

# Map-guided Trajectory-based Position Verification for Vehicular Networks

Mervat Abu-Elkheir<sup>1</sup>Hossam S. Hassanein<sup>1</sup>Ibrahim M. Elhenawy<sup>2</sup>Samir Elmougy<sup>3</sup><sup>1</sup> School of Computing, Queen's University, Kingston, ON, Canada<sup>2</sup> Computer and Information Sciences, Zagazig University, Zagazig, Egypt<sup>3</sup> Computer and Information Sciences, Mansoura University, Mansoura, Egypt

{mfahmy|hossam}@cs.queensu.ca

henawy2000@yahoo.com

mougy@mans.edu.eg

**Abstract**— Vehicular networks are expected to enable vehicles on the road to exchange safety information; enhancing traffic flow and minimizing accidents. With vehicle positions being the most frequently exchanged information in vehicular networks, it becomes imperative to establish a strong level of trust in the announced positions before a vehicle initiates a response. This paper proposes a position verification scheme as part of a misbehavior detection framework that encompasses analysis techniques needed for the verification of exchanged vehicular messages. The scheme involves the estimation of a vehicle's trajectory via the integration of road and map information, as opposed to only depending on the vehicle's kinematics for future trajectory prediction. The trajectory estimation procedure uses the vehicle's previously received position updates to find a best-fit area within the road topology in which the vehicle is expected to be. The vehicle's current position announcement is compared against this plausible area to decide whether the position is consistent with the expected location. Our proposed scheme is verified via extensive simulations, developed on ns2, demonstrating the importance of integrating road and map information into position verification.

**Keywords**— component; vehicular networks; position verification; trajectory prediction; VANETs

## I. INTRODUCTION

Vehicular networks will provide the opportunity for communication-based road safety applications to have an extended vision of the road beyond what is typically provided by vehicles' onboard sensors. Most of the envisioned safety applications of vehicular networks require the knowledge of vehicles' locations, which is made available via the exchange of periodic position information among vehicles [1]. The required level of accuracy of such information is usually high for safety applications [2]. Errors in exchanged position information can be the result of temporary sensory malfunctions or the deliberate abuse of the vehicular network by misbehaving drivers in order to enhance their own traffic conditions, or even worse, disrupt the functionality of the network and perhaps cause accidents.

The IEEE 1609.2 Wireless Access in Vehicular Environments (WAVE) standard for security services in vehicular networks [3] addresses authentication and some privacy issues. However, it does not define mechanisms to detect inaccurate or falsified information that may be present in

the exchanged messages due to temporary sensory errors or misbehavior by "authenticated" drivers. There is a need to develop misbehavior detection schemes for vehicular networks in order to mitigate system misuse and enable the efficient implementation of vehicular safety applications. These schemes need to be ad hoc and distributed to accommodate the intermittent availability of dedicated infrastructure. A framework that defines how these schemes integrate and interact is necessary in order to provide an efficient and methodical way of implementing such schemes in vehicular networks.

This paper proposes both a framework for misbehavior detection in vehicular networks and a position verification scheme as part of that framework. The misbehavior detection framework defines the components that will be used to detect potential misuse of the vehicular communications system, and the proposed verification scheme addresses one aspect of misbehavior detection; the broadcast of falsified position information. The proposed position verification scheme models the movement of vehicles on the road network to estimate a vehicle's current position based on its expected movement trajectory according to its location history. The estimated trajectory is enhanced by the integration of the underlying roadmap topology, and is then used to define a plausible geographic area within which a vehicle should reside. The current position announcement of the vehicle is then compared to this plausible area to check if the announcement is consistent with the expected behavior of the vehicle.

The rest of this paper is organized as follows. Section II describes the related work in trajectory-based position verification. Section III briefly describes the envisioned misbehavior detection framework is introduced. In Section IV, the proposed map-guided trajectory-based position verification scheme is explained. Section V includes a description of simulation and results. Finally, Section VI concludes the paper.

