VSHDRP: Vehicle Second Heading Direction Routing Protocol in VANET

Moath Muayad Al-Doori, Francois Siewe, and Ali Hilal Al-Bayatti

Abstract—**Routing protocols in VANET are considered as one of the critical dilemmas that need to be tackled, especially in sparse environment. Thus designing an efficient routing mechanism has an impact on enhancing the network performance in terms of disseminating messages to their desired destinations. This paper proposes a novel routing protocol in VANET for sparse environment called Vehicle Second Heading Direction Routing Protocol (VSHDRP), which is designed to leverage the probability of delivering a data packet to its destination and to increase connectivity and route stability by utilizing the knowledge of the Second Heading Direction (SHD) in the process of selecting the next-hop node. To the best of our knowledge this is the first paper that takes the SHD into account to improve the routing mechanism in VANET. Moreover, the VSHDRP protocol is formalised in the Calculus of Context-aware Ambients (CCA) and simulated using the CCA interpreter in order to analyse the behaviour of the protocol.**

Index Terms—**VANET, DTN, SHD, VSHDRP, CCA**

I. INTRODUCTION

Highlight a Vehicle Ad hoc Networks (VANET) is a specific state of Mobile Ad hoc Networks (MANET) [1]. Communication in VANET can be accomplished via one of two main alternatives: Vehicle to Infrastructures (V2I) and Vehicle to Vehicle (V2V), which use a Dedicated Short Range Communication (DSRC) method between either nearby vehicle or roadside equipment facilities. The DSRC is based on IEEE 1609 standards of the Wireless Access in Vehicular Environments (WAVE) family. Traffic congestion caused by vehicles accidents is considered to be a vital concern on the roads. Therefore safety applications are the focus of most researchers in VANET; increasing the efficiency of these applications has a vital impact on their contribution to limiting the number of fatal accidents and providing safer, cleaner and more comfortable travelling on roads. Vehicle drivers have no ability to predict the conditions ahead of time [2]; with the aid of sensors, ubiquitous computing and wireless communication devices, this combination of equipped devices assists in providing more capabilities to vehicles on the roads to foresee hazard (e.g. the speed of other vehicles). In that way, warning messages could be sent every 0.5 second to predict vehicle speed in order to eliminate the occurrence of accidents [3].

Numerous studies in the last few years have been investigating (VANET). In particular, owing to the impact of routing issues in improving the efficiency of network's performance,

Many routing techniques have been designed in MANET to tackle the limitations of the transmission packet delivery delay, packets being dropped, wasting bandwidth, mobility and security. These techniques could not be fitted on VANET owing to the particular characteristics of VANET, (e.g. restricted mobility pattern). As a result of the above-mentioned argument, it is anticipated that designing an efficient routing protocol will aid in accomplishing the task of delivering packets to their destinations in a more realistic method.

This paper proposes a novel mechanism in VANET to mitigate these limitations, by using the *Second Heading Direction* (SHD) besides other parameters already used in other proposed routing mechanisms such as node's position, current direction and speed. Indeed, the SHD provides information about whether a node will change its direction ahead, and this knowledge is utilised to select the best next-hop node with respect to network connectivity and route stability. The contributions of this paper are summarized as follows:

- We propose a novel notion of SHD in routing protocol in VANET to increase the packet delivery ratio, route stability and connectivity (Sect. IV-B).
- We propose a novel routing algorithm, VSHDRP, based on the filtration process, which comprises four stages: position, direction, SHD and speed (Sect. IV-C).
- The new protocol is fully formalised in the Calculus of Context-aware Ambients (CCA) (Sect. V).
- We analyse the behaviours of the protocol using the CCA interpreter (Sect.VI).

II. RELATED WORK

Many researchers have been interested in VANET, especially in designing a suitable routing protocol [4] to overcome the problems of high node mobility and restricted movement (e.g. connectivity, latency and unnecessary overhead) [5]. Zhao and Cao [6] introduced the Vehicle-Assisted Data

Delivery Routing Protocol (VADD), which tackle the problems of packet delivery ratio, data packet latency and overhead. In addition, a Connectivity-Aware Routing protocol (CAR) [7] has been initiated by Naumov and Gross to support the vehicular network in city and highway environments.

