A Position-based Routing Module for Simulation of VANETs in NS-3

Konstantinos Katsaros, Mehrdad Dianati
Centre for Communication System Research
University of Surrey, United Kingdom
{K.Katsaros, M.Dianati}@surrey.ac.uk

Karsten Roscher
Fraunhofer ESK
Munich, Germany
karsten.roscher@esk.fraunhofer.de

ABSTRACT

Geonetworking and the corresponding routing protocols play an important role in application of Vehicular Ad-hoc Networks (VANETs). This paper presents the architecture and implementation of CLWPR (Cross-Layer, Weighted, Position-based Routing), a position-based routing protocol optimized for VANETs in NS-3 simulation environment. It utilizes mobility information from nodes and cross-layer information from PHY and MAC layer in order to increase its efficiency and reliability, respectively. More specifically, we present the changes that have been introduced to facilitate geonetworking, as well as the carry-n-forward mechanism. In addition, we developed a basic navigation facility that can provide information about the road and distance between two nodes.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: [Distributed networks, Store and forward networks,Wireless communication]; C.2.2 [**Network Protocols**]: [routing protocols]; I.6.5 [**Simulation and Modeling**]: [Model development, modeling methodologies]

General Terms

Algorithm, Design

Keywords

position-based routing, vehicular ad-hoc network, cross-layer

1. INTRODUCTION

Intelligent Transportation Systems (ITS) aim to apply Information and Communication Technologies (ICT) to improve safety and efficiency as well as the passenger experience in modern transport systems. It is envisaged that dynamic vehicular networks, particularly, Vehicular Ad-hoc Networks (VANETs) will be an important part of the future ITS. VANETs are a category of Mobile Ad-hoc Networks (MANETs) where the nodes are vehicles and roadside units. The differences that distinguish VANETs from MANETs are the lack of strict energy constraints, the high mobility of the nodes (vehicles) constrained by the road topology, relatively short lived communication links and the characteristics of the communication channel (path loss and fading due to buildings and other vehicles). For these reasons, routing protocols designed for MANETs exhibit unsatisfactory performance as presented in [16]. Therefore, position-based routing which exploits mobility of nodes is employed to increase efficiency and reliability of these networks.

Simulation based study of VANETs is important and cost effective method of evaluating the impacts of deployment of different applications, as well as, designing effective protocols for VANETs. Such simulation scenarios have specific characteristics including large number of nodes, high mobility and different communication facilities. Moreover, in order to simulate realistic VANET scenarios, different simulators have to be coupled, including traffic, driver, network and others. Platforms for coupled simulators such as VSimRTI [14] and iCS from iTETRIS [2], are available and provide open interfaces for different network simulators. Therefore, a network simulator that is capable of providing such scenarios with the best performance in terms of computation time, memory usage and advanced communication facilities as well as open interfaces for coupling with one of the aforementioned platforms is preferable. Recent research on network simulators suggests that NS-3 is one of the fastest and more efficient for large scale networks, such as VANETs [17]. However, NS-3, being relatively new, does not support many routing protocols for ad-hoc networks. Currently only three are implemented, OLSR [5], AODV [12] and DSDV [13], none of which are based on position information which is the preferred routing mechanism in VANETs.

This paper aims at presenting the architecture and implementation of CLWPR (Cross-Layer, Weighted, Position-based Routing) protocol in NS-3[1]. This protocol is a unicast position based routing protocol optimized for VANETs, utilizing mobility information in order to increase efficiency and cross-layer information from PHY and MAC to increase its reliability. It supports caching of packets with the *carry-n-forward* mechanism to overcome lack of network connectivity. Initial results of its performance analysis are presented in [9].

The remainder of the paper is organised as follows. In Section 2, we present related work on position-based routing.

