named GLS is proposed. In GLS, a node is assigned with an ID, and updates its location servers periodically with its up-to-date location information. A small set of nodes having the IDs "closest" to its ID are chosen as its location servers. GLS works well when nodes' mobility is low. While it becomes much less efficient when network mobility grows. It is because in order to find appropriate location servers, a node needs to scan the entire network to find nodes with "closest" ID. In a network with high mobility, a node needs to scan the entire network frequently in order to keep the "closest ID" property, as a result a lot of overhead is introduced. Besides, high network mobility makes querying location information from location servers more difficult.

There are some other location management schemes proposed for ad hoc networks. Xue *et al.* [8] proposed a distributed location management scheme named DLM, which makes use of hash function to map the ID of a node to a set of sub-domains in the network. Then some nodes in those sub-domains would be picked as location servers. whenever a node crosses the borders of certain region or when periodic updates for maintaining a soft state are transmitted, all location servers have to be updated. Since the location servers are evenly distributed in the network the resulting message overhead is similar to that of network-wide flooding, which reduces its scalability. Basagni *et al.* [2] proposed a self-optimizing location updating scheme called DREAM. In DREAM, the farther two nodes are separated, the less often they exchange location information. However, with such scheme, higher network mobility makes a node updating its location more often. This finding indicates that DREAM is not adaptive enough for highly dynamic networks.

As discussed above, in a highly dynamic network, it is not feasible to update a node's location constantly and explicitly. To cope with the highly dynamic topology of VANETs, in this paper we propose a novel location management protocol called MALM (Mobility-Assisted Location Management). The basic idea of MALM is to take advantage of high mobility in VANETs to disseminate a node's location update information to other nodes. Then each node independently applies Kalman filtering to process the received location information. As a result, a node is able to predict other nodes' location information without explicit location query. Compared with client-server architecture, which is commonly adopted by traditional location management protocols, MALM has the following merits. First, without location information update and query on location servers, MALM is distributed. Second, different from traditional location management schemes whereby before sending out data packets, a node has to explicitly query for the target node's location and wait for the reply from location servers, in MALM a node can get the target nodes' location information real-time by applying Kalman filtering to do prediction. The performance of MALM is extensively evaluated via simulations and the results show that MALM can provide location information with low delay.

The rest of this paper is organized as follows. Section II describes the details of MALM. Section III shows the performance evaluation results, and Section IV concludes the paper.

II. The MALM Protocol

In this section we describe the details of the MALM. The protocol consists of two parts: the mobility-assisted location information dissemination, and the location information prediction by using Kalman filtering.