Motion Vector Routing Algorithm [8] introduced by Lebrun and Chuah has been designed to deliver a message from a vehicle to a static destination in a sparse environment;

Manuscript received June 21, 2012; revised August 6, 2012.

M. M Al-Doori is with the STRL, De Montfort University, Leicester, UK (e-mail: maldoori@dmu.ac.uk).

F. Siewe and A. H. Al-Bayatti are with the Software Technology Research Laboratory (STRL) at Faculty of Technology, De Montfort University, Leicester, UK (e-mail: fsiewe@dmu.ac.uk, alihmohd@dmu.ac.uk)

it focuses on predicting which vehicle from its neighborhood will travel towards the fixed destination by utilizing data from the knowledge of its neighboring vehicles, such as velocity and trajectory.

III. THE CALCULUS OF CONTEXT-AWARE AMBIENTS

This section presents the syntax and the informal semantics of CCA. Due to the space limit, only features relevant to our work are presented. We refer interested readers to [9] for the full details of the calculus. Table I depicts the syntax of CCA, based on three syntactic categories: processes (denoted by P or Q), capabilities (denoted by *M*) and context-expressions (denoted by *E*). The simplest entities of the calculus are names. We let ỹ denote a list of names and | ỹ |the size of such a list.

TABLE I: SYNTAX OF CCA

$$\begin{array}{lcl}
P, Q & ::= & 0 \mid P|Q \mid (\nu\ n)\ P \mid !\,P \mid n[P] \mid \{P\} \mid E?\,M.P \ \mid \text{find } \tilde{x} : E \text{ for } P \\
M & ::= & \text{in } n \mid \text{out} \mid \alpha \text{ recv } (\tilde{y}) \mid \alpha \text{ send}(\tilde{y}) \ \mid \text{del } n \\
\alpha & ::= & \uparrow \mid n \uparrow \mid \downarrow \mid n \downarrow \mid \ :: \ \mid \ n :: \ \mid \epsilon \\
E & ::= & \text{True} \mid \bullet \mid n = m \mid \neg E \mid E_1 \mid E_2 \mid E_1 \wedge E_2 \mid \oplus E \mid \diamond E
\end{array}$$

Processes. The process 0, aka inactivity *process*, does nothing and terminates immediately. The process $P|Q$ denotes the process P and the process Q running in parallel. The process $(\nu\ n)\ P$ states that the scope of the name n is limited to the process P. The replication $!P$ denotes a process which can always create a new copy of P, i.e. $!P$ is equivalent to $P|!P$. Replication can be used to implement both iteration and recursion. The process $n[P]$ denotes an ambient named n whose behaviours are described by the process P. The pair of square brackets '[' and ']'outlines the boundary of that ambient. The process $\{P\}$ behaves exactly like the process P, so the pair of curly brackets '{' and '}' are used simply as parentheses.

A context expression E specifies the condition that must be met by the environment of the executing process. A *context-guarded prefix E?M.P* is a process that waits until the environment satisfies the context expression E, then performs the capability *M* and continues like the process *P*. The dot symbol '.' denotes the sequential composition of processes. We let *M.P* denote the process True*?M.P*, where True is a context expression satisfied by all context. A *search prefix* $find\ \tilde{x} : E\ for\ P$ is a process that looks for a set a names $\tilde{n}$ such that the context expression $E\{\tilde{x} \leftarrow \tilde{n}\}$ holds and continues like the process $P\{\tilde{x} \leftarrow \tilde{n}\}$, where the notation $\{\tilde{x} \leftarrow \tilde{n}\}$ means the substitution of n_i for each free occurrence of x_i, $0 \leq i < |\tilde{x}|$.

Capabilities. Ambients exchange messages using the output capability α send$(\tilde{z})$ to send a list of names $\tilde{z}$ to a location α, and the input capability α recv $(\tilde{y})$ to receive a list of names from a location α. The location α can be '↑' for any parent, '$n\uparrow$' for a specific parent *n*, '↓'for any child, '$n\downarrow$' for a specific child n, '::' for any sibling, 'n ::' for a specific sibling *n*, or ∈ (empty string) for the executing ambient itself.

