International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Performance Evaluation of a Novel Target Tracking Protocol for Vehicular Networks

Naima Iratni, Ahmed Louazani and Larbi Sekhri

Computer Science Department, University of Oran1, Ahmed Benbella, Algeria
Industrial Computing and Networking Laboratory

Naima.iratni@gmail.com, ahmedlouazani@yahoo.fr, larbi.sekhri@univ-oran.dz

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: Currently**,** for various reasons, authorities are facing the problem of locating a particular vehicle. Some cars are equipped by a geographical positioning system (GPS), but fake or those used in criminal acts have their GPS off. Therefore, locating and target tracking in Vehicular ad-hoc networks (VANETs) is gaining more attention in the research field due to all the envisioned applications that rely on it, such as endlessly tracking a particular vehicle knowing only its license plate number. This is a vital and challenging problem to merge with the dynamic nature of VANETs and the high speed of its nodes. In this paper we try to alleviate this problem by proposing a novel infrastructureless and lightweight tracking algorithm that allows authorities to constantly track a target node. Based on the clustering model, nodes surrounding the target have to collaborate to keep the target in sight and continuously inform the control center (the authorities' base station). A performance evaluation of our proposed solution is performed by using the NS2 simulator.

Keywords: Vehicular Networks, Clustering algorithm, Target tracking, Dissemination, Performance Evaluation.

1. Introduction

Vehicular Ad-hoc Networks (VANETs) are distributed and self-configurable networks. Nodes are vehicles equipped with onboard wireless devices and sensors, able to communicate with other vehicles (V2V) and with roadside units (RSU) (V2I)

Despite the fact that VANETs are a sub-category of MANETs [18], they have become a research field in their own right. Due to their intrinsic characteristics such as high node mobility, intermittent connectivity and frequent topological changes, VANETs are becoming a stimulating and challenging research area that has attracted researchers' interest in the last decade. With the growing development of vehicular on-board sensors and communication technologies, many applications for VANETs are surfacing. These applications can be classified into two categories: i) safety-related applications that improve the safety and reduce the number of accidents (e.g., collision avoidance applications), ii) comfort related applications that enhance the travel experience for drivers and passengers (e.g., video streaming and internet connectivity) [12]

Target tracking is defined as the ability to detect a target and continuously track its state [9]. It is considered one of the promising VANET applications. Although target tracking has been widely studied in WSN [2], the proposed algorithms cannot be dimly mapped to VANETs despite their peculiar characteristics. For this reason, VANET target tracking algorithms present a specific challenge (i.e., highway vehicle tracking or urban areas using vehicles' on-board sensors and VANET communication capabilities). The few existing solutions rely heavily on the pre-installed infrastructure (i.e., RSU, surveillance cameras); deploying such infrastructure is a high-cost solution [1, 9, 10]. Moreover, the target may be lost in areas with no coverage or infrastructure. The proposed algorithms in [3, 4] are independent of the deployed infrastructure and are based on cluster architecture generating a huge overhead for cluster maintenance and cluster reorganization, in addition to the overhead generated by the tracking process.

The shortcomings scheduled in the section of related works dedicated to the field of target tracking in VANETs enable us to develop a novel lightweight and

E-mail:Naima.iratni@gmail.com, ahmedlouazani@yahoo.fr, larbi.sekhri@univ-oran.dz

infrastructureless tracking approach. The main goal of this study is to propose a tool that aids authorities such as military or police departments to trace a target vehicle by means of other vehicles equipped with embedded cameras present in the target vicinity. The proposed solution requires neither pre-installed surveillance cameras along the roads, nor RSU or any dedicated equipment; it takes advantage of the vehicles' embedded cameras present in the vicinity of the target and their communication capabilities without resorting to clusters. Instead, we use a tracking list variable containing nodes' identifiers (IDs) of all the trackers at a given time t in the network to exchange control packets for the network's maintenance

The remainder of the paper is organized as follows: Section2 presents the related work, while Section3 introduces our solution by giving some algorithms that enhance existing ones in the literature. Performance evaluation and simulation results analysis are discussed and presented in Section4. Finally, Section5 concludes this paper.