## II. RELATED WORK

Trajectory prediction for vehicular networks can be used to predict a vehicle's own future path (*ego* vehicle) to assist with collision avoidance [4] and estimate the risk of collisions [5]. Techniques used for prediction depend either on the utilization of vision-based tracking systems [6], or on the collection of vehicle information via inter-vehicle communication and using

This research is funded by a grant from Natural Sciences and Engineering Research Council of Canada (NSERC).

vehicle dynamics [7] or incorporating a mathematical model of the road geometry and fusing it with a vehicle motion model [5]. All of this work targets the prediction of a future path either for the ego vehicle depending on its detailed sensory input or by relying on the exchanged information among vehicles and assuming that they are correct.

Little work has been done in exploring the deployment of trajectory estimation for position verification in vehicular networks. The scheme in [8] revolves around a Minimum Distance Moved behavioral module, which analyzes and verifies the movement of the sending vehicle relative to the receiving vehicle; the longer the contact between two vehicles the higher the probability that both are moving. The scheme in [9] deals with verifying the trajectory of a vehicle sending a crash alert, and depends on periodic sensing of the monitored vehicle sending the alert.

### III. MISBEHAVIOR DETECTION FRAMEWORK

To secure inter-vehicle communications in vehicular networks, a comprehensive security solution is needed that integrates preventive measures such as authentication, privacy protection, antivirus solutions, and firewalls together with reactive measures to detect the presence of malicious behavior that may be taking place in the system; namely, misbehavior detection. Misbehavior in the context of vehicular networks can be defined as *any significant deviation from the normal behavior of vehicles in the network, with the intent of abusing the system for personal advantage or disrupting the system's proper functioning*. Normal behavior here indicates the expected behavior from an entity conforming to the rules and regulations of the system within which it is operating.

Behavior of vehicles within a vehicular network is represented by the messages they exchange (via vehicular applications), which in turn represent their actions during their existence in the network. Misbehavior detection thus aims at identifying the information conveyed in these messages that indicates behavior deviating considerably from the expected norm; signifying probable malicious intent by the message source (i.e. the vehicle's driver). This can be achieved via defining three main components within a misbehavior detection framework; data sources that represent behavior, data analysis techniques, and reaction mechanisms.

Communications-based applications for vehicular networks depend for their operation on the exchange of three general types of data messages [1]: periodic position beacons to announce the presence of vehicles and help with maintaining traffic flow, event messages triggered upon the occurrence of abnormal road incidents, and infotainment messages related to Internet access and location-based services. Each of these message types needs to be verified upon reception in order to ensure that they hold correct information. As can be seen in Fig. 1, analysis techniques that can be used by a receiving vehicle will differ according to the targeted message type. Position verification aims at analyzing the position information sent by vehicles about their own locations, and thus will be used for position beacons. Event messages; conveying information about a road incident together with its location, need both verification of the event occurrence and its announced location; which will be

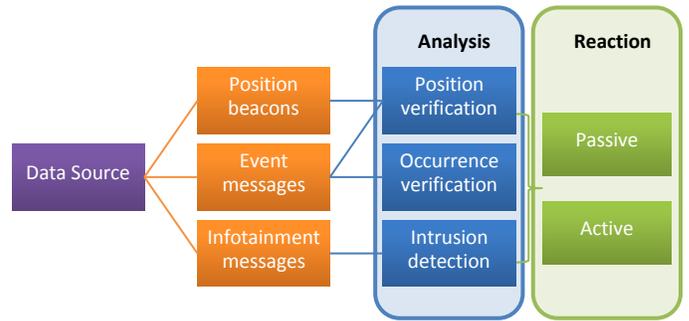


Figure 1. Misbehavior detection framework.

addressed in future work about *event occurrence verification*. Infotainment messages should be verified via proper and already available intrusion detection mechanisms that are adapted to the nature of vehicular networks.

The outcome of an analysis technique is a decision concerning the presence or absence of misbehavior in a message exchange, and thus proper action needs to be taken to alleviate the potentially harmful effects that may be caused by that message. There are two main reaction measures that can be considered by the receiver: passive measures, which merely ignore the message content and do not signal driver to take action based on that message; and active measures, which may involve notifying the corresponding authorities (e.g. the Certification Authority, or CA, that issued the misbehaving vehicle's certificate) so that they can act accordingly (e.g. revoke the certificate or take legal action against the corresponding driver if significant damage could be caused).

In the following section, a scheme is proposed for the verification of position beacons that are exchanged among vehicles in vehicular networks, with the remaining analysis and reaction components to be addressed in our future work.