WNS3 2012, March 23, Desenzano del Garda, Italy

DOI 10.4108/icst.simutools.2012.247678

---

[1] The source code for the module is submitted for code review at `http://codereview.appspot.com/5343044`

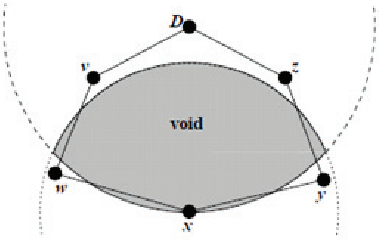


Figure 1: Local maximum in GPSR [8]

In Section 3, the routing architecture in NS-3 is presented while Section 4 describes the proposed protocol with the appropriate enhancements in NS-3 architecture. The use of CLWPR and possible configurations are presented in Section 5. Finally, in Section 6 we conclude our work.

2. RELATED WORK

The early position-based routing protocols were introduced in the 1980's. The best known position-based routing protocol is GPSR[8]. It has two modes of forwarding, Greedy and Perimeter. The first selects the the next hop based on minimum Euclidean distance from the destination, while the second is selected when the forwarding node is faced with *local-maximum*, and the packet is forwarded along the perimeter of the void. This is the case when the forwarding node (Node x in Figure 1) is the closest node towards the destination Node D within the communication range of Node x. GPSR was developed for generic MANETs, therefore it performs badly in VANETS [10]. CAR [11] on the other hand, was designed specifically of inter-vehicle communications. It exploits information learnt through the route discovery process, therefore it is capable to maintain connected paths adjustable on-the-fly, through the utilization of "guards" to keep track of the current destination position. A set of protocols for VANETs, called VADD, are presented in [18]. The key idea of these protocols is the *carry-n-forward* mechanism which is employed when a node is faced with *local-maximum*. The difference from previous protocols using this mechanism, is the utilization of node mobility and the predictable trajectories of the nodes. Finally, GyTAR [7] exploits vehicle traffic information to optimize the forwarding selection in an urban vehicular network. GPSR, CAR and VADD are implemented in NS-2, while GyTAR is implemented in Qualnet and only GPSR's source code is publicly available [1]. Recently, iTETRIS [2] project, which was focusing on integration of vehicular communications and traffic simulation, announced that they provide their platform to the public. They used NS-3.7 as their communication simulator and they developed an infrastructure assisted position-based routing protocol [4] which exploits the reliable backbone connection between infrastructure nodes to increase its performance. Since this protocol is highly integrated with the iTETRIS platform and NS-3.7 was not nodular, it is not reusable for other applications.

Additionally, recent research is focused on cross-layer techniques in order to increase the reliability of the routing protocol. The key idea behind this is to use information about the link quality and channel state (e.g. SINR, MAC retransmissions) to estimate the reliability of that link and act accordingly at the routing protocol. Using information about the received signal strength and arrival time of packets at the PHY, authors in [15] calculated the Link Residual Time (LRT) metric. This is an indicator of the remaining time that the specific link can be used for transmission. LRT is "exposed" to upper layers, such as routing. However, calculating LRT is not trivial. It requires removal of the noise from the data, estimation of the model parameters and finally renewing LRT. The advantage of this approach is that is generic; LRT can be used by any other upper layer. PROMPT [6] on the other hand, is a geographic routing protocol which has a bi-directional cross-layer design. It is developed for Vehicle-to-Infrastructure applications and provides (a) delay-aware routing through traffic statistics collected in MAC and (b) robust relay selection at MAC layer supported by mobility information from NET. Again, PROMPT was implemented in NS-2, while LRT was evaluated in Matlab.

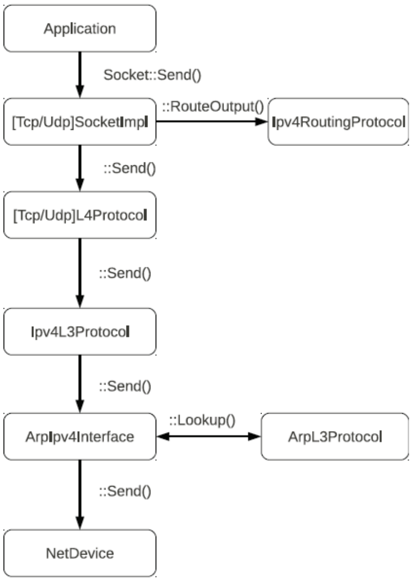


Figure 2: Flow chart NS-3 internet stack (Send)[3]

3. OVERVIEW OF NS-3 ROUTING

The architecture of NS-3 internet stack and routing for IPv4 is described in its manual [3] and represented with Figures 2 and 3 for sending and receiving a message, respectively. Any implementation of a routing protocol in NS-3 should provide two methods declared in `ns3::Ipv4RoutingProtocol` base class, `RouteOutput` and `RouteInput`. The first will provide to the transport protocol a valid route towards the destination for a specific packet. On the other hand, the latter is called by `Ipv4L3Protocol` when a packet is received from a NetDevice and appropriate action is taken to forward it either to upper layers (local delivery) or to other nodes (forwarding).