2. Related Work

Few recent works have tackled the target tracking problem in VANETs to take advantage of communication capabilities (i.e. V2V and V2I) and embedded sensors in these vehicles. The authors in [10] proposed a system to track a moving vehicle in a metropolitan VANET partitioned into zones. The described technique consists of three steps: i) target identification and localization, ii) tracking data collection, and iii) prediction of future target position. When a vehicle detects the target, it begins by collecting tracking data, and then predicts the future position of the target. Therefore, it broadcasts tracking data only to a relevant region where the target is expected to be found in scope of reducing the number of RSUs involved in the broadcasting [14, 15, 16, 17]. However, the proposed solution relies heavily on the employed RSU infrastructure, and consequently it is not operational in the non-covered areas. This solution doesn't take into consideration scenarios in which traffic is limited and/or with low vehicle density. The authors assumed that a car would be available whenever required.

In order to improve the mobility estimation model proposed in the previous work [10], other authors used a conditional Logit estimation model [11]. Although the prediction model was improved, this work still suffered from the same issues as the previous one. In [9], target tracking in VANETs was tackled as an estimation problem. A cooperative tracking system requires a target motion model, the target's position measurements, a data association model and a Bayesian filter from which the target position is predicted; then this estimation is updated based on the measurements. In [1], the authors proposed a protocol allowing authorities to trace any target in urban areas by means of cameras mounted at intersections enabling them to recognize a vehicle by its license plate number (LNB).Their technique was inspired by the concept of trap coverage holes used in wireless sensor networks (WSN) [8]. The coverage holes have limited diameter and are tolerated because the target can only make one known displacement before being detected by a sensor, thus trapping it inside the hole. The analogue of the coverage hole is an area called trap coverage area (TCA) delimited by two intersecting roads in which a vehicle is trapped by means of cameras installed at each intersection. The target is tracked by creating successively new TCA every time it moves from a TCA to another one. However, this work heavily depends on the pre-installed infrastructure of security cameras, since the authors assumed that each intersection would be equipped with a camera and a roadside router to manage it, and it works only on the Manhattan road topology. The authors did not take into account the possibility that the target might park while being tracked.

Khakpour et al. proposed two clustering schemes for target tracking in VANETs using built-in cameras on vehicles able to detect a target by its visual features [3, 4]. Nodes detecting the target and those with a high probability of detecting it in the near future form a cluster moving along with the target in order to be able to track it continuously. The same approach was proposed for connectivity maintenance in VANET [5]. In [3], the former work was a distributed algorithm using a cluster head selection metric named tracking failure probability (TFP) to estimate mobility similitude (velocity vector, distance from target) between nodes and target. The latter work is a centralized clustering algorithm in which the cluster head (CH) is the central entity [4]. For CH election metric, prediction was used to calculate how much time the target will stay in the field of view of each node. Despite the fact that this work does not depend on the employed infrastructure, it generates a lot of overhead for cluster maintenance and re-clustering in addition to the overhead generated by the target tracking process itself.

3. Protocol's Description

The following section provides a detailed description of the main solutions that are evaluated in this paper. Our basic idea is to keep tracking the target by only one node although it is tracked by more than one to reduce network traffic. A solution was widely presented that started by defining the algorithm assumptions, followed by variables and packet exchange description.

A. Definitions and Assumptions

In the proposed algorithm we assume that each vehicle is equipped with a front and a rear camera, and is capable of identifying another vehicle by its license plate number (LPN). The LPN doesn't exceed 15 characters and uses one of the image processing algorithms given in [7, 6, 13] to detect an LPN. The algorithm runs continuously in the background during the tracking process, and nodes are identified using only their MAC addresses. We also assume that a control center (CC) such as a police station initiates the tracking process by broadcasting a target tracking request (TTR packet) containing the target LPN.

Nodes in the area of interest receive the TTR using one of the broadcasting algorithms proposed in the literature [13, 14, 15, 16]. The distance from the target is visually calculated by an image processing algorithm. This distance is combined with digital maps and GPS to calculate the target position. Our algorithm enables us to track only one target at a time. Exchanged packets in this algorithm are described as follows:

- Target Tracking Request (TTR) as shown in Figure 1: this packet is broadcast in the network by the CC to initiate the tracking process.