### IV. MAP-GUIDED TRAJECTORY-BASED POSITION VERIFICATION

In the context of position verification, as a vehicle receives periodic position updates from a set of neighbors  $N$  within its transmission range, it has no access to other vehicles' internal information; only the information they announce periodically. Given the position information of neighbor vehicle  $V_s$  stored as a tuple ( $x$ - $y$  coordinates  $x_{t-1}$  and  $y_{t-1}$ , speed  $s_{t-1}$ , heading  $\phi_{t-1}$ ) by verifying vehicle  $V_r$  at time  $t_{recent}$ , the prediction model at the verifying vehicle will construct a trajectory starting from  $(x_{t-1}, y_{t-1})$  based on  $s_{t-1}$  and  $\phi_{t-1}$  to estimate the  $x$ - $y$  coordinates  $(\hat{x}_t, \hat{y}_t)$  for that neighbor up to time  $t_{current}$ . These estimated coordinates are then compared with the neighbor's announced  $x$ - $y$  coordinates  $(\hat{x}_t, \hat{y}_t)$ , which were received by the verifying vehicle at time  $t_{current}$ . If the difference between the predicted position and the announced position is above a *misbehavior threshold*, the vehicle announcing that position is considered misbehaving and flagged accordingly.

Trajectory prediction can be prone to errors resulting from the probabilistic parameters that are used for path estimation over a prediction interval. In the proposed position verification scheme, *dynamic* vehicle kinematics parameters –received and stored by a verifying vehicle– will be integrated with *static* and deterministic road topology information to estimate the path a

vehicle being verified will take and determine a plausible area within which the vehicle should reside. Road topology information is usually stable and thus can be acquired a priori and stored at each vehicle, and its integration into the trajectory estimation procedure is expected to increase its accuracy, and thus the accuracy of trajectory-based position verification.

The map and road topology information will be processed as follows. For each road segment  $R_i$ , we define three basic parameters: the speed limit of a road segment  $maxS_{R_i}$ , the road width  $W_{R_i}$ , the road length  $L_{R_i}$ , and the road's heading angle  $H_{R_i}$ . Using the speed limit  $MaxS_{R_i}$  of a road segment, the maximum theoretical bounded distance that can be traveled by a vehicle during the beaconing time window  $T_b$ :

$$l_{maxRi} = maxS_{Ri} \times T_b + \frac{1}{2} \times a \times T_b^2, \quad (1)$$

where  $T_b$  is the beaconing interval and  $a$  is the acceleration. Each road segment is then divided into sub-segments of fixed length  $l_{R_i}$ . The rationale behind this division is that it would accommodate the finer changes in the road curves and would enable associating a more detailed road heading angle per sub-segment. The coordinates of each sub-segment's start and end points are recorded as vertices in a customized graph adjacency list, with each entry representing a sub-segment as (ID  $R_k$ , start vertex  $v_{k-1}$ , end vertex  $v_k$ , length  $l_{Rk}$ , heading angle  $h_{rk}$ , ID(s) of next sub-segment(s), number of hops to the next intersection). Intersections are represented as anchor vertices  $v_i$ , with possible multiple sub-segments associated with them.

The proposed position verification scheme is illustrated in Algorithm **Trajectory-Based Position Verification** below.