The mobility capabilities in and out are defined as follows. An ambient that performs the capability in *n* moves into the sibling ambient *n*. The capability out moves the ambient that performs it out of that ambient's parent.

Example 3.1: The process following describes the behaviours of two sibling ambients *n* and *m* concurrently willing to move in and out of one another:

n[in *m*.out.0] | *m*[in *n*.out.0]

Context expressions: In CCA, a context of process is modelled as a process with a hole in it. The hole (denoted by ⊙) in a context represents the position of the process that context is the context of. For example, suppose a system is modelled by the process $P \mid n[Q \mid m[R \mid S]]$. So, the context of the process R in that system is $P \mid n[Q \mid m[\odot \mid S]]$, and that of the ambient named m is $P \mid n[Q \mid \odot]$. Properties of contexts are called context expressions (CEs in short).

The CE true always holds. A CE $n = m$ holds if the names *n* and *m* are lexically identical. The CE • holds solely for the hole context, i.e. the position of the process evaluating that context expression. Propositional operators such as negation (¬) and conjunction (^) expand their usual semantics to context expressions. A CE $E_1|E_2$ holds for a context if that context is a parallel composition of two contexts such that E_1 holds for one and E_2 holds for the other. A CE *n*[*E*] holds for a context if that context is an ambient named n such that *E* holds inside that ambient. A CE ⊗E holds for a context if that context has a child context for which *E* holds. A *CE ✧E* holds for a context if there exists somewhere in that context a sub-context for which *E* holds.

Example 3.2: The following CE *has*(*n*) holds if the executing ambient contains an ambient named n[1]: has (*n*) $\widehat{=}$ $\oplus\ (\bullet \mid n[\text{True}] \mid \text{True})$

The symbol '$\widehat{=}$' means defined by.

IV. THE VSHDRP MODEL

This section shows the assumption, overview and the mechanism of the VSHDRP:

A. Assumption

VSHDRP works under the following assumptions: the transmission range of each vehicle in the network is up to 300m, and each vehicle has sufficient knowledge about its surrounding neighbours through exchanging a HELLO beacon message periodically, i.e. vehicle *id*, its position, direction and speed. We assume in this proposed protocol that each vehicle is supplied with a *GPS* device and navigation system (NS), and vehicles are equipped with preloaded digital road maps; therefore we assume that each vehicle can know its own location, direction through the fitted *GPS* device and *NS*, and can predetermine its route to its destination from the beginning.

B. VSHDRP Overview

The approach proposed in this paper takes the advantage of utilising the *GPS* system, *NS* and digital road map; in this way each vehicle can acquire the knowledge of its position and direction, and can establish a predetermined route; at the same time each vehicle can provide its surrounding neighbours with all this information through broadcasting a periodic HELLO beacon message. The new proposed protocol aims to initiate a robust and long-life route between the source node of the data packet and its destination.

The main idea of the VSHDRP is to take the (*SHD*) into account in the process of selecting the next-hop node, which has an impact on making the packet route more stable.

For example, if we assume that node *S* is travelling on a highway and is intending to deliver a packet to a destination node *D*, located at the end of the road, according to this assumption the forwarding vehicle (which holds the packet) can select the next-hop vehicle (node) based on its *SHD*; the next-hop node has an *SHD* that leads the vehicle to next exit (if we know that the route packet will continue ahead). In this situation, the forwarding vehicle will ignore this vehicle and will look for another vehicle that has an *SHD*, which will not drive the vehicle to the next exit. In other words, the forwarding vehicle will select the next-hop vehicle that continues on the same road of the packet route without making any diversion in its route; thus the probability of delivering messages to its destination will increase.

This proposed work mitigates the Delay Tolerant Network (*DTN*) issues; it is based mainly on the carry-and forward strategy. When a disconnected area appears to split between the vehicle that holds the packet and other vehicles moving toward the packet's destination, there will be no opportunity to forward the packet to a next-hop node. Having information about the position of the source node and destination is insufficient.