Source ID	Packet type	LP

Figure 1. Target Tracking Request (TTR)

- Tracking Detection Packet (TDP) depicted by Figure2: this packet is broadcast by trackers that joins or leaves the tracking process to inform other tracker nodes in the network to update their tracking list.

Source ID	Packet type	TDV	MSM	Source MTN

Figure 2. Target Detection Packet (TDP)

- Target Information Packet (TIP) illustrated by Figure3: this packet contains target information collected by trackers and sent to the control center (CC).

Source ID	Packet type	Data

Figure 3. Target Information Packet (TIP)

Terms used in this paper are defined in Table I while information about each message field is depicted in Table II.

TABLE I. TERMS DEFINITION TABLE

CC	Control Center
TDF	Target Detection Flag
MSM	Mobility Similitude Metric
TN	Node that detects the target
TL	Tracking List
V(i)	Network Node
LPN	License Plate Number

TABLE II. TTR, TDP AND TIP PACKETS STRUCTURE

	Field	Size	Definition
TTR	ID	48 byte	Source ID
	Packet type	2 bit	10 for TTR
	LP	15 byte	license plate number
TDP	ID	48 byte	Source ID
	Packet type	2 bit	01 for TDP
	TDV	1 bit	1 if T in FOV else 0
	MSM	*	MSM value
	Main TN	1 bit	1 if node is MTN else 0
TIP	ID	48 byte	Source id
	Packet type	2 bit	11 for TIP
	Data	>2048 byte	position, images

Tracking list (TL)

During the initialization phase each tracking node (TN) creates a list TL of node's IDs (variable). This is used throughout the tracking process and contains IDs of all other TNs present in the target's neighborhood; thus, each TN remains aware of the state of the target's neighborhood (number of TNs and their IDs) during the tracking process.

To efficiently ensure the tracking process, the TL must be kept up to date; this is done systematically when a node joins or leaves the tracking process using TDP packets. Whenever a node detects the target it will broadcast a TDP packet with TDV field set to true; for example, other TNs in the network will add its ID to their TL (see Section 3.2). When a TN loses the target, it multicasts to its TL members a TDP packet with TDV field set to false such that it's ID will then be removed from all TL (see Section 3.3).

Mobility Similitude Metric (MSM)

MSM is a metric describing the mobility similitude of a node compared to the target's mobility [4]. More node's mobility and target's one are similar smaller this value is. To do this, we will use velocity vector rather than speed only. We define:

Node vector velocity $\overrightarrow{V_n}(S_n, \theta_n)$ (1)

Target vector velocity $\overrightarrow{V_n}(S_t, \theta_t)$ (2)

Where S_n and S_t are the speed of vehicle n and the target T respectively, θ_n and θ_t are the velocity angles of vehicle n and target T with X axis respectively. We also define:

$$\left\|\overrightarrow{V_{nt}}\right\| = \left\|\vec{V_n} - \vec{V_t}\right\| = \|(S_n - S_t, \theta_n - \theta_t)\| \quad (3)$$

It does not matter if the values $S_n - S_t$ and $\theta_n - \theta_t$ are negative because we calculate the magnitude of $\left\|\overrightarrow{V_{nt}}\right\|$ which describes the divergence between nodes' and targets' velocity.

D$_{nt}$ is the distance between node n and target T. We divide it on the node's view field range (r) to normalize it because vehicles are heterogeneous with different view ranges of fields, allowing us to apply the same threshold for all TNs in the network (see Section 3.3).

$$\frac{D_{nt}(m)}{r(m)} \quad (4)$$

MSM of node *n* at time *t* is described by equation 5, where α and β are coefficients of distance and velocity respectively, and are used to ensure the effectiveness of MSM. In highways there are no intersections, so the change in velocity is almost non-existent, unlike in urban areas, hence the importance of using α and β coefficients.

$$MSM(n)_t = \alpha.\frac{D_{nt}}{r} + \beta.\left\|\vec{V}_{nt}\right\| \qquad (5)$$

B. Initialization Phase

Both control center (CC) and nodes have to initialize their tracking process to define the parameters' initial values to trigger the target node tracking process. CC initiates the tracking process by periodically broadcasting the target LPN within a TTR packet in the network until it receives the first TIP packet. Algorithm 1 describes the CC initialization procedure.