---

**Algorithm 1: Trajectory-based Position Verification**


---

```

1. Input:
2. Position Beacon  $V_s(x_s, y_s, S_s, \Phi_s)$ .
3. Neighborhood list  $DN_r$  for  $V_r$ .
4. Adjacency list  $MAP$ .
5. Output:
6. T/F decision – is  $V_s$  falsifying its position?
7. begin
8. Search for  $V_s$  in  $DN_r$ .
9. If  $V_s$  not in  $DN_r$  do // New neighbor
10.   Compute  $RSS\_dist$  between  $V_r$  and  $V_s$ ; // Distance according to
    RSS of  $V_s$ 
11.   Compute  $XY\_dist$  between  $V_r$  and  $V_s$ ; // Distance according to
     $(x_r, y_r), (x_s, y_s)$ 
12.   If  $diff(RSS\_dist, XY\_dist) > \tau$  do
13.     return True
14.   else
15.      $DN_r := V_s \cup DN_r$ ; // Add to neighborhood list
16.     return False
17.   endif
18. else //  $V_s$  is in  $DN_r$ , its info is  $V_s(x_{t-1}, y_{t-1}, S_{t-1}, \Phi_{t-1})$ 
19.    $Distance_{probable} := S_{t-1} \times T + \frac{1}{2} a T^2$ ; //  $a$  is acceleration, T is time
    elapsed since most recent beacon entry
20.   Find sub-segment  $R_k$  in  $MAP$  such that location  $(x_{t-1}, y_{t-1}) \in R_k$ ;
21.    $Distance_{max} := maxS_{Rk} \times T + \frac{1}{2} a T^2$ ; // Max distance according to
    speed limit for sub-segment  $R_k$ 
22.    $R_j := R_k$ ; // Start from this sub-segment
23.    $j := k$ ;
24.    $[GeoArea_{start}, GeoArea_{end}] = \mathbf{FindGeoArea}(R_k, Distance_{max},$ 
     $Distance_{probable})$ 
25.   If location  $V_s(x_s, y_s) \in [GeoArea_{start}, GeoArea_{end}]$  do
26.     return False
27.   else
28.     return True
29.   endif

```

---



---

**Algorithm 2: Finding a plausible area for position estimate**


---

```

1. Function FindGeoArea( $R_k, Distance_{max}, Distance_{probable}$ )
2. Input:
3. Start entry  $R_k$  in adjacency list  $MAP$ .
4. Maximum theoretical distance  $Distance_{max}$  traveled by  $V_s$ .
5. Probably distance  $Distance_{probable}$  traveled by  $V_s$ .
6. Output:
7. Plausible geo-area  $[GeoArea_{start}, GeoArea_{end}]$ 
8. begin
9. foreach subsequent  $R_j$  in  $MAP$  do
10.    $GeoArea_{start} = v_{j-1}$ ; // Start vertex of  $R_j$ 
11.   while  $Distance_{max} > 0$  and  $Distance_{max} > l_{Rk}$  do
12.      $Hops\_to\_Anchor_{Rj} := Hops\_to\_Anchor_{Rj} - 1$ ;
13.     If  $Hops\_to\_Anchor_{Rj} = 0$  do //  $R_j$  is last hop before
        intersection
14.       Search for adjacent  $R_{j+1}$  with  $h_{Rj+1} \approx \Phi_{t-1}$ .
15.        $R_j = R_{j+1}$ ;
16.     endif
17.      $Distance_{max} = Distance_{max} - l_{Rj}$ ; // Length of  $R_j$ 
18.     If  $Distance_{probable} > 0$  and  $Distance_{probable} > l_{Rj}$  do
19.        $Distance_{probable} := Distance_{probable} - l_{Rj}$ ;
20.        $GeoArea_{start} = v_{j-1}$ ; // Update start of geo-area
21.     endif
22.      $j := j + 1$ ;
23.   endwhile
24. endfor
25. If  $Distance_{max} > 0$  do
26.    $GeoArea_{end} := v_{j+1}$ ; // End vertex of last reached sub-segment  $R_j$ 
27. else
28.    $GeoArea_{end} = v_j$ ;
29. endif
30. return  $[GeoArea_{start}, GeoArea_{end}]$ 

```

---

Vehicles are assumed to store and update the periodic position information sent by their neighbors. A neighbor's entry will expire if no communication is received from that neighbor for a freshness period  $f$ , in which case the neighborhood is considered "lost". Upon receiving a position beacon from vehicle  $V_s$  at time  $t$ , a verifying vehicle  $V_r$  will check if  $V_s$  has recent position information recorded, say at time  $t-1$  – considered within the *freshness* window but not necessarily within the most recent beaconing window. If no information about the sending vehicle is found, *Received Signal Strength* (RSS) measurements are used to assess the consistency of the distance between  $V_s$  and  $V_r$  based on the announced position with the distance expected to exist based of RSS measurements.