Therefore, using the *SHD* is vital in deciding the next hop node. Suppose a forwarding node in a *DTN* Network needs to forward the packet to the next-hop node which has the position and direction towards the destination region, without knowing the *SHD* of the next-hop. This next-hop node might take another route. If it has a *SHD* that leads it to another route before it reaches its destination. The *SHD* can take two values, either *0* or *1*; $SHD = 0$ means that the vehicle will not divert its route in the next exit or intersection, which makes it a candidate node, while $SHD = 1$ represents the condition that the vehicle will divert its route in the next exit or intersection.

In this situation, this node is not suitable for delivering the packet; as a result, the forwarding node will need to re-forward the packet, and that will lead to a reduction in the bandwidth consumption, delay in the packet transmission time and the system stability and connectivity.

C. VSHDRP Mechanism

The VSHDRP comprises three parts: the first one represents the general algorithm of the process of sending the packet, while the second part depicts the core of the VSHDRP, which is the filtration process of selecting the next-hop node; the third part represents the process of packet delivery confirmation between any two nodes during the sending process.

1) Packet Sending Process: VSHDRP considers that each vehicle (node) in the network has sufficient knowledge about its own location, direction, speed and *SHD*. As illustrated in Figure 1, when a source node \$S\$ needs to send a packet to a destination node *D*, it will look for node *D* in its cache (neighbours table), and if the node *D* is found as a neighbour in its cache, node *S* will start forwarding the data packets to node *D*. If *D* is not found in the cache of source node *S*, then node *S* will set the current direction and the *SHD*, and then it will start to look for an appropriate next-hop node by using the filtration process for selecting the next-hop node, as illustrated in Figure 2, which comprises four main stages: position stage, current direction stage, *SHD* stage and speed stage respectively. The output of the filtration process can be either *Yes* or *No*.
   a. *Yes:* means an appropriate next-hop node is found from the neighbour nodes. Therefore the source node will forward the packet to this node.
   b. *No:* means that no appropriate next-hop node is found *from* the surrounding neighbour nodes. For that reason, the source node will keep the packet in its buffer and continue listening for new neighbours to become available. If a new neighbour node is found, the source node will run the same procedure until it delivers the packet to its destination.

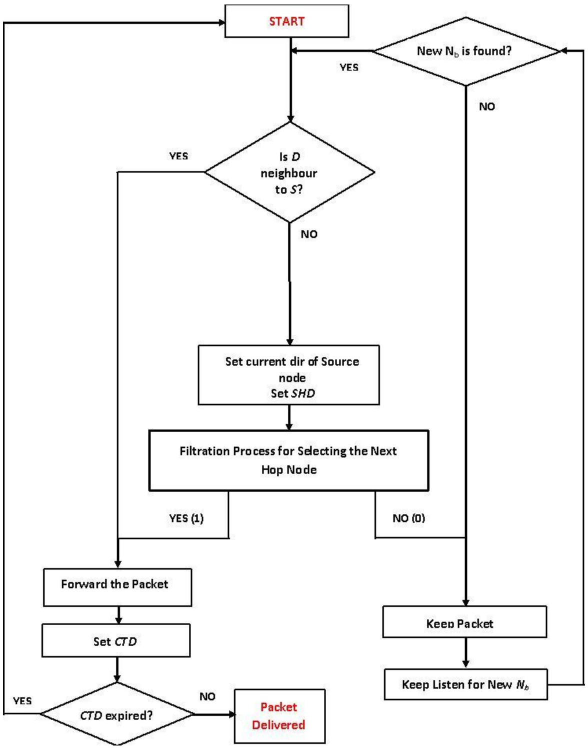

Fig. 1. The algorithm of VSHDRP.

2) Filtration process for selecting the next hop node: these stages are as follows:
   a. *Position Knowledge Stage:* in this stage, the *packet's* holder node will select neighbour nodes that have a closer position to its destination than itself.
   b. *Current Direction Knowledge Stage:* in this stage, the selected nodes in the previous stage will be processed by the operation of this stage, in order to check if they have an appropriate current direction, (the direction towards the packet's destination).
   c. *Second Heading Direction Knowledge Stage (SHD): this* stage nominates the candidates' nodes in the previous stage according to their SHD.
   d. *Speed knowledge stage:* this stage will select the node with the highest speed in case more than one candidate is available.