A. The Mobility-assisted Location Information Dissemination

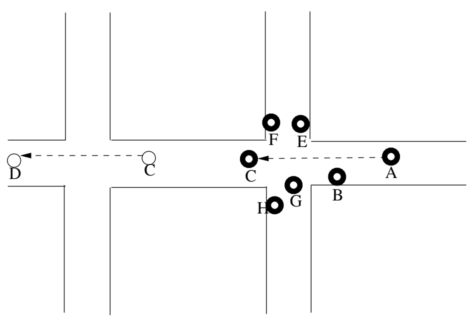


Fig. 1. An Illustration of Mobility-Assisted Dissemination

Since in VANETs nodes[1] move fast along roadway trajectory, MALM takes advantage of such characteristic to disseminate a node's location information to other nodes efficiently. The basic idea is shown in Figure 1. At time t nodes are in locations depicted by dark circle, and node A disseminates its location update message to its neighbors. Node C overhears node $A's$ location information from the received message. After a time period, all nodes reach new locations depicted by empty circles. Then node C disseminates its location update message, which includes node A's historical location information, to its neighbors, including node D. Thus a node who was not able to directly receive another node's location information can indirectly get that information with the aid of high mobility in VANETs. Following the basic idea aforementioned, we design our dissemination mechanism as follows. In MALM each node maintains two tables: *location table (LT)*, which stores the location update information overheard from other nodes, and *dissemination table (DT)*, which stores the location information to be disseminated. A location update message includes a number of *Location Information Units (LIUs)*. As Shown in Figure 2, a LIU includes node ID, node's location (represented as X coordinate and Y coordinate), direction, hop number and timestamp (the time associated with a node's location and direction). Initially the hop number is set to zero. When a node (let's say node B) receives a location update message from another node (let's say node A), it stores all *LIUs* from the location update message in its LT for future prediction use. Meanwhile, node B compares the $LIUs$ in the received location update message with the $LIUs$ in its *DT*. If there exists a LIU that is in the location update message but not in its DT, node B adds this *LIU* into its DT. If both the location update message and $B's$ DT have $LIUs$ with the same node ID, the timestamps of these two *LIUs* are compared. The *LIU* with the fresher timestamp would be kept in DT and the other one would be discarded. In this way, only most recent location information will be disseminated further. To save the storage consumption, a node may only need to store the location information of other nodes it is interested in. Since by applying Kalman filter only recent historical information is needed, to save storage, only location information within a certain freshness threshold is needed to be kept in LT and DT. Note that the freshness thresholds of LT and DT may be different. With a number of disseminations, a LIU can be sent to nodes in different locations. However, there is a tradeoff between the availability of $LIUs$ and the network bandwidth consumption. For example, if a LIU is only allowed to be disseminated one time, then only one-hop neighbors of this node are able to receive this LIU. On the other hand, if a LIU is allowed to be disseminated many times, the corresponding bandwidth consumption becomes considerable and may hurt the overall performance of the network. In the following we will first show the calculation of the availability of a LIU, then we study the impacts of network conditions to the balance of the tradeoff.

X Coordinate	Y Coordinate	Dirct
Node ID	Timestamp	Hop No

Fig. 2. The Format of a Node's Location Information

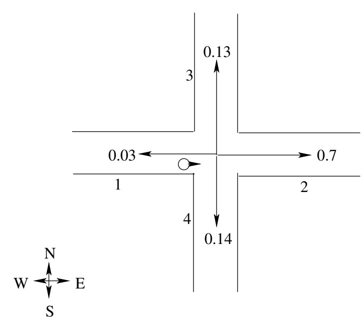


Fig. 3. An Example of How to Generate A Row in Transition Matrix

We assume an area is divided into m segments (a section of a street in between two intersections). There are n nodes in the networks. For simplicity, we assume that every node moves with the same velocity and the time for a node to

[1]For convenience, we use vehicle and node interchangeable in the rest of paper.

move across a segment is T [2]. Initially nodes are deployed in the network randomly. The direction selection of a node at any intersection is approximated by first order Markov chain, in which a state in the Markov chain corresponds to a segment. Every node has a transition matrix P with size of $m \times m$, which describes the probability of moving from one segment to another. The sum of each row in P equals to one, which describes all possible directions of a node at an intersection. By generating the state transition matrix for each node , we can calculate the stable state probabilities[3] of a node on each segment. An example of generating a row of transition matrix at an intersection is shown in Figure 3. A node on segment 1 moving towards east arrives at an intersection. The probabilities that this node moves to segment 2, segment 3, segment 1 and segment 4 are 0.70, 0.13, 0.03 and 0.14, respectively. The probability that this node moves to any other segments is 0, thus we can get the transition matrix for this node as follows: $P[1][1] = 0.03, P[1][2] = 0.7, P[1][3] = 0.13$ and $P[1][4] = 0.14$. A realistic mobility can be simulated by letting a node having different transition matrix at different intersections. Therefore, a $1 \times m$ probability vector associated with a node can be calculated to describe the this node's possible location at any time interval. For instance, node i is initially located on segment $s(i)$ and moves according to its transition matrixes. At time t, the probability vector of node i's location is $s(i) * \prod_{r=1}^{\lceil t/T \rceil} P_i(r)$ where $P_i(r)$ is node $i's$ r^{th} transition matrix at the r^{th} time interval. Since time interval T is much longer than the time needed to transmit a packet (queueing delay plus transmission delay), it is reasonable to assume that two nodes are able to exchange location update message when they are on the same segment (or, equivalently, stay in the same state). As a result, after n time intervals, a node (say, node A) can receive another node's (say, node B) location information generated at time t_0 if one of the following events happens:

1) Node A and node B meet with each other in a segment during n time intervals, and exchange location update messages. Such situation is called zero-relay-dissemination.
2) Node B disseminates location update message to another node (say node C) and then node C meets with node A and disseminates node $B's$ location information to node A. Such situation is called one-relay-dissemination.
3) Node $B's$ location information is indirectly disseminated to node A by more than one relays during n time intervals. Such situation is called multiple-relay-dissemination.