If information about  $V_s$  is found in  $V_r$ 's neighborhood list, the previous  $xy$  coordinates and speed of  $V_s$  will be used to compute the *probable* distance that  $V_s$  may have traveled in the time window  $[t-1, t]$ , as in line 19 in Algorithm 1. The maximum road speed of the road segment on which  $V_s$  was traveling at  $t-1$  is then used to compute the *maximum* distance that can be traveled by  $V_s$  from its previous position (line 21 in Algorithm 1). The adjacency list is consulted to find the sub-segment at which  $V_s$  was at time  $t-1$ . To determine in which sub-segment(s)  $V_s$  should most probably be found if it traveled with the previous and maximum speeds respectively, the probable and the maximum distances assumed to be traveled by  $V_s$  are passed to the function **FindGeoArea**, detailed in Algorithm 2, in order to determine the most plausible sub-segment(s) that  $V_s$  should be in at the current time instance. If the number of hops to the next intersection stored at the starting vertex indicates that the intersection is further than the maximum distance traveled, the series of subsequent vertices are visited and the lengths of their sub-segments are aggregated

until they sum up to or become larger than the *probable distance traveled*, and then to the *maximum distance traveled* (lines 9-23 in Algorithm 2).

If, on the other hand, the starting vertex indicates that the *maximum distance traveled* would involve the inclusion of an anchor vertex, then multiple paths may result if the intersection represented by the anchor vertex has multiple sub-segments associated with it. The same steps as above would be applied, except when the anchor vertex is “visited”, in which case the sub-segment whose heading most closely matches  $V_s$ 's heading would be chosen as the next sub-segment to be followed, as shown in lines 12-15 in Algorithm 2.

The last vertex reached (or vertices, in case the probable and maximum distances yield different target sub-segments) would correspond to the target sub-segment(s) where the most plausible position of  $V_s$  would be. The announced position of  $V_s$  would be compared to the geo-area (line 25 in Algorithm 1). If the announced position of  $V_s$  at time  $t$  is outside of this plausible geo interval, the vehicle is considered to be falsifying its position. In most cases, the time interval  $[t-1, t]$  over which the verification takes place will be equal the beaconing interval  $T_b$ , so the scheme will look into at most two sub-segments as the most probable places where the sending vehicle would be.

## V. PERFORMANCE EVALUATION

In this section, the simulation setup and environment used for implementing the verification scheme are detailed.

### A. Simulation Setup

The proposed scheme was implemented using the NS-2 network simulator. The NS-2 simulation parameters and network configurations are shown in Table I. Simulations are conducted for a period of 1000 seconds over a  $652 \times 307$  area.

Experiments were performed over three different scenarios representing different traffic densities; 50, 150, and 250 nodes. To generate realistic vehicular topology and movements, we used MOVE (MObility model generator for Vehicular networks) on top of the SUMO vehicular simulator. Fig. 2 shows a snapshot of the simulated vehicular scenario. We assume a normal traffic scenario which involves no accidents, major traffic jams, or otherwise abnormal road conditions. The heading angle threshold signifying intent to change routes at an intersection is set to 10 degrees. Acceleration was computed as a function of the last two position updates by a vehicle, and was set to  $3.8\text{m/s}^2$  if no fresh observations were available.

TABLE I: SIMULATION PARAMETERS AND CONFIGURATIONS

Simulation Parameter/Configuration	Value
Number of nodes	50, 150, 250
Max. Node Velocity (Km/h)	40
Number of Lanes/Road	2
Beaconing Interval	1 sec
Beacon Size	64 Bytes
Antenna Type	Omni-directional
Propagation Model	Two-ray Ground
Link-/MAC Layer	802.11

About 10% of the nodes would fake their positions during the simulation, and the falsified positions are generated at random locations within street boundaries. The simulation was run 5 times per scenario with different seeds, and the averages of the performance measures were taken. Three performance measures were used to evaluate the proposed verification scheme: Detection rate, false alarms rate, and false negatives rate. *Detection rate* refers to the percentage of the correctly detected misbehaving nodes that have falsified their positions. *False positives rate* refers to the percentage of the incorrectly flagged nodes that did not falsify their position information. *False negatives rate* refers to the percentage of misbehavior incidents that were missed by the position verification scheme. In addition, the *Mean Square Prediction Error* for each vehicle whose position was estimated was measured as:

$$MSPE = \sqrt{(\bar{x}_t - \hat{x}_t)^2 + (\bar{y}_t - \hat{y}_t)^2} \quad (2)$$

To take into account only the most recent observations of a sending vehicle, a verifying vehicle will only consider the observations reported in the last three beaconing windows.