4. CLWPR SPECIFICATIONS

In contrast to the work done by the iTETRIS project, the proposed implementation of the position-based routing protocol CLWPR focuses on modularity and thus reusability of the code. The initial code was from OLSR module imple-

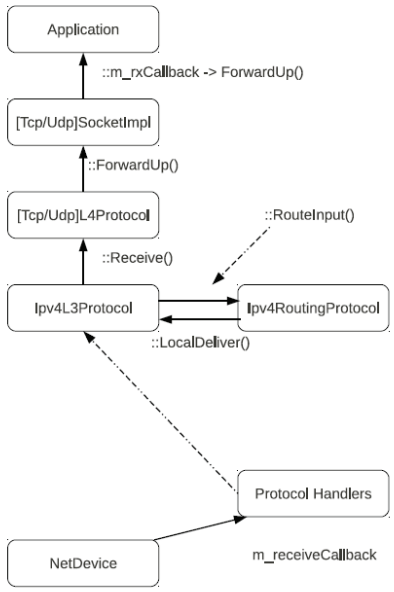


Figure 3: Flow chart NS-3 internet stack (Receive)[3]

mentation which is a pro-active ad-hoc protocol with periodic "HELLO" messages and provides also the HNA (Host and Network Association) facility, which will be utilized for interconnection with infrastructure and other networks in the future. Several parts of the protocol had to be altered or implemented from scratch for the development of CLWPR and are described in the following subsections. First of all, in Subsection 4.1, we describe the modification of neighbor repository and the addition of position association. In Subsection 4.2, we present the changes of "HELLO" message in order to provide position information for the nodes, and in Subsection 4.3, the changes in the routing table calculation. Finally, in Subsection 4.4, we present the enhancements that were introduced to exploit navigation and cross-layer information as well as the *carry-n-forward* mechanism to improve efficiency and reliability of the protocol.

4.1 Repositories

There are two basic repositories in CLWPR, the neighbor set and the position association set.

Neighbor Set is a modification of the `NeighborSet` found in OLSR to facilitate additional information such as position, velocity, heading, etc, as it will be described in the following subsections. This repository is maintained in order to know the 1-hop neighbors. Each time we receive a "HELLO" message we look into this repository and either update an entry or create a new entry with the elements presented in Table 1. Each entry will be automatically deleted after a period of 2.5*HelloInterval time in order not to keep neighbors that are not in the vicinity but also hold the information in case a "HELLO" is lost.

Position Association Set keeps track of the position information of any destination that the specific node has data to deliver, a local copy of location service. If there is an entry in this set, that is also a neighbor, it is updated with the "HELLO" messages. Otherwise, a location service - currently the global view of the simulator - will provide this information to maintain the repository. The information stored in this set are described in Table 2.

Table 1: Neighbor entry for CLWPR

`Main Address`
`Interface`
`Status`
`Position`
`Velocity`
`Heading`
`RoadID`
`Utilization`
`MAC Info`
`SNIR info`
`CnF info`
`Times-tamp`
`Expire-Time`