Algorithm 1: Control Center: Initialization

```
1.  Begin
2.  While Not(Received TIP) do
3.  Broadcast TTR
4.  Timer ← get-time ( )
5.  // the CC will wait for the first TIP packet during the timer
6.    While Not (Received TIP) and(Timer>0)do
7.        receive
8.        -- Timer
9.    End while
10. End while
11. End
```

Once a node receives a TTR from the CC it triggers the images processing algorithm to detect the target. Once the target is detected, the node will:

a) Set TDV value to true, (this value is set to false when the TN loses the target) [4].

b) Send the first TIP packet to the CC directing it to cease broadcasting TTR. The TIP message contains TN's location, since the target must not be very far from the TN and, at this moment, it seems hard to identify the exact position of the target. The dedicated function to calculate the target's location and take pictures of it will be launched later in the tracking phase (see Section 3.3).

c) Create a tracking list (TL), afterward it will broadcast a TDP packet with TDV field value equal to true and its MSM value computed at time *t* (time of the TDP creation) informing other TNs in the network that the target is or has been detected, and waits for TDPs from other potential TNs in the network to add their IDs in its TL.

At the end of the initialization phase, each TN will have a TL with the IDs of all the trackers in the network and their MSM values at time *t* respectively. The TL is updated systematically whenever a node joins or leaves the tracking process. Other nodes (not TN nodes) that receive the TTR keep it in case they detect the target later, but discard broadcasted TDPs. Algorithm 2 describes the node initialization process.

Algorithm 2: Node: Initialization

```
1.  Begin
2.  If ((Received TTR) and (TDV=true)) then
3.  // as soon as the TN detects the target, it sends the first TIP message to the CC with its own position
4.  (Node Id; PacketType; Data) ← (TN Id; 11; my position)
5.  send TIP to CC
6.  Create TL
7.  (Node Id; PacketType; TDV; MSM; MTN) ← (my ID; 10; true; MSM value; false)
8.  // broadcast TDP and wait for TDPs from other TN to add their Ids to the TL
9.  Broadcast TDP
10. Timer ← time
11. While Timer >0 do
12.       If (Received TDP with TDV=true) Then
13. // MTN value = true if the source of the received TDP is Main TN else MTN value = false
14.       Add (Source Id; Source MSM (t); MTN value) to TL
15.       Exit while
16.       End If
17.       --Timer
18. Endwhile
19. ElseIf ((Received TTR) and (TDV = False)) then
20.       Store target's information in case the node will detect it later
21.       forward TTR
22.       EndIf
23. EndIf
24. End
```

C. Tracking Phase

In this phase, nodes start by seeking the target according to the LPN launched by the CC via the TTR packet done in the previous initialization phase. Algorithm 3 describes the tracking process and is executed by each TN node after the initialization phase. During this phase one node among the TNs will be elected as a cluster head to send the TIP messages to the CC, avoiding generating considerable overhead. We will call this node Main Tracker Node (MTN). Nodes will also monitor the values of MSM metric, and wait for TDP messages from nodes joining or leaving the tracking process. These two tasks will be executed concurrently.

To monitor MSM values efficiently, each TN must periodically compute its MSM in every period t_i (every 1s) and compare it to a fixed threshold (equal to 0.6) obtained by the simulation operation. A threshold is a MSM value from which we consider the TN will soon lose the target from view. The values MSM monitoring process allows us to know when the TN is about to lose sight of the target.