### B. Simulation results and discussions

The detection rate achieved by the proposed verification scheme is shown in Fig. 3 together with the false negatives rate, since both performance measures complement each other. It can be seen that the scheme achieves a detection rate between 93% and 95%. Analysis of the falsified position beacons that were missed showed that the falsified positions generated by the misbehaving nodes were within the plausibility region defined by the estimated and maximum distances assumed by the receiving vehicles. The scheme shows a consistent performance throughout the three scenarios and is not affected by whether the scenario is sparse or dense.

The false positives rate was measured on two levels; the percentage of nodes that were falsely flagged as misbehaving, and the percentage of nodes receiving beacons and reporting them as falsified. Respectively,

$$\text{False positives rate (nodes)} = \frac{N_f}{N_G}, \quad (3)$$

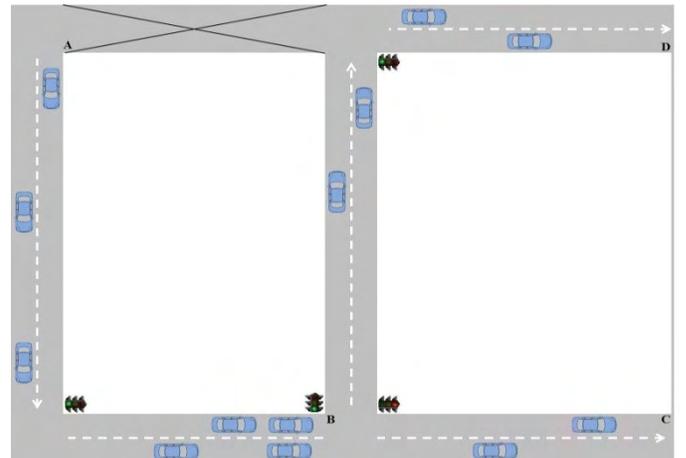


Figure 2: The simulation scenario. The vehicles' injection point is point A. They get separated at point B (intersection); some of them go to a destination at point C and the others head to destination D. Traffic lights are deployed at each intersection to provide more realism to the topology.

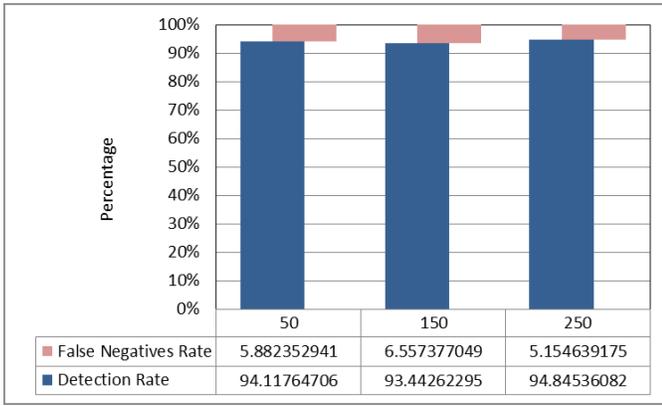


Figure 3: Detection rate and false negatives rate of the proposed scheme for the three simulation scenarios 50, 150, and 250 vehicles.

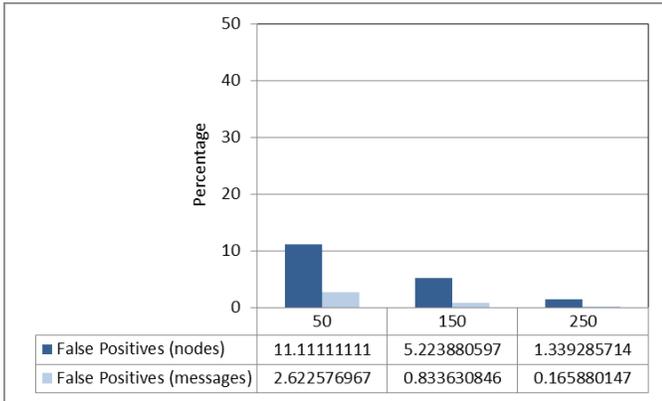


Figure 4: False positives rate (per node count and per beacon count) of the proposed scheme for the three simulation scenarios 50, 150, and 250 vehicles.