Fig. 2. Filtration process algorithm for selecting next-hop node.

As shown in Figure 2, the filtration process states that, if the destination node *D* is not found in source node's cache, then the source node *S* will start the filtration process to look for an appropriate next-hop node; *S* will start the filtration process by looking for neighbours with a closer destination to *D* than itself. If no neighbours' nodes (*Nb*) are found, it will buffer the packet; otherwise it will check if any of *Nb* nodes hve a current direction towards the packet's destination. If an appropriate node is found then it will check to see if its *SHD = 0*; if more than one *Nb* node is found, then the *Nb* with highest speed will be selected. If *SHD = 1* for all *Nb* , then *S* will check *SHD = 0* for *Nb* of *Nb* and if more than one *Nb* of *Nb* is found, then the *Nb* of *Nb* with the highest speed will be selected. If the current direction for all *Nb* is opposite to the packet's destination, then *S* will check the *Nb* of *Nb* that have current direction towards the packet's destination, and repeat the same procedure for *SHD* and speed.

3) Packet Delivery Confirmation: When the forwarding node intends to forward the packet to another intermediate node, it needs to make sure that the packet has been delivered successfully to that node; therefore it is important in this stage to send back an acknowledgment to the sending process. Thus a time counter needs to be established at the same moment of sending the packet; this counter is called Confirmation Time Duration (*CTD*). When the forwarding node wants to send the packet, it sets the *CTD* to a specified threshold value, and the value of this counter will be decremented; if the confirmation is received before the *CTD* has elapsed, then the *CTD* will be halted. Otherwise, if the time is elapsed and no confirmation message (*acknowledgment*) has been received, that means the packet has been dropped or discarded for some reason; therefore the sender node will look for an alternative node and resend the packet.

V. FORMAL SPECIFICATION OF VSHDRP

We now give a formal specification of the VSHDRP protocol in CCA in a compositional manner. Individual nodes in a VANET are specified independently and then composed in parallel to form the whole system.

A. System Model

VANET is modelled in CCA as a parallel composition of all the nodes in the network (e.g. vehicles and road side units), i.e

$$VANET \ \widehat{=}\ node_0 \mid node_1 \mid \ldots \mid node_{k-1} \qquad (1)$$

Each node, *node$_i$*, in the VANET is modelled as an ambient of the following structure:

$$id[P_{id} \mid NH1\,[P_1] \mid NH2\,[P_2] \mid Hspeed\,[P_3] \mid Lspeed\,[P_4] \mid SHD\,[P_5]\ ]$$

where

- id is the node's id. For the sake of simplicity, we use *SN* to denote the source node, *DN* for the destination node and IN_j for intermediate nodes, $j \geq 0$.
- P_{id} is a process that specifies the capabilities of the node, e.g. its ability to communicate or to sense the presence of other nodes in its range.
- *NH1* is an ambient that contains the ids of the neighbouring node moving in the direction of the destination node (aka direction 1) and closer to the destination node.
- *NH2* is an ambient that contains the ids of the neighbouring node moving in the direction opposite to direction 1 (this is called direction 2) and closer to the destination node.
- *Hspeed* is an ambient that contains the ids of the neighbouring nodes moving in high speed (i.e. speed greater than or equal to a specified threshold).
- *Lspeed* is an ambient that contains the ids of the neighbouring nodes moving in low speed (i.e. speed less than the threshold).
- *SHD* is an ambient that contains the ids of the neighbouring nodes with Second Heading Direction (SHD).
- Each processP_j, $1 \leq\ \ j\ \leq\ 5$, is either the inactivity process **0** or a parallel composition of ambients of the form *n*[**0**] where *n* is a node's id.

B. Context Expression

As explained in Section II, a context expression is a predicate that states the condition the environment of the executing process must meet.