If MSM is less than the threshold this means that TN has a similar movement to the target, thus it will not lose the target anytime soon, so the nodes keep the tracking process. Otherwise, TN is about to lose sight of the target, so each TN will check its TL. If the TL list is not empty (there are other TNs in the network) the TN will leave the tracking process and:

- Set TDV to false,
- Send TDP packet within TDV sets to false and MTN (if MTN is true and the node is leaving the tracking process, another one will be elected),
- Stop Fn-Tracking(),

If the tracking list is empty then TN node is the only tracker in the network so it will leave the tracking after:

- Setting TDV to false,
- Sending an alert message to the control center warning it to restart the tracking process by broadcasting a TTR in the network,
- Stop Fn-Tracking(),

```
Function Fn-Tracking ()
1. Begin
2. While TDV = True do
3. take pictures of the target
4. calculate the position of the target
5. (Node Id; Packet Type; Data) ← (TN Id; 11;(picture; position)
6. send TIP to CC
7. End while
8. End
```

Meanwhile, TL must be updated and MTN has to be reelected when the current MTN is about to leave the tracking process. While tracking, TN node will receive TDP messages from nodes joining or leaving the tracking process.

When a TN receives a TDP with TDV value set to true from a node joining the tracking process, it will insert the tuple (source ID, source MSM, source MTN) to its TL.

When a TN node receives a TDP packet with TDV value set to false from a node leaving the tracking process, it will:

- Be deleted from the TL if the leaving node is not the MTN,
- If the leaving node is the MTN then a new MTN has to be elected. Each node will compute its new MSM and send it to other TNs in its TL. Then all nodes will update their TL with the newest MSM values. Each TN will compare its MSM with the MSM of other TNs in its TL. If it has the best MSM, then it will be elected as the MTN and launch Fn-Tracking(). The elected MTN will remain until it loses sight of the target even if another TN with better MSM joins the tracking process to avoid generating overhead.

```
Algorithm 3: Node-Tracking
Input: MTN = False, TDV = True
1 Procedure: MSM monitoring
2. Begin
3         every $t_i$ do                    // each period of time = $t_i$
4         If (MSM>threshold) Then// if (MSM < threshold) then the node continue the tracking
5                 TDV = False;
6                 If (TL is not empty) Then
7                         If (MTN) Then //(MTN = True)
8                                 TDP ← (my ID; 01; False; 00; True)
11                        Else/ (MTN=False)
12                                TDP ← (my ID; 01; False; 00; False)
14                        End if
15                        Send TDP to TL members
16                Else// (TL is empty)
17                        Send alert message to CC
18                End if
19                Stop Fn-Tracking()
20        End if
21 End

22 Procedure: TL maintenance
23 Begin
24        If (received TDP with TDV = True) Then
25                Insert_into_TL (src ID, src MSM, src MTN)
26        Else// (received TDP with TDV = False)
27                If (src MTN = True) Then
28                        delete_from_TL TN with ID = src ID from TL
29                        calculate MSM(t)
30                        TDP ← (my ID; 01; True;MSM(t); false)
31                        send TDP to TL members
32                        timer ← Get-time()
33                        While (timer > 0) do
34                                If (received TDP with TDV=True) Then
35                                        update TL with new MSM values
36                                End If
37                        Endwhile
38                        If (My MSM is the best) Then
39                                launch Fn-Tracking()
40                        End If
41                        Elseif (src MTN=False) Then
42                                delete_from_TL TN with ID = src ID from TL
43                        End If
44                End If
45        End If
46 End

47  Main :
48  Begin
49        While (TDV = true) do// if the initialization phase is finished and TL is empty (there is only one tracker in the network)
50                If (TL is empty) Then
51                        launch Fn-Tracking();
52                Else // execute in parallel
53                        MSM monitoring;
54                        TL maintenance;
55                End if
56        End while
57  End
```

D. *Control center tracking process*

When the CC receives the first TIP, it will cease broadcasting the TTR packet in the network, and will wait for target information (snapshot and location). The CC sets a countdown tracking timer variable *TIP-timer* which is initialized every time the CC receives target information. If *TIP-timer* expires before the CC receives target information, it will assume that there is no tracker in the network thus it will restart the process by rebroadcasting the TTR packet. Algorithm 4 describes the CC's Tracking process.