Table 2: Position association entry for CLWPR

`Main Address`	`Position`	`Velocity`	`Heading`	`RoadID`	`Time-stamp`

4.2 Neighbor Discovery

The neighboring discovery mechanism is based on 1-hop periodic broadcast "HELLO" messages. In order to provide position information, navigation information, as well as, cross-layering information, the payload of these "HELLO" messages includes the following node's information: position, velocity, heading, road id, hello interval time, utilization, MAC layer information and number of cached packets from *carry-n-forward* mechanism. The format of this message is presented in Figure 4. The *serialize* and *deserialize* of data in NS-3 allows only unsigned integer values and since we need decimal values (double or float) for time, position, velocity, and heading, proper care should be taken in the handling of these values. Time is encoded using *mantissa* representation because of the increased accuracy needed. However, for position and velocity the x-y co-ordinates are integers interpreted as millimeters and millimeters per second, respectively; the heading is also an integer interpreted as milli-degrees. The utilization of a node is calculated as the total number of packets in a node's MAC queue. The MAC_info is the MAC Frame Error Rate that is provided by the `ns3::WifiRemoteStationInfo` class. Finally, the CnF_info is the number of the cached packets from the *carry-n-forward* mechanism that wait in the queue.

4.3 Forwarding in CLWPR

Forwarding in CLWPR is based on minimum weight. The routing table of each node is populated by entries as presented in Table 3 (e.g. a node with two neighbors and two destinations). A node calculates the distance and weight from the entries it has in its Neighbor Set, for each entry in a node's Position Association Set. Therefore, if a node does not have any entry in Position Association set, it does not have to calculate a routing table which reduces computations. Furthermore, each time one of the two sets is