The context expressions used in the specification of the VSHDRP protocol are summarised as follows:

- *hasNb1(n)* holds if the node *n* is a neighbour closer to destination in direction 1, i.e.
  $hasNb1(n)\ \widehat{=}\ ✧\,(\bullet \mid NH1[\text{True} \mid n[\text{True}]] \mid \text{True})$
- *hasNb2(n)* holds if the node *n* is a neighbour closer to destination in direction 2, i.e.
  $hasNb2(n)\ \widehat{=}\ ✧\,(\bullet \mid NH2[\text{True} \mid n[\text{True}]] \mid \text{True})$
- *hasNb(n)* holds if the node *n* is a neighbour closer to destination (regardless its direction), i.e.
  $hasNb(n)\ \widehat{=}\ hasNb1(n) \vee\ hasNb2(n)$
- *noNb1* (n) holds if there are no neighbours in direction 1, i.e.

$$noNb1(n) \widehat{=} \neg\text{✧}\,(\bullet \mid NH1[\mathrm{True} \mid \oplus\,\mathrm{True}] \mid \mathrm{True})$$

- *hasSHD(n)* holds if the node *n* is a neighbour with second heading direction (SHD), i.e.
  $$hasSHD(n) \widehat{=} \text{✧}\,(\bullet \mid SHD[\mathrm{True} \mid n[\mathrm{True}]] \mid \mathrm{True})$$
- *noSHD( )* holds if there are no neighbours with SHD
  $$noSHD(n) \widehat{=} \neg\text{✧}\,(\bullet \mid SHD[\mathrm{True} \mid \oplus\,\mathrm{True}] \mid \mathrm{True})$$
- *hasNbofNb1(n)* holds if node *n* is a neighbour in direction 2 that has a neighbour closer to destination and moving in direction 1

$$hasNbofNb1(n) \widehat{=} hasNb2 \\ \wedge \text{✧}\left(\mathrm{n}[\mathrm{True} \mid \mathrm{NH1}[\mathrm{True} \mid \oplus\,\mathrm{True}]\right] \mid \mathrm{True})$$

- *highestSpeed(n)* holds if the node *n* has the highest speed:

$$highestSpeed(n) \widehat{=} \text{✧}\left(\bullet \mid Hspeed[\mathrm{True} \mid n[\mathrm{True}]\right] \vee \\ (\neg\text{✧}\,(\bullet \mid Hspeed[\mathrm{True} \mid \oplus\,\mathrm{True}] \mid \mathrm{True}) \wedge \\ \text{✧}\,(\bullet \mid Lspeed[\mathrm{True} \mid n[\mathrm{True}]] \mid \mathrm{True}))$$

We now specify the behaviours proper to each type of node (source node, intermediate node and destination node).

A source node *SN* is the node that initiates a run of the VSHDRP protocol, willing to send a message *msg* to a destination node *DN*. So its capabilities are modelled by the following process:

$$P_{SN} \widehat{=} (\nu\, n)\,\{(3) \mid (4) \mid (5) \mid x[\uparrow \mathrm{send}(y).0]\} \qquad (2)$$

where the restricted name *x* is used to guarantee that not more than one of the processes Eq3, Eq4, Eq5 are executed. Indeed, *x* is an ambient that sends a single signal to its parent; this signal will be captured by exactly one of these processes, eventually. These processes are specified as follows:

- if the destination node is a neighbour, send message to destination node and wait for acknowledgement. This is formalised as:

$$hasNb(DN)?\,x \downarrow \mathrm{recv}(y).\,DN :: \mathrm{send}(SN, msg). \\ DN :: \mathrm{recv}(y).0 \qquad (3)$$

- if the destination node is not a neighbour then look for an intermediate node moving in direction 1 with SHD and highest speed, and send the message to that intermediate node and wait for acknowledgement, viz.

$$\mathrm{find}\; n : E1(n, DN)\,\mathit{for}\; x \downarrow \mathrm{recv}(y). \\ n :: \mathrm{send}(SN, DN, msg).\,n :: \mathrm{recv}(y).0 \qquad (4)$$

where: $E1(s,t) \widehat{=} \neg hasNb(t) \wedge hasNb(s) \wedge hasSHD(s) \wedge highestSpeed\,(s)$

- if no such intermediate nodes then look for an intermediate node moving in direction 2 which has a neighbour in direction 1 closer to the destination node, and send the message to that intermediate node and wait for acknowledgement, viz.