Algorithm 4: CC-Tracking

```
1   **Begin**
2   **Wait** for TIP
3   start TIP-timer
4   **If** (Received TIP = True) **Then**
5   initialize (TIP-timer)
6   Treat and store target information
7   **Else If** (TIP-timer expired or Alert message received)**then**
8           **Call** algorithm 1
9           **End If**
10  **End If**
11  **End**
```

4. Simulation and Results

NS2 simulator is used to evaluate packet delivery ratio and end-to-end delivery delay. We made slight modifications on AODV routing protocol focusing on two parameters: network size or density and node mobility. Files were adjusted in order to study the new solution behavior in which velocity angle function is implemented referring to these assumptions.

Let (x_{c0}, y_{c0}) and (x_{c1}, y_{c1}) be the node positions at time t_0, t_1 respectively. The velocity angle is defined by the equation:

$$\theta = tang^{-1}\left(\frac{y_{c1}-y_{c0}}{x_{c1}-x_{c0}}\right) \quad (6)$$

We deal with two situations: first we consider the network size invariable and node speed (velocity) variable, while in the second situation velocity is fixed and network size is variable.

Case 1: Network size is fixed to 20 nodes, target one (id=1) and the RSU (id=0) as shown in Table III. The obtained results are depicted by Figures 4 and 5 below.

TABLE III. Simulation parameters

Parameters	Value
Routing Protocol	Tracking protocol
MAC layer	MAC /802_11.p
Node speed	23, 26, 29, 32, 35,38 m/s
Number of nodes	20 + (CC node + Target node)
Target node	Id = 1
CC node	Id = 0
Simulation time	200 s

As shown in Figure 4, we remark that packet delivery ratio curve nearly changed (it is a steady function of node velocity) although the node speed changes from 23 m/s to 38 m/s. So we realize that first our proposition is not influenced by the node's speed in the network, and second it operates in both urban zones and highways. Indeed, the node speed does not affect the end-to-end delay as depicted by Figure 5. The curve is a slightly horizontal line.

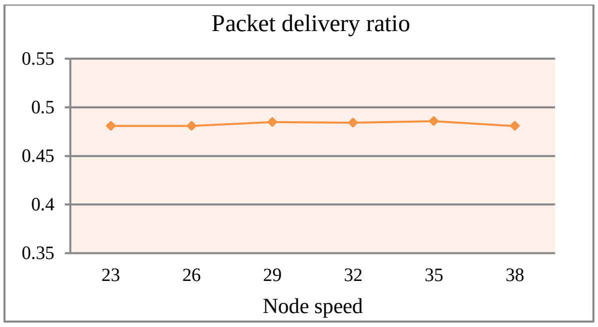

Figure 4. Nodes' velocity versus packet delivery ratio

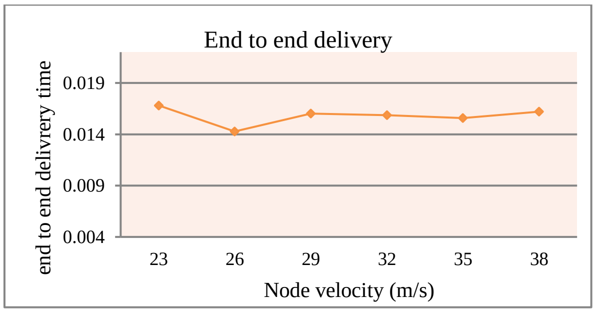

Figure 5. Nodes' velocity versus end-to-end delay

Case 2: We fix nodes' speed at 27 m/s and varying network size from to 20 to 70 nodes. Target one (id=1) and RSU (id=0). Table IV lists simulation parameters. The obtained results are given by Figures 6 and 7 below.

TABLE IV. SIMULATION PARAMETERS

Parameters	Value
Routing protocol	Tracking protocol
MAC layer	MAC /802_11.p
Node speed	27 m/s
Network size	20, 30, 40, 50, 60, 70
Target Node	Id = 1
CC Node	Id = 0
Simulation time	200 s

Adding a node to a network produces more overhead packets and enhances the probability of packet loss factor. Figure 6 highlights the fact that more packets are lost (packet delivery ratio decreases) whenever the network size grows. According to Figure 7, we notice that adding more nodes to the network raises the end-to-end delay. This is justified by the time elapsed by the packets exchanged between lined neighbors. Whenever you add a node, you increase the time needed for packet delivery. We conclude that our solution works well in a smaller sized network.