Figure 4: "HELLO" message format for CLWPR

```
0 bit                  15                   31
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                 |
|                                                 |
|        ORIGINATOR POSITION (X,Y)                |
|                                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                 |
|        ORIGINATOR VELOCITY (X,Y)                |
|                                                 |
|                                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           ORIGINATOR HEADING                    |
|                                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Orig.RoadID |  Htime   |       Utilization       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Utilization    |         MAC INFO        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         MAC INFO      |    Carry-n-Forward      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Table 3: Routing table entry in CLWPR

Destination Address	Next Hop Address	Interface	Weight
Destination1	Neighbor1	if1	w11
Destination1	Neighbor2	if1	w12
Destination2	Neighbor1	if1	w21
Destination2	Neighbor2	if1	w22

changed, the routing table is recalculated so as to check for new routes the queued packets from the *CnF* mechanism. The implementation of the table is a `std::multimap` since we have multiple entries with the same key (Destination Address). The weight is calculated using (1) which is described in [9].

$$\begin{aligned}\textbf{Weight} = \boldsymbol{f_1} * Distance + \boldsymbol{f_2} * NormAngle + \\ \boldsymbol{f_3} * NormRoad + \boldsymbol{f_4} * Utilization + \\ \boldsymbol{f_5} * MAC_{info} + \boldsymbol{f_6} * CnF_{info} + \\ \boldsymbol{f_7} * SNIR_{info}\end{aligned} \tag{1}$$

4.3.1 RouteOutput

`RouteOutput` flow chart is presented in Figure 5. When it is called, information regarding the source and destination of the packet is extracted from the IP header. The destination of the packet is looked-up by the Location Service (LS) and the Position Association Set (PAS) is updated. If the destination's position is already known, the packet can progress to the selection of the next hop, otherwise it has to wait for the reply from the LS. However, since at the moment there is no specific LS implemented, we rely only on NS-3 global view of the nodes. Then, since the PAS is updated, the routing table has to be re-calculated. After that, the destination is looked-up in the routing table and the entry with the minimum weight is extracted. If that entry is the current node, then, we are in *local-maximum* state and the *carry-n-forward* mechanism is employed. Since the output of `RouteOutput` is a "route" towards a node, the *carry-n-forward* mechanism has to create a virtual route towards the same node. This is achieved with the help of a `DeferredRouteOutput Tag` added to the packet that creates a "route" for local delivery. Else, a route for the selected entry is created. The `ns3::Tag` class adds meta-data information to a packet and can be extracted from the same node at different stages. These meta-data can not be transmitted over the network so they are not to be used to carry data.