Thus, the probability $B's$ location information available for node A during n time intervals would be the summation of the probabilities for the three cases mentioned above. We will give the calculation of probabilities for the first two cases. The probability that zero relay dissemination happens at least once during n time intervals, denoted by $Pr^0_{A,B}(1,n)$ is calculated as following:

$$Pr^0_{A,B}(1,n) = 1 - \prod_{q=1}^{q=n}(1-(s(A)*\prod_{r=1}^{q} P_A(r)) \otimes (s(B)*\prod_{r=1}^{q} P_B(r))) \quad (1)$$

where $s(A)$ and $s(B)$ are the initial location of node A and B, and $P_A(i)$ is the transition matrix for node A at i^{th} time interval. The calculation of $P(i) \otimes P(j)$ is defined as following:

$$P(i) \otimes P(j) = 1 - \prod_{k=1}^{m}\prod_{l=1}^{n}(1 - P(i)[k][l] * P(j)[k][l]) \quad (2)$$

where m is the number of segments in the area.

The probability of one relay dissemination through node C during time t, denoted by $P^1_{A,B,C}(1,n)$ is calculated as following:

$$\begin{aligned} Pr^1_{A,B,C}(1,n) &= 1 - \prod_{q=1}^{n}(1-(1-\prod_{x=1}^{q}(1 - Pr^0_{A,C}(1,x))) \\ &\quad (1 - \prod_{y=q}^{n}(1 - Pr^0_{C,B}(q,y))) \end{aligned} \quad (3)$$

Then, the probabilities of one relay dissemination through any node is calculated as following:

$$Pr^1_{A,B}(1,n) = 1 - \prod_{h \in H}(1 - Pr^1_{A,B,h}(1,n)) \quad (4)$$

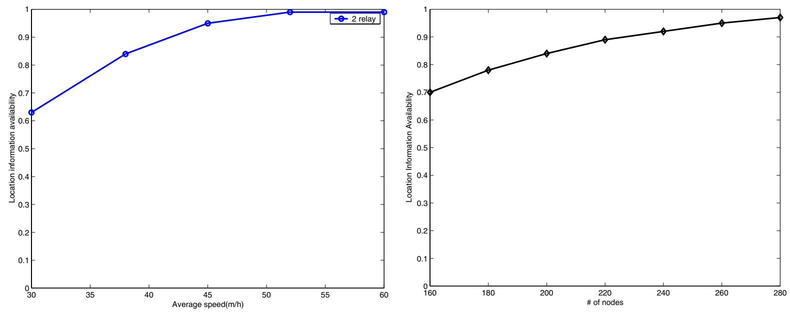
where H can be any node except node A and B. $Pr^1_{A,B,h}(1,n)$ is defined in Eq. (3). To save space, the probability calculation of two or more relay dissemination is not shown.

Higher availability of a LIU can be achieved with more multi-hop disseminations, whereas more bandwidth consumption will be consumed. Thus to save bandwidth consumption, *hop threshold* is used to limit the maximum number that a LIU is allowed to be disseminated. An example of location information availability in a network with 200 nodes and 60 segments is shown in Figure 5. We observe that when the the hop number threshold is set to be 1, the availability is much lower than the availability when the hop number threshold is set as 2 or 3. Compared with hop number of 3, the availability when hop number is set as 2 is a bit lower, while much overhead can be saved with the hop number of 2.

In addition to the hop number allowed to be disseminated, our study indicates that network mobility and density also affect the availability of a LIU. As a node moves faster, it is able to meet more nodes, therefore there are more chances for other nodes to get its location update messages. An simulation of 200 nodes moving with different average speeds in a network is run and the simulation results are shown in Figure 4 (a). It shows that the location information availability increases as the average node velocity increases. Such property indicates our dissemination mechanism can exploit high mobility of VANETs. Also, the location information availability increases as the node density becomes higher. The reason is that higher

[2]by applying different value of T, different velocities of nodes can be simulated.

[3]Since we assume each node moves within the area, the Markov chain of each node is recurrent.



(a) Impact of Velocity (b) Impact of Density

Fig. 4. Impact of the Velocity and Density on the Location Information Availability

density brings more relay nodes that may participate in disseminating location update messages, thereafter higher location information availability can be achieved. An simulation with different node densities is run and the simulation results are shown in Figure 4 (b). We can see that higher density does bring higher location information availability.