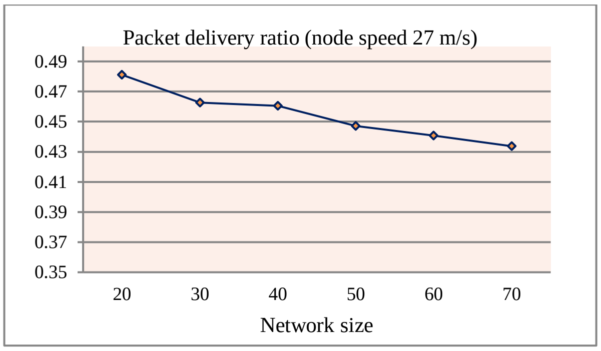

Figure 6. Network size versus packet delivery ratio

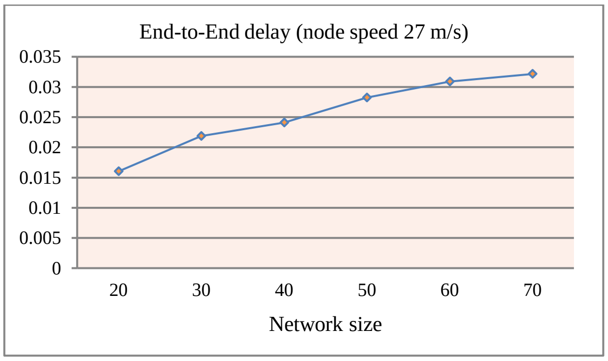

Figure 7. Network size versus end-to-end delay

5. CONCLUSION AND FUTURE WORKS

Our objective behind this paper is to overcome some works' limitations in the literature dealing with target tracking for VANETs. Our contribution is to develop a new tracking approach for vehicular networks by proposing new algorithms. The approach does not require any preinstalled infrastructure such as RSUs and surveillance cameras or dedicated equipment. Instead it leverages vehicles present in the vicinity of the target by using their onboard cameras and sensors. As shown by the simulation results, our solution is not consequently influenced by the nodes' speed and works in both urban zones and highways. We have introduced the concept of tracking list enabling tracker nodes to be aware of the network 'state at any time during the tracking process. In future work, we aim to address the overhead issue, in order to make our solution more suitable for large sized networks and develop more experiments and comparison studies.

REFERENCES

[1] P.Bellavista, A. Boukerche, T. Campanellaand L. Foschini, "The Trap Coverage Area Protocol for Scalable Vehicular Target Tracking," IEEE Access, vol. 5,pp. 4470–4491, 2017.

[2] S. Bhatti and J. Xu, "Survey of Target Tracking Protocols Using Wireless Sensor Network,"Proceeding of ICWMC 2009, pp.110–115, 2009.

[3] S. Khakpour, R. W. Pazzi and K. El Khatib, "A prediction based clustering algorithm for target tracking in vehicular ad-hoc network," 4th ACM Press, pp. 39–46, 2014.

[4] S. Khakpour, R.W. Pazzi, and K. El-Khatib, "A distributed clustering algorithm for target tracking in vehicular ad-hoc networks," 3rdACM Press, pp. 145–152, 2013.

[5] A. Louazani and L. Sekhri, "Petri Net Model for Connectivity Maintenance in VANET Clustering-Based Routing Algorithm," International Conference on Advanced Aspects of Software Engineering (ICAASE'2016), Constantine, Algeria; October 29-30, 2016.

[6] B. Li, B. Tian, Ye Li, and D. Wen, "Component-Based License Plate Detection Using Conditional Random Field Model," IEEE Transactions on Intelligent Transportation Systems, Vol. 14, issue. 4, pp.1690–1699, December 2013.