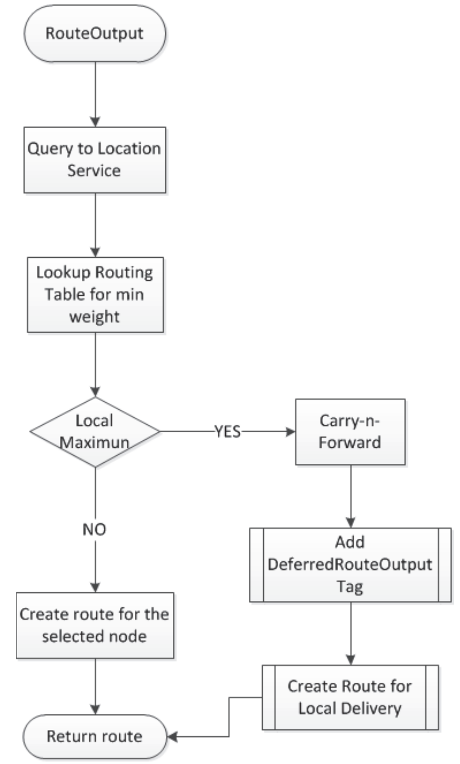


Figure 5: Flow chart for CLWPR `RoutOutput` method

4.3.2 RouteInput

`RouteInput` flow chart is presented in Figure 6. When a packet is received from the `NetDevice` it is firstly checked to see if it has the `DeferredRouteOutput Tag` which denotes that the *carry-n-forward* mechanism is employed. Then, an entry at the queue is created and the packet is stored. If there is no tag in the packet and is for local delivery, the LocalDelivery Callback is called and the packet is passed to higher layers. If the packet is not for local delivery, then, it has to be forwarded. Similar procedure for `RouteOutput` is performed and the UnicastForward Callback is called.

4.4 Enhancements

Apart from the basic position information that is used by the greedy forwarding, we provide several enhancements: navigation information, cross-layering information, *carry-n-forward* mechanism.

4.4.1 Navigation Information

Using the position, velocity, and heading we are able to predict the position of a neighbor node (and the destination node) because their mobility is restricted on the road topology; therefore we reduce the frequency of "HELLO"

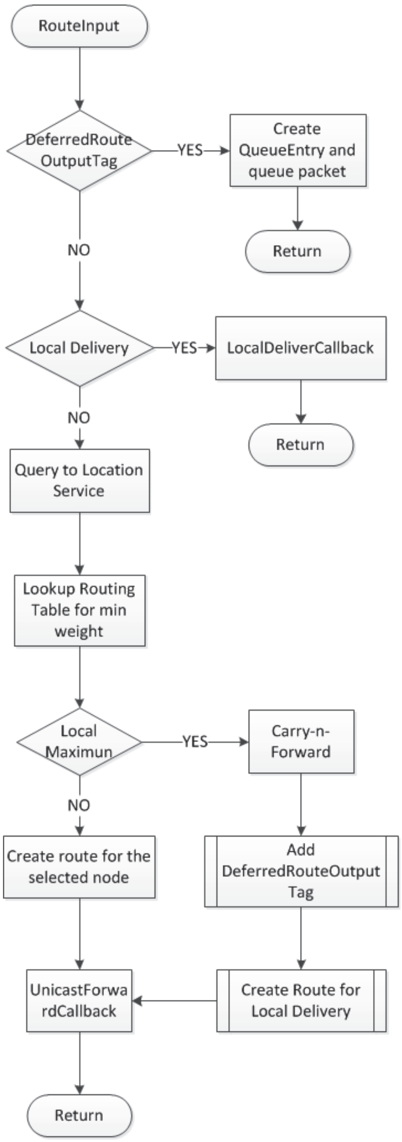


Figure 6: Flow chart for CLWPR `RoutInput` method

messages. The effect of the frequency was studied in [9] and it is shown that even low frequency (>2sec), CLWPR can provide high packet delivery ratio when prediction is used. Apart from the prediction, we utilize navigation information. Nodes are equipped with digital maps of the road topology and are able to find the road they are travelling (Orig. RoadID in "HELLO" message). A special class (`ns3::GridMap`) has been developed to provide information related to the navigation. The information that is available from this class is the road ID and the calculation of "curvemetic" distance, which is the distance between two nodes following the underlying road topology. This class currently supports only grid topologies with a maximum dimension of 15 roads. This is due to the fact that we have assigned only 1 byte for the road ID, where its first 4 bits denote the x-axis segment and 4 last digits the y-axis segment of the grid topology.

4.4.2 *Cross Layering*

As we presented in section 2, research has been focused on cross-layer optimizations to increase reliability of the communication links. To that end, we have used two key indicators of link quality. First, the SINR value of a received packet (currently using only "HELLO" messages for consistency) and second the Frame Error Rate calculated in MAC layer. Furthermore, in order to balance the network load across different nodes, we also use the MAC queue size of a node as an indicator of utilization.