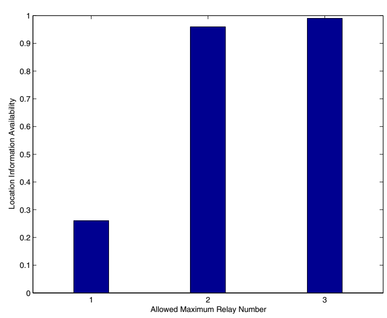


Fig. 5. Location Information Availability

B. The Kalman-Filter Based Location Prediction

Kalman filter is a well-known tool to estimate a linear system's past, present or future state by using a time sequence of measurements of the system state and a statistical model that characterizes the system and measurement errors, along with initial conditions. In MALM, with another node's (let's say B) historical location information stored in a node's (let's say A) LT, Kalman filter can be implemented by node A to predict node B current or future location information. Extended Kalman filter can also be used to provide higher prediction accuracy, but much calculation overhead may be involved. In this paper we will focus on Kalman filter to save calculation overhead. If $B's$ historical location information is not available for node A, two methods could be used by node A. Node A may defer the prediction until it gets $B's$ location information from newly received location update message or A may query its neighbors for $B's$ location information by sending a request message. For simplicity, in this paper, we adopt the first method. However the second method may significantly reduce the delay of obtaining the node $B's$ location information. In the future work the second method will be discussed.

Since location update messages are disseminated periodically (e.g., every 5 seconds), any two $LIUs$ for node B have at least 5 seconds' time difference. To fill up the time slots between two historical location updates, interpolation is used by node A. In our current work, liner interpolation technique is adopted. After interpolation, a series of $B's$ historical location information are generated to be the input for node $A's$ Kalman filter to predict node $B's$ current location. To describe a node's location and mobility status in Kalman filter, four system states are needed: the node's location at x-dimension and y-dimension plus the node's velocity projected to x-dimension and y-dimension. The state vector x with time index k is shown as below:

$$x_k = (px_k, py_k, vx_k, vy_k)^T \tag{5}$$

Where px_k and py_k are the location at time index k, vx_k and vy_k are the projected velocity at time index k. We assume a node's velocity along one direction changes linearly, thus the state at time $k+1$ can be calculated as follows:

$$x_{k+1} = Dx_k + w_k \tag{6}$$

and the input equation is shown in the following:

$$y_k = Cx_k + z_k \tag{7}$$

Where D and C are matrix index, w is process noise and z is GPS measurement noise. GPS measurement noise is considered since GPS can't provide accurate location information. Nodes moving towards different directions may implement different matrix D. An example of the matrix D when nodes move along X-dimension is shown as below:

$$D = \begin{pmatrix} 1 & t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{8}$$

where t is the time interval between two Kalman filter's inputs. matrix C is shown as following:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{9}$$

With all these parameters, Kalman filter takes recurrent steps to calculate a node's current or future states by using the following equations [7].

$$K_k = DP_kC^T(CP_kC^T + S_z)^{-1} \tag{10}$$

$$x_{k+1} = Dx_k + K_k(y_{k+1} - Cx_k) \tag{11}$$

$$P_{k+1} = DP_kD^T + S_w - DP_kC^TS_z^{-1}CP_kD^T \tag{12}$$

where S_z is GPS measurement noise covariance, S_w is process noise covariance, K matrix is Kalman gain and P matrix is estimation error covariance.

III. Performance Evaluation

We build the simulation using the ns-2 simulator [1], which supports for simulations of routing protocols over wireless networks. Vehicles are equipped with 802.11 network interface, global position system (GPS), and digital maps (e.g. MapMechanics [4]). Therefore a vehicle can achieve its own location (imprecise) and street-level road geometry at any instance. IEEE 802.11 DCF protocol is used as the MAC layer transmission protocol, and the shadowing propagation model is used to reflect the dynamic channel conditions in VANETs.