$$\mathrm{find}\; n : E2(n, DN)\,\mathit{for}\; x \downarrow \mathrm{recv}(y). \\ n :: \mathrm{send}(SN, DN, msg).\,n :: \mathrm{recv}(y).0 \qquad (5)$$

where:

$$E2(s,t) \widehat{=} \neg hasNb(t) \wedge \left(noNb1(\,) \vee noSHD(\,)\right) \wedge \\ hasNbofNb1(s)$$

An intermediate node *IN* receives a triple (*sender*, *dest*, *msg*) where *sender* is the sender's *id*, *dest* is the destination node's id and *msg* is the message being sent. The intermediate node confirms the receipt by sending an acknowledgement to the sender and forwards the message to an appropriate node. This is specified as:

$$P_{IN} \widehat{=} \,!\, :: \mathrm{recv}(sender, dest, msg). \\ \mathrm{send} :: \mathrm{send}(ack).\,(\nu\, x)\{\,(7) \mid (8) \mid (9) \mid \\ x\,[\mathrm{send}(y).0]\} \qquad (6)$$

where the restricted name *x* plays the same role as in (2) for selecting at most one of the processes (7), (8) and (9). The replication operator '!' means that an intermediate node repeats this pattern of behaviour its whole lifetime. Moreover, an intermediate node determines the next node to forward the message to as follows:

- if the destination node *dest* is a neighbour, send message to destination node and wait for acknowledgement, i.e.

$$hasNb(dest)?\,x \downarrow \mathrm{recv}(y). \\ dest :: \mathrm{send}(IN, msg).\,dest :: \mathrm{recv}(y).0 \qquad (7)$$

- if the destination node *dest* is not a neighbour then look for another intermediate node moving in direction1 with SHD and highest speed, and send the message to that intermediate and wait for acknowledgement, viz.

$$\mathrm{find}\; n{:}\,E1(n, dest)\,\mathit{for}\; x \downarrow \mathrm{recv}(y). \\ n :: \mathrm{send}(IN, dest, msg).\,n :: \mathrm{recv}(y).0 \qquad (8)$$

- if no such intermediate nodes then look for another intermediate node moving in direction 2 which has a neighbour in direction 1 closer to the destination node, and send the message to that intermediate and wait for acknowledgement, viz.

$$\mathrm{find}\; n{:}\,E2(n, dest)\,\mathit{for}\; x \downarrow \mathrm{recv}(y). \\ n :: \mathrm{send}(IN, dest, msg).\,n :: \mathrm{recv}(y).0 \qquad (9)$$

The destination node DN receives a pair (*sender; msg*) and sends an acknowledgement to the sender. This behaviour is formalised as:

$$P_{DN} \widehat{=} :: \mathrm{recv}(sender, msg). \\ sender :: \mathrm{send}(ack).0 \qquad (10)$$

This formal specification of the VSHDRP protocol is executable by the CCA interpreter (ccaPL) and is used in the following section to simulate runs of the protocol.

VI. ANALYSIS OF THE PROTOCOL

The formal specification of the VSHDRP protocol presented above is executable by the CCA interpreter. The result of each run of the protocol is a sequence of reductions showing the interactions that happened among the

VANET’s nodes. This output can then be analysed to detect flaws in the protocol at an early stage, prior to implementation and deployment.

For illustration, we consider the following scenario where the destination node is not neighbour to the source node; rather the following neighbourhood relationship is considered: {(SN, IN1), (IN1, IN2), (IN2, IN3), (IN3, DN)}. In addition, the node IN2 moves in direction1, while IN1 and IN3 move in direction2. Each of these nodes has a second heading direction (SHD). The routing protocols mentioned earlier in (sec. II) such as VADD and CAR are suffering from the possibilities of dropping the packet, if the vehicle that hold the packet change its direction by taking the exit in the highway that leads to decrease the system connectivity, however in VSHDRP packet dropping ratio will be less, because the vehicle that will be hold the packet will be selected from the beginning, based on its SHD, which promise to keep the vehicle in the same packet’s route, that means there is no need to retransmit the packet and that will reduce the links breakage (increase stability), The output of the protocol’s run is given in Table II, which shows that the message reached to the destination successfully based on the SHD.