SINR is available from `ns3::WiFiPhy` and specifically the `ns3::YansWifiPhy` which we used. When a packet is received (denoted by the method `YansWifiPhy::EndReceive()` the signal and noise (plus interference) values are calculated. In order to provide this information to higher layers (routing) we developed a tag (`struct SnrTag : public Tag`) and add it to each received packets. This tag is then read at the routing protocol when a "HELLO" message is received and the value is stored along with the rest information at the neighboring set. Regarding MAC information, we used the Frame Error Rate which is available from the `ns3::WifiRemoteStationInfo` class. Therefore, when we prepare a "HELLO" message, we access the `ns3::WifiRemoteStationManager` of that node which provides the station info. Furthermore, accessing the MAC route (`ns3::WifiMacQueue`) provides us with the queue size which again is inserted in the "HELLO" message as an indicator of the utilization of that node.

4.4.3 *Carry-n-Forward*

The final enhancement is the implementation of the *carry-n-forward (CnF)* mechanism. When a node realizes that the packet is faced with *local-maximum* it employs the *CnF* mechanism to cache locally the packet until a more suitable route is found. The cache queue has limited size (number of packets it stores) and expire time for the cached packets, both set as attributes to the routing protocol. The implementation *CnF* is based on the `ns3::aodv::RequestQueue`, which is used when a packet is queue temporarily waiting for the route reply. Each time the routing table is updated, the cache queue is searched in order to check if a better node is found for forwarding, thus dequeuing the packet.

5. HOW TO USE CLWPR

CLWPR comes with a helper class (`ClwprHelper`) that can be used in order to install the routing protocol to the nodes, similar to the implementation of the other IPv4 Routing protocols. Moreover, CLWPR has several attributes that can be used for the configuration and optimization. A complete list of the attributes is presented in Table 4.

5.1 Current Limitations

One limitation presented in section 4.4.1 is that CLWPR can only be used in grid topologies with maximum 15 roads per dimension. This is because of `ns3:GridMap` limitations. A more enhanced navigation system that can provide road information and curvemetric distance between two nodes would solve it. Furthermore, CLWPR does not provide interfaces with other networks. In the future, a functionality similar to OLSR's HNA could increase the capabilities of the protocol.

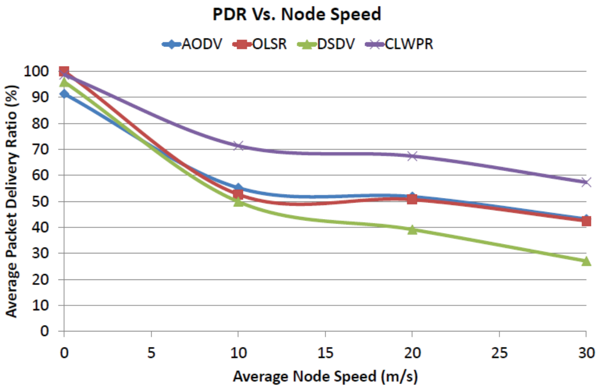
5.2 Example using CLWPR

In this subsection, we present small code snippets of how to use and configure CLWPR in a simple wifi scenario. In addition, we compare the performance of CLWPR with AODV, DSDV and OLSR in terms of packet delivery ratio, average end-to-end delay and elapsed simulation time in a vehicular environment, where 100 nodes travel for 50sec on a 5x5 grid topology with blocks of 500m. In the simulation, there are 10 simultaneous, randomly selected, UDP connections between nodes. The results presented in Fig. 7 show that the proposed protocol can provide higher packet delivery ratio and lower elapsed simulation times from all other protocols. End-to-End delay is higher than OLSR due to the caching mechanism but lower than AODV and DSDV.

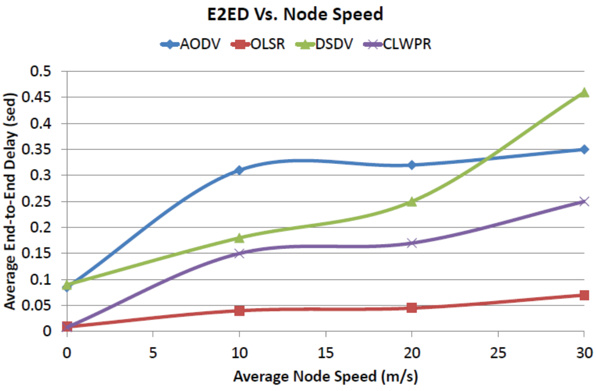
```
// Add header files for CLWPR and mobility
// modules needed for routing and map info.
#include "ns3/clwpr-module.h"
#include ""ns3/mobility-module.h"
..