[7] K. Lin, H. Tang and T. S. Huang, "Robust license plate detection using image saliency," 2010 IEEE International Conference on Image Processing, pp. 3945-3948, Hong Kong, 2010.

[8] P. Balister, Z. Zheng, S. Kumar and P. Sinha, "Trap Coverage: Allowing Coverage Holes of Bounded Diameter in Wireless Sensor Networks," IEEE INFOCOM 2009, pp.136-144, Rio de Janeiro, 2009.

[9] H. S. Ramos, A. Boukerche, R. W. Pazzi, A. C. Frery, and A. A. F.Loureiro, "Cooperative target tracking in vehicular sensor networks,"IEEE Wireless Communications, vol. 19, no. 5, pp. 66–73, Oct 2012

[10] T. A. Reza, M. Barbeau and B. Alsubaihi, "Tracking an on the run vehicle in a metropolitan VANET," 2013 IEEE Intelligent Vehicles Symposium (IV), pp. 220-227, Gold Coast, QLD, 2013.

[11] T. A. Reza, M. Barbeau, G. Lamothe and B. Alsubaihi, "Non-cooperating vehicle tracking in VANETs using the conditional logit model," 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, pp. 626-633, 2013.

[12] Y. Wang and F. Li, "Vehicular Ad Hoc Networks,"book chapter, Guide to Wireless Ad Hoc Networks, Computer Communications and Networks; pp. 503–525. Springer London, London, 2009.

[13] W. Zhou, H. Li, Y. Lu and Q. Tian, "Principal Visual Word Discovery for Automatic License Plate Detection,"in IEEE Transactions on Image Processing, vol. 21, no. 9, pp. 4269-4279, Sept. 2012.

[14] H. Alshaer and E. Horlait, "An optimized adaptive broadcast scheme for inter-vehicle communication," 2005 IEEE 61st Vehicular Technology Conference, vol 5, pp. 2840-2844, Stockholm, 2005.

[15] S. Ahmed, S. Ariffin, N Fisal, S. Syed-Yusof and N.Latif, "Survey on Broadcasting in VANETs," Research Journal of Applied Sciences, Engineering and Technology; pp. 3733-3739, May 2014.

[16] E. Fasolo, A. Zanella and M. Zorzi, "An Effective Broadcast Scheme for Alert Message Propagation in Vehicular Ad hoc Networks," 2006 IEEE International Conference on Communications (ICC'2006), pp. 3960-3965, Istanbul, 2006.

[17] O. Tonguz, N. Wisitpongphan, F. Bai, P. Mudalige and V. Sadekar, "Broadcasting in VANET," 2007 IEEE Mobile Networking for Vehicular Environments, pp. 7-12, Anchorage, AK, 2007.

[18] S. Yousefi, M. Mousavi, and M. Fathy, "Vehicular Ad Hoc Networks (VANETs): Challenges and Perspectives," In 6thInternational Conference on ITS Telecommunications, pp. 761–766, Chengdu, China, June 2006.

Naima Iratni, obtained her Master in 2015 and actually she is a PhD candidate in the Department of Computer science, Oran University; she is member of Industrial Computing and Networking laboratory (ICNL). Her current research includes intelligent transportation systems and Protocols Modeling.

Ahmed Louazani is an associate Professor at the Computer Science Department of Relizane University. His current research area of interests include Internet of things, formal modeling using Time Petri Nets, in distributed and mobile systems, Intelligent transportation systems and sensor networks. He is member of the Industrial Computing and Networking Laboratory at Oran University.

Larbi Sekhri is a Professor at the Computer Science Department of Oran University. His current research area of interests include Internet of things, formal modeling in distributed and mobile systems, Intelligent transportation systems and sensor networks, systems modeling using Petri nets, diagnosability and monitoring of automated production systems. He is member of the Industrial Computing and Networking Laboratory at Oran University. He has been a visiting professor at Cedric-CNAM research laboratory, in Paris, France, and Ecole Centrale de Lille (LAGIS) where he worked in Diagnosis of Industrial systems; and LIUPA Laboratory at the University of Pau, France; and distinguished lecturer at PARADISE laboratory, University of Ottawa, Canada.