To evaluate the performance of MALM, we evalute the performance of MALM coupled with MURU [5], a geographical routing protocol in VANETs. MURU [5] has been shown to be an efficient geographical routing for urban VANETs. To evaluate the efficiency of MALM, we evaluate the performance of MURU with the help of MALM and MURU working with ideal location service (the source always knows the destination's accurate location information). The performance metrics are packet delivery ratio and end-to-end packet delay. Since the ideal location service doesn't involve any overhead, we don't compare the overhead of these two scenarios. The data traffic is generated by five source nodes randomly selected from networks and is sent to five randomly selected destinations. The number of nodes in the simulation varies from 60 to 150 to evaluate the impact of network density to our protocol performance. The simulation area is $700m \times 700m$ with block length of $100m$. Simulation runs 300 seconds and each case is repeated 20 times to achieve a high confidence of the results.

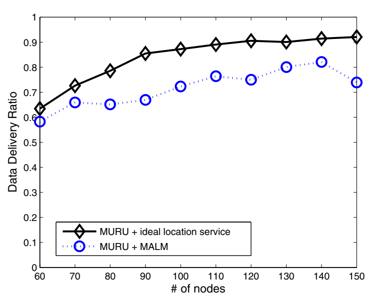


Fig. 6. Data Delivery Ratio for Different Network Densities

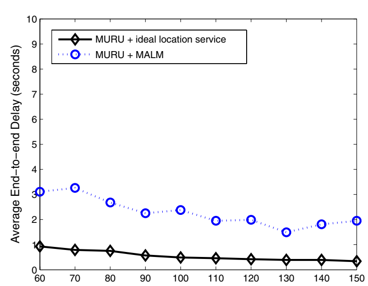


Fig. 7. Average Delay for Different Network Densities

Figure 6 shows the data delivery ratio of MURU working with MALM and ideal location service. In both scenarios, MURU can provide high data delivery ratio. The data delivery ratio increases as network density increases, which shows that MURU makes good use of network density. Compared with ideal location service, MURU with MALM only delivers less than about 10 percent of data. It indicates that MALM provides MURU with efficient location service. Figure 7 shows the average end-to-end delay of MURU with MALM and ideal location service. The end-to-end delay of MURU with ideal location service comes from MURU's routing delay. The end-to-end delay of MURU with MALM mainly comes from MALM's location acquisition delay. It is the time that a source needs to wait to get the destination's location information when the destination's location information is not available for the source. In both scenarios, the average end-to-end delay decreases with the increase of networks density since the performance of MURU improves as network density increases. The average delay of MURU with MALM decreases faster than that of MURU with ideal location service. The reason is that higher network density improves the performance of MALM.

IV. Conclusion

In this paper, we present a novel Mobility-assisted Location Management protocol, called MALM, for VANETs. MALM utilizes the high mobility of VANETs to help disseminate location information. With other nodes' historical location information, a node can predict other nodes' current or future location by using Kalman filter or extended Kalman filter. We performed theoretically analysis to show that MALM is able to achieve high location information availability with small protocol overhead. The performance of MALM is extensively evaluated through simulations. In our future work, a reactive scheme of reducing location acquisition delay by allowing a node to query the destination's location information from its one-hop neighbors will be studied.

References

[1] "Vint project," http://www.isi.edu/nsnam/ns/.

[2] S. Basagni, I. Chalamtac, V. R. Syrotiuk, and B. A. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," in *ACM MOBICOM'98*, Oct. 1998.

[3] J. Li, J. Jannotti, D. Decouto, D. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad-hoc Routing," in *ACM MobiCom'00*, 2000.

[4] MapMechanics, "GB Traffic Volumes," www.mapmechanics.com, May 2005.

[5] Z. Mo, H. Zhu, K. Makki, and N. Pissinou, "MURU: A Multi-Hop Routing protocol for Urban Vehicular Ad Hoc Networks," in *Int'l Conf. on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 2006)*, 2006.

[6] V. Naumov, R. Baumann, and T. Gross, "An Evaluation of Inter-Vehicle Ad Hoc Networks Based on Realistic Vehicular Traces," in *ACM MOBIHOC'06*, May 2006.

[7] Dan Simon, "Kalman Filtering," *Embeded System Programming*, 2001.

[8] Y. Xue, B. Li, and K. Nahrstedt, "A scalable location management scheme in mobile ad-hoc networks," in *IEEE Conference on Local Computer Networks*, 2001.

[9] J. Zhao and G. Cao, "VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks," in *IEEE INFOCOM'06*, 2006.