// Use the command line attributes for configuration
cmd.AddValue ("map", "Enable the MAP integration", map);
cmd.AddValue ("emap", "Enable navigation", enchance);
cmd.AddValue ("predict", "Enable prediction", predict);
...

// Create a 5X5 GridMap with 500m distance
// which is the navigation system
GridMap gridMap = GridMap(5, 5, 500);

// Should install mobility and wifi before setting up
// CLWPR since it requires them.

// Create CLWPR from helper
ClwprHelper clwpr;

// Configure it
clwpr.Set("MapFlag", BooleanValue(map));
clwpr.Set("EMapFlag", BooleanValue(enchance));
clwpr.Set("PredictFlag", BooleanValue (predict));
clwpr.Set("RoadMap", PointerValue(&gridMap));
...

// Install it nodes
Ipv4ListRoutingHelper list;
list.Add (clwpr, 10);
InternetStackHelper internet;
internet.SetRoutingHelper (list);
...

// Run Simulation and gather results
...
```

Table 4: CLWPR attribute list

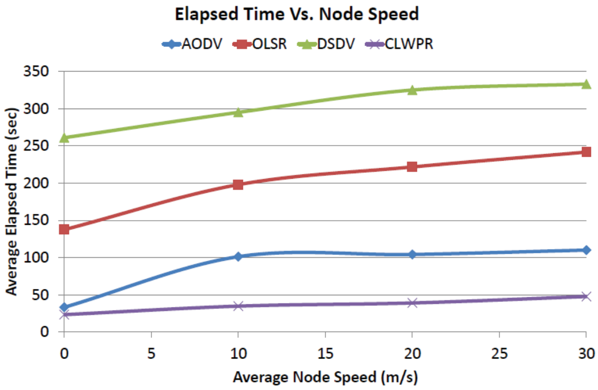
Name	Description	Type (Default value)
`HelloInterval`	`HELLO messages emission interval`	`TimeValue (1.5sec)`
`HnaInterval`	`HNA messages emission interval.`	`TimeValue (5sec)`
`MapFlag`	`Activate Map Integration (curvemetric distance)`	`BooleanValue (False)`
`EMapFlag`	`Enchanted Map Integration (navigation)`	`BooleanValue (False)`
`PredictFlag`	`Activate Position Prediction`	`BooleanValue (False)`
`CacheFlag`	`Activate Carry’n’Forward`	`BooleanValue (False)`
`MaxQueueLen`	`Maximum number of packets that we allow a routing protocol to buffer`	`UintegerValue (50)`
`MaxQueueTime`	`Maximum time packets can be queued (in seconds)`	`TimeValue (10sec)`
`DistFact`	`Distance Weighting Factor`	`DoubleValue (1)`
`AngFact`	`Angle Weighting Factor`	`DoubleValue (1)`
`RoadFact`	`Road Relevance Weighting Factor`	`DoubleValue (1)`
`UtilFact`	`Utilization Weighting Factor`	`DoubleValue (1)`
`MACFact`	`MAC Frame Error Rate Weighting Factor`	`DoubleValue (1)`
`CnFFact`	`Carry-N-Forward Weighting Factor`	`DoubleValue (1)`
`SNRFact`	`SNR Weighting Factor`	`DoubleValue (1)`
`RoadMap`	`Pointer to a CLWPR map used for navigation`	`PointerValue (0)`



(a) Packet Delivery Ratio



(b) End-to-End delay



(c) Elapsed Simulation Time

Figure 7: Performance comparison of CLWPR with AODV, OLSR and DSDV

6. CONCLUSIONS

In this paper, we presented an implementation of a novel position-based routing protocol optimized for urban vehicular environments. This implementation is highly configurable which makes it desirable for optimization. In addition, initial performance results suggest that the proposed protocol shows better performance compared to protocols already implemented for NS-3 in a VANET scenario. Since research on VANETs is an active topic currently, and there is no other position-based routing protocol module available for ad-hoc networks implemented with NS-3, we hope that our contribution can help with that.

7. ACKNOWLEDGMENTS

The work was done within the joint research project DRIVE-C2X, which is funded by the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n^{o} 270410.

8. REFERENCES

[1] Greedy Perimeter Stateless Routing (GPSR) source code. Available online at: `http://www.icir.org/bkarp/gpsr/gpsr.html`.

[2] iTETRIS FP7 Project. Available online at: `http://ict-itetris.eu/`.

[3] ns-3 manual, 2011. Available online at: `http://www.nsnam.org/docs/release/manual/html/index.html`.

[4] D. Borsetti and J. Gozalvez. Infrastructure-assisted geo-routing for cooperative vehicular networks. In *Vehicular Networking Conference (VNC), 2010 IEEE*, pages 255–262, 2010.

[5] T. Clausen and P. Jacquet. RFC 3626: Optimized Link State Routing Protocol (OLSR). Technical report, 2003.

[6] B. Jarupan and E. Ekici. PROMPT: A cross-layer position-based communication protocol for delay-aware vehicular access networks. *Ad Hoc Networks*, 8:489–505, 2010.

[7] M. Jerbi, S.-M. Senouci, T. Rasheed, and Y. Ghamri-Doudane. Towards Efficient Geographic Routing in Urban Vehicular Networks. *IEEE Transactions on Vehicular Technology*, 58:5048–5059, 2009.

[8] B. Karp and H. T. Kung. GPSR : Greedy Perimeter Stateless Routing for Wireless Networks. In *International conference on Mobile computing and networking*, pages 243–254, 2000.

[9] K. Katsaros, M. Dianati, R. Tafazolli, and R. Kernchen. CLWPR - A Novel Cross-Layer Optimized Position Based Routing Protocol for VANETs. In *IEEE Vehicular Networking Conference*, pages 200–207, 2011.

[10] V. Naumov, R. Baumann, and T. Gross. An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 108–119, 2006.

[11] V. Naumov and T. R. Gross. Connectivity-Aware Routing (CAR) in Vehicular Ad-hoc Networks. In *IEEE International Conference on Computer Communications*, pages 1919–1927, 2007.

[12] A. Paul, S. Konwar, U. Gogoi, A. Chakraborty, N. Yeshmin, and S. Nandi. Implementation and performance evaluation of AODV in Wireless Mesh Networks using NS-3. In *International Conference on Education Technology and Computer (ICETC)*, pages 298–303, 2010.

[13] C. E. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance-Vector routing

(DSDV) for mobile computers. *SIGCOMM Comput. Commun. Rev.*, 24:234–244, 1994.

[14] B. Schuenemann. V2X simulation runtime infrastructure VSimRTI: An assessment tool to design smart traffic management systems. *Computer Networks*, 2011.

[15] N. Sofra, A. Gkelias, and K. K. Leung. Link Residual-Time Estimation for VANET Cross-Layer Design. In *IEEE International Workshop on Cross-Layer Design (IWCLD)*, pages 9–13, 2009.

[16] V. Timcenko, M. Stojanovic, and S. B. Rakas. MANET routing protocols vs. mobility models: performance analysis and comparison. In *International Conference on Applied Informatics and Communications*, pages 271–276, 2009.

[17] E. Weingartner, H. vom Lehn, and K. Wehrle. A performance comparison of recent network simulators. In *IEEE International Conference on Communications*, pages 1–5, 2009.

[18] J. Zhao and G. Cao. VADD : Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 57(3):1910–1922, 2008.