TABLE II: OUTPUT OF SCENARIO

```
1.←····→ {renaming of a resricted name: x to x$0}
2. −→ {binding: n −→ IN1}
3.−→ {Child to parent: x$0 == ( ) ==⇒ SN}
4. −→ {Sibling to sibling: SN == (SN, DN, hello) ===⇒ IN1}
5. −→ {Sibling to sibling: IN1 == (ack) ==⇒ SN}
6.←····→ {renaming of a resricted name: x to x$11}
7. −→ {binding: n −→ IN2}
8.−→ {Child to parent: x$11 == ( ) ==⇒ IN1}
9. −→ {Sibling to sibling: IN1== (IN1, DN, hello) ===⇒ IN2}
10. −→ {Sibling to sibling: IN2 == (ack) ==⇒ IN1}
11.←····→ {renaming of a resricted name: x to x$20}
12. −→ {binding: n −→ IN3}
13.−→ {Child to parent: x$20 == ( ) ==⇒ IN2}
14. −→ {Sibling to sibling: IN2 == (IN2, DN, hello) ===⇒ IN3}
15. −→ {Sibling to sibling: IN3 == (ack) ==⇒ IN2}
16.←····→ {renaming of a resricted name: x to x$29}
17.−→ {Child to parent: x$29 == ( ) ==⇒ IN3}
18. −→ {Sibling to sibling: IN3 == (IN3, DN, hello) ===⇒ DN}
19. −→ {Sibling to sibling: DN == (ack) ==⇒ IN3}
```

The symbol ‘−→’ denotes a system transition and the notation ‘A == X ==⇒ B’ means that an ambient ‘A’ sent a message ‘X’ to another ambient ‘B’ during the transition. In line 4 the source node SN forwards the message to the node IN1. The message then goes from IN1 to IN2 then to IN3 and finally to DN as showed in lines 9, 14 and 18 respectively.

VII. CONCLUSION

A novel routing technique in VANET has been introduced in this paper, which concentrates in leveraging the probability of delivering packets to their destination, increasing stability and connectivity, while maintaining a high advantage in safety application to reduce the risks associated with fatal accidents. We examine the behaviour of the VSHDRP protocol by modelling it in CCA; Section IV shows the output of the simulation of VSHDRP protocol using the CCA interpreter. In future, the system performance will be examined using the Network Simulator-2 (NS-2), and its performance will be compared with other routing protocols in VANET.

REFERENCES

[1] Y. Lin, Y. Chen, and S. Lee, “Routing protocols in Vehicular Ad Hoc Networks: A Survey and future perspectives,” *Journal of Information Science and Engineering*, 2010.

[2] A. Ebner and H. Rohling, “A self-organized radio network for automotive applications,” *Proceedings of the 8th World Congress on Intelligent Transportation Systems*, 2001.

[3] M. Al-Doori, A. Al-bayatti, and H. Zedan, “Context aware architecture for sending adaptive hello message in VANET,” In *Proceeding of Casemans 2010, the 4th ACM International Workshop on Context Awareness for Self-Managing Systems*, Copenhagen, Denmark, pp. 26-29 Sep, 2010.

[4] M. Zhang and R. S. Wolff, “Routing protocols for vehicular ad hoc networks in rural areas,” *IEEE Communication Magazine*, pp. 126-131, 2008.

[5] W. Chen, R. K. Guha, T. J. Kwon, J. Lee, and Y. Hsu, “A survey and challenges in routing and data dissemination in vehicular ad hoc networks,” *Wiley Inter Science*, 2009.

[6] J. Zhao and G. Cao, “VADD: Vehicle-assisted data delivery in vehicular ad hoc networks,” *IEEE Computer Communications*, 2006.

[7] V. Naumov and T. R. Gross, “Connectivity-aware routing (CAR) in vehicular ad hoc networks,” *IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1919- 1927, 2007.

[8] J. Lebrun, C. Chuah, D. Ghosal, and Michael Zhang, “Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks,” *IEEE Vehicular Tech. Conference*, 2005, 2289-2293.

[9] F. Siewe, H. Zedan, and A. Cau, “The calculus of context-aware ambients,” *Journal of Computer and System Science*, doi:10.1016/j.jcss.2010.02.003, 2010.