where  $N_f$  is the number of nodes falsely flagged as misbehaving, and  $N_G$  is the total number of well-behaving nodes, and

$$\text{False positives rate (beacons)} = \frac{M_f}{M_G}, \quad (4)$$

where  $M_f$  is the number of flags raised by receiving nodes, and  $M_G$  is the total number of “good” beacons received by all vehicles. As shown in Fig. 4, the rate of falsely flagged vehicles decreased as the number of vehicles increased. This was mainly because the percent of well-behaving vehicles in the 50 vehicles scenario (12%) was lower than that of the other two scenarios, both 10%. The rate of alerts raised by vehicles also decreases as the number of vehicles increases. The increasing number of vehicles translates into an increase in neighborhood size, which in turn means that more vehicles get to receive, verify, and flag vehicles. In the sparse 50 vehicles scenario, a vehicle has at most two neighbors, making the volume of exchanged messages considerably smaller than in the other two scenarios and resulting in a higher percentage of false alarms. In the other two scenarios, the neighborhood size grows to 9 and 16 vehicles respectively, and this in turn results in a beacon sent from one vehicle to have a large number of recipients, which should increase the false alarms rate if one vehicle is mistakenly flagged. But since the overall volume of messages exchanged is between “good” nodes, this results in a diminishing rate of falsely flagged vehicles.

The mean squared prediction error was measured for each position received in a position update beacon. For each vehicle, this square error was averaged to assess its associated estimated position accuracy. The prediction accuracy was consistent throughout the scenarios, staying between 5.055m and 6.995m and within the ~12m maximum theoretical distance that is assumed for the simulation experiment. More in-depth analysis of the prediction performance, as well as the effects of the scheme on privacy, are skipped due to space constraints and will be presented in subsequent work.

## VI. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a misbehavior detection framework to be integrated into vehicular networks security measures. We also proposed a position verification scheme for vehicular networks that integrated the roadmap information into vehicles’ trajectory estimation. The proposed position verification scheme is cooperative only in the sense that information exchange is used to help a vehicle construct a temporal movement profile of its neighbors. We showed via simulations that the proposed scheme produces realistic and accurate predictions. We are studying the performance of the proposed scheme in irregular topologies subject to different heading angle thresholds. We are also working on integrating the proposed scheme into the misbehavior detection framework proposed in order to present a comprehensive misbehavior detection solution for vehicular networks.

## REFERENCES

- [1] F. Bai, T. Elbatt, G. Hollan, H. Krishnan, and V. Sadekar, "Towards characterizing and classifying communication-based automotive applications," in *Proceedings of IEEE Workshop on Automotive Networking and Applications (AutoNet)*, 2006.
- [2] S. E. Shladover and S. K. Tan, "Analysis of vehicle positioning accuracy requirements for communication-based cooperative collision warning," *Journal of Intelligent Transportation Systems*, vol. 10, no. 3, pp. 131-140, 2006.
- [3] IEEE, 1609.2: Trial-use standard for wireless access in vehicular environments-security services for applications and management messages, 2006, IEEE Standards.
- [4] A. Polychronopoulos, M. Tsogas, A.J. Amditis, and L. Andreone, "Sensor Fusion for Predicting Vehicles’ Path for Collision Avoidance Systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 3, pp. 549-562, September 2007.
- [5] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *IEEE 5th International Conference on Intelligent Computer Communication and Processing (ICCP)*, August 2009, pp. 417-422.
- [6] F. Han, Y. Tan, and J. Eledath, "Preceding Vehicle Trajectory Prediction by Multi-Cue Integration," in *Preceding Vehicle Trajectory Prediction by Multi-Cue Integration*, Tokyo, Japan, 2007, pp. 575-578.
- [7] P. Lytrivis, G. Thomaidis, and A. Amditis, "Cooperative Path Prediction in Vehicular Environments," in *11th International IEEE Conference on Intelligent Transportation Systems (ITSC’08)*, 2008, pp. 803-808.
- [8] R. Schimdt, T. Leinmüller, and A. Held, "Defending Against Roadside Attackers," 2009.
- [9] M. Ghosh, A. Varghese, A.A. Kherani, and A. Gupta, "Distributed Misbehavior Detection in VANETs," in *Wireless Communications and Networking Conference (WCNC’09)*, 2009, pp. 